Universität Bielefeld
Technische Fakultät
Arbeitsgruppe Bioinformatik/Medizinische Informatik

CELLmicrocosmos

Integrative Cell Modeling at the Molecular, Mesoscopic and Functional Level

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

vorgelegt der Technischen Fakultät
der Universität Bielefeld

von: M.A., B.Sc. Björn Sommer
geb. am: 25.01.1980 in Oberhausen

Björn Sommer:

CELLmicrocosmos -
Integrative Cell Modeling at the
Molecular, Mesoscopic and Functional Level

Der Technischen Fakultät der Universität Bielefeld
im Oktober 2012 vorgelegt.
Am 14. Januar 2013 verteidigt und genehmigt.

Gutachter/Referees:
Prof. Dr. habil. Ralf Hofestädt, Universität Bielefeld
Prof. Dr. habil. Falk Schreiber, Martin Luther Universität Halle-Wittenberg
Prof. Dr. habil. Karl-Josef Dietz, Universität Bielefeld

Prüfungsausschuss/Board of Examiners:
Prof. Dr. Robert Giegerich, Universität Bielefeld
Prof. Dr. Karl-Josef Dietz, Universität Bielefeld
Prof. Dr. Falk Schreiber, Martin Luther Universität Halle-Wittenberg
Prof. Dr. Ralf Hofestädt, Universität Bielefeld
Dr. Thies Pfeiffer, Universität Bielefeld

343 Seiten/Pages
174 Abbildungen/Figures
26 Tabellen/Tables

Gedruckt auf alterungsbeständigem Papier nach DIN-ISO 9706.
Printed on non-aging paper according to DIN-ISO 9706.

Acknowledgments

Because this is an interdisciplinary work, it is naturally that I would like to thank a number of persons. Without the discussions and contributions of these people, this work would be only a sketch of the vision developed throughout this thesis.

First of all, I want to thank my supervisor Ralf Hofestädt, who gave me on the one hand the freedom and support to develop this project over the last years, and who on the other hand knew how to focus my attention on the important topics. He gave me the opportunity to participate in many international conferences and workshops, he helped me to develop new teaching approaches and I learned a lot during this time.

Second, I want to thank Klaus-Ove Kahrmann who in the beginning of this thesis was my second supervisor. During the time, the topic of this thesis changed the direction and was heading towards Bioinformatics, therefore it was decided to choose other reviewers. Anyway, during my time as a master student I learned a lot from him and in the seminars we held together were very important for my future teaching.

Falk Schreiber, who is now the second supervisor, gave me the first opportunity to talk outside Bielefeld University at the IPK Gatersleben. The talk had the title "CELLmicrocosmos – Building an Environment for the Information Center Cell" in 2007. This was an important starting point for me to be prepared for giving talks at a number of international conferences.

Karl-Josef Dietz collaborated with me on the MembraneEditor when we started with the first bachelor theses on this topic. He taught me a lot about how to search for different publications and appropriate journals and prepared the first publication on the MembraneEditor with me.

Additional thanks go to the participants of the DFG Graduate College on Bioinformatics (GK635) and the International NRW Graduate School in Bioinformatics and Genome Research; as well as their leaders Robert Giegerich and Jens Stoye, and the coordinator Susanne Schneiker-Bekel. During our time as stipend students, we had a lot of nice meetings. Thanks go of course to the German Research Foundation for sponsoring me for three years.

Thanks of course go to my bachelor, master and diploma students: Özgür Ates, Gunther Lukat, Sebastian Rubert, Rudolf Warkentin, Sebastian E. Schneider, Philipp Unruh, Thomas Waltemate, and Marion Zysik.

Special thanks in this context go to Marco Civico and Roland Orlik, who invested a lot of time in the development of CE^3, the perfect basis for our future cooperation.

Very special thanks go to my former student assistant Tim Dingersen, who developed and improved the new version of the MembraneEditor. It was a perfect collaboration: I went to different conferences and came back with a bunch of new ideas – and Tim integrated them, usually even better than expected. My second former student assistant, Christian Gamroth, also implemented many important improvements in the MembraneEditor. The third one,

Alexander Schäfer, currently has to integrate a few new ideas into CE^3 developed by Marco and me. Thanks also to you, guys!

Of course I also would like to thank all students who participated in the different CELLmicrocosmos student projects during the last years: Christina Ander, Ufuk Aydin, Matthias Bartneck, Christian Bender, Regina Bisdorf, Yvonne Dyck, Christian Fink, Florian Heißenberg, André Heißmann, Michael Hertrich, Tobias Hoppe, Arne Husemann, Sebastian Janowski, Lukas Jelonek, Matthias Koch, Christina Kropp, Jörn Künsemöller, Chan Lin, Olga Mantler, Dominik Mertens, Ralf Mertens, Jonas Osterloff, Jan Pieniak, Raphael-Elias Reisch, Nils Rothe, Ralf Rotzoll, Madis Rumming, Norbert Sand, Robert Schmieder, Samy Slaih, Xiaoliang Song, Andrea Steinmetz, Dominik Vahrenhorst, Jörn Winnebald, Pascal Witthus, Martha Zakrzewski, and Yan Zhou. Some of you guys made very important improvements also discussed in this thesis! In the context of CE^3, I also would like to thank Jan Philipp Platenius, Juliane Reich, and Rüdiger Siek for participating in the first beta tests of this software.

Very important for the PathwayIntegration was the data warehouse implementation from my two colleagues and friends Klaus Hippe and Benjamin Kormeier. In addition, we had and we will have many deep “philosophical” discussions at the coffee bar. So thanks a lot for that!

Many thanks go of course also to my other colleagues from our Bio-/Medical Informatics Department of Bielefeld University for many interesting discussions and collaborations: Daniela Borck, David Braun, Sebastian Janowski, Hang-Mao Lee, Klaus Kulitza, Alban Shoshi and our former colleagues Thoralf Töpel and Sridhar Hariharaputran. Of course, special thanks go to Sabine Klusmann, who has a lot of work with all of us and who has always been lending us a patient ear.

The text-mining-based data is provided by Pavel S. Demenkov. He belongs to the same group of Russian colleagues and friends from the Institute of Cytology and Genetics in Novosibirsk as the PhD students Evgeny S. Tiys, Timofey V. Ivanisenko and Anatoly O. Bragin. They were involved in the first application paper which we published on the PathwayIntegration in 2010. The same applies to Vladimir A. Ivanisenko, Alexey V. Kochetov and Nikolay A. Kolchanov, who are the (group) leaders in the aforementioned institute. Together we are meeting several times a year and it is always accompanied by a lot of fruitful discussions and funny meetings in the evening.

For the full implementation information please see Appendix 9.16 and 9.17. For actual information on the people participating in this project the list is found at:

http://team.CELLmicrocosmos.org

Thanks go also to my Italian colleague Patrizio Arrigo, who taught me a lot on cardiovascular diseases and how to analyze its data.

Thanks also to Ming Chen, who accompanied some of my first lectures on Interdisciplinary Cell Visualization and we are looking forward for a nice future cooperation on this topic!

Thanks go to Lars Schäfer who worked in the Marrink MD Group and helped me to make

the PDB format of the MembraneEditor compatible to GROMACS.

That was an important building block for the ongoing cooperation with Jens Krüger. We both established and coached the MD Edition project for the MembraneEditor. It generated already a number of promising results! His initial workshop on MD simulation was very important for me and my students. Thanks in this context go also to Gregor Fels, who allowed our collaboration in the initial phase. Thank you go also to Jens for the valuable remarks after reading some parts of this thesis.

Special thanks in this context go especially to Sebastian Schneider, who checked a large part of my thesis and gave me some important hints about how to shorten it (still it hurts).

Thanks go also to Hendrik Rohn, who developed the HIVE plugin and checked my review.

For some nice discussion and the English correction of this thesis and as well for a bunch of manuscripts, I would like to thank Barbara Davis.

Thanks go to Christian Grove who gave a very nice Blender workshop at the last VizBi 2012 conference. This was important for our first approach to show the first 3D-stereoscopic animation in the cinema with Tobias and the Bender. In this context, also thanks to the guys of the CinemaxX Bielefeld and of course to Paul John and Fabio Magnifico. Their movie lectures were already a lot of fun during my time as a master student!

Because we are talking about institutions, I also would like to thank the Niklas-Luhmann-Gymnasium Oerlinghausen and the Gymnasium Schloß Holte-Stukenbrock who allowed us to test CE^3 in 2010 and 2011 in their school classes.

I have to thank a number of colleagues and friends which I met at some conferences and workshops who often gave me important hints for my work: Wolfgang Brandt, Jochen Heil, Leonhard Henkes, Peter-Paul Heym, Donglin Huang, Andrea Kandlbinder, Ji Kong, Roman Koning, Michael Kreim, Damien Larivière, Dmitry Osolodkin, Peter Krüger, Shaolei Lv, Beate West, and Thomas Wolf. I hope to see you guys again soon at some nice conferences!

In this context, thanks go to Andrei L. Lomize for helping me on the OPM integration and showed me how to work with the topology information. Thanks go also to Gabor Tusnady, who gave Christian and me some information on the transformation of PDBTM matrices.

Anette Hall provided me with a great Packmol example which I used to elaborate the generation of heterogeneous membranes with this tool, thanks for that!

Thanks go to Maryann Martone and the Cell Centered Database which is supported by NIH grants from NCRR RR04050, RR RR08605 and the Human Brain Project DA016602 from the National Institute on Drug Abuse, the National Institute of Biomedical Imaging and Bioengineering and the National Institute of Mental Health. For the modeling of cell components, the CCDB was, is and will be a very important source for us.

Thanks go also to some experts in their fields who lend me their – sometimes critical – ear for usually only a short time but inspired me for the future development of the project: Tony Cass, Jonathan W. Essex, David S. Goodsell, Nicholas Hamilton, Lars Juhl Jensen, Peter D. Karp, and Oliver Kohlbacher. I am sorry to say that not all ideas are implemented yet, but we are working on it! Embarrassingly, I learned very late about the famous cell

illustrations of David, but on the other hand this is quite good, because his approaches are perfectly supported and represented by the work of Graham Johnson, who I would like to thank for the possibility to have a look at his PhD thesis. Of course, our approaches are quite different, but we share the same passion.

In this context, thanks go also to all other image contributors who were so kind to give me the permission to print their images. Their names and/or affiliations are found in the captions.

Thanks go to Ulrich Krohs and the participants at the Synthesis workshop 2011 in Bielefeld. I learned a lot during these two days.

I would like to thank also Tim Nattkemper for the fruitful and critical discussions concerning information visualization. He gave me the valuable hint to look into the book from Haim Levkowitz to find out more about coloring schemes.

Much thanks go also to Mario Botsch. Our quite new collaboration lead already to quite promising results and we have some nice ideas for the future!

I want to make sure that the CELLmicrocosmos project basically follows the aims of the free Open Source community. Many aspects of the different software applications would not have been possible without the use of external Open Source projects. So thanks to all you guys! Those projects used in the context of this work will be listed throughout the thesis. Of course, the publication of CELLmicrocosmos tools is always accompanied by the release of the source code as we did it with the MembraneEditor.

Usually, I would have to dedicate this work to the following persons. But I learned over the years, that although I was able develop this project relatively independently from the beginning, this would never have been possible without the man I dedicated this work towards.

Finally, I would like to thank my family, especially my parents, Lothar and Bärbel Sommer, and my good friends who did not see much of me during the last years. But anyway I know that they always would support me, just like they supported me during the time as a bachelor and master student.

And in the end, I would like to thank Roswitha Hornschu, who always gave me the strength to carry on, who often helped me to focus on the important aspects of this work, who surely had to miss out on a lot of things (especially a real vacation), who inspired me for new interdisciplinary ideas and often had to remind me that there is a life besides Virtual Cells by chasing me through the Teutoburg Forest!

Björn Sommer, September 2012

This work is dedicated to

Dr. Dieter Lorenz

the first person
who gave me the opportunity to work in this field
during my time as a bachelor and master student and
who bequeathed me the perfect laboratory to work on
the vision of a Virtual Cell.
And though he left our group years ago,
this work would not have been possible without him.

Abstract

The modeling of cells is an important application area of Systems Biology. In the context of this work, three cytological levels are defined: the mesoscopic, the molecular and the functional level. A number of related approaches which are quite diverse will be introduced during this work which can be categorized into these disciplines. But none of these approaches covers all areas. In this work, the combination of all three aforementioned cytological levels is presented, realized by the CELLmicrocosmos project, combining and extending different Bioinformatics-related methods.

The mesoscopic level is covered by CellEditor which is a simple tool to generate eukaryotic or prokaryotic cell models. These are based on cell components represented by three-dimensional shapes. Different methods to generate these shapes are discussed by using partly external tools such as Amira, 3ds Max and/or Blender; abstract, interpretative, 3D-microscopy-based and molecular-structure-based cell component modeling. To communicate with these tools, CellEditor provides import as well as export capabilities based on the VRML97 format. In addition, different cytological coloring methods are discussed which can be applied to the cell models.

MembraneEditor operates at the molecular level. This tool solves heterogeneous Membrane Packing Problems by distributing lipids on rectangular areas using collision detection. It provides fast and intuitive methods supporting a wide range of different application areas based on the PDB format. Moreover, a plugin interface enables the use of custom algorithms. In the context of this work, a high-density-generating lipid packing algorithm is evaluated; The Wanderer. The semi-automatic integration of proteins into the membrane is enabled by using data from the OPM and PDBTM database.

Contrasting with the aforementioned structural levels, the third level covers the functional aspects of the cell. Here, protein-related networks or data sets can be imported and mapped into the previously generated cell models using the PathwayIntegration. For this purpose, data integration methods are applied, represented by the data warehouse DAWIS-M.D. which includes a number of established databases. This information is enriched by the text-mining data acquired from the ANDCell database. The localization of proteins is supported by different tools like the interactive Localization Table and the Localization Charts. The correlation of partly multi-layered cell components with protein-related networks is covered by the Network Mapping Problem. A special implementation of the ISOM layout is used for this purpose.

Finally, a first approach to combine all these interrelated levels is represented; CellExplorer which integrates CellEditor as well as PathwayIntegration and imports structures generated with MembraneEditor. For this purpose, the shape-based cell components can be correlated with networks as well as molecular membrane structures using Membrane Mapping.

It is shown that the tools discussed here can be applied to scientific as well as educational tasks: educational cell visualization, initial membrane modeling for molecular simulations, analysis of interrelated protein sets, cytological disease mapping. These are supported by the user-friendly combination of Java, Java 3D and Web Start technology.

In the last part of this thesis the future of Integrative Cell Modeling is discussed. While the approaches discussed here represent basically three-dimensional snapshots of the cell, prospective approaches have to be extended into the fourth dimension; time.

Overview

List of Figures

List of Tables

1 Introduction

1.1 Modeling and Visualizing the cytological Cosmos

One of the smaller units of life is the cell. Since it is dramatically smaller than the moon, no human being has been able to physically visit this biological instance. But since the invention of the light microscope it has been possible to observe the cell. Despite being able to closely study the cell for this relatively long period of time and although living in the post-genomic area, mankind is far from understanding the secrets of life, enclosed in the membranes of the biological cell. Many important factors like basic pathways and a large subset of the proteome are already known. But all this knowledge is represented by a large patchwork distributed throughout a steadily growing corpus of publications, different types of databases and web applications featuring dramatic quantitative as well as qualitative differences.

In the past, there have been scientists such as Gottfried Wilhelm Leibniz (17th/18th century) and Alexander von Humboldt (18th/19th century), centralizing a major amount of wisdom of the *status quo* in one person[1]. An inherent part of the present educational system is concerned with a characteristic drama character for this type of scientist: Goethe's Faust, the story of a man on the search for ultimate wisdom. Therefore, the principle of the "universal scientist", the polymath, is part of today's cultural heritage. And it is exactly this type of scientist who would be needed to answer the questions to solve the secret of the cell.

The *information overload* of the post-genomic area running in parallel with the information age features an immense amount of data, contributed by millions of scientists. While in the past the scientific community was a relatively small family-like alliance sharing the wisdom only with the establishment, today's knowledge is decentralized, spread throughout millions of people and the Internet in a quite democratic way. But accompanied with the positive developments of the last decade the amount of superficial knowledge and disinformation is growing ever faster. Therefore, approaches are indispensable to filter the false positives. Again, the need for the universal scientist emerges.

Resulting from these first observations it can be concluded, that one of the most important tasks of the information age is the compensation of the absence of the polymath which is caused by today's information overload.

One key to this problem is interdisciplinary research. To understand the reality of the cell, it is not sufficient to exclusively examine a distinct research area. For this purpose, it is not sufficient to analyze the cell on a platelet with a microscope from a biological point of view. It is not sufficient to focus the research at the chemical interactions of metabolic pathways. And it is not sufficient to simulate the atomistic behavior of a membrane using physical properties. All the mentioned areas of research need scientists having expertise acquired over many years of high-specific studies. The question resulting from these observations is:

1 In biological context these persons could be defined as instances similar to a hub.

How to combine these complex research fields? What can be done to bridge the gap between these highly diverse research areas?

One answer to this question is: visualization. Scientific as well as information visualization are widely accepted as appropriate approaches to share, specify and classify today's knowledge. It is used in interdisciplinary communication of ideas and results between scientists and to present their results to the community and to the public as well as potential supporters and investors.

It was already mentioned that cytology is relevant for many different disciplines. Therefore, the process of creating a cell is quite complex. Another drawback is the fact that only fragmentary knowledge is available. For example, the extraction of membrane proteins from their environment is a task solved only for a relatively small set of molecules. Their structural properties are therefore widely unknown. This aspect alone is sufficient to illustrate the impossibility of generating a realistic cell model.

Another key term was mentioned: the “model”. The vision of cytology is to scan a biological cell with techniques similar to Electron Tomography, and to immediately generate a digital copy of the cell, featuring all information needed: intercellular and intracellular trafficking, chemical and physical interactions and membrane compositions, in addition to all functional as well as structural properties needed to understand the cell. Since this is not presently possible, the combination and synthesis, respectively, of today's fragmentary knowledge is a task which has to be fulfilled prior to the generation of explanatory visualizations. And in the context of this work, this process is called modeling.

From a functional perspective, one of the first approaches trying to visualize the aggregate of all biochemical pathways was first published in the year 1968 by Gerhard Michal: the “Biochemical Pathways” map. This impressive visualization, as shown in Figure 1, is an accumulation of the work between scientists and designers, trying to reduce the available information to create a refined overview.

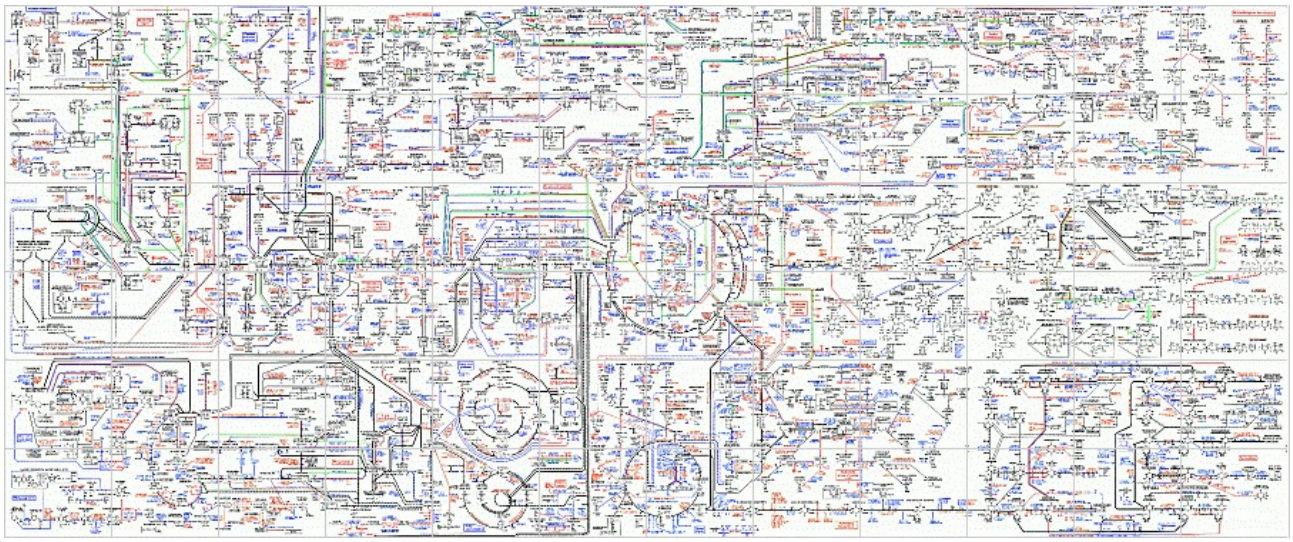


Figure 1: A fragment of Gerhard Michal's Metabolic Pathways map
As available from ExPASy (Courtesy of/Copyright © 2012 by Roche Diagnostics GmbH. Reprinted with permission from [Mich12])

Examining microscopic images, the visual and structural complexity of the cell is not assessable. Also here, the generation of a visualization has to focus on specific properties – visual compromises are indispensable. For decades, David S. Goodsell's illustrations have been famous for the simplification of complex structures. Figure 2 shows a recent example of his work, a complete Mycoplasma mycoides cell. Although painted in two dimensions, his work takes three-dimensional structural properties of the proteome into account.[2]

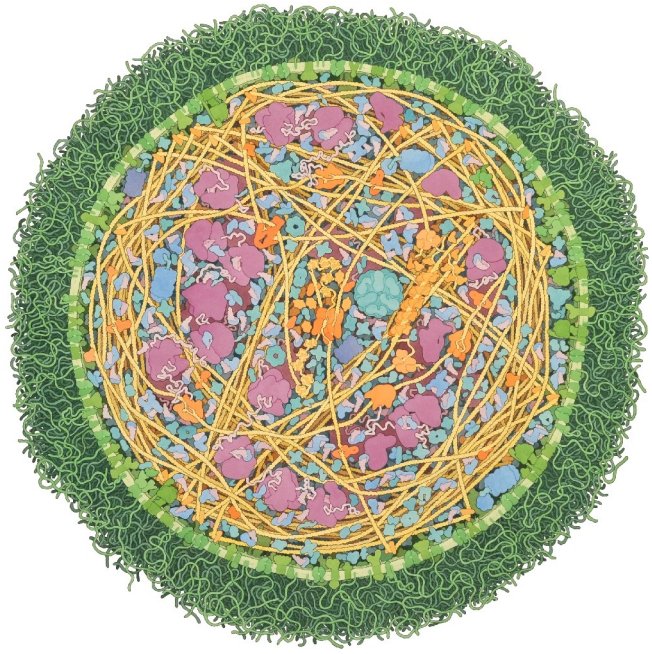


Figure 2: A complete Cell painted with Watercolor: Mycoplasma Mycoides (Courtesy of/Copyright © 2011 by David S. Goodsell, Scripps Research Institute. Reprinted with permission)

Cellular Landscapes is a term which appropriately describes Goodsell's as well as related works [CBBK10].[3] The three-dimensional reality of the cell is projected to a two-dimensional canvas to generate precise images of different cell types.

With the advent and commercialization of computer technology, the possibilities of visualization have dramatically increased. Today, cellular animations can be found in movies like the “War of the Worlds” adaptation of H. G. Wells novel by Steven Spielberg (2005) as well as high-quality educational animations [Lok11].

But if the focus should be laid on the modeling of a cell, combining different information sources, the interactivity of such an approach is a key aspect of this implementation. And because the structural reality of the cell is three-dimensional, also this modeling approach should be a navigable three-dimensional cell environment. In this context the question arises, how can this spatial cell model be combined with a pathway as shown in Figure 2.

The term “cellular landscape” was already mentioned before, but does the term also apply to this approach? An elementary aspect of a landscape is the position inside the environment, from an earth-bound position. Indeed, looking at two-dimensional projections of the cell this impression applies also to the cellular landscapes. But navigating through an interactive cell environment, the impression will be quite different. Cell components appear to be floating to different positions like planets embedded in the cellular liquid, the cytosol. In the center of most cells a core exists, the nucleus, a compartment with life-giving significance for the other cell components like the sun for its surrounding planets. Continuously, high traffic throughout the cell exists: the different cell components are

2 Indeed, David S. Goodsells illustrations are based on PDB structures which will be discussed later (Section 3.2.2.1.1). He is also famous for his contributions to the RCSB section “Molecule of the Month”.

3 The term “Cellular Landscapes” was also used by David S. Goodsell during his key note lecture entitled “Communicating Biology Visually” at the VizBi conference in Heidelberg, March 6, 2012.

interchanging matter as well as information. Small proteomic units – the ribosomes – are constantly translating information important for the synthesis of new cellular elements. And there are many other smaller molecules and even larger protein complexes distributed throughout the cell like comets in the universe. The duplication as well as starvation are elementary features of cellular components as well as planets and/or stars. And while the *homo sapiens* is usually traveling earth-bound, the navigation through an interactive cell would be like hopping from planet to planet.

Considering these observations an appropriate metaphor for an interactive cell is the universe – the *cosmos*. But especially when focusing at a single cell, it is important to avoid the impression that this cosmos is autarkic. Cells usually co-exist, they are dependent from their cell-external environment, the tissue. An exemplary hierarchical structure of a hepatocyte cell in conjunction with its environment is shown in Figure 3.

```
Organism:              Eukaryote
Class:          Mammalian
Species:               Homo Sapiens
Body Part:             Torso/Chest
Organ:          Liver
Tissue:                Liver Tissue
Cell:                  Hepatocyte
Cell Component:        Mitochondria
Cell Membrane: Inner Membrane
Molecular Raft:        No Raft (could be e.g. a cholesterol raft)
Protein Complex:       2-Oxoglutarate Dehydrogenase E1
Protein:               2-Oxoglutarate Dehydrogenase E1 component (EC 1.2.4.2)
Polymer/Chain:         1/A
Atom:                  C
```

Figure 3: The hierarchical structure of a human Hepatocyte cell
Beginning at the organism and finishing at the C atom

Obviously it cannot be claimed that a cell is a self-sustaining system, therefore the addition of the term *micro* is an important step to take.[4] Even if the cell is often regarded as a closed system, this microcosm depends on its environment, the macrocosm.

The synthesis of the preceding aspects leads to the new term appropriately describing the three-dimensional reality of the cell: *cell microcosmos*.

Figure 4: Elementary Properties of the CELLmicrocosmos logo
Left: elements of the logo; center: the logo; right: the first frame of the cell animation for the initial Bachelor Thesis [Somm04]

4 Interestingly, it is a better known fact that the interrelation of microcosmos and macrocosmos is also an important aspect in Goethe's Faust.

When the first idea for the eponymous project was developed in 2003 during the author's bachelor thesis, the design of the logo was adjusted to this idea [Somm04]. Two interrelated circles represent the microcosm – the cell – (small white circle in the focus) and macrocosm (large black circle representing the elementary background in the shadow)[5]. Simultaneously, the logo contained abstractions of the letters in the term "CELL". The scroll with the name was cut at the bottom and the top to outline the imperfection, the fragmentary character of the cell.

Finally, the chosen font style is elementary for the logo. But before going into too much detail at this point, a step backward should be taken, reevaluating the fundamental problem of cell modeling. It was already pointed out that cytology is a very complex area. Even if it would be possible to insert all components in a realistic concentration into the cell model, a visual chaos would result. And especially if the objective of a visualization is the illustration of distinct cellular aspects of the cell, a reduced form would be the first choice. For this purposes, reduction and abstraction are the key terms.

For the creation of protein-based shapes, cell components or even complete cellular environments, segmentation tools are used. A mitochondrion for example, consists of different sub-compartmental structures like the inner membrane, the outer membrane and the intermembrane space. When creating a shape, it is necessary to define concrete points from microscopic image stacks. The intermembrane space is in reality the complete area between the inner and outer membrane, but the modeler has to define discreet points representing this space. Obviously the reduction is indispensable in this case.

From 1919 to 1933, Walter Gropius in Weimar headed the school of arts known as "Bauhaus". The eponymous style (also known as the "international style") was and is famous for its visually reductive design approaches. One of the initial intentions of Gropius was the return to interdisciplinary craftsmanship with the focus on the architecture.

"Das Endziel aller bildnerischen Tätigkeit ist der Bau! [...] Architekten, Maler und Bildhauer müssen die vielgliedrige Gestalt des Baues in seiner Gesamtheit und in seinen Teilen wieder kennen und begreifen lernen, dann werden sich von selbst ihre Werke wieder mit architektonischem Geiste füllen, den sie in der Salonkunst verloren." [Grop19]

This completes the initial idea of this text: the modeling of a cell is not a task for one single scientific discipline; cooperation and the synthesis of knowledge will be tasks for the scientific craftsmanship of the future.

And simultaneously, the appropriate font style for the CELLmicrocosmos logo was – of course – a Bauhaus font.

Considering this text the following aspect should be eroded. Bauhaus is a style which rose out of architecture. The cell – even if highly depending on structural properties – is a biological entity featuring organic, not technical, formal structures. Architecture and biology, are these not two controversial disciplines?

5 The large circle can be exemplarily interpret as the earth, characterized by the unique feature (based on the recent knowledge, until eventually another planet with life will be found) of transporting multiple forms of life through the universe, originated by the biological cell.

The answer to this question is: no. And this answer can be proven by the impact of the architectural research and its implications of a man living and teaching on the other side of the Atlantic during the triumph of the international style: Buckminster Fuller[6]. This autodidact published and realized many avant-garde ideas during those time. He was known for his organic-related structures, a supporter of the "form follows function" principle. In 1947, Fuller invented the Geodesic Domes [Full70, p.33]. These architectural scaffold-related structures following strict mathematical rules was used as the basis for different constructions. For example, at the Expo 1967 in Montreal a Geodesic Dome featuring 250 ft. in diameter and 137 ft height was constructed. This prototype, shown in Figure 5, was intended as a physical microcosm, providing enough space for small communities [Full70, p.368].

Figure 5: The Dome at the Expo 1967 in Montreal, built by Fuller, Sadao and Geometrics In 1976 the acrylic outer shell burned away, but the superstructure is still intact (Courtesy of/Copyright © 2010 by Charles Hoffman/flickr.com. Reprinted based on CC [Hoff10])

In 1997, Robert F. Curl Jr., Sir Harold W. Kroto and Richard E. Smalley won the Nobel Prize for a discovery which was essential for nanotechnology: the self-assembly of a C_{60} molecule. Three years after the creation of the Expo dome, the existence of this molecule was already suggested, but it took another 15 years until it was experimentally proven. And the name those scientists gave to this molecule was "C_{60} buckminsterfullerene". Besides the Geodesic Dome, Fuller patented world-map projections like the "Stardome map of the sky" which had exactly the same shape as the C_{60} molecule. Its discovery was the "birth of fullerene science", an essential event of the today widely nanotechnology[7] [Krot97, p.1579].

6 In this context it has to be mentioned that Buckminster Fuller – although parallels to the international style can hardly be completely denied (Fuller's style as well as the one of Bauhaus were following the principle "form follows function") – was always emphasizing the difference between his vision and the one of Bauhaus ([Full70, p.64 ff.]). Anyway, it can be stated that Fuller insisted in showing the superstructure like those of the Expo 1967 dome while a Bauhaus-affiliated architect might tend to hide this structure beyond a smooth surface. From this point of view, the CELLmicrocosmos project also tends to the Bauhaus style for two reasons: a) the complete intracellular structure with all its molecular properties is – as mentioned before – not known, therefore it cannot be realistically modeled and b) it will be very problematic to navigate through such a structure or to highlight distinct intracellular functional relationships without losing the overview.

7 C_{60} is the smallest stable fullerene featuring a size of 7 Å in diameter. It has a shape like a soccer ball, containing 12 pentagons and 20 hexagons, a truncated 1-Icosahedron [Krot97].

Obviously, architectural design provided a prediction of an atomistic structure prior to its discovery. Simultaneously it emerges that the cosmic metaphor always fascinated and inspired researchers of life-science-related disciplines[8].

1.2 CELLmicrocosmos: A bidirectional Approach

Finally, the discussed principles should be summed up to highlight the objectives of the CELLmicrocosmos project.

The aim of the CELLmicrocosmos project is the three-dimensional, interactive and interdisciplinary modeling and visualization of cells based on various data sources. Problem-specific visualization approaches should enable the understanding of internal cellular structures and their functioning.

The complexity of this topic is accommodated with a bidirectional approach:

- *Top-Down Approach:* The modeling process starts from the mesoscopic scale. Subcellular structures like cell components are modeled as characteristic three-dimensional instances enclosed in the cell membrane. Models featuring a high grade of abstraction are possible at this level as well as those derived directly from microscopic data.
- *Bottom-Up Approach:* The molecular level is addressed by the modeling of molecular membranes which can be associated with the components of the mesoscopic scale. In this way “islands of knowledge” can be created, containing highly-specific molecular structures based on published material.

The mesoscopic and the molecular level are interconnected by the functional level represented by biochemical pathways (Figure 6).

This thesis will give an overview of the different approaches and techniques needed for the modeling of a Virtual Cell, with the major focus on Integrative Bioinformatics.

```
Top-down (mesoscopic level) → (functional level) ← Bottom-Up (molecular level)
```

Figure 6: The Bidirectional Approach of CELLmicrocosmos

1.3 CELLmicrocosmos: An interdisciplinary Approach

The term interdisciplinarity was already mentioned in different contexts. By now, it should be clearly recognizable to the reader that interdisciplinary research cannot be done by one single researcher. A topic like the modeling of a cell needs many researchers from different fields. And although the work presented in this thesis is already quite complex, there is of course a great scope for improvements and extensions into various directions.

The author of this work could normally present in this thesis a structure, a rudimentary framework and many ideas to be implemented in the future. But with the help of many students, after many discussions with researchers and the participation in various conferences and workshops, it was possible to build the framework presented in this work. During its development one published, quite stable tool was created as well as a second tool

8 Only two examples interrelated with this work: “Symmetry, Space, Stars, and C_{60} (Nobel Lecture)” by Harald Kroto [Krot97]; “Reactome: clear view of a starry sky” by Morag Robertson [Robe04].

which to date has not officially been published but was used for different publications. The latter software will be issued during the coming months.

From the beginning of the CELLmicrocosmos project, it was initiated, structured and led by the author of this work. Many thousands of lines of codes have been written during this project. But of course, the implementation work could not have been done only by the author. Because of the scope of all the contributors, they should be mentioned in particular in the Acknowledgments. In addition, a detailed overview of people participating in the student projects, writing bachelor, master and diploma theses, modeling cell components and – of course – writing code, are mentioned in Appendix 9.16 and 9.17. There will be – however – some chapters, where concrete algorithms and methods will be discussed which have been personally developed and/or implemented by the author of this thesis.[9]

This work does not focus on the introduction of new algorithms nor the optimization in the field of graphics programming. Instead of this, new methods will be introduced and their implementation will be sketched. It will be shown that cell modeling is an extremely complex topic. And to cover only all approaches which have been developed by the CELLmicrocosmos project in detail, this thesis would have to contain more than thousand pages. For this purpose, the reader is referred to the aforementioned student theses already finished and those to come.

Therefore, only an overview will be presented: an introduction in cytology and cell modeling, in most of the leading projects in this field and – of course – the CELLmicrocosmos approach itself.

1.4 Project and Thesis Structure

In the context of this work, four different cytological levels will be discussed. Resulting from the different underlying levels, this thesis is divided into four major sub-sections which will be classified in the following sections.

1.4.1 Mesoscopic Level [MES]

The mesoscopic level [MES] describes the structure of the cell: the different cell types, the different cell components and their different layers. It does not take into account the molecular structure of the single components. The resolution of these aspects roughly lies between a few thousand to hundreds of Nanometers.

Sub-project name: CELLmicrocosmos 3 CellEditor (Cm3/CmCE)

1.4.2 Molecular Level [MOL]

The molecular level [MOL] is examining the 3D structures of proteins, lipids and membranes, descending to the atomic level. Therefore, the resolution goes down to a few Ångstrom.

Sub-project name: CELLmicrocosmos 2 MembraneEditor (Cm2/CmME)

9 The Wanderer was developed and implemented by the author (Section 5.2.2.6). The ISOM and UUUSphere Layout were implemented by the author (Sections 5.3.2.5.1 and 5.3.2.5.2).

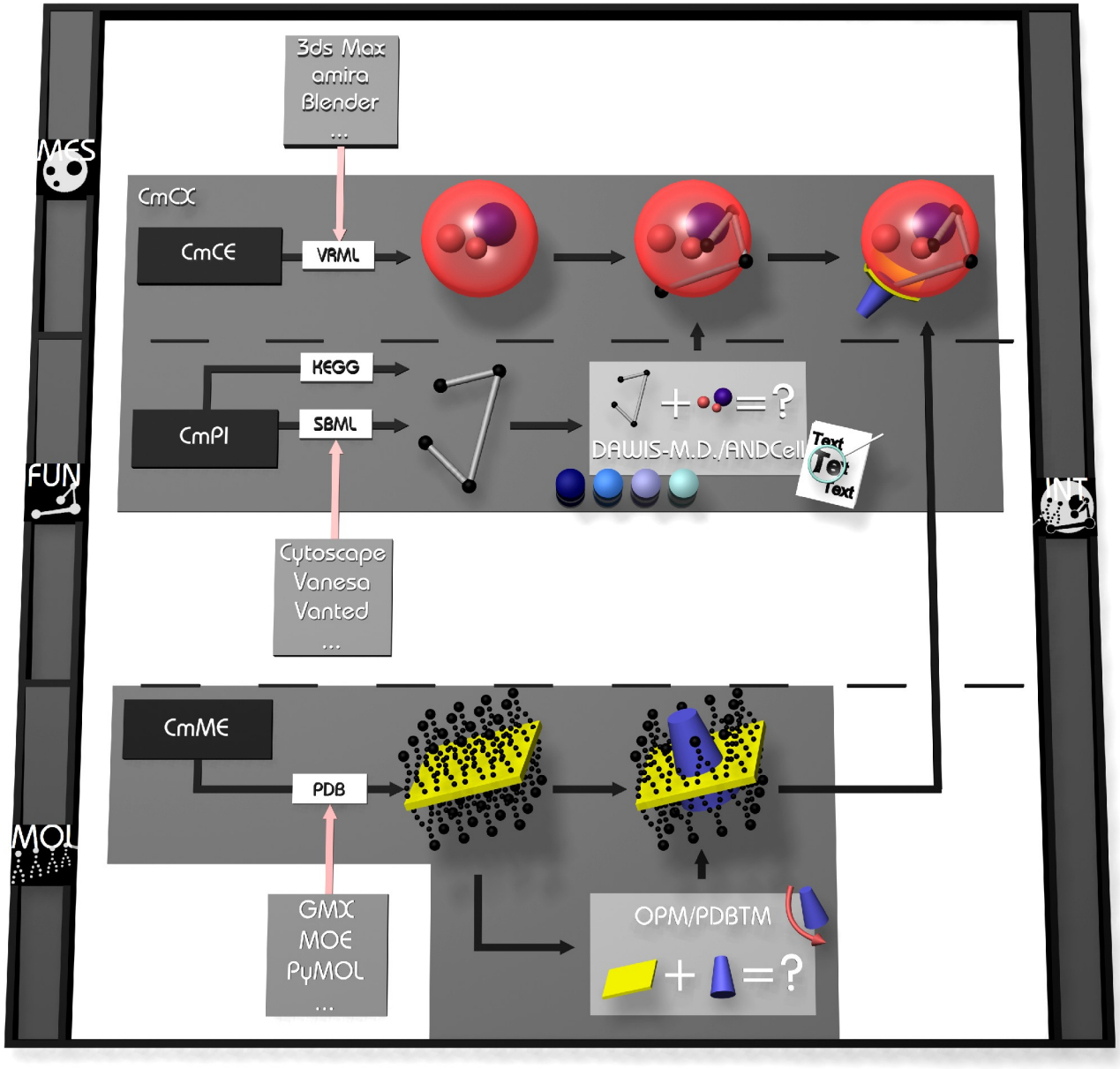


Figure 7: The modeling pipelines of the CELLmicrocosmos Projects

1.4.3 Functional Level [FUN]

The functional level [FUN] is examining the structure of the pathways correlated with cell components and molecules – the layer combining the mesoscopic and molecular level. These may apply to metabolic pathways, protein-protein-interaction networks, signaling pathways, etc. In this work, the focus is made exclusively on protein-associated networks with a further specialization to metabolic pathways.

Sub-project name: CELLmicrocosmos 4 PathwayIntegration (Cm4/CmPI)

1.4.4 Integrative Level [MES+MOL+FUN]

A future outlook is presented in the chapters regarding the integrative level [MES+MOL+FUN]: the combination of all aforementioned topics. How can the mesoscopic, the molecular as well as the functional level be united visually? How can the different specialized approaches previously presented be correlated? These are first

approaches which have to be improved in the near future, but it will be shown that the first results already look promising.

Sub-project name: CELLmicrocosmos 1 CellExplorer (Cm1/CmCX)

The biological background, Chapter 2, provides an introduction to the basics of cell biology which should enable readers of other disciplines to access this work. It will be elucidated, how the different cytological levels can be differentiated and how they are interrelated. In Chapter 3 the technical background will be explained. It will be shown, in which way the microcosm of a cell is visualized and digitalized. Moreover, first basic concepts of cell modeling will be introduced. Chapter 4 introduces related approaches developed during the last decade. Then, the methods and implementation Chapter 5 follows, introducing the basic concept of the CELLmicrocosmos project. In this context the modeling pipelines associated with the different aforementioned projects – as shown in Figure 7 – will be explained in detail. Then, a number of application cases are discussed in Chapter 6. Finally, Chapter 7 will cover the conclusions and the outlook.

Each of the following chapters 2 to 7 will be concluded with the graphic shown in Figure 8, containing the progress indicator. The four symbols associated with the cytological levels are also found in Figure 7. Moreover, the usage of the digits in the chapter numbers is illustrated: the first digit is the chapter number, like 2 for the biological background. The second digit, the subsection, defines the level category: 2.1 concerns the biological background at the mesoscopic level, 6.4 the application cases at the integrative level.

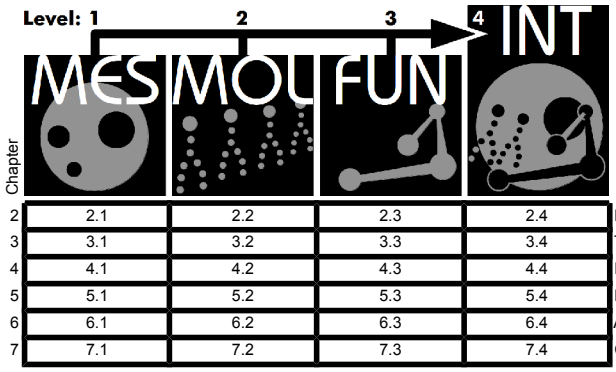


Figure 8: The progress indicator including the links to all major chapters of this thesis

2 Biological Background

Before the methodology of cell modeling and the subsequent realization can be discussed, the reader should be familiar with the background. This chapter starts with the biological background, discussing the basics and the cytological developments which form the prerequisites for the following chapter – the Technical Background – as well as for the whole thesis. This introduction follows – just like the cytological history – the top-down approach (Section 2.1.3.3), starting from the mesoscopic scale, slowly approaching the molecular structure.

2.1 [MES] Mesoscopic Composition

Prior to the generation of a cell model or the localization of proteins it is indispensable to know the cell structure, its components and the different types of cells. An exhausting list of cell components lies beyond the scope of this work. In the following sections a number of cell components are listed which are very important for a membrane-centered look at the cellular environment.

2.1.1 The Cell

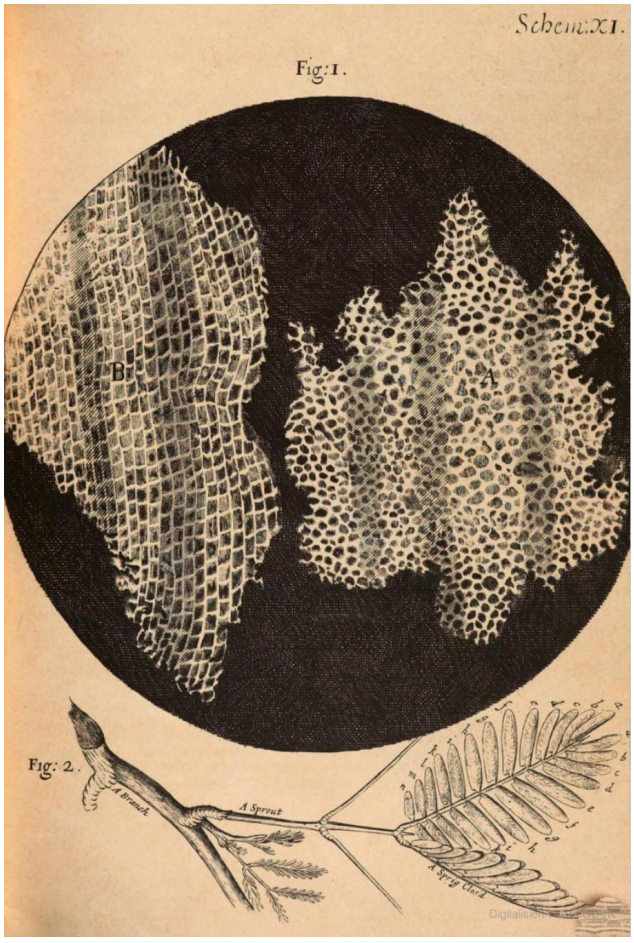


Figure 9: The first observed "cell"
The cellular structure of cork as published by Robert Hooke in 1665 [Hook65, p.114]

In 1665, by looking at a slice of cork, Robert Hooke first introduced the term *"cell"*. This was one of the first large achievements of the light microscope which was invented a few decades earlier around 1610.

In the chapter entitled "Observation XVIII. Of the Schematisme or Texture of Cork, and of the Cells and Pores of some other such frothy Bodies" of his "Micrographia", the following sentence is found:

"Next, in that these pores, or cells*, were not very deep, but consisted of a great many little Boxes, separated out of one continued long pore, by certain Diaphragms, as is visible by the Figure B* (here: Figure 9, the author)*, which represents a sight of those pores split the long-ways"* [Hook65, p.113].[10]

10 It should be mentioned that the investigations of Hooke were not exclusively focused on the microscopic world. The development of the microscope was accompanied by the invention of the telescope. Therefore, the Micrographia also contained an essay discussing and illustrating the surface of the moon [Hook65, p.244p]. This is only one example for the correlation between the microscopic and macroscopic world.

Just a few years later in the 70s, Antony van Leeuwenhoek was able to observe a number of different cells, for example: bacteria, red blood cells and spermatozoon. Until the term *cell theory* was established, it would continue more than hundred years. Theodor Schwann and Matthias Schleiden introduced their cell theory in 1838. Already in those days a differentiation was made between plant (Schleiden) and animal (Schwann) cells. Twenty years later Rudolf Virchow completed the first version of the cell theory by stating that all organisms consist of cells and that life depends on cell division [Burg08, Virc58, VoVP06].

2.1.2 Cell Types

Today, there are three major classes of living organisms known: the *eukarya/eukaryotes*, the *(eu-)bacteria* and the *archae*. The latter two classes belong to the *prokaryotes*.

The eukaryotes include a nucleus, a usually sphere-like area inside the cell which contains the DNA. Animals, plants and fungi belong to this class. They may be unicellular or multicellular and contain different types of cell components. Many of them are surrounded by intracellular membranes.

The *prokaryotes* do not contain a nucleus, the DNA and all other components are distributed throughout the cell, because they do not contain any intracellular membrane-surrounded components. They are all usually unicellular. Archae are usually found in areas where eukaryotes are not able to live and the differences to the bacterial as well as to the eukaryotic DNA are similar significant [AJLR02, VoVP06].

2.1.3 Cell Components and their Substructures

In this work, the focus will be laid on eukaryotes. Therefore, this section will cover some of the most important cell components which will later have relevance for the modeling process. Of course, a large number of additional cell components exist.

Every cell component is involved in different biochemical reactions. A small subset will be introduced in Subchapter 2.3. Moreover, intentionally no pictures of the components will be shown in this section. The visual structure of the cell will be revealed by microscopy techniques, starting in Section 3.1.1.

2.1.3.1 Plasma membrane

The inner cell is bordered by the *cell/plasma membrane*. It subdivides the interior from the surrounding external plasma and neighboring cells. But the membrane also regulates communication with the environment: Substances needed by the cell are imported, intracellularly synthesized substances are exported. Low-molecular substances are able to pass directly through the membrane: passively, by diffusing the lipid bilayer or actively by passing through specialized protein channels. High-molecular matter are either directly disassembled on the surface of the membrane, or they are imported by different types of *endocytosis*: *pinocytosis* describes the import of solvated material, *phagocytosis* the one of solid material. Using this two mechanisms, small vesicles dissolve from the membrane, surrounding the substance, penetrating the membrane and finally releasing its freight on the intracellular side [Coop97].

The plasma membrane consists of specialized regions and compositions. Apical plasma membrane regions, e.g. bile canaliculus[11] in hepatocyte, are in contact with the cellular environment. Therefore, they contain specialized components like microvilli which extend the surface of the membrane and which approach the surrounding substances. In contrast to apical membranes, basolateral membranes communicate with neighboring cells. In the hepatocyte, the basolateral membrane contains specialized structures:

- Tight junctions between two hepatocytes prevent the mixing of the cellular content with the environment.
- Gap junctions enable small molecules to travel between two cells by forming intercellular pores.
- Desmosomes support the attachment of a cell to the cytoskeleton as well as to a neighboring cell [Genn89].

2.1.3.2 Cell Walls

Many eukaryotes are additionally surrounded by a *cell wall* which consists mainly of polysaccharides (Section 2.2.1.1): algae and higher plants tend to contain cellulose, whereas fungal cells contain chitin. Often the thin flexible outer *primary cell wall* surrounds *secondary cell walls*, which start to generate after cell growth has ended and increase the stabilization by forming cellulose fibers of opposite orientations. Inside the secondary cell wall the plasma membrane is located [Coop97].

The shape of the cell walls is also regularly forming the plant: they are highly organized, sometimes resembling a jigsaw puzzle. The intracellular high turgor pressure, the driving force for cell growth, needs a very stable structure of the surrounding walls [AJLR02].

2.1.3.3 Cytosol

The *cytosol* surrounds all membrane-enclosed cell components inside the cell. Together with most intracellular components listed here, it forms the *cytoplasm*. The cytosol contains thousands of different enzymes. For example, they are catalyzing reactions like glycolysis (Section 2.3.5) as well as the biosynthesis of nutrients like sugars and fatty acids. In addition, the cytosol contains the cytoskeleton, a large number of different interconnected filaments with stabilizing and mobile functionalities [ABLR83, VoVP06].

2.1.3.4 Nucleus

The DNA (Desoxyribonucleic Acid, Section 2.2.1.2) of eukaryotic cells is mainly located inside the *nucleus* which regularly reserves 10% of the cell volume. Large loops of DNA form the *chromatin*. The *nuclear envelope* surrounds the nucleus and consists of two concentric membranes (the *inner* and *outer nuclear membrane*) which are connected by *nuclear pore complexes*, each about 120 nm in diameter. These pores allow the export of RNA (Ribonucleic Acid, Section 2.2.1.2), macro- and polar molecules to the extranuclear area, the cytosol. Then again they enable the import of regulatory proteins into the nucleus. Both membranes are phospholipid bilayers (Section 2.2.1.3), allowing small non-polar

11 Bile canaliculus are small tubular channels in hepatocytes which collect the secreted bile.

molecules to penetrate. The nuclear envelope is directly connected to the endoplasmic reticulum (its internal space is continuous with the space between the inner and outer nuclear membrane, the perinuclear space) and surrounded by a stabilizing network of intermediate filaments. The second, more regularly organized intermediate filaments are called the *nuclear lamina*, located inside the nucleus. The nuclear lamina creates a thin layer besides the inner nuclear membrane. It is connected to the chromatin [AJLR02, Coop97, Genn89].

2.1.3.5 Nucleolus

The *nucleolus* is found in the center of the nucleus. There it represents the nuclear site of rRNA transcription (Section 2.2.1.2), processing and assembling of ribosomes [Coop97].

2.1.3.6 Endoplasmic reticulum

The *endoplasmic reticulum (ER)* occupies a large amount of the animal cell. There are two characteristic structures of the ER known: the *rough* and the *smooth ER*. The rough ER is a large stacked disc-like structure called *cisternae*. Here, many substances are prepared to be incorporated in other cellular processes and components. It is the site for the biosynthesis of proteins which are afterwards exported intracellularly to other components, e.g. for the internalization in lysosomes or the incorporation into the plasma membrane, which may be partly redirected by the golgi apparatus. In addition, it manipulates proteins imported from lysosomes to mature hydrolytic enzymes (Section 2.3.1) to be exported afterwards to the extracellular environment. It is called rough ER because it is surrounded by ribosomes involved in the aforementioned protein biosynthesis. They are translating the messenger RNA (mRNA, Section 2.2.1.2) coming from the nucleus into proteins. The smooth ER is a complex tubular network which is involved in other reactions: the fatty acid desaturation and the sterol biosynthesis are important steps of the lipid biosynthesis. In addition, detoxification reactions are located here [Genn89].

2.1.3.7 Ribosome

The *ribosome* is relevant for the protein synthesis. They are composed of RNA and proteins [Coop97].

2.1.3.8 Golgi apparatus

Similar to the rough ER, the *golgi apparatus* consists of a number of cisternae. But they are not occupied by ribosomes, normally do not demand a large amount of cellular space and are surrounded by vesicles. Two regions are differentiated: vesicles enter the golgi apparatus on the *cis-golgi side* and leave it on the *trans-golgi side*. These vesicles transport proteins which are modified by the golgi apparatus, e.g. glycosidases and glycosyltransferases. In addition, it finalizes proteins exported from the ER and transfers them to the plasma membrane, to lysosomes or to the cell environment [Genn89].

2.1.3.9 Peroxisome

The *peroxisome* breaks down small molecules like xanthine, amino and fatty acids (Section 2.3.1). The name results from a byproduct of oxidative reactions: hydrogen peroxide. Because this compound is harmful to the cell, the peroxisome also contains the enzyme *catalase*, which breaks down this product [CoHa07, Genn89].

2.1.3.10 Mitochondrion

The *mitochondrion* is a rather complex cell component: it consists of an *inner* and *outer membrane*. While the outer membrane is freely permeable to small molecules and handles the import and export of proteins, the inner membrane is semi-permeable and highly involved in many mitochondrial reactions. In between, the *intermembrane space* is located, which is equivalent to the cytosol in the concentration of ions and metabolites. The inner membrane forms fibrillar structures – the *cristae* – surrounding the so-called *matrix*. Here, enzymes are generating the energy needed by the cell for various chemical reactions: ATP is gained as a product of the citrate cycle (Section 2.3.5). In addition, the mitochondrion is the only cell component besides the nucleus in eukaryotic cells which possesses its own DNA [CoHa07, Genn89, VoVP06].

2.1.3.11 Chloroplast

In contrast to the mitochondrion the *chloroplast* is only found in plant cells where it is responsible for photosynthesis. But similar to the mitochondrion it has a quite complex structure: The surrounding *envelope* consists of an *inner* and *outer membrane*. The inner part of the chloroplast is called *stroma*. Here, a number of so-called *thylakoid membranes* are found where the photosynthetic reactions take place. They exhibit different structural compositions: Some of them are tightly stacked in cylinder-like structures, others are unfolded and exposed to the stroma [Genn89].

2.1.3.12 Vesicle

In the cytosol, different types of *vesicles* are trafficking throughout the cell membrane and different cell components (Section 2.1.3.1). The following three cell components represent all subclasses of vesicles [ABLR83].

2.1.3.13 Endosome

Another vesicular compartment is the *endosome* which is involved in the sorting of nutrients taken up by endocytosis. Afterwards, these nutrients are transported to lysosomes [Coop97].

2.1.3.14 Lysosome

In contrast to the peroxisome which degrades smaller molecules, the *lysosome* breaks down larger molecules. All intracellular material and even obsolete cellular components are normally disassembled in lysosomes. They contain a number of hydrolytic enzymes like lipases and proteases which were finalized by the ER [Genn89].

2.1.3.15 Vacuole

The *vacuoles* are large membrane-enclosed globular structures which are especially found in plant cells where they maintain turgor pressure. They contain nutrients and waste products and degrade macromolecules [Coop97].

2.1.3.16 Extracellular matrix

The *extracellular matrix* surrounds all cells. It consists of secreted proteins and polysaccharides connecting cells and tissues [Coop97].

2.1.3.17 Volume of Cell Components

Intracellular Compartment	Percentage of total cell volume
Cytosol	54
Mitochondria	22
Rough ER cisternae	9
Smooth ER/golgi cisternae	6
Nucleus	6
Peroxisomes	1
Lysosomes	1
Endosomes	1

Table 1: Relative volumes occupied by major intracellular compartments in a hepatocyte
[AJLR02]

The different types of cell components fill different volumes. Table 1 compares the most important cell compartments of a hepatocyte sorted by their volumes. The overall volume of an average hepatocyte is approximately 5,000 μm^3 [AJLR02].

2.2 [MOL] Molecular Composition

Now that the major components of the cell on the mesoscopic level have been discussed, this subchapter will focus on the components at the molecular level. These entities will be especially relevant during bottom-up modeling approaches (Subchapter 1.2). Because this work follows a membrane-focused view of the cell which is accompanied by simplification processes, it is important to highlight the underlying biological complexity. Therefore, the focus will lie on the structural constituents of the cell, creating the building blocks for the previously discussed cell components.

Moreover, it will be shown that the development of theoretical models was and is indispensable during communication and comprehension processes related with these complex cytological coherencies.

2.2.1 The Molecules of Cells

Most molecules in cells can be subdivided into five major classes: the carbohydrates, the nucleic acids, the lipids, the proteins and small molecules.

2.2.1.1 Carbohydrates

Carbohydrates are sugars which are the basic constituents of many cell components and which are indispensable for the cellular energy budget.

The basic and eponymous formula of a carbohydrate is $(CH_2O)_n$: C for carbo and H_2O for hydrate. If *n* is one, then it is a *monosaccharide* (mono: Lat., solely), the simplest form of sugar. If n is between two to 99, it is a *oligosaccharide* (oligo: Lat., few) and if more sugars are joined together – which may be hundreds or thousands of sugars – it is called *polysaccharides* (poly: Lat., many).

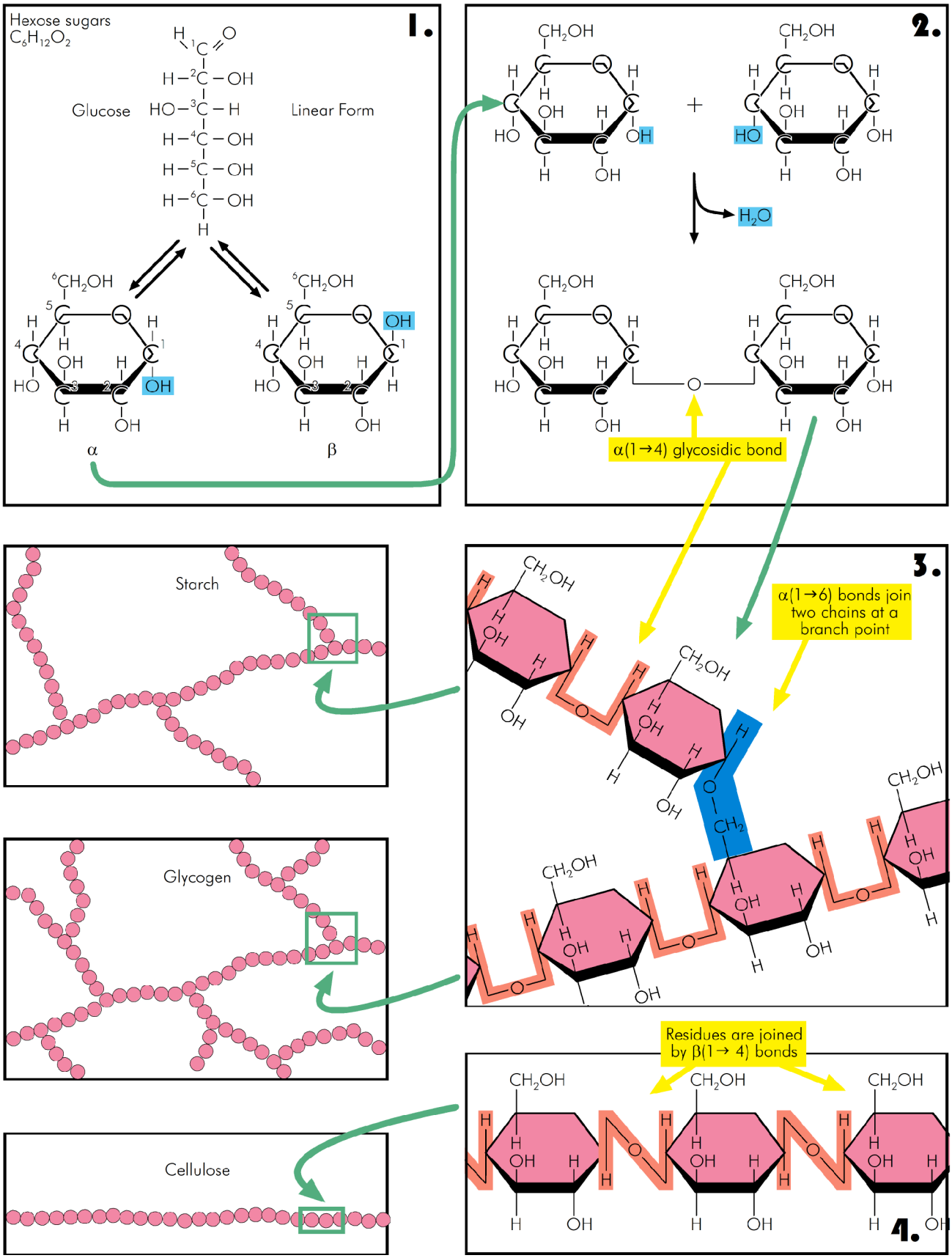


Figure 10: Structure of (poly)saccharides
1. Structure of a six-carbon sugar; 2. Formation of a glycosidic bond; 3.-4. Structure of polysaccharides (redrawn based on Cooper and Hausman 2007 [CoHa07, pp.45–46])

The principle form of energy-storage in cells is the six-carbon (or hexose) sugar *glucose*. It is

seen in Figures 10.1/2, where it is the basic building block of polysaccharides. These polysaccharides show a cycle-like structure, because all sugars with more than four carbons tend to create intra-connected molecule structures. In these ring-like forms the configuration of the 1st carbon in a monosaccharide (which is more precisely the position of the H and OH atoms) defines one of two alternative forms: the *α- or β-configuration* (Figure 10.1). The single sugars of an oligo- or polysaccharide are joined together by a dehydration reaction, by which a H_2O molecule is removed and both saccharides are connected by a glycosidic bond (Figure 10.2).

In animal cells, the storage form of carbohydrates is called *glycogen*. It contains only sugars in α-configuration. The corresponding structure in plant cells is *starch*. Opposed to these two polysaccharides, the wall of the plant cell is composed of extremely robust structures (Section 2.1.3.2), the *cellulose*, which features the β-configuration. In Figure 10.3-4 the structural effect of this difference is illustrated. Glycogen and starch may be connected by an *α(1→4)* or, less frequently, *α(1→6)* linkage (Figure 10.3). In contrast to this, in cellulose only exists the so-called *β(1→4)* linkages. This means that two monosaccharides are connected by the 1st residue of one monosaccharide and the 4th residue of the opposing one (Figure 10.4). This leads to net-like structures with irregular meshes. Obviously, these meshes in case of glycogen and starch and these fibers in case of the cellulose are important building blocks of the cell.

But the saccharides are also important for inter- and intracellular processes. Oligosaccharides are often connected to proteins and function as a marker for target proteins. These transport proteins may transport the protein-oligosaccharide complex to its destination on the cell surface or to other cell components, where they are inserted e.g. into the membrane. There, oligo- as well as polysaccharides are involved in cell-cell recognition, interaction and adhesion processes (Section 2.1.3.1). In addition, oligosaccharides are important for the folding of proteins. But sugars are not only found in proteins: in glycolipids, for example, a monosaccharide forms the headgroup (Section 2.2.1.3) and in nucleic acids they link the bases [CoHa07].

2.2.1.2 Nucleic Acids

While the structure of glycogen, starch and cellulose are composed of six-carbon glucose residues, the structures of *nucleic acids* are based on five-carbon (or pentose) sugars. In the case of *deoxyribonucleic acid (DNA)*, this is the *2´-deoxyribose*, and in the case of *ribonucleic acid (RNA)*, it is the *ribose*, respectively. These sugars are linked to bases, forming the *nucleoside*. In the case of DNA, all nucleosides are in β-configuration. After the sugar was undergoing phosphorylation – which means, that phosphate was added to the molecule – the resulting molecule structure is called the *nucleotide*. And these nucleotides are the main building blocks of nucleic acids. While *oligonucleotides* contain only a few of these nucleotides, the cellular nucleic acids contain thousands to millions of them. Therefore they are called the *polynucleotides*.

The phosphates bound to the sugars are the connecting elements of the nucleic acid chain: phosphodiester bonds between the 5´phosphate of one nucleotide and the 3´hydroxyl of

another one are created. The succession of these two molecules, the phosphate and the hydroxyl group, predefines the synthesis and reading direction of the polynucleotides: always starting at the 5´ and finishing at the 3´ end.

The DNA and RNA contain four different base pairs: two *purines* – *adenine* and *guanine* – and two *pyrimidines*. While DNA and RNA contain both *cytosine*, DNA contains as the fourth base *thymine* and RNA *uracil*, respectively. The information enclosed in the nucleic acids is encoded in the combination of these different bases.

In contrast to RNA, DNA contains two oppositely-running nucleotide chains. The bases are located on the inner side of the DNA and are interconnected to its counterparts by *hydrogen bonds* (Figure 11). The latter are strong interactions between a weak acidic donor and a weak basic acceptor (Section 2.2.1.4). The counterpart for guanine is cytosine and for adenine it is thymine or, in the case of RNA, uracil. The big advantage of this mirrored nucleotide chain is the ability of DNA to use one strand as a template for self-replication.

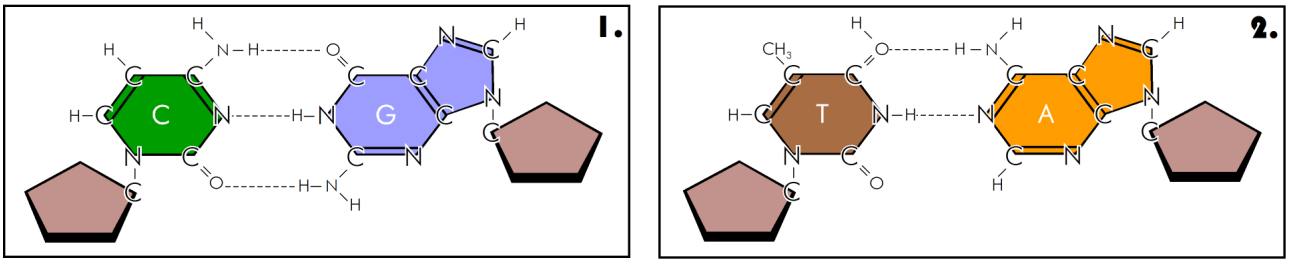


Figure 11: Complementary pairing between nucleic acid bases
1. The bases: C; cytosine, G: guanine; 2. T: thymine; a: adenine; and at each side: sugars (redrawn based on Cooper and Hausman 2007 [CoHa07, p.51])

The DNA is located in the nucleus and contains the major part of the eukaryotic genetic information (Section 2.1.3.4). This DNA is used as a template to synthesize RNA. This process is called *transcription*.

Different kinds of RNA are partly involved in protein synthesis (Section 2.2.1.4). An important part of this process – RNA to protein – is called *translation*. The *messenger RNA (mRNA)* delivers information from the DNA to different locations in the cell, e.g. the ribosomes (Section 2.1.3.7), where it is used as a template for protein synthesis. Also the *ribosomal RNA (rRNA)* and *transfer RNA (tRNA)* are involved in the protein synthesis. Other kinds of RNA are part of transport processes.

But also single nucleotides detached from polynucleotides play an important role in the cellular context. They are indispensable for many metabolic processes: the *adenosine 5´-triphosphate* (*ATP*, see Section 2.3.3), for example, is the basic chemical energy for all metabolic processes and the cyclic *adenosine/guanosine monophosphates (cAMP/cGMP)* have important cellular signaling functions in conjunction with hormones [CoHa07, VoVP06].

2.2.1.3 Lipids, van der Waals and covalent Forces

Lipids are very important for the cellular structure because they are a main component of the biological membrane. Figure 13 shows the different lipid types found in membranes (Section 2.2.3).

The simplest form of a lipid is a *fatty acid* (Figure 13.2). These are *hydrocarbon chains*, containing 10 to 24 carbon atoms, where the even atom number results from their synthesis from the precursor acetyl-coenzyme A. One end of this chain is terminated by a *carboxyl group* (COO-), the *C-terminus*. On the other end the alkyl group CH_3 (also abbreviated with *R1* and *R2* – with "R" for residue – in Figure 13) is localized. The *hydrophobic* (water-repulsing) nature of the hydrocarbon chains is caused by the nonpolarity of the C—H bonds: these chains tend to avoid the energetically unfavorable contact with water. *(Triacyl-)glycerol* (Figure 13.1) is the typical manifestation of fatty acids in the cytosolic environment. Three fatty acids are bound by a glycerol molecule. Because they are water-insoluble, they tend to form vesicle-like structures in the cytosol, which can be broken down for energy-yielding usage in metabolic processes [CoHa07, Luck08].

The hydrocarbon chains do not only differ according to their length. The structure is also an important indicator for the state and function of a lipid. Figure 12.1 shows a regular hydrocarbon chain in the *all-trans* state, a *stearic acid*. To point out the degree of *unsaturation*, the notation C18:0 can be used for stearic acid[12]. While C is the regular abbreviation for carbon, the first number indicates the chain length and the second one the number of double bonds. The double bond found in the *oleic acid* in Figure 12.2 causes a structural kink. It usually occurs in mammalian cell membranes between the 9^{th} and 10^{th} carbon. This form is called the *cis* form. It is the regular chain structure in biological membranes but occurs normally on only one of two hydrocarbon chains. For oleic acid, the term cis-9-C18:1 is used, indicating a saturation of 1 and the term in parenthesis shows, at which carbon the kink is found. In opposition to that the *trans* double bond in *elaidic acid*, or trans-9-C18:1, does not change the structure at all but is quite rare in the membrane context (Figure 12.3). It is obvious that the structure of the hydrocarbon chains directly affects the grade of membrane perturbation [Yeag87]. But also in other context the structure of fatty acids may have broad implications. For example, there are assumptions that double-bond-containing fatty acids like elaidic and *vaccanic* (trans-11-C18:1) play crucial roles in cardiovascular diseases (see also Section 6.3.2.1) [DDGL10].

The major component of most biological membranes is the *(glycero-)phospholipid* (Figure 13.5-9). This lipid consist of two hydrocarbon chains bound to a glycerol instead of the third hydrocarbon chain in the triacyl-glycerol, one phosphate group. This *phosphatidic acid* (Figure 13.5) is normally bound to another polar and eponymous headgroup: *ethanolamine* (Figure 13.6), *choline* (Figure 13.7), *serine* (Figure 13.8) or *inositol* (Figure 13.9). The *sphingolipid sphingomyelin* is also a phospholipid, but it contains a serine instead of a glycerol (Figure 13.10). The other components are the same as in phosphatidylcholine (Figure 13.7). Because of the polar headgroup and the nonpolar hydrocarbon chains, phospholipids have an *amphipathic* character which is vital for membrane formation (Section 2.2.3). Therefore, the headgroup tends to be in contact with water or the cytosol, while the tail is normally shielded from water [CoHa07, Luck08].

Still, two other important lipid types have to be discussed: glycolipids and cholesterol.

12 See Table in Appendix 9.13 for the comparison between the common names and unsaturation notation.

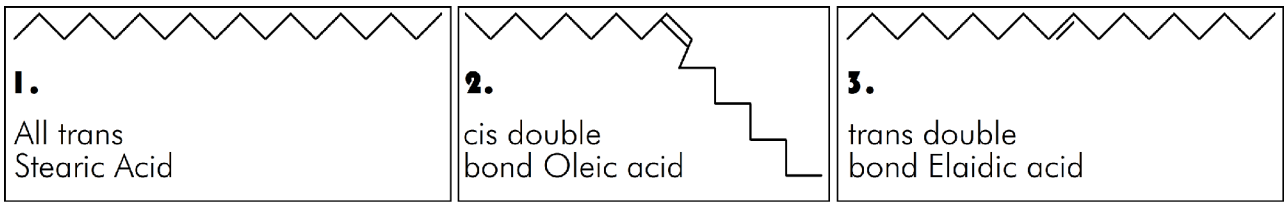


Figure 12: Forms of fatty acids
(redrawn based on Yeagle 1987 [Yeag87, p.29])

Glycolipids contain a carbohydrate (here glucose) bound to the structure known from sphingomyelin, consisting of two hydrocarbon chains bound to a serine (Figure 13.3). Finally, *cholesterol* is a lipid which is structurally significantly different from the other lipids (Figure 13.4). Moreover, properly speaking, the cholesterol is not a lipid but a sterol from a chemical perspective. Four hydrocarbon rings are joined to a hydroxyl group (HO). While the tail is strongly hydrophobic, the headgroup is weakly *hydrophilic* (water-loving). Although this lipid is therefore also amphipathic, its nature is more hydrophobic than those of the other lipids [CoHa07]. For this reason the structurally small *cholesterol* is known to be shielded by the headgroups of the larger phospholipids. This behavior is known as the *umbrella model* (see also Section 6.2.1.2) [HuFe99].

Additionally there are some minor lipid types which can be found in membranes. Free fatty acids are normally only found in small portions of overall lipids. Other components may be *lysophospholipids*, *polyisoprenoid lipids*, *diphosphatidyl glycerols, monoacyl-* and *diacylglycerides* [Genn89, Luck08]. All in all, a large variety of lipids are in existence. Yetukuri et al. proposed an estimated theoretical number of 180,000 different molecular lipid types, where only 9,600 belong to the group of phospholipids [YEVO08].

Besides energy-storage as in the case of glycerols, lipids also play other important roles outside of membranes. Derivatives of cholesterol are used as steroid hormones such as testosterone and derivatives of phospholipids are important for intra-cellular as well as inter-cellular signaling (Cooper and Hausman 2007).

Finally, the question remains how the different sizes of the spheres already shown in Figure 13 are defined. For this purpose, two terms should be explained which will be used throughout this work. The example here is based on water, H_2O:

- *Covalent Distance:* the distance of closest approach between two bonded atoms (e.g. between O and H: 0.958 Å),
- *Van der Waals Distance*: the distance of closest approach between two non-bonded atoms (e.g. between O and H: 2.72 Å).

Based on this distances, the radii are computed which are used, for example, to define the size of the spheres representing the different atom types in Figure 13:

- *Covalent Radius*: half of the bond length between two similar atoms (e.g. for O: 0.66±0.02 Å and for H: 0.31±0.05 Å),
- *Van der Waals Radius*: the radius limiting the *non-covalent, van der Waals forces* (e.g. H atom: 1.2 Å and O atom: 1.52 Å) [VoVP06, Wint12].

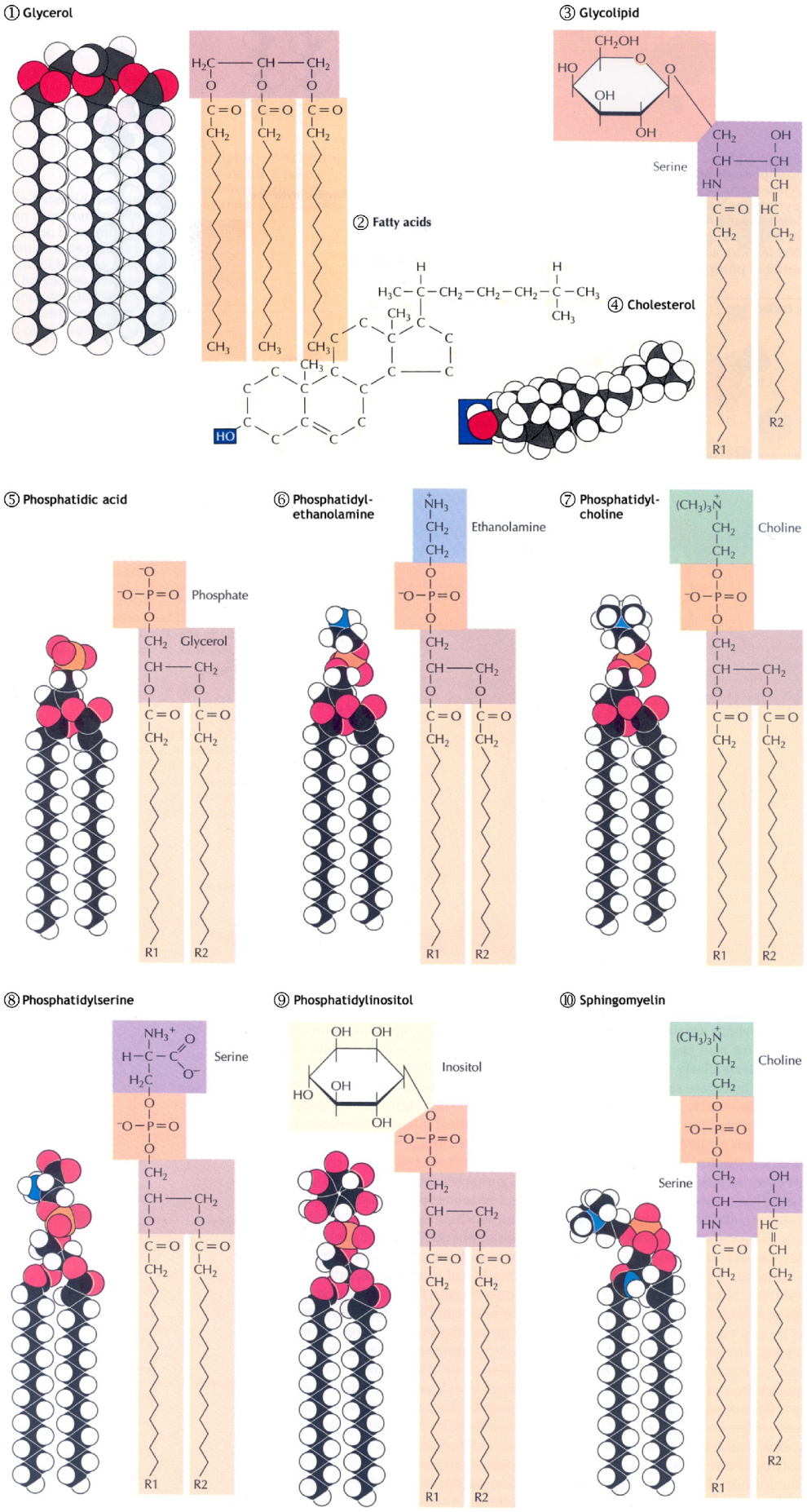


Figure 13: Lipid types
(Courtesy of/Copyright © by Cooper 2007. Reprinted with permission from [CoHa07, pp.47–49])

Based on this information it can be seen that Figure 13 uses covalent radii to visualize the atoms of the lipids.

2.2.1.4 Amino Acids and Proteins

As already mentioned in the previous sections, *proteins* play an important role in the cellular environment. Already the name protein (Gk.: proteios, of the first rank) includes its high priority in the cellular context. Each cell contains thousands of characteristic proteins which are deduced from the mRNA sequence. And each protein has a special task, for example: molecular transport, filtering and storage, information traversal and structural stabilization. A special class of proteins, the enzymes, are catalyzing chemical reaction and are thus relevant for metabolic processes (Subchapter 2.3).

The structure of the protein – from an abstract point of view – is subdivided into four levels: the primary, secondary, tertiary and quaternary structure.

The *primary structure* is predominated by the linear composition of *amino acids*. There are 20 different types of amino acids. Each of them is based on the following molecular structure: a carbon atom (C_α), which is bonded to a carboxyl group (COO-), an amino group (NH_3^+) and a hydrogen atom. But the specific chemical properties of the amino acid are defined by the side chain. These properties are often used to subdivide the amino acids into different groups, for example, into the *nonpolar*, *polar*, *basic* and *acidic group*.

All amino acids are joined together by *peptide bonding* between the (α) amino group of the first and the (α) hydroxyl group of the second amino acid (Figure 14, Structure of amino acids). These interconnected chains may consist of hundreds or even thousands of amino acids. They are called *polypeptides*[13]. Each of these chains has two distinct ends: one terminating in the α amino group (the amino terminus or N terminus) and the α carboxyl group (the carboxy terminus or C terminus). All polypeptides are synthesized from the N to the C terminus. This direction is defined by the word *downstream*. Analyzing a polypeptide from the C terminus to the N terminus is called *upstream*.

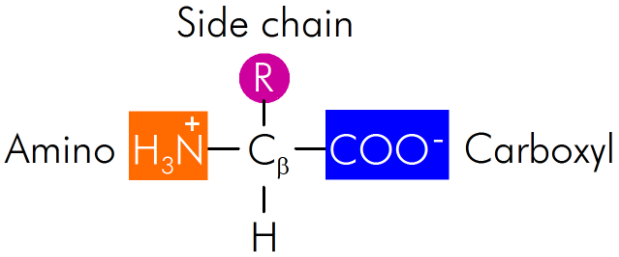


Figure 14: Structure of amino acids
(redrawn based on Cooper and Hausman 2007 [CoHa07, p.53])

As discussed before, the primary structure defines the one-dimensional chaining of the amino acid sequence. The *secondary structure* defines the formation of two-dimensional-similar shapes[14], created by hydrogen bonds between the CO and NH groups of peptide

13 A peptide is a polypeptide with less than 40 (amino acid) residues.
14 In reality the secondary structure has also an impact to the three-dimensional structure of the protein. But for the explanation of the theoretical concept of the four proteomic structure types it is defined here as similar to 2D.

bonds, without the involvement of the amino acids' side chains. Two structures are predominant in proteins: the *α helix* and the *β sheet* (Figure 15) [PaCB51]. The α helices are created by the interconnection between a CO and the NH group lying four residues downstream. A tightly coiled, spring-like structure is created by this characteristic. The rotation of the linear chain can run clockwise or counter-clockwise. But in proteomic context, all α helices were found to run clockwise, because this rotation was proven to be energetically more favorable. The β (pleated) sheet consists of two or more longer segments of continuous polypeptide chains which are connected by hydrogen bonds between two amino acids approximately 3,5 Å apart from each other (in contrast to ~1,5 Å in the α helix). They can run parallel or antiparallel. Other important secondary structures accompany chain reversals. One is called *β turn* (also: reverse turn or hairpin turn) which is often stabilized by a hydrogen bonding between the CO group and the NH group four residues upstream. Another reversal structure is the *loop* (also: Ω loop). In contrast to the helices and sheets, the turn and loop structures are normally found on the outer side of proteins and show irregular structures. Another important secondary structure is the combination of two to three long α helices wrapping around each other: the *coiled-coils*. They form long and stable fibers in the proteins α-keratin and collagen.

While the former two abstract structure definitions are theoretical formations, the *tertiary structure* now defines the spatial, three-dimensional structure of a protein. The process creating this spatial adjustment is called folding. It is defined by the combination of different domains. Proteins may contain only one domain, like the ribonuclease (Figure 15) or many different domains. All these domains possess special functions which can be relocated throughout different protein types.

The creation of the tertiary structure is guided by the interplay of the hydrophilic and hydrophobic amino acids. The polar, basic and acidic amino acids are placed outside the protein, forming their polar side chains – they are hydrophilic. In the case of the polar amino acid group, these are: *serine*, *threonine*, *tyrosine*, *asparagine* and *glutamine*. In opposition, to that, the nonpolar amino acids are located inside the protein, because they are hydrophobic. For example, the nonpolar amino acid *cysteine* is very important for intra-proteomic connections. It contains sulfur atoms which are able to form sulfhydryl groups together with hydrogen atoms. These sulfhydryl groups can form disulfide bonds between the side chains of two cysteines, creating the backbone of the intra-proteomic, secondary structure. The different secondary structures like the α helix and the β sheet are interconnected by the loop regions whose polar elements of the peptide bonds are either hydrogen-bonding with the outer polar side chains mentioned before or with the water surrounding the protein (Figure 15).

Previously the protein was expected to form a stable structure after finishing the folding process. But today proteins are known which are unstable, commonly known as *intrinsically disordered proteins (IDPs)* [OrOv11]. For these proteins it is not possible to determine a stable 3D structure and they have evolutionary as well as medical implications [DaES10, TFOS09]. In this context it is important to state that it is not possible to predict the three-

dimensional structure of all proteins by just examining the amino acid sequence.

Finally, there is also a *quaternary structure*. But it does not represent another dimension of the protein which could be expected by the former three theoretical composition types. It defines the modular composition on a higher level: the interaction and connection between different polypeptide chains. For example, hemoglobin contains four polypeptide chains which are all connected by the same interactions maintaining the tertiary structure [AJLR02, BeTS06, CoHa07, Groß11].

Figure 15: Secondary and tertiary structure of ribonuclease (1PYL.pdb in Jmol)

2.2.1.5 Small Molecules

The preceding four classes of cellular molecules cover the basic chemistry of most cells and they are all organic (carbon-containing) molecules. Of course, there is also a large number of small inorganic as well as organic molecules found in the cell, but most of them are not directly relevant in the context of this work.

One of the most important inorganic constituents is water which was already mentioned in the context of lipids and proteins. It accounts for approximately 70% of cell mass and interacts steadily with most of the aforementioned molecules as well as with inorganic ions which apply only to approximately 1% of the cell mass. For example, phosphate (HPO_4^{2-}), sodium (Na^+) and potassium (K^+) belong to the group of inorganic ions [CoHa07].

2.2.2 Membrane Model History

Centuries of development lead to the currently widely-accepted fluid mosaic membrane model [SiNi72]. Here, a short overview of the historical achievements is provided. It is shown that different models were developed during the last few centuries, and some of them were redefined or completely discarded after decades of acceptance, or: they have been ignored, although they were closer to reality than competing models. These are facts which are important to know and should be always taken into account when discussing models of the past, the present as well as future models.

The relevance of membrane-related theories was already indicated in the year 1774, when the famous American statesman Benjamin Franklin published observations on the wave-calming effect of oil monolayers [FrBF74]. This correspondence-based publication discussed different experiments with practical implications for dives and maritime traffic. Based on his observations, many different experiments were done in the following decades. In 1890, Lord Rayleigh a.k.a. John William Strutt repeated Franklin's experiment and he was able to determine the size of an oil layer in different experiments ranging between 0,81 and 2,65 nm [Lord90].

Moritz Traube, a wine merchant and science hobbyist, made a number of important scientific achievements during the 19th century. One of them was the first synthesis of an

artificial, semi-permeable membrane in 1864 [Pfef77, p.3], by combining water, tannic acid and droplets of boiled glue: the so-called “Niederschlagsmembranen” [Trau75, p.57]. This membranes played an important role for another pioneer in this area, Wilhelm Friedrich Philipp Pfeffer [Pfef77, p.V]. He used this membranes for the first measurements of osmotic pressures and additionally introduced the term *Plasmamembran* (German, plasma membrane) [Pfef77].

In 1899 Charles Ernest Overton and Hans Meyer independently developed the later so-called *Meyer-Overton hypothesis*. It roughly suggested that anesthesia occurs, if the amount of anesthetic agents which passed the cell membrane is sufficient [Meye99, Over99]. Although this theory with long-standing relevance is now outdated [CaMF03, Cant01, NoCa97], it was accompanied by a number of new findings which are inseparable from today´s cytology. It was Overton who verified some important membrane features, e.g. the selective permeability and the similarity between biological membranes and the fatty liquids, which was first proposed by Georg Hermann Quincke [Quin88]. Additionally he assumed that a lipid-mix of cholesterol and phospholipids and passive osmotic transport may be extended by an active one [Klei97].

Pfeffer's colleague Alexander Nathansohn combined the theories of Pfeffer and Overton and proposed a filter-like mosaic-concept, assuming a cholesterol and phospholipid containing layer surrounding “lebende Protoplasmatheilchen” (German, living protoplasm particles) [Nath04, p.643], today better known as proteins [Klei97]. Perhaps the fluid mosaic membrane model could have been developed decades earlier if the scientific community would have followed these first approaches in those early days.

The investigation of the structural properties of the cell membrane layer started in 1925 with Gorter and Grendel. They proved the existence of a bilayer in mammalian red blood cells (erythrocytes) with a simple experiment: They extracted lipids from a known number of cells with a known surface area. These lipids spread out at an air-water interface, showing that the self-assembling monolayer was twice as large as the surface area of the blood cells [CoHa07, GoGr25, Luck08]. In the same year, *Hugo Fricke* published first investigations concerning the thickness of the cell membrane by measuring its electric resistance and capacity [Fric25].

In 1935, Hugh Davson and James Danielli introduced the first concept about the structural involvement of proteins. They explicitly refused a mosaic-like model with a mix of lipids and proteins [DaDa35, p.504], which once was hypothesizes by Nathansohn. The *Davson-Danielli model* rather supposed a protein film to each side of the lipid bilayer.

In the following years there was a discussion about the observed number of lipid bilayers and protein layers surrounding the cell membrane accompanied by the invention of Electron Microscopy [Luck08]. Based on this new developments, J. David Robertson discussed his concept of the *unit membrane model* at the Symposium of the Biochemical Society in London 1957, pointing out that the biological membrane consists exactly of one lipid bilayer [Robe59].

Then the first electron microscopic images using the freeze fracture technique appeared (Section 3.1.1.2), showing bumps inside the lipid layers [Luck08]. This holes were interpreted as the left-overs of proteins and lead to the first models inserting proteins as globular repeating units into the lipid bilayer [GrTz66]. It was shown that the inner mitochondrial membrane can be broken up into several lipoprotein subunits and afterwards reassembled in order to regain proteomic activity. The fluid, reconstitutable and electro-resistive nature of lipid membranes was already shown a few years earlier [MRTW62]. Because the bilayer appeared black during examination with a wide field light microscope, the term *black lipid membrane (BLM)* was established during the sixties [TiCD66]. Today the acronym BLM is also used as the abbreviation for the *bimolecular lipid membrane* going back to the Davson-Danielli model [TiDi67]. Therefore, BLM can be seen as the direct predecessor model of the currently widely accepted *fluid mosaic model* introduced 1972 by Seymour J. Singer and Garth L. Nicolson [SiNi72]. Very important for their approach was the work of Frye and Edidin, who experimented with cell fusions and discovered first aspects about membrane fluidity around 1970 [FrEd70].

During the last decades, many small extensions to this model has been introduced. However, the major principles are unchanged (see also the following section). In addition, another important work has to be mentioned: Kai Simons and Elina Ikonen introduced *rafts* to the fluid mosaic concept [SiIk97]. This principle was not excluded by the original model of Singer and Nicolson, but 1997 was the first time this topic was focused on by the scientific community (Section 2.2.5).

2.2.3 The Fluid Mosaic Model

As mentioned before, in 1972 Singer and Nicolson introduced the membrane model which is widely accepted today [SiNi72]. It mainly consists of lipids and proteins which are steadily moving inside a fluid, two-dimensional environment. Although the original model has steadily improved, the major properties remained the same throughout the years. Today it is known that membranes play also a critical role in metabolism (Subchapter 2.3) as pacemakers: The fatty acid composition of a membrane affects the activity of membrane-intrinsic proteins and the permeability of Na^+ and K^+ ions [HuEl99, SBEG10].

The typical biological membrane is a *bilayer*. Two layers of lipids to each side of a hydrophobic core separate the external from internal space of a cell, a vesicle or other cell components. These biological barriers are inevitable to partly enclose and separate biochemical processes from the environment, to transport enclosed molecules from one destination to another or to secure cells from viral attacks. *Monolayers* within the cell also exist: *micelles*. These are small aggregates of lipids with their tails inward. A number of lipids enclosed by water tend to form a micelle in order to prevent the energetic unfavorable contact of the lipid tails with the solvent [AJLR02, Luck08].

A selection of fundamentals of the fluid mosaic model is summed up here [Jain88]:

1. Green and Tzagoloff already mentioned in 1966, that “apparently lipid has the effect of excluding three-dimensional stacking or association, and restricting the association of nesting particles to one plane” [GrTz66, p.597]. This *two-dimensional nature* is caused by the hydrophobic nature of the lipid's acyl chains and the protein's side-chains of the nonpolar amino acid residues (Sections 2.2.1.3, 3.2.4, 5.2.2.5) [Jain88].
2. *Non-covalent forces* (Section 2.2.1.3) prevent the membrane components from drifting apart. The polar head regions are dominated by hydrogen bonding and electrostatic interactions. The energy that stabilizes the bilayer is mainly produced by the hydrophobic effect [Jain88].
3. The *bilayer asymmetry* is an existential property for its structure and function. Therefore the composition of the two membrane layers always differs. This is a logical consequence of the two differing aqueous environments to each membrane side. Another reason for the compositional disequilibrium of the two layers is the fact that a transbilayer movement of amphipathic solutes is energetically unfavorable because polar and apolar groups and regions come into contact.
4. *Selective orientation and segregation* of membrane components relative to its layer side may be caused by specific interactions.
5. The biological bilayer is not a static material in a frozen state. Its components are steadily undergoing motions. This is the reason why the mosaic model was extended by Singer and Nicolson to be *fluid*. The *lateral diffusion coefficient* – defining the frequency lipids are interchanging their position – lies between 10^{-13} to 10^{-8} $cm^2 \cdot sec^{-1}$. An increasing disorder towards the membrane center is caused by the segmental motion of acyl chains every 0.01-1 nsec. For lipids, every 0.1-100 nsec and for proteins, every 0.01-100 msec rotations occur parallel to the layer's plane.
6. Bilayers may contain areas of special component compositions. These *lipid rafts* or *microdomains* may contain distinct proteins or differing lipid compositions floating island-like through the surrounding membrane area [LiSi10]. These are important e.g. for the formation of caveolae (Section 2.2.5).
7. Bilayers undergo different *temperature-dependent phases*. The *lamellar liquid crystalline (or fluid) phase (L_α)* is a state of highly disordered characteristics for higher temperature regions: the hydrocarbon chains of the lipids are regularly in cis-state (Section 2.2.1.3). In opposition to that the *lamellar gel (or solid) phase (L_β)* is a more ordered state which is characteristic for lower temperature regions. In this phase the lipids tend to be in an all-trans state with the tails maximally extended. In between this two phases there is the *ripple phase (P_β)*, showing a wave-like surface structure. These phases affect the molecular movement, rotation and tilting, the membrane surface appearance, density and flexibility [Genn89].
8. There are *intrinsic* (also: *integral*) and *extrinsic (also: peripheral) proteins*, as will be discussed in the next Section 2.2.4 [Luck08].

9. There are three types of *lipid diffusion*: 1. *Rotational Diffusion:* A molecule rotates around its orthogonal axis fixed to its position to change the interaction with the neighboring lipids, 2. *Lateral Diffusion:* neighboring lipids change places using Brownian motion, 3. *Transverse Diffusion:* Lipids change the membrane side *(flip-flop)* [Luck08].
10. The *layer thickness* depends on the saturation of the lipid's acyl chains and on the number of carbons. Particularly cholesterol is known to increase the layer thickness, because it stabilizes the acyl chains of neighboring lipids (e.g. phospholipids) and enables them to maintain their most extended conformation [Luck08].

2.2.4 Proteins in Membranes

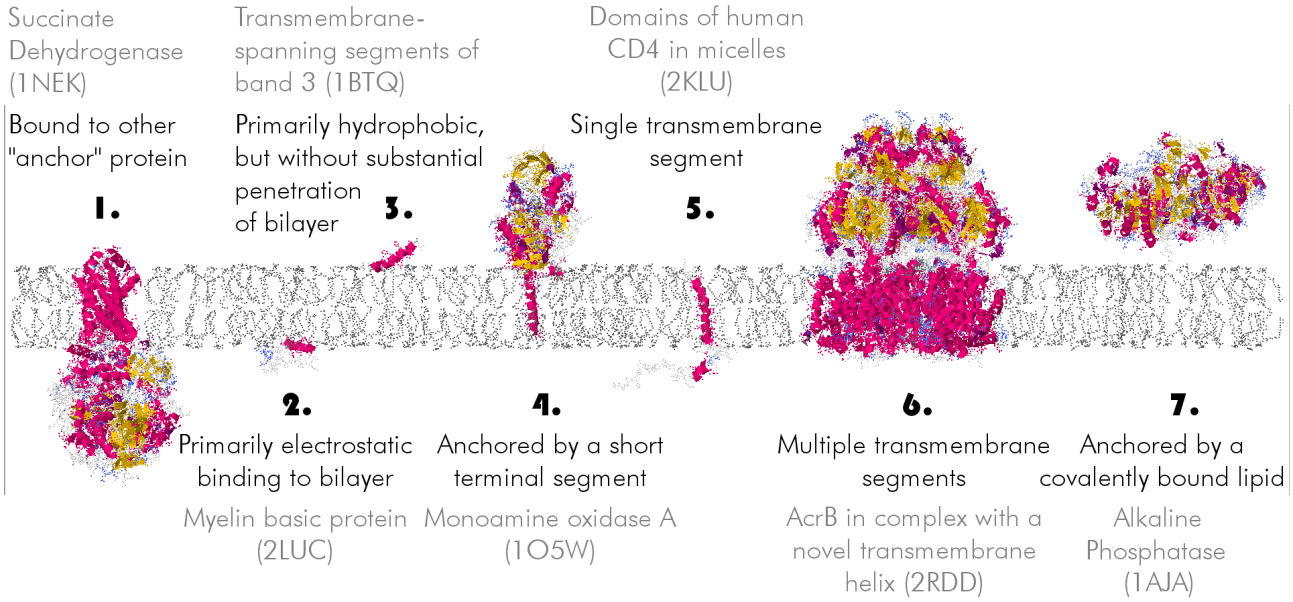


Figure 16: Attachment of membrane proteins (Redrawn based on [Genn89, p.86])

The major quantity of cellular proteins are known to be temporarily or permanently associated with membranes [LPJM12]. These membrane proteins are subdivided into different types. Two main classes of proteins are described by the Fluid Mosaic Model (Section 2.2.3). The *extrinsic* (also: *peripheral*) proteins are found outside of the membrane and can be easily removed by washes of the membrane (Figure 16.1-3,7). But the *intrinsic* (also: *integral*) proteins are located inside the membrane. They normally have one *transmembrane (TM)* peptide segment. To isolate them, the membrane has to be disrupted (Figure 16.4-6), because the natural distance between lipids and inserted proteins has to be very small in order to sustain the permeability barrier. There are three different kinds of intrinsic proteins: *monotopic* proteins are inside the membrane, but do not span it (Figure 16.4), *bitopic* ones have one TM span (Figure 16.5) and *polytopic* ones have at least two TM spans (Figure 16.6). Extrinsic proteins are protein-anchored or lipid-anchored proteins which are connected to the membrane by covalent bonds to protein or lipid groups (Figure 16.1/7). Since the introduction of the Davson-Danielli model it is known that some membranes are almost completely crowded with extrinsic proteins (Section 2.2.2). The predominant electrostatic and also the hydrophobic bonds are often weak or reversible

(Figure 16.2). Electrostatic bonds may emerge between anionic lipids and/or charged protein groups. *Cytochrome C*, for example, uses both electrostatic bonding types to connect to the membrane. It can connect to anionic lipids and to the acidic residues of Cytochrome bc1 and to *Cytochrome-C Oxidase*. There are also different ways for a protein to connect to a lipid: in the case of the cytoplasmic side of a plasma membrane to terpenes or acyl chains and in the case of the extracellular side to *GPI (glycosylphosphatidylinositol)*. Because of the aforementioned weak binding a protein is normally connected to multiple lipids.

In addition, there are dynamic proteins which cannot be assigned to one distinct class. They can be found separated from membranes in the cytosol, as well as extrinsic or intrinsic proteins. For example, the change from a water-soluble to the membrane-bound state of special proteins is relevant for their activity state and is also important for connecting the cytoskeleton to the plasma membrane.

There are various functions for membrane proteins. For example, some are involved in transport or recognition processes, signal transduction, acting as enzymes (Section 2.3.1) or they have membrane transforming properties as discussed in the following section [Genn89, Luck08, VoVP06].

2.2.5 Microdomains and Rafts

The bilayer is not always uniformly mixed. *Microdomains* are specific regions in membranes that are enriched in certain lipid and protein compositions floating like an island throughout the bilayer. A *Lipid rafts* is a special type of microdomain which contains a cholesterol and sphingolipid mix.

Lipid rafts are believed to interact with extrinsic proteins by GPI-anchoring. The raft-forming, monotopic protein *caveolin* is inserted into the plasma membrane by vesicles coming from the golgi apparatus. Because it is connected to the cytoskeleton, it stays in the inner membrane leaflet. It forms hetero-dimers with cholesterol. The surrounding lipid raft contains in addition to cholesterol, sphingolipids and glycolipids. The raft in combination with the caveolin forms small caves (called caveolae) which are relevant during the endocytosis [Luck08].

In addition, there are planar lipid rafts. They play an important role in the lateral heterogeneity of bilayer membranes [LiSi10]. There were different sizes reported for rafts. The minimal size was defined to be from ~100 Å up to hundreds of nanometers [Pike06, PKFS00, PMPH03]. Atshaves et al. presented the composition of a cholesterol-sphingomyelin raft found in a plasma membrane of mouse hepatocytes [AMPG07]. These findings will be applied in Section 6.2.5 (see also Table 22 for the lipid raft composition).

It is still disputed if lipid rafts also reside in mitochondrial membranes [McNW06, VoVP06, ZhBF09].

2.2.6 Membrane Compositions

The heterogeneity of cell membranes results in a large number of different membrane types. By combining different lipids types these membranes are optimized for their special

tasks. Specific patterns of lipid and protein distribution are found in different membrane types, depending on age, environment, diet and pathological conditions. These patterns are yet not fully understood and will be an important topic for future analyses for the understanding of biological membranes [Jain88].

Cell Component Membrane	Average Density (g/ml)	Dry Weight (%)		Lipid Composition (lipid %)								
		Lipid	Protein	Chol	PC	SM	PE	PI	PS	PG	DPG	PA
Plasma	1.15	30-50	50-70	20.0	64.0		17.0	11.0		2.0		
Rough ER	1.20	15-30	60-80	6.0	55.0	3.0	16.0	8.0	3.0			
Smooth ER	1.15	60	40	10.0	55.0	12.0	21.0	6.7			1.9	
Mito. Inner	1.19	20-25	70-80	<3.0	45.0	2.5.0	25.0	6.0	1.0	2.0	18.0	0.7
Mito. Outer	1.12	30-40	60-70	<5.0	50.0	5.0	23.0	13.0	2.0	2.5	3.5	1.3
Nuclear	1.25	15-40	60-80	10.0	55.0	3.0	20.0	7.0	3.0			1.0
Golgi	1.14	60	40	7.5	40.0	10.0	15.0	6.0	3.5			
Lysosome	1.20	20-25	70-80	14.0	25.0	24.0	13.0	7.0	7.0		5.0	

Table 2: Membrane compositions of different rat liver membranes [Jain88, p.23]

Intracellular Compartment		Percentage of total cell membrane
Plasma membrane		2
Rough ER membrane		35
Smooth ER membrane		16
Golgi apparatus membrane		7
Mitochondria	- Inner membrane - Outer membrane	7 32
Nucleus	- Inner membrane	0.2
Secretory vesicle membrane		-
Lysosome membrane		0.4
Peroxisome membrane		0.4
Endosome membrane		0.4

Table 3: Relative Amounts of Membrane Types in Hepatocyte [AJLR02]

Every single membrane is supposed to contain hundreds of different lipid types. There are different tasks expected from the heterogeneous lipid compositions, for example: stabilization of the bilayer in certain regions, the approaching of enzymes to support their activity, regulatory and metabolic functions, incorporation in communication processes and vitamin balance [LiSi10, Yeag05].

Table 2 includes percentaged protein and lipid distributions of different rat liver membranes. Comparing the values of the different lipid types, a few facts can be observed. The plasma membrane holds the largest number of cholesterol. This is due to the fact that the plasma membrane is known to contain lipid rafts, which have a large amount of cholesterol (Section 2.2.5). *Cardiolipin* (a diphosphatidyl glycerol) is an important lipid for the cardiolipin/K pathway. It consists of two interconnected phospholipids. CL supports the activity of the cytochrome-bc_1 enzyme complex which is normally located at the matrix side of the mitochondrion. Therefore the maximal percentages of cardiolipin are found in the inner membrane of the mitochondrion in Table 2 [Luck08, NeCL08]. In addition, carbohydrates are found in some membranes, in plasma membranes the amount is about 10% (see Section 2.2.1.1 for functionalities) [Yeag05].

Finally, in Table 3, the relative amount of the overall cellular membrane surface area of each cell component is shown. It is interesting to note that the cell membrane – which of course encloses all other cell components – has a relatively small proportion on the overall membranes. It is outperformed especially by the rough ER and the mitochondrion. The reason are the aforementioned folded structures of these cell components: the cisternae and the cristae respectively (Sections 2.1.3.6 and 2.1.3.10).

2.3 [FUN] Metabolism

Now the mesoscopic as well as the molecular level were introduced. Until this point, mainly the structural aspects of the cell have been discussed. But of course, the life of a cell does not depend only on the structure: there have to be biochemical processes creating and maintaining this structure.

In all subsequent [FUN]-sections, the functional quality of the cell will be discussed. There is a large number of processes located in or depending on the living cell. And the term *metabolism* summarizes all these chemical/biosynthetic reactions [AJLR02, VoVP06].

Even a sketchy discussion of all those processes lies beyond the scope of a thesis and there are enough publications on the book market trying to give an overview from different perspectives [AJLR02, CoHa07, NeCL08]. Therefore this work will mainly focus on a very small set of metabolic processes which will be discussed in the following sections.

2.3.1 Enzyme, Products and Substrates

Cellular processes depend on the energy supply. Energy is needed to catalyze reactions. Some chemical reactions occurring in a cellular environment would regularly need years to accomplish their specific tasks by reaching the so-called energy barrier. Because the life of a cell is limited, the involvement of special catalysts is inevitable: *enzymes*. The group of enzymes includes all RNA as well as proteins which are catalyzing reactions. Only a small number of biological reactions are catalyzed by RNA, the majority are driven by proteins. Enzymes do not alter the equilibrium state between the reactants and products and they are not consumed by a reaction. They are able to accelerate reactions dramatically, compressing the duration of some reactions to a few seconds by dramatically lowering the *energy*

barrier/activation energy, which defines the amount of energy needed to start a chemical reaction.

A reaction may be formulated in this manner:

$$S + E \leftrightharpoons ES \leftrightharpoons E + P \quad (1)$$

where S is the substrate, E is the processing enzyme and P the resulting product. The *substrate* is the substance changed by a reaction, whereas the substance resulting from a reaction is called the *product*. The bidirectional arrow used here represents the equilibrium state: the involved enzyme E is not changed during the reaction process. While reacting, the substrate docks to the enzyme at its active site. Different chemical non-covalent interactions may be involved: hydrogen bonds, ionic bonds and/or hydrophobic interactions. But often a number of substrates are involved in a single reaction, for example:

$$S_1 + S_2 + E \leftrightharpoons E\,S_1 S_2 \leftrightharpoons E + P \quad (2)$$

Here, two substrates are combined to one product by an enzyme.

The docking process in respect to the enzyme-substrate reaction features two different models:

- the *lock-and-key model* does not alter the configuration of the substrate(s) and the enzyme during the reaction, and
- the *induced-fit model* alters both, the configuration of the substrate(s) as well as the one of the enzyme. The structural change may additionally activate binding sites on other positions of the enzyme.

To describe the large variety of enzymes, the enzyme classification has been established. It contains six major classes:

- Oxireductases: driving oxidation-reduction reactions,
- Tranferases: transferring functional groups,
- Hydrolases: hydrolysis reactions,
- Lyases: group elimination to form double bonds,
- Isomerases: isomerization, and
- Ligases: bond formation coupled with ATP hydrolysis.

Another important class of substances involved in metabolic processes is composed of *coenzymes*. They are supporting enzymes during the catalysis by enabling their activity or by enhancing the reaction rate. Similar to the substrates, they bind to the active site of the enzyme. An appropriate example is the NAD^+ (nicotinamide adenine dinucleotide, oxidized form) coenzyme. By binding H^+ and two electrons e^- from a substrate, it forms NADH (nicotinamide adenine dinucleotide, reduced form). The bonded electrons can be transferred to another substrate forming NAD^+ again. This example shows that coenzymes may function as electron carriers between two substrates, inducing first an oxidation and then a reduction process. In addition, they may transfer various chemical groups, e.g. acyl, aldehyde, amino, carboxyl or one-carbon groups.

But the enzyme's activity is also changed by other regulation types involving small molecules or even other proteins. Different *allosteric regulations* (allo=other, steric=site) change the conformation of the enzyme by molecules docking to a site different from the catalytic site – in contrast to the coenzymes which bind to the active site. The change of the enzyme's conformation may be accompanied by the deformation of the active site's shape, manipulating the catalytic activity. The feedback inhibition, another allosteric regulation, causes a product to disable further reactions of its catalyzing enzyme after the reaction is finished. This is important if the preferred concentration is reached and the catalysis of new products should be prevented. The enzymatic activity is also often regulated by the docking of phosphor groups to the serine, threonine or tyrosine groups of enzymes (see also Section 2.2.1.4), the phosphorylation [CoHa07, VoVP06].

2.3.2 The Gain of Energy for Reactions

The aforementioned energy needed to initiate chemical reactions is not automatically present in the inner cell. It has to be acquired from the cell's environment. Most intracellular reactions are energetically unfavorable, which simply means, they need energy. This can be written as $\Delta G>0$. In this term, *G* symbolizes *Gibbs free energy*, a standard to measure the change in chemical energy. Two terms are very important in the intracellular-energetic context:

- *entropy* describes the degree of disorder or randomness of a chemical reaction, and
- *enthalpy* defines the heat released or absorbed by a chemical reaction.

The thermodynamic function describing the change in enthalpy and entropy is called ΔG. This *change in free-energy* is combined with the assumption of standard conditions, the so-called *NPT ensemble,* which means: the number of particles (N), the pressure (P) and the temperature (T) are at equilibrium state.

All chemical reactions proceed in the energetically favorable direction. Looking at the single reaction A⇋B, there are three possibilities:

- if $\Delta G=0$, means the reaction is at equilibrium and will not proceed,
- if $\Delta G<0$, A→B, means the reaction will proceed towards B, or
- if $\Delta G>0$, B→A, means the reaction will proceed towards A.

The free-energy for A⇋B change can also be written as follows:

$$\Delta G = \Delta G^\circ + RT \ln [B]/[A] \quad (3)$$

This term introduces a few new variables: *R* is the gas constant, *T* the absolute temperature, ΔG° *the standard free-energy change* and the brackets indicate the molar concentration of A and B. Therefore not only one single reaction is taken into account but many reactions depending on the concentration of the substrates A and B. This equation can be used to derive ΔG° by integrating the equilibrium constant $K=[B]/[A]$ at equilibrium state, $\Delta G=0$ respectively:

$$\Delta G^\circ = -RT \ln K \quad (4)$$

- If the actual ratio [B]/[A] is equal to the equilibration ratio *K*, then $\Delta G=0$ and no reaction will take place,
- if the actual ratio [B]/[A] is lower than the equilibration ratio *K*, then $\Delta G<0$ and the reaction proceeds in the forward direction (A→B), and
- if the actual ratio [B]/[A] is larger than the equilibration ratio *K*, then $\Delta G>0$ and the reaction proceeds in the reverse direction (B→A).

In summary, the standard free-energy change ΔG° of a reaction describes its chemical equilibrium and predicts the direction of the reaction under any possible condition and with respect to the concentration of the participating molecules.

To describe the cellular environment appropriately it is important to take the correct pH value into account. This requirement is met by $\Delta G^{\circ\prime}$, which defines the *standard free-energy change* of a reaction combined with a pH value of 7 and an aqueous solution.

To focus now on the initial problem of $\Delta G>0$ for most intracellular reactions, first a thermodynamic unfavorable reaction is shown here:

$$A \leftrightharpoons B, \text{ where } \Delta G = +5 \text{ kJ/mol} \quad (5)$$

This reaction needs energy to be catalyzed. It takes 5 kJ/mol. To initiate this reaction, the energy has to be acquired from another reaction:

$$C \leftrightharpoons D, \text{ where } \Delta G = -10 \text{ kJ/mol} \quad (6)$$

This reaction is energetically favorable: -10 kJ/mol.

By coupling both reactions, the energy generated by 6 is used to initiate the reaction 5 and finally -5 kJ/mol remains to be used in other reactions:

$$(5+6)\ A + C \leftrightharpoons B + D, \text{ where } \Delta G = -5 \text{ kJ/mol} \quad (7)$$

And this is exactly one of those processes guided by enzymes [CoHa07, VoVP06].

2.3.3 Biological Energy: ATP

After the explanation of the theoretical background for the gain of energy, the question remains, how is the energy is achieved in the cellular context. ATP (adenosine triphosphate, Section 2.2.1.2) is the primary energy carrier in biological cells. It contains high-energy bonds (which means $\sim\Delta G^{\circ\prime}<-25$ kJ/mol), which release their energy during hydrolysis:

$$ATP \rightarrow ADP + P_i \text{ with } \Delta G^{\circ\prime} = -30{,}5 \text{ kJ/mol} \quad (8)$$

In addition, ATP can be hydrolyzed to AMP and two phosphate groups from pyrophosphate:

$$ATP \rightarrow AMP + PP_i, \text{ with } \Delta G^{\circ\prime} = -45{,}6 \text{ kJ/mol} \quad (9)$$

directly followed by the reaction

$$PP_i \rightarrow P_i + P_i, \text{ with } \Delta G^{\circ\prime} = -19{,}2 \text{ kJ/mol}, \quad (10)$$

resulting in $\Delta G^{\circ\prime} = -64{,}8$ kJ/mol.

ATP is not the only energy carrier in the cell. Alternative energy is provided exemplarily by nucleoside triphosphates like *Guanosine Triphosphate (GTP)* [CoHa07, VoVP06].

2.3.4 Metabolic Pathways and Energy

It has now been theoretically described how energy is created. But how is this energy generated inside a living organism? Of course, this is done by the aforementioned metabolic pathways. The definition of a *metabolic pathway* is as follows:

"Metabolic pathways are series of connected enzymatic reactions that produce specific products." [VoVP06]

In this section a concrete example will be shown. To gain energy, glucose is broken down to CO_2 and H_2O in the following example. The complete reaction is written as follows:

$$C_6H_{12}O_6 + 6\,O_2 \rightarrow 6\,CO_2 + 6\,H_2O \quad (11)$$

Glycolysis is the initial pathway for the generation of energy in aerobic and anaerobic cells. While in anaerobic cells the glycolysis produces all metabolic energy needed for the organism, in aerobic cells it is only the first step in the energy generation process. The first reaction of the glycolysis, the generation of glucose-6-phosphate by combining glucose and phosphate, is energetically unfavorable:

$$\text{Glucose} + \text{Phosphate } (HPO_4^{2-}) \rightarrow \text{Glucose-6-phosphate} + H_2O, \text{ with } \Delta G^{\circ\prime} = +13{,}8 \text{ kJ/mol} \quad (12)$$

As discussed before, this reaction will not spontaneously begin into forward direction because energy is needed. This reaction is now coupled with the energy-yielding ATP hydrolysis with $\Delta G^{\circ\prime}$ = -30,5 kJ/mol:

$$ATP + H_2O \rightarrow ADP + HPO_4^{2-} \quad (13)$$

resulting in

$$(12+13)\ \text{Glucose} + ATP \rightarrow \text{Glucose-6-phosphate} + ADP, \text{ with } \Delta G^{\circ\prime} = +13{,}8 - 30{,}5 \text{ kJ/mol} = -16{,}7 \text{ kJ/mol} \quad (14)$$

This shows the relevance of ATP hydrolysis for energy-requiring reactions. On the contrary, there are many energy-yielding reactions initiating ATP synthesis. Also the glycolysis needs energy only during the first few reactions. The second part of the pathway which is leading to the final product, the glucose, generates energy [VoVP06].

2.3.5 The Citrate Cycle

The *citrate cycle* is the major source for energy needed by the animal cell, its central metabolic hub. It is also an important source for precursors needed for components like nucleotide bases and amino acids (Sections 2.2.1.2 and 2.2.1.4). But as can be seen in Table 4, it depends on other pathways. This simplified overview shows the net energy gain of the oxidation of glucose to citrate: two ATP, six NADH and two $FADH_2$ (flavin adenine dinucleotide, reduced form) molecules are generated[15]. They are listed in terms of the sequence of irreversible reactions.

15 Note in Table 4 that two pyruvate molecules are generated from one glucose molecule during glycolysis, therefore, two citrate cycles are invoked.

First, glycolysis generates pyruvate which is needed to initiate the citrate cycle. The glycolysis is located in the cytosol. Its product, the pyruvate, is transported by a specific protein through the inner mitochondrial membrane to the mitochondrial matrix. This transporter protein is needed, because the inner mitochondrial membrane is a barrier for most small molecules and ions. This property is very important for the *oxidative phosphorylation*, because it depends on the maintenance of the *proton gradient*. In contrast to this, the outer membrane is passable by most molecules, because it contains many proteins called porins allowing the free diffusion of many molecules. Therefore, the intermembrane space – which is located between the inner and outer membrane – shows many compositional similarities to the cytosol. The pyruvate is now oxidatively decarboxylized (the *oxidative decarboxylation of pyruvate*) by the pyruvate dehydrogenase complex, a large assemblage of three different proteins inside the mitochondrial matrix. The product of this process is the acetyl CoA which then enters the citrate cycle. The last-mentioned pathway is also located in the mitochondrial matrix and generates citrate during a number of reactions [BeTS06, CoHa07].

But these three reactions do not produce enough energy to enable life in a mammalian. For this purpose the oxidative phosphorylation provides a very complex energy-recycling process. As mentioned before, the ATP produced by different metabolic pathways is converted to ADP by simultaneously releasing energy. The oxidative phosphorylation reuses this ADP to extend the amount of ATP again. The aforementioned proton gradient across the inner membrane drives the phosphorylation of ADP. Focusing for instance on the glycolysis, the amount of produced ATP (Table 4) is extended by another 26 ATP molecules during the oxidative phosphorylation process [BeTS06].

Pathway	Initial Substrate	Major Product	Energy Produced	Localization
Glycolysis	Glucose	2 Pyruvate	2 X ATP 2 X NADH	Cytosol
Oxidative Decarboxylation of Pyruvate	Pyruvate	Acetyl CoA CO_2	2 X NADH	Mitochondrion (Matrix)
Citrate Cycle	Acetyl CoA	Citrate/ Oxaloacetate	1 X ATP 3 X NADH 1 X $FADH_2$	Mitochondrion (Matrix)

Table 4: Three citrate-cycle-depending pathways, their net energy gain and localization [BeTS06, p.475 ff., CoHa07, p.81 ff.]

2.4 [MES+MOL+FUN] Integrative Cytology?

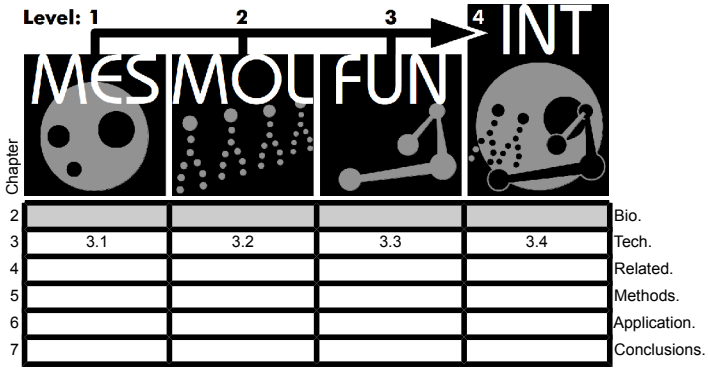
The three levels are introduced here as three different topics: the Mesoscopic and Molecular Composition respectively, and the Metabolism. But of course they cannot be differentiated in nature. Everything is interrelated: the atomic composition of the membrane changes the outer shape of the whole cell component, a metabolic reaction may trigger the beginning of the cell division or vice versa, the apoptosis. But all biology-related research areas usually tend to focus on highly specialized topics, excluding many aspects of interrelated fields. This is due to the fact that wet lab experiments are naturally narrowed to their specific applications and environments, and that today´s knowledge in cytology is still quite fragmentary.

This work will present an approach to generate a visual unity among these different research areas using visual modeling techniques. In addition, a suggestion will be discussed in the last chapters how Integrative Bioinformatics and Integrative Biology could be united in the future.

3 Technical Background

Now that the biological basics of cytology have been introduced, the question emerges in which way the cell has been examined during the last centuries. How was complex structural knowledge acquired? The answer was already partially anticipated in Subchapter 2.1 by mentioning the development of different microscopy techniques. In Section 3.1.1, a selection of relevant microscopy types are introduced, showing how the mesoscopic level is visualized. The visualization of the molecular level is covered in Section 3.2.1.

For the accessibility and analysis of data acquired by these visualization techniques, it is important to store and categorize it appropriately. Of course, the collection and analysis of a vast amount of data is one of the major tasks of the information age. The enormous number of experiments, the enhanced speed of gene analysis, the large size of microscopic data – these are all aspects databases have to deal with. As well as the ever-growing amount of data, the number of different databases approaches is continuously growing. In the latest *Nucleic Acids Research Database Issue (Nucleic Acids Res.)*, for example, 1380 actual life-science-related databases are found [GaFe12]. Of course, only a small subset of these databases is discussed here. However, the use of molecular databases was inevitable for this work. Section 3.1.2 discusses databases at the mesoscopic scale, Section 3.2.1 at the molecular level and Sections 3.3.2 at the functional level. An overview of all databases discussed here in conjunction with a comparison of the Nucleic Acids Res. categories is discussed in Section 3.4.2.



3.1 [MES] Cell Microscopy and Modeling

A short look at the different microscopic techniques will be provided beginning with the mesoscopic level (Section 3.1.1). After introducing the different approaches to generate concrete visualizations of the cellular structure and how this information can be stored and accessed appropriately (Section 3.1.2), a major topic of this thesis will be addressed: the modeling of cell components (Section 3.1.3).

3.1.1 Microscopy Types

Of course, it lies beyond the scope of this work to provide an exhaustive overview of microscopy techniques. But it is important to emphasize some of the major principles leading to the picture of the cell which the community has today.

3.1.1.1 Light Microscopy

The natural visual limitation of the microscope is twofold: the wavelength of visible light with ~4,000 Å and the light gathering power of the lens. Both factors affect the resolution of the microscope which is the smallest difference in distance which can be projected. For light microscopy the resolution limit is ~0.2 μm. This leads to the fact that most cell types are distinguishable, because their size lies approximately between 1 and 100 μm. Also larger cell components can be seen, such as a nucleus, a chloroplast or a mitochondrion. However, smaller components lie beyond the scope of the light microscope.

Light microscopy will be introduced and thereafter the sections showing relevant techniques to reach higher resolutions in microscopy.

The bright-field microscopy is the simplest version (Figure 17.1). The light beams are directed towards the specimen. While passing through it, light beams are absorbed depending on the contrasts within the specimen. If an untreated object is used, nearly nothing can be seen in the microscope due to the mostly colorless, transparent nature of cells and their components. Therefore, after fixation with substances like acetic acid, alcohol or formaldehyde, different dyes are used to stain cells. Depending on the dye, special components of the cell are emphasized. The disadvantages of this process are not only the low contrast of transparent cell components such as the nucleus but also the termination of the cell.

To overcome these obstacles, the *phase-contrast microscopy* (Figure 17.2) and the *differential interference-contrast microscopy* (Figure 17.3) were developed. Both approaches enrich the optically achieved image by adding contrast based on intra-cellular differences in density and thickness. Because these approaches do not rely on staining, it is possible to observe living cells, e.g. by using video-enhanced differential interference-contrast microscopy.

Another established method is the *fluorescence microscopy* (Figure 17.4). Fluorescent dyes are used to mark different regions of the cell, in living as well as in the fixated state. Probably the most famous method is by applying the *green fluorescent protein (GFP)* – which is extracted from the jellyfish – to a large variety of proteins. Therefore GFP, together with similar proteins red, yellow or blue colorings, is an important factor in the localization of proteins (see also Section 3.3.2.2), especially in living cells. Derived methods are used to track the movement of a protein *(fluorescence recovery after photobleaching, FRAP)* or the evaluation of the interaction between two proteins *(fluorescence resonance energy transfer, FRET)*. A disadvantage of this method is the blurriness which can be computationally improved by using image deconvolution filters (see also Section 3.1.3.1). There are also approaches to use fluorescence microscopy in conjunction with 3D structure determination and ultrafast imaging techniques achieving resolutions of up to 40 nm laterally [SHFP00].

Another method used to overcome the problem of blurriness is the *confocal microscopy* (Figure 18.1). While the regular fluorescence microscopy reflects the complete specimen at a point in time, the confocal microscopy scans it by using a concentrated light bundle, a laser beam. The latter is focused onto one special position.

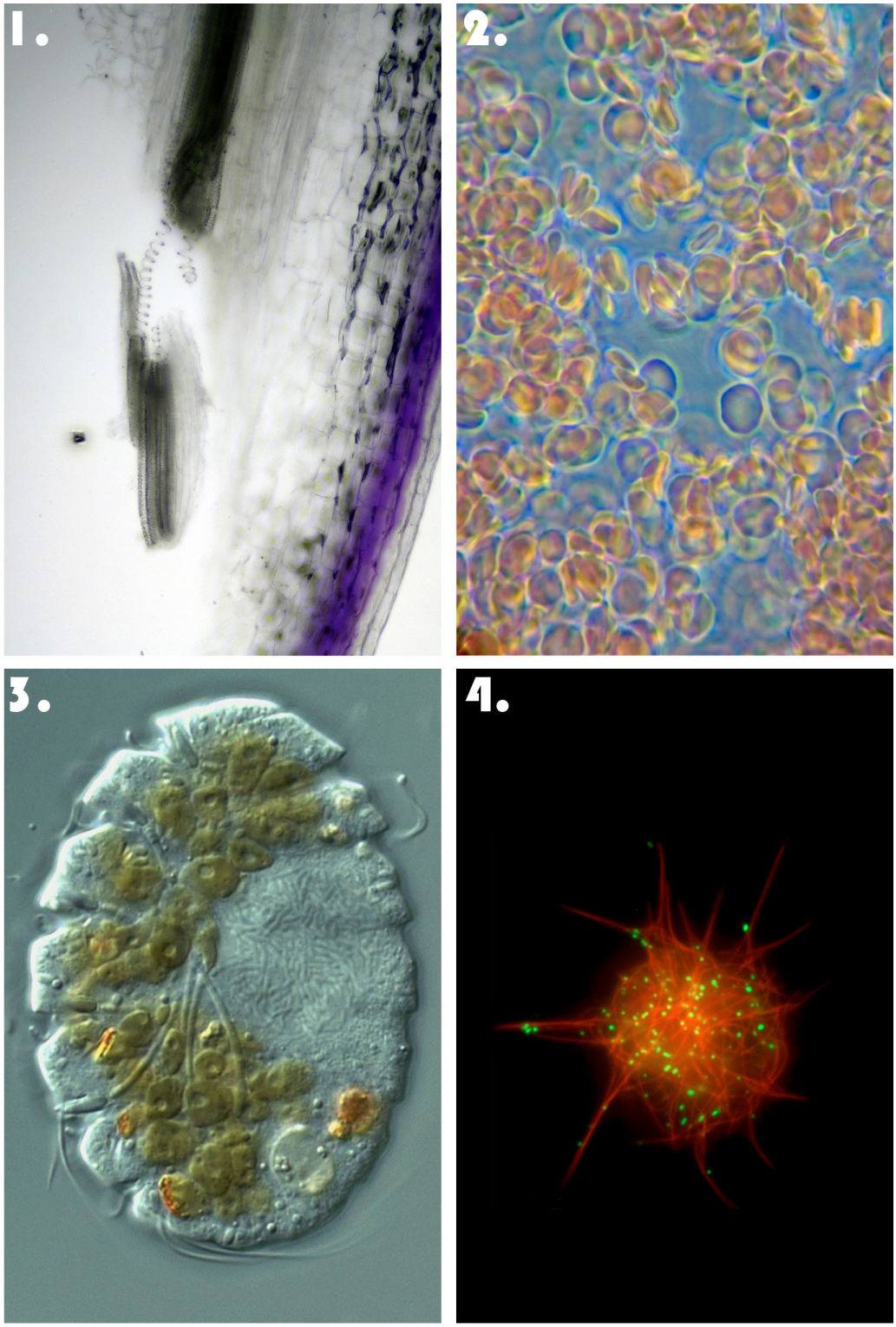


Figure 17: Light microscopy images
1. Bright field microscopy: a kale leaf, 100x (Courtesy of/Copyright © 2011 by Courtesy of Specious Reasons/flickr.com [Spec11]); 2. Phase-contrast Microscopy: human blood cells under oil immersion, 1200x magnification (Courtesy of/Copyright © 2006 by Jasper Nance and nebarnix/flickr.com. Reprinted based on CC [NaNe06]); 3. Differential interference-contrast microscopy: a phagotrophic dinoflagellate isolated from Chappaquoit beach (Courtesy of/Copyright © 2005 by Courtesy of Census of Marine Life E&O/flickr.com. Reprinted based on CC [Cens05]); 4. Fluorescence Microscopy: intracellular trafficking along the microtobules (red), cargo (green) (Courtesy of/Copyright © 2010 by The Journal of Cell Biology/Kim et al. 2007/flickr.com. Reprinted based on CC [KLRK07, Thej10])

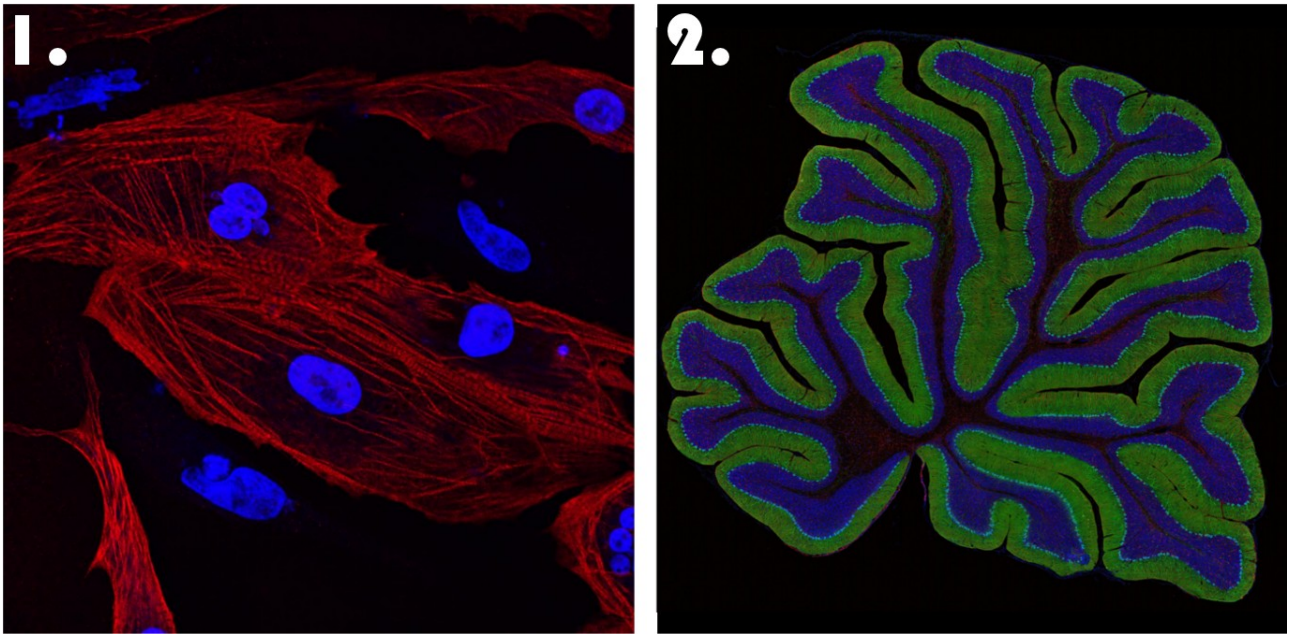


Figure 18: Confocal and Multi-photon excitation microscopy
1. Confocal Microscopy: Cardiomyocytes from chicken embryo heart showing nuclei (blue) and muscle fibres (red) (Courtesy of/Copyright © 2008 by arboreus/flickr.com. Reprinted based on CC [Arbo08]); 2. Multi-photon excitation microscopy: montage of a cerebellar cortex triple labeled for Hoescht stain (blue), IP3 receptor (green) and GFAP (red) (Courtesy of/Copyright © 2002-2012 by The Regents of the University of California. All Rights Reserved. CCDB-ID: 23 [MGWQ02])

In this way, a specimen can be scanned not only two-dimensionally line by line, but also three-dimensionally, by changing the focus to another layer. Again, computational methods are used to combine the different information achieved.

Of course, this method is not applicable to living cells. But again, there is an alternative: the *multi-photon excitation microscopy*, which is also able to scan tissues to a depth of 1 mm (Figure 18.2). While the former method needs a pinhole-like filter to process only the light which focuses, this method omits this filter by increasing the barrier for the light to pass. As a special criteria, the excitation of dye needs at least two photons to produce the fluorescent effect. And this is only the case at exactly the actually focused point [CoHa07, VoVP06].

Another important aspect of light microscopy is the staining of the probes. The staining has a direct impact on the color of the resulting images. Figure 19 shows the frequently used *hematoxylin and eosin staining (H&E)* (Figure 19). The first staining results in a dark blue color of the nucleus which is basophilic, because of its high content of nucleic acids. By contrast, eosin causes the red color of the cellular environment. The already mentioned GFP was used in Figure 19 to identify the cargo moving along the microtobules. Figure 18 shows the contrary coloring then known from the H&E staining. An explanation of the enormous number of different staining methods lies beyond the scope of this work. But in conclusion, the choice of appropriate staining depends highly on the type of cell components to be visualized.

The following microscopy types are used to analyze and visualize structures lying far beyond the scope of Light Microscopy.

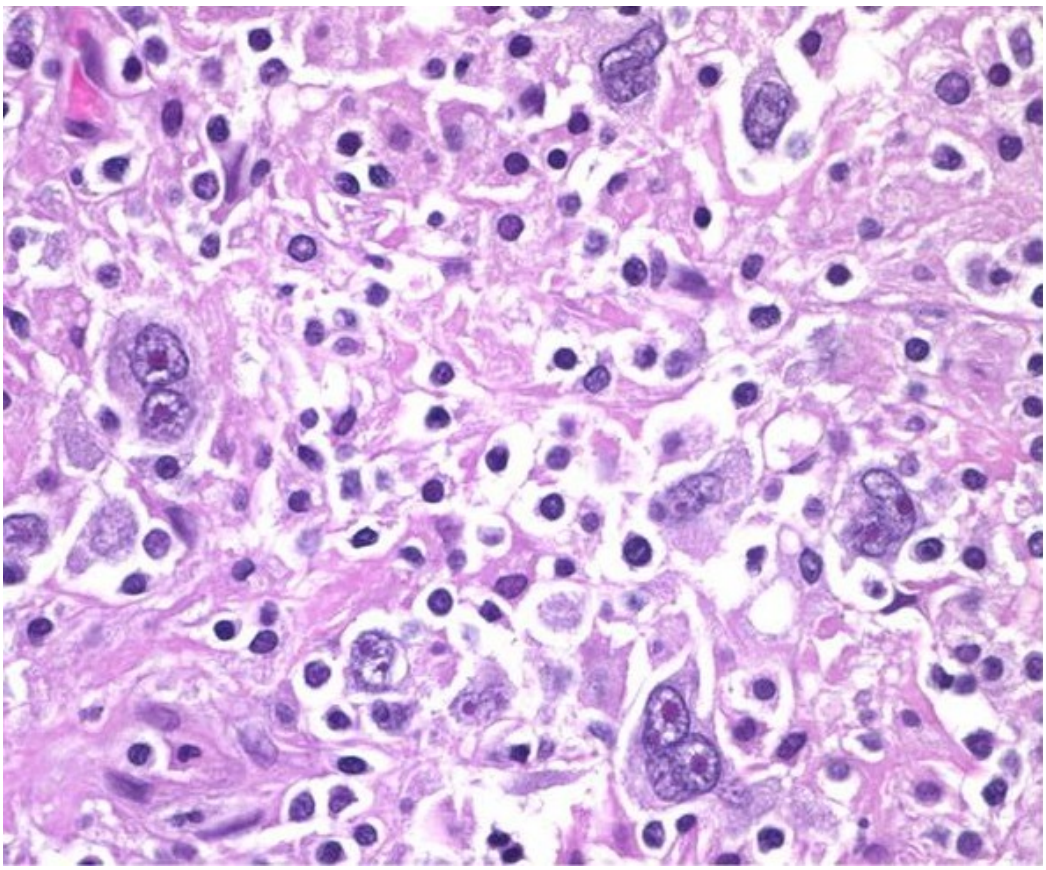


Figure 19: Hematoxylin and eosin staining of a human lymph node tissue
The nuclei are shimmering in dark blue, while the surrounding cells tend to red (Courtesy of/Copyright © 2011 by UMHealthSystem/flickr.com. Reprinted based on CC [Umhe11])

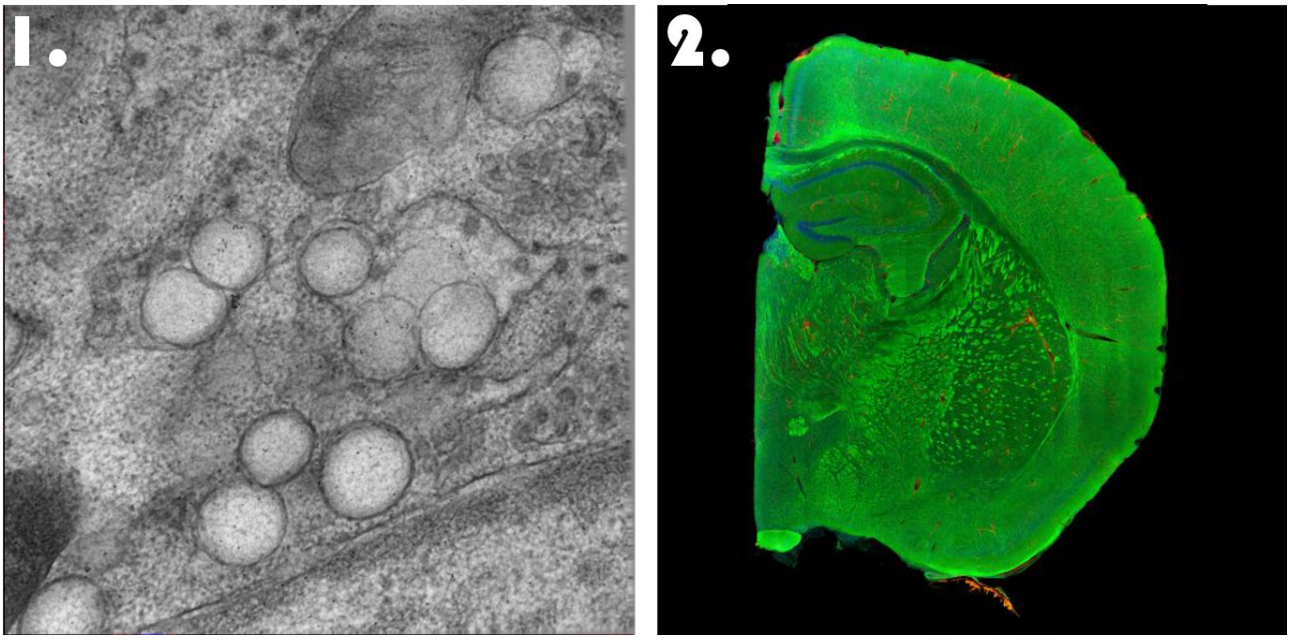


Figure 20: Transmission electron microscopy and Electron Scanning Microscopy
1. Transmission electron microscopy: African Green Monkey epithelial cells post-infected with SARS-coronavirus (Courtesy of/Copyright © 2002-2012 by The Regents of the University of California. All Rights Reserved. CCDB-ID: 6022 [KKWZ08, MGWQ02]); 2. Electron Scanning Microscopy: Large scale brain mosaic of a coronal section of a mouse brain stained with fluoromyelin (green), DiI (red), and TO-PRO3 (blue) (Courtesy of/Copyright © 2002-2012 by The Regents of the University of California. All Rights Reserved. CCDB-ID: 7290 [MGWQ02])

3.1.1.2 Electron Microscopy

Electron microscopy uses electrons to get insight into areas where the resolution of a light microscope fails. Although the minimal wavelength of electrons is 0.004 nm, there are other factors limiting the resolution of the microscope, e.g. a limited contrast and the properties of the electromagnetic lenses. Therefore, the practical limit of an electron microscope lies between 1 and 2 nm. In the following, two different major approaches should be exemplarily introduced.

The *transmission electron microscope* needs a specimen which is stained by heavy metal (Figure 20). A beam of electrons is passed through it. In contrast to light microscopy, where the light beams are marking the stained areas, the deflected electrons – which do not reach the fluorescent area – mark the structure. There are also different staining methods. If structures like lipids, nucleic acids and proteins should be shown, *positive staining* is used. *Negative staining* can be used to mark only special proteins.

Electron tomography uses the methods of transmission electron microscopy to combine different two-dimensional images of a specimen – an image stack – to achieve the three-dimensional structure. One obstacle for the microscopist is usually the alignment of the different images in the stack. In Figure 20.1, dark particles can be seen in the image. Those ones are colloidal gold particles of ∼10 nm which have been applied to the image as fiducial markers to enable the alignment of the different tilt series. This image stack, for example, consists of 131 images.

The *freeze fracture technique* aims at a similar direction (Figure 21). The specimen is shock-frozen in liquid nitrogen and a thin knife is used to remove its upper part. Platinum is added to the surface of the specimen. After the platinum reached the solid state, the organic material is removed and a metal template of the original structure remains which can be analyzed now with the microscope. This method is especially important for analyzing the structure of membrane-surrounded components.

Cryoelectron microscopy is an extension of the technique mentioned above. The sample is rapidly frozen to maintain the original shape. Temperatures like -196°C are reached so quickly that the surrounding water does not have time to crystallize, resulting in a clear transparent state. The included molecule of interest remains a hydrated structure, close to the native one.

Another method using electron transmission microscopy is discussed in Section 3.2.1.2.

The second major electron microscopy approach is the *scanning electron microscopy* (Figure 20.2). Here, the surface of a specimen is analyzed in reference to the scattered electrons which are reflected from the cellular surface which was initially coated with heavy metal. The collected data is used to generate a three-dimensional image. But the resolution of maximal 10 nm cannot compete with that of transmission electron microscopy, therefore normally only complete cells are analyzed using this technique [CoHa07, VoVP06].

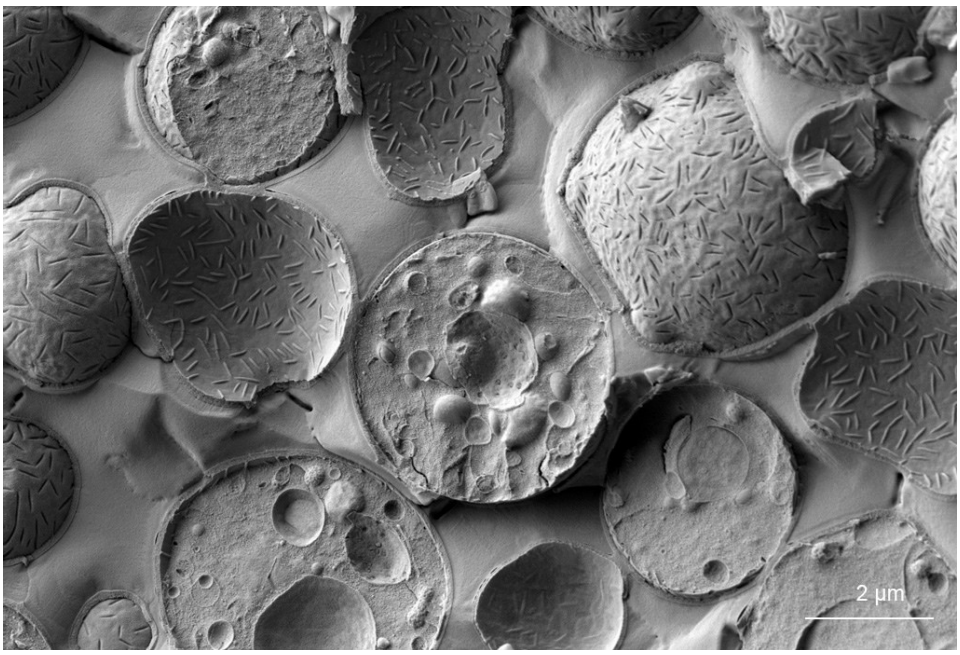


Figure 21: Freeze-fracture microscopy of baker's yeast
(Courtesy of/Copyright © 2011 by Gerhard Wanner and Carl Zeiss Microscopy/flickr.com. Reprinted based on CC [WaCa11])

3.1.1.3 Summary

It was now shown that light microscopy features a large variety of different staining methods affecting the colors of the cell representation. Therefore there are many options for establishing a color selection for a specific cell component. Of course, most of these techniques represent artificial effects caused by special florescent dyes. In addition, it was shown that electron microscopic techniques are very important to show the cellular structure with a high resolution of a few nanometers. Especially the freeze-fracture images provide insights into the three-dimensional structure of cell components, but at a lower resolution. The disadvantage of these techniques is that the specimen has to be – partly drastically – modified by heavy metal staining. But the major obstacle is the low resolution: atomic structures cannot be extracted by these methods. This issue will be addressed in Section 3.2.1.

3.1.2 Microscopy Databases

The data gathered by microscopy techniques can often be found published in on-line databases. Here, two scientific approaches are introduced, offering scientific microscopic data sets.

3.1.2.1 CCDB

The *Cell Centered Database (CCDB)* hosts two-dimensional, three-dimensional and four-dimensional data:

http://ccdb.ucsd.edu

The name of the database motivates the focus of this approach: the collection of all cell-related data. Images, image stacks, and three-dimensional models of cell components or smaller molecules are available as well as those of complete cells. Unfortunately, most of

this models or images show only fragments of components. Searching for the term “mitochondrion” results in 16 different models. Nearly all models represent slices through this cell component (e.g. CCDB-ID: 54, 3487, 3611, 3864, see Figure 22). Additionally, there are some models representing cell regions (like CCDB-ID: 3603) or even a complete cell (CCDB-ID: 7503).

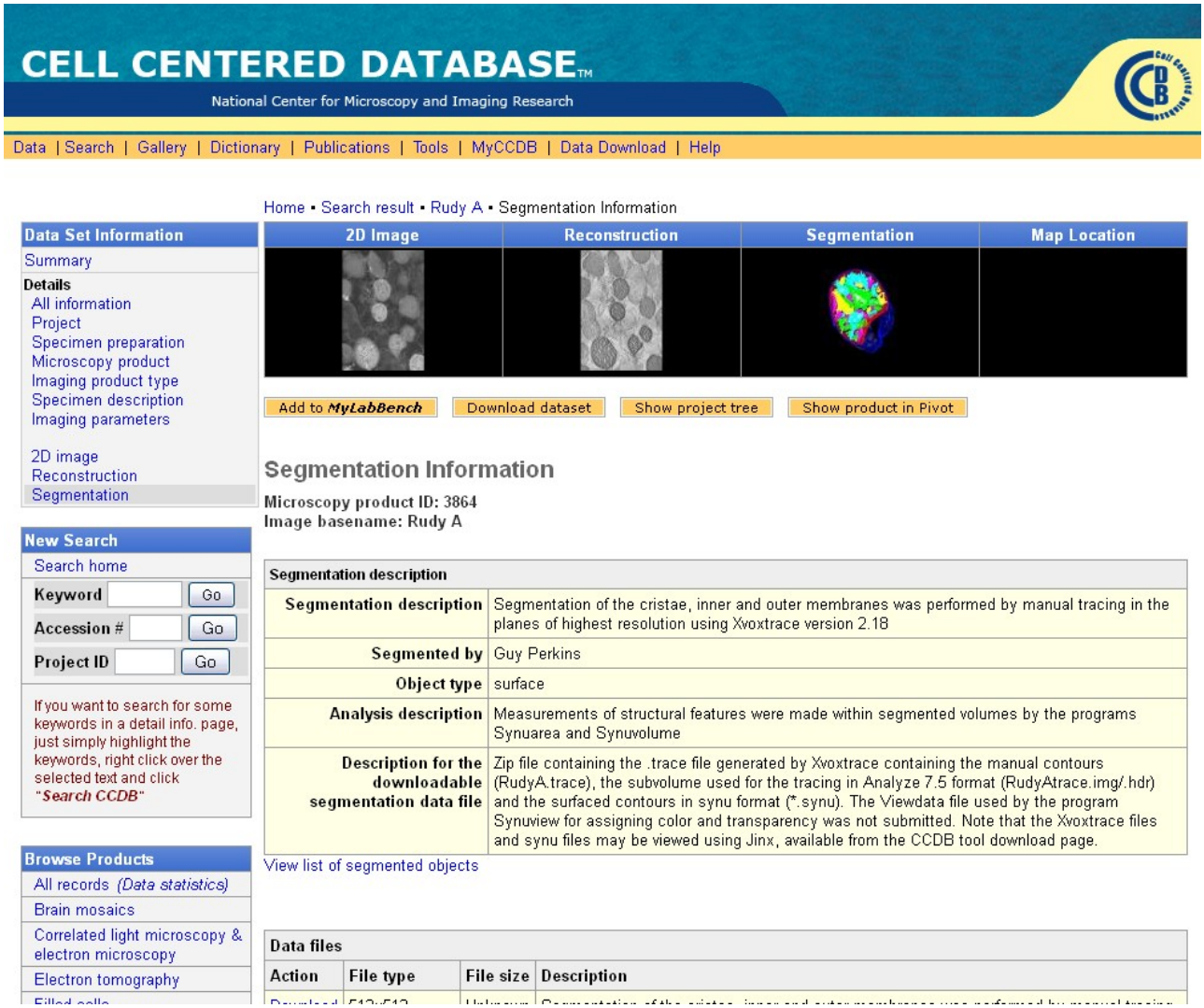


Figure 22: A slice through a mitochondrion as provided by the CCDB (CCDB-ID: 3864)

Of course, the resolution differences are significant. While the small mitochondrion slice (CCDB-6659) has a resolution of 0.55 nm/pixel in the X/Y plane, but only 250.0 nm/pixel along the Z axis, a complete cell like the protoplasmic astrocyte (CCDB-7503) may have a resolution of 12.24 nm/pixels.

In addition, the CCDB provides a set of tools which can be used to browse, manipulate and analyze image stacks at the local computer. These are alternatives to expensive commercial tools like Amira (Section 3.1.3.1). There are also some special web tools which can be used to preview and analyze the data to be downloaded.

Another important resource found at CCDB is the *Subcellular Anatomy Ontology (SAO)*:

http://ccdb.ucsd.edu/SAO

This approach describes the cell in terms of dimensionality, taking the cellular and

subcellular structure, supracellular domains as well as macromolecules into account. It should be mentioned that it was developed with a focus on the nervous system [LFGC07, MGWQ02].

3.1.2.2 EMDB

The *Electron Microscopy Data Bank (EMDB)* is a sub domain of the PDB database (see Section 3.2.2.1.1) and provides a number of electron microscopy density maps. They were created using techniques like electron tomography, electron crystallography and especially cryoelectron microscopy (see Section 3.1.1). In comparison to the CCDB, the molecular assemblies are smaller, subcellular structures and macromolecular complexes are found here, no complete cell components or even complete cells. As of now (31.07.2012), EMDB contains 1485 unique entries.

It is a joint effort of the Protein Databank in Europe (PDBe), the Research Collaboratory for Structural Bioinformatics (RCSB), and the National Center for Macromolecular Imaging (NCMI) and can be accessed via:

http://www.emdatabank.org

Figure 23 shows a nuclear core complex at a resolution of 85 Å revealed by cryoelectron tomography [11a, BFEP04]. The structures can be downloaded in map-format, containing the electron density maps which can be used for the visualization of the surface.

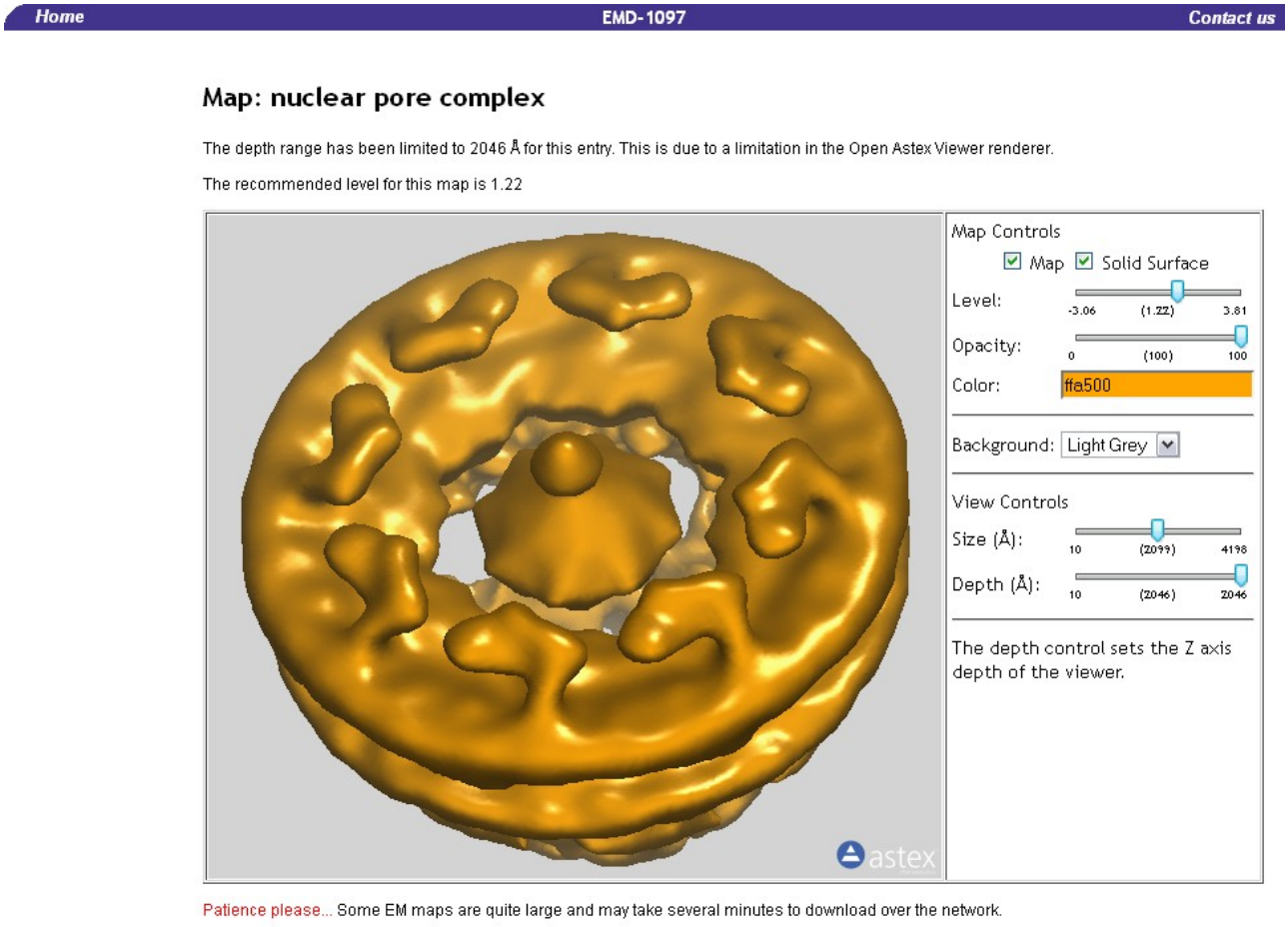


Figure 23: A nuclear core complex (EMD-1097) from the EMDB

3.1.3 Cell Modeling and Simulation

A variety of software tools have been developed to visualize the data generated with microscopy methods. Three frequently used tools shall be introduced in the following paragraphs.[16]

3.1.3.1 Amira

The software *Amira*® from Visage Imaging®, is a commercial visualization and analysis tool optimized for handling a large variety of data, for example from biology, chemistry and physics. The modular base application can be extended by different packages optimized for specific application areas like microscopy, image optimization and analysis, simulation, handling of different data formats, scripting and programming. Each extension has to be paid for.

The deconvolution package, for example, provides special filters to optimize microscopic images by reducing their blurriness (Section 3.1.1.1). The developer edition contains C++ header files which enable software developers to create custom modules for Amira. Originally it was based on Open Inventor, a C++/OpenGL tool kit developed at the Konrad-Zuse-Zentrum für Informationstechnik in Berlin [Konr02].

A very important feature in the context of this work is the image segmentation. Stacks of microscopic images which may be derived from CCDB (Section 3.1.2.1) can be visualized, analyzed and segmented by Amira. With different tools which will be discussed in Section 5.1.2.1.1 it is possible to generate three-dimensional shapes based on these images and export them to the VRML97 format (Section 3.1.3.4).[17]

In addition, in Section 4.1.1.1 it will be shown that Amira was also an important tool during the initial development of the concept presented and realized in this thesis.

3.1.3.2 3ds Max

Autodesk® *3ds Max*® (formerly known as: 3D Studio Max and 3dsmax), is a well-established commercial three-dimensional modeling tool. The application areas are wide-spread, for example: model and character generation and animation, movie effect generation, game development, architecture and product design, and rendering [12a].

During the development of CELLmicrocosmos, 3ds Max version 5, 6 and 7 were used for the modeling of cell components[18]. 3ds Max provides a large variety of tools and plugins to generate, model and manipulate objects and their meshes.

For the modeling of cell components, 3ds Max provides different objects and modifiers which are very interesting for the modeler. To provide a short overview, an excerpt of

16 It should be clarified that these software packages represent no related works in the context of this thesis. The packages discussed here represent only a small selection from a large pool of tools which can be used to support the development and modeling process of related approaches. The related work will be covered in the following Chapter 4.

17 One alternative to Amira in many application cases is UCSF Chimera. It is also commercial but for scientific use it is free. Since it was only used for testing purposes up till now, this package will not be discussed in this work.

18 Is should be mentioned that these versions of 3ds Max are relatively old. But for the basic modeling of cell components, the functionality of this program is still far from being exhausted.

interesting functions follows:

- Geometry
  - Primitives: Box, (Geo-)Sphere, Tube,
  - Extended Primitives: Capsules,
  - Compound Objects: Boolean, Scatter,
- Shapes
  - Splines: Line
- Modifiers
  - Noise, Normal, Relax, TurboSmooth, Wave.

In addition, 3ds Max features a native scripting language: MAXScript. By using this interface, especially complicated cell components can be constructed. For the modeling process of a complex structure like the tubular smooth endoplasmic reticulum, two helper scripts had to be applied (see e.g. Figure 120, the tubular structure at the upper left part). Other cell components like the cell membrane are quite trivial. Figure 24 shows the plasma membrane which is basically a sphere combined with Wave and Noise modifiers generating the liquid-like structure at the surface. In addition, 3ds Max provides a VRML97-export-plugin. After the modeling process is finished, this plugin is used to export to VRML97 format (Section 3.1.3.4).

Different modeling techniques for cell components using 3ds Max were discussed by [Somm04].

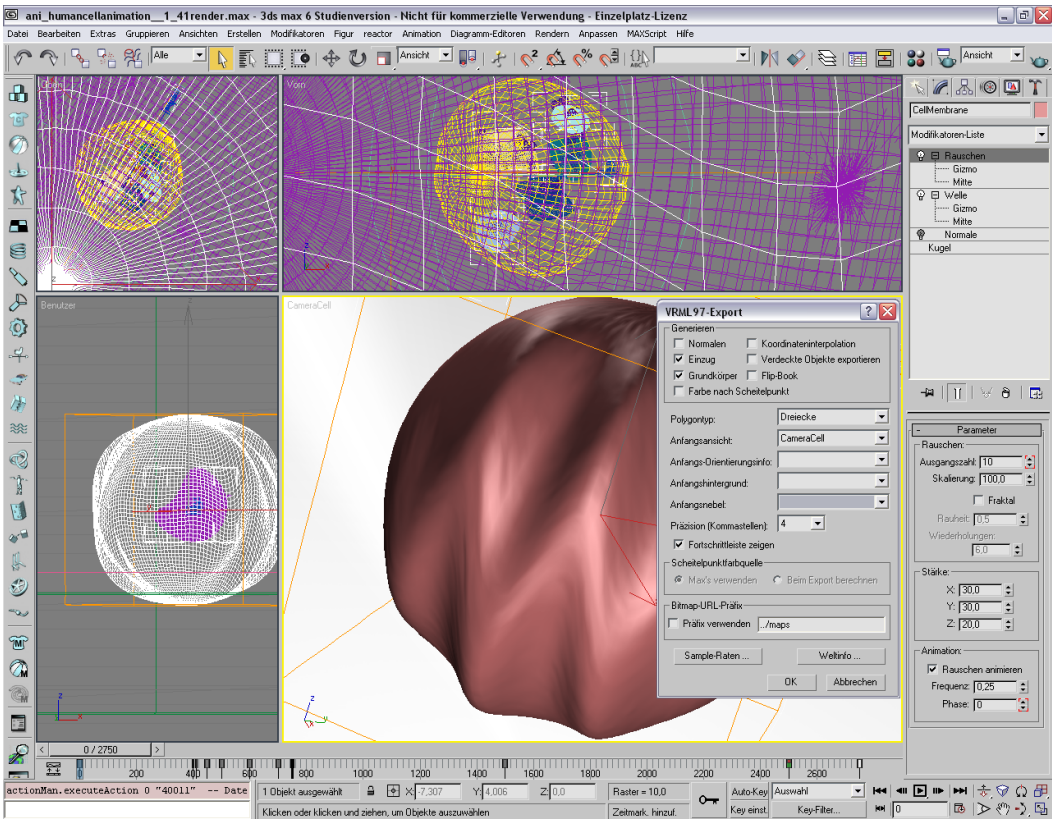


Figure 24: 3ds Max: this plasma membrane was modeled in 3ds Max and can be exported with the shown VRML97-export-plugin

3.1.3.3 Blender

Although most of the models shown in this work were modeled with 3ds Max, *Blender* has to be mentioned. It is a free and open source 3D-modeling tool which includes similar features like 3ds Max. Of course, it cannot outperform yet this commercial software, but many basic functions important for cell modeling as well as a VRML97 import plugin are already included.

Another important aspect is an initiative called *BioBlender* which aims to simplify and enhance molecular animation and visualization processes with Blender [ACZL10, Lok11, ZPAL10].[19]

3.1.3.4 The VRML-Format

In the preceding sections, the term VRML was mentioned quite often. Therefore, it should be shortly discussed in this section. This format is also a good example for showing the properties which have to be taken into account during the modeling process.

VRML stands for Virtual Reality Markup Language. The first version was introduced in 1995 and two years later, the new and today widely used version *VRML97* or *VRML 2.0* was introduced.[20]

In contrast to native formats like *.3ds* or *.max* of 3ds Max, this is a text-based format. The usual extension is *.wrl* (for world).

Figure 25 shows the excerpt of a listing describing a mitochondrion model. The top level *DEF* descriptors contain one shape each. Obviously, this mitochondrion model contains five shapes (see also Section 5.1.2.4). Different features of the model are described in the first blocks. The *translation* defines the position of the complete model in 3D space. The *appearance* describes the visualization: the different colors used, the transparency and the shininess. The *geometry* block is subdivided into two sections. The first section, *coord*, describes the points in 3D space. Each triplet describes one point, subdivided by commas. The *coordIndex* groups this points to a vertex area. Because the VRML-file contains a triangular mesh, one vertex always contains three points, subdivided by “-1,”.

Only a few years later the successor format X3D was introduced. It is an XML-compatible format with the XML-implicit overhead. But since 2004 it is the official standard for three-dimensional content in the web. Nevertheless there are many free and commercial applications supporting the VRML97 format. The size of its files is much smaller than the ones of X3D and they can be manipulated by any text editor [CaBM97].

3.1.3.5 Cell Simulation

Another important topic in the context of cell modeling is the simulation. Today, the term cell modeling is often synonymously used for cell simulation. Of course – as is shown in this work – the term “cell modeling” also applies to quite different approaches.

If cells are simulated, this has to be done on a mesoscopic, extremely simplified level.

19 For information regarding the authorship and tools involved in the modeling process, please refer to Appendix 9.17.3.

20 All applications discussed in this work support only the VRML97 format, because version 1.0 is outdated.

```
#VRML V2.0 utf8

# Produced by 3D Studio MAX VRML97 exporter, Version 7, Revision 1,76
# Date: Thu Dec 10 17:21:10 2009

DEF mito_0_matrix Transform {
  translation 0.002445 -0.001789 0.0001093
  children [
  DEF mito_0_matrix-TIMER TimeSensor { loop TRUE cycleInterval 3.333 },
    Shape {
      appearance Appearance {
        material Material {
          diffuseColor 1 0.502 0.251
          ambientIntensity 0.05556
          specularColor 0 0.002316 0.002316
          shininess 0.05645
          transparency 1
        }
      }
      geometry DEF mito_0_matrix-FACES IndexedFaceSet {
        ccw TRUE
        solid FALSE
        coord DEF mito_0_matrix-COORD Coordinate { point [
          -364.6 192.2 98.68, -369.4 196.6 81.45, -394.6 186.7 104.6,
          -412 187.8 85.26, -336.3 193.1 96.82, -335.1 197.4 80.28,
          -356.6 184.7 113.6, -381.8 180.6 121.5, -339.2 186.1 107.9,
          [...]
        }
        coordIndex [
          0, 1, 2, -1, 1, 3, 2, -1, 4, 5, 0, -1, 5, 1, 0, -1, 6, 0, 7, -1,
          0, 2, 7, -1, 8, 4, 0, -1, 8, 0, 6, -1, 9, 10, 11, -1,
          9, 11, 12, -1, 13, 14, 10, -1, 13, 10, 9, -1, 4, 9, 12, -1,
          [...]
        }
    }
  ]
}
DEF mito_1_inner_membrane Transform {
  [...]
}
DEF mito_2_intermembrane_space Transform {
  [...]
}
DEF mito_3_outer_membrane Transform {
  [...]
}
DEF mito_4_cloud_ Transform {
  [...]
}
```

Figure 25: VRML97: the plain text of mod_mitochondrium_1600X1000nm.wrl

Usually this is done by using stochastic/mathematical methods which simulate electrophysiological and/or biochemical processes. Because this work focuses on spatial, three-dimensional approaches, simple cell simulations such as cell automata operating only in two dimensions are not taken into account. The simulation of cells will be discussed in Section 4.1.2.

3.2 [MOL] Membrane Modeling

In the preceding subchapter, microscopy and modeling techniques for the mesoscopic level have been introduced. Now, at the molecular level, the resolution achievable by microscopy techniques is no longer sufficient. For this purpose different spectroscopy types will be introduced (Section 3.2.1). It will be shown that – in contrast to the mesoscopic level – the molecular level is covered by an adequate number of well-established databases (Section 3.2.2). Based on this data, membrane modeling and simulation approaches will be introduced (Section 3.2.3). In the following sections, the theory behind the Membrane Packing Problems will be introduced (Sections 3.2.4,3.2.5 and 3.2.6). And finally, appropriate criterion for the evaluation of packing quality will be discussed (Section 3.2.7).

3.2.1 Spectroscopy Types

Three different spectroscopy methods will be introduced in the following sections. The mesoscopic structure is known, but how is it possible to access the atomic level? How is it possible to examine the molecular structures of the membranes which dominate the cellular visualizations today?

3.2.1.1 X-Ray Crystallography

While light microscope depends on the minimal wave length of visible light of 4,000 Å, the wavelength of *X-rays* is approximately 1.5 Å which is approximately the same value as found for covalent bond distances between different atoms in molecules (Section 2.2.1.3).

To determine the position of the different atoms included in a protein, often the *X-ray crystallography* is used. Instead of a microscope, an X-ray generator or a so-called synchrotron[21] is used to create a collimated beam of X-rays. It is aimed at the protein in crystallized form. Crystalline proteins often assume very nearly the same structures that they have in solutions. After passing the protein, the X-rays are recorded by a radiation counter or recorded on X-ray film. On this film, a typical pattern, the so-called *diffraction pattern*, is created (Figure 26.1)[22]. It results from the specimen's electron density, which is the arrangement of electrons. Interaction of the X-rays with the nuclei of the atoms does not usually occur. The mathematical analysis of the diffraction pattern leads to the distinct atom positions which are characteristic for the analyzed protein. The visual representation of this result is often the electron density map, showing a three-dimensional mesh structure of the specimen. The width of these meshes indicates the density around the according atom in the molecule.

21 A particle accelerator which produces X-rays of a larger intensity than those of the traditional X-ray generator.

22 In this context the term *X-ray diffraction* is also often used.

It is important to note that the resolution of the proteins analyzed is limited not only by the X-ray wave length. The resolution of the crystal structure varies with the size and the type of the protein. While small structures are known to give a good resolution equal to the wave length of the X-ray, larger structures are normally disordered. The typical resolution limit of proteins lies between 1.5 and 3 Å. At a resolution of 1.1 Å, which applies only to a small amount of proteins, the exact atomic structure can be determined. On the contrary, there are proteins with a low resolution of 6 Å, which leads to a very vague determination of the atomic positions. In such cases, it is hard or even impossible to identify the type of a protein. Moreover, it is important to know the primary structure of the analyzed protein (see Section 2.2.1.4). It can be mathematically fitted to the electron density map and be used to determine the structure fairly correctly. The uncertainty can be reduced to 0.1 Å, while the most accurately determined small molecules feature an uncertainty of only 0.001 Å [CoHa07, VoVP06].

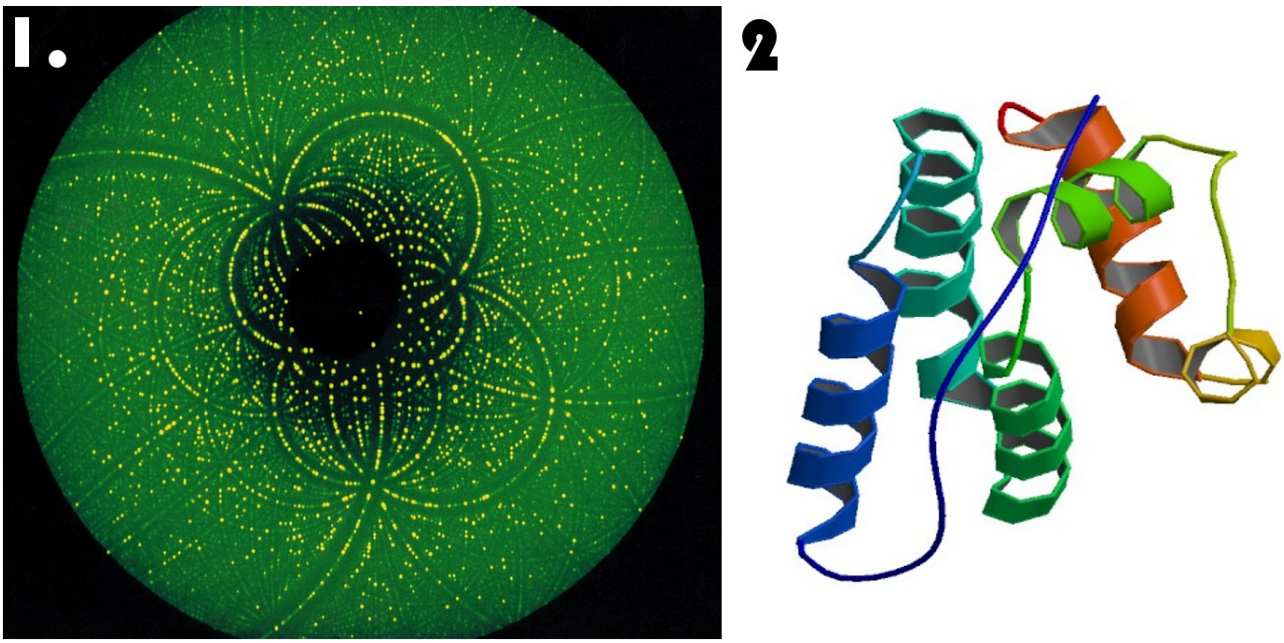


Figure 26: X-ray crystallography
1. Diffraction image has to be decoded to obtain the three-dimensional protein structure (Courtesy of/Copyright © 2012 by Max Planck Working Groups on Structural Molecular Biology. Reprinted with permission from [Mpi11]); 2. NMR Structure of a mitochondrial succinate dehydrogenase assembly factor 2 from saccharomyces cerevisiae (PDB-ID: 2LM4) [BWFG00]

3.2.1.2 Electron Crystallography

X-ray crystallography performs very well in the analysis of globular proteins. They are localized, for example, in the cytosol and it is normally possible to extract them from their natural environment and dissolve them into an aqueous solution. This aggregate can normally be crystallized. A problem arises if the protein of interest is part of a membrane complex. These membrane proteins cannot be simply extracted from the membrane into an aqueous environment, because they will automatically aggregate. Therefore the solution of the membrane protein has to be very similar to the natural environment.

To overcome this problem, *electron crystallography* combines the methods of electron

microscopy and X-ray crystallography. The beam of an electron transmission microscope is used to measure the diffraction intensities of a two-dimensional crystal. The latter included, for example, the first structurally defined transmembrane protein *bacteriorhodopsin*, located inside a purple membrane patch[23]. Because of the damaging nature of electron beams, only low electron intensities could be used for this process and a number of repetitions with different samples were inevitable. But the result was a structure with a quite high resolution of 3.5 Å in the plane [VoVP06].

3.2.1.3 Nuclear Magnetic Resonance (NMR) Spectroscopy

While the X-ray crystallography is only applicable to proteins in crystal, the *nuclear magnetic resonance (NMR) spectroscopy* is able to determine the structure of proteins in aqueous solutions. In addition, it is not based on the tracking of electrons but of the atomic nuclei, the protons. NMR processes signals emitted by radiofrequency-excited atomic nuclei in a magnetic field. It measures the inter-proton distances which are smaller than 5 Å. These distances may correspond to the space *(nuclear Overhauser effect spectroscopy, NOESY)* or bonds *(Correlated Spectroscopy, COSY)* in between the protons. The precondition is a previously identified sequence, because this sequence can be used to merge the knowledge about the geometric constraints[24] with the NMR measurements. The outcome is a geometric structure (Figure 26.2). But this structure with an approximately resolution of 2 to 2.5 Å cannot compete with the accuracy of the X-ray crystallography. Anyway, there are often structures which cannot be identified with the last-named method but rather with NMR. In other cases both methods are combined to approve the results.[25] Another large advantage of NMR is the ability to determine three-dimensional structures over time [VoVP06].

3.2.1.4 Summary

The computer is already usually used to save and especially combine images captured by the microscope as discussed in Section 3.1.1, but for spectroscopy it is indispensable to compute the three-dimensional structure of the abstract patterns. So again the derived structures are only partly reliable and they represent a single or a number of snapshot(s) of the molecule in time. In addition, many proteins are not accessible to these techniques because they cannot be extracted from their natural environment – especially in the case of membrane proteins.

Obviously every technique has its special pros and cons, but to create a cell model, an approach is needed to combine the results of different methods: the molecular level of the spectroscopy techniques and the mesoscopic level of the microscopy techniques. In this work, Integrative Bioinformatics will be introduced as an appropriate method to solve this task.

---

23 It is interesting to note that both the protein Bacteriorhodopsin and the surrounding membrane patch are purple. This fact shows that there also are colorful cellular elements [HeUn75].

24 Geometric constraints may be the covalent bonds distances, the van der Waals radii, the group planarity, the chirality or the angles etc.

25 In this context it should be mentioned that the search for the buckminsterfullerene mentioned in Subchapter 1.1 was accompanied by the competition of NMR technique used by Curl, Kroto and Smalley vs. X-ray crystallography applied by Krätschmer et al. [Krot97].

3.2.2 Structure Databases

Now that spectroscopy methods were introduced, the focus will be made on the storage of the generated molecular structures. This section will focus on different possibilities to obtain the three-dimensional structure of molecules from the Internet. A number of relevant databases providing different molecular models will be introduced. The overview of all databases discussed in this work is found in Section 3.4.2.

3.2.2.1 Protein Databases

As discussed previously, one of the major membrane components are the proteins. Moreover, this category provides the most comprehensive and established databases on molecular models.

3.2.2.1.1 The Protein Data Bank (PDB): Database and Format

The central database for protein research is the *Protein Data Bank (PDB) [BWFG00]*. It contains more than 83,000 structures (as of 31.07.2012) and celebrated its 40th anniversary in 2011. The success of this database is correlated with the decision of most protein-related journals to accept only publications in combination with freely accessible three-dimensional protein models in *PDB format*.[26] Table 5 shows the different methods used to obtain these experimentally-derived coordinate files. X-ray is superior, but there are also a lot of NMR structures found. Electron microscopy is only used for a few hundred proteins. But it is an interesting fact that in this category a relatively large number of proteins, namely approx. 40%, is found in nucleic acids complexes, whereas only approx. 5% of the total X-ray-derived PDBs are found there. That shows the importance of electron microscopy for protein in complexes, such as in membranes (see Section 3.2.1.2). In addition, there are few combinatorial or other approaches found in Table 5, but they are underrepresented.

Experimental Method	Proteins	Nucleic Acids	Protein/NA Complexes	Other	Total
X-ray	68368	1382	3464	2	73216
NMR	8337	999	188	7	9531
Electron Microscopy	305	22	120	0	447
HYBRID	44	3	2	1	50
other	141	4	5	13	163
Total	77195	2410	3779	23	83407

Table 5: PDB Current Holding Report as of 31.07.2012 [12b]

The PDB database provides a powerful web front-end at:

http://www.pdb.org

All structure files containing proteins, nucleic acids or other molecular coordinates can be

26 In the context of this work, the term “PDB” is used as abbreviation for the PDB file (format), unless the term “database” is explicitly added.

explored and downloaded from this website. It provides links to publications correlated with the PDB files and a lot of tools to visualize and analyze the structures. A typical representation of a PDB file taken from the website is shown in Figure 28 [DBYS09].

The PDB format is a text-based file format. It has many limitations. For example, the length of each text line is restricted to 80 characters. Each line has a special definition. The line for the type *ATOM* is used to store the coordinates of a single atom together with the type definition, the charge and the coordinates. Each coordinate is restricted to an 8.3 format. Therefore, the size of a molecule or membrane encoded in PDB format is limited to approximately 11,000 Å (incl. negative coordinate numbers). This shows that a number of limitations have to be taken into account if the PDB format is used for the modeling of larger structures.

Important advantages are that the PDB files are quite compact and that they are human-readable. An alternative is the PDBML format, an XML-based format. But the problem of this format is an overhead resulting in files which are up to twelve times larger than the PDB file[27]. An extensive introduction to the PDB format and its limitations is found in Appendix 9.6.

In Section 3.2.1.3 it was already mentioned that it is possible to analyze atomic structures over time by NMR. However, this was not possible for high resolution structures like those needed for the PDB format. Two independent groups published in 2009 new developments using rapid NMR 3D measurements [IOTI09, SSIH09]. They lead, for example, to the solution structure of Thermus thermophilus TTHA1718 (2ROG.pdb) containing 20 models which were taken from living E. coli cells. This shows that there is still much potential for the PDB database in the future.

The websites discussed in the following sections are all related to the PDB database and/or provide PDB-compatible structure files.

3.2.2.1.2 DisProt

The PDB database contains ordered proteins structures, this means the atomic structure is relatively stable. In contrast to this, the DisProt database contains disordered structures which continuously change their conformation. The *Database of Protein Disorder (DisProt)* is accessible at the website: http://www.disprot.org

As of 31.07.2012, 667 different proteins and 1467 regions are found in this database. For many of these disordered regions there are links to the corresponding PDB files included. The experimental detection methods used are partly already known from the PDB database, but furthermore there are also special ones which lie beyond the scope of this work [SHLV06].

3.2.2.1.3 Protein Modal Portal

As mentioned before, there is a large gap between those proteins whose atomic structure was resolved and those where the structure is partly or completely unknown. This is

27 Based on the protein containing the most atoms, 2KU2.pdb: approx. 102 MB for the PDB file, 1.2 GB for the XML file (as of 30.08.2011).

particularly the case for membrane proteins (Section 3.1.1.2). Therefore, efforts like the *Protein Model Portal (PMP)* were established. The PMP is a collection of tools and contains a collection of theoretical protein models in PDB format. The website is found at: http://www.ProteinModelPortal.org

The basis for this database is the UniProt database (Section 3.3.2.2.3). By using the UniProt identifiers, the website searches for similar structures which can be directly downloaded. If no structures are found, there are tools provided to compute the three-dimensional structure based on different information. For example, experimentally derived protein structures from the PDB database can be used as a template to create similar structures. The similarity as well as the quality of a predicted model can also be verified using this website.

As of 31.07.2012, the database contains 24.2 million comparative protein models for 3.9 million distinct UniProt sequences. This means that the amount of structures in the PDB database is quantitatively less than 0.35% of those found in the PMP database [AKKB09].

3.2.2.1.4 PDBTM

The *Protein Data Bank of Transmembrane Proteins (PDBTM)* is based on the PDB database. It addresses the Protein Packing Problem (Section 3.2.4) of PDB files and is found at:

http://pdbtm.enzim.hu

The PDB file does not contain information regarding the justification for a membrane patch as can be seen in Figure 28. One approach to overcome this issue is the application of special algorithms to automatically align PDB coordinates to a virtual membrane patch. For this purpose PDBTM uses the *TMDET algorithm*:

http://tmdet.enzim.hu

The PDB format contains an area to state the original transformation matrix of the protein in the REMARK section. The PDBTM database provides an XML file for every transmembrane PDB file, containing information on the transformation matrix, the *TMATRIX* (Figure 27). It includes the 3 X 3 rotation matrix and the translation vector for shifting whereas the Z-axis is parallel to the bilayer normal. The Z-coordinate of the *NORMAL* entry represents the half of the membrane width. After the matrix was applied to the original PDB coordinates, the new origin of the PDB file is the center of the bilayer. The TMDET algorithm can also be directly applied to a custom PDB structure by uploading it to the website stated above.

```
<MEMBRANE>
    <NORMAL  X="0.00000000"  Y="0.00000000"  Z="15.50000000"/>
    <TMATRIX>
        <ROWX  X="0.59916121"  Y="0.67847639"  Z="-0.42505959"  T="28.82857513"/>
        <ROWY  X="0.24127664"  Y="0.35321173"  Z="0.90389550"  T="-150.82160950"/>
        <ROWZ  X="0.76340777"  Y="-0.64413607"  Z="0.04793027"  T="-77.72867584"/>
    </TMATRIX>
</MEMBRANE>
```

Figure 27: PDBTM membrane section containing the matrix information for 1O5W.pdb see also Figure 28)

Here, a simplified description of the method should be given. First, the crystallization process of the protein is reversed by trying to remove chains forming non-biological contacts. Then, the lipid exposed solvent accessible area is computed. The prepared structure is applied to a direct search algorithm trying to find the most probable position relative to the membrane planes. Therefore the hydrophobicity as well as a set of different structural feature is measured and evaluated while the protein is rotated and the two virtual bilayer planes are shifted multiple times along the bilayer normal. The optimal position is chosen and the corresponding matrix is saved in an XML format.

The PDBTM database is frequently updated with the newest transmembrane proteins from the PDB database which are automatically realigned using the TMDET algorithm. The website for the Monoamine oxidase A (1O5W.pdb) is shown in Figure 28 [TuDS05a, TuDS05b].

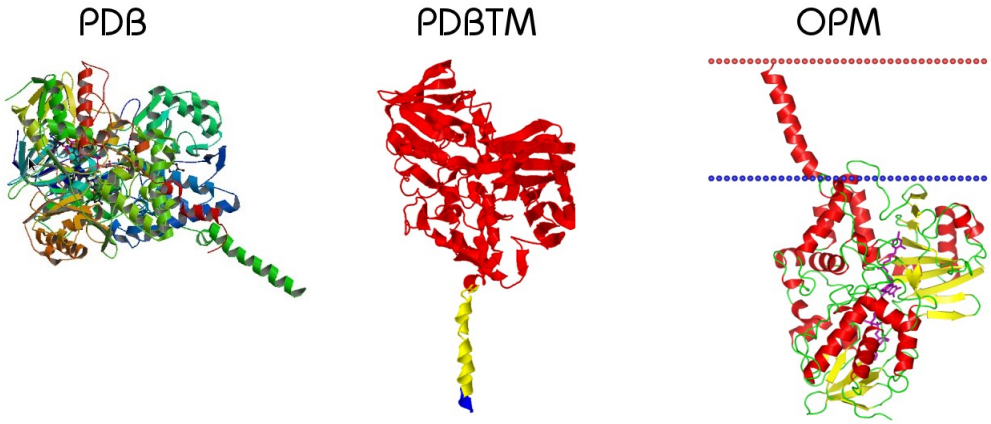


Figure 28: Images of the 1O5W.pdb taken from the websites PDB, PDBTM and OPM

3.2.2.1.5 OPM

An approach comparable to PDBTM is the *Orientation of Proteins in Membranes (OPM)* database. The website is found at:

http://opm.phar.umich.edu

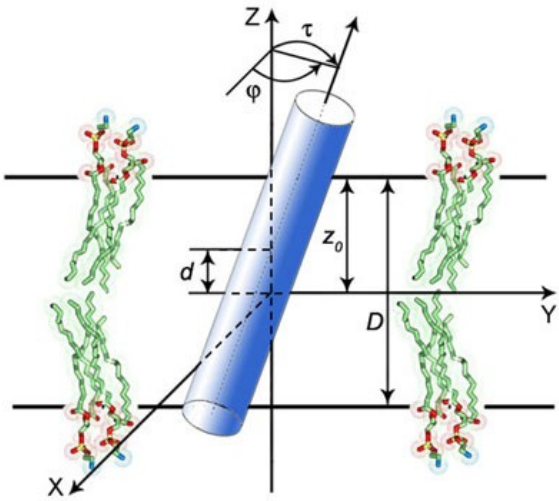


Figure 29: The relevant variables for the OPM protein alignment
(Courtesy of/Copyright © 2011 by Andrei Lomize, Mikhail Lomize and Irina Pogozheva. Reprinted with permission from [11b])

But there are a few differences which can already be seen by comparing the images taken from the websites of PDBTM and OPM, Figure 28. OPM uses *dummy atoms* to visualize the membrane patches: red O atoms for the top layer, blue N atoms for the bottom layer. There is no access to a transformations matrix. Instead of this, a PDB file containing the dummy atoms can be downloaded. The dummy layers can be used for the alignment to a membrane.

In contrast with PDBTM, the OPM database also contains transmembrane proteins as well as monotopic/extrinsic proteins and peptides. But it does not contain the full set of membrane proteins found in the PDB database. Protein models where the membrane alignment is not computable by the *PPM (Positioning of Proteins in Membrane) 2.0 method* are omitted.

Figure 29 shows the relevant variables for the computational alignment of a protein using OPM. As in PDBTM, the Z axis is parallel to the bilayer normal. The protein orientation is computed by minimizing the transfer energy $\Delta G_{transfer}$ with respect to the following values:

- the shift along the bilayer normally, d,
- half of the hydrophobic thickness, z_0,
- the rotation angle, φ,
- the tilt angle, τ.

The approach combines an anisotropic solvent bilayer representation with the universal solvation model which computes $\Delta G_{transfer}$. It takes hydrophobic, hydrogen-bonding, van der Waals as well as electrostatic solute solvent interactions into account to reach a high level of realism.

Plasma membrane:
- IN – cytoplasmic,
- OUT – extracellular;

Endoplasmic reticulum, Golgi, nuclear, peroxisome, endosome, vacuole, and vesicle membranes:
- IN – cytoplasmic,
- OUT – luminal;

Inner bacterial membrane:
- IN – cytoplasmic,
- OUT – periplasmic or extracellular space;

Outer bacterial membrane:
- IN – periplasmic space,
- OUT – extracellular;

Inner nuclear membrane:
- IN – lumen,
- OUT – perinuclear space;

Inner mitochondrial or chloroplast membrane:
- IN – matrix/stroma,
- OUT – intermembrane space;

Outer mitochondrial, chloroplast or nuclear membrane:
- IN – cytoplasmic,
- OUT – intermembrane space;

Chloroplast thylakoid membrane:
- IN – stromal,
- OUT – thylakoid space.

Figure 30: The topology used for the OPM database [11c]

Another important factor for protein alignment is the topology. If there is no topology, what is inside or outside of the membrane is not defined. This problem can also be seen by looking at Figure 28: the PDBTM alignment places the protein 1O5W at the top, OPM at the bottom. Therefore, OPM – in contrast with PDBTM – explicitly defines the topology of membranes with respect to the associated cell components, as shown in Figure 30. It defines the intention of “IN” (towards the bottom) and “OUT” (to the top) concerning the

images and models of the OPM databases and the according membrane types [11c].

It is also possible to use a custom PDB file for the computation of the OPM layers. For this task the *PPM server* can be used, found at [LLPM06, LoPM11, LPJM12, LPLM06]:

http://opm.phar.umich.edu/server.php

The authors of the PPM 2.0 server compared their method with other existing servers and showed that "[...] PPM clearly outperforms all of them in scope and accuracy and represents the only server that correctly predicts membrane-binding sites of peripheral proteins" [LPJM12, p.D375].

Figure 31 quantitatively compares the data of OPM (1991 entries) and PDBTM (1690). It shows that a subset of 500 proteins is included in both databases (as of 31.07.2012). Of course, the larger sets are unique, because both databases apply different approaches.

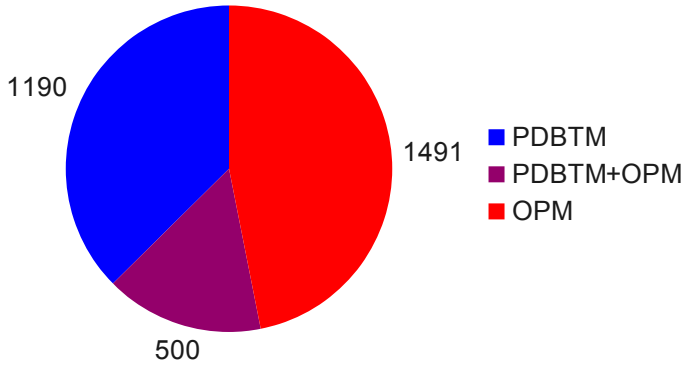


Figure 31: Comparison of 3181 proteins found in the PDBTM and/or OPM database

3.2.2.2 Lipid Databases

Lipids are the second major component of membranes. But in contrast to the protein databases introduced before, those databases are not so highly specific.

3.2.2.2.1 HIC-UP

The *Hetero-compound Information Centre of Uppsala (HIC-UP)* database is found at:

http://xray.bmc.uu.se/hicup/

This database provides experimentally derived as well as ideal coordinate files of small molecules found in the PDB database. For this purpose, single small molecules have been extracted from protein assemblies, for example, lipids which were crystallized together with a membrane protein.

HIC-UP contains 7,870 different structures since the last release 12.1, which was already in March 2008.

Although HIC-UP is no longer up-to-date, it still provides a good service to those scientists looking for a special small molecule like lipids, sorted by compound or trivial names, chemical formulas or compositions [11d, KlJo98].

3.2.2.2.2 PDB Ligand Expo

The reason for the discontinuation of the updating process of HIC-UP may be due to the fact that the PDB database now offers a similar service called *PDB Ligand Expo* which was

constantly improved and embedded into the PDB website service (see also Section 3.2.2.1.1). There are different search options to find and download a lipid – or other molecule – of interest [FCMA04]:

http://ligand-expo.rcsb.org

3.2.2.2.3 Avanti Lipids

Another possibility to get PDB models of lipids are commercial websites such as

http://www.avantilipids.com

Avanti® Polar Lipids provides a website where ingredients for wet lab experiments can be acquired. Browsing the website one sees that there are many lipid products which provide a PDB file to get a better impression of the lipid to buy [11e].

3.2.2.2.4 Klotho

The name of the website – *Klotho* – mediates a correct expectation: in terms of today´s fast-growing web community, this website is ancient. It was last updated in 2002, but it was obviously a pioneer website on this topic. However, it still contained a collection of 439 unique compounds with ideal coordinates which are released into the public domain. Many of these different small molecules were lipids which might be useful for the modeling process. These molecules were once created with a command line tool called “Moirai” [DFHW02]:

http://www.biocheminfo.org/klotho/

But for the present (last checked on 31.07.2012), Klotho is no longer available. But the complete PDB collection can still be downloaded using the WayBackMachine of web.archive.org [11f].

3.2.2.2.5 Chemistry Molecular Models

The website *Chemistry Molecular Models* is provided by Tom Zamis. It is a collection of various molecular models, partly collected from other databases like Klotho. There is no publication for this website. One smaller subset also contains lipids [Zami11].

http://www4.uwsp.edu/chemistry/pdbs/

3.2.2.3 Membrane Databases

In addition, to archives containing separate lipid and protein coordinates, there are also a few collections of membranes. These membranes are normally simulated, have been already released in conjunction with a publication. They were modeled under certain conditions. One of these archives is found on the CHARMM-GUI website:

http://www.charmm-gui.org/?doc=archive

Pure lipid bilayers as well as protein/membrane complexes can be downloaded there. For more information on the CHARMM-GUI project please refer to Section 4.2.2.3 [JoKI07].

3.2.2.4 Quantitative Comparison of Databases

Figure 32 shows the major problem of mesoscopic databases[28]. While the data of the relatively new EMDB grows constantly, the contributions to the CCDB seem to stagnate. In

contrast to this, the data of the PDB is growing continuously. Obviously there is some restraint for microscopists to release their data to the public. The reason is the partial high personal and financial efforts to produce these datasets. And in contrast to the protein structures it is not necessary to submit a mesoscopic structure to a database prior to publication. This is a problem, because the success of the PDB database shows the importance of such an approach. But as long as the mesoscopic data is hidden in laboratories, a lot of knowledge will be lost over time which could be important for gaining new insights for future experiments.

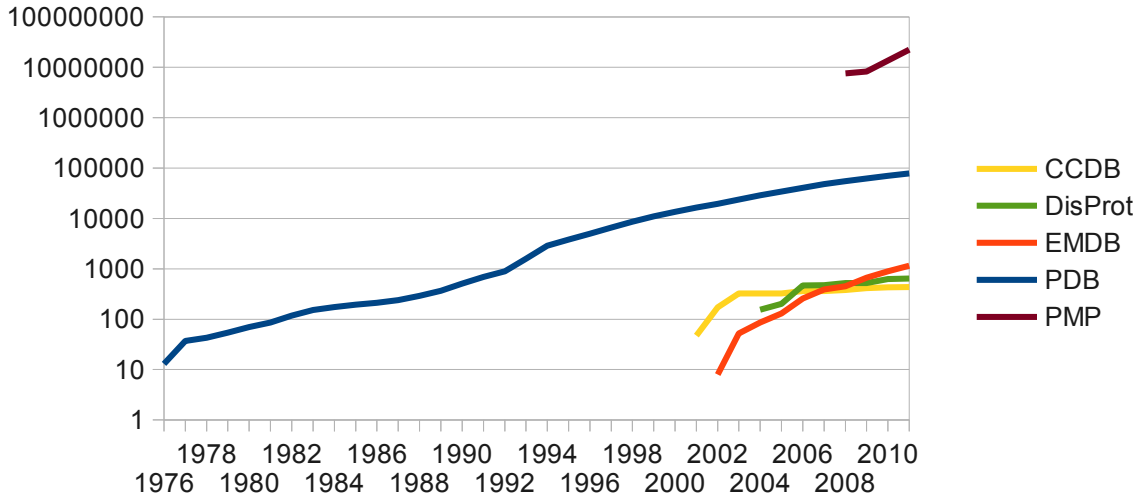


Figure 32: A comparison of the total data growth of five protein-related databases CCDB, DisProt, EMDB, PDB, and PMP[28]

3.2.3 Membrane Modeling and Simulation

In many life science-related application areas like (bio-)physics, (bio-)chemistry, biology, medicine or pharmacy the examination of membranes play a crucial role. But although life imaging techniques and nanotechnology exist, it is still a significant problem to investigate the structural reality of a biomembrane completely. A lot of facts are known today and since the mosaic model was introduced by Singer and Nicholson, many studies have experimentally confirmed and extended the theory behind this model (Section 2.2.3). Those *in vitro* – artificially, created outside of the living organism – membranes are also often referred to as *membrane models*. This work is of course aiming at *in silico* models which means, that they were generated with computational methods.

But still there are a large number of questions which cannot be proved experimentally today. Different molecular simulation approaches are applied today to problems which lie beyond the scope of the work of the experimentalist. But although a lot of work has been published in this area, crucial aspects of biological cell membranes are often ignored.

28 The Y-axis is scaled logarithmically. The PDB, EMDB and DisProt data is directly derived from the according websites. The CCDB data is an estimation by manually evaluating single data entries of the website. It is not possible to get exact data in this way, because entries may have been made accessible to the public years after the creation and submission date. The PMP data is derived by using the WayBackMachine (http://web.archive.org/web/).

- Most recent simulations take only one to two different lipid types into account. For the simulation of homogeneous membranes the first case is of course sufficient. But an important aspect of today's research is the heterogeneity of the cell membranes (see Section 2.2.6). To study this area, new membrane modeling approaches are needed.
- Lipid rafts play a crucial role in the inter-cellular and intra-cellular transport and communication (see Section 2.2.5). At the moment there are still only a small number of publications analyzing these approaches.
- Multi-layered organelles like the mitochondrion, the chloroplast or *multilamellar vesicles* contains different monolayers or bilayers (see Section 2.1.3.10). The study of the communication of these layers might play a crucial role in the understanding of intra-compartmental communication.
- The insertion of proteins into membrane layers is an important step prior further simulations of membranes. There exist a number of databases which take this aspect into account, using computational methods to support the protein placing process into a membrane (see Sections 3.2.2.1.4,3.2.2.1.5).
- In the following sections it will be shown, that collision-detection plays a crucial role during the generation of membranes. Overlapping atoms are omitted in nature, so they should be omitted also *in silico*.
- Due to the steady increase of computational power, the size of the simulated membranes and the number of atoms simulated is constantly growing. Therefore also the modeling process should enable researchers to generate large-scale membranes.

An appropriate discussion of simulation approaches lies beyond the scope of this work. But because the modeling process is often an important step prior to simulation, the *Molecular Dynamics (MD)* method should be roughly drafted. In Section 2.2.1.3 a few of the different physicochemical forces operating in biological membranes have been mentioned. MD simulation uses a set of these forces and simulates aspects like the attraction and repulsion of atoms with respect to the connectivity of a lipid or protein. The velocities of the atoms are computed by solving Newton equations of motion. For this purpose a *force field* is needed which includes all information like the bond lengths and types, various force constants, or the atomic charge and mass. Bonded as well as non-bonded interactions are taken into account. Of course the model to simulate needs an appropriate environment. This is usually a water bath, consisting of thousands of H_2O molecules. In addition, *periodic boundary conditions* are usually applied. This means, that the *simulation box*, consisting of the model and the water molecules, is handled as a continuous instance. The left border of the box virtually adjoins the right border of the box. Therefore a lipid molecule at the left border interacts with the one on the right. In addition, most MD simulations assume a neutral state. This means that the overall charge of all molecules in the simulation box is 0. Of course, still this method is also only a reductive variant of the complex forces occurring in a membrane *in vivo*. But today it is an established method to predict, for example, the behavior of membranes [VZHP93, WoRo96], proteins [TiBe98, WoRo94a] or viruses

[KrFi08]. A drawback of these molecular-simulation-related approaches is the normally indispensable expert knowledge and the need for extreme computational power, usually only provided by computer clusters.

A tool which is mentioned very often in this context is *GROMACS (GROningen MAchine for Chemical Simulations)*. The development of this freely accessible command-line tool has continued since more than a decade. It supports the parallelization of MD simulations and is applied to the simulation of small molecules, proteins, membranes or vesicles [HKSL08]. In addition, it is usually used in conjunction with PDB files as input and – often also – output format. It was also used to simulate membranes generated with MembraneEditor [SDGS11].

3.2.4 Membrane Packing Problems

Although a number of membrane generation approaches exist, the theoretical description from a computational point of view was very vague in recent literature. A first approach in collecting and combining different methods to get a first distillate on this topic was recently published [SDGS11].

In the preceding section it was already mentioned that an important aspect for a reasonable membrane generation is the collision-detection. In the case of molecules, this regards the atoms which are repulsing or attracting forces which are strongly correlated with the van der Waals or covalent radii of the atoms (Section 2.2.1.3). The well-known visualizations from different programs (see Section 4.2.1) are normally directly correlated with spheres having corresponding radii. For MD simulations it is not sufficient to take only the collisions of the outer radii into account. Many different atomic forces, the temperature and the pressure are involved in the computation of the dynamics. But for the preparation of model membranes it will be shown that the simplified approach focusing on geometrical properties already leads to reasonable results (Subchapter 6.2).

There is also an area in computational science dealing with similar problems which are ignoring molecular forces: the *packing problems* [LoMM02]. They cover a wide range of economical and ecological applications. Obviously the logistic sector existentially depends on the optimization of transport space. A number of publications in this area exist, for example [Dyck90, DyFi92, Nauj95]. Packing problems belong to the *NP-hard problems* [Wott96]. Therefore, there is no efficient as well as exhaustive solution known for this kind of problems. The naive way to solve this problem would be to test all possible solution in order to find the best one. The problem is that there are many complex problems which will not be solved by naive approaches by even the fastest computer in the world until the end of time as we know it. Therefore, different optimization approaches were created in the past, and a small set of them will be discussed in this work (see also Section 3.3.4).

A membrane mainly consists of proteins and lipids, as it was already discussed in Section 2.2.3. As it is shown in Figure 33, proteins as well as lipids can be interpreted as subclasses of molecules. On the one hand the membrane “has” proteins and lipids. Both both molecule types are very distinct. While the differences in the size and structure of lipids are relatively

small, proteins are large complexes which may contain a number of substructures (see Section 2.2.1.4). Therefore, it is essential to differentiate in terms of the packing process between this two molecule types.

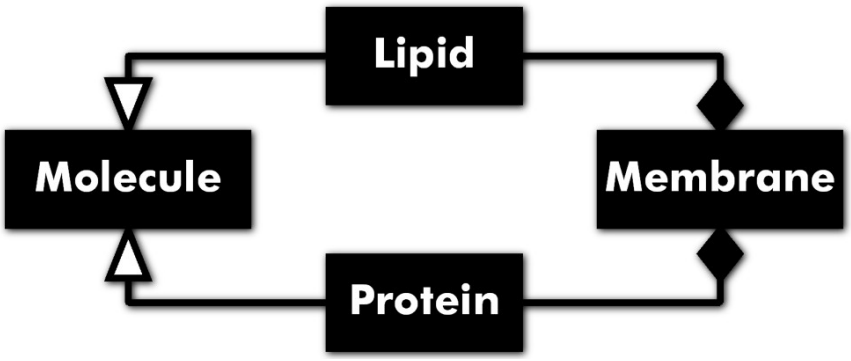


Figure 33: Class diagram: relationship between molecule, lipid, protein and membrane (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

Now the adequate definition for the different packing problems can be given.

- *(LPP): Lipid Packing Problem:* How could one or multiple lipid models be optimally packed onto a mono- or bilayer?
- *(PPP): Protein Packing Problem:* How could one or multiple protein models be optimally packed onto a mono- or bilayer?
- *(MPP): Membrane Packing Problem:* How could one or multiple lipid and protein models be optimally packed together with respect to LPP and PPP onto a mono- or bilayer?

There are only a small number of publications mentioning these terms to date. MPP is mentioned as a mathematical packing problem in conjunction with elastic membranes contacting a rigid obstacle [HBLL97, LóSt07, OuKZ98]. But these approaches do not take the molecular level into consideration. They are focusing on the elastic features on a higher level. Israelachvili et al. used the term LPP during the investigation of vesicles [IsMN76, Mars96]. Lipids are the decisive factor during the vesicular arching formation. Also the term PPP is found in literature. Feng et al. described the orientational distribution of contacts in protein clusters and proposed an icosahedron as an appropriate descriptive model [FeJK08]. We recently published the combination of all these terms [SDGS11].

Now that the different packing problems are defined, it is important to define additionally a set of optimization criteria which will be examined in the application chapters (Subchapter 6.2). This is especially important because naturally there is a large set of physicochemical properties which could be taken into account and which lie beyond the scope of this work (Section 2.2.1.3). The optima discussed in this work are defined as follows:

- *OPT_LPP_MIN_AREA:* The optimum is defined by the minimal possible area per lipid ($Å^2$) (Section 6.2.1, 6.2.4).
- *OPT_LPP_DEF_AREA:* The optimum is defined by the user-defined area per lipid ($Å^2$) (Section 6.2.1).
- *OPT_LPP_2D_AREA:* The optimum is defined by the highest possible membrane area occupied by lipids (%) (Section 6.2.2).
- *OPT_LPP_RATIO:* The optimum is defined by reaching the correct lipid type ratios (%) (Section 6.2.1, 6.2.2, 6.2.4, 6.2.5).

- *OPT_PPP_POS:* The optimum is defined by the correct protein positioning in relation to the bilayer (Section 6.2.3).
- *(OPT_LPP_ENERGY:* The optimum is defined by achieving the energetic equilibrium state.[29])

The optimization problems as well as the optimization criteria are now defined. Now the optimization solutions have to be defined. Analog to the packing problems, these are as follows:

- *(LPA) Lipid Packing Algorithms*,
- *(PPA) Protein Packing Algorithms* and
- *(MPA) Membrane Packing Algorithms*.

The significance of this definition emerges from the need to separate terms like MPA from *membrane algorithms*. These algorithms are found in the area of parallel and distributed computing, describing a set of biological motivated theoretical models [PaMa05].

3.2.5 Traditional Membrane Modeling Methods

Before analyzing the different tools which can be used to model membranes in a more or less automatized fashion, first the traditional, manual methods should be mentioned. These methods were needed by the authors of the following work to create new methodologies in those days. They had to invest a lot of time in the membrane generation process prior to the simulation of the membrane.[30]

The methods used to create pure membranes or protein/membrane complexes can be subdivided into two methods, the replacement and the insertion method [JoKI07].

3.2.5.1 Replacement Methods

The *replacement method* was initially described by Pastor et al. [PaVK91]. It follows a generalization of this process:

1. A single lipid model used during the generation process is represented by a simple three-dimensional structure, a sphere. This can be a simple sphere representing the headgroup [PaVK91], a Lennard-Jones sphere of 4.8 Å radius corresponding to its average cross-sectional area [WoRo96], a van der Waals radius sphere [PGME00] or spherical beads with an approximated size of the polar headgroups [ZMHR09].
2. If the resulting structure should contain one protein or one peptide assembled to the membrane, this molecule is used as the starting structure [PGME00, WoRo96]. It defines the center of the membrane and during the following steps (Step 3 to 7) it remains fixed at its initial position.
3. These spheres are initially distributed onto a plane (and around the protein or peptide, if present). This may be a regular hexagonal [PaVK91] or octagonal arrangement [WoRo96], or they are just distributed randomly onto a plane [PGME00, ZMHR09].

29 To solve OPT_LPP_ENERGY, for example, MD simulations are an appropriate approach. We showed in our recently published work that membrane models resulting from MembraneEditor can be used as an appropriate initial configuration for such simulations [SDGS11].

30 An extended overview of Membrane Modeling Methods and Tools (Section 4.2.2) will be discussed in the forthcoming publication: Membrane Packing Problems – A short Review on computational Membrane Modeling Methods and Tools. *Computational and Structural Biotechnology Journal* 5, 2013 [Somm13].

4. If homogeneous bilayers are constructed, this step might not be necessary [PaVK91]. For heterogeneous bilayers, the packing of the spheres representing the lipids have to be optimized using different approaches. This might be a MD simulation [PGME00, WoRo96] or a shrinking process using exemplarily the steepest descent (SD) minimization algorithm [BrBO83, ZMHR09]. During the repositioning process the vertical positions of the spheres are normally fixed. But in some special cases their vertical movement is only limited to a short range [PGME00].
5. After the distribution process is finished, the spheres have to be substituted by the original lipid models. The lipid models used are normally randomly chosen from a trajectory, this means a library containing a limited number of different conformations of a lipid chain [PaVK91]. These trajectories have been previously created by simulations and are often taken from already published and verified data (for example [VZHP93, WoRo94a]). During the substitution process the lipid is also rotated randomly in a range of 0 to 360° in the X/Z plane and tilted, for example, in a range of 0 to 45° with respect to the bilayer normal [PaVK91]. An important aspect of this placing process is the fact that this placing does not take collision detection into account. Therefore, in the end there are a lot of steric collisions inside the generated membrane model.
6. To reduce the collisions of the different lipid atoms, different methods are now applied which are all similar or equal to minimization processes. During the energy minimization the radii of the atoms are growing from 1 to 100% [PaVK91]. An alternative approach is that the lipids remain stiff structures while they are systematically horizontally moved and tilted (which also can be done with the peptide, if present) [WoRo96]. This method can also be combined with an increasing process of the atom radii [PGME00]. In addition, the SD minimization algorithm can also be used for this task [ZMHR09].
7. However, after these packing optimization approaches have been applied, the bad contacts are reduced but do not vanish (for example a decrease of factor 2 is observed [WoRo96]). Now a number of atomistic energy minimizations have to follow in order to drastically reduce the number of atomic radii collisions.
8. After these steps have been applied, the model membrane is now ready for the equilibration process. Now all constraints, like the fixation to a position in the X/Z plane, are removed and the molecules are able to move and interact freely.
9. And finally, the MD simulation can be started.

3.2.5.2 Insertion Methods

The *Insertion Method* is usually used to integrate larger molecules like proteins into a previously generated bilayer. These methods apply to optimum classification OPT_PPP_POS.

1. Shen et al. used a method combining the previously discussed replacement with the insertion method. As the starting structure a previously published and well-equilibrated membrane model was adapted. Then, two lipids were removed from each membrane side. But the created void was not large enough to accommodate the peptide. Therefore,

weak cylindrical repulsive forces were appended to the void, causing an increasing of the hole's radius during the following minimization and MD simulation. After the volume of the hole was large enough, the peptide was inserted there. Still, there were some collisions found between the atoms of the lipids and those of the peptide. The system was minimized again to get rid of these collisions and then the equilibration and MD simulation could follow [ShBS97].

2. Tieleman and Berendsen used a slightly different approach. First, a rather small bilayer was created using a random rotating method containing 64 lipids in each layer. This bilayer was solvated and simulated for a short time. Then the bilayer was multiplied by 4, containing 256 lipids in the end at each side. Two methods were now used to insert the protein into the membrane.
    1. The protein was inserted into the membrane and an abstract grid was laid on the membrane area. Each quadratic area was then checked for coexistence of the protein with lipids. If this was the case, all lipids in the affected quadratic area were removed. The result is a membrane without any atomic collisions, but with large distances between the lipids and the protein.
    2. The second approach used a similar technique but with the allowance of minor intersections. Therefore, a number of energy minimization attempts combined with force-related investigations and removal of some intersecting lipids were needed. The result was more complicated to achieve then the first approach but because of its nearly consistent structure it provided higher quality [TiBe98].
3. Faraldo-Gómez, Smith and Sansom generated a solvated and equilibrated lipid bilayer in the first step. Then they created the solvent-accessible surface area of the protein which was used in the following process as its placeholder. Thereafter a special method was applied to estimate a cylinder-shaped region for each side of the bilayer where no lipids should be situated. As the next step, all lipids were removed if their headgroup P atom was intersecting the region created before. Afterwards, only some lipid tails extended into this region. Therefore there was no dramatic change in the area per lipid during the following MD simulations [FaSS02].
4. Yesylevskyy developed a program called *ProtSqueeze*. This is a program which was initially developed as a plugin for the VMD (see also Section 4.2.2.4) [Yesy07], but it can be used with virtually any other program. It consists of three stages.
    1. The first stage is the preparation. A pre-equilibrated bilayer has to be used and a protein has to be pre-aligned into the bilayer. ProtSqueeze is able to use OPM structures for the pre-alignment (Section 3.2.2.1.5). Another option is to use the supplied scripts of ProtSqueeze, which try to place the protein in relation to the bilayer based on its residues defining the two membrane-water interfaces. Then, all lipids are deleted which overlap the protein by a user-defined percentages based on the steric clashes.
    2. The squeezing stage shrinks the protein until all steric clashes are eliminated. A special method is used to take possible tilts of the protein into account.

3. Finally, the expansion starts. The protein is expanded by a very small value at each step, followed by an energy minimization (Section 3.2.3) of the surrounding lipids to remove possible new overlaps. Step 3 is repeated until the original size of the protein is reached.

3.2.5.3 Discussion

In summery it should be mentioned, that the border between the replacing and insertion methods are floating. Often both methods are combined, lipids are often initially substituted by simple structures and proteins remain as atomic structures, or the proteins are temporarily represented by shapes, while the lipids are handled as atomic structures.

However, obviously the generation process it quite complex. Often these methods are not straight-forward. For example, multiple minimization attempts are needed to generate a membrane model which is usable for the following simulations steps or during the minimization of the radii of the molecules as increase to perform alignments and so forth.

In addition, none of this approaches explicitly uses collision-detection to generate membranes. The reason is that the applied programs or scripts are not able to provide collision-detection. Therefore, a more automatized assembly process is needed. Moreover, the generation protocols found in the articles discussed above are very vague. An Example:

"The configurations were assembled as a set of rigid units, with each GA or DMPC (with their primary waters) being translated and rotated in a systematic search for an optimum packing" [WoRo94b, p.11632].

It would be preferable if the generation of a membrane is accompanied by the creation of a protocol which explicitly states the different steps used to achieve the result, for example, showing variables and geometric settings. In addition, the results should be reproducible. Many approaches are using a random distribution of lipids. But the professional generation of membranes should also use seed-based random numbers like all professional simulation packages do, allowing other users to re-generate the assemblies.

Section 4.2.2.1 discusses a program, namely MOE, which is used in the context of this work to show how the approaches discussed above could be applied in practice. The ensuing sections then introduce tools which are related to CmME, implementing automatization approaches for the membrane generation process.

3.2.6 The Knapsack Problem

Now that the traditional solutions are discussed, the Lipid Packing Problem should be examined now from a computational perspective. As mentioned before, packing problems are a well-known area in computer science, but which was so far – to my knowledge – not directly related to the generation of biological membrane models [SDGS11].

One well-known computer science-related packing problem is the *Knapsack Problem (KP)*, which is formally defined as follows [KePP04]:

$$\text{maximize } \sum_{i=1}^{n} v_i x_i \text{ , subject to constrain } \sum_{i=1}^{n} w_i x_i \leq W \text{ , } x_i \in \{0,1,\ldots,c_i\} \text{ .} \quad (15)$$

In detail, this means that there are n kinds of items x with values of v_i and a weight of w_i. The overall weight of the given container is restricted to W. In regular packing problems, the container – which is the distribution area – is normally defined as the object. In this work, the term *container* is preferred, because it is more intuitive for the KP. The maximum c_i restricts the number of copies x_i for each item and is a characteristic criterion for the *Bounded KP (BKP)*. This means that the area in the knapsack is limited. On the contrary, if c_i is defined as infinity, the problem is an *Unbounded KP (UKP)*. In addition, for v and w, only positive integer values i are relevant in the context of LPP.

The approximate theoretical description of the membrane as a two-dimensional liquid arrangement was already proposed by the mosaic membrane model (Section 2.2.3). How can the LPP now be applied to the KP?

The virtual value w_i – the *weight* of the knapsack – is often used to describe the dimensionality.

Moreover, there is the so-called *Multidimensional KP* for knapsacks featuring multiple dimensions [KePP04]. Because the focus of the LPP lies in the description of rectangular bilayers, the *two-dimensional* [DyFi92, LoMM02] or geometric [Caga94] *KP* class is appropriate *(2D-KP)*.[31]

The regular KP is restricted to one container. Of course, there is also a class for *multiple* related containers, the *Multiple KP (MKP)* [Dyck90][32]. But in the case of the LPP, a monolayer can be defined as one container and a bilayer as two containers. In contrast to the MKP these two containers of a bilayer are not related as in the case of the LPAs discussed in this work (Section 5.2.2.5). The reason is that all application cases are focusing strictly on the geometric properties of the molecules, refusing physicochemical interactions as well as periodic boundary conditions (Section 3.2.3). Therefore, both layers are regarded as being strictly divided by the hydrophobic core in between. Moreover, the percentaged lipid distribution in MembraneEditor is defined separately for each layer, which makes sense in the light of the membrane asymmetry (Section 2.2.3)[33].

Now that the 2D-KP was defined temporarily as the appropriate description for the LPP[34], the different variables of Equation 15 have to be assigned. As mentioned before, the container represents exactly one membrane layer. The different items are the molecules. For

31 It should be mentioned that it is already possible – by using the algorithm-plugin-interface of MembraneEditor in combination with special algorithms – to generate vesicles or multiple membrane stacks without a hydrophobic core in-between (for example with the full version of "The Wanderer", Section 5.2.2.6). In this case, the three-dimensional KP (3D-KP) would be the appropriate description.

32 The MKP is different from the Multidimensional KP mentioned before! The MKP is similar to the bin packing problem, another problem class for packing problems.

33 Of course, in nature interactions between both layers steadily occur, for example in case of Cholesterol flip-flops [BMHM09]. Using the algorithm-plugin-interface of MembraneEditor it would be also possible to model interactions between these layers, following the definition of the MKP.

34 The definition of the 2D-KP will be extended in Section 5.2.2.5. This is also important in the light of another feature of the lipids, namely the shape.

the regular LPP, many items of relatively few different figures or shapes exist [Dyck90]. The frequency of the items is defined by the lipid ratio. The weight w_i is represented by the width (along the x-axis) and length (along the z-axis) of the molecule. Each Molecule is counted as one and added to the overall number of lipids, therefore the value v_i is one. The UKP is the appropriate definition if the according layer contains only one lipid type and for this reason c_i is infinity. On the other hand, if different lipid types are part of the layer (for example 80% phospholipid and 20% cholesterol) c_i is restricted by the user-defined lipid percentages. In this case, a lipid will not be placed into a vacant space if the actual percent value for the according lipid type is equal or larger than the lipid percentages.[34]

Most classical KP are not only depending on two dimensions, the orientation of the items is regularly also fixed or is only altered in 90° steps. A number of packing algorithms are taking advantage of these restrictions by optimizing the runtime [DyFi92, LoMM02]. But nature does not restrict the orientation of lipids. Therefore, a lot of KP solutions are not applicable to LPP. Because the KP is a subclass of packing problems, it is also NP-hard [KaAg10, MaTo90]. A number of various alternative non-exhaustive solutions exist, for example: approximate algorithms, greedy algorithms, heuristic algorithms [KePP04]. For many application cases a single good solution is sufficient – in contrast to an exhaustive one.[35]

How do the traditional membrane modeling methods now apply to the KP? The title of the former section already anticipates the answer by omitting the term “packing” in the title. Of course, the initial problem is the same: an area has to be filled with lipids. In addition, the Replacement Method utilizing spheres as lipid-placeholders can be interpreted as a typical 2D-UKP. This knapsack is unbounded, because the spheres are all of the same type. It is a two-dimensional packing problem, because the movement of the spheres is restricted to the X/Z-plane. But this applies only to the spheres. After the spheres are replaced by the lipids, different methods are used to remove collisions. Methods like the minimization and equilibration, which are especially used for the Replacement Methods, are not typical packing problems, because here already physicochemical methods are applied which are not restricted to geometrical problems. Critics could say that this is still a packing problem but with different or multiple new criteria. But these criteria are not restricted to the shape. During minimization processes they apply to every single atom. Every atom can move into different directions. But packing problems, especially the KP, do take stable shapes or outer boundaries into account. In the case of the LPP these are the stiff lipid structures. The KP optimization takes advantage of the fact that the structure of the lipids is not changed during the distribution process. The criterion “many items of relatively few different figures or shapes” would not be met.

35 In Section 6.2.2 a special case of a rotation-restricted KP will be discussed. There, the implication of this theoretic background on MembraneEditor LPA will be discussed as well as the question, if the finding of an optimal solution is inevitable for LPP.

3.2.7 Packing Densities as a qualitative Criterion

Now that a first overview of MPP has been discussed the question emerges, how the result of such an approach could be verified. For this purpose, the lipid packing density should be shortly introduced here. On a homogeneous membrane layer, the *lipid packing density* D_l per membrane layer l is defined as:

$$D_l = \frac{X \times Z}{N} \tag{16}$$

where X is the width, Z the depth of the layer and N is the number of all lipids on this layer.

In many publications discussing wet-lab-based as well as computational experiments an important factor for judging the quality of a generated and/or simulated membrane is the lipid packing density. For this reason, a short excerpt of different density values will follow.

A value of 42.5 $Å^2$ space for each phosphatidylcholine (PC) was derived from fluorescence microphotolysis [PeBe83] and theoretically confirmed for DLPC monolayers [Olea87]. A similar value of 40.7 $Å^2$ was derived for DPPC bilayers from electron diffraction studies [HCPP75].

Average values of PC, e.g. with unsaturated chains, often range from 65 to 70 $Å^2$ [PESM95]. Hulbert and Else observed cross-section areas of 60 to 70 $Å^2$ for membranes at a natural pressure of 40 dyn/cm [HuEl99]. An area of 62 to 68 $Å^2$ was found in POPC membranes during the liquid-crystalline phase [JóMa07]. Experimentalists compared the area per lipid in monolayers and bilayers and found that the lipid density differences range within ± 1.6 $Å^2$ [WaSt09] or that they are not even observable [HCPP75].

Contrasting with these values, there are also extreme low values like like 26 $Å^2$ for DLPC [PeBe83] or 38.3 $Å^2$ for cholesterol [GhWT73]. Maxima like 134 $Å^2$ for PC have also been reported, derived by thin-layer chromatography [GhWT73].

A good overview on experimentally derived lipid densities using X-ray diffraction studies is found in the work of Shipley [Ship73].

But also the values of the simulation community are found within similar ranges. By using MD simulations, a value of 64.5 $Å^2$ was derived from a simulated DPPC [PZKZ09] and $\sim$54.8 $Å^2$ from a simulated POPC bilayer [JóMa07]. These all-atom-based simulations generate similar results concerning lipid packing density like many coarse-grained approaches [IzVo06, LuVo09, ShDK08, SSRB01]. Values of $\sim$59 to $\sim$69 $Å^2$ for different PC lipids were exemplarily generated by using the MARTINI model [KGUB09].

A MD simulation-based overview on lipid densities was provided by Nagle and Tristram-Nagle [NaTr00].

Obviously, all these values apply to homogeneous layers. Definition 16 can also be used for heterogeneous layers, but the results will be usually only be a rough estimation, depending on the difference of the structural lipid properties. An optimized method to compute the lipid packing density for heterogeneous layers lies beyond the scope of this work.[36]

36 The interested reader is referred to an excellent diploma thesis developed in the context of MembraneEditor by Gunther Lukat [Luka12].

3.3 [FUN] Network Reconstruction and Localization

Now the basics of the mesoscopic as well as the molecular level has been discussed. By now, the reader should have an idea of how a cell model could be visualized using abstract or microscopy-based modeling approaches as well how to work on the molecular level. Now the more abstract level has to be taken into account: how can metabolic interactions – introduced in Subchapter 2.3 – be visualized in an appropriate way? And how can proteins – representing the molecular level – be localized at cell components – representing the mesoscopic level. For this purpose, a set of databases will be introduced with a focus on those databases relevant for this work.

3.3.1 Experimental Methods

The reader should by now already have an idea of how proteins can be localized inside the cellular environment by reviewing Section 3.1.1 – the microscopy – and Section 3.2.1 – the spectroscopy. In this section metabolic pathways will be an important subject. How can these complex maps combining large sets of biochemical reactions be experimentally examined?

One technique was already mentioned before: NMR (Section 3.2.1.3), which is also able to detect specific *(radioactive) isotopes* by their specific nuclear spins. These isotopes are used to label metabolites. A simple approach to elucidate a particular reaction is to feed nutrients incl. the isotope-labeled metabolites to an animal and to examine bodily fluids like blood or urine afterwards for the effects. For traditional approaches like *chemical labeling* this method was used. The problem of the latter method was that the chemical alteration of the metabolites may distort the reactions. Another approach is the adding of metabolic inhibitors which block a metabolic pathway at specific points and analyzing the impact on the specimen or organism. Using NMR techniques, it is even possible to analyze these reactions noninvasively.

Finally, a quite different technique should be introduced: the *DNA microarray* or *DNA chip*. Of course, the DNA also contains most of the information regarding the metabolic properties of an organism. *Transcriptomics* examine the whole set of RNAs which is transcribed by the organism-specific cellular genome. The experimental process usually begins with mRNAs extracted from an organism. These mRNAs are labeled with a specific fluorescent dye. Now the mRNAs are applied to the DNA microarray and the *hybridization* occurs (Section 2.3.3)[37]. The remaining, non-hybridized mRNAs are washed away and now the fluorescent intensity of the spots of the DNA microarray indicate the amount of mRNA bound to a complementary DNA sequence [VoVP06].

3.3.2 Pathway and Localization Databases

In the introduction of this chapter (3) it was already mentioned that a large variety of databases exists. In contrast to the preceding two categories, this applies especially to

37 Hybridization is the generation of double-stranded segments of complementary RNA and/or DNA sequences.

pathway and localization databases. Therefore, the following section covers mainly databases which were used during this work. A detailed technical analysis of these databases lies beyond the scope of this work.[38]

3.3.2.1 Pathway Databases

Here, two relevant databases will be introduced which cover biochemical reactions, especially – but not exceptionally – metabolic pathways.

3.3.2.1.1 KEGG

Surely one of the most acknowledged biochemical databases is the *Kyoto Encyclopedia of Genes and Genomes (KEGG)* [KGSF12]. Since its beginning in 1995 it has been developed in the Kanehisa Laboratories of the Kyoto University and University of Kyoto. It is available at: http://www.kegg.jp/kegg/

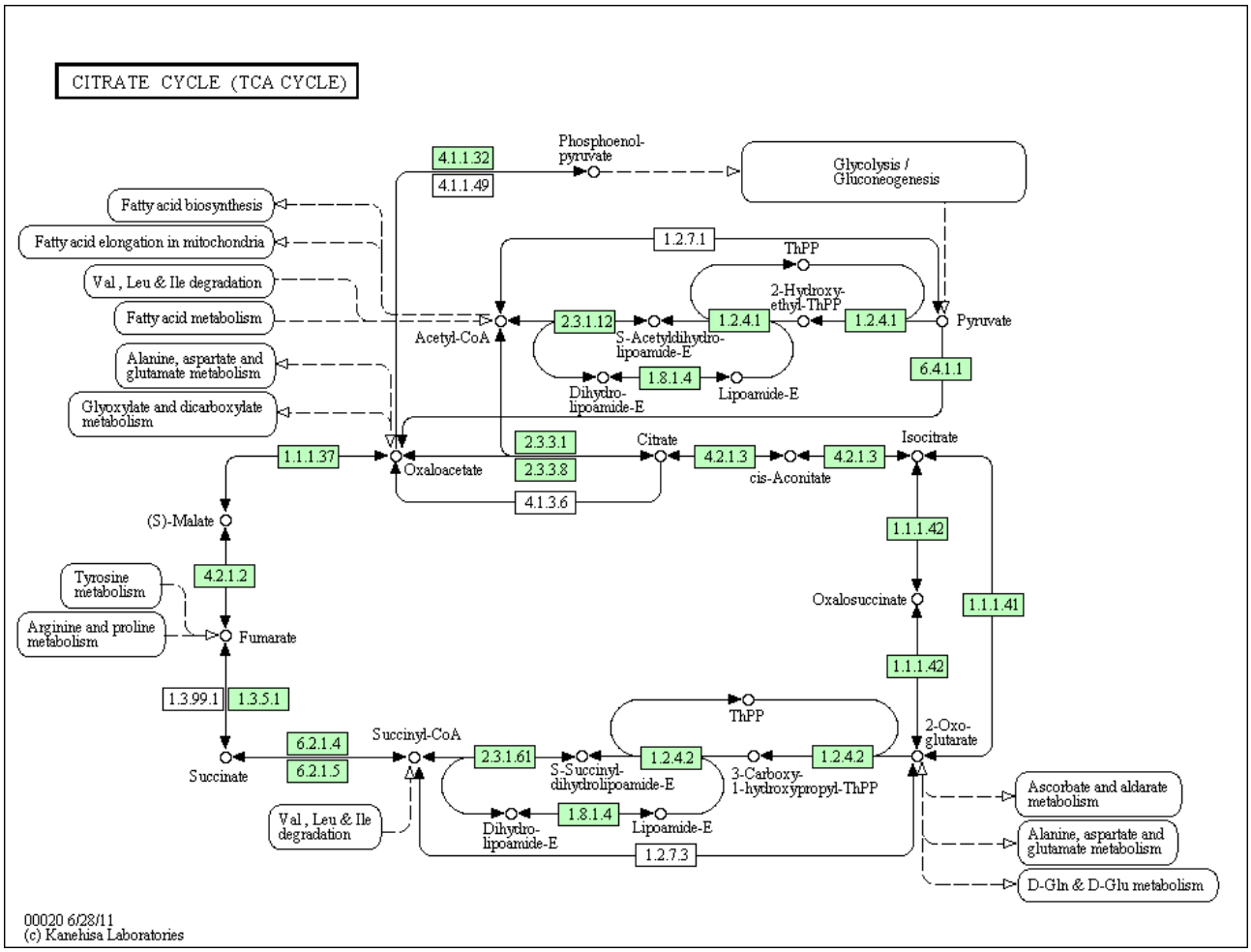


Figure 34: KEGG: the citrate cycle pathway of homo sapiens, hsa00020 (Courtesy of/Copyright © 2011 by Kanehisa Laboratories, source: http://www.kegg.jp/kegg/kegg1.html. Reprinted with permission from [KGSF12])

It includes genomic, chemical as wells as systemic functional information. The database is manually curated. Different completely sequenced genomes were linked to distinct cell-related and/or organism-related functions. One reason for the wide distribution of KEGG

38 A comprehensive overview of the databases discussed here is provided by the doctoral thesis of Benjamin Kormeier [Korm10].

was the free availability of the databases dump. Since the end of 2011, this option was canceled because public funding expired. The database dump must now be paid for. KEGG is now only available free of charge via the website and the web service [Kane12].

KEGG is well known for its two-dimensional, relatively static but interlinked pathway maps. Figure 34 shows the human citrate cycle (Section 2.3.5), with the KEGG identifier hsa00020, where *hsa* is an abbreviation for *Homo sapiens*, and 00020 is the number of the pathway map. KEGG contains many different organisms which are all linked to different versions of the same pathway. For example, mmu00020 is the citrate cycle pathway of Mus musculus (mouse). The enzymes are stated as EC numbers which will be described in the BRENDA Section 3.3.2.2.2. The products and substrates are named by their commonly known identifiers. The direction of a reaction is indicated by arrows. The remaining elements in the map are interrelated metabolic pathways like the glycolysis or the fatty acid biosynthesis.

Section 2.3.4 introduced metabolic pathways and Section 2.3.5 was focusing at the citrate cycle. In the same section Table 4 included two pathways directly interacting with the citrate cycle. Both pathways are found in the map of Figure 34. The citrate cycle is linked to the glycolysis by phosphoenol-pyruvate (out) and by pyruvate (in). The oxidative decarboxylation of pyruvate is not a separate pathway. It is included in the upper half of the shown map. The reactions starting with pyruvate and resulting in the synthesis of acetyl-CoA describe very precisely this pathway.

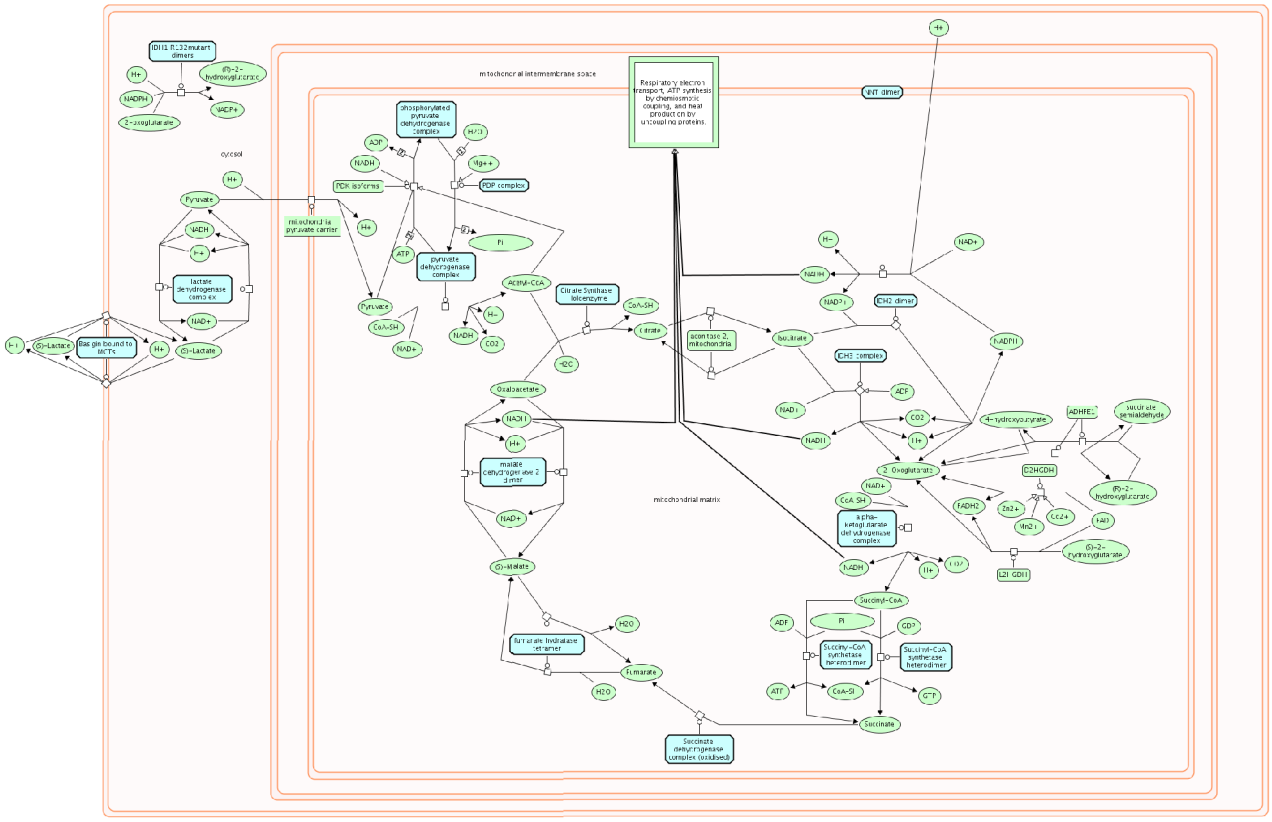


Figure 35: Reactome: the citrate cycle of homo sapiens
(Courtesy of/Copyright © 2003-2010 by Cold Spring Harbor Laboratory (CSHL), Ontario Institute for Cancer Research (OICR) and the European Bioinformatics Institute (EBI). Reprinted with permission from [COWH11])

3.3.2.1.2 Reactome

Reactome is a freely available, curated Open Source project developed in cooperation between the European Bioinformatics Institute (EBI) and different American institutes. It contains a large variety of different pathways, for example, regulatory pathways, membrane trafficking, disease-related or metabolic pathways correlated with gene-/protein-related data. The major focus lies on the human organism, but there are also a number of alternative, comparable organisms available [COWH11, MGGC09, Robe04]. The database is found at:

http://www.reactome.org

Like KEGG, Reactome provides pathway maps as shown in Figure 35. In contrast to KEGG, they are navigable, the user can zoom in to special points of interest. Similar to KEGG, the pathway elements can be clicked invoking related information. Expert users may integrate data inferred from experiments into Reactome data.

3.3.2.2 Localization Databases

Summarizing the information presented thus far, the localization of proteins is a relevant factor for two reasons:

1. the localization of proteins representing the molecular level, as shown in Section 3.2.2.1, and
2. the localization of proteins representing the catalysts, the enzymes of the metabolic pathways, as presented in the preceding section.

Five databases will be introduced which can be applied to this problem.

3.3.2.2.1 Reactome

Reactome was already presented in the previous section. Its unique feature in comparison to the other databases presented here is the fact that it combines pathway data with localization information. Figure 35 shows the citrate cycle and the different protein localizations inside the mitochondrial matrix, the mitochondrial intermembrane space and the cytosol.

3.3.2.2.2 BRENDA and the Enzyme Classification

Brenda (BRaunschweig ENzyme Database) is an enzyme databases developed and curated at the TU Braunschweig. It contains functional, structural and property-related data which is mainly based on manually annotated references from primary literature [SGCS11]. It is available at:

http://www.brenda-enzymes.org

The classification of the different enzyme types follow the *Enzyme Commission number (EC)* classification, which basically consists of four numbers subdivided by a period [Webb92]. An example is shown in Appendix 9.8.

This classification normally does not describe one particular protein, it applies to a number of different proteins which satisfy the criteria of the specific EC definition. One EC number may describe a set of different UniProt-listed proteins defined by the UniProt IDs (see the following section).

Often these EC numbers are linked to localizations which are based on manually curated literature research. The previously mentioned uncertainty also applies to the localization, because it is correlated to a list of enzymes.

In contrast to all other databases introduced here, BRENDA is the only database which additionally features a commercial version with extended content. The freely available database dump is usually an older version of BRENDA.

3.3.2.2.3 UniProt

UniProt is the *universal protein database* for curated and automatic acquired data. It is freely accessible to the public and regularly updated. It is a collaboration started in the year 2002 between the European Bioinformatics Institute (EBI), the Protein Information Resource (PIR) and the Swiss Institute of Bioinformatics (SIB). The data contained is based on accurately annotated protein sequence data which is linked to a large number of external databases like the previously discussed BRENDA, Gene Ontology, Reactome as well as PDB (Section 3.2.2.1.1) [DHAS12, Unip10]. It is accessible at:

http://www.UniProt.org

It consists of different sub-databases. In the context of this work, the *UniProt Knowledgebase (UniProtKB)* is the relevant information source. It contains expertly curated, non-redundant data in the Swiss-Prot section (originally from the SIB) as well as automatically annotated and unreviewed data in the TrEMBL (originally from the EBI) section.

Often proteins included in Uniprot are correlated with localizations. They are found in different sections of the protein information page, for example:

- General Annotation (Comments)
  - Subcellular location
- Ontologies
  - Keywords
    - Cellular component
- Gene Ontology
  - Cellular component

The General Annotations include also the references to the found localizations. The terms may be a cell component, an intra-compartmental location or a sentence describing location-related facts.

3.3.2.2.4 The Gene Ontology and the Redundancy of Terms

The Gene Ontology (GO) contains gene-related protein information in conjunction with structured, controlled vocabularies. The provided ontologies contain the so-called GO-terms which represent a standard-widely accepted in the Bioinformatics community [ABBB00, CKSA12]. GO can be found at:

http://www.geneontology.org

The most important ontology in the context of this work is the “cellular component”. To

give an example of the relevance of this approach, the cellular localization “plasma membrane” should be examined. It follows an excerpt of terms linked to this localization provided by the previously introduced databases:

- BRENDA
  - cell membrane
  - plasma membrane
  - cytoplasmic membrane
  - cell outer membrane
- UniProt
  - associated with the synaptic plasma membrane (by similarity)
  - integral to plasma membrane
  - intrinsic to internal side of plasma membrane
  - localized on the cell surface
- Reactome
  - integrin cell surface interactions

Obviously, a large variety of terms describes the same localization. Therefore, special distinctive vocabularies are needed like those provided by GO to avoid redundancy. In this case, the major GO term is “plasma membrane” having the GO identifier “GO:005886”. The terms “cell membrane”, “plasma membrane” and “cytoplasmic membrane” are all directly correlated to this term.

But the other terms are more specific, so the following GO-terms have to be added:

- GO:0005887: integral to plasma membrane
- GO:0009279: cell outer membrane
- GO:0031235: intrinsic to internal side of plasma membrane

And what about the fact that these terms should also be associated with GO:005886? Of course, GO takes also these hierarchical dependencies into account. Figure 36 shows the GO Graph View for GO:0005887. It contains also the terms GO:005886 and GO:0031235 and shows its spatial interdependencies: “integral to plasma membrane” → “intrinsic to plasma membrane” (→ “plasma membrane part”) → “plasma membrane”.

However, for three terms no GO-identifiers were found:

- localized on the cell surface,
- associated with the synaptic plasma membrane (by similarity), and
- integrin cell surface interactions.

This fact shows that the creation and curation of vocabularies is an ever ongoing effort.

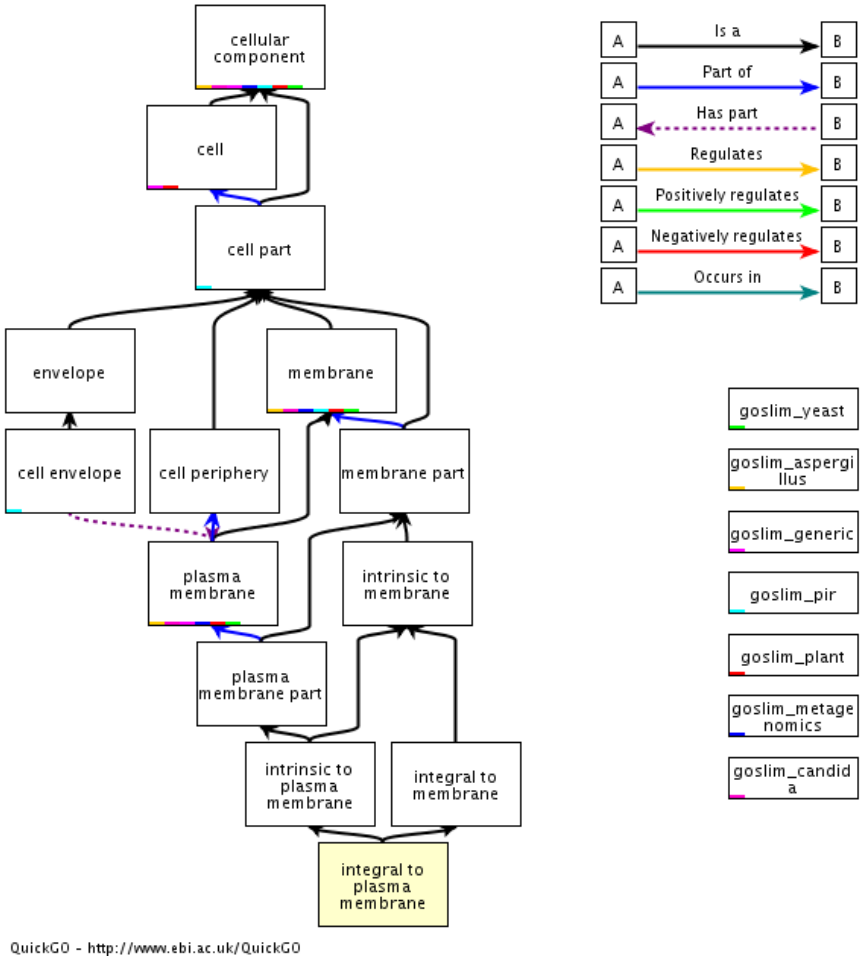


Figure 36: Gene Ontology: Graph View for the GO-term "integral to plasma membrane" (Courtesy of/Copyright © 2012 by The Gene Ontology, http://amigo.geneontology.org. Reprinted with permission from [11g])

3.3.2.2.5 SUBA

SUBA is the *SUB-cellular location database for Arabidopsis proteins developed at the the University of Western Australia.* The system consists of a database which is accessible – just like the preceding discussed approaches – via a website interface:

http://suba.plantenergy.uwa.edu.au

But whereas the other databases are not restricted to localization information nor a special species, SUBA explicitly provides protein localization data for the plant *Arabidopsis thaliana* by uniting the information of various heterogeneous databases.

Moreover, SUBA provides a search form which can be used to enter queries to narrow the search criteria. For example, it is possible to search for proteins located at one of 13 distinct cell component types based on experimental results or on predictions [HVTS07].

3.3.3 Data Integration

Different potential data sources have been introduced. Now the question emerges, how this different data sources can be accessed. Of course, it is possible to manually visit all websites, collect the data and try to evaluate it afterwards. But this process would be very time-consuming for the user. Therefore, another approach should be introduced, integrating

data into one single information source: the data warehouse. In addition, databases do not represent the exclusive option to obtain localization data. Text mining is another solution which will be introduced in the next section but one.

3.3.3.1 The Data Warehouse: BioDWH and DAWIS-M.D.

Databases like KEGG often provide a web service which can be accessed directly via Java (Section 3.4.3.1). Using these web services, the databases can be queried and information can be acquired directly by using an Internet connection. But there are two major drawbacks of these methods: first, one´s own software development strongly depends on third party software. Therefore, potential changes have to be implemented anytime, the reliability of the web service cannot be guaranteed and it is normally not possible to successfully propose changes and improvements. The second major drawback is the performance: these web services are usually very slow, especially if a larger amount of data has to acquired via Internet.

The solution to this problem is a *data warehouse*. This contains a collection of different databases while avoiding ambiguity; it links the databases.

The *Bio DataWareHouse (BioDWH)* is one such solution developed in the Bio-/Medical Informatics Department of Bielefeld University. It consists of different tools, combining techniques like Java, Web Services and XML-Parsing. It was developed independently from the underlying database management system (like MySQL, Oracle) and is query able directly via MySQL queries (only in the Intranet) or via web service connection (from the whole Internet) [TKKH08]. The project which features a number of GUI-based administration tools is available at:

http://sourceforge.net/projects/biodwh/

Based on BioDWH, the *Data Warehouse Information System for Metabolic Data (DAWIS-M.D.)* was developed. It uses MySQL as its underlying database management system and contains a variety of different databases which are related to metabolism. Particularly all databases discussed in Section 3.3.2.2 are found in this system [HKTJ10]. A web front-end exists which can be used to gain access to the information in the data warehouse (Figure 37):

http://agbi.techfak.uni-bielefeld.de/DAWISMD

3.3.3.2 ANDCell/ANDVisio

The topics of the previous sections converged in the aspect of being related to data integration. The underlying databases of DAWIS-M.D. predominantly contain curated data. Naturally, the curation of data is a time-consuming task and therefore much information is not included since it has not been extracted from published literature thus far. To overcome this obstacle, the automatic extraction of knowledge is an important task of Bioinformatics tools. As mentioned before, most of the information included in databases like UniProt is based on published literature. As a logical consequence, text mining is an appropriate approach to extract information from a textual corpus.

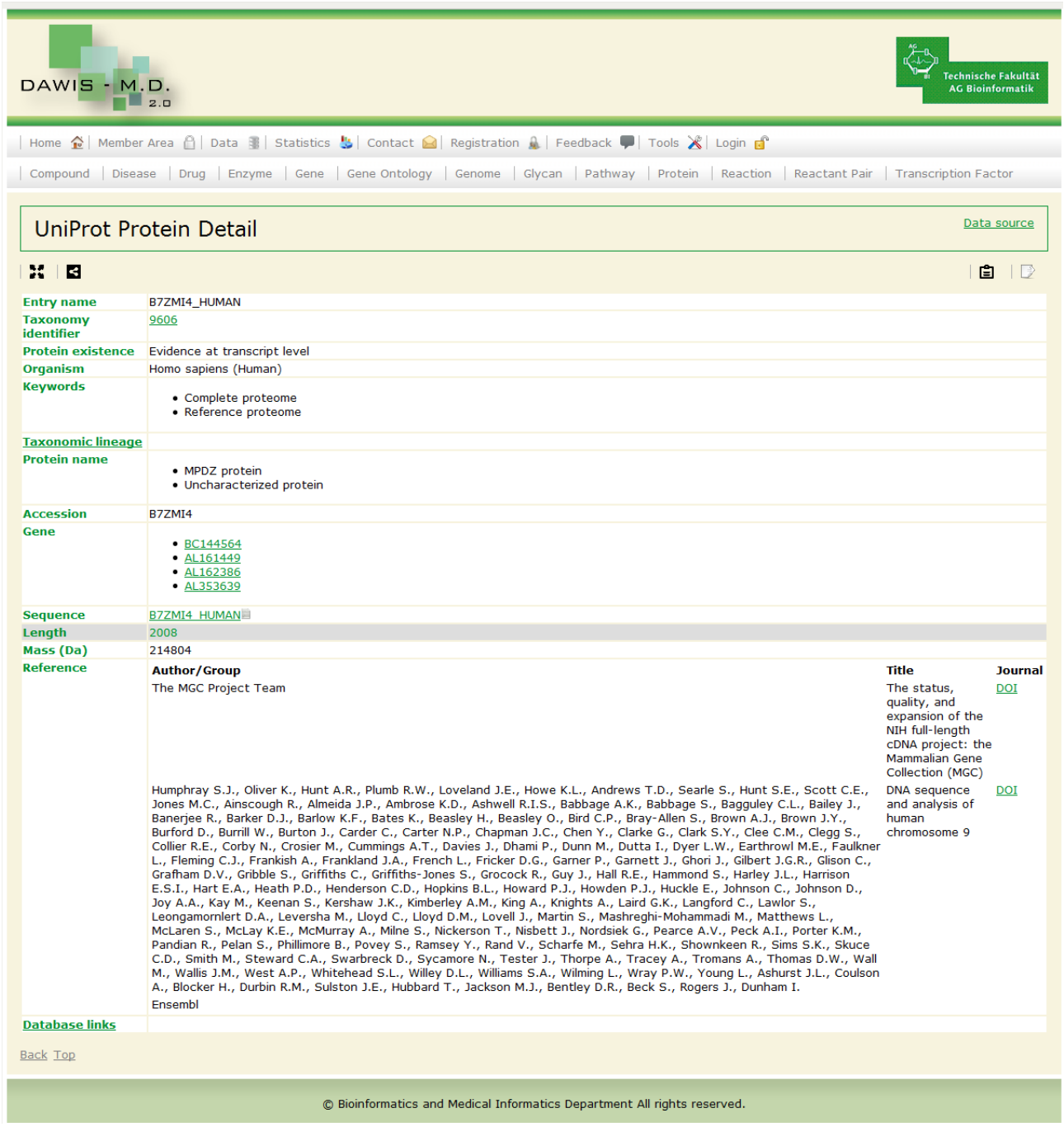


Figure 37: DAWIS-M.D.: Result Page of the MPDZ protein

An extensive source of biomedical knowledge is the *PubMed database*, containing more than 21 million abstracts of published literature [12c]. And exactly this is the source which is utilized by the *ANDCell (Associative Network Discovery)* database. Based on text mining results, different interaction types are extracted:

- physical protein interactions with other proteins, low-molecular compounds or DNA,
- the formation of molecular complexes,
- biochemical processes and reactions,
- regulation of gene expression or protein activity and stability, and/or
- associative bonds between proteins, genes, compounds, cellular components, and diseases.

The ANDCell database was originally developed at the Institute of Cytology and Genetics in Novosibirsk. There is also a commercial version which is distributed by PBSoft Ltd.:

http://www.pbiosoft.ru

The complete ANDCell system also contains a number of other databases. Some of these databases are part of DAWIS-M.D. Therefore, in the context of this work, only the PubMed-based data from ANDCell is relevant.

Because the data is not curated, ANDCell always provides the text fragments on which a computed association is based.

For ANDCell there also exists a front-end, *ANDVisio*, a tool for two-dimensional generation and visualization of associative networks which is shown in Figure 38 [DeAI08, PYDK11]. This application features an XML-compatible format (.and) which can be used to export the generated networks.

In our previous work it was already shown, that data integration approaches like DAWIS-M.D. and text mining approaches like ANDCell can be used for the verification and extension of knowledge and for the identification of false positives. The data of curated databases is naturally more reliable, but the automated extraction method often finds additional results which are also valid [STKH10].

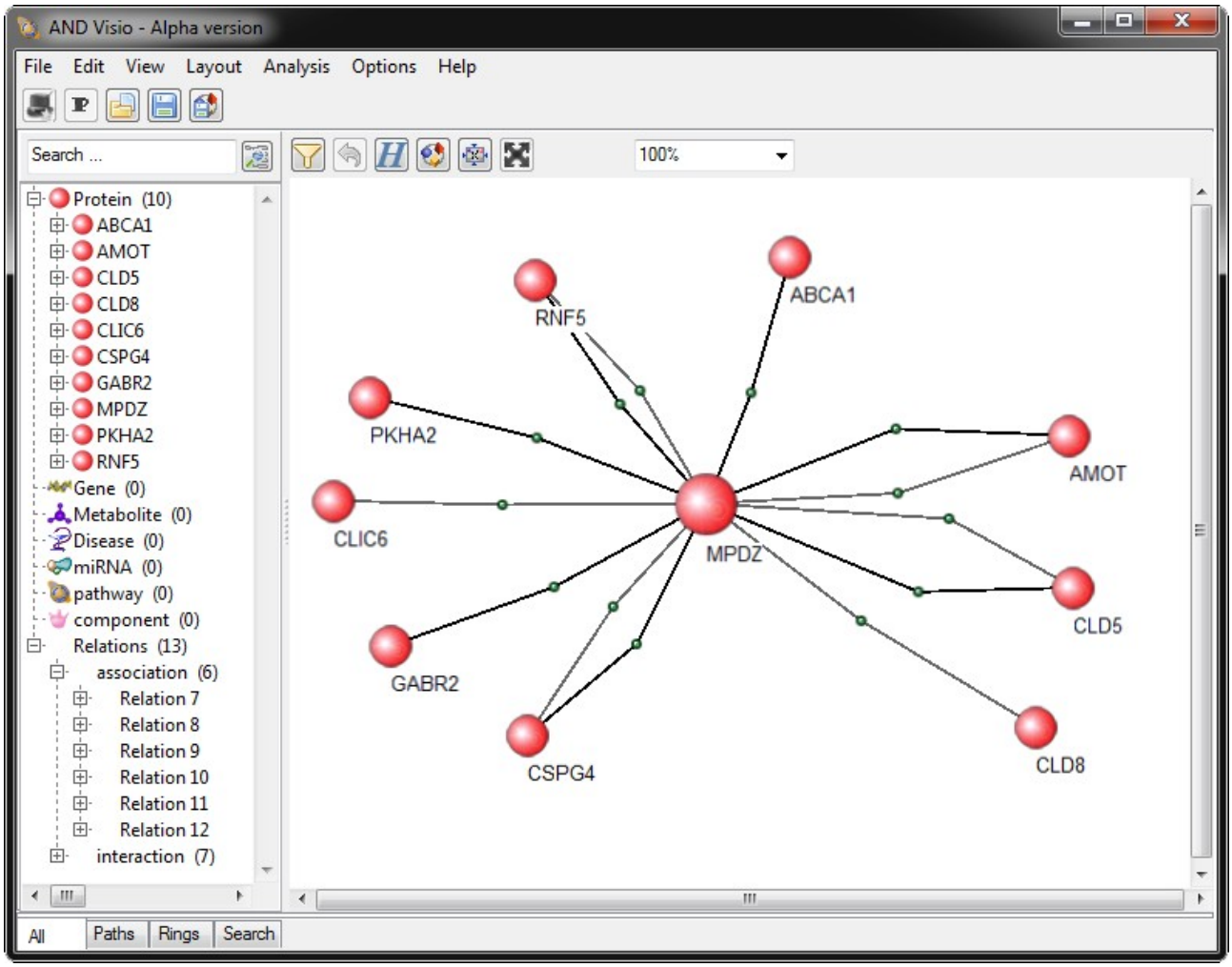


Figure 38: ANDVisio: Visualization of a MPDZ protein-protein interaction network

3.3.4 Network Mapping Problem

After discussing the localization of proteins, the question develops, how the protein-related/metabolic networks can be combined with the mesoscopic level, that means, how they can be correlated with the cell components. Because the cell components are three-dimensional objects, solutions have to be found to place the nodes – the proteins – on top of these cell components, with regard to the edges connecting the nodes – the reactions (this will be more thoroughly discussed in Section 5.3.2.2.1).

For internal use in this thesis, this problem will be referred to as the *Network Mapping Problem (NMP)*. Externally, it should be termed the "Geometric Network Mapping Problem" to distinguish it from regular network mapping problems which usually apply to the domains of network-connectivity and Internet architecture [HoLZ08].

The NMP will be solved in three consecutive steps, each solving a sub-problem of the Network Mapping Problem: the Node Distribution Problem, the Network Layout Problem and the Node Mapping Problem.

3.3.4.1 Node Distribution Problem

First, the *Node Distribution Problem* has to be discussed: the nodes have to be distributed equally onto the surface of a simple shape, for example, a sphere. The distance from the center of the sphere to the node on the surface is therefore equal among the nodes. In addition, the edges are ignored in this first step. These two aspects simplify the problem. But how can this problem be formally defined?

Remembering the Membrane Packing Problems (MPP) discussed in Sections 3.2.4 and 3.2.6, some parallels emerge; again there is a container-like object – the sphere – onto which different items of the same type – the nodes – have to be distributed. In Section 3.2.6 the term "container" was used to define the two-dimensional geometry which was the base for the distribution. In the three-dimensional context, the term *object* should be more appropriate and will be used hereafter. But of course there are also important differences.

The first aspect is the three-dimensional surface of the sphere in contrast to the two-dimensionals of the rectangular membrane patches. Therefore it is not possible to apply similar packing solutions to this problem. At any rate, it will be shown that the addition of a third dimension is in many cases not very complicated.

But the second aspect is even more reasonable. MPP are aiming at the minimization of free space (OPT_LPP_MIN_AREA, OPT_LPP_2D_AREA). The optimal solution of packing a number of nodes onto a sphere is – in this case – the equal distribution on the surface: the *Node Mapping Problem*.

In the context of this work (extending the termini of Section 3.2.4), this should be defined as

- *OPT_NMP_MAX_AREA:* The optimum is defined by the maximal possible area per node.

The maximal possible area per node simultaneously is the prerequisite for an equal distribution of the nodes on the surface area of the object.

The surface of a sphere consists of an unlimited number of points, vectors which all have the same length starting from the origin of the sphere. The problem is now to find discrete points which can be taken into account for solving OPT_NMP_MAX_AREA. There are a number of approaches which can find a good solution for this problem. Some of them will be discussed in Section 5.3.2.5.1.

3.3.4.2 Network Layout Problem

Now that OPT_NMP_MAX_AREA is defined, the *Network Layout Problem* has to be addressed by also taking the connections between the nodes into account. This has to be done, because in the case of a metabolic pathway it would not be reasonable if two interconnected nodes were located on the opposite sides of the object: the sphere.

Again, an appropriate optimum definition has to be found. To follow the initial optimum definitions of Section 3.2.6, a related metaphor should be chosen. A well-known solution for this kind of problems are *force-directed layouts* which were first introduced by Thomas M. J. Fruchterman and Edward M. Reingold [FrRe91]. The basic idea of these approaches is relatively simple. The principle used as an appropriate metaphor to solve this problem is the magnetism (which is also known in the molecular context, for example: the van der Waals and covalent forces, Section 2.2.1.3). Between all nodes of a graph exist repulsing forces. But in addition all interconnected nodes are attracted to one another. Resulting from this mechanism, the connected nodes are approaching each other, whereas the repulsing forces prevent node collisions. Therefore, this layout tries to reach a state close to equilibration, resulting in the following optimum definition:

- *OPT_NMP_ENERGY:* The optimum is an equilibrated layout.

The solution to this problem in the context of this work will be examined in Section 5.3.2.5.2.

3.3.4.3 Node Mapping Problem

But after solving OPT_NMP_MAX_AREA and OPT_NMP_ENERGY, still this is not the perfect position for the protein node, because the cell components are usually not shaped like a sphere. Instead of this, the shapes are irregular. Therefore, the resulting layout has to be mapped onto a cell component. This is the *Node Mapping Problem*.

- *OPT_NMP_POS:* The optimum is defined by the correct network positioning in relation to the center of the cell component.

This problem will be solved in Section 5.3.2.4.

3.4 [MES+MOL+FUN] Summary

Concluding the technical background chapter, a few aspects should be now summarized: a short review of the different levels associated with this work is followed by an overview of all mentioned databases, resulting in the introduction of an appropriate programming language.

3.4.1 Overview of Levels

Figure 39 shows the correlation of the mesoscopic and molecular level with the different microscopy and spectroscopy techniques. It should be emphasized that there is no strict limit between the two levels. For example, the electron microscopy is already able to scan specimens with a resolution within the single-digit nanometer range which is quite close to the resolution where atoms can be differentiated (Section 3.1.1.2). The hybrid technique electron crystallography already approaches the single-digit Ångstrom resolution (Section 3.2.1.2).

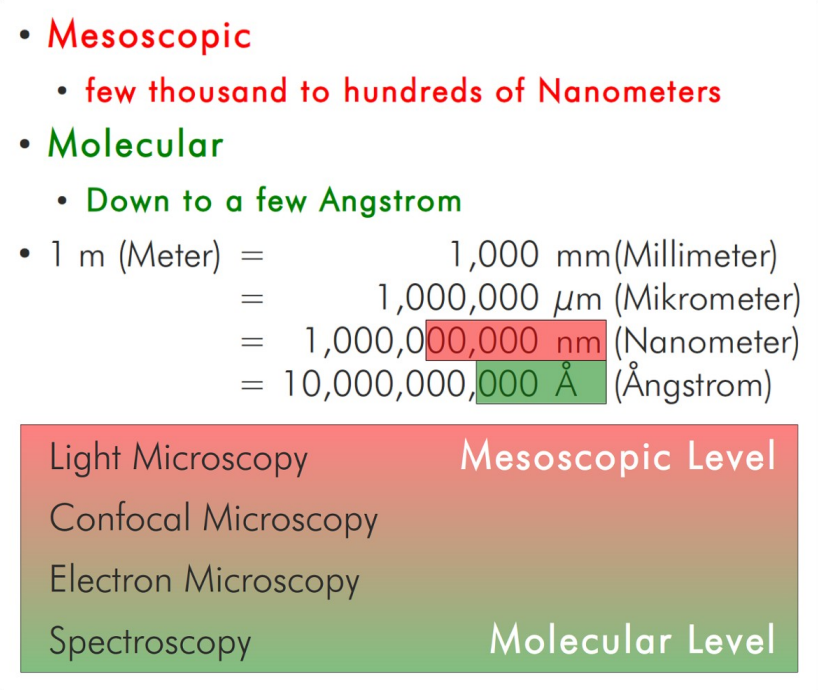


Figure 39: Interrelation of the Mesoscopic and the Molecular Level in this Work

3.4.2 Overview of Databases

Table 6 shows an overview of all databases discussed and the according categories found in the Nucleic Acids Res. mentioned above, in comparison to the ones found in this work. Comparing the categories it shows that the definitions do not completely match, because the application of some databases is specialized to some limited datasets.

Two examples: [MOL] is completely described by databases listed as “Structure Databases” in Nucleic Acids Res., here the comparison matches. But one database from this set is also found in the category “Metabolic Pathway”: Klotho. [FUN] is more differentiated: Focusing, for example, at Reactome, in the context of this work it is used only for the localization. But as can be seen by the Nucleic Acids Res. Category, it is mainly defined as a “protein-protein interactions” database.

Obviously, many databases are needed to create a Virtual Cell in an appropriate manner. But there was no need to create new databases. Therefore, the complex database creation process and different database types will not be discussed in this work. But it should be mentioned in which way the databases are accessed: by using techniques like *MySQL* or *SOAP*. In the following sections, the focus will be on the type of data stored in the covered databases.[39]

39 A comprehensive overview of molecular databases interesting for the scope of this work, their background and theory, is provided by the thesis of Benjamin Kormeier [Korm10].

	ANDCell	Avanti Lipids	BRENDA	CCDB	CHARMM Database	Chemistry Mol. Models	DisProt	EMDB	Gene Ontology	HIC-UP	KEGG	Klotho	OPM	PDB	PDBTM	Protein Modal Portal	Reactome	SUBA II	UniProt
Nucleic Acids Research Categories																			
Genomics Databases																			
General genomics databases											X								
Genome annotation terms, ontologies and nomenclature									X										
Metabolic and Signaling Pathways																			
Enzymes and enzyme nomenclature			X																
Metabolic Pathways											X	X							
Protein-protein interactions																	X		
Proteomics Resources																			
General sequence databases																			X
Protein localization and targeting													X					X	
Structure Databases																			
Protein Structure							X							X	X	X			
Small molecules										X		X							
CELLmicrocosmos Categories																			
Mesoscopic Level (MES, 3.1)																			
Microscopy Databases (3.1.2)				X				X											
Molecular Level (MOL, 3.2)																			
Protein Databases (3.2.2.1)							X						**X**	**X**	**X**	X			
Lipids Databases (3.2.2.2)		X			X	X				X		X							
Membranes Databases (3.2.2.3)					X														
Functional Level (FUN, 3.3)																			
Pathway Databases (3.3.2.1)	(X)										**X**								
Localization Databases (3.3.2.2)	**X**		**X**						**X**								**X**	X	**X**
DAWIS-M.D. Databases (3.3.3.1)			**X**						**X**		**X**						**X**		**X**

Table 6: An overview of all databases used in this work
Included is a comparison with the categories found in Nucleic Acids Research Database Issue 2012 [GaCo11]. The respective sections are in brackets. The bold crosses indicate those databases directly accessible with different CELLmicrocosmos tools.

3.4.3 Programming Tools

The theoretical basics of the data basis for a Virtual Cell have been introduced. Now the digital (cyto-)skeleton should be shortly introduced: the applied programming language.

One important requirement for the developed software was the independence from other – especially commercial – software packages and tools. In addition, the installation process should be very simple, supported by web technologies and the tools should run on many different operation environments. In addition, the preceding section mentioned the databases needed for the Virtual Cell environment. The programming language has to also support the database access via MySQL. Last but not least it is important that the language is also known in educational environments, to allow students to participate in the development of new tools.

As a logical consequence, Java™ from Sun microsystems® (today Oracle®) was chosen as the appropriate development environment.

3.4.3.1 Java

Java is an object-oriented programming language. A great advantage in comparison to languages like C++ is the fact that it was developed right from the beginning around 1993 as a web-technologies-oriented programming language. Resulting from this initial decision, it is system-independent, which means that it is compatible with all systems supporting Java, like Microsoft® Windows®, Linux or Mac™ OS X. Simultaneously it implements many aspects which were already established by C++. Java programs are able to run within common web browsers like Microsoft® Internet Explorer® or Mozilla Firefox as Java applets. But for CELLmicrocosmos projects, another technology was used: Java Web Start. Web Start programs can be installed directly from the browser and can be run independently from an Internet connection after the installation process is finished. In addition, these programs can be updated automatically on-line if the original resources in the web were actualized.

Multithreading, exception-handling, garbage collection and dynamical memory management are additional features, which made Java the first choice for this project. In addition, it is much easier to learn than similar languages [12d, LoMü01].

3.4.3.2 Java 3D

Graphic Programming in three dimensions is usually done by OpenGL, a language directly communicating with the graphic hardware. Although this language is not very complicated, it is a quite time-consuming task to generate a complex scene. In addition, it is usually used in conjunction with C/C++ and therefore the compiled binary data is not system-independent.

To overcome these obstacles, Java 3D was established and first released in 1998. Since 2008, Java 3D was released under GPL2 with linking exception, enabling the use in commercial as well as open source projects.

Java 3D is a communicating language between the original graphic languages like OpenGL (or DirectX on Windows systems) and Java. The compatibility with Java inherits a lot of

advantages like the Web Start technology which can also be used with Java 3D. Because of its direct access to OpenGL or DirectX, visually challenging results can be achieved by using the advantages of the graphics hardware.

Java 3D is based on a *Scene-Graph*-concept. Several nodes are connected to a tree structure as shown in Figure 40.

The starting point the *VirtualUniverse*, the three-dimensional environment where all objects are located. *Locale* is the initial localization node, the reference point in the universe. The *BranchGroup* contains and groups different objects. They are especially important if objects should be manipulated – for example, added or removed – during runtime. The BranchGroup on the left contains a *Shape3D* object. These are three-dimensional objects which in the following chapters will represent, for example, the cell components which could be imported by using the VRML97-import-module of Java 3D. In addition, this Shape3D could be substituted by a primitive object like a sphere or a cube. The Shape3D object features two additional nodes: *appearance* and *geometry*. These terms were already mentioned in conjunction with the VRML97-content and reflect exactly the same functionality (Figure 25, Section 3.1.3.4). The right BranchGroup contains first a *TransformGroup*. This consists of a 4x4 matrix, encompassing a 3x3 rotation matrix reflecting the orientation of the appended object, and a vector (x,y,z) containing its translation. In this case, the appended object is the *View Platform*. All objects communicating with this object on the same level, the *View*, *Canvas3D* and *Screen3D* represent the user´s camera position and field of view. Finally, the *Physical Body* as well as the *Physical Environment* can be important to define the left and right eye or ear position. The user´s avatar and its reference to the three-dimensional environment are described by this variable [12e, Bouv99].

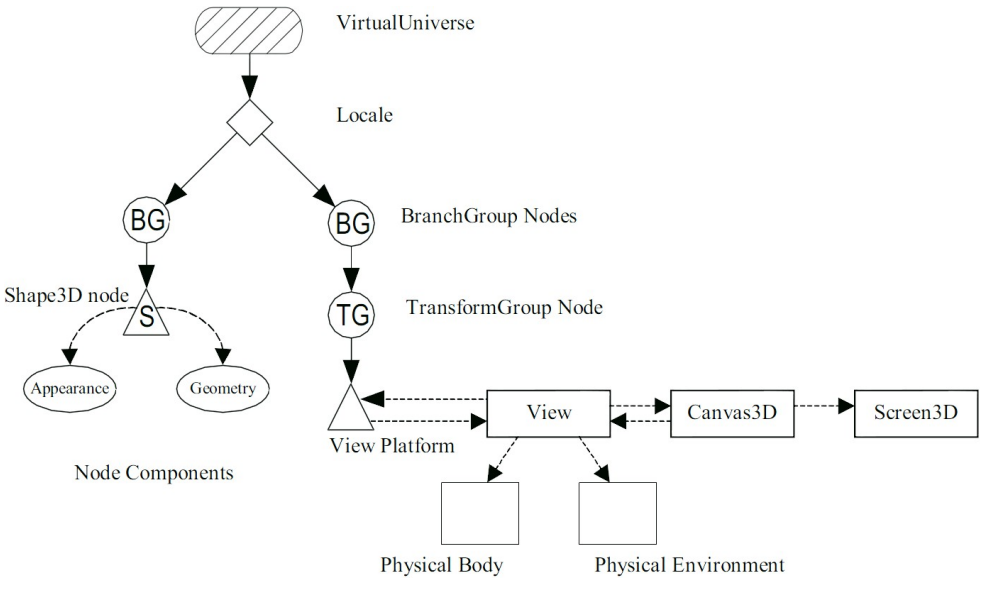


Figure 40: Java 3D: the scene graph concept (Courtesy of/Copyright © 1999-2000 by Sun Microsystems, Inc. [Bouv99])

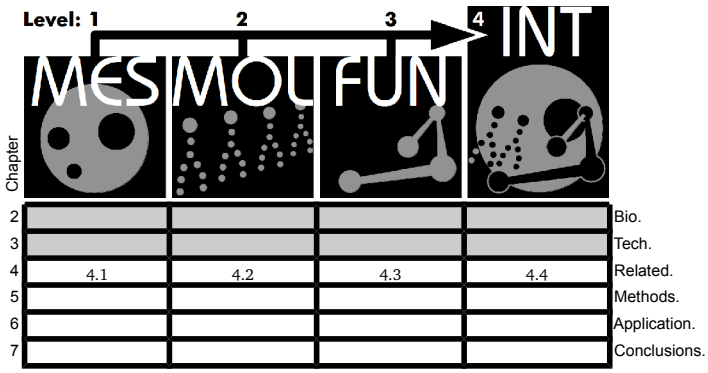
Of course, this was only a short overview. The explanation of all objects relevant for the generation of a Virtual Cell universe such as light, sound or behaviors lies beyond the scope of this thesis.[40]

40 There are many good books on Java 3D, some are available on the internet ([Davi05, Davi07]. Another nice and freely available German book from Michael Pfeiffer is very feasible for beginners and was often used as the starting point for Java 3D programming in different CELLmicrocosmos projects [Pfei11].

4 Related Approaches

The biological aspects of the cell have been discussed as well as the way to acquire and administrate this knowledge. In this chapter the already published area in the related fields will be introduced. These fields are Cell Modeling (Subchapter 4.1), Membrane Modeling (Subchapter 4.2) and Network Modeling (Subchapter 4.3). Finally, in Subchapter 4.4, the missing link between the aforementioned areas will be thematized: mesoscopic level.

Here, only published and/or established projects will be introduced. Of course, this work does not claim to cover all projects published worldwide in this field, but it tries to introduce most of the relevant work in this research area.



4.1 [MES] Cell Modeling Tools

The term *cell modeling tools* brings together two quite different approaches: Applications based on *visual cell modeling* and those based on *mathematical cell modeling*. The latter approach partly includes visual cell modeling techniques in order to enable the linking of biological and simulation data. But one common feature of all programs is their affiliation at the mesoscopic level.

4.1.1 Visual Cell Modeling

Visual cell modeling is the approach to model cells based on microscopic datasets. The grade of realism may vary, depending on the intention of the cell visualization. It may be more abstract to indicate special intracellular relationships, but it may also be an approach strictly reflecting the view as seen through a microscope.

4.1.1.1 CELLmicrocosmos 1.0 DisplayCell

Initially in the year 2005/2006, Amira was used to develop the first interactive Virtual Cell application of the CELLmicrocosmos project. This was very important for its future development: a sketch for the functionality needed by a Virtual Cell environment [Somm06]. By using the developer edition of Amira, a module called *DisplayCell* was developed by using C++ and the TCL/TK scripting interface (Section 3.1.3.1).

This project was an educational approach: it allowed navigating through a Virtual Cell environment with the standard navigation techniques offered by Amira. A number of cell

components were included which could be explored. Each of them was correlated with an information text which was available as an auditive as well as a textual version. In addition, different small animations were included which showed different cell components as well as a membrane generated with the first version of MembraneEditor (2.0).

For three reasons, the future development of DisplayCell was discontinued:

1. It depends on Amira and Amira is quite expensive: every institution which would like to use DisplayCell, would have to buy a license.
2. The development and installation process is quite complicated and only partly achieved results of moderate quality (this applies especially to the animation mode).
3. Most of the source code of Amira is not easily accessible.

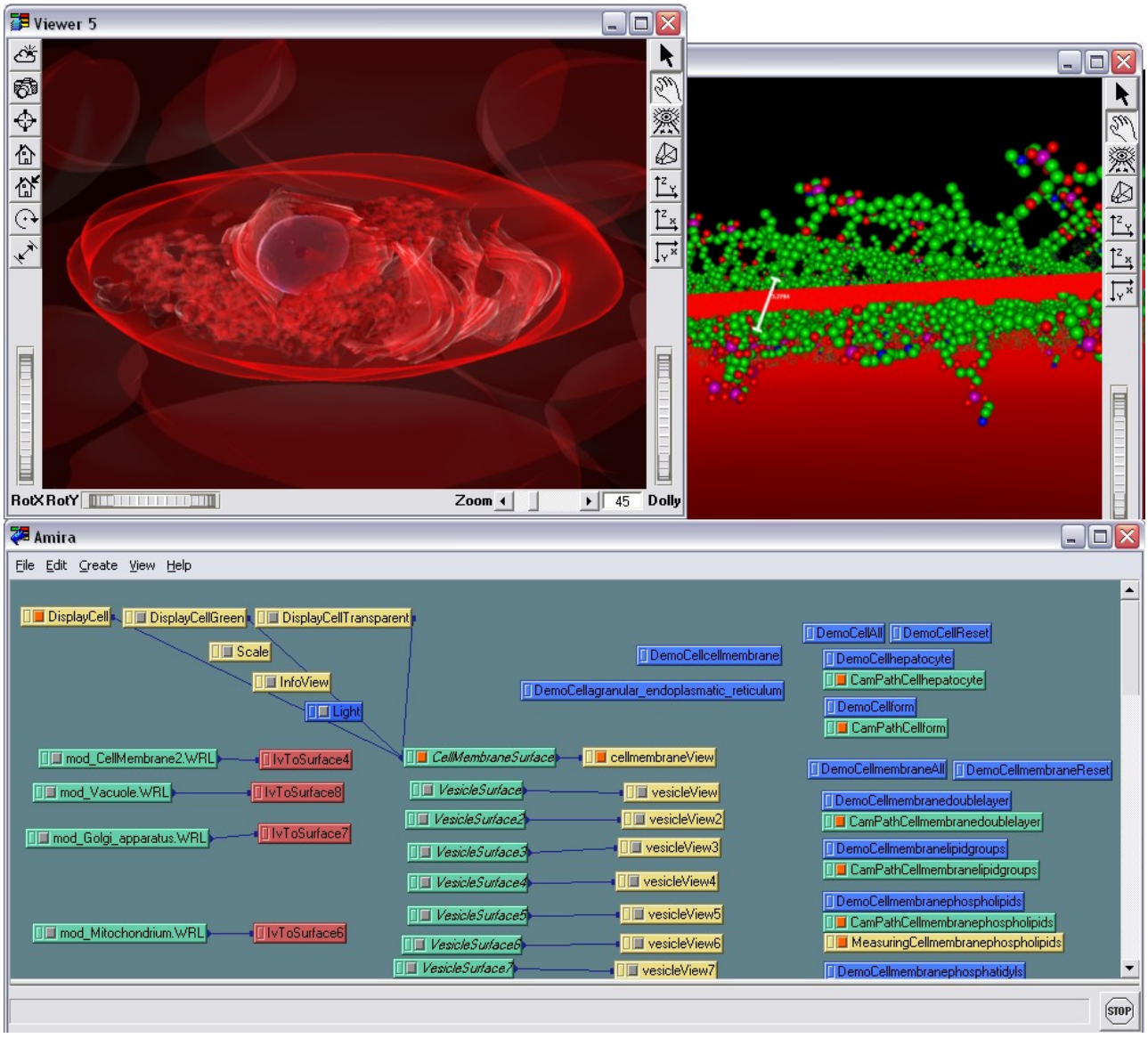


Figure 41: CELLmicrocosmos DisplayCell in Amira

4.1.1.2 Meta!Blast

Meta!Blast is an approach between all these differing poles. It is specifically designed to be attractive for video-game-playing students – quite a large target audience today. The main idea is to combine the process of learning with gaming experiences, also known as edutainment.

The project is being developed at the Iowa State University and funding has been provided by different organizations. Many professions are contributing to the software: biologists, computer scientists, designers, art directors and writers. In addition, all those active persons are being supported by different teachers helping to improve the software.

The subject of Meta!Blast is a plant, linking the mesoscopic to the anatomic level [WBDK10]. Until now, only a demo game has been available. It can be downloaded from the website. A virtual plant cell can be explored on the mesoscopic level, containing different areas with questions which have to be answered by the student (Figure 42). Still, complex animations are missing, but the preview pictures on the website look promising. In the future there will also be parallax scrolling game elements more similar to an action game which will be used to add a link to the molecular level. Obviously, at present the merging of both levels has not been planned [11h].

The first concept of this gaming experience was already published in 2007 and initial ideas of this approach can already be found in publications regarding the MetNet3D project, discussed in Section 4.3.3.2. Therefore, the outcome will probably be amazing and an important contribution to communication between science and school [CHDW07].

A powerful feature of Meta!Blast is the support of NVIDIA® 3D Vision®, therefore 3D Stereoscopy can be used on consumer computers supporting this technique (see also Section 5.3.2.11).

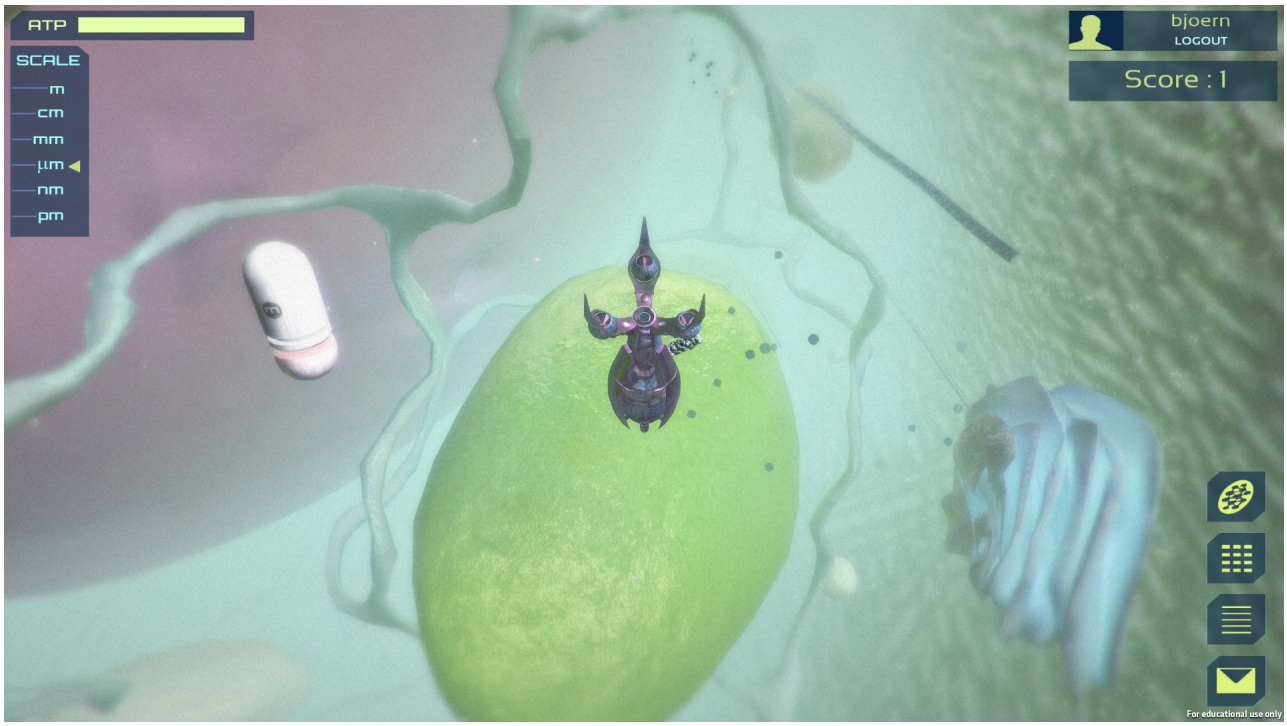


Figure 42: Meta!Blast: inside the cell
The vehicle used to navigate through the plant cell confronted with a simple question [11h]

4.1.1.3 The Visible Cell® Project

The *Visible Cell*® Project is located at the University of Queensland. The aim is the creation of a three-dimensional Virtual Cell model based on microscopic data sets of mammalian beta cells. Special methods have been developed to extract the needed cell lines and to use electron tomography at a resolution of ~5 nm.

The main tool used to support this project is called *Illoura™* [MCNW09]. Similar to Amira (Section 3.1.3.1), this tool is used by the Visible Cell group for image segmentation of 2D, 3D and 4D data sets. Because the aim is the creation of a multi-scale and multi-resolution approach, different data sets should be combined providing different cell regions. However, until now it is not possible to see any published results of this project and there is only an older version of Illoura available for download which seems to be not very functional yet (version 0.9, as of 31.07.2012).

In the meantime, an overview of cell segmentation tools and other approaches related to Illoura™ can be found in [WSBB10].

4.1.1.4 AutoFill/AutoCell

The visualization of cellular environments by using complex rendering tools like Autodesk® 3ds Max and particularly Autodesk® Maja®, Maxon Cinema 4D or the Open Source tool Blender gains steadily more attention. Still it is a problem to generate cellular structures. Normally, a number of scripts are used for their generation which are customized by the user for their particular purpose.

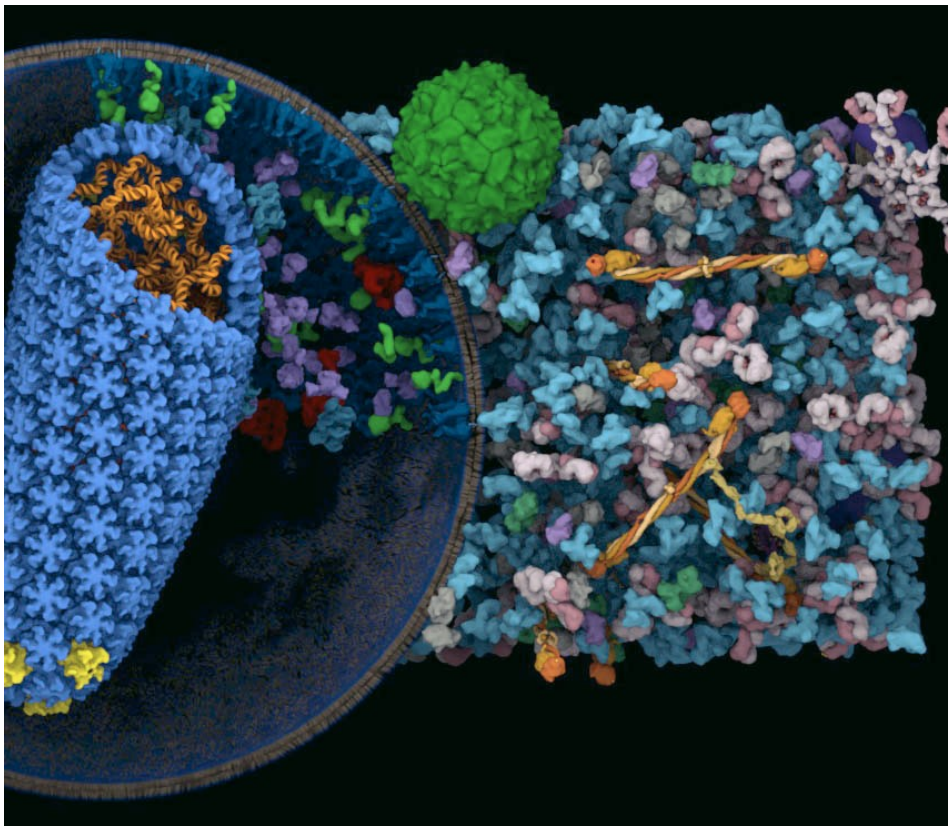


Figure 43: AutoFill/AutoCell: A HIV spheroid
This approach fills an octant of a HIV spheroid featuring a size of 140 nm with HIV-only components (Courtesy of/Copyright © 2011 by Graham Johnson, autofill.scripps.edu. Reprinted with permission from [John11])

The approach of *AutoFill* tries to overcome this obstacle. It provides combinable scripts which solve different three-dimensional packing problems generating visually impressive results (see also Section 3.2.4). It is not a stand-alone program; it is based on the Python programming language which makes it compatible with a few other different modeling

tools like Cinema 4D, Maja, and Blender. Its alpha version is available at:

http://autofill.grahamj.com

AutoCell is an extension of *AutoFill* optimized for cytological modeling. Molecular shape-based models generated, for example, from PDB files are packed onto the surface of a predefined surface or distributed inside a predefined surrounding shape. This may be a primitive like a sphere or a cube, but also complex structures like a cell component model. Therefore, complexly filled cells can be generated. Because the packing algorithms do not take biophysical forces into account, these algorithms work relatively fast without the demand for biological correctness [John11].

4.1.1.5 LifeExplorer

The *LifeExplorer* is a tool developed by the Foundation Scientifique Fourmentin-Guilbert. The aim is reflected by the title of the first publication: “a visualization tool for molecular and synthetic biology” [FoLa07]. Therefore, the aims are quite similar to those of the CELLmicrocosmos CellEditor (Subchapter 5.1). In their first and second publication [LaFo08] discuss – each of them on one page – the vision of a Virtual Cell environment. The first prototype tried to model all the components involved in the transcription process of the lactose operon of Escherichia coli. In the second paper, four concrete aims for this software are mentioned:

1. interactive navigation inside the cell,
2. a graphical tool to build an abstract three-dimensional cell model,
3. an annotation tool to integrate and share information concerning the actual model and
4. a database to integrate biological data with the cell model.

This is a great vision; but as of yet (31.07.2012), the program is only in a very early state as shown in Figure 44. It is available on the website:

http://www.lifeexplorer.eu

It is possible to import 3d models in a native format. Here, a septosome is shown. The interaction with the model is limited to activating different components and combining them to selection groups in the menu on the left side of the screen. Only standard navigation techniques are provided.

In addition, there is a new molecular modeling tool now found at the web page, called GraphiteLifeExplorer. It is based on the Graphite toolkit which is installed as a stand-alone application [12f]. This is a powerful and fast visualization toolkit. It was adapted for the LifeExplorer project and extended by a few functions. For example, it is possible to combine, manipulate and animate different molecular structures based on the PDB format. A special workflow was integrated to manipulate geometrically DNA strands, shown in Figure 45. Obviously this project is still in its beginning phases, but this tool looks quite promising. It may feature a similar target group like the AutoFill/AutoCell project in the near future (preceding section).

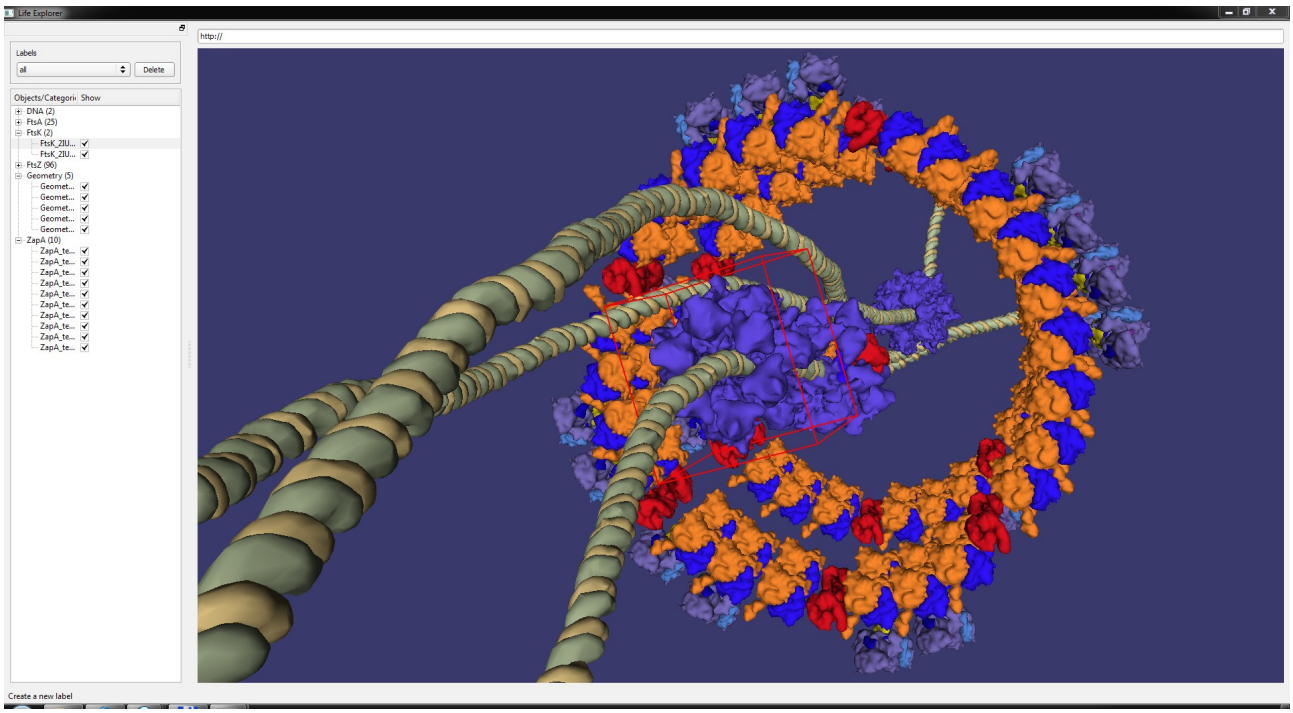


Figure 44: LifeExplorer
This software is a simple demo of a 3D model viewer, showing a septosome

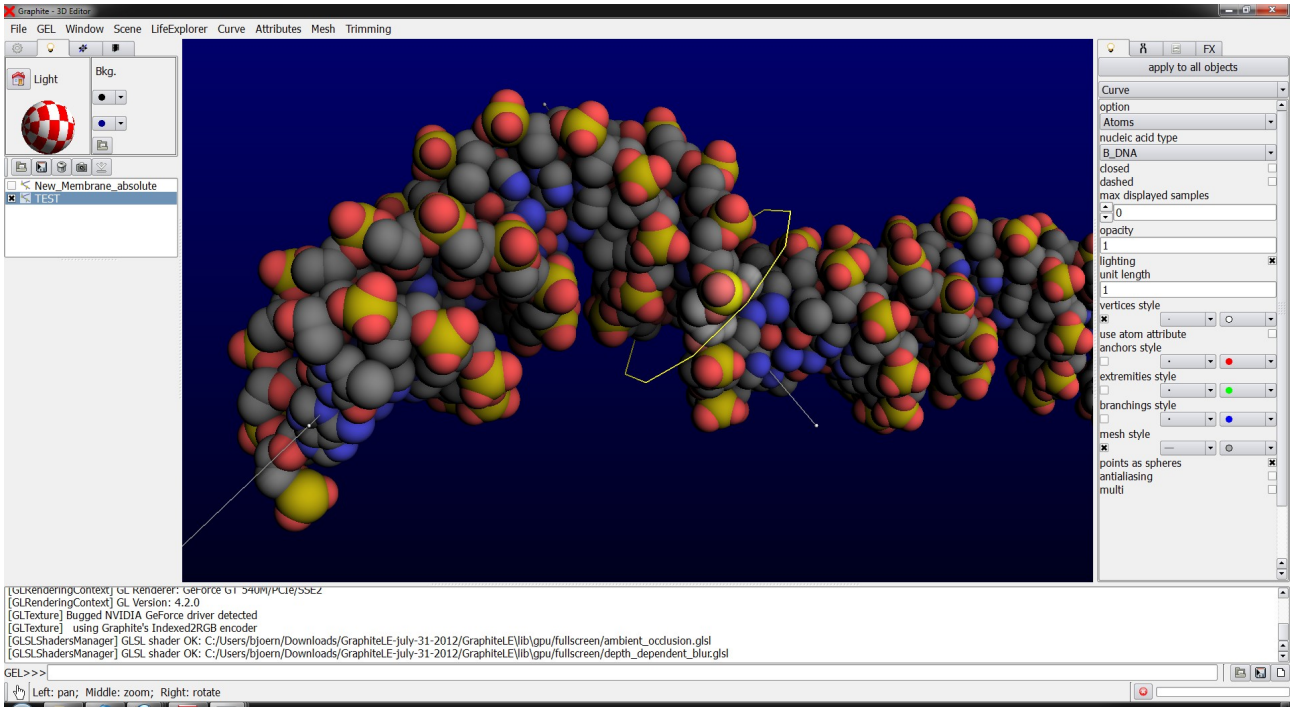


Figure 45: GraphiteLifeExplorer
This modeling software can be used to model DNA structures, based on the PDB format etc.

4.1.2 Mathematical Cell Modeling and Simulation

The preceding approaches are basically static approaches: they generate a time stamp of a biological cell. If the fourth dimension – time – should be taken into account, the cell has to be simulated computationally.

The establishment of cellular modeling environments enabling *in silico* experiments began already in the early nineties with tools like GEPASI or E-Cell [Mend93, THTS99]. Moreover,

the first computational approaches into this direction could be already found before 1970 [Garf68]. But in contrast to today, the powerful computational hardware capable for such simulations was only accessible to a small number of experts. The approaches discussed in this section are mainly focusing on the mesoscopic level. Therefore, molecular interactions will be included only indirectly.

Because this work is not focusing on simulation, only a subset of established environments will be discussed and the simulation properties will only be roughly described. In addition, only programs which support the three-dimensional simulation and visualization will be taken into account.[41]

4.1.2.1 E-Cell

The *E-Cell* project is developed at the Keio University at the Institute for Advanced Bioscience. The first version of E-Cell was already developed and published before 2000 [THTS99]. E-Cell is able to combine stochastic and deterministic algorithms in one simulation. And as of version 3, the parallelization of complex mathematical analyses is additionally provided [STKI05, TISS03].

E-Cell was developed from the start to construct whole cell models. As a proof of concept, a self-sustaining, primitive cell was already introduced with the first version. E-Cell is based on the so-called *Substance-Reactor Model (SRM)* consisting of three classes: Substance, Reactor and the System.

As of version 2, the simulation core of E-Cell is written in C++, while the GUI was developed in Java. Version 2 provided executable files for Linux and Windows. Figure 46 shows one of these releases, simulating a model of a mitochondrion [YuTo04]. The figure also highlights the differentiation between reactors like PDC and substances like 16Acyl-CoA.

E-Cell 3D is a visualization front-end for E-Cell 3 and it was only developed for MacOS X, taking advantage of its advanced and simplified graphical programming support. Since it was released in 2007, only a demo version of this software is available containing only one single model. The development of this 3D visualization front-end seems to have stopped. At any rate, it represents a very nice visualization approach. Because E-Cell 3D was not published until now (as of 31.07.2012), the information given in the following paragraph is based on the website of the E-Cell 3D project [07].

The three-dimensional model is visualized by a simple but colorful network consisting of nodes and edges (Figure 47). Instead of working with rigid objects the developers have chosen to use light sources, similar to small stars or suns, to represent nodes. They symbolize molecular assemblies. The nodes differ in size and brightness, colored from blue to red. These properties indicate the molecular concentrations of the nodes. Between the nodes, Bezier curves – the edges – are drawn. Particles are moving along these edges. Their speed indicates the actual flux. If the flux exceeds a predefined limit the position of the involved node changes. A simple GUI is integrated into the 3D view. The *Selector* with all actual components is represented by a list with selectable items on the left side of the

41 For an overview on cellular modeling environments please refer to [LoSc01].

viewport. The brightness of the different identifiers indicates the molecular amount. The so-called *Tracer*, which is also part of regular E-Cell applications (Figure 46), shows the time progression of each component. The Tracers of all components are shown at once. They surround the centered cell in a circle which can be rotated in order to find the component of interest.

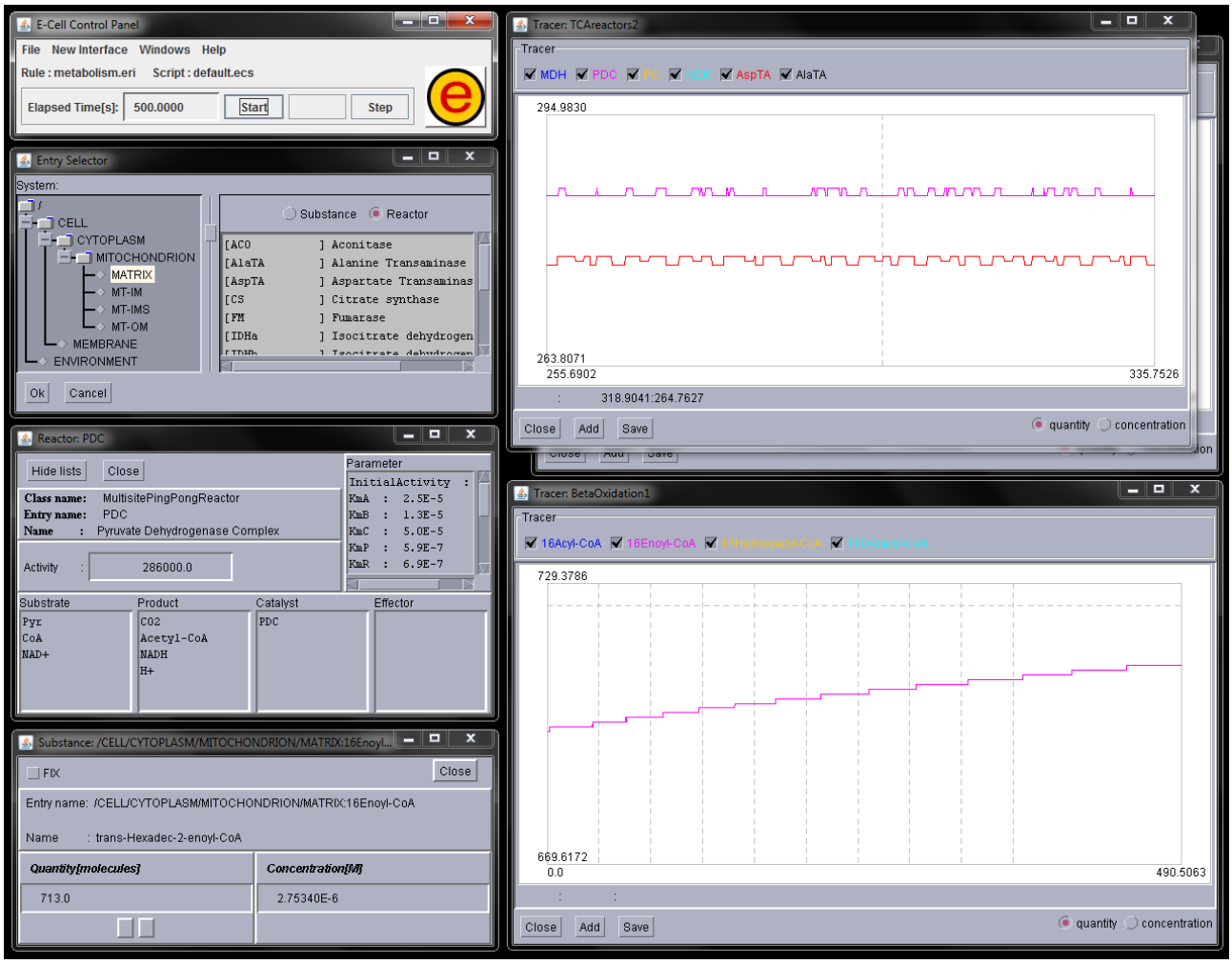


Figure 46: E-Cell 2: The simulation environment with the Java front-end on Windows 7 Showing the mitochondrion model. On the left side (top-down): the simulation control panel, the Selector, the properties for the Reactor PDC and the Substance 16Enoyl-CoA; on the right side, top: the Tracer showing the fluxes of the Reactors PDC and AspTA, and on the bottom: the flux of 16Enoyl-CoA.

4.1.2.2 The Virtual Cell (VCell)

The *Virtual Cell simulation environment (VCell)* was first introduced in 2001 by the University of Connecticut Health Center and the underlying category was entitled as *Cell biological modeling* [LoSc01]. For over a decade a remarkable community has been using and supporting this software. It is intended to bridge the gap between (experimental) cell biologists and mathematical biologists and bioengineers. For this purpose, it provides a user-friendly browser GUI written in JAVA encouraging biologists who have only a little additional mathematical and/or physical training to work with this tool. It is able to automatically convert a biological model into a mathematical system of ordinary and/or partial differential equations, a base for the subsequent simulation.

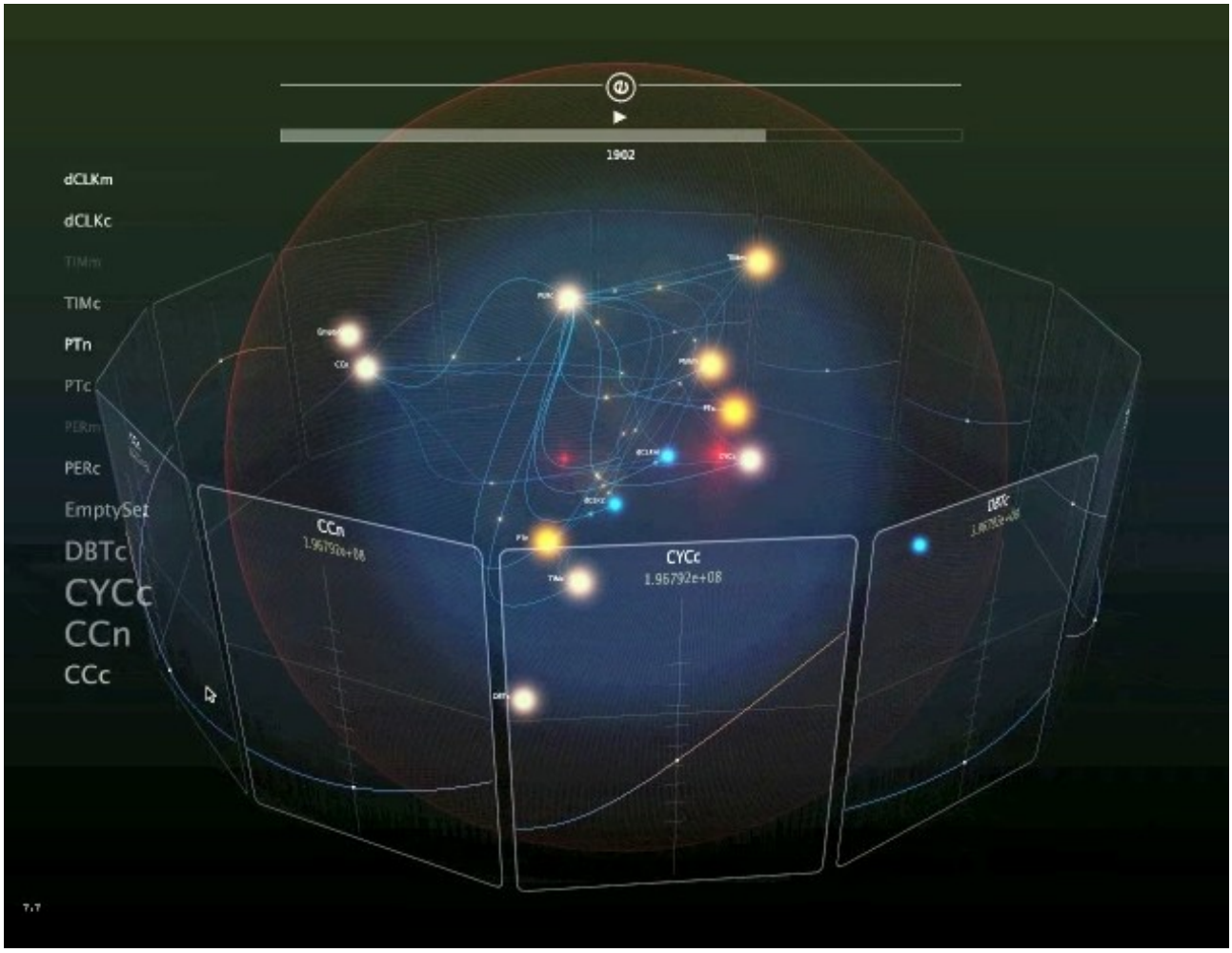


Figure 47: E-Cell 3D
The cell is surrounded by the Tracers for all components shown in the center. Left the Selector is shown (Courtesy of/Copyright © 2007 by Kazuharu Arakawa, Nozomu Yachie and Masaru Tomita. Reprinted with permission from [ArYT08])

It enables the modeling of electrophysiological as well as biochemical processes. These processes can also be compartmentalized. These structures may feature idealized dimensions in one, two or three dimensions, but they may also represent complex cellular structures.

Although the original program was written in C++, it can be easily accessed by using a declarative language called *VCMDL (Virtual Cell Mathematics Description Language)*. For its users, VCell provides a direct cluster access as well as a database where all results may be saved and publicly made available. The front-end of the actual version is a Java Web Start application which can be directly installed from the browser.

The Figures 48, 49 and 50 show a deterministic simulation of a Fluorescence Redistribution After Photobleaching (FRAP) of nuclear substances.[42] FRAP is a special fluorescent optical technique to measure the time-dependent dynamics of a molecule together with the chemical changes of molecular species. The shown simulation was developed by the VCell team [11i].

42 Related methods are described in Section 3.1.1.1.

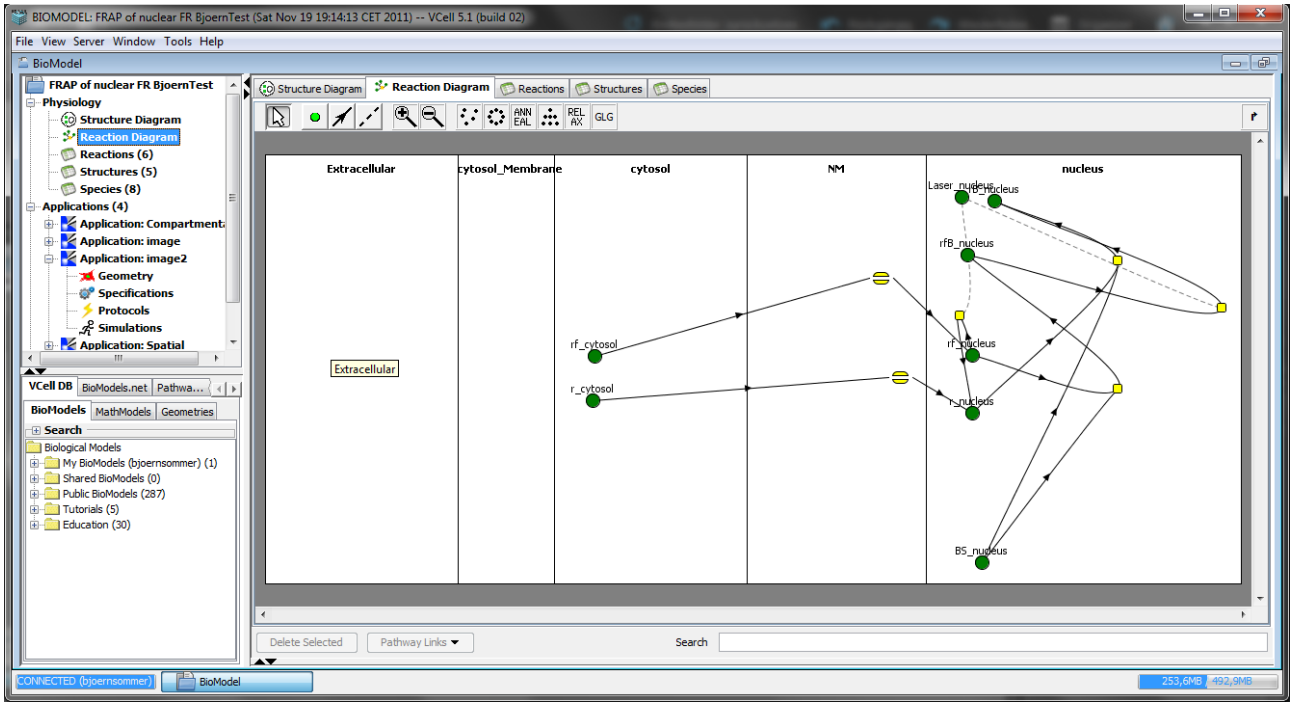


Figure 48: VCell: The FRAP of nuclear FR: the network

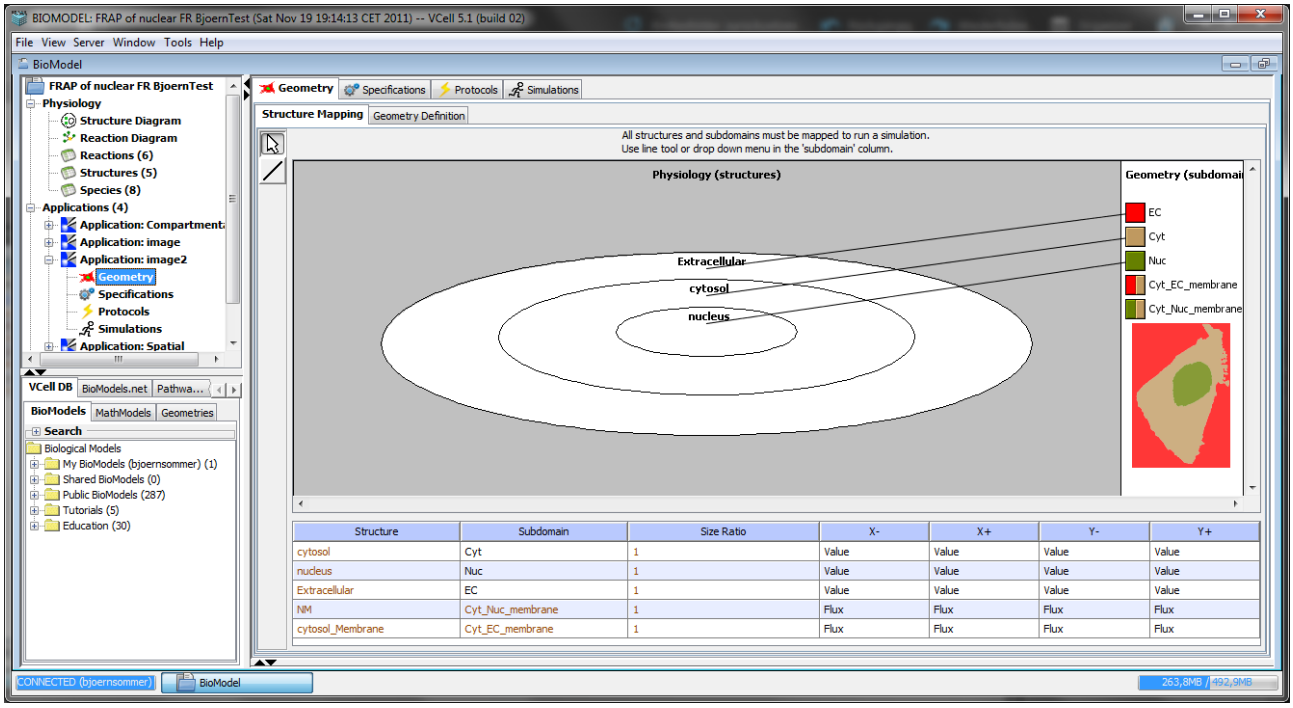


Figure 49: VCell: The FRAP of nuclear FR: the two-dimensional distribution

For enabling the direct comparison of wet lab experiments with the data generated by VCell, the same spatial as well as temporal records are saved. They can be analyzed by statistical as well as image analysis methods. The export of images, movies and spreadsheets is possible. Interchangeability with other tools is ensured by import of SBML and CellML format [LoSc01, MSSL02].

Figure 48 shows the interaction network. The species and reactions are the nodes of the network which are interconnected by edges. Every node is assigned to a localization which may be a membrane or a compartment. The localizations are also shown in Figure 49. The

abstract ellipsoid structures which are normally used for the visualization are connected to a two-dimensional microscopic picture. The microscopy-derived shapes are shown on the right side of Figure 49. The simulation results are shown in Figure 50. The colors indicate the reaction rate inside the nucleus for the chosen species after two seconds.

The newest version is also able to import surface-based 3D geometry which may be derived by microscopy (as of 31.07.2012, Figure 51). In addition, it is possible to correlate quantitative live cell microscopy images with the simulation.

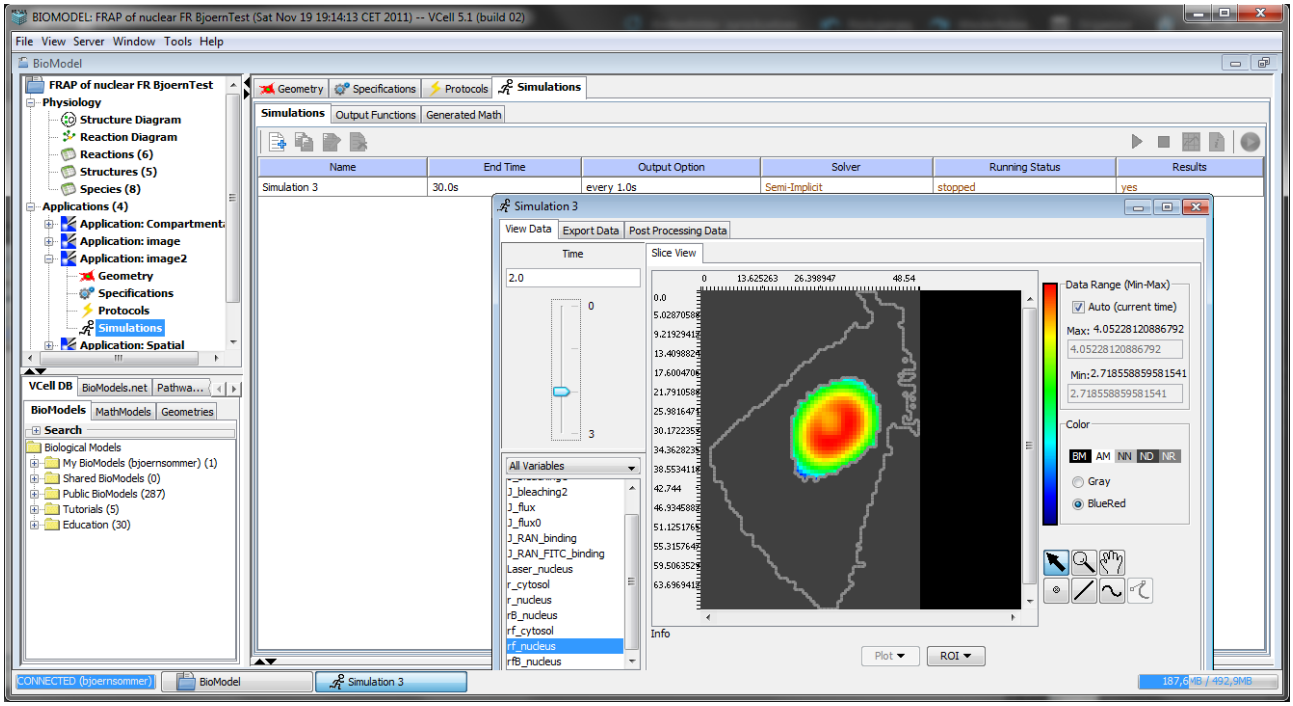


Figure 50: VCell: The FRAP of nuclear FR: the simulation results of the nucleus

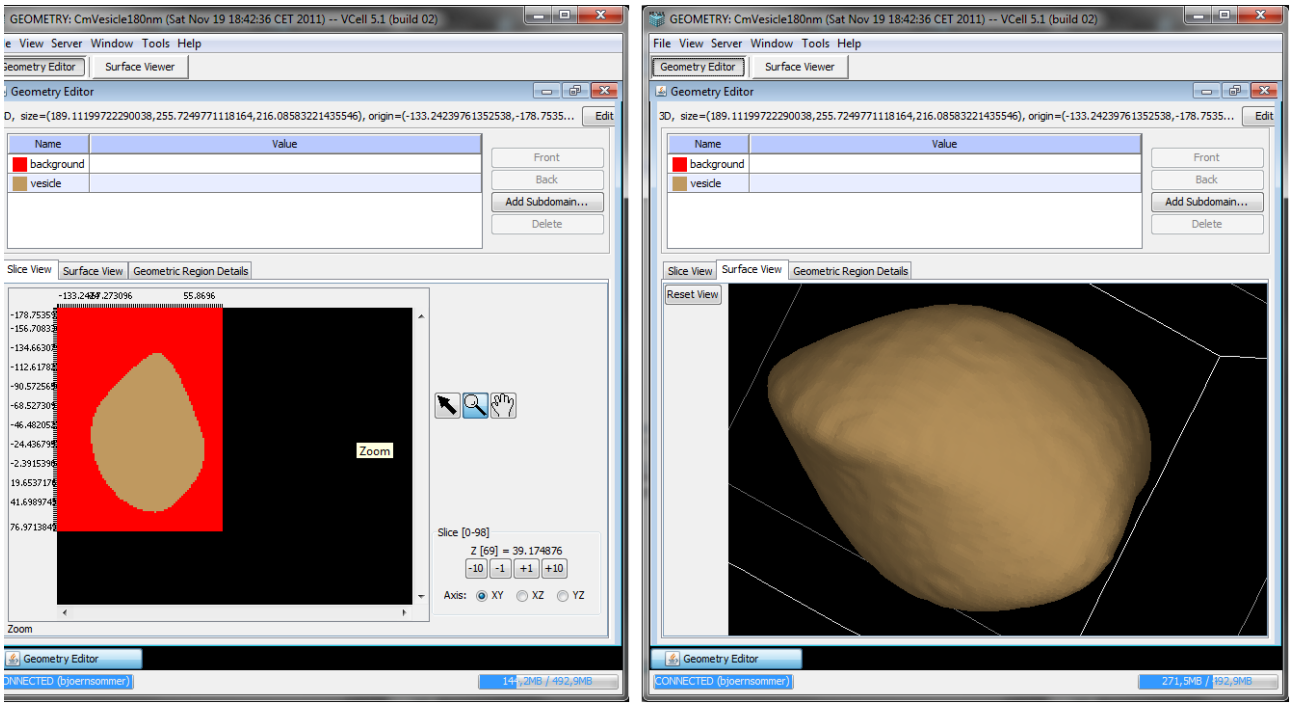


Figure 51: VCell: Import of geometries.
The new feature since version 5.1 Beta. The imported vesicle structure is a model of the CELLmicrocosmos project

4.1.2.3 CompuCell3D

CompuCell3D was introduced in 2005 [CHCG05, MeGl05] and was an extension of the CompuCell system published the preceding year [ICHC04]. It is maintained by the Indiana University at Bloomington and the affiliated Biocomplexity Institute. The main purpose is the simulation of morphogenesis: the time evolution of differentiating, growing cells. For this purpose, stochastic rules as well as differential equations were integrated.

The incorporation of experimental data is supported by a so-called *Cell-Type-Map*. This map contains information about the development of connective tissue to model the cell growth. The stochastic methods are based on the *Cellular Potts Model (CPM)*, an originally two-dimensional, grid-based model to design cell movements and interactions [GrGl92]. Although cellular substructures like the cytoskeleton are not explicitly taken into account, it is able to reproduce cell membrane fluctuations measured by cell dynamic experiments. This aspect becomes possible by the integration of relative cell-adhesion parameters into the CPM, derived from surface-tension experiments with different cells.

Therefore, by using CompuCell3D, measurements derived from a single cell may be applied to a large number of different cells to analyze and verify the impact of the experimentally derived values. This method should be used to identify inaccuracies or even faults which have to be omitted by subsequent experiments. If the results of experiment and simulation finally match, gene networks can be created based on these results.

Developed in C++ and the Qt package, the computation is based on a kinetic Monte Carlo process. Because it was complicated to parallelize this method, it took several years until a parallel version (3.5.0) of CompuCell3D was available [11j, CHCG05].

Figure 52 shows the CellSort demo found at the CompuCell3D website [11j]. It is possible to open multiple views to compare the simulation of three-dimensional environment from different sides in the 3D view or on different slices in the 2D view. In addition, CompuCell3D is packaged with CellDraw. This tool can be used to generate cells by two-dimensional drawing as well as by importing single images or image sequences as TIFF (Tagged Image File Format), DICOM (Digital Imaging and Communications in Medicine) or PIF (Pots Initialization Files) files, a native CompuCell3D format. Therefore, the correlation with microscopic images is also possible.

4.1.2.4 ENVIRONMENT

Another simulation approach should be shortly mentioned, representational for those computational approaches which do not support a graphical representation of the cell: *ENVIRONMENT*. This software integrates the mathematical/stochastic simulation of vesicles as well as complete protocells based on Monte Carlo algorithms. Moreover, it considers membrane heterogeneity by taking different lipid types into account under aqueous conditions (Section 2.2.3). For example, the development of a number of homogeneous POPC and oleic acid vesicles was simulated. Over time, the oleic acid vesicles were completely gripped by the POPC vesicles and a heterogeneous mixture was created. The results are visualized as statistical representations over time.

The stated publication also compares the computational results with already published experimental work supporting the observed results of the simulation. The major intention of ENVIRONMENT is to support and predict experimental research focusing at protocells or artificial cell systems [MaRu10].

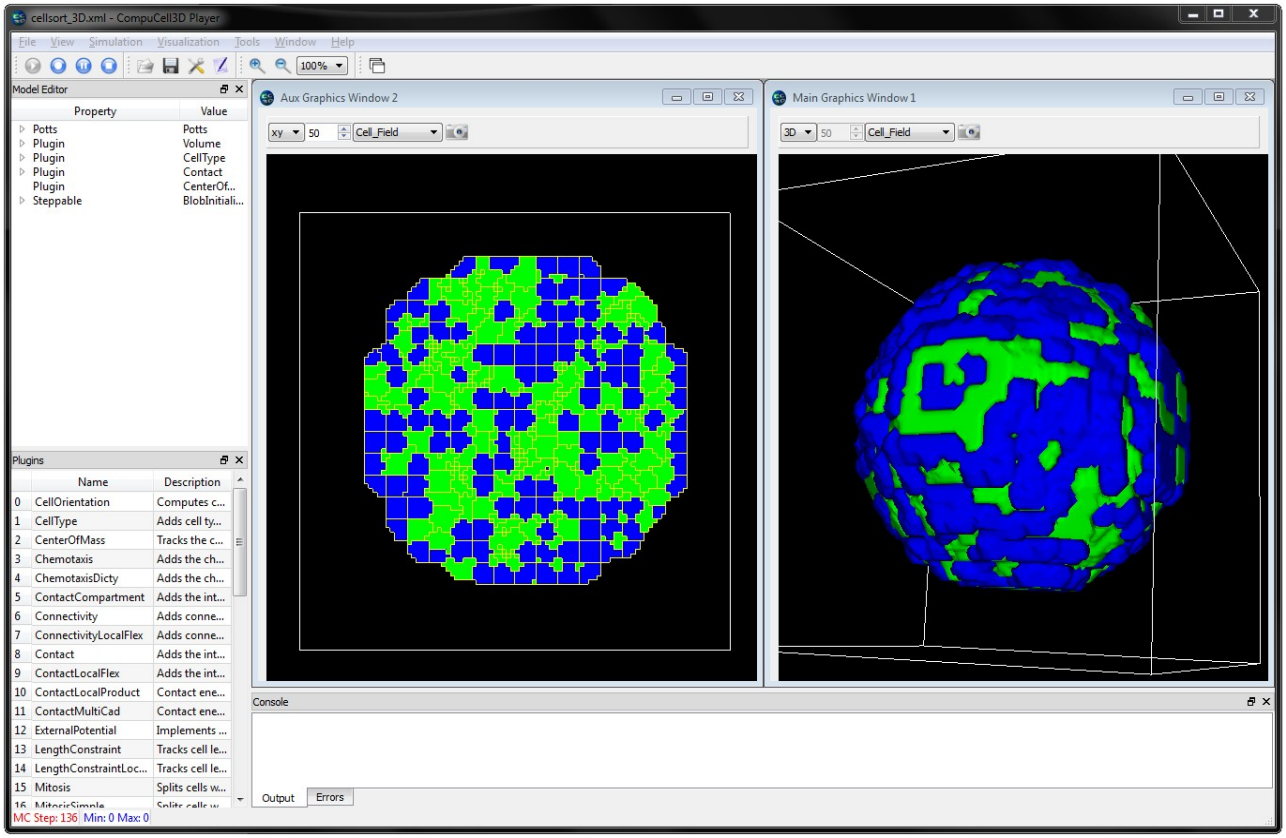


Figure 52: CompuCell3D: The "CellSort" demo
Showing the 2D and 3D view during the running simulation

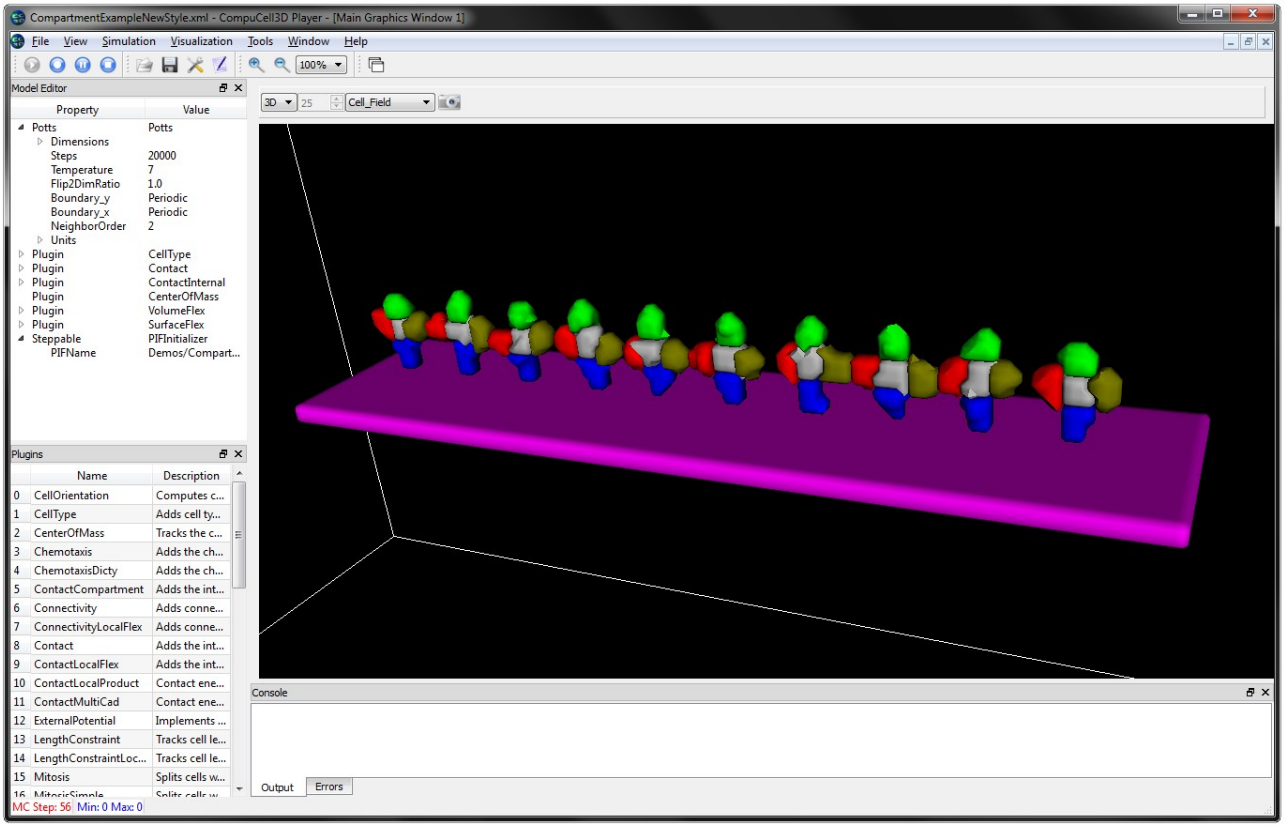


Figure 53: CompuCell3D: The "CompartmentExample" demo
Showing the 3D view of a dendrite-like structure

4.1.3 Summary

Category	Feature	AutoPack/AutoCell[43]	DisplayCell (Amira)	LifeExplorer Project	Meta!Blast	CompuCell3D	E-Cell (3D)	ENVIRONEMENT[43]	VCell
Availability	Plugin	X	X						
	Standalone			X	X	X	X	X	
	Web Start								X
	Source Code					X	X		X
	Licenses					Open Source	GPL		MIT
Dimensions	2D					X	X		X
	3D	X		X	X	X	3D		X
	Time (incl. 4D)				(X)	X	X		X
Granularity	Atomistic								
	Molecular	X							
	Intra-Compartmental	X			X	X		X	X
	Subcellular	X			X	X			X
	Organisms				X				
Interactivity	3D Navigation	X		X					
	Dynamic Tables								X
	Links to data sources								X
	focus and context								X
	Sorting Tables								X
Modeling	abstraction level, cell components	1/2	2	1	2	3			(1/2/) 3[44]
	abstract models	X				X			X
	mathematical					X	X	X	X
	microscopic models	X				X			X

43 Because these software tools are not completely available as of yet (31.07.2012), the information in this table are given with reservations and are perhaps partly fragmentary.

44 ABS_LEVEL_1/2 are only available via data import in VCell since version 5.1 Beta.

Category	Feature	AutoPack/AutoCell	DisplayCell (Amira)	LifeExplorer Project	Meta!Blast	CompuCell3D	E-Cell (3D)	ENVIRONEMENT[43]	VCell
	node distribution algorithms								2D
Programming	Scripting	X				X	X		X
	Scripting at Runtime								X
Visualization	2D Mapping								
	2D Visualization					X	X		X
	3D Visualization	X		X	X	X	X		X
	3D Visualization: Live-Animation			X		X			
	3D Visualization: Geometric	X		X	X		X		X
	3D Visualization: Volumetric								
	Graph Visualizations						X		X
	Internal Graph Mapping								
	Marking of POI			X					X
	Molecular Visualization	X							
	Primary Structure View			X					
	Secondary Structure View								
	Raytracing	X							
	Runtime Graphs								X
	Statistics					X	X		X
	Stereo Support								

Table 7: A comparison of all cell modeling tools discussed in this work
The complete table is found in Appendix 9.19.1

According to the communities and the usage frequency of the websites, the VCell and CompuCell3D project reach an active community, while the E-Cell project seems to be presently stagnating. But the development of E-Cell is ongoing and the fourth version is already in discussion. Obviously, all of these projects are vital. The ENVIRONMENT is quite new, therefore a community first has to be established. But it will be quite a challenging task to combine these simulation approaches with a Virtual Cell visualization in 3D. This work will not introduce simulation approaches, but will introduce a framework which could be correlated in the future with a simulation environment.

The target audience of Meta!Blast is of course completely different. It does not provide a tool for scientists, but a tool which may acquire future scientists from schools. It is also not a simulation environment. AutoFill/AutoCell is an approach which looks very promising for the future of geometrical cell modeling. It will be especially interesting for those people who plan to make cellular visualizations of filled cells by using tools like Cinema 4D. The LifeExplorer project aims at a similar direction as the work discussed here, but it is more focused on the vision of a live exploration of the cell: the DNA modeling tool provides a good first impression.

What all the projects introduced in this section have in common is that they are operating on the mesoscopic level. They all contain abstractions from the complex molecular structures and chemical processes.

In conclusion it can be stated that none of these tools provides a way to generate a modular cell environment in a quick and easy way. There is no possibility to generate a customized cell environment with a few clicks, providing a large variety of different cell components. In addition, the exploration of cells also needs special navigation techniques. Only Meta!Blast provides a navigation which is easy to use. The LifeExplorer provides only basic movements which will be a problem if different granularities have to be taken into account.

Table 7 shows an overview of all projects discussed in this section. Starting with Subchapter 5.1, the aspects will be discussed needed for a cell modeling environment. It will be shown that this work follows quite a different approach from those works discussed in the preceding paragraph.

4.2 [MOL] Membrane Modeling Tools

The tools on the mesoscopic level have been introduced: it has been represented by different tools which usually understand the cell as a cluster of different membrane-surrounded shapes representing cell components. Now, approaching the molecular level, the question should be asked, how can these membranes be resolved into their particular ingredients: the molecules. How can these membranes be modeled on the molecular level?

Again, a selection of projects is presented with no guarantee of completeness. But the most important tools in this area should be covered. First, tools focusing mainly at molecular visualization should be summed up (Section 4.2.1), followed by a detailed discussion of cell modeling tools (Section 4.2.2).

4.2.1 Membrane Visualization

Prior to analyzing membrane modeling tools, a short overview of molecular visualization should be presented. The following tools are known to be often used in conjunction with the PDB format (Section 3.2.2.1.1). Today, there are a large number of PDB viewers on the market. Most of them are free of charge, a small number of them are only commercially available. Although the CmME (Subchapter 6.2) is able to visualize membranes, the main task lies in the modeling process of the membrane. Therefore, here only a list of PDB viewers is given, which have been tested in the past with membranes exported from CmME. Although the PDB format has a clear definition, there are a number of exceptions which may cause viewers to be incompatible with certain PDB exports. Therefore it was a challenge in the design of CmME to export PDB files which are compatible to different PDB viewers.

CmME was tested with the following viewers (where available, the publication is stated):

- Chemsite Pro® (ChemSW®) [10a],
- Jmol (free, LGPL) [00a],
- Chime (MDL) ,
- Chimera (UCSF) [PGHC04],
- Amira/AVIZO (Mercury) [Konr02],
- ICM Browser (Molsoft),
- MOE (Chemical Computing Group) [10b],
- PyMOL (Schrödinger),
- QuteMol (free, GPL) [TaCM06],
- Swiss PDB Viewer/DeepView (Swiss Institute of Bioinformatics),
- VMD (University of Illinois) [HuDS96].

4.2.2 Membrane Modeling

Five tools will be introduced providing capabilities to generate membranes without the need for computing complex molecular interactions (Section 3.2.3). In addition, these tools do not assume physicochemical expert knowledge and the regular computer user is able to learn the work with them in an appropriate time frame.

4.2.2.1 A Grid-based Method using MOE

MOE, the Molecular Operating Environment is a commercial, well-established and powerful molecular modeling tool used, for example, in drug discovery [10b]. In contrast to the other tools discussed in the following sections, MOE does not provide an explicit plugin to create membranes. Therefore it is used here to show, how the bilayer modeling approaches discussed in Section 3.2.5 would be implemented in practice.

The script was re-implemented by the author based on the method of Jens Krüger [KrFi08].

MOE provides the scripting vector language (SVL) which can be used to program scripts interacting with the molecules. Figure 54 shows the SVL script "MOE Membrane Grid". For creating this script, a programmer who has never used MOE before, needs approximately one day to read the according parts of the documentation, learn SVL and create the script. This script needs a pre-aligned PDB file as input. In this case, a DPPC lipid was used. This lipid must be oriented upwards and must be physically centered: the center of the molecule must be the center of its coordinate system and the Y axis must be oriented vertically. This PDB model is imported into MOE. Now the MOE Membrane Grid script is started directly in MOE by using its SVL Text Editor. Next, the user interface shown in Figure 54, right bottom, appears. The user can define the distance between each lipid and the number of lipids per row for each X and Z axis. In addition, a monolayer or bilayer can be created. After confirming the settings, the membrane is generated by using the loaded PDB lipid as a template. Figure 54 shows the source code.

The first loop iterates through the membrane sides. The second and third loops process the X and Z rows of the membrane. The lipid template is copied, randomly (between 0 and 359°) rotated around the Y axis and shifted according to the X/Z spacing and the loop count.

The result of a DPPC lipid using ten lipids per row with a distance of 10 Å can be seen in Figure 55. In a next step, a protein could be manually assembled to the membrane by using the replacement method (Section 3.2.5.1).

This membrane grid method is sufficient to create a simple homogeneous bilayer for special application cases. The problem which arise in this case are manifold.

- A special software is needed, which sometimes, like in case of MOE, might be commercial.
- The software needs a scripting interface to interact with imported molecules.
- Not every life-science-related researcher is able to create scripts. Chemists and physicists, who are involved in strongly math-related disciplines, are normally more familiar with programming tasks than Biologists.
- The placement of this grid-based method is very simple and far from being natural. Collisions of the atoms are not taken into account. Therefore, many different minimization and equilibration steps will be needed before the simulation can start (see also Section 3.2.5.3 and the MOE application in [SDGS11]).

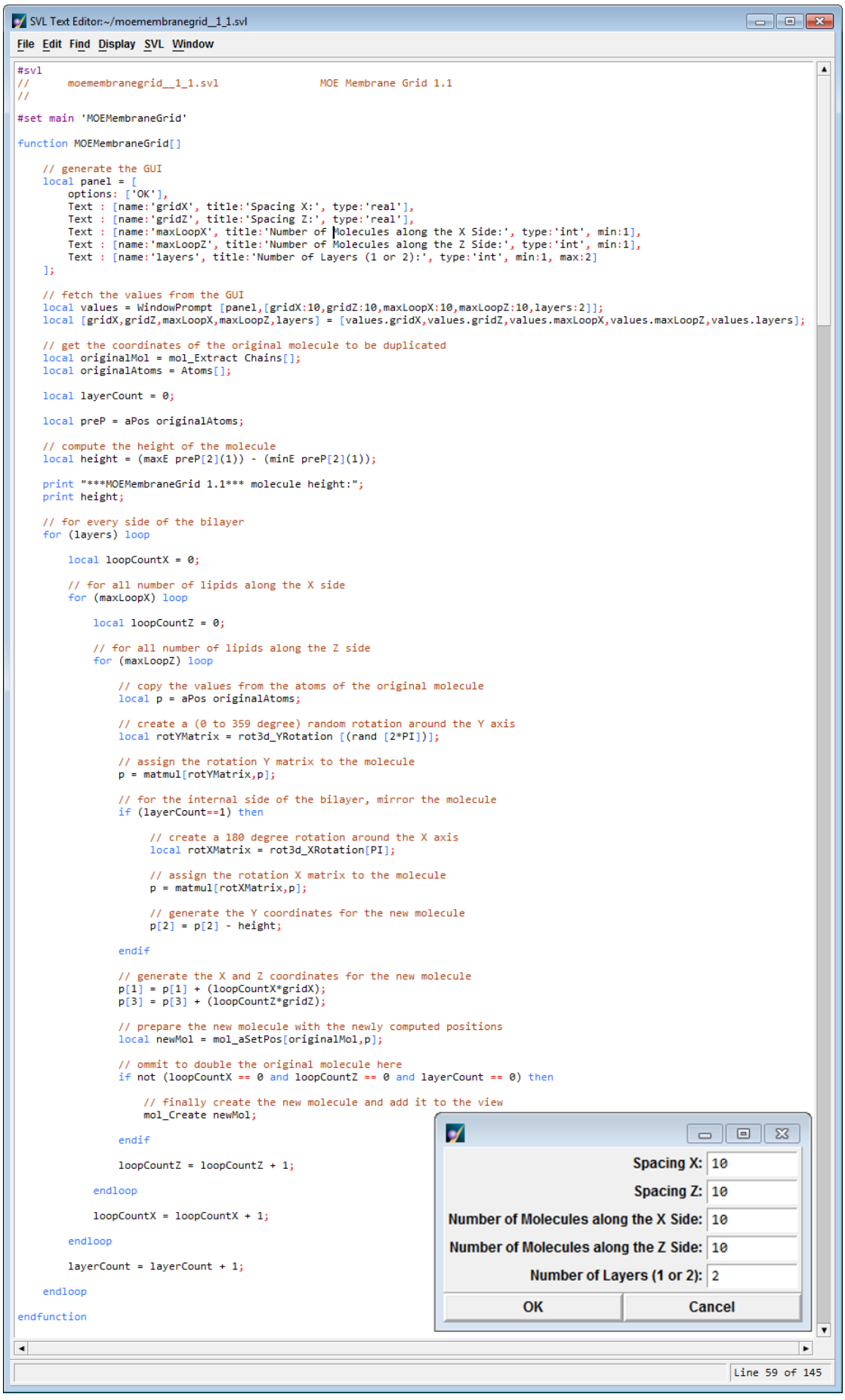


Figure 54: The MOE Membrane Grid script
In the right bottom corner the GUI defined by the panel (see Figure 55 for the script's results)

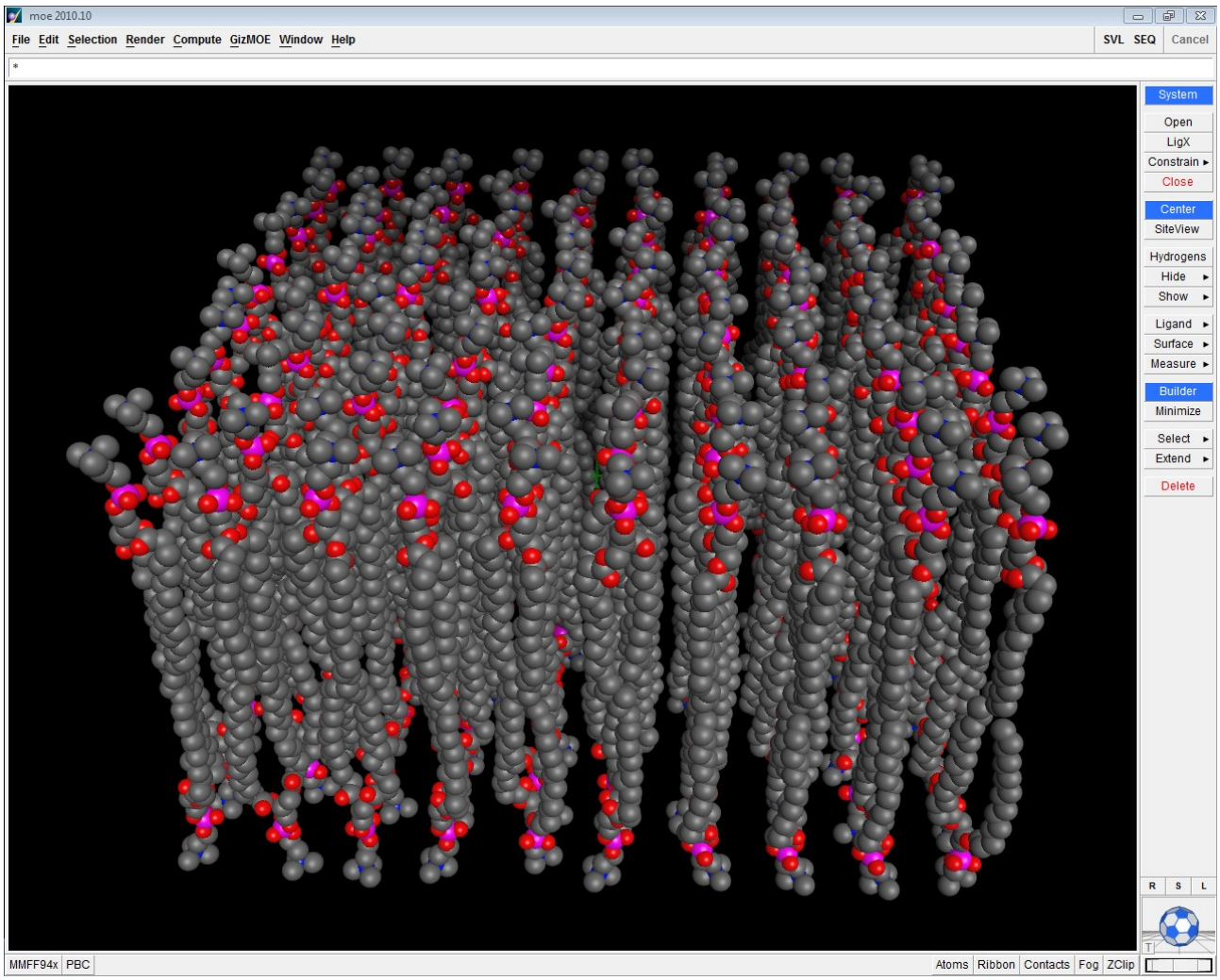


Figure 55: The result in MOE using the properties of Figure 54

- Heterogeneous membranes cannot be created with the script introduced here. It would be certainly possible to write a complex script also capable of handling this problem, but only with a lot of effort.
- The results of this script are not reproducible, because simple random numbers are used. Generally MOE also provides seed-based random number generators for SVL scripts, so it would be possible.

It was shown that this is not a quick and user-friendly method to create a membrane model. During the past few years, a small number of plugins, programs and web tools were created addressing this problem. They are introduced in the following sections.

4.2.2.2 ChemSW® Chemsite Pro®

The commercial software *ChemSite Pro*® from the company ChemSW® is a package to model and simulate membranes [10a]. Compared to all other stand-alone programs showing parallels with MembraneEditor, it provides the most intuitive approach to model membranes for regular users. The program version discussed here is 7. A full functional demo version was used for this review. In addition, the program can also be achieved packaged with the MolSuite which comes with improved MD packages as well as a database containing different molecular structures.

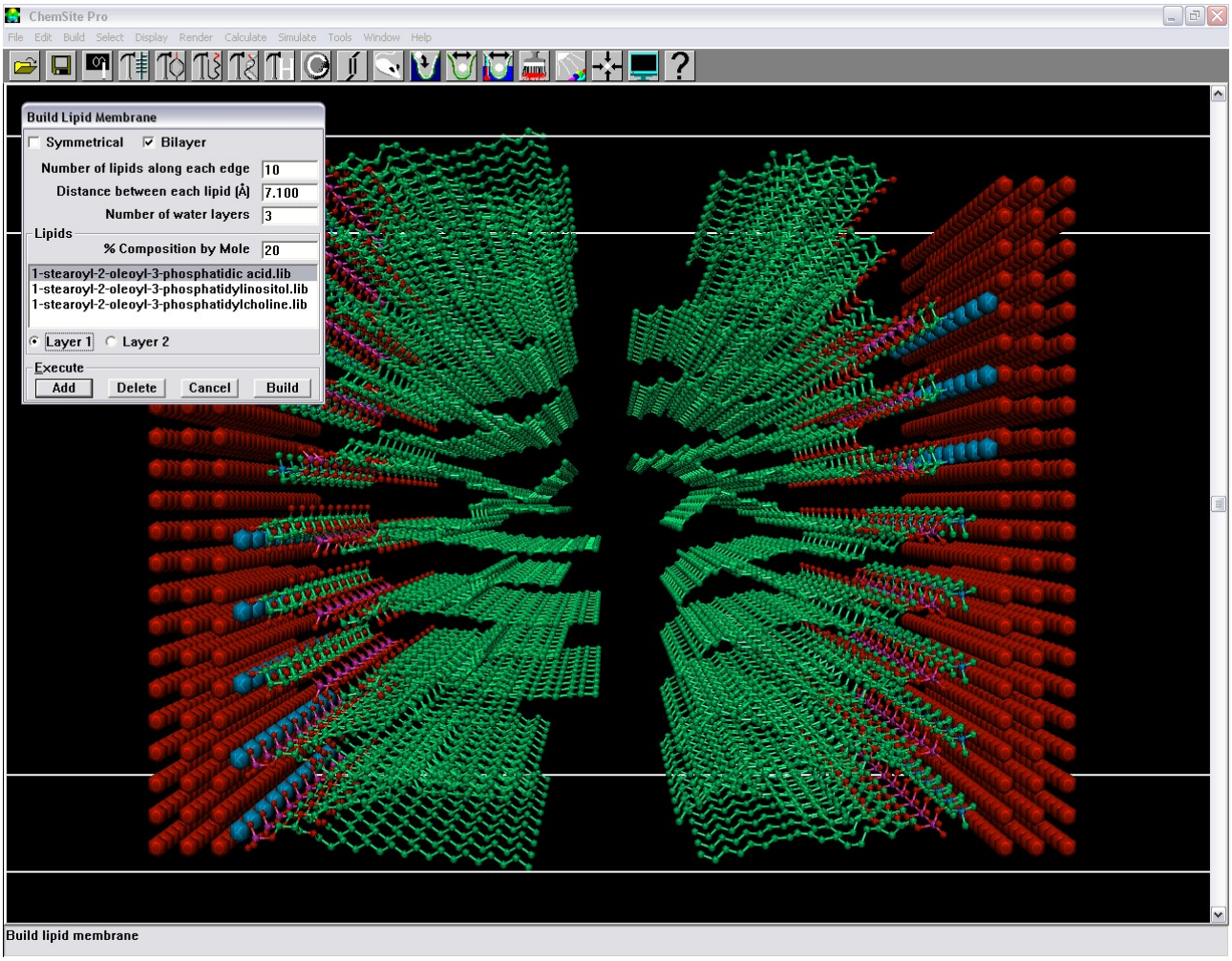


Figure 56: A lipid bilayer created with ChemSite Pro®

After starting the module *Build Lipid Membrane* the user has different options. First he can choose, if a symmetric or asymmetric monolayer or bilayer is modeled. The width of the strictly quadratic shaped membrane is defined by the number of lipids along the edge. Therefore it is also a grid-based approach without the option to take different sizes of lipids into account. Then, the distance between the molecules can be defined in Ångstrom. The width of each membrane side can be computed by multiplying the distance with the number of lipids. The number of water layers at each side of the membrane can be defined (Section 3.2.3). Finally, there is a section where a virtually unlimited number of lipid types can be correlated with the membrane model. Each molecule type is correlated with a percent value for the mole composition. If the membrane should not be symmetrical and a bilayer was chosen, the values for each lipid type have to be defined for both sides. Therefore, the asymmetry of the bilayer can be correctly modeled here (Section 2.2.3). Finally, the “Build” button has to be pressed and the membrane is generated. Automatically counterions are added to the membrane if the contained lipids are charged in order to maintain equilibrium state.

The resulting membrane is not a mixed bilayer. The grid-like structure is in a highly ordered state. The different lipid types are placed iteratively. This means, the first rows are filled with lipid type1 until the percentages are reached, then the following rows are filled with lipid type 2 and so forth. Therefore, the mixed nature of a bilayer cannot be taken into account.

The addition of water layers is often needed for MD simulations, but these layers have to be adjusted to the heads of lipids. As can be seen in Figure 56, parallel rows of water layers are placed beneath the heads of the highest lipid heads. This creates a large distance between small lipids like cholesterol, and the water layer. Regularly, these membrane models will be very problematic for a sophisticated simulation, because the layer might collapse.

The lipid structures used are in a native format called *lib*. This native ChemSite library format contains the coordinate information and some further information regarding charges and so forth. It is not possible to use PDB files to generate these membranes. But it is simple to translate PDB files to lib files by using ChemSite Pro and to use them afterwards for the membrane building process. The library coming with ChemSite Pro contains six different types: four phospholipids, one phosphatic acid and one cholesterol.

An important tool included in ChemSite Pro is the *Lipid Builder* which can be used to construct lipids. The constructed lipids can be saved as *lib* files and then be used for the Membrane Builder. First, a backbone can be chosen from glycerol, C16 - C20 Sphingosines or C16 - C20 Dihydrosphingosines. The two to three rest groups (the number depends on the backbone type chosen) surrounding the backbone can also be defined. For R1 and R2 (Section 2.2.1.3) there are the acetic, capric, lauric, myristic, palmitic, stearic, arachidic, behenic, lingnoceric, cerotic, palmitoleic, oleic, linoleic, alpha-linolenic, gamma-linolenic, arachidonic, nervonic acid or EPA. For R3, water, ethanolamine, choline, serine, myo-inositol or glycerol may be chosen. For the glycerol backbone it can be chosen if the resulting lipid should be a phospholipid. In the latter case the type of linkage for the R1 group is also variable: ester, alkyl ether or alkenyl ether. Therefore, a large number of different lipid types can be constructed, although of course not every possible lipid can be created.

A significant disadvantage of this program is low performance. Exploring a membrane with 100 lipids with this tool results in juddering unless a high end computer is used. In addition, the layout of the GUI is outdated and there seems to be no community supporting or using this tool.

The simple grid-based approach to generate membranes causes an extreme energy state after the membrane has been generated. The test membrane has an energy of approx. 50,000,000 kcal/mol. Therefore a very long minimization run is needed before the MD simulation can be started. This is particularly a problem because it is not possible to use a cluster system for the minimization process, it has to be computed on a desktop system. Therefore it will be a very long time until the membrane is ready for the simulation process.

However, if a simple membrane containing e.g. only 25 lipids (the smallest possible size in ChemSite Pro) of the same lipid type, the minimization finishes quite quickly and the MD simulation might start. But the MD simulation performed by this program is not of a high quality. It will not be possible to use results of this program for reasonable publications. Analyzing the results of a MD simulation shows, that the water molecules penetrate the bilayer and are found everywhere in the system, even between the two lipid layers in the hydrophobic area. Obviously the system is corrupted in this case.

4.2.2.3 CHARMM-GUI Membrane Builder

Chemistry at HARvard Macromolecular Mechanics, abbreviated with *CHARMM*, is a highly established academic program package providing different tools and force fields for the minimization and simulation of different molecular structures [VZHP93]. The difference with free tools like GROMACS is the licensing issue. Academic as well as commercial licenses are available. The current version number is CHARMM 36 [BBMN09]. In contrast to the commercial simulation program, *CHARMM-GUI* is available free of charge.

The CHARMM-GUI initiative is quite new. The idea is to provide users of CHARMM a set of tools which should simplify the process of generating starting structures for MD simulations [JKII08]. The usage is very simple because the CHARMM-GUI can be directly accessed via the website [11k]:

http://www.charmm-gui.org

One tool collection found at this website is called the “Input Generator”. There, the *Membrane Builder* tool is located. Homogeneous as well as heterogeneous bilayers can be created. Monolayers are not possible. For homogeneous bilayers six lipid types are provided: DMPC, DPPC, DOPC, POPC, DLPE and POPE. For heterogeneous bilayers, 32 lipid types can be used: Cholesterol, DLPC, DLPE, DLPS, DLPG, DLPA, DMPC, DMPE, DMPS, DMPG, DMPA, DPPC, DPPE, DPPS, DPPG, DPPA, DOPC, DOPE, DOPS, DOPG, DOPA, POPC, POPE, POPS, POPG, POPA, SOPC, SDPC and SAPC. Moreover, each lipid type in this library is represented by 2000 different conformations [JLKI09].

There are two different shapes for the bilayers possible: The well-known rectangular shape as well as a hexagonal shape.

The generation process is subdivided into five different steps for membrane only systems (without proteins).

In the preliminary step (Step 0), the conformation of the bilayer is defined. The user has to choose the appropriate shape, the thickness of the water layer, the initial guess for the rectangular size and if ratios and numbers should be used as a reference point. All available lipids are listed and the user can choose the absolute or the percentaged distribution values for each bilayer side. An additional value shown for each lipid type is the surface area used as an initial guess. This value can also be modified by the user.

In Step 1 (the website starts here to count the steps) an overview of the first generated files is shown and the user can choose if the insertion method (only available in homogeneous bilayers) or the replacement method should be used for the assembling process (Section 3.2.5). If charged molecules are found, the program is able to inhibit counterions into the membrane.

Step 2 now combines the previously generated membrane components to build an ion and water box. This is also the first opportunity to download or to look at a preview of the PDB structure using a Java applet called MarvinSpace (Version 5.4) from ChemAxon Ltd.

In Step 3, options for the equilibration process are provided. The temperature, the type of ensemble[45] and the surface tension may be defined.

45 The ensemble type defines, in this case, which physicochemical aspects remain constant throughout the simulation, this could be for example: the temperature, the pressure or the charge.

Step 4 finally provides all downloadable files needed for the further simulation with CHARMM. The files are also compatible for NAMD simulations which can be used with VMD discussed in the following section.

In addition, the two methods discussed above were implemented to place proteins into the bilayer (Section 3.2.5):

The first method is called the Replacement method. The protein/membrane complex is generated by distributing lipid-like pseudo atoms around the protein. These pseudo molecules are substituted by lipids. For each replacement step the program chooses one lipid model randomly from a 2.5 ns trajectory containing 2,000 different conformations for each lipid type. The process of the membrane generation takes 5 to 50 minutes and it is possible to generate rectangular as well as hexagonal shapes. Of course, this does not depend on the local machine, but on the (web)server used by CHARMM-GUI in the background.

The second method applied is the Insertion method. For that the CHARMM-GUI provides a number of pre-equilibrated lipid bilayers. These membranes have a hole in the middle, which can be used to insert especially symmetrical and cylindrical proteins into the bilayer. The holes are generated by applying weak repulsive radial forces around the area where the protein should be placed, moving the atoms apart from the membrane center. But there are only a limited number of pre-equilibrated membranes available. It is only possible to generate membranes with a maximal size of 90x90 $Å^2$ with maximal 128 lipids for each layer containing only one lipid type [JoKI07]. Therefore it is not possible to use this method in conjunction with heterogeneous membranes.

In addition, it is possible to automatically align proteins to the membrane by using OPM data or to align the proteins manually by selecting the appropriate values. If a protein/membrane complex is modeled, the workflow discussed above is extended by two steps.

In the Step 1, the manual placement is done by using the web interface with the following values:

- “Use PDB Orientation” should be applied if the OPM coordinates are used or the PDB model was prepared by the user manually on his local machine.
- “Align the First Principal Axis Along Z” should be used for homo-oligomer[46] or small helical bundles.
- “Align a Vector (Two Atoms) Along Z” may be needed to place irregular shaped proteins or hetero-oligomers.

There are also three options regarding the translation:

- translate the molecule along Z-axis in Ångstrom,
- flip the molecule along the Z-axis, and
- rotate the molecule with respect to the X-axis in Degree.

46 A special, relatively small type of molecule/protein.

In Step 2, the results of the placement are shown. Because all manual placement steps are defined by using the GUI in Step 1, it is not possible to interactively and visually translate and place the proteins. Therefore, CHARMM-GUI gives the user the opportunity to visualize the membrane complexes with the program MarvinSpace via the link “view structure” which is found behind every link to a generated PDB file. Unfortunately, this program often does not start, giving error messages like “Slab creation failedundefined” and if it does start, it is very slow. For this reason it might be a problem to judge the quality of the generated membrane without using PDB viewers on the local computer (see Section 4.2.1).

Step 2 is also correlated with the aforementioned Step 0, which means, that here the composition of the membrane is additionally defined. The generation process continues now with Step 3, which is equal to Step 1 mentioned above, and so forth.

Although the process of building a membrane is automatized, the authors concede in their publication that it is usually necessary to verify and adjust the resulting membrane model:

“Since Membrane Builder determines the system size in X Y only based on the size of the transmembrane domain, the system size generated by the default “1,5 lipid layers” was not large enough to accommodate the cellular domain” [BBMN09, p.6].

However, the workflow and the results provided by CHARMM-GUI are very powerful, especially for users of the commercial CHARMM software package. In addition, it provides output files compatible with NAMD/VMD (see the following section), so users of this software package also might be interested in using this tool. Analyzing publications citing [JLKI09] using Google Scholar [00b] and ISI Web of Knowledge from Thomson Reuters [11l], 14 out of 20 publications are using CHARMM, and even so six from 20 publications are using additionally or exclusively NAMD for the following equilibration and production MD runs (all citing publications from 8 July 2009 to 7 August 2011). But there seems to be no published results combining the Membrane Builder with GROMACS, although it was initially planned to provide simulation files also for GROMACS in near future [JoKI07, p.8]. Until now, this has not been implemented (as of 07.08.2011).

The reason might be that the Membrane Builder also has a number of disadvantages:

Although the generation process is well described in the publications, the source code is not available. Especially users of Open Source software like GROMACS prefer access to all data of the simulation process. It might be a problem to use a module in the modeling and simulation process which is a little bit like a black box.

In addition, the program is only available via the Internet. The reproduction of the results is only possible as long as the website is accessible and as long as the provided methods are not changed by the developers. While the membrane generation process from the CHARMM developers is free of charge (which of course could change in the future), the further simulation needs a paid license. These are problems which do not occur using Open Source programs.

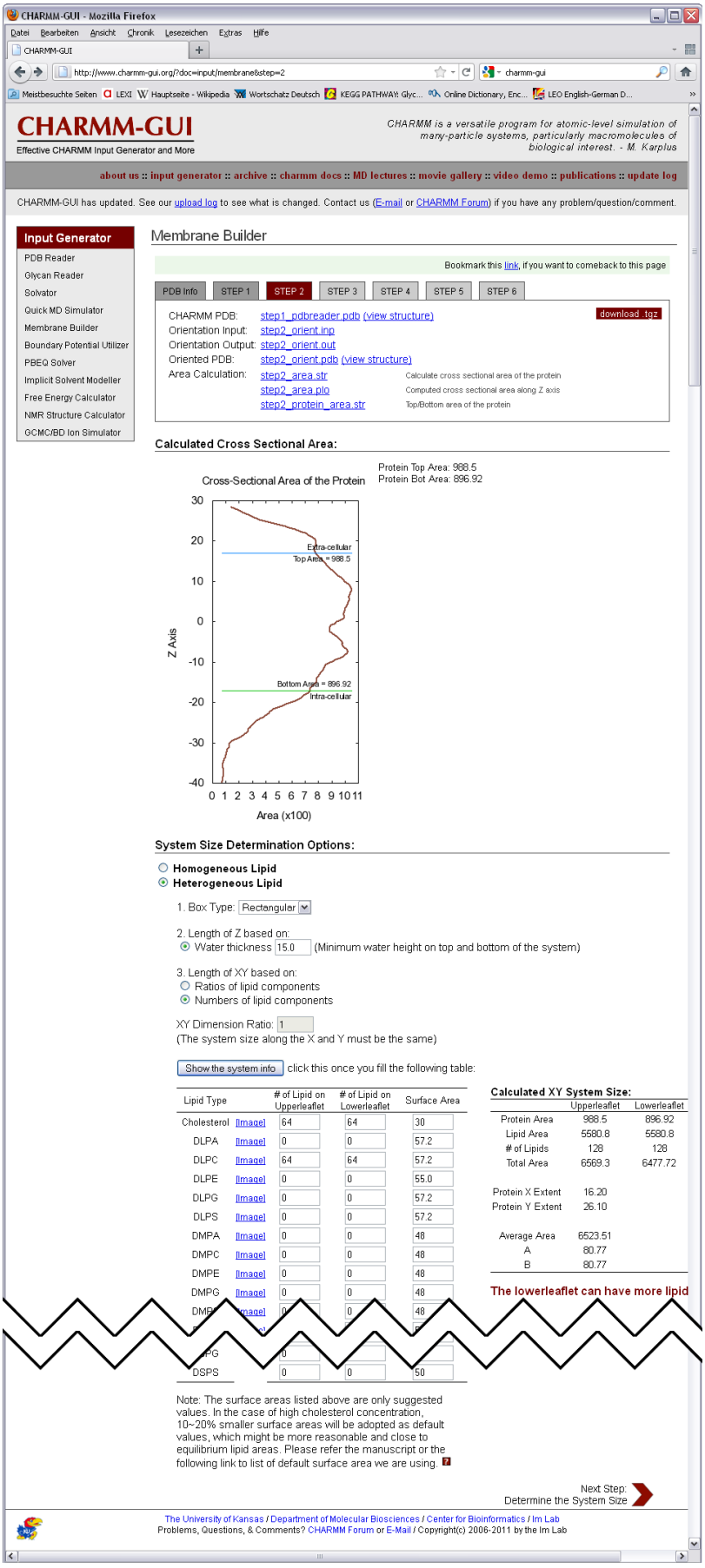


Figure 57: The Membrane Builder of CHARMM-GUI in step 2 of the modeling process
All available lipid types which can be used to model a heterogeneous membrane and the cross-sectional area of the protein to place (here: 1AHE) are shown

Of course, pre-equilibrated membranes are a nice feature, but regularly the user will have to rely on more complex, heterogeneous and/or larger membrane systems which have to be generated with the replacement method. But if the other criteria enable the usage of the insertion method, it might be anyway a problem to pre-equilibrate a membrane with a MD program and a parameter set which differ from the ones used during the following equilibration and production process using GROMACS or NAMD.

Another limitation is the use of only one protein per membrane systems. If the user wants to use protein assemblies, he has to combine the PDB files in the first step and export them to a single PDB files. The alignment method has been described before. Where it is possible, OPM coordinates should be used to place a protein, because the manual alignment with this package is neither interactive nor intuitive.

Moreover, it is not possible to use custom lipid models, therefore many MD packages like GROMACS will probably require a refactoring process of the PDB membrane exported from the Membrane Builder (Section 3.2.3).

The generation methods do not take collision-detection of covalent nor van der Waals radii into account. Therefore, collisions have to be eliminated by more expensive minimization and equilibration processes using MD packages. The collision-detection in CmME solves this problem right from the start.

To judge the packing quality of the membrane, the Membrane Builder uses the “average overall surface area” which is simply the box length raised to the second power and divided by the total number of lipids. This is the same method which is used by MembraneEditor (see Page 56, [BBMN09]). But additionally it uses the quite complex Voronoi tessellation to compute the “individual average surface area” [PVCJ04].

4.2.2.4 VMD Membrane-Plugin

VMD (Visual Molecular Dynamics) is a powerful and established program to visualize, analyze, edit and simulate molecular structures [HuDS96]. In contrast with GROMACS (Section 3.2.3) it can be used to directly visualize the results of molecular simulations. VMD supports plugins to extend the functionality of the software. One of these plugins is simply called “Membrane” and was developed by Ilya Balabin years ago [Bala06]. The web page concerning this plugin was first published on-line in 2006. It is a quite simple tool, but the fact that it is also part of the actual VMD release 1.9 shows, that it is usable for a large community. The command line tool of the membrane plugin can be used as follows:

```
membrane -l <lipid_name> -x <size_in_X> -y <size_in_Y> {-o <output_prefix>}
```

The variables show the capabilities of this tool: *-l* is followed by the lipid name or better: lipid type. The membrane plugin includes only two lipid types: POPC (a phosphatidylcholine) and POPE (phosphatidylethanolamine). Although it is stated at the page that the author wants to provide additional lipids in near future there seems to be no development in this matter for more than five years. The variables *-x* and *-y* define the size of the rectangular membrane. In contrast with ChemSite Pro, the shape of the membrane need not be quadratic in shape. But it is not possible to generate monolayers and the

generated bilayers may only contain one lipid type. Finally, the variable *-o* defines the prefix for the membrane output files. These files are two-fold: a PDB file containing the coordinate information and a *PSF file* containing the structural information of the membrane model.

The rectangular membrane is generated by combining hexagonal patches of POPC or POPE bilayers which have been previously generated and simulated. These patches have been additionally solvated by using the *Solvate plugin* from Grubmüller and Groll which generates an irregularly-shaped disordered water layer [GrGr96]. The resulting membrane structure can be directly used via VMD for MD simulations.

In addition, Ilya Balabin provided a simple method to place a protein into the bilayer generated before. But in addition to the PDB file containing the coordinates, again the PSF file is needed. Therefore VMD contains the plugin *psfgen*, which can be used to generate this file. Afterwards, the protein needs to be manually placed into the membrane. In addition, to the horizontal positioning the vertical positioning has to be done. The author proposes using as much experimental data as acquirable for this purpose. This means, there is not semi-automatic placement possible, everything has to be done manually. After the alignment, Mr. Balabin provides a script which is able to remove all overlapping lipids from the protein-membrane complex.

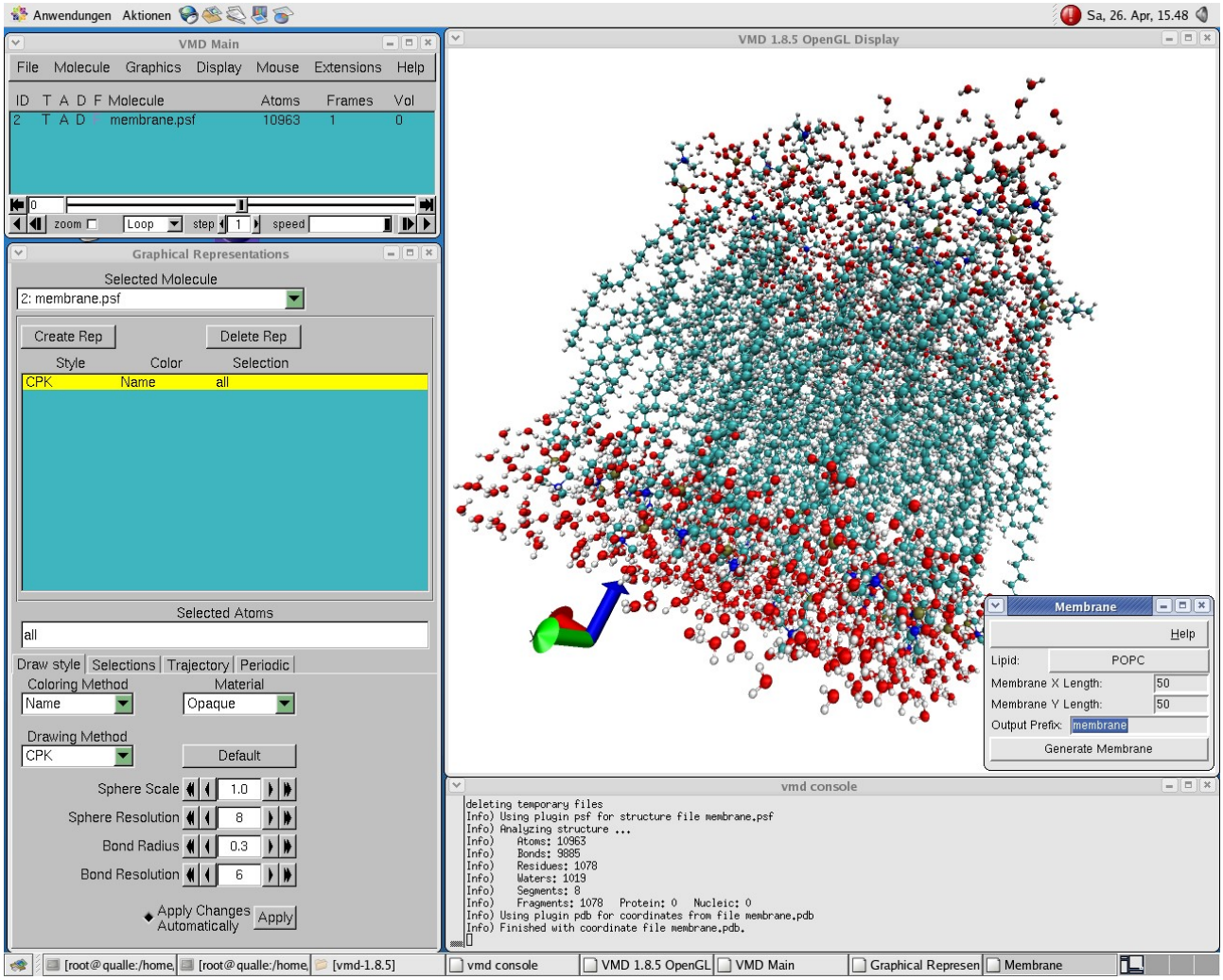


Figure 58: The Membrane Builder of VMD
It is the small program found at the right corner

4.2.2.5 Packmol

Packmol is a powerful command line tool which is able to generate molecular structures of different shapes like spheres, cylinders, planes, boxes and/or different mixtures. The old version was already used by many publications since 2003 [MaMa03]. The new version supports a more effective distribution algorithm combined with the ability to parallelize the computation [MABM09]. In addition, the algorithm automatically equilibrates the mixture.

Often this tool is used to generate homogeneous membranes. But there are also published configurations using heterogeneous membranes with Packmol, for example, from Hall et al. [HRKV10]. In this case, the user has to compute exactly the number of lipids which will be placed inside the membrane layer. All lipid models have to be pre-aligned, there is no automatic alignment done by Packmol. Therefore, the position and orientation along the vertical axis of the different lipid models should match. The placement of lipids can be defined via constraints. For example, it is possible to restrict the vertical position of the headgroups to be placed beneath a predefined plane. This is quite powerful, but also requires the user to have an exact view of the coordinate space. And although everything might be configured correctly, the generation of a heterogeneous membrane may be accompanied by many problems. Packmol immediately tries to equilibrate the structure. And this process often will not work if the structures are too complex, because the equilibration might never be reached. Accordingly the user has to use the try and error approach to find out, which size and lipid ratios are compatible. In addition, the final structure has to be equilibrated again if other tools like GROMACS are used (Section 3.2.3). In this case, the question arises if it makes sense to equilibrate a membrane model with two different approaches instead of using only the simulation software from the start applied to the final production runs. A minimal problem is the visualization: in Section 4.2.1 many PDB-based tools were discussed which can be used to examine the membrane.

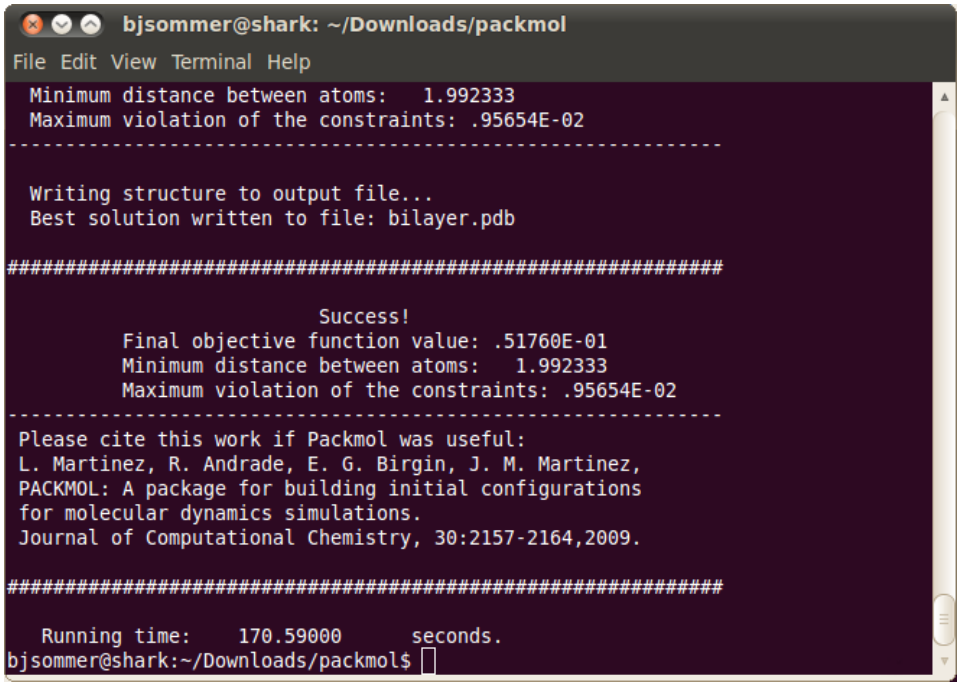


Figure 59: The execution of the Packmol script shown in Figure 60
Some information regarding the modeling and equilibration process are printed

bilayer-comment-100+100.inp (~/Downloads/packmol/packmol_membrane) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

bilayer-comment-100+100.inp

```
#
# Lipid double layer with water over and below
#

# Every atom from diferent molecules will be far from each other at least 2.0 Anstrons at the solution.

tolerance 2.0

# Coordinate file types will be in pdb format (keyword not required for pdb file format, but required
# for tinker, xyz or moldy).

filetype pdb

# The output pdb file

output bilayer.pdb

# Water molecules below the lipids: The first three numbers are the minimum x, y, z coordinates for
# this molecules, the last three are maximum coordinates. The box defined here has 56. Angstron sides
# in the x and y directions, and a 10. Angstron side in the z direction.

structure packmol_membrane/water.pdb
  number 2000
  inside box 0. 0. -10. 56. 56. 0.
end structure

# Water molecules over the lipids: The same as the input above, but the box of water molecules will
# be placed in a diferent region of space.

structure packmol_membrane/water.pdb
  number 2000
  inside box 0. 0. 28. 56. 56. 38.
end structure

# First lipid layer: the polar head is oriented to down to the water molecules.
# 100 lipids will be put inside a box of side 56. in the x and y directions and 14. in the z
# direction (14 is a little more than that length of the lipid). The atoms 31 and 32 of the pdb file
# of this lipid, which belong to the polar head, will be constrained to be below the the plane z = 2.,
# and the atoms 1 and 2, which are the hydrophobic end of the lipid will be constrained to be over
# the plane z = 12. Therefore, all the lipids will be oriented, with their polar heads pointing to the
# water box below.

structure packmol_membrane/water.pdb
  number 2000
  inside box 0. 0. -10. 56. 56. 0.
end structure

# Water molecules over the lipids: The same as the input above, but the box of water molecules will
# be placed in a diferent region of space.

structure packmol_membrane/water.pdb
  number 2000
  inside box 0. 0. 28. 56. 56. 38.
end structure

# First lipid layer: the polar head is oriented to down to the water molecules.
# 100 lipids will be put inside a box of side 56. in the x and y directions and 14. in the z
# direction (14 is a little more than that length of the lipid). The atoms 31 and 32 of the pdb file
# of this lipid, which belong to the polar head, will be constrained to be below the the plane z = 2.,
# and the atoms 1 and 2, which are the hydrophobic end of the lipid will be constrained to be over
# the plane z = 12. Therefore, all the lipids will be oriented, with their polar heads pointing to the
# water box below.

structure packmol_membrane/palmitoil.pdb
  number 100
  inside box 0. 0. 0. 56. 56. 14.
  atoms 31 32
    below plane 0. 0. 1. 2.
  end atoms
  atoms 1 2
    over plane 0. 0. 1. 12.
  end atoms
end structure

# Second lipid layer: the polar head points up to the water molecules. The same thing as the input
# above, but defining a new lipid layer, with the oposite orientation of the lipid molecules in such a
# way that the polar head points to the water box that is over the lipid bilayer.

structure packmol_membrane/palmitoil.pdb
  number 100
  inside box 0. 0. 14. 56. 56. 28.
  atoms 1 2
    below plane 0. 0. 1. 16.
  end atoms
  atoms 31 32
    over plane 0. 0. 1. 26
  end atoms
end structure
```

Makefile Tab Width: 8 Ln 17, Col 1 INS

Figure 60: A Packmol script for the creation of a bilayer
This is the original script containing comments from the Packmol author, only the membrane values were adjusted: each layer contains 100 lipids and the box has a size of 56x56x10 Å³

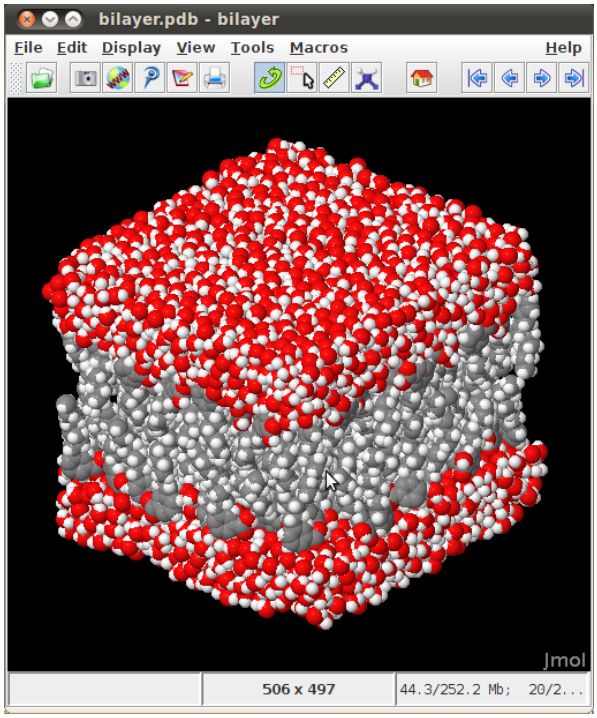


Figure 61: The membrane generated with Packmol in Figure 61 imported into Jmol.

Prior to the generation of a membrane, the program has to be compiled with a Fortran compiler. After this, Packmol is invoked with this script on the command line. Figure 60 shows a script which is able to generate a bilayer. It is based on the original script provided at the Packmol website, containing information about the size, the number of lipids, the file paths and the position of the molecular assemblies. It was extended to build a membrane containing 100 lipids in each layer with a side length of 56 Å. The output is given in Figure 61. The result is visualized in Figure 61 by using Jmol because Packmol does not provide a direct visualization.

4.2.3 Summary

The summary table 8 shows the programs related to MembraneEditor. The traditional membrane modeling methods discussed in Section 3.2.5 are not taken into account (see Section 3.2.5.3 for the discussion). They are represented in this table by the “MOE+script” approach, because this method shows the way user-defined membrane modeling methods could be implemented.

The ChemSW is the only plugin offering a fast and easy way to generate model membranes, but this grid-based approach generates only very simply-structured membranes. The VMD Membrane plugin is also quite easy, but it supports only the generation of homogeneous membranes, providing two lipid types. MOE is only able to generate membranes if the user creates scripts. The creation of complex membrane models will be quite a huge problem.

But membrane modeling for professional applications can only be achieved with the tools CHARMM-GUI and Packmol. CHARMM-GUI already provides many options, especially for the generation of heterogeneous membranes, but of course it is optimized for use with CHARMM-based force fields. It is not possible to use customized molecule models like special lipids. In addition, it might be a problem to use a membrane which was equilibrated

based on the CHARMM force field with another application like GROMACS which is usually used in conjunction with other force fields. The same problem applies to Packmol which uses a custom equilibration process. The use of Packmol is more complicated and will be a big obstacle for a biologist who is not familiar with the mathematical basics of a coordinate system and geometry. The lipid models used with Packmol have to optimized and geometrically adjusted. For this purpose, other tools are needed, for example, PyMol or the PDB files and the coordinates have to be written from scratch. In addition, Packmol does not provide an option to automatically use the alignment of OPM tools; therefore all proteins also have to be placed by hand. This is an advantage CHARMM-GUI offers.

Moreover, none of the tools discussed here provide a three-dimensional visualization of the membrane generation process.

In Subchapter 5.2 the requirements will be discussed to establish an application which unites the positive aspects of the related approaches discussed here.

Category	Feature	Charmm GUI	ChemSW	MOE+ Script	Packmol	VMD MP
Availability	Standalone		X	X		X
	Command line tool				X	
	Web service					
	Website	X				
	Web Start					
	Source Code				X	X
Computational Acceleration	Multithreading					
	Parallelization				X	
Direct Database Connections	PDB database	X		X		
	PDBTM database					
	OPM database	X				
Formats	GRO					X
	native format		X	X		
	PDB	X	X	X	X	X
Libraries	Lipid library (>2 lipid types)	X	X			
	Lipid library compatible to MD	X	X			
Modeling	Atom-based Molecule Editor		X	X		X
	Bilayers	X	X	X	X	X
	Collision-Detection				X	
	Counterions support	X				
	Lipid Packing Density	X				

Category	Feature	Charmm GUI	ChemSW	MOE+ Script	Packmol	VMD MP
	Heterogeneity	X	(X)		X	
	Hexagonal Shape	X				
	Relative Lipid Ratio	X				
	Monolayers		X	X	X	
	Multilayers				X	
	Percentaged Distribution	X	X			
	Protocol					
	Raft Support				X	
	Rectangular Shape	X	X	X	X	X
	Reproducibility					
	Total lipid number	X		X	X	
	Water Layer Build	X	X	X	X	
Pipelines	Ext. Simulation Package	X	X	X		X
	Ext. Visualization Package	X	X			X
Programming	Scripting			X	X	X
	Scripting at Runtime					X
Protein Placement	Automatic Protein-Placement using OPM	X				
	Manual Placement			X	X	
	Semi-automatic Protein Placement					
Simulation	Equilibration	(X)		X	X	X
	Minimization	(X)	X	X		X
	Simulation		X	X		X
Visualization	Atomic Structure View	X	X	X		X
	Graph Visualizations			X		
	Live Distribution Visualization					
	Secondary Structure View			X		X
	Raytracing		x			
	Runtime Graphs					
	Stereo Support			X		X

Table 8: A comparison of all membrane modeling tools discussed in this work
The complete table is found in Appendix 9.19.2

4.3 [FUN] Network Modeling Tools

The structural modeling tools related to this work have been discussed. Now the functional level has to be addressed. The modeling of biological networks is part of the daily work of many Bioinformatics scientists. This subchapter explains different approaches of network modeling. The subchapter is subdivided into three categories: two-dimensional (Section 4.3.1), two-and-a-half-dimensional (Section 4.3.2), and three-dimensional (Section 4.3.3) network modeling tools.

4.3.1 2D Network Modeling Tools

The *two-dimensional modeling* of biological networks is an extremely large field. There are hundreds of tools which can be used or utilized to create, import and export, grow, reduce, enrich and explore these pathways.

An analysis of these tools lies behind the scope of this work; especially because this work is focused on three-dimensional modeling. Explicitly this thesis takes only those approaches into account which exceed the second dimension starting with the next section.

Nevertheless, three tools should to be mentioned exemplarily in this section, because they will be relevant for the following section: Cytoscape, VANTED and VANESA. A fourth tool was already discussed in Section 3.3.3.2; ANDVisio.

4.3.1.1 Cytoscape

Probably the most famous two-dimensional network modeling tool in Bioinformatics is *Cytoscape* [SMOB03]. It is an open source application and provides a plugin interface which was used by hundreds of research projects. An important plugin in context of this work is BioNoM, which enables the export of SBML files [12g] (Section 5.3.2.12.2). Experimental molecular and genetic data can be integrated into the system as well as data coming from different databases. A vast number of layout algorithms is available for this tool – to name but one – like the commercial *yFiles* package [11m].

Recently, an extension for the direct visualization of Cytoscape networks was introduced which will be discussed in Section 4.3.3.6.

4.3.1.2 VANTED

VANTED (Visualization and Analysis of Networks with related Experimental Data) is a Java Web Start application which is developed at the IPK Gatersleben. This tool was initially intended as a pure two-dimensional modeling environment supporting exploration and analysis of experimental data [JuKS06]. For this purpose various external databases like KEGG (Section 3.3.2.1.1) as well as in-house databases like MetaCrop were integrated [SCCG11]. First published in 2006, a large number of biological and/or analytical works benefiting from this tool exist today.

Right from the beginning, VANTED featured a well-defined plugin interface enabling the development of various extensions. Two of these – going beyond two-dimensional visualization – are discussed in Sections 4.3.2.1 and 4.3.3.5.

4.3.1.3 VANESA

VANESA (Visualization and Analysis of Networks in Systems Biology Applications) is a Java-based tool developed in the Bio-/Medical Informatics Department Bielefeld during the recent years [JKTH10]. It can be used to query many databases contained in DAWIS-M.D. (Section 3.3.3.1). It was especially optimized to generate and layout biological networks of different types such as protein-protein interaction networks, metabolic pathways and/or signaling networks. The generated networks can be analyzed with VANESA by using graph-theoretical approaches, for example, shortest-path algorithms. Moreover, Petri net simulations are possible using a special plugin.

VANESA as well as VANTED nowadays also support the three-dimensional visualization of networks, although this is not their main application area. Similar approaches are discussed in Section 4.3.2.1.

4.3.2 2.5D Network Modeling Tools

To the knowledge of the author, the first approach combining two-dimensional visualization with the idea of three-dimensional positioned areas was published in 2004. This approach will be discussed in the next section. Today a number of very similar programs exist which will also be mentioned in the proceeding sections. The idea of 2.5D visualization will also be found at the molecular level. The idea will be discussed in Section 5.2.2.5.

4.3.2.1 2.5D Network Comparison

As mentioned before (Section 4.3.1.2), VANTED is often used to visualize and analyze KEGG maps (Section 3.3.2.1.1). Those KEGG maps usually handle specific biological pathways which apply to a large number of organisms. Naturally, these organism-specific pathways do often only vary in details. To create the possibility to also compare small differences across species, Brandes, Dwyer and Schreiber introduced a method to compare these pathways with each other, using the software package *WilmaScope*. Initially, WilmaScope was created as a Java 3D application to visualize UML diagrams [DwEc04]. Then it was extended to visualize networks in three dimensions. The development seems to have stopped since 2004.

Using WilmaScope, a set of (two-dimensional) layers are stacked on top of each other in three-dimensional space. Best visual results are achieved if the data is relatively consistent. If the deviation is too large, the visual comparison will be very difficult. Therefore, a poor visualization might be also an indicator for bad data quality. Figure 48 shows a pathway stack of seven related pathways [BrDS04a, BrDS04b].

A similar approach is now also part of the VANTED tool discussed before. The major difference is that WilmaScope is able to sort the different networks based on their compositions using a special heuristic. Therefore, the visual quality of the stacking in WilmaScope currently outperforms the one of VANTED.

A similar idea is the comparison of different overlapping networks in three-dimensional space, Figure 63 shows an example. The first pathway, a signal transduction network (*TGF ß*), is placed on layer 1 and the second pathway, a related nuclear protein interaction network, is placed on layer 2. In 3D space a third layer is centrally inserted, parallel to these two network layers. Only those nodes are mapped onto this layer which are present in pathway 1 as well as in pathway 2. Using this method, new insights into interrelations of functional differing biological complexes are achievable [FHKS08].

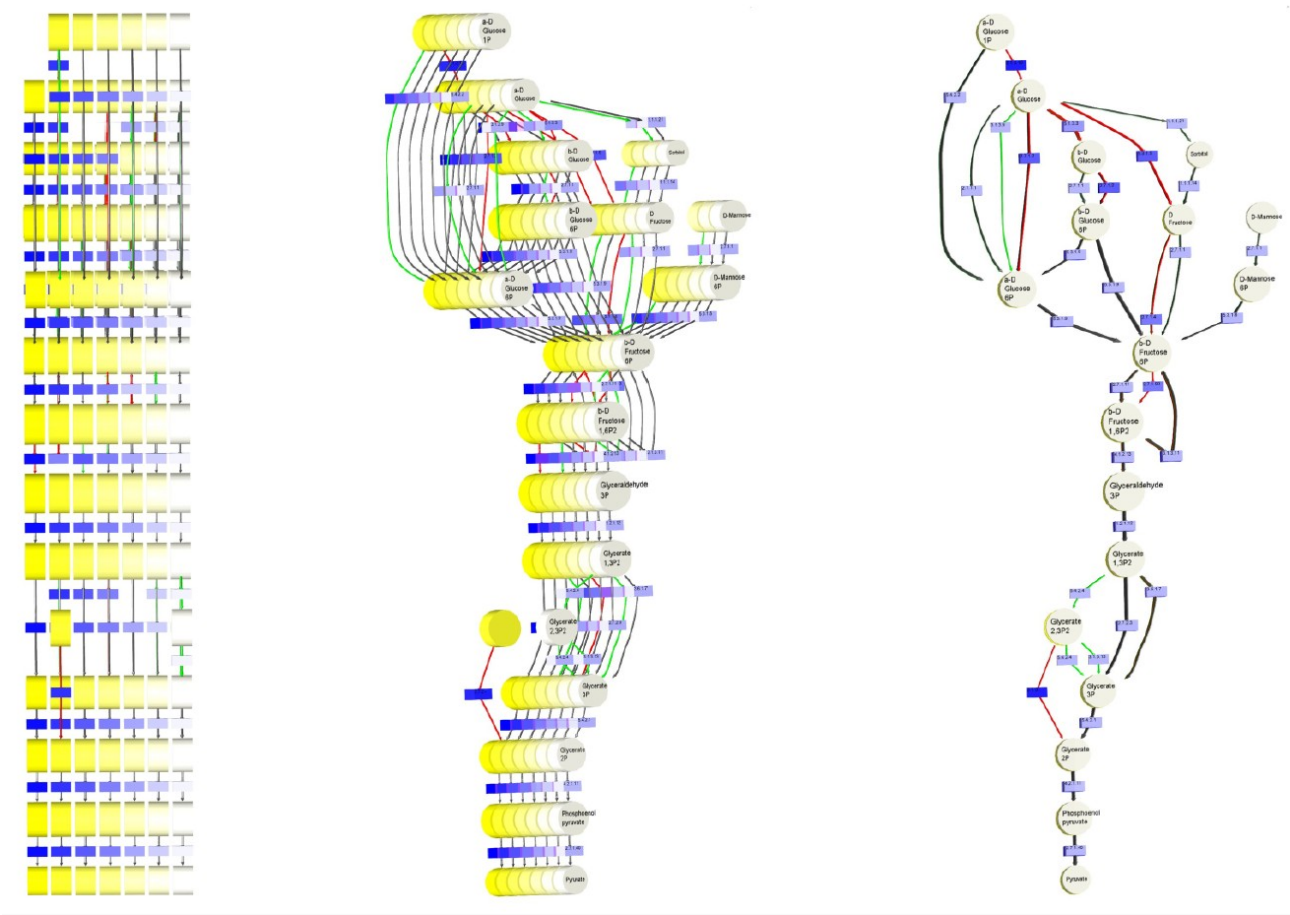


Figure 62: 2.5D visualization of seven related pathways (parallel projection) (Courtesy of/Copyright © 2004 by Journal of Integrative Bioinformatics. Reprinted with permission from [BrDS04b])

4.3.2.2 Arena3D

Arena3D advances the idea of the previously discussed approach. Instead of comparing only two different networks, multiple networks are compared. This approach should enable inter-domain comparison of large-scale networks which are – in the definition of the related publication – hundreds of nodes and thousands of connections [POSS08].

Different methods like Fruchterman-Reingold or hierarchical layout algorithms can be applied (Section 3.3.4.2). For example, Figure 64 shows a network originated by the Huntington´s disease. On the first layer, the involved genes are shown, then, on the second layer, the gene-encoded proteins are shown. And finally, the third layer shows the structural associations. The circle-like structure and the grouping of nodes gives an idea of the method of the clustering algorithms. Moreover, the new version 2.0 introduces the comparison of phenotype-related networks over time [SPAS12].

The application is accessible via Java Web Start and the 3D visualization via Java 3D.

4.3.2.3 BioCichlid

BioCichlid is another tool which focuses on the combination of genetic and physical data into one visualization featuring only two layers. Just like the new version of Arena3D, it is also able to integrate time-dependent data with animation [IMOT09].

Figure 65 shows a hierarchical layout of an RMAPII-related network. On the bottom layer,

red nodes represent the genes and on the top layer, blue nodes symbolize the gene-encoded proteins. The coloring of the edges codes different interaction types: red, genetic interactions; blue, physical interactions and yellow, transcriptional regulations from transcription factors to genes (Section 2.2.1.2).

BioCichlid is based on the Cichlid client-server system. The networks are generated using a website. Three different datasets have to be uploaded by the user: a physical interaction dataset, a gene expression dataset and a transcriptional dataset (containing the physical-genetic interactions). After this workflow is finished, a local client has to be installed to enable the user to download, visualize and animate the data. But until recently (22.11.2011), only an executable file for Mac OS X is provided, although the publication dates back to 2009. Moreover, the original website is currently not available (as of 31.07.2012).

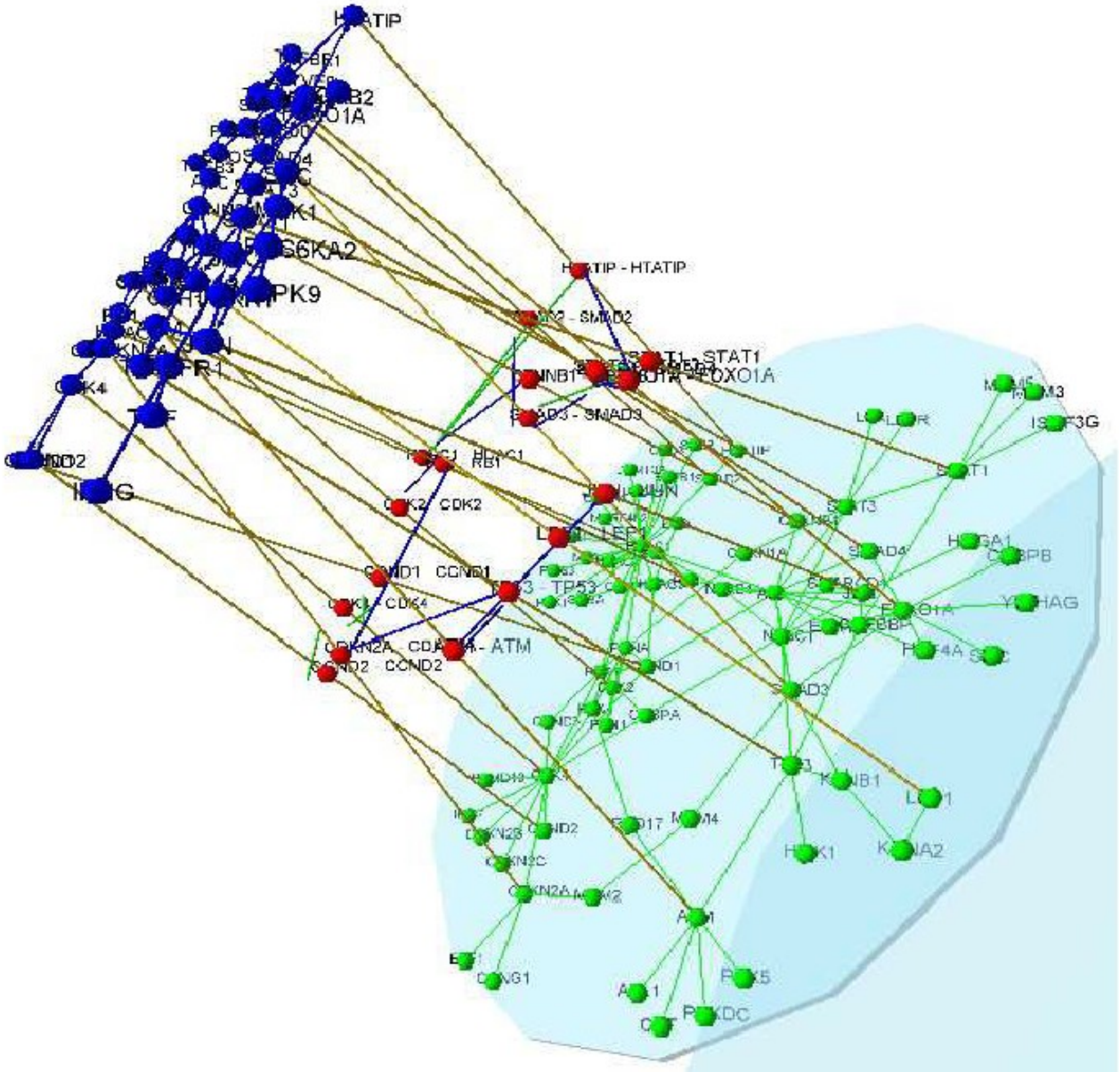


Figure 63: 2.5D visualization of two different pathways
A signal transduction network and a corresponding part of a nuclear protein interaction network (Courtesy of/Copyright © 2008 by Journal of Integrative Bioinformatics. Reprinted with permission from [FHKS08])

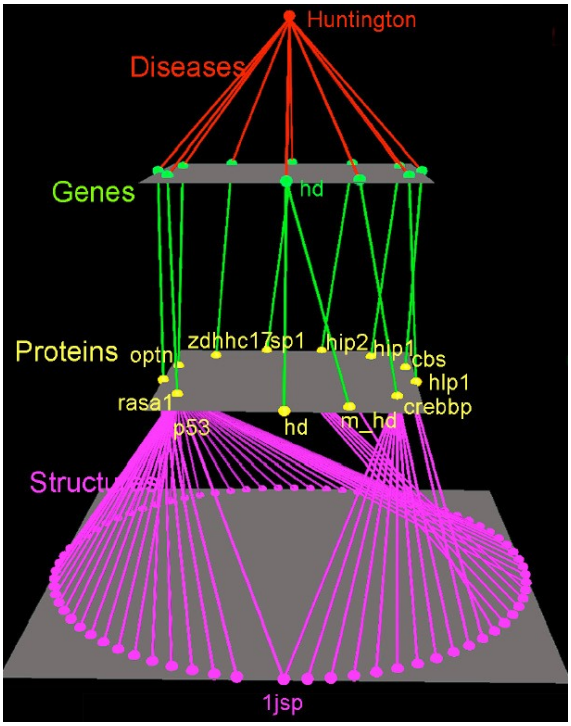


Figure 64: Arena3D: hierarchical 2.5D visualization of a Huntington´s-disease-related network On three layers (Courtesy of/Copyright © 2008 by Georgios Pavlopoulos. Reprinted with permission from [POSS08])

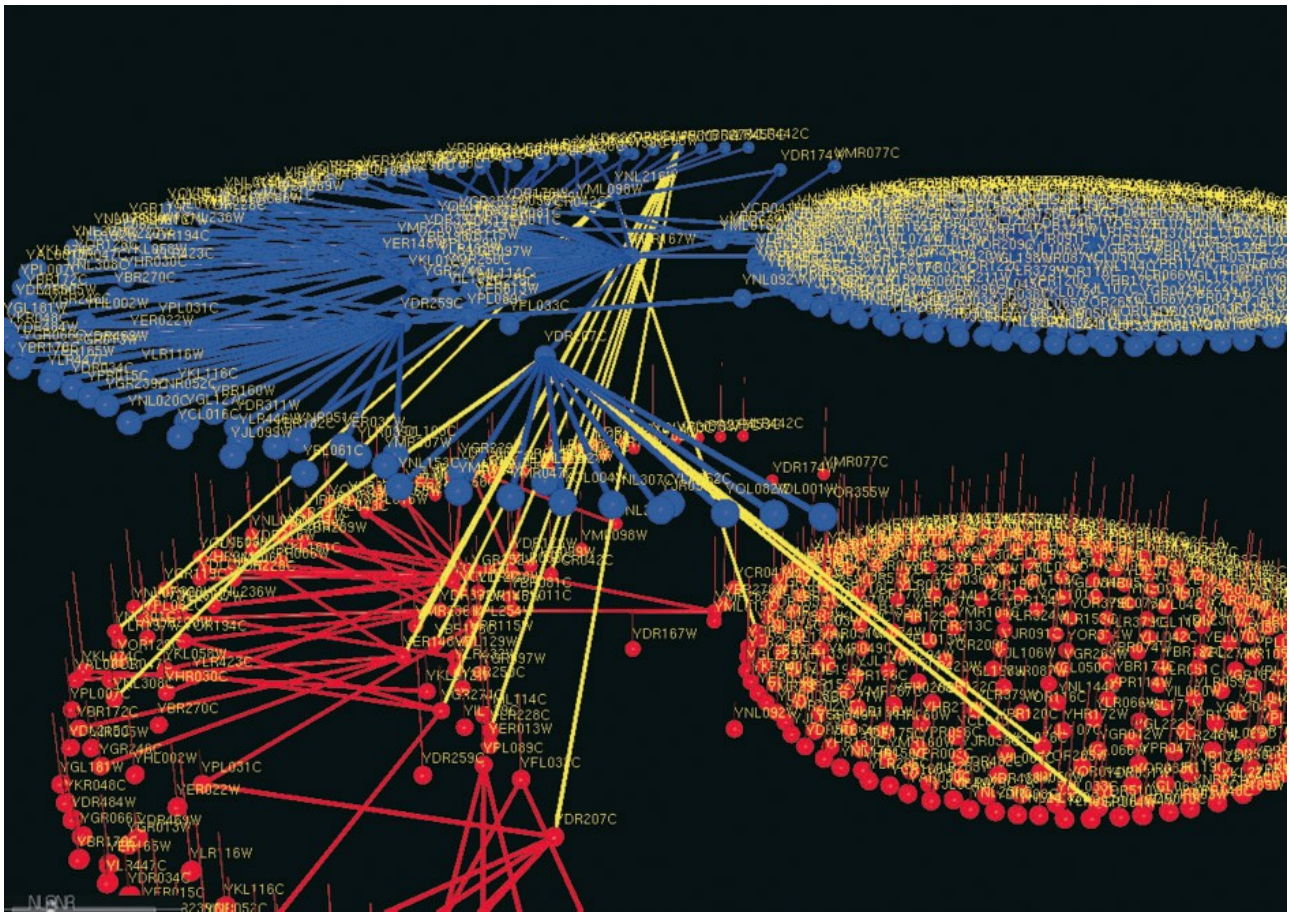


Figure 65: BioCichlid: 2.5D visualization
Of a gene cluster (red) and a protein cluster (blue) and their interactions (Courtesy of/Copyright © 2009 by Oxford University Press. Reprinted with permission from [IMOT09])

4.3.3 3D Network Modeling Tools

In the preceding section the step between two and three dimensions has been discussed. Different two-dimensional layers holding networks featuring a two-dimensional layout were aligned in three-dimensional space mainly for comparison issues. The following section now deals with networks which are not restricted to multiple layers. The elements of these networks can be theoretically placed anywhere in the three-dimensional space. Therefore, the positions of these network elements are based on three-dimensional layouts.

In addition, it should be mentioned that only those projects lie in the scope of this work which provide interactive elements. Static three-dimensional visualizations of cells consisting of a number of rendered images and/or animations, are not taken into account.[47]

4.3.3.1 Metabolic Pathways in VRML

Igor Rojdestvenski is one of the pioneers in the area of 3D visualization of biological networks. In 2002/2003 he introduced the *Metabolic Network Visualizer (MNV)*. It supported an XML-based format which showed similarities to the currently established SBML language [HFSB03]. His approach was available to web-users by a simple HTML-front-end. The website was implemented using CGI programming. It generated VRML models (Section 3.1.3.4) which could be visualized with VRML-browser-plugins like Cortona3D [11n]. There was a constructor-mode which enabled the user to generate custom pathways and, respectively, a viewer-mode which could be used to visualize pathways [Rojd03a, Rojd03b]. The MNV prototype was applied to the AgroCyc© database [KRPP02].

The major intention of MNV was the improvement of the visualization of complex data structures by taking advantage of three-dimensional space. Especially aspects like line-crossing and the reduction of topological constraints known from two-dimensional visualizations should be omitted. For this purpose the MNV implementation followed the Focus+Context paradigm [RoFl97]:

- *focus:* the user can concentrate at a special *point of interest (POI)* while simultaneously keeping track of the
- *context:* the current, relative position in the whole data pool.

An automatic layout was used for the network visualization based on the *spring embedding algorithm* [KaKa89] (see also Section 3.3.4.2). In addition, Rojdestvenski used already different visual attributes like color, shape and/or transparency to visualize different properties of network components.

Unfortunately MNV is no longer available.[48]

47 The interested reader is referred to an informative article concerning molecular rendering from Lok [Lok11].

48 There are low quality screenshots still available via the Internet Archive WayBackMachine [11o]. These screenshots are part of the supplementary material of the major publication [Rojd03a]

4.3.3.2 MetNetVR

A first prototype of *MetNet3D* developed at the Iowa State University was introduced in 2003 [DYBR03]. This system used the databases MetNetDB [WLDZ03] and ARACYC [RBBC03] as a starting point to generate metabolic networks in 3D. Already at that time there was a plan to correlate these networks with a Virtual Cell environment. A *CAVE (Cave Automatic Virtual Environment)* with 3D stereoscopic glasses was used to visualize the network structures. Furthermore a small tablet PC was applied to navigate through the environment. A VR environment called VR Juggler providing a JAVA GUI was combined with an OpenGL-based API, OpenSG, and R, supporting on-line clustering and other data processing techniques.

The molecular networks contain cubes representing cell components, nodes representing proteins as well as genes and cylinders representing connecting edges. Different layouts algorithms were implemented like a fan layout, a GEM-3D layout [BrFr96] and a radial layout [Eade92]. In addition, the final published version of MetNet3D released in 2005 combined metabolic networks with gene expression data [YEWC05].

This approach is especially important in the context of this work because of its extension to hierarchical layouts first introduced in 2006, called *MetNetVR* [YWCD06]. This update included a first approach of the compartmentalization of a molecular network. Figure 66 and 67 show such a network in which molecules are involved in six different cell components. The overview of the whole network is visualized in Figure 66. Three Arabidopsis-related pathways are combined here: Acetyl-CoA biotin, starch degradation and starch synthesis. The compartments are symbolized by simple, labeled boxes. In Figure 67 the plastid has been expanded, showing the whole network correlated with this cell component. This feature, namely *detail-on-demand*, was implemented to reduce the complexity of the layout.

Another interesting aspect for this work is shown in Figure 68: a chloroplast modeled by Andres Reinot. This is not an interactive model part of MetNet3D, this is rather a vision for the future development of MetNet3D. This graphic can be found in the MetNet3D publication from 2003 [DYBR03]. This vision of a sub-compartmental localization was never realized by MetNet3D, but it will be an important topic in this work (Subchapter 5.3). The sources of a functional program of MetNet3D/VR are also not available.

4.3.3.3 GEOMI: Protein Interaction Networks in 3D

While the preceding tools were mainly for visualizing metabolic networks, the following approach deals exclusively with protein-protein interactions [HoWW07].

The according publication discusses a dataset containing 1379 Saccharomyces cerevisiae proteins and 2493 interactions from yeast two-hybrid experiments. The proteomic classification is based on The Gene Ontology database (see Section 3.3.2.2.4). The authors were able to achieve subcellular localizations for approx. 85% of the dataset according to the aforementioned publication.

Special Perl scripts were generated to integrate the interaction data sets and associated proteomic data into an XML-based format.

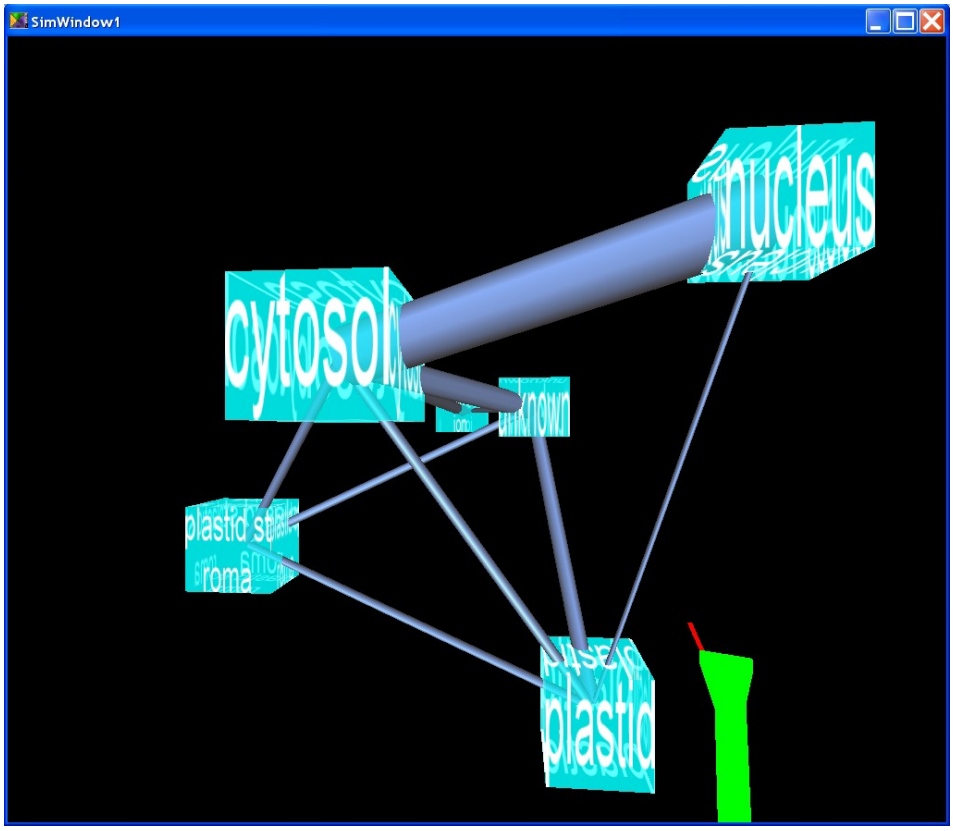


Figure 66: MetNetVR: this network contains molecules from six compartments (Courtesy of/Copyright © 2006 by Association for Computing Machinery, Inc. Reprinted with permission from[YWCD06])

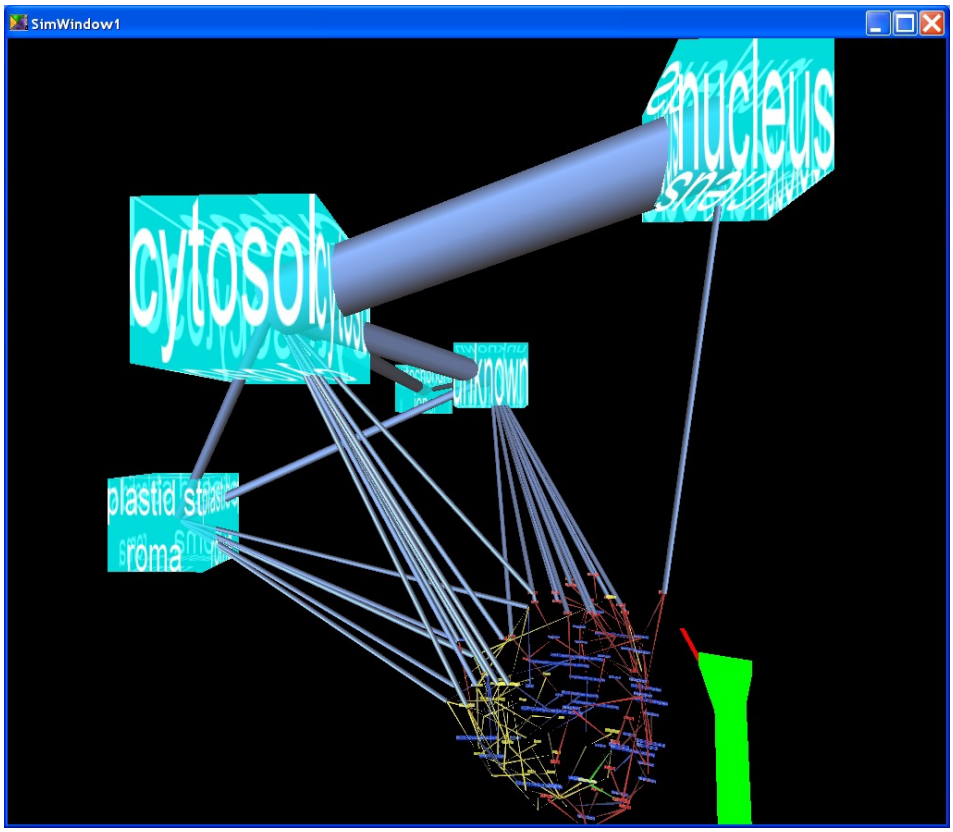


Figure 67: MetNetVR: the plastid was expanded showing the correlated network (Courtesy of/Copyright © 2006 by Association for Computing Machinery, Inc. Reprinted with permission from[YWCD06])

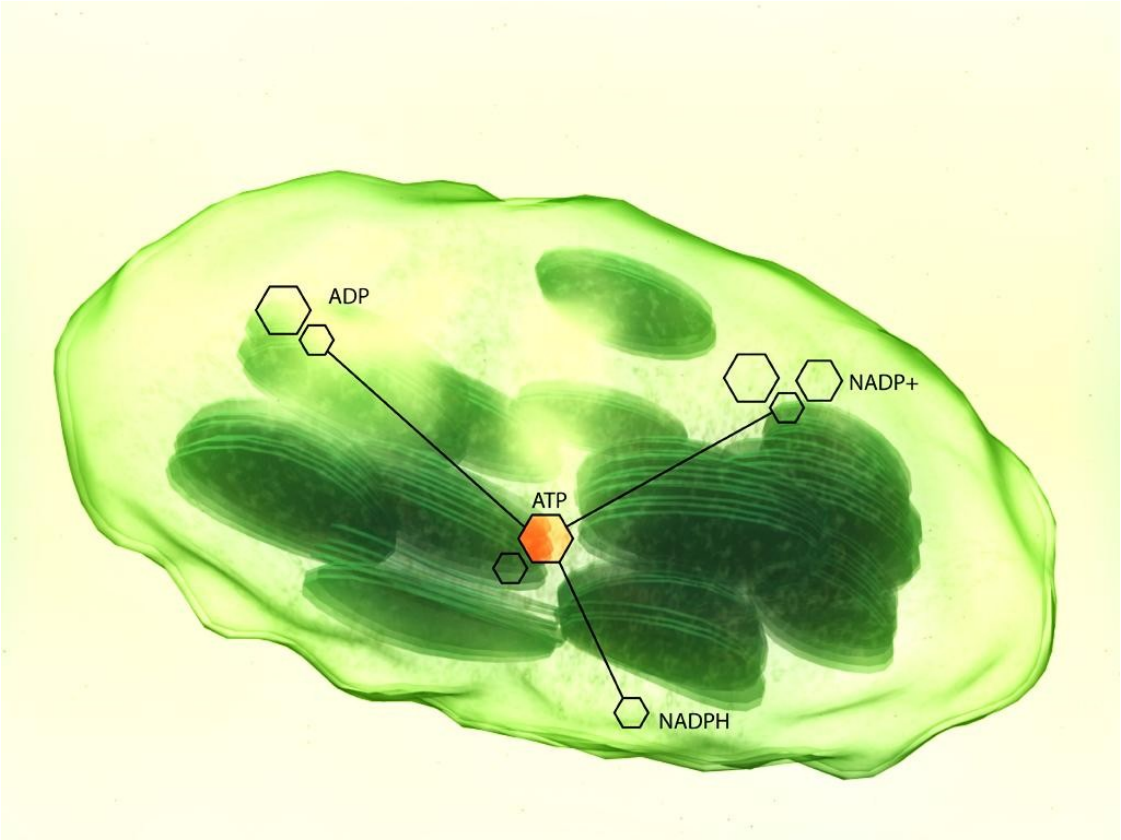


Figure 68: A visionary static rendered image of a Chloroplast
(Courtesy of/Copyright © 2006 by Andres Reinot. Reprinted with permission from [Rein11])

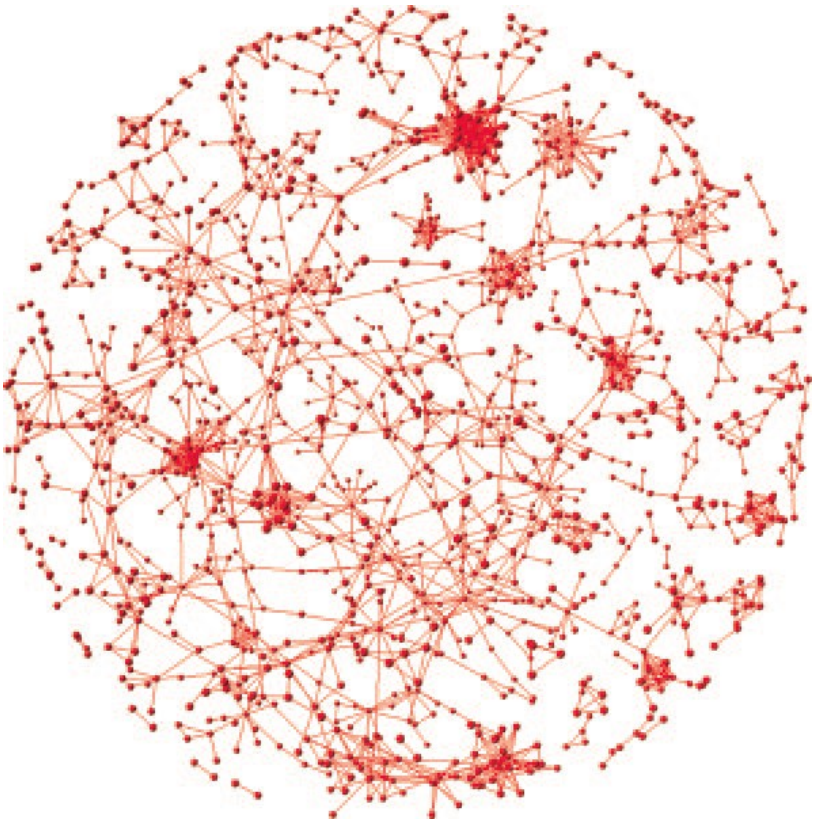


Figure 69: GEOMI: The whole filtered yeast interactome dataset
Visualized as a spherical force-directed layout (Courtesy of/Copyright © 2007 by American Chemical Society. Reprinted with permission from [HoWW07])

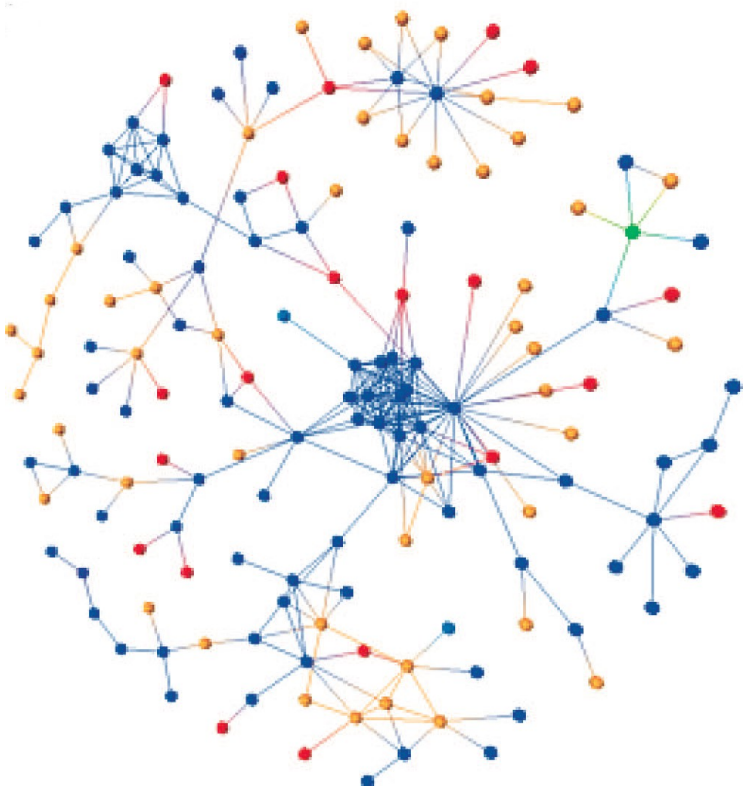


Figure 70: GEOMI: The proteomic nodes are colored according their localization
Blue: nucleus; light blue: nucleolus; green: mitochondrion; gold: cytoplasm; red: unknown
(Courtesy of/Copyright © 2007 by American Chemical Society. Reprinted with permission from [HoWW07])

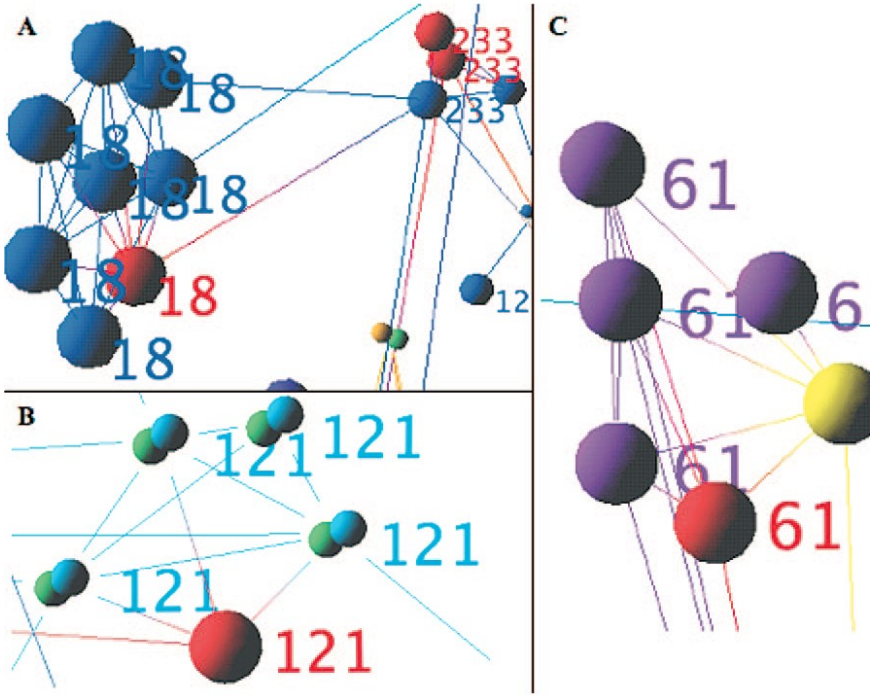


Figure 71: GEOMI: One protein found in multiple localizations
(Courtesy of/Copyright © 2007 by American Chemical Society. Reprinted with permission from [HoWW07])

This XML file was imported into software called *GEOMI (Geometry for Maximum Insight)*. The interactions are symbolized as edges and the proteins as nodes.

The network is constructed based on a force-directed layout. Different parameters like the repulsion or the spring parameter can be adjusted to change the structure of the network. The network is projected into a sphere which can be navigated or projected onto a two-dimensional circle.

GEOMI provides different interesting approaches to visualize protein-protein interactions (Figure 69). For example, it groups together proteins according to their membership in a stable complex and/or bilateral interactions. In addition, it is possible to adjust the complexity of the visualization, because different visual experiments revealed that the visualization of the whole interactome together with all associated proteomic data would be visually overwhelming.

But for this work, particularly the ability to handle differing protein localizations of interest. The node colors are especially used to visualize different aspects of a protein like interaction types and other proteomic data, and of course: localizations. In Figure 70, five different localizations are color-coded: blue: nucleus; light blue: nucleolus; green: mitochondrion; gold: cytoplasm and red: unknown. In addition, GEOMI is able to handle different aspects for one protein. In such a case, multiple nodes are drawn. Figure 71 shows different approaches where multiple localizations have to be visualized.

Figure 71.A shows eight components of complex 18. Seven of these components are localized at the nucleus (navy). Only one component, FIP1 is unannotated (red). By the colocalizations of the other components it is a simple task to predict the nucleus as localization for FIP1. In Figure 71.B, complex 121 is localized at the bud neck (green) as well as the cell periphery (blue). CDC3 is an unannotated member (red). But the localization can be inferred by the other four proteins. Finally, in Figure 71.C, most components of complex 61 are localized at the actin (purple). But one element, ARP2, is localized at the punctate composite (yellow). In contrast to the preceding two examples, the unannotated element, ARP3, is connected to neighbor nodes with different localizations. Because ARP2 is central to this stable complex and all other elements are localized at the actin, also ARP3 can be inferred to be localized at the actin.

4.3.3.4 The Interactorium

In contrast with the previously discussed work, the *Interactorium* integrates different cellular levels into one application: the protein as well as protein complexes, protein interaction networks and compartmental localization [WPLW09]. This is also the application closest to the integrative level.

An existing project developed by one of the authors – the *Skyrails system* – was used, to generate a virtual yeast cell. Developed in C++ and OpenGL, the biggest advantage of this system is the high performance provided: according to the authors, 6,000 complexes containing 40,000 protein nodes can be visualized.

The interaction data was taken from two-hybrid experiments, affinity purification studies and extended by information from literature. To generate the edges of the network, the data was weighted:

1. the lowest rank applies to all data where the following two criteria do not apply (the publication lists a table with all relevant interaction types which apply to rank 1, [WPLW09, table 1]),
2. the middle rank was given to interactions resulting from high-throughput two-hybrid experiments and
3. the highest rank applies to interactions detected by using high affinity capture and co-crystal structure techniques.

The overall weight of an edge was computed by summarizing all weights of an interaction. The edge weight affects its visual thickness as well as the strength of the spring forces (Figure 72). Again, a spring-embedding layout is used to generate the placing of the nodes [Eade84].

In addition, a special grouping technique was applied to define protein complexes automatically.

A unique feature of the Interactorium in comparison to the other tools discussed here, is the ability to integrate PDB structures into the visualization (Figure 73). If one protein node is fixed and a PDB structure is provided, the three-dimensional model is loaded and shown in the borders of the selected protein node circle.

Three different visualization modes are provided which can be dynamically switched:

1. the interaction network view,
2. the protein complex view and
3. the protein structure view.

The cell contains different cell components which are visualized as sphere-like geometrical structures. Each cell component type has a different color (Figure 74). In addition, these cell components feature partly specific decorations, but they are hardly recognized. The nodes are placed onto these cell components according to their localizations. If nodes are localized at the cytoplasm or actin, they are placed outside the cell components and inside the outer spherical border representing the cell membrane. The nodes are represented by double-bordered spheres.

The navigation is fully interactive. All elements are clickable. The menus and information boxes are shown inside the Virtual Cell environment. It is also possible to highlight different edges according the involved experiments.

Another similarity to GEOMI is the handling of different localizations for one node. If a node is located in more than one cell component, an edge labeled with the term “twin” leads to the other localization (Figure 72). The node is visualized at each position, but interaction edges are only drawn between those nodes which are localized at the same cell component.

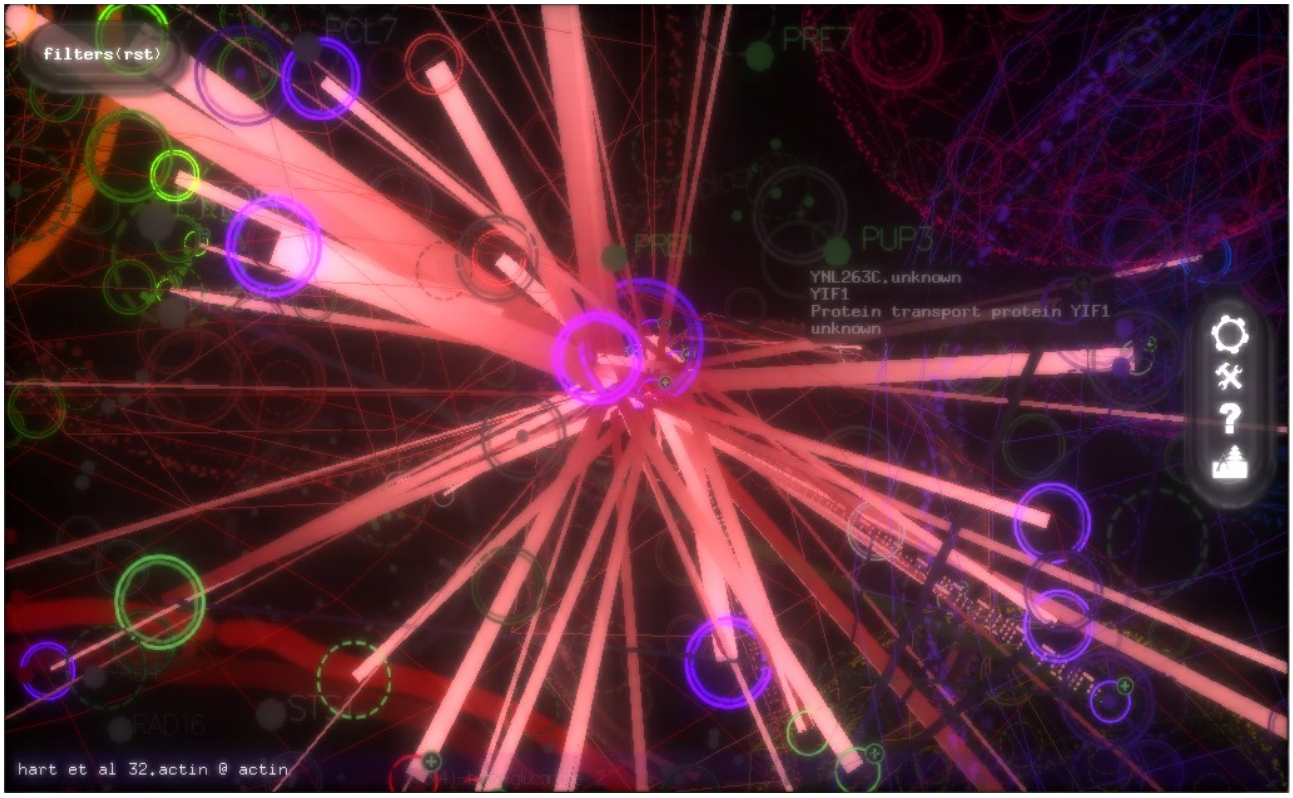


Figure 72: The Interactorium: a central node
It is connected by different edges to interacting proteins. The edge labeled with "twin" (right bottom corner) leads to the same protein type localized at a different cell component

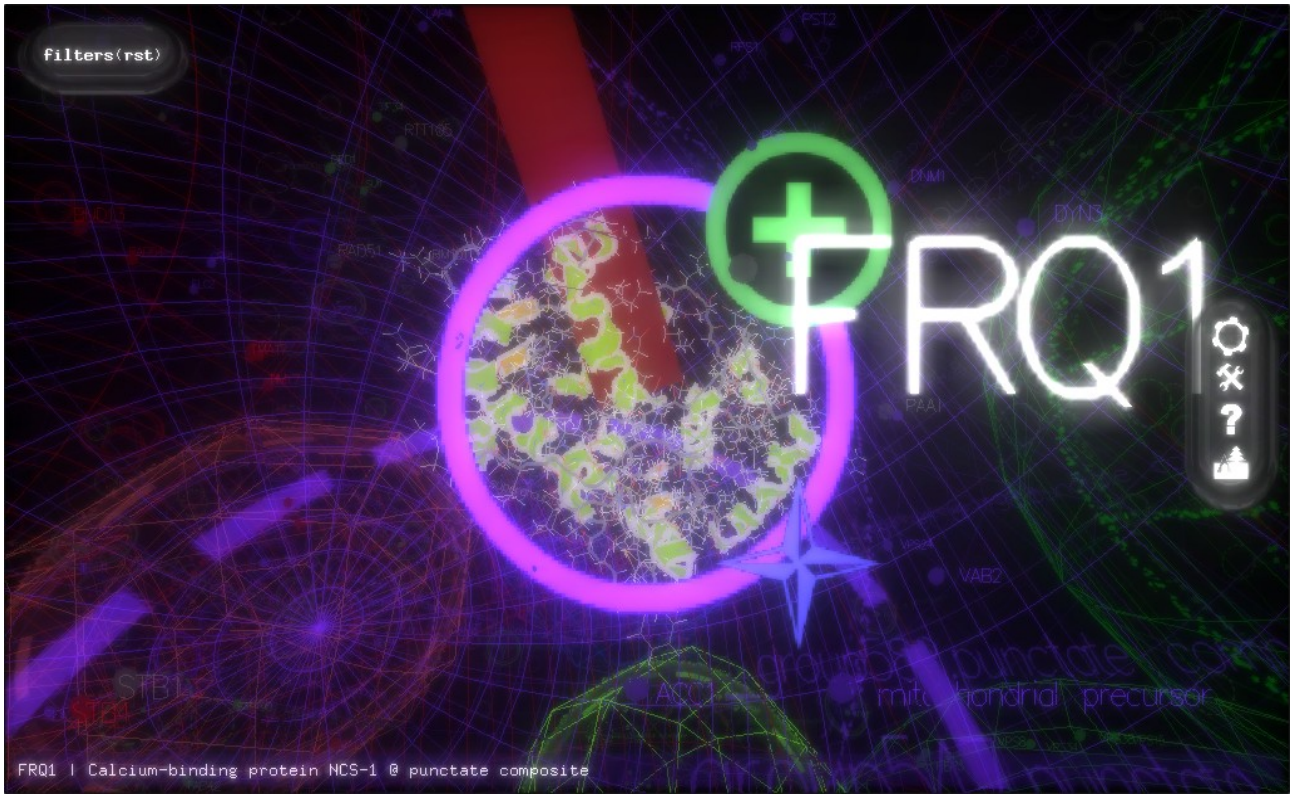


Figure 73: The Interactorium: for a few proteins structural data is provided
This data is only shown if a node is focused

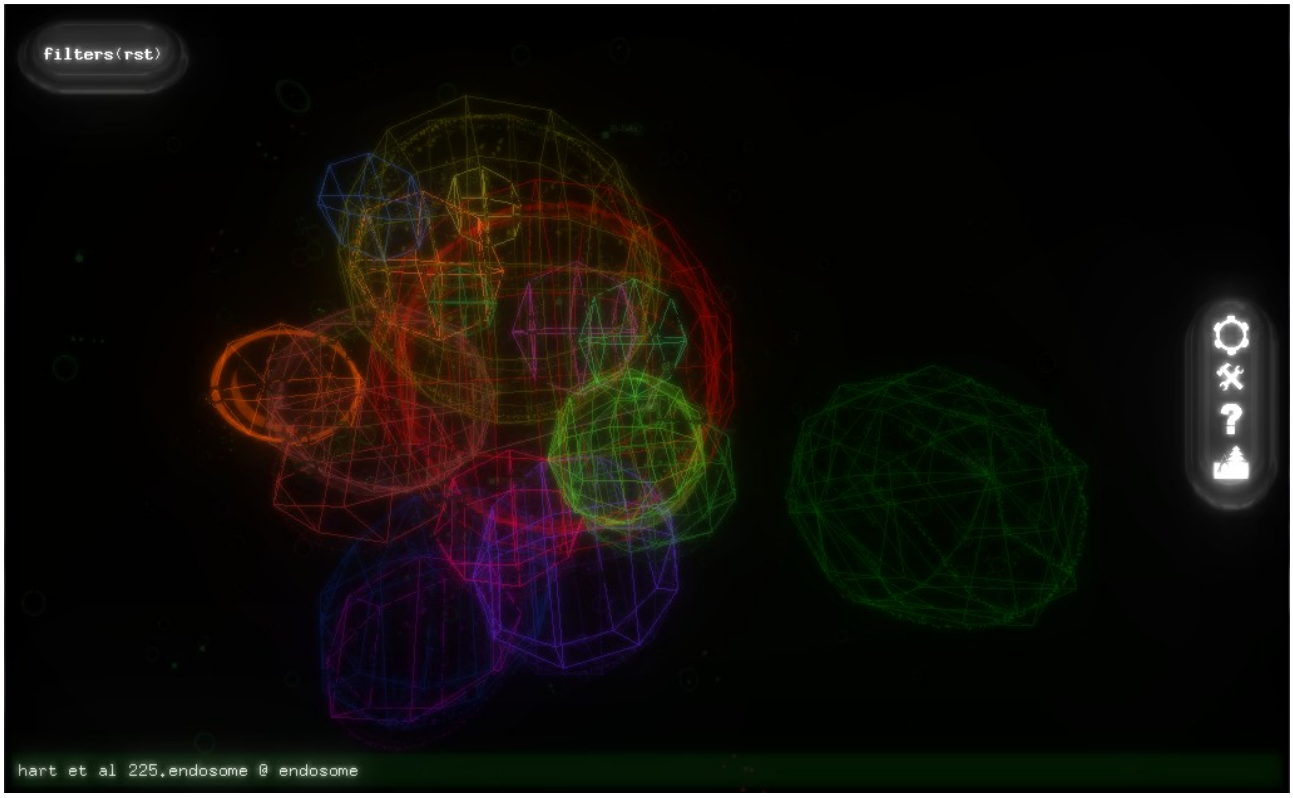


Figure 74: The Interactorium: the different cell components are visualized as geometric spheres

Figure 75: The Interactorium: inside the nucleus

When starting the application, all components are steadily moving, caused by the spring-embedding layout. The layouting process can be stopped manually, but there seems to be no opportunity to reproduce the layout. In addition, the colors of a selected node and its interaction edges are changing. This glowing effect is visually very nice, but impedes the navigation and analysis of the network. Figure 75 shows another problem: in many cases it is visually difficult to distinguish different cell components or to recognize if the user navigates inside or outside the nucleus.

The Interactorium is not a modeling framework. It is a framework intended to provide fast proteomic information visualization on the subcellular level. As mentioned before, it is the only project which is quite close to the integrative level, because it is also able to integrate molecular data: the PDB structures. But of course the cell is only symbolized by simple spheres and it is not possible to integrate microscopic data, therefore it also does not completely apply to the integrative level.

4.3.3.5 Integration and Visualization of Multimodal Biological Data

The work introduced in this section was developed as a plugin for the Java-based VANTED system which was also developed at the IPK Gatersleben (Section 4.3.1.2): *HIVE (Handy Integration and Visualization of multimodal Experimental Data)* [RoKS09, RoKS11].

This tool extends VANTED to handle and visualize experimentally derived data from different domains. Scientists with different biological and technical backgrounds can use this application to visualize and compare their data. HIVE visualizes image stacks or data which has been previously segmented with tools like Amira (Section 3.1.3.1). Therefore, HIVE is the only tool in this section able to work with microscopic data. But microscopic data can be volumetrically rendered in full three dimensions using Java 3D.

Figure 76 gives an overview of the different domains HIVE is capable of. The crop plant Barley is used as an example. The metadata is visualized similar to the modules known from Amira (Section 3.1.3.1 and Figure 76.1/2). Experimentally-derived measurements are integrated into the actual projects and also correlated with the three-dimensional visualization. Figure 76.3a shows a gradient associated with the view and Figure 76.3d shows the volumetric representation color-coding the different measurement types.

A huge advantage of this approach is the direct visualization of image stacks using HIVE. But the volumetric rendering might often be a problem on desktop computers due to its high hardware requirements.

Obviously also HIVE is quite close to the integrative level. But there are two aspects – following the definition of this work – which are missing to meet the requirements of this category: the visual integration of molecular structures as well as the separation of the network structures and the cell model.

4.3.3.6 3DScape

As HIVE is a plugin for VANTED, *3DScape* is a plugin for the Cytoscape (Section 4.3.1.1). But the application cases are quite different *[Wang11]*. In this case, the C++/OpenGL-based VTK package was used for the three-dimensional visualization *[ScML04]*.

As most approaches discussed above, a force-directed layout is applied to visualize networks *[FrRe91]*. In addition, a Fast Multipole Multilevel Method (FM3) *[HaJü05]* and a Span Tree algorithm are available.

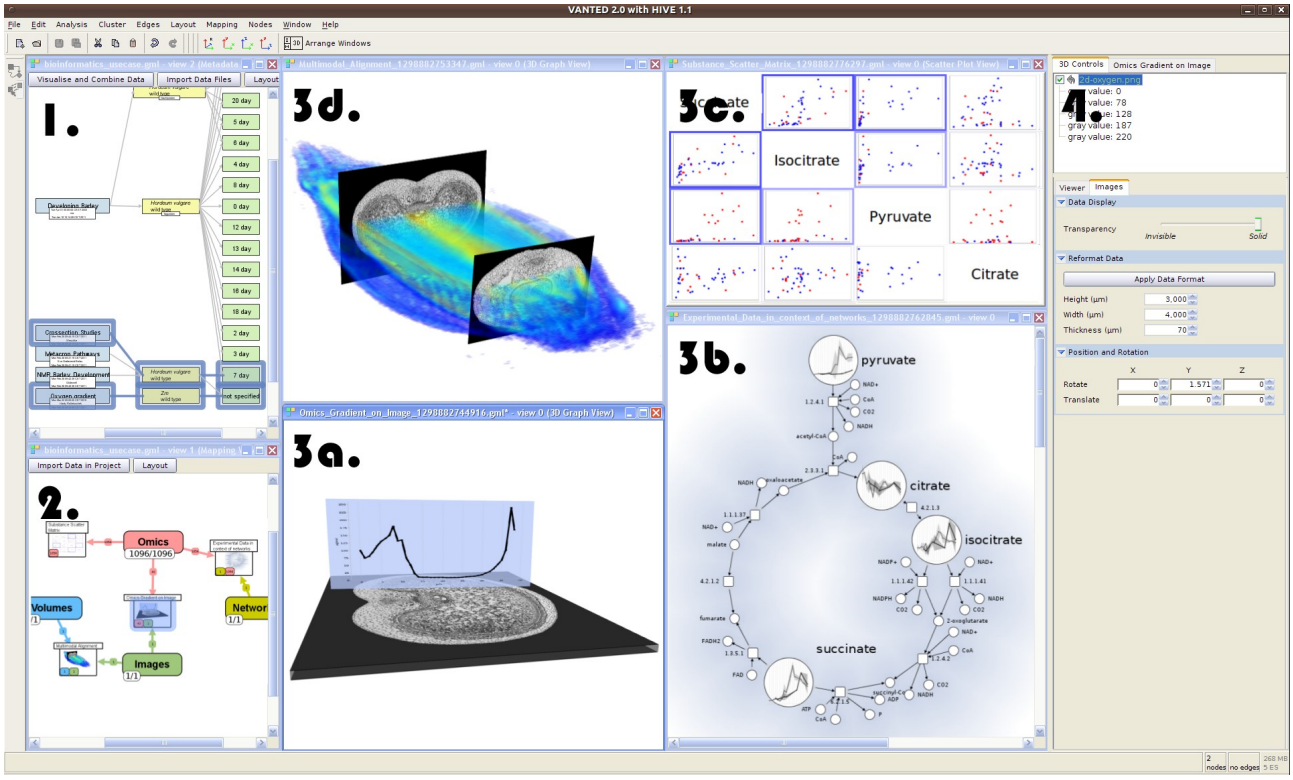


Figure 76: HIVE: the VANTED plugin
It enables the integration of multimodal data: 1. the metadata graph; 2. measurement level data; 3. different integration views: a. gradient on two-dimensional image, b. omics network context, c. scatter plots, d. multimodal alignment with volumetric visualization; 4. controls for the actually selected view (Courtesy of/Copyright © 2011 by Hendrik Rohn. Reprinted with permission from [RoKS11])

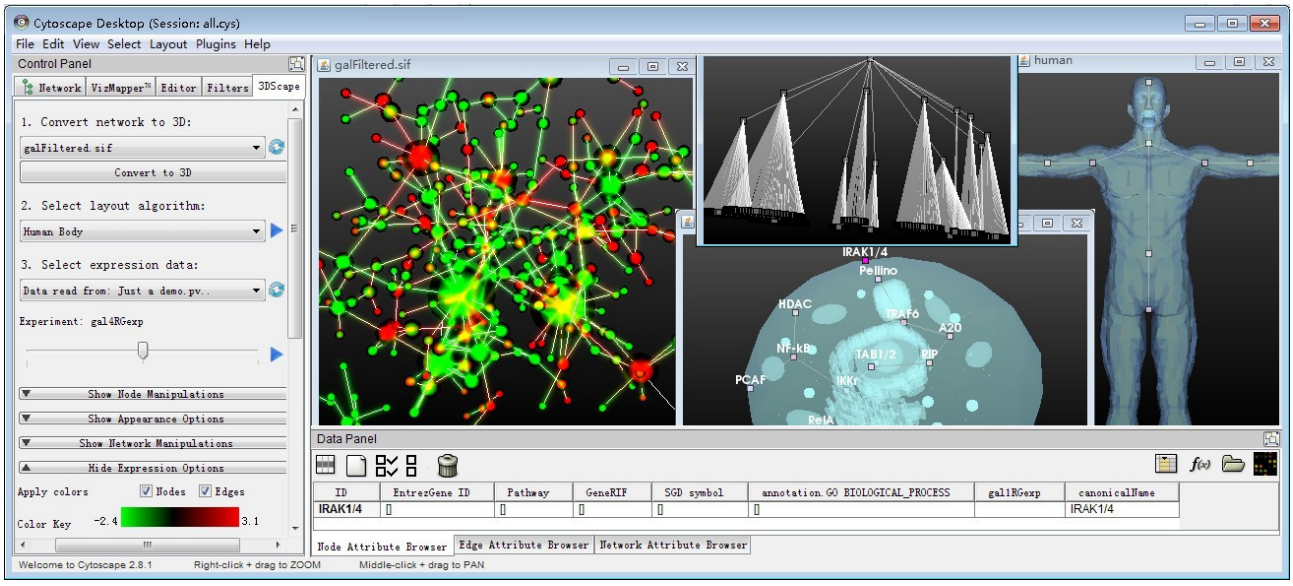


Figure 77: 3DScape: The Cytoscape plugin showing different Application Cases (Courtesy of/Copyright © 2011 by Qi Wand/Nature Preceedings [Wang11])

By using the *VistaClara* plugin for Cytoscape [KiKC08], it is also possible to work with a heat map based on differences in expression densities. In 3DScape, colored nodes are combined with a gradient of colored edges of heat map colors (left image of Figure 77). Different node properties like the color, size and shape of the two-dimensional Cytoscape layout are maintained.

The central image of Figure 77 shows a cell model which can be associated with a network. By using VTK, 3ds and Wavefront files can be imported into 3DScape. The import of localization information into Cytoscape is supported by the additional Cerebral plugin [BGHM07].

During the practical work with 3DScape and Cerebral, the following sentence from the original pre-publication manifested as only partly correct:

"There are plant/animal cell and human body models included in 3DScape [...], networks can be easily mapped onto these models by providing corresponding localizations [...]." [Wang11]

In contrast to this, in the blog of Ken Wang – the developer of 3DScape – wrote:

"Using force directed placement and ´animal cell´ model, I was able to map the example network from Cerebral plugin to cellular components (though not corresponding locations). I am lazy so I would not manually curate the coordinates of each node. In future release, I plan to re-implement the Cerebral layout algorithm making it match to 3D space." [Wang11]

Therefore, as of yet (31.07.2012) it seems not possible to automatically map the nodes of the network onto the corresponding localization of cell components (see Figure 78 showing an approach to map nodes onto cell components with 3DScape). The nodes have to be manually placed onto the corresponding positions by using the *Node Manipulations* dialog. Of course, the stated manuscript is not a full publication, because it was issued in *Nature Precedings*, which has been discontinued[12h].

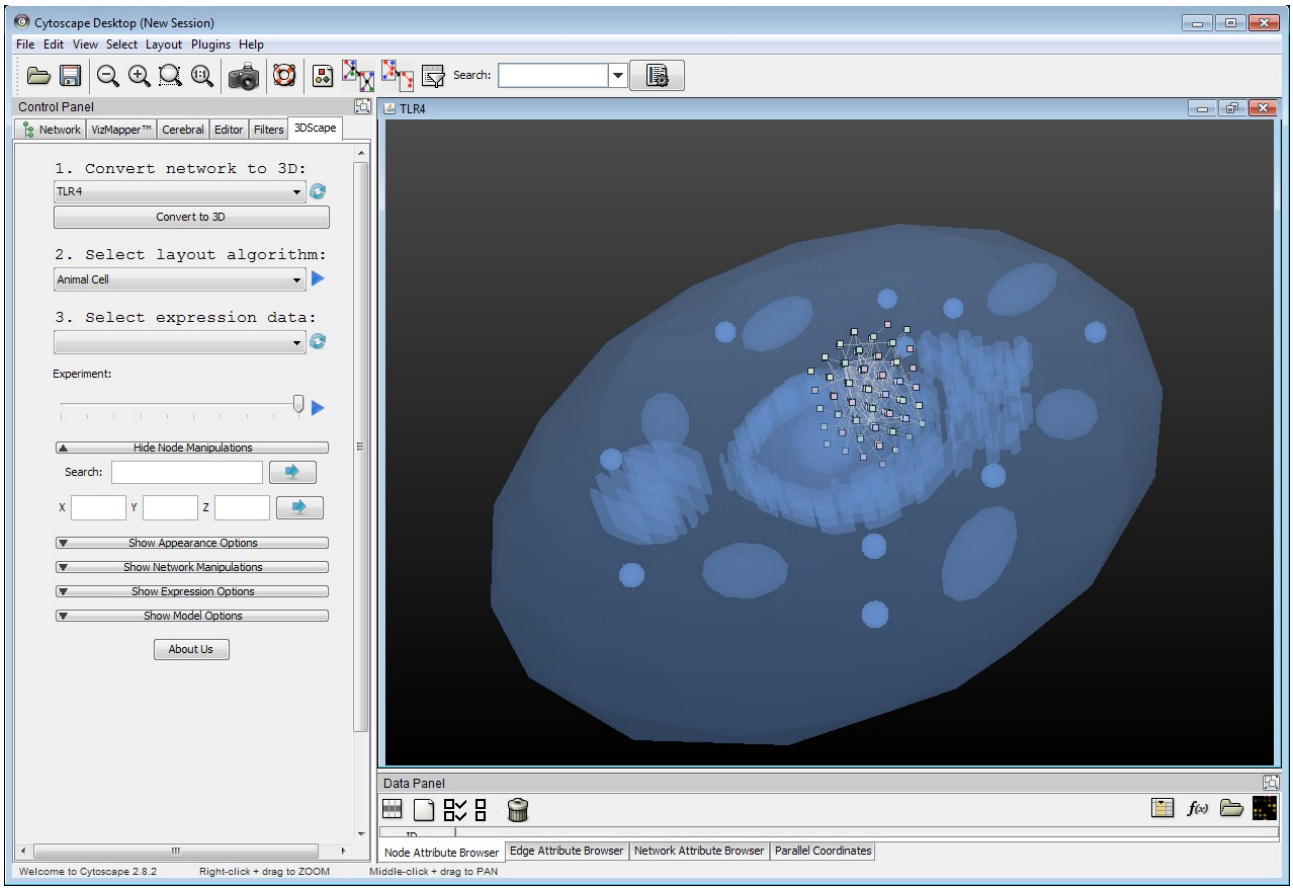


Figure 78: 3DScape: The Cell Component Mapping does not work automatically

The source code of this program is not available and the installation process needs at least four different packages to be copied manually into the correct system locations as described on the corresponding webpage [11p].

4.3.4 Summary

Obviously the most related approaches were found in the area of 3D network modeling. Table 9 shows an overview of all projects. The 2D visualization tools are not listed in this table, the reason was already mentioned (Section 4.3.1); implicitly Cytoscape is represented by 3DScape and VANTED by HIVE.

The two-and-a-half-dimensional network visualization is presented by WilmaScope, Arena3D and BioCichlid. Arena3D seems to be the most vital project, because it was recently extended by time-data-related aspects. Although the authors do not mention WilmaScope, it can be seen as a consequent further development of this approach first published in 2004. Especially the support of multiple layers is an interesting aspect. It will be shown that the application developed in this work does not directly support a 2.5D visualization of networks, but uses another approach to combine 2D and 3D visualization.

Similar to this, the HIVE approach can be seen. Although it does not correlate cell components with 3D networks, it connects the volumetric 3D visualization of microscopic data sets with the 2D network visualization of VANTED. This can also be understood as a contribution to the focus+context paradigm which was already used by the earliest 3D network visualization approach introduced in this work: the Metabolic Network Visualizer.

There are only three projects which integrated sub-compartmental placing of nodes: MetNetVR, The Interactorium and 3DScape. The latter mentioned does not support automatic node placement – therefore the user has to place nodes manually onto the shapes of the cell components. In contrast with this tool, The Interactorium and MetNetVR use only very abstract representations of cell component. These are very simple geometrical forms. In MetNetVR the cell components can only be recognized by their labels whereas The Interactorium uses different geometric forms which are hard to differentiate. GEOMI uses an alternative method: the nodes are colored according their localizations.

However, none of the approaches discussed here are able to differentiate between different cell component layers or to integrate different cell component models and localizations interactively. In addition, all these approaches do not contain modeling features to change the composition of the cell. Of course, this topic is more related to the Visual Cell Modeling section (Section 4.1.1).

In addition, none of these tools provide (semi-)automatic localization capabilities for proteins. The localization information has to be acquired manually.

The requirements for the creation of a Virtual Cell correlated with a localized protein-associated network resulting from these observations is discussed in Section 5.3.1.

Category	Feature	3DScape+Cytoscape	Arena 3D	BioCichlid	GEOMI	The Interactorium,	MetNetVR	Metab. Netw. Visualizer	VANTED HIVE+	WilmaScope
Availability	Plugin	X							X	
	Standalone			X	X	X	X[49]			X
	Web Service									
	Website Applet							(X)[50]		
	Web Start		X						X	
	Source Code		X	X					X	X
	Licenses	LGPL	Acad. Free	Acad. Free	LGPL	Acad. Free			GPL2	LGPL
Dimensions	2D	X			X				X	
	2.5D		X	X					X	X
	3D	(X)			X	X	X	X	(X)	(X)
	Time (incl. 4D)		X	X						
Formats	ANDVisio									
	CSV	X							X	
	native format	Cytoscape formats	X		XWG				X	
	PDB					X				
	SBML	X	X						X	
	VRML							X		
Granularity	Atomistic									
	Molecular					X				
	Intra-Compartmental									
	Subcellular	X			X	X	X	X	X	
	Organisms								X	
Interactivity	3D Navigation	X			X	X	X		X	
	Detail-on-Demand					X	X		X	
	Dynamic Tables									
	Links to data sources								X	

49 The software MetNetVR is not downloadable, but parts of the website are still on-line (as of 31.07.2012).
50 The Metabolic Network Visualizer as well as its website is no longer available (as of 31.07.2012).

Category	Feature	3DScape+Cytoscape	Arena 3D	BioCichlid	GEOMI	The Interactorium,	MetNetVR	Metab. Netw. Visualizer	VANTED HIVE+	WilmaScope
	Focus + Context	X						X	X	
	Sorting Tables									
Layouts	Clustering	X								X
	Fan						X			
	FM3	X								
	Force Directed	X	X		X	X		X	X	X
	GEM-3D						X			
	ISOM									
	Radial						X		X	
	Span Tree	X								
Locali-zations	different localizations for one molecule				X	X				
	datawarehouse results									
	Experimental data localization	X							X	
	text mining results									
	manual prediction									
	re-localization									
Modeling	cell component abstraction level	2				3	3		1	
	abstract models	X								
	microscopic models								X	
	different pathways at once		X	X						
	node distribution algorithms		X	X			X	X		X
Pathways	Metabolism	X	X					X	X	X
	Protein-Protein Interaction	X	X	X	X	X			X	X
	Signaling	X	X						X	X

Table 9: A comparison of all network modeling tools discussed in this work
The complete table is found in Appendix 9.19.3

4.4 [MES+MOL+FUN] The missing Link

Summarizing examinations of the different related approaches discussed here shows that a tool is missing which unites all these different levels into one single project: the mesoscopic, the molecular and the functional level.

Figure shows a first sketch of the application to be developed throughout this work. It was created when the topic of this thesis was introduced for the first time years ago. It was a visual rudimentary – not functional – combination of CELLmicrocosmos DisplayCell with a three-dimensional pathway exported from a stand-alone network modeling tool created in the diploma theses of Claas Faber and Ralf Brand at Bielefeld University in 2005 [Bran05, Fabe05]. As discussed in Section 3.1.3.1 there were various important reasons to cancel the development with Amira.

The following methods and implementation subchapters will now discuss the three sub-levels required for the creation of the Integrative Modeling approach which will be finally introduced in Subchapter 5.4.

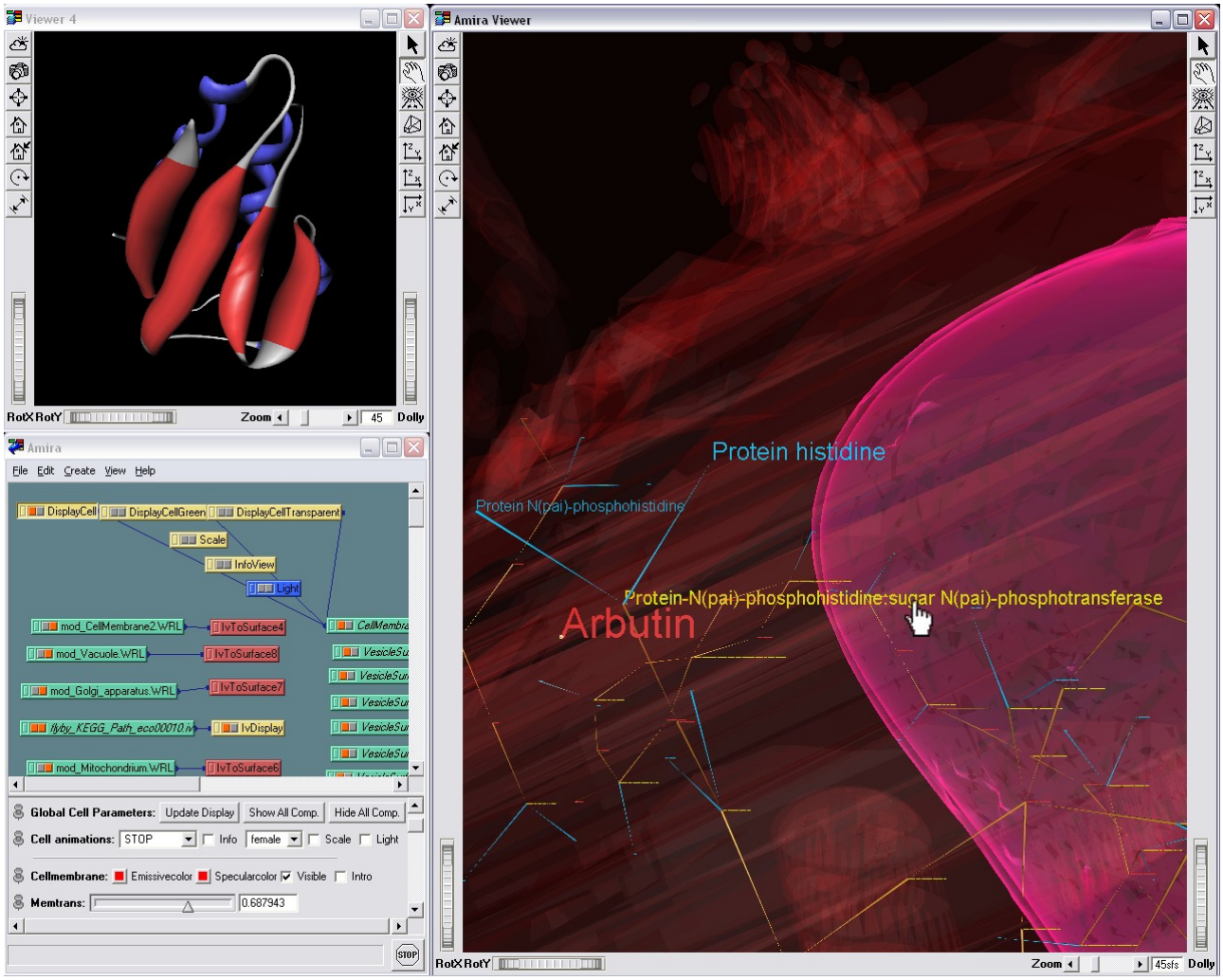


Figure 79: A first sketch of the functionality needed for a Virtual Cell application
It was based on the first master thesis of the CELLmicrocosmos project based on Amira, combined with a network generated by a tool developed during two diploma theses [Bran05, Fabe05, Somm06]

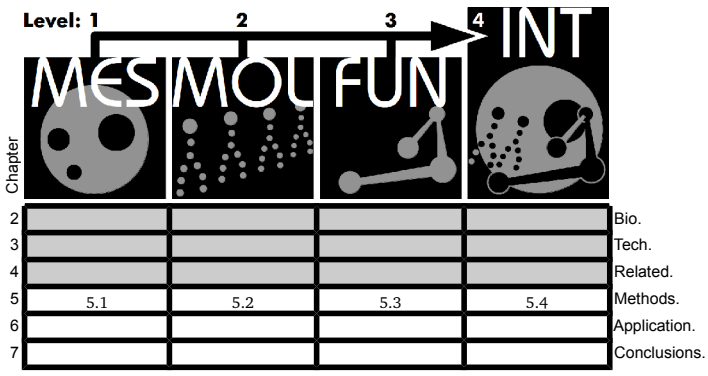
5 Methods and Implementation

Remembering the first visualization techniques applied to cell visualization by Robert Hooke in 1665 – as discussed in Section 2.1.1 – at that period in time there were no photo cameras which could be used to make an image of the cell. What Hooke saw through the microscope he had to record by drawing with a pencil. Of course, today there are many microscopy and spectroscopy techniques available and there is usually no need to take the pencil in the hand (Subchapter 3.1). But as was shown before, there are many aspects of the cell which cannot be directly visualized using these techniques, caused by the limited resolutions or by the finite capabilities to record molecular movement over time. Therefore, even today a substitute for Hooke's pencil is needed – an electronic substitute: the computer. This chapter is dedicated to exactly this topic.

The basics for an Integrative Cell Modeling environment have been introduced and the related approaches have been discussed. It was already mentioned in the introductory chapter that in the recent years three different tools have been developed. The following subchapters will be subdivided into three sections:

1. Requirements, giving a short description of the aspects the related project should provide,
2. Methods, introducing the different procedures and further developments of the theoretic aspects partly inaugurated in the Technical Background (Chapter 3), and finally the
3. Implementation Details, shortly summarizing the way the methods have been realized in form of a program structure; there will also be a list of external libraries which were needed for different projects.

Now the methods and implementation for each of these approaches will be discussed: Mesoscopic Modeling (Subchapter 5.1), Molecular Modeling (Subchapter 5.2), and Functional Modeling (Subchapter 5.3). Also this chapter will conclude with Integrative Modeling (Subchapter 5.4), showing in which way the preceding three projects can be united.



5.1 [MES] Mesoscopic Modeling: Constructing a Virtual Cell Environment

In terms of the content of the preceding subchapters, many different application cases for a Virtual Cell environment were discussed. In this chapter it will be concretely defined which ideas and requirements have to be implemented in the context of a cell modeling environment. And of course, many of these aspects can be generalized and could possibly be taken into account during further development of the approaches developed in other working groups.

For this purpose, Subchapter 5.1 is completely devoted to the application called CELLmicrocosmos 3 CellEditor (Cm3/CmCE).

5.1.1 Requirements

R1.I Realistic Cell Component Models (Abstraction Level 1)

Initially it is important to enable the incorporation of cell component models directly derived from microscopic data sets.

R1.II Interpretative Cell Component Models (Abstraction Level 2)

As mentioned already in the introductory chapter, the main intention of CELLmicrocosmos is the break-down of complex cell-biological information to a model providing an appropriate overview. For this purpose, also the complex structure of cell components, consisting of millions of molecules, have to be simplified. Different abstraction levels should be provided. The second abstraction level is – not directly derived from, but – based on microscopic data sets. This means, the cell components should be recognizable, but they do not have to copy the complete structure of a cell component shape.

R1.III Abstract Cell Component Models (Abstraction Level 3)

The third abstraction level uses simple symbols and/or objects as substitutes for cell components. The visual differentiation of these models might be a problem.

R1.IV Multiple Cell Component Types

A large variety of cell component types exists (Section 2.1.3). The application should provide a subset of cell components to be useable for regular application cases.

R1.V Cell Component Layers

As discussed before (Section 2.1.3), cell components contain different layers like membranes, inner cores or other entities. This fact should be taken into account when modeling cell components.

R1.VI Multiple Cell Compositions

Different cell models have to be created by the user. For this purpose, adequate features must be implemented. In addition, an editing of the cell model would be an important extension (Section 2.1.2).

R1.VII Cell Component Distribution

The exact position of cell components is not predefined in an abstract cell environment. But there are some basic rules which can be taken into account for their distribution. The application should provide different methods to place cell components into the cell.

R1.VIII Color Schemes

The color of the microscopic view of a cell is usually characterized by the staining method used (Section 3.1.1.1). Therefore many different color themes are reasonable. It should be possible to use cell components with different color schemes to meet the perceptions of the user.

R1.IX Import Capabilities

The three-dimensional modeling of 3D shapes – like cell components – needs special modeling software like those discussed in Section 3.1.3. The generated files can be exported from the external program and have to be imported into the application to be developed.

R1.X Export Capabilities

After generating a cell assembly, those structures have to be saved. In this way they can be used for further tasks like those discussed in Subchapter 5.3.

5.1.2 Methods

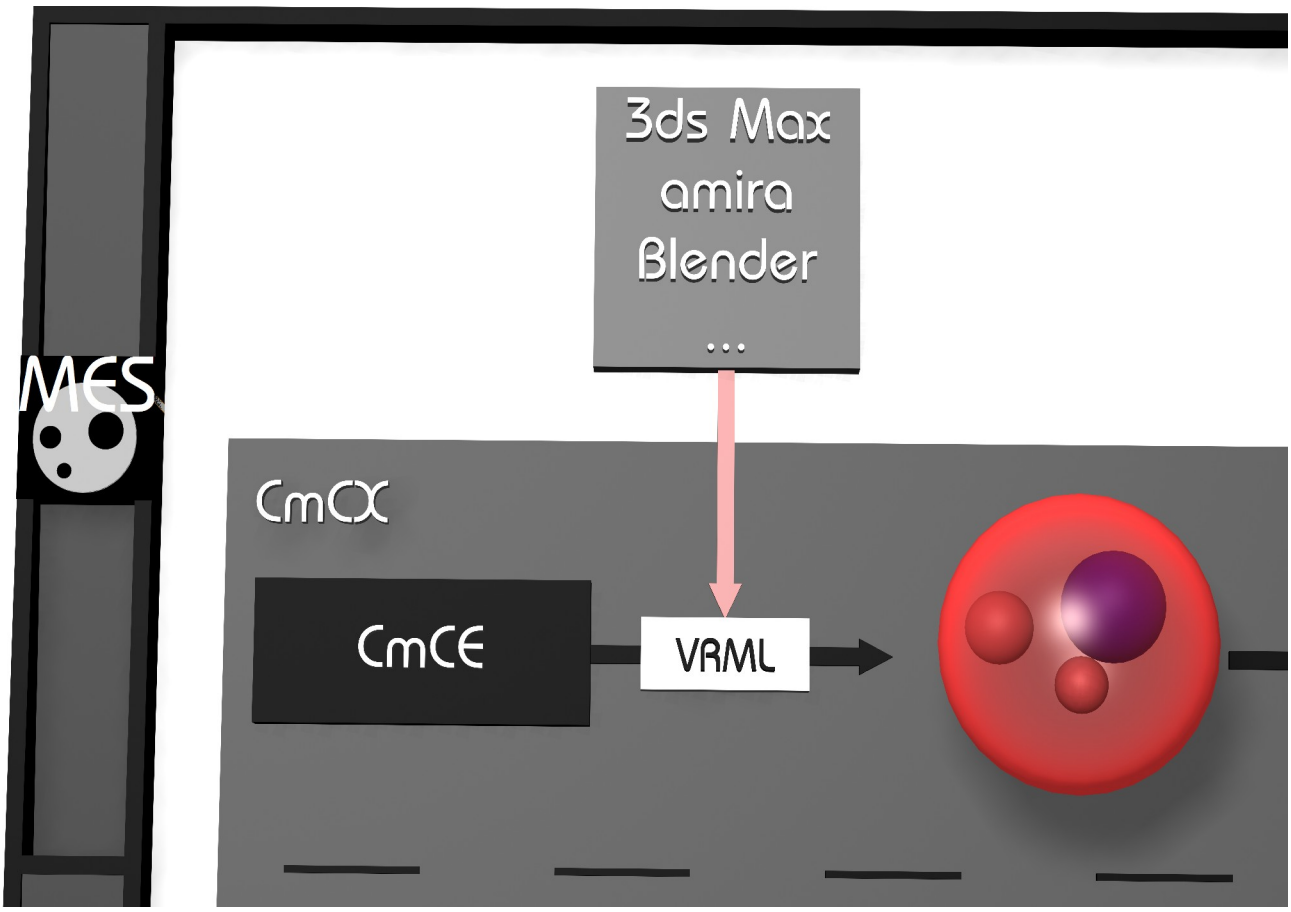


Figure 80: The CELLmicrocosmos modeling pipeline at the mesoscopic level

How should the requirements be implemented? Which methods are possible? This will be discussed in the following section. Figure 80 shows the modeling pipeline associated with this subchapter, a detail of the whole pipeline was shown in Figure 7.

5.1.2.1 Cell Component Modeling and Abstraction Levels

Prior to discussing the different methods to model cell components visually in three dimensions, some initial thoughts about the granularity of cell modeling should be formulated.

The problem of cell modeling arises from the different types of image generation techniques. In contrast with the modeling of a toy car, where the original model as well as all technicalities are completely known (at least if the original blue print is available), today's cytology is far from completely deciphering the whole structure of the cell. Therefore, the modeler is restricted to available information. In Subchapter 3.1 many ways to get those information fragments are described. The question arises now, how this data should be dealt with in an appropriate manner.

While modeling is always an abstraction, the level of abstraction should be clarified first. Roughly, four abstraction levels can be defined (Figure 81):

0. ABS_LEV_0: this is an ideal value: an abstraction level 0 would be the perfect copy of a cell component, completely reflecting its natural shape (also: Clone),
1. ABS_LEV_1: 3D-microscopy- or spectroscopy-based cell component models which have to be imported from other applications like Amira and improved by 3ds Max (also: Image),
2. ABS_LEV_2: interpretative cell component models based on different two- or three-dimensional microscopic images but designed by mouse-based and script-based modeling techniques (similar approach is known from 4.3.3.6, also: Allegory) and
3. ABS_LEV_3: abstract cell component models, for example: spheres (also: Abstraction).

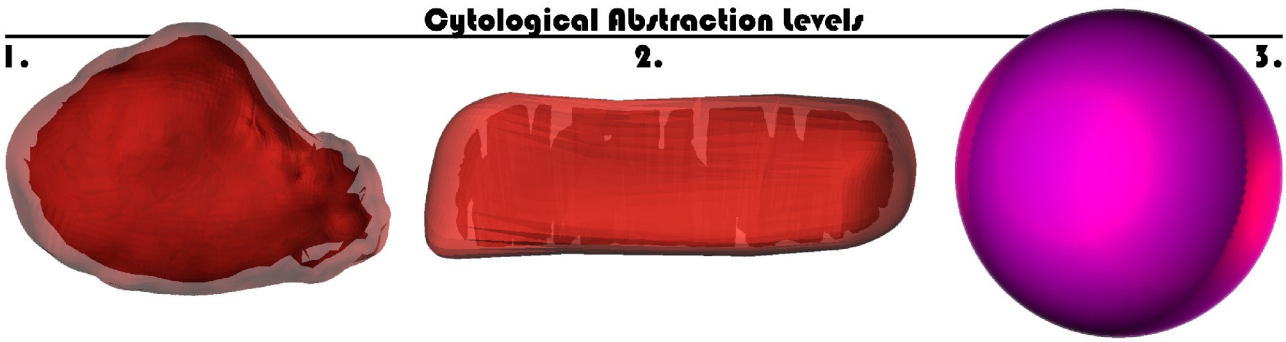


Figure 81: Different abstraction levels of cell components
1. 3D-microscopy-based cell component (ABS_LEV_1), 2. interpretative cell component (ABS_LEV_2), 3. sphere cell component (ABS_LEV_3)

As mentioned before, the abstraction level zero is an ideal instance. Perhaps in the future it will be possible to use a special technique to scan the complete molecular structure of a cell. In that case it might be possible to reconstruct the whole cell component based on its original coordinates. This would be a revolution for the modeling of cells. But as it was already discussed: today there is only fragmentary knowledge of the cellular structure. Therefore it is even more important to think about the way to represent cells. How far does a visual representation differ from the realistic structure of a cell?

Therefore, the following three sections will discuss the realization of the abstraction levels 1 to 3.

5.1.2.1.1 3D-microscopy-based Modeling (Abstraction Level 1)

Meets Requirement R1.I

The first abstraction level (ABS_LEV_1) applies to models which were directly derived from microscopic data. Figure 82.1 shows a segment, a cut through a mitochondrion, while Figure 82.2 shows a mitochondrion with a closed surface (Section 2.1.3.10). Both models are based on electron tomography (Section 3.1.1.2). As pointed out by the microscopy-related sections, these structures still do not reflect reality, but they are close to what the researcher sees through a microscope.

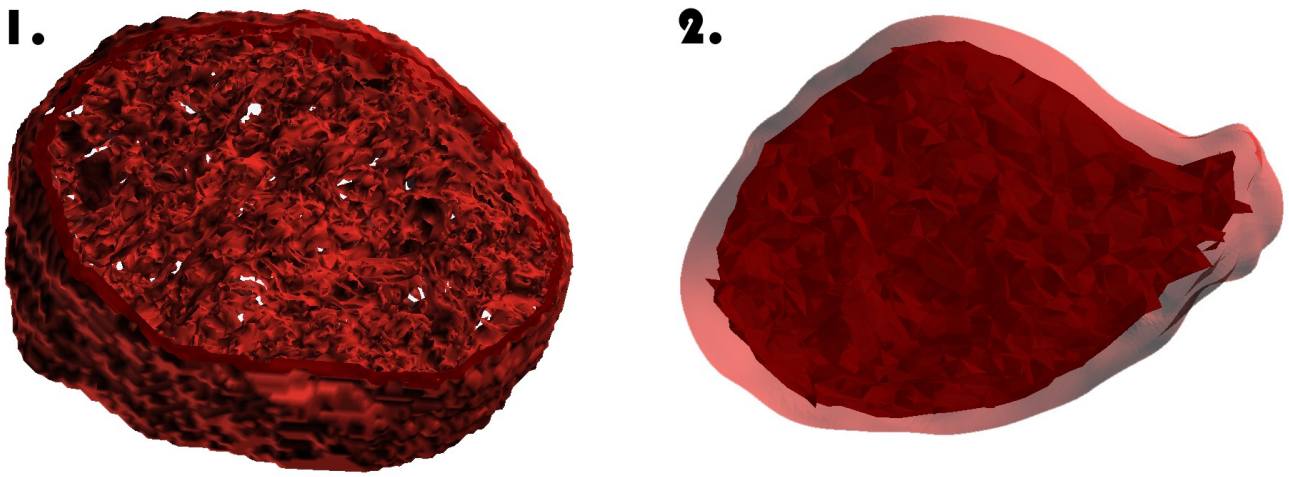


Figure 82: Two electron-tomography-based mitochondrion models in CellEditor
1. a mitochondrial slice model generated by image segmentation (see Figures 83 and 84); 2. a mitochondrion model with a closed shape generated by image analysis algorithms and manual surface corrections (see Figure 85)

The most important tool for microscopy-based modeling in context of this work is Amira (Section 3.1.3.1) and the most important data source, respectively, the CCDB (Section 3.1.2.1).

Figure 83 shows a cut through a mitochondrion from the CCDB containing 82 of 124 image slices along the vertical axis (CCDB-3864) [YLPS08]. The other images from the original dataset at the top and the bottom of the image stack were omitted because they were too blurry. The input format for Amira in this case as well as the next model to be discussed was a *hdr/img-file*. It contains a whole stack of aligned images. In this case the image stack was semi-manually segmented by using the *Image Segmentation Editor* of Amira. The screenshot in Figure 84 shows the image stack from three different perspectives. Especially if a closed shape is needed – like the outer membrane of a mitochondrion – the tracking of a complex shape by an algorithm will often fail. The reason is that a mitochondrion is surrounded by two membranes. Analyzing the images, the two lines of both membranes sometimes converge (see Figure 84, top-center image). On the left top of Figure 84 the resulting three-dimensional shape is shown which has to be exported for external use. For this purpose, Amira provides a VRML97 export module (Section 3.1.3.4).

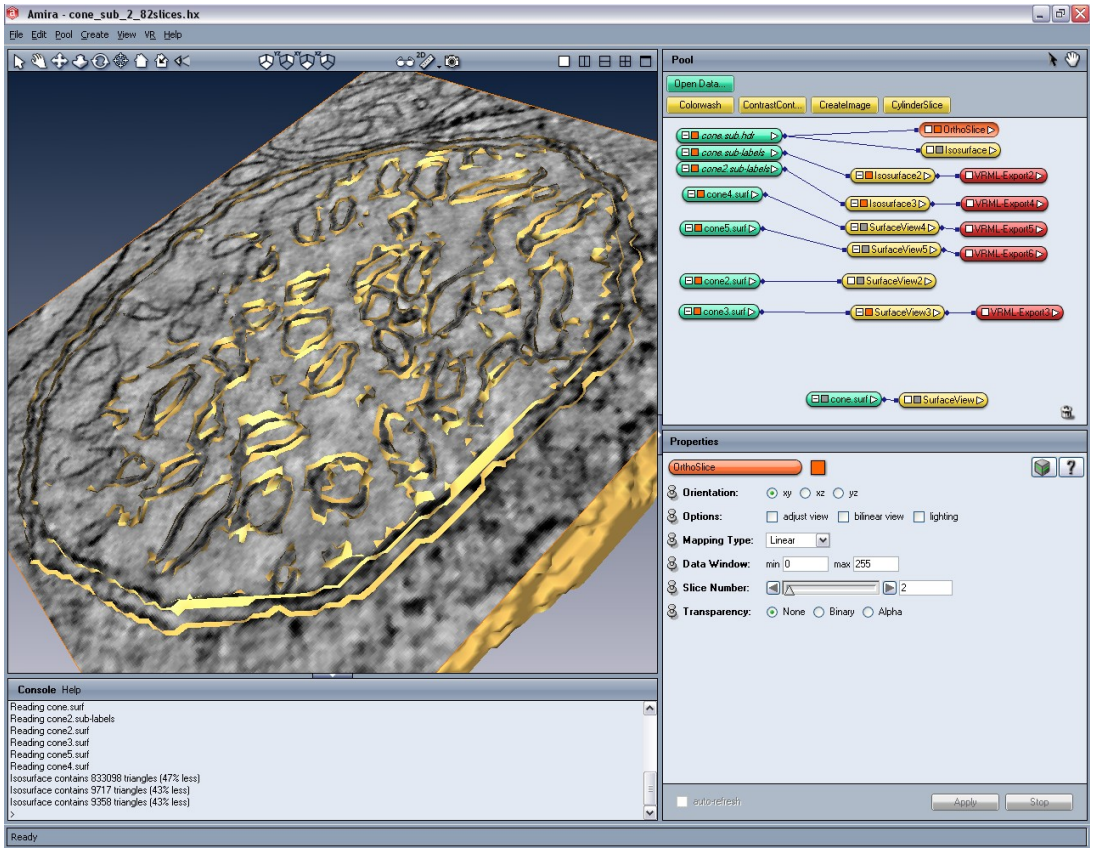


Figure 83: Amira: a microscopic image of the mitochondrion and the three-dimensional shape Segmented by using the Amira segmentation tool (CCDB-ID: 54)

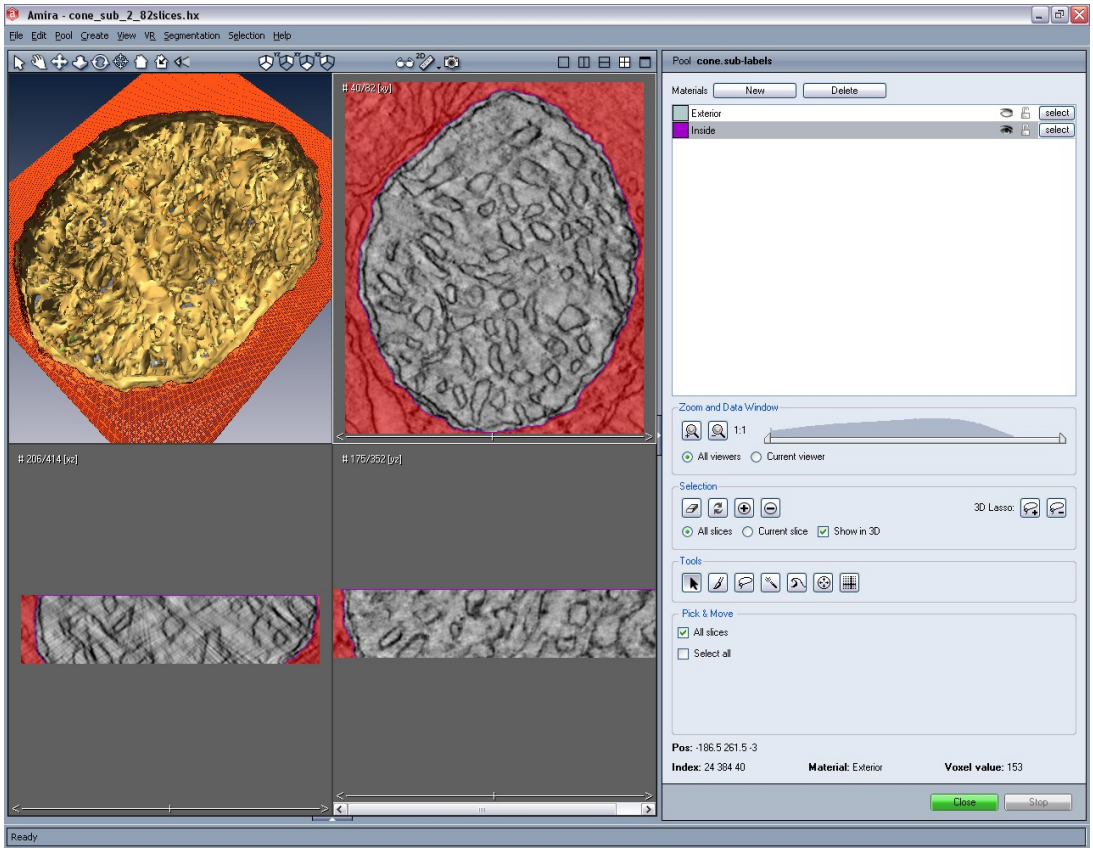


Figure 84: Amira: the Image Segmentation Editor
It is used to create three-dimensional shapes by segmenting each image similar to image manipulation in software like Photoshop® (CCDB-ID: 54)

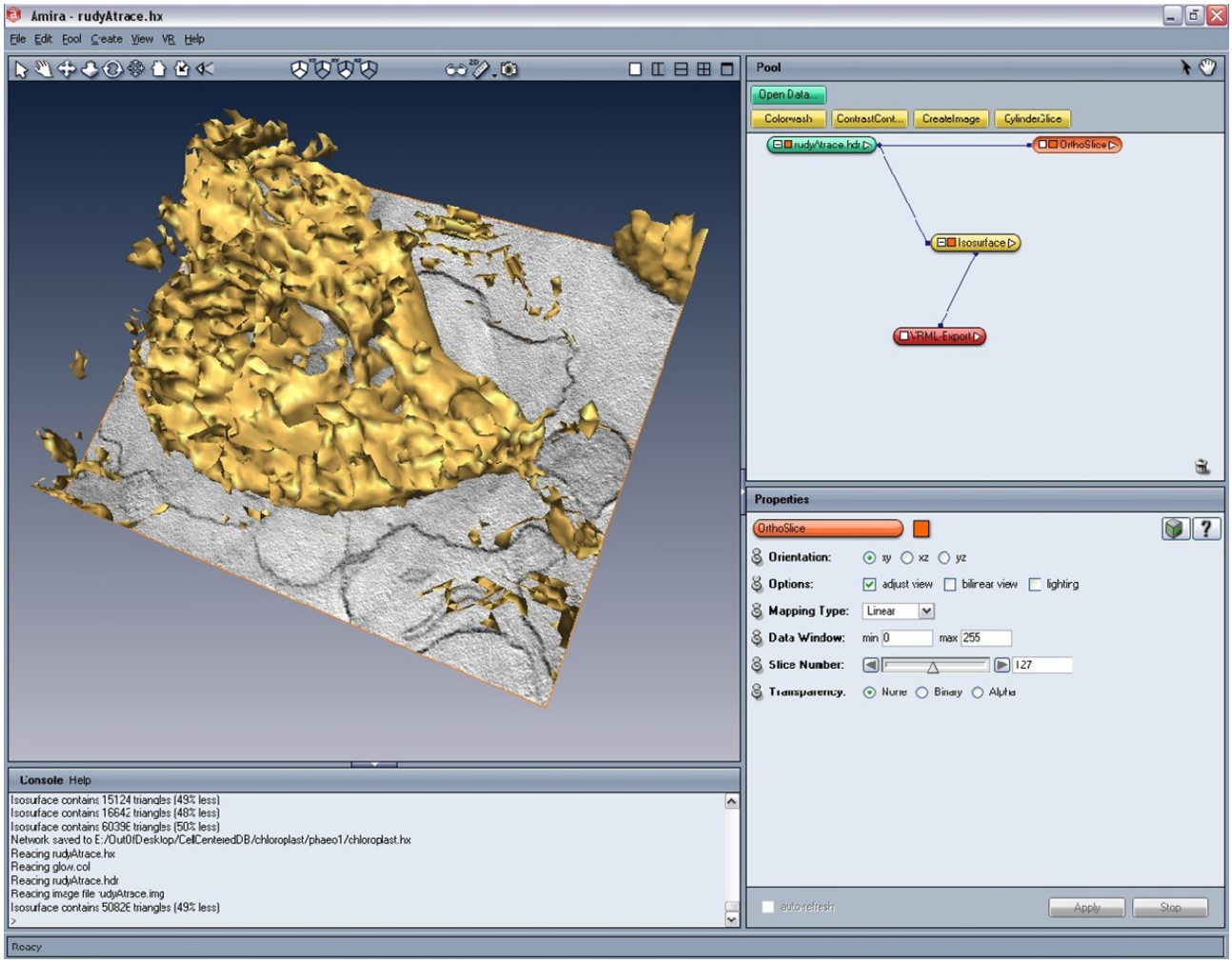


Figure 85: Amira: The surface of a mitochondrial data set derived from the CCDB
It is computed by a special algorithm part of Amira (CCDB-ID: 3864)

Figure 85 shows the surface generation process of the other mitochondrion model downloaded from the CCDB (CCDB-3864) [YLPS08].[51] 271 microscopic slides derived by electron tomography (Section 3.1.1.2) contain the front segment of the mitochondrion. By using the *compactify* and *downsample algorithm* of *Amira´s Isosurface* plugin, the surface of the model was computed. The algorithm generates polygon shapes based on the defined range of pixel values which depends on the threshold value. As can be seen in the image, this approach generates many artifacts as well as holes inside the membrane. These problems could be fixed by using the modeling capabilities of Amira. But those are very rudimentary and it would be quite time-consuming to close the surface manually. Therefore, 3ds Max was used for this purpose, as shown in Figure 86 (Section 3.1.3.2).

In addition, another workaround was done using 3ds Max. As mentioned before, the image stack contained only the upper fragment of a mitochondrion. Therefore, this part was duplicated and mirrored to create a closed model. This is a concession to the fact that a high resolution image stack of a complete mitochondrion was not available during the modeling process.

51 This data was previously shown on the CCDB webpage in Figure 22.

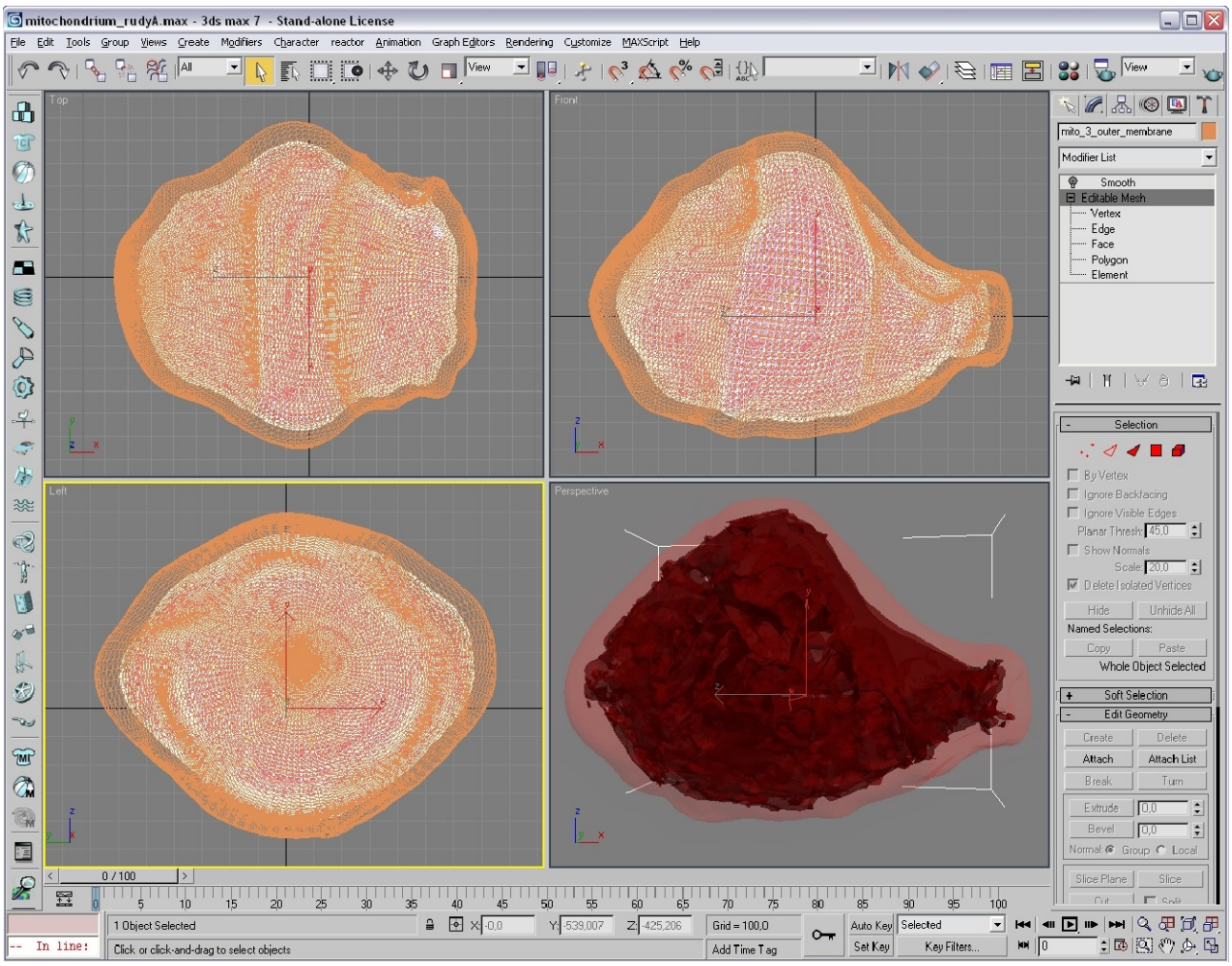


Figure 86: 3ds Max: the mitochondrion model imported from Amira (Figure 85) It is manipulated with native modeling functions of 3ds Max, resulting in a coherent model

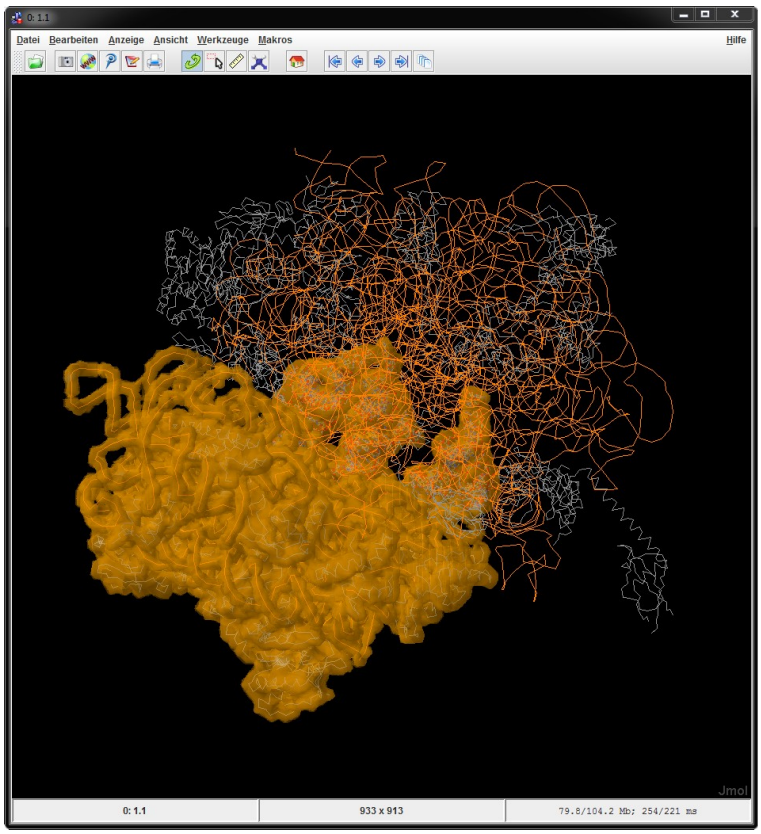


Figure 87: Jmol: the Van-Der-Waals surface of the ribosome (as seen at the bottom) is generated from the atomic structure (as seen at the top)

5.1.2.1.2 Molecular-structure-based Cell Component Modeling (Abstraction Level 1)

Meets Requirement R1.I

Usually there are no atomic structures for cell components available. An exception is the eukaryotic ribosome (Section 2.1.3.7), the largest asymmetric biological molecule structure currently known. The determination of its atomic structure was honored with the Nobel Prize in 2009. A

model of a ribosome can be generated by combining the PDB structures 1GIX and 1GIY. Figure 87 shows how Jmol is used to compute the Van-der-Waals-surface based on the atomic structure. This surface can be used to generate a shape-based model.

5.1.2.1.3 Interpretative Cell Component Modeling (Abstraction Level 2)

Meets Requirement R1.II

In the second abstraction level (ABS_LEV_2) the deviation from reality will naturally be even larger than in the first level. In this case, exclusively 3ds Max or Blender were used for the modeling process. The base for cell component models providing an appropriate quality is a stack of good microscopic images, an optional analysis of third-party visualization approaches in combination with experiences with the used modeling tool. Usually, for this purpose different two-dimensional images will be taken into account, featuring different microscopy techniques. Giving an example, the Figure 88.1 shows the original image, a freeze fracture (Section 3.1.1.2) image of a liver cell, used as a base for the form of the nucleus model shown in Figure 88.2. The reason for the blue coloring will be discussed in Section 5.1.2.3.

Obviously, similar modeling approaches were used for the cell components used in Meta! Blast (Section 4.1.1.2) and 3DScape (Section 4.3.3.6).

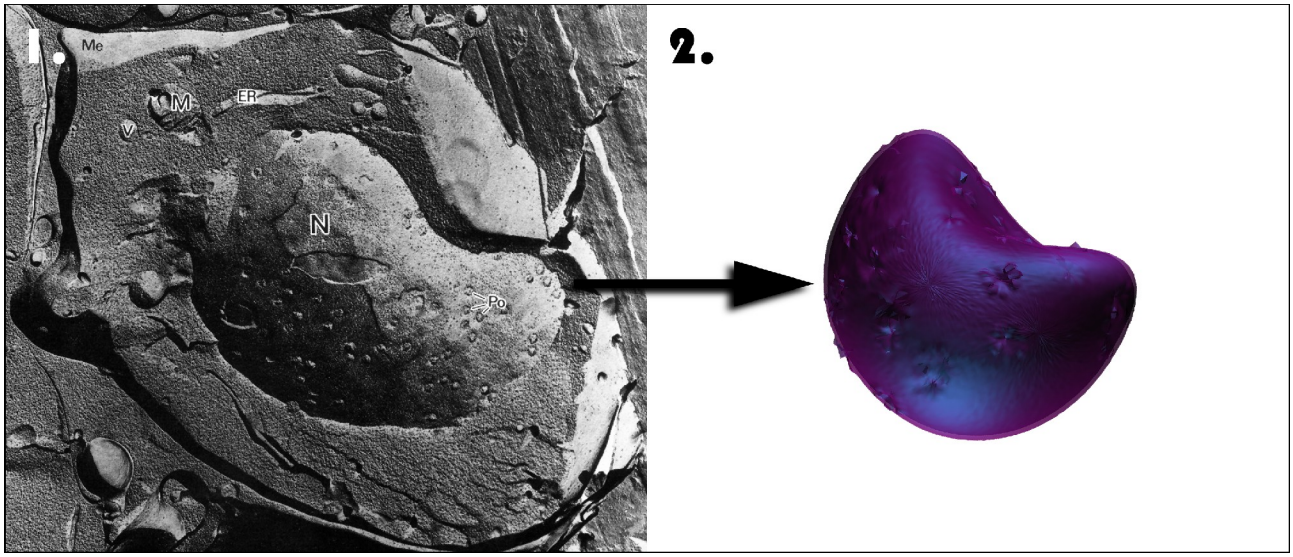


Figure 88: Interpretative Cell Component Modeling: Nucleus
1. freeze fracture photograph of a nucleus (Courtesy of/Copyright © 1982 by Helmut Meyer. Reprinted with permission from [UdKo82, p.27]; 2. the cell component model compatible to the CellEditor

5.1.2.1.4 Abstract Cell Component Modeling (Abstraction Level 3)

Meets Requirement R1.III

Finally, the third abstraction level (ABS_LEV_3) provides models featuring an oversimplification. These models might be spheres as can be seen in Figure 89 and also known from The Interactorium (Section 4.3.3.4) or simple cubes as seen in MetNetVR (Section 4.3.3.2). Here, the aim is not a good visualization of cellular reality. The aim is a simple structure, which might be utilized for functional visualizations (see also Section 6.3.1.6).

The generation of these models is very simple, because they are basic models of every modeling tool as well as three-dimensional programming environments.

5.1.2.2 Cell components in CellEditor and the SphereCell

Meets Requirements R1.IV,VI

The section introducing the cell components showed that there are a large variety of cell components (Section 2.1.3). During the development of CellEditor, the portfolio of available cell components steadily grew. For the ABS_LEV_2 and ABS_LEV_3 all cell components used in CellEditor are available. The size of each single cell component belonging to ABS_LEVEL_1 and 2 tries to reflect its natural size. Because the generation of cell components featuring ABS_LEV_1 is quite complicated and high-resolution microscopic data is rare, only ABS_LEV_1 models for mitochondria and chloroplasts are available until now.[52]

Eukaryotic Cell		Prokaryotic Cells	
SphereCellComponent	Size (nm)	SphereCellComponent	Size (nm)
Nucleolus	500	Storage Granule	500
Nucleus	1000	Nucleoid	1000
Rough ER	1500		
Ribosome	2000	Plasmid	2000
		Ribosome	3000
Smooth ER	2500		
Golgi	3000		
Lysosome	3500		
Mitochondrion	4000	Cytoplasmic Membrane	4000
Chloroplast	4500		
Peroxisome	5000	Capsule	5000
Vesicle	5500		
Cytosol	6000	Cytosol	6000
Vacuole	6500		
Endosome	7000		
Cell Membrane	8000		
Cell Wall	8500	Cell Wall	8500
Extracellular Matrix	9000	Extracellular Matrix	9000
Unknown	10000	Unknown	10000

Table 10: SphereCell Composition showing all CellEditor cell component types

Now, ABS_LEV_3 should be focused. In context of CellEditor, the SphereCell was created. This is a sphere-shaped cell containing all cell components in a centered position. In

52 For a full list, please see Appendix 9.17.3.

Table 10 the SphereCellComponents together with the nm-values are listed which apply to the sphere´s diameters. These diameters do not explicitly represent the natural size of the cell components. Moreover, the succession of cell component sizes does not imply the natural order in the cell. But where possible it was tried to take the natural cell component order into account. For example, it is possible to claim that the nucleolus is located inside the nucleus, which is surrounded by the rough ER. It is also possible to locate the cell membrane inside the cell wall which is again surrounded by the extracellular matrix. But it is not possible to fix the position of floating cell components like the mitochondria in comparison to a vesicle.

In the aforementioned table the cell component Unknown is listed. It will be explained in Section 5.3.2.3.3.

5.1.2.3 Coloring Methods for Cell Components

Meets Requirement R1.VIII

Color codes and their application to combinatorial problems is a well-known aspect of information visualization. In 1965, Kenneth L. Kelly published one of the most important works on this topic, introducing 22 colors of maximum contrast [Kell65]. Carter and Carter referenced to the work by Kelly and extended the set to 25 high-contrast colors nearly two decades later [CaCa82]. The question which arises in the context of this work: how can color coding be applied to a cell model for the purpose of supporting its visual perception by the user?

In Section 3.1.1 different microscopy types were discussed. The choice of the staining method changes the color of the cell component on the resulting microscopic image. Therefore, there are arguments for many colors being used in conjunction with cell components.

There are different options to define the coloring of a cell component:

- a histology/staining-based coloring method (for example eosin dyes mitochondria red),
- a cytological-based coloring method, trying to reproduce the “real” coloring of a cell component ignoring dyes (for example ribosomes would certainly appear black without a staining method because of the high RNA-density),
- a function-based coloring method (for example: mitochondria produce energy, energy is often color-coded with red or yellow),
- an aesthetic-oriented coloring method, trying to create a visually qualitative composition of coloring, and/or
- high-contrast coloring methods, maximizing the differentiation of cell component types.

The optimum would be to combine all these aspects to guarantee a beautiful visualization in combination with a high grade of realism and distinction. Of course, it is already obvious that this will not be possible, due to the different focuses of the different color coding methods.

Already at the beginning of the CELLmicrocosmos project the hematoxylin and eosin (H&E, Section 3.1.1.1) coloring was chosen as a good solution in conjunction with cell models of ABS_LEV_1 and/or ABS_LEV_2. First of all, this is one of the most common coloring technique in cytology, therefore the recall value is very high for experts and amateurs who are facing a cell (model) for the first time so they do not get a false first impression. Second, the coloring reflects from a certain perspective the functionality of the cell components. The nucleus is colored in blue, caused by the hematoxylin coloring. The reason is the overall positive charge of the nucleus which is normally associated with blue. The nuclear environment is colored by eosin, because it is on the average negatively charged which is normally associated with the color red. And the third important aspect is the beauty of the resulting visualization, because the use of too many colors normally results in visual chaos [Somm04].

This coloring method works very nice for cell components which can be recognized by their outer shape (ABS_LEV_1 and ABS_LEV_2). But of course the ABS_LEV_1 has a natural problem: all spheres have the same shape. Because of this fact the size, position and the color are the only way to create a distinction. In this case, the H&E coloring is no longer sufficient and the range of colors has to be extended.

The quality of the visual differentiation of a set of variables – in our case the cell components – diminishes with increasing quantity (inversely proportional). A first naive approach is the use of primary colors. The discriminability is very high, but the problem is: there are only three primary colors. In the case of the RGB color space, these are: red, blue and green. There are now two major approaches to be discussed which can be applied to a larger set of variables:

- a contrast coloring method, or
- a sequential coloring method: a color scale, trying to reflect the sequential order of the SphereCellComponents as introduced in Section 5.1.2.2.

5.1.2.3.1 Contrast Color Codes

The high-contrast coloring method, as already proposed by Kelly, Carter and Carter, is the first choice if the distinction of cell components has the highest priority. Table 10 already contains 23 unique cell components (incl. Unknown). The fact that only the most common cell components were implemented in CellEditor leads to the demand to use color codes with the maximum number of high-contrast colors. Kelly´s original color scale is already outperformed, because it contains only 22 colors. Paul Green-Armytage introduced an extensive work on creating a color alphabet, featuring 26 different high-contrast colors [Gree10]. He based his work on the results of Kelly, Carter and Carter and developed various color alphabets which were tested and improved during different workshops prior publication, taking even synaesthetic aspects into account [RiBM05]. The developed scales showed high similarity to the previously mentioned work. It was shown that the alphabetic color codes performed well applied to texts in comparison to shapes, the Georgian alphabet

and faces. In most cases the colors outperformed the alternative alphabets. Green-Armytage states the RGB values of the color alphabet which were applied to the cell components. The first row in Figure 81 shows the resulting colors for the eukaryotic cell. For applying the different colors to the cell components, histological, functional, hierarchical and cell-type-specific aspects were combined.

As a starting point – in the assignment of coloring to cell components – the H&E coloring was used (Section 3.1.1.1). *Ebony*, a nearly black color, was assigned to the most-inner cell component, the nucleolus. The surrounding cell component, the nucleus, shimmers in *navy* and the attached rough ER in *blue*. The last-mentioned cell component is associated with ribosomes, which were assigned to *caramel* because of their dark structure in electron microscopes. The smooth ER is attached to the rough ER, but usually heading towards the cell membrane. Because H&E causes a reddish core environment, the smooth ER lies in the color scale between blue and the *red* of the cell membrane. The cytosol, surrounded by the cell membrane is *pink*. The mitochondria, one of those cell component groups free floating through the cell, was assigned to *mallow*. And the vesicle, found in the neighborhood of many cell components, has a color between blue and red: *amethyst*. Now the colors related to H&E-staining are exhausted. The cell components typical for plant cells were assigned to greenish colors: the cell wall, the vacuole and the chloroplast. Because the chloroplast and the mitochondrion are indispensable for cellular energetic processes, their coloring is very intense. The golgi apparatus which is highly-involved in transportation processes, is *orpiment/orange*, a color often found in the logistic sector. The lysosome, transporting and processing waste materials, shimmering in *(s)lime*. Peroxisomes are involved in the break-down of fatty acids, therefore they were assigned to *yellow*, a color often used in the conjunction with fats. Endosomes are colored in *sky*: they are involved in many cell-membrane-involving processes, featuring a high mobility, communicating and interacting with the outside as well as internal cell components like the lysosomes and – closer to the core region – the golgi apparatus. The extracellular space is colored *xanthin*, the brightest available color from all 26 color codes provided. Finally, the unknown area features a neutral *iron*.

The prokaryotic cell contains only a few additional cell components (see also Table 10). The cell components substituting the cell/plasma Membrane of the eukaryotic cell are colored in *zinnia* (capsule) and *wine* (cytoplasmic membrane). The core region, the nucleoid, has a red/bluish coloring called *damson*. The plasmid which are “naked” DNA (which means, they are not surrounded by a membrane) is colored in *honeydew*. Finally, the storage granule is colored in *quagmire*, a dark green which is often used in conjunction with biological storage (for example in plants). Except the last mentioned one, all color names are found in the Color Names Dictionary published by the US Department of Commerce [KeJu76].[53]

53 The full list containing the cell component color codes in comparison to the original color alphabet is found in the Appendix, Subchapter 9.14.

5.1.2.3.2 Color Scales

For the implementation of color scales, following the previously introduced order of cell component, the comprehensive book of Haim Levkovitz, "Color Theory and Modeling for Computer Graphics, Visualization and Multimedia Applications", was taken into account [LeHe92, Levk97, Natt04]. In this work, different color scales are discussed, which are defined as follows:

"A color scale is a pictorial representation of numerical values in which each value is assigned its own color." [Levk97, p.112] (17)

It is important to notice that the difference to a simple color coding method is the implementation of sequential values.

Levkowitz stated, that the linear gray scale outperforms even optimized color scales in terms of a sequential, ascending order. Even his newly proposed scale (discussion follows) is outperformed by the linear gray scale in certain tasks like image analysis and the distinction of small abnormalities [Levk97, pp.126, 132]. But the examinations of Levkowitz were based on two-dimensional visualizations. For three-dimensional visualizations, the gray scale is normally not appropriate.

In a two-dimensional visualization, the position in relation to the viewport is always equal; this is not the case in three-dimensional space. 3D objects may be hidden or only partly overlapped; they may be located far in the background or very close to the user in the foreground. For this reason the shading of nodes is an important factor to distinguish the position in the environment, because shading is used to emphasize the spatial relations in virtual space; especially if techniques like 3D Stereoscopy are not available (see also Sections 5.3.2.10 and 5.3.2.11). But shading implies different gradations of gray. Therefore a gray scale for the differentiation of objects is inadequate. An appropriate alternative are color scales.

Levkowitz evaluates different coloring methods in his work. These coloring schemes, and, the one originally developed for CellEditor, were applied to the SphereCell as can be seen in Figure 89.

5.1.2.3.3 The Color Codes/Scales in CellEditor

A short list of all coloring codes/scales shown in this Figure 89, stating the author and the scale´s full name follows:

- *Contrast Colors:* Based on Green-Armytage`s Color Alphabet [Gree10],
- *BTC:* Blue to Cyan (Thrasos Papas, AT&T Bell Labs, Murray Hill, NJ),
- *BTY:* Blue to Yellow (Canon Research, Sydney, Australia),
- *Heated Object:* Perceptually linearized Heated Object Scale (Steve Pizer, UNC Chapel Hill),
- *H&E:* Hematoxylin and Eosin (Björn Sommer),
- *Linear Gray:* Perceptually linearized gray scale (Haim Levkowitz),

- *LOCS:* Linearized Optimal Color Scale (Haim Levkowitz),
- *Magenta:* Perceptually linearized Magenta Scale (Steve Pizer, UNC Chapel Hill),
- *OCS:* Optimal Color Scale (Haim Levkowitz), and
- *Rainbow:* Perceptually linearized Rainbow Scale (Steve Pizer, UNC Chapel Hill).

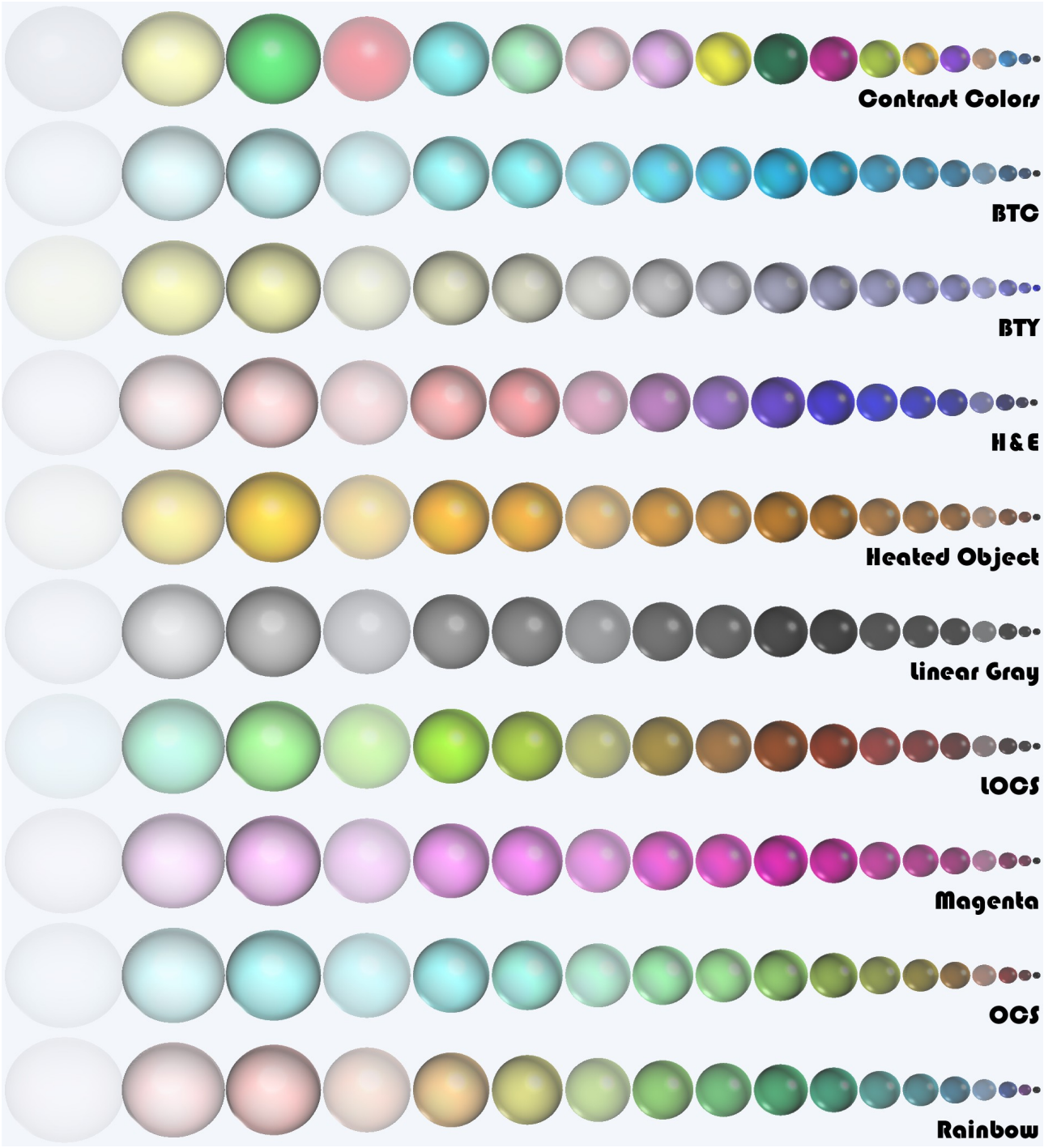


Figure 89: The different color scales used in CellEditor with the SphereCellComponents The color distance at each step was set to 14 (~256 colors/18 cell components), starting from Unknown at the left to Nucleolus at the right[54]

54 The eukaryotic cell´s cell component order is derived from Table 10, left to right: Unknown, Extracellular Matrix, Cell Wall, Cell Membrane, Endosome, Vacuole, Cytosol, Vesicle, Peroxisome, Chloroplast, Mitochondrion, Lysosome, Golgi, Smooth ER, Ribosome, Rough ER, Nucleus, Nucleolus.

The Contrast Colors were explained in Section 5.1.2.3.1.

The Linear Gray scale is used as a reference. But as mentioned before, it is not appropriate for a 3D-environment.

The BTC and BTY show simple transitions between two colors.

The Heated Object scale is based on the fact that the human visual system is known to be most sensitive to luminance changes for the orange-yellow hue. Vice versa, the color scale which relates to the fact that the human visual system is most sensitive to hue changes for the magenta hue, is the scale with the according name [Levk97, p.112].

Traversing the color solid along a path from black to white through all the hues of the rainbow generates the Rainbow scale [Levk97, p.111].

OCS is a scale developed by Levkowitz, trying to integrate an optimal sequence of color gradations. For this purpose he developed a special algorithm, the "Optimal Scales Algorithm" [Levk97, p.123].

In this context the question arises: what is an optimal color sequence? If the values of two points in a color sequence differ in their RGB values, there is no guarantee that human vision can perceive these distinct differences. Therefore, the term *just noticeable differences (JNDs)* was introduced, defining the number of visually distinguishable colors. The optimization of a color scale tries to maximize the number of distinct perceived colors on a monitor while simultaneously maintaining the natural order among its colors. The naturalness depends on the hue and saturation of the colors and how these aspects match the human viewing-pattern. To generate a color scale matching these prerequisites, different restrictions have to be applied. And this was also done during the development of the OCS. Finally, the LOCS is the linearized version of the OCS generated with another algorithm developed by Levkowitz [Levk97, pp.118 ff., 138 ff.]:[55]

"Linearization of a color scale is a process in which additional colors are inserted in the scale in such a way that the perceived distances between adjacent colors of the extended scale are as uniform as possible." [Levk97, p.126] (18)

Finally, it can be stated, that the different color scales provide appropriate possibilities to color cell components. As will be discussed later, normally not all cell components are needed for functional visualizations. For this purpose the Primary Colors will also be sufficient. Simple color scales representing transitions between two colors like the BTC and BTY are somewhat problematic caused by the shading and transparency effects. In Figure 89 the first (unknown), fourth (cell membrane), seventh (cytosol) and 14th (ribosome) column all show a relatively light color, caused by a higher grade of transparency. The reason for this will be discussed in the following section.

It can be seen that the use of different colors like in the (L)OCS and Rainbow scale is reasonable because the visual differentiation is much easier for the observer.

55 The explanation of these algorithms lies beyond the scope of this work. Please refer to the according chapters in Levkowitz book "Color Theory and Modeling for Computer Graphics, Visualization and Multimedia Applications" [Levk97].

5.1.2.4 Cell Component Layers

Meets Requirement R1.V

In the section discussing the cell components it was shown that many cell components contain more than one layer 2.1.3. In contrast to other Virtual Cell approaches ignoring this fact during the generation of cell models, like Meta!Blast (Section 4.1.1.2), MetNetVR (Section 4.3.3.2), The Interactorium (Section 4.3.3.4) and 3DScape (Section4.3.3.6), CellEditor supports multiple cell component layers.

Figure 90 shows an example of a mitochondrion compatible with CellEditor. It contains five layers: the *matrix*, the *inner membrane*, the *intermembrane space*, the *outer membrane* and the *cloud*, representing the close environment of the mitochondrion.

Resulting from these definitions it is inevitable, that a mitochondrion model designed for CellEditor contains these five different shapes maintaining the order of layers. In Figure 25 the code of a mitochondrion model in VRML97 format is given. The structure of the "DEF"-sections in the VRML-file follows exactly the definition of the layers needed for CellEditor.

In addition, another issue of the preceding section can be revealed: the higher transparency of cell components like the unknown or cell membrane component result from the fact that they contain only one transparent layer, in contrast to cell components like the mitochondrion, which appears to be quite opaque in Figure 89. In the case of the mitochondrion model, the resulting structure from Figure 90 shown in Figure 82.2 contains two invisible layers, the cloud and the intermembrane space. All other layers feature medium transparency, so that the underlying structures can also be seen.

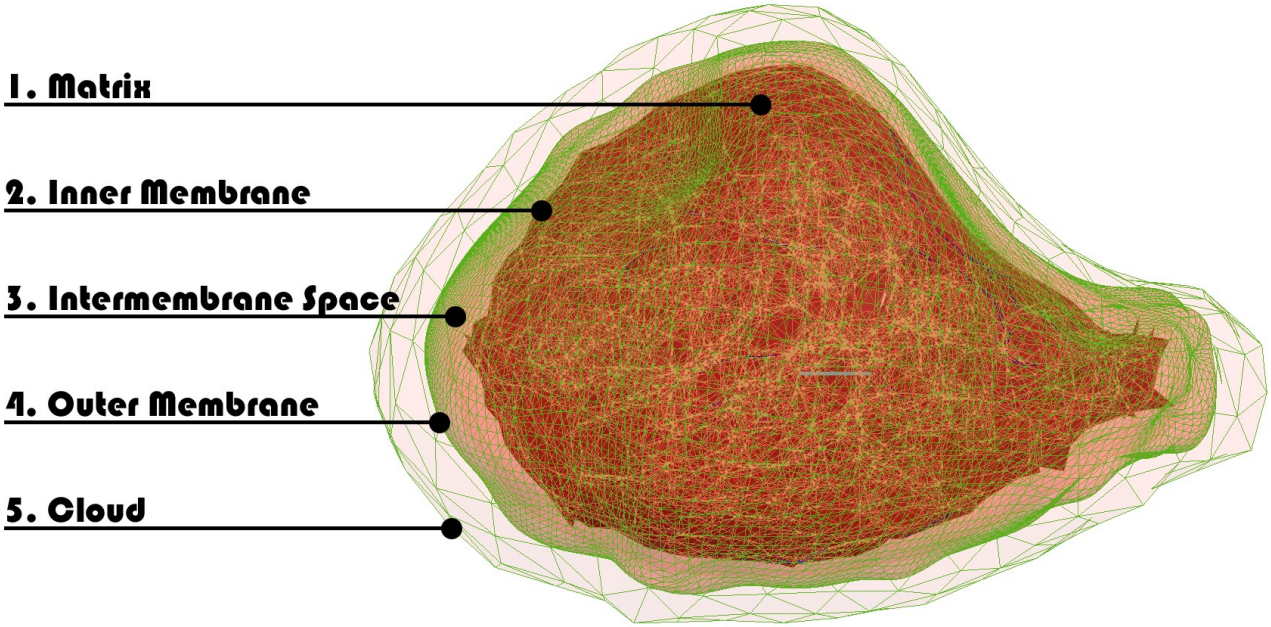


Figure 90: The localization layers in CmCX of the mitochondrion

5.1.2.5 CellEditor

Meets Requirement R1.VI

CellEditor is part of CellExplorer application (see also Section 5.1.3). It is used to create a Virtual Cell model. Figure 91 shows this process.

First, the user chooses the type of cell he wants to create, eukaryotic or an prokaryotic cell. Here, the first-mentioned option is chosen.

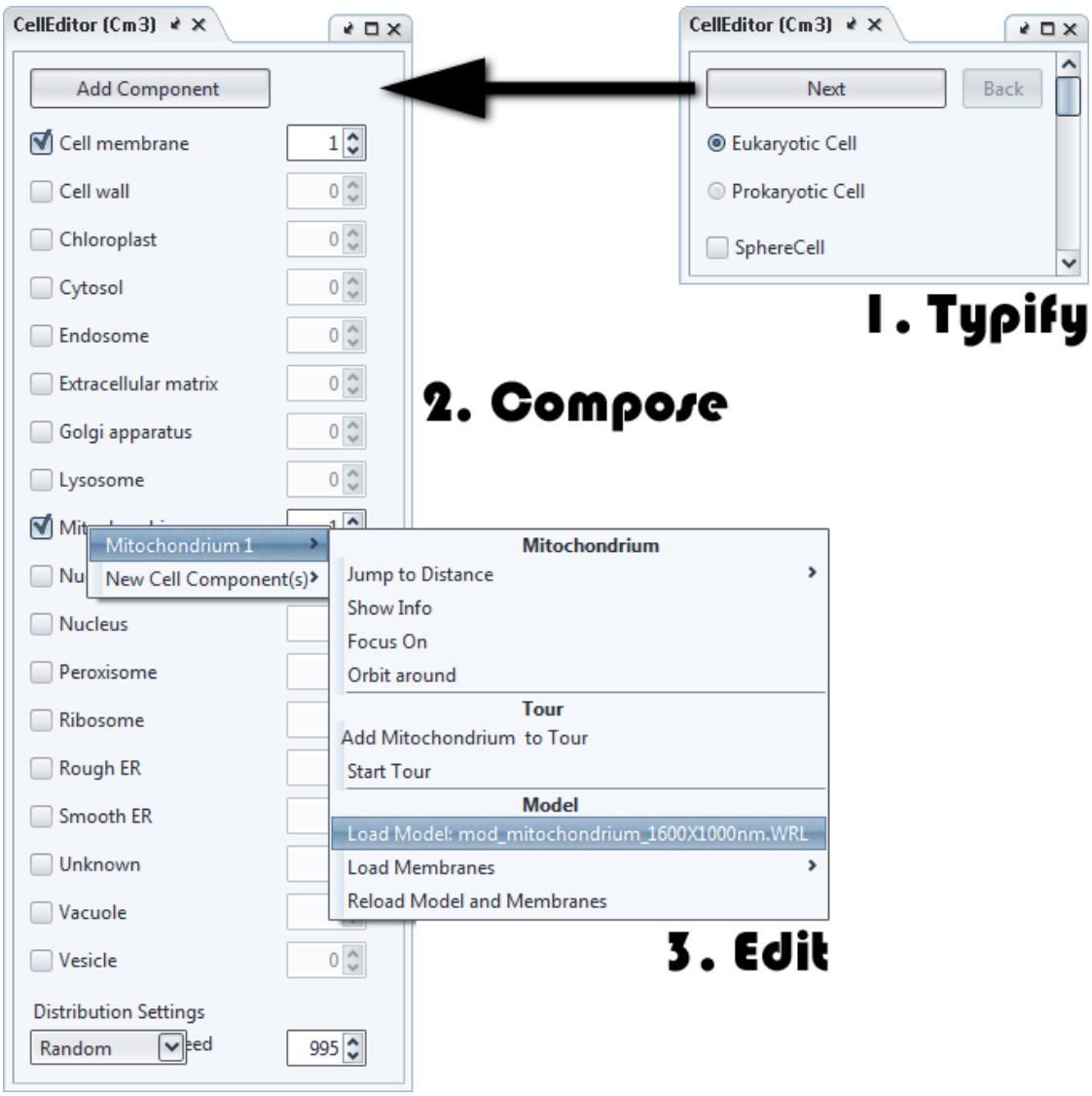


Figure 91: CellEditor: the menus
1. the cell type is chosen; 2. the cell model is composed and generated; 3. the generated cell model can be optionally edited

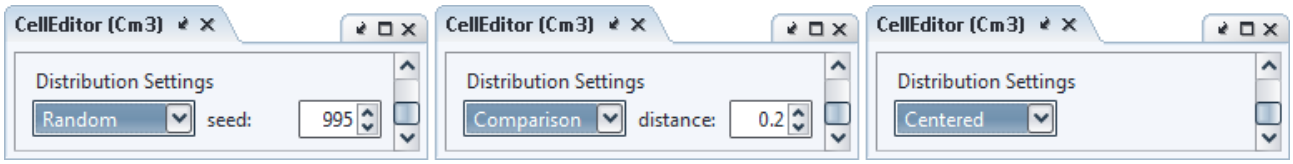


Figure 92: CellEditor: the 3 distribution modes of the integrated CellEditor

In the next step, the main window of CellEditor appears. Here, all cell components are listed, matching the ones already shown in Table 10. First, all cell components are deactivated. If now a check box is activated, normally a window appears providing the opportunity to choose the appropriate VRML97 model. If in the first step the “SphereCell” check box was activated, this window will not appear and the program will automatically generate spherical objects after pressing “Next”. But in the case shown below, a regular cell is built, also providing the option to place more than one cell component of the same type. Only the cell membrane is restricted to one single component, because they are the reference object for the cell component distribution which will be discussed in the next section.

After the cell components are distributed, the cell model can still be customized, as shown in Step 3. For example, cell components can be substituted or new cell components can be added. In addition, by deactivating the check box left of the cell component names, the according cell component type is hidden. The other options shown in the pop-up of step 3 will be discussed later (Section 5.4.2.3).

5.1.2.6 Placing cell models

Meets Requirement R1.VII

A very simple but effective method was implemented to distribute cell components.

Figure 92 shows the three placing modes using CellEditor. The Random mode is default. Cell components are distributed randomly in the place. But there are exceptions.

The reference point is the cell membrane for eukaryotic cells and the cytoplasmic membrane for prokaryotic cells. The algorithm tries to distribute the cell components inside the boundaries of the outer cell membrane. But in contrast to the complex packing algorithms discussed in Section 5.2.2.5, these are only very rudimentary approaches.

In this case, instead of the geometric structure of the cell components used for distribution, only the bounding spheres are taken into account as shown in Figure 93. The distribution algorithm first places the cell membrane. Then, the nucleus is placed inside the cell membrane and in its center; the nucleolus is activated. Then the remaining cell components are placed. If the algorithm fails to find a position, where the actual cell component does not intersect with an already placed cell component and the outer border represented by the cell membrane in a predefined period. The algorithm is stopped and the object is just placed somewhere near the center of the cell membrane.

There are, however, some cell components which are excluded from the distribution process: the cytosol, the cell wall, the extracellular matrix and – naturally – the cell membrane. Those are placed all with the same rotation in the origin of the coordinate system. For the prokaryotic cell, this collection of fixed cell components is extended by the storage granule, the nucleoid, the plasmid, the ribosomes, the cytoplasmic membrane and the capsule.

In the context of modeling a very important aspect is the reproducibility of the results. For this purpose the random generation is seed-based. Using the same seed combined with the same selected cell components will generate the same results again.[56]

The position of the cell components can be also changed manually after the placing process has completed. For this purpose, CellEditor provides rudimentary mouse-based placement features (in conjunction with the features discussed in Section 5.3.2.10).

Still, there are two distribution modes to explain. Figure 92 shows the “Comparison” mode which can be used to align cell components of the same type along an axis (Section 6.1.1) and the “Centered” mode which is especially important for the SphereCell (Section 6.1.4).

56 The MembraneEditor also uses a seed-based random generation. See Section 5.2.2.8.

Figure 93: CellEditor: the bounding spheres of the cell components
(Result from student project: CELLmicrocosmos 3 CellEditor, SS2007, see Appendix 9.17.2)

5.1.2.7 Import and Export Capabilities

Meets Requirements R1.IX,X

As mentioned before, VRML97 models are imported and used as three-dimensional shapes. If a cell composition is finished, it can be saved in a special XML-compliant format called ".Cm3".

This format is optimized for CellEditor and was inspired by the similar packaging techniques used, for example, by Amira (Section 3.1.3.1). Figure 94 shows a Cm3-package of a Prokayotic Cell. It includes one Cm3-file and a set of linked VRML97-files.

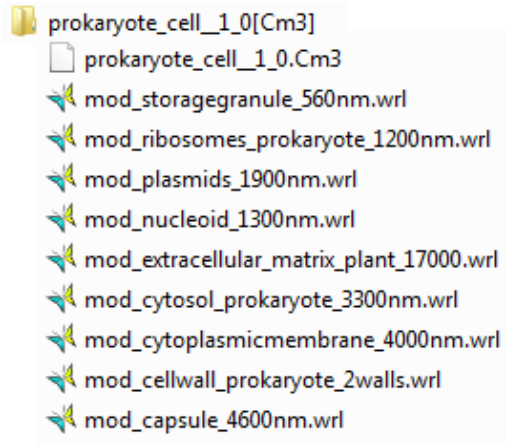


Figure 94: CellEditor: the package structure of a Cm3 package

The content of a Cm3-file is shown in Figure 95. It defines the cell type in the "placePDB"-tag, the linked VRML97-files in the "vrmlFile"-tag and in the "vrmlObject"-tag, the 4x4 matrix, containing the rotation and the translation of the VRML-object. The "pdbFiles"-Tag will be described later (Section 5.4.2.3).

```
<?xml version="1.0" encoding="UTF-8"?>
<placePDB cellType="p">
  <execution type="mapObject">
    <vrmlFile name="mod_extracellular_matrix_plant_17000.WRL" />
    <vrmlObject name="Extracellular Matrix" matrix="1.0,-0.0,0.0,-0.0,-0.0,1.0,-0.0,-0.0,0.0,-
0.0,1.0,-0.0,0.0,0.0,0.0,1.0" />
    <pdbFiles>
      <pdbFile name="null" />
    </pdbFiles>
  </execution>
  […]
</placePDB>
```

Figure 95: CellEditor: a snippet of a .Cm3-file

5.1.3 Implementation Details

The CELLmicrocosmos 1.1 CellExplorer (CmCX) was developed based on the CELLmicrocosmos 3.1 CellEditor (CmCE), whose capabilities were discussed in the preceding section.

These programs were developed in Java 6, but they are also upward-compatible with Java 7. For 3D visualization the Java 3D version 1.5.2 library is used. Currently, it is available as an access-limited Web Start application to start directly from the browser (Section 3.4.3.1). Potential users need to login to test this application. It will be released after the first major publication is issued. The project is located at:

http://Cm1.CELLmicrocosmos.org

Please refer to the Appendix 9.17 for information according the implementation work and to Appendix 9.15 for version information. The included third-party libraries and their licenses are shown in Table 11.

Name	Version	Usage	License	More Info
Infonode Docking Windows	1.6.1	Manages the different windows in the main GUI	GPL	http://www.infonode.net
j3d-core, vecmath	1.5.2	Displaying and picking of the 3D objects in the Membrane View	GPL 2 (with CLASSPATH exception)	http://www.j3d.org
j3d-core-utils	1.5.2	Picking and Navigation	BSD (without advertising clause)	http://www.j3d.org
j3d-vrml97	0.1.0	VRML97 Import	BSD	http://java.net/projects/j3d-vrml97
JDOM	1.1.1	Loading and saving of XML files	Apache License (without acknowledgment clause)	http://www.jdom.org
Substance	4.0	The L&F of the GUI	BSD (without advertising clause)	http://substance.dev.java.net

Table 11: Program packages included in the CELLmicrocosmos 1.1 CellExplorer relevant for this subchapter (CELLmicrocosmos 3.1 CellEditor)

5.2 [MOL] Molecular Modeling: Integrating Lipids and Proteins into a Membrane

Modeling at the molecular level is a quite complex task, as it can be seen by reviewing the Subchapter 2.2: how can a membrane be computationally generated? How is it possible to place lipids in combination with proteins? In Subchapter 4.2 related membrane modeling tools were introduced. It was shown that these programs provide a) good solution for specific problems but they are quite complicated for the user or b) are very simple to use but produce impractical results. Here a user-friendly program will be introduced which provides a very broad field of applications: the CELLmicrocosmos 2.2 MembraneEditor (CmME).

The theoretical background was mainly discussed starting from Section 3.2.4 covering the Membrane Packing Problems. These definitions will be extended by concrete algorithms in the next section. Again, before examining MembraneEditor, different requirements have to be taken into account (Section 5.2.1). Then the methods will be discussed needed for the development of the application (Section 5.2.2) and the implementation details will be subsumed in the last part of this subchapter (Section 5.2.3).

5.2.1 Requirements

R2.I Compatibility with different molecule types

The import of different molecules – like lipids, proteins, carbohydrates – has to be possible (Sections 2.2.1.3,2.2.1.4).

R2.II Compatibility with the Fluid Mosaic Model

The major aspects of the Fluid Mosaic Model discussed in Section 2.2.3 have to be taken into account, in order to generate a model which is most widely compatible with the biological reality of a membrane environment.

R2.III Membrane Composing

As was seen in Section 2.2.6, the composition of the membrane can be defined by percentaged or absolute quantification of molecules, especially lipids. Therefore the ability to define the composition is required.

R2.IV Incorporation of microdomains

Microdomains and lipid rafts play an important role in membranes. Therefore it is important that areas can be defined featuring special lipid compositions.

Based on the technical background (see Subchapter 3.2) the following aspects have to be implemented:

R2.V Performance for Desktop PCs and Laptop PCs

The application has to show good performance values at regular PCs without the need to use server PCs or even cluster environments (Section 3.2.3). It has to be possible to generate preview membranes within a few seconds as well as more precise compositions which may need a few minutes. Of course, for special, extreme application cases it may be

necessary to run an algorithm for a few hours or even days, but this should be no prerequisite for models featuring a high packing quality (see also R2.XI).

R2.VI Solution to Membrane Packing Problems

The computer-science-related term was defined as the Membrane Packing Problems (see Section 3.2.4). Algorithms implemented to generate membrane assemblies have to be classified according the MPP.

R2.VII Restriction to Geometrical Problem Solving

The former two requirements result in the restriction of shape-based, geometry-based problem solving approaches. For the generation of membranes featuring a high grade of realism at the atomistic level, molecular simulations are needed. But reasonable simulations require also high computational power. For the composing of membranes this step is not urgently needed, as later will be shown (see Section 6.2.4). Therefore the modeling of membranes will be restricted to the molecular level without taking atomic interactions into account. In addition, it is sufficient to generate one snapshot of a membrane, because the time-dependent aspects would be subject to an external simulation with appropriate tools like GROMACS (Section 3.2.3).

R2.VIII Semi-automatic Placement of Proteins

A number of databases exist which provide information to enable the automatic placement of proteins. The discussion of PDBTM and OPM (see Sections 3.2.2.1.4 and 3.2.2.1.5) showed, that they provide averaged placing information. Therefore in many cases – even if the approaches of these two databases are applied – it will be a reasonable step to verify the automatic placing process manually. This is the reason for describing this placing process as *semi-automatic placement*.

R2.IX Access to Databases: PDB, PDBTM and OPM

The former requirement as well as requirement R1.I is accompanied by the need for a direct connection to these databases.

R2.X Visualization

The packing process as well as the result should be viewable and explorable.

R2.XI Packing Quality Verification

After a membrane is generated there should be an option for verification the quality of the result.

R2.XII Export capabilities for further visualizations and simulations

Because this approach is restricted to the molecular level it should be possible to export membranes for further molecular simulations and external visualizations.

R2.XIII Import capabilities to analyze and visualize externally modified membranes

There should be a possibility to re-import membrane models generated with MembraneEditor in case the atomic structure was changed by external programs like GROMACS (Section 3.2.3).

R2.XIV Modularity

To enable external users access to the basic functionalities of the program and the development of plugin algorithms, an interface should be provided.

R2.XV Reproducibility

The results should be reproducible.

5.2.2 Methods

Now the major features of MembraneEditor will be discussed, always taking the aforementioned requirements into account. Figure 96 shows the modeling pipeline at the microscopic level. The whole pipeline is found in Figure 7.

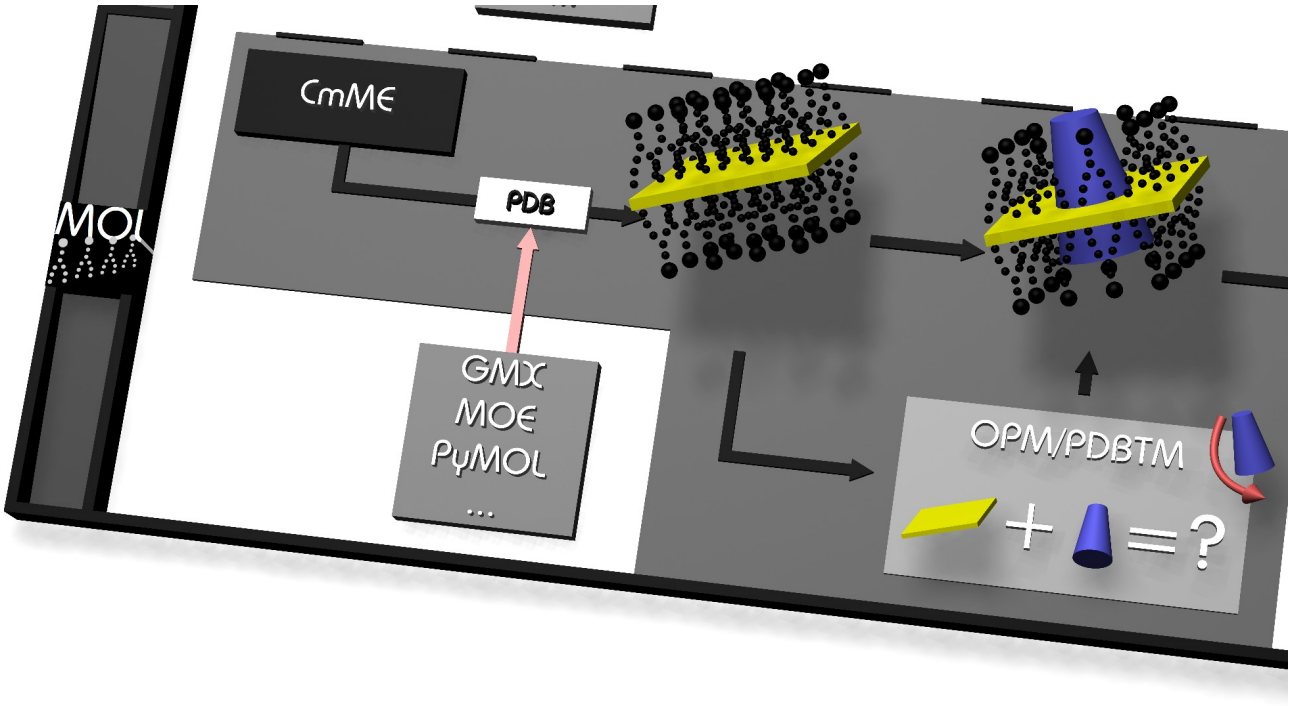


Figure 96: The CELLmicrocosmos modeling pipeline at the microscopic level

5.2.2.1 PDB Integration

Meets Requirements R2.I,XII

As elaborately discussed in Section 3.2.2.1.1, PDB is today's most acknowledged format for molecular structures: proteins, lipids as well as many other molecules. Therefore this format was chosen as the appropriate import/export format right from the beginning of the development process.

By choosing PDB as the base format, nearly any three-dimensional structure of a molecule can be imported into MembraneEditor. In addition, all structures generated with MembraneEditor can be exported to PDB.

A problem of the PDB format is that some applications need a special formatting to be able to import the files[57]. For example, some applications accept the use of case-sensitive digits for the chain identifier (Appendix 9.6).

57 A list is of programs compatible to MembraneEditor is found in Section 4.2.1.

For this reason, MembraneEditor provides a PDB Property dialog which can be used to adjust the settings to the respective application. This dialog will also be important if the number of different identifiers (such as chain identifiers and serial numbers) in a membrane model exceed the number of possible combinations. In this case, the secondary structure cannot be saved to the PDB file and the PDB Properties dialog should be used to deactivate the saving of the secondary structure.

5.2.2.2 Lipid and Protein Packing

Meets Requirements R2.III,VII,VIII,IX

Looking back to the chapters introducing molecules from a biological (Subchapter 2.2) as well as from the technical point of view (Subchapter 3.2) showed that there are two major molecule types which have to be differentiated (Figure 33); proteins and lipids.

5.2.2.2.1 Lipid Packing

In contrast with proteins, lipids are relatively similar in shapes and size. Of course, it was discussed that there are dramatic differences between different types of lipids from a biological, physicochemical perspective. But when focusing on the geometrical properties of lipids, the differences are not that crucial. Therefore, the lipids are distributed onto the membrane area by using Lipid Packing Algorithms (see Section 5.2.2.5). To enable the generation of realistic lipid distributions, their percent ratios feature a precision of up to three decimal places [Rube08].

Because there is no extensive lipid database on the web providing download functionality, lipids can be only imported from the local file system.

5.2.2.2.2 Protein Packing

The proteins are quite complex structures which extremely vary in size, composition and structure. All original PDB proteins can be directly downloaded from the PDB database. Additional protein models can be imported from the local file system.

The horizontal position of a protein is defined manually via the Drag and Drop mechanism or automatically by placing it at a random position into the membrane. The vertical placement is a process which can be done manually by rotating and translating the protein with the mouse, until it meets the biological expectations (Figure 98). Of course, this process should be automatized where possible (OPT_PPP_POS). Therefore, two approaches have been implemented.

By directly accessing the PDBTM website, a list of all transmembrane proteins included in the PDBTM database is generated. The MembraneEditor uses this list to mark those PDB files providing the PDBTM matrix information. This matrix, originally generated by the TMDET algorithm, can be used to automatically place a protein into the membrane (Section 3.2.2.1.4) [Gamr09].

The second option to automatize the protein placing process is the download of OPM PDB structures. These structures can also be directly downloaded from the OPM database by MembraneEditor. Afterwards it is possible to use the dummy atoms contained in these special PDB files to re-orient the protein with respect to the membrane layer (Section 3.2.2.1.5).

First, the existence and the position of N and O dummy atoms is used to define the initial position of a protein:

- If there are only N dummy atoms, the protein is an extrinsic one and has to be placed above the lipid layer.
- In contrast to that, if there are only O dummy atoms, they belong to an intrinsic protein which is initially placed at the bottom outside the lipid layer. In the next step, the protein is moved towards the lipid layer until the dummy atoms collide with the lipids.
- In addition, the existence of N and O dummy atoms indicate a transmembrane protein. In this case, the center of both dummy layers is placed in the center of the lipid layer.

Recently it is also possible to use custom PDB files (like those from the Protein Model Portal) as the base for computing an OPM file by using the following website (Section 3.2.2.1.5):

http://opm.phar.umich.edu/server.php

After generating the corresponding OPM file, it can be downloaded to the local file system, from where it can be directly imported into MembraneEditor.

Figure 97 shows the “Download a PDB” window. This is used to download files directly from the databases PDB and OPM. In addition, icons indicate for which proteins PDBTM information is available. Moreover, the information for all three databases can be updated here: the two index files containing all PDB files available from the PDB and OPM database and the XML-file containing all information needed for the automatic alignment parsed from the PDBTM database.

5.2.2.2.3 Membrane Packing

There are two possibilities for combining a protein with a lipid layer.

- Prior Placement: The protein is first placed onto the empty membrane layer and afterwards the lipids are distributed onto the layer by omitting collisions with the protein. By using the OPM method, it may be necessary to reposition the non-transmembrane proteins vertically because the collision-detection needs concrete lipids for an accurate placement.
- Post Placement: The protein is placed into a precalculated layer and the intersecting lipids can be removed in the next step.

5.2.2.3 Shape-based and Atomic-based Computation and Visualization

Meets Requirement R2.II,V,VII,X

For minimizing the complexity of the packing processes it is important to reduce the complexity of the different molecules containing up to thousands of atoms. The MembraneEditor provides two approaches to make abstractions of the complex molecular structures:

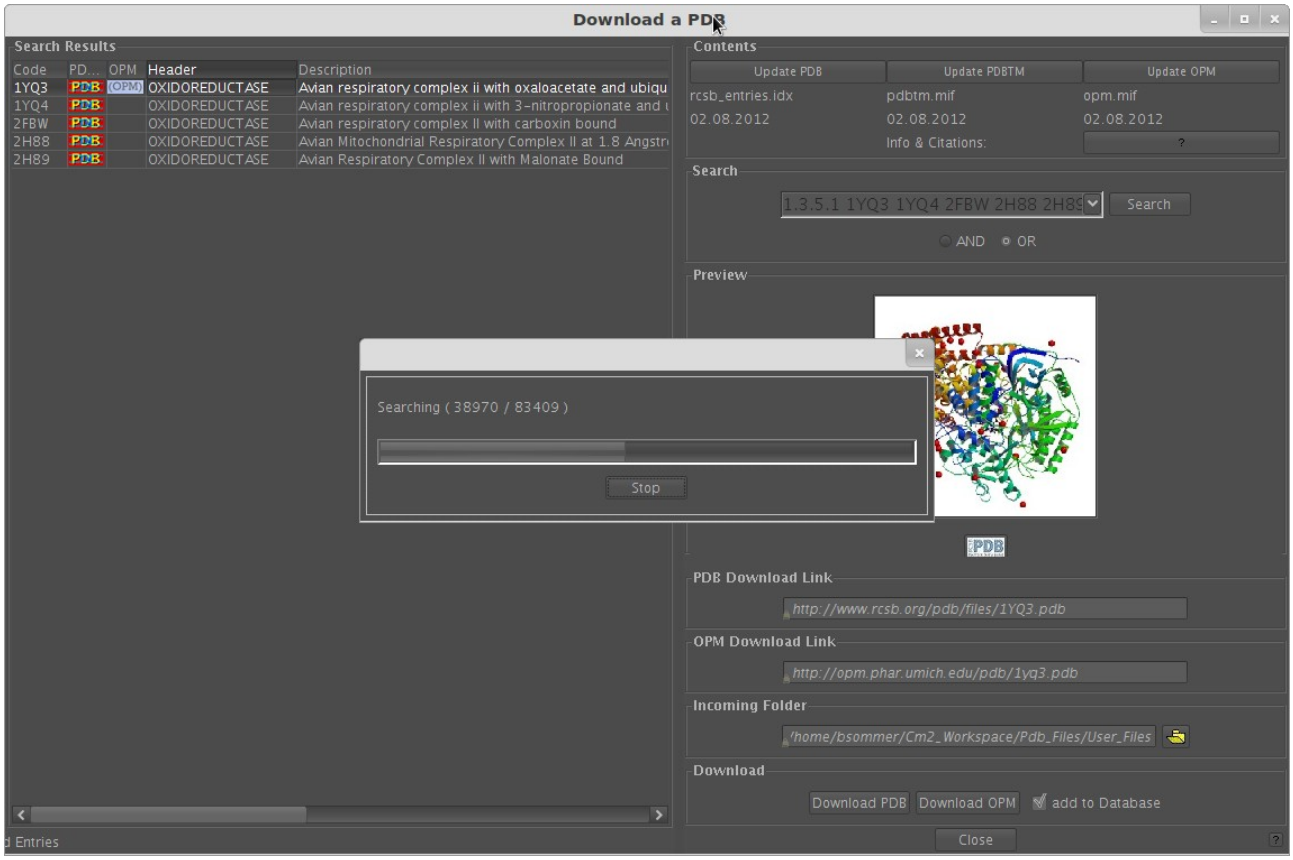


Figure 97: The "Download a PDB" dialog
It is used to download PDB files from the PDB database and OPM files from the OPM database and to indicate for which proteins PDBTM information are available

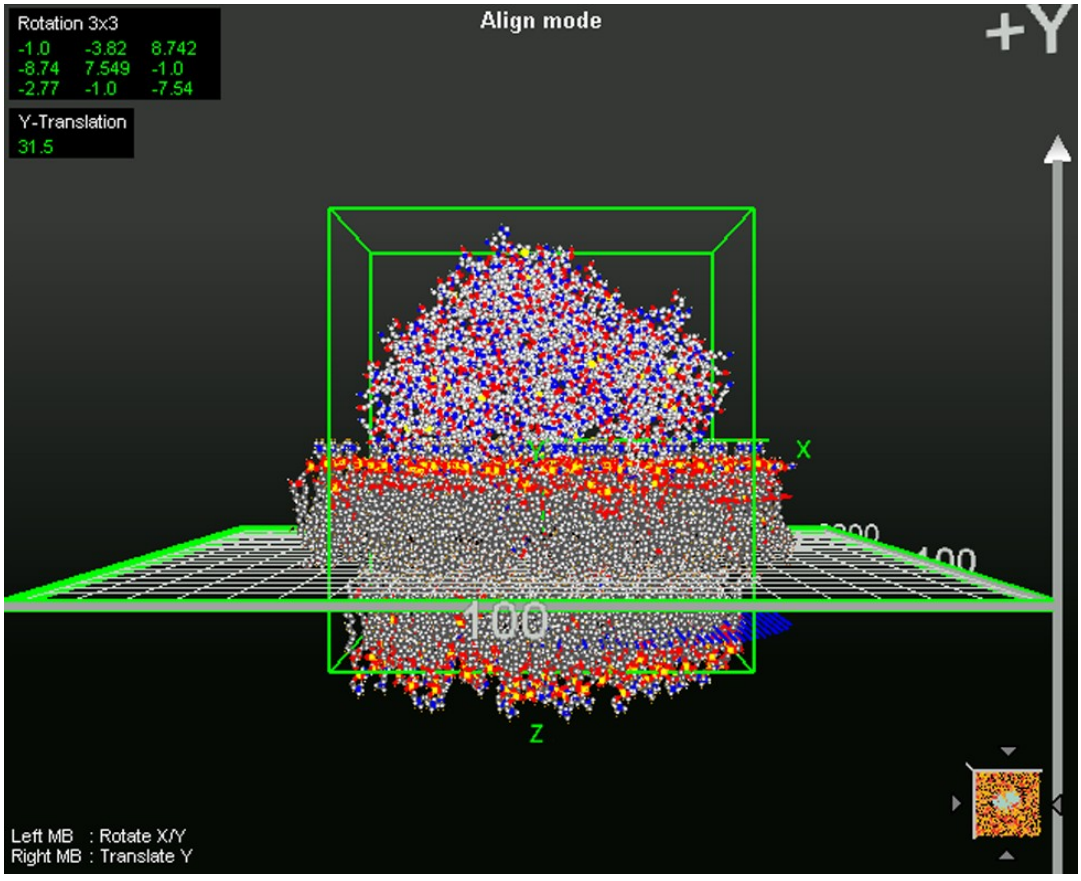


Figure 98: Automatic placement of cytochrome c oxidase The Alignment mode shows 1O5W surrounded by the OPM layers marking the outer (red) and inner (blue) membrane leaflet and the semi-transparent bilayer (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

1. The shape collision method (Figure 99 left) generates a three-dimensional outer shape of the molecule providing a closed surface for collision-detection. The closed surface prohibits smaller parts of other proteins, lipids or even smaller molecules from extending exemplarily into the interior of the pores of a protein channel. The detail level of the shapes may be changed, affecting the number of triangle points, the shape visualization and the computational complexity of the collision-detection.
2. The atom collision method (Figure 99 right) takes relevant atoms of the outer shape of the molecule into account. During the collision-detection process those atoms are tested for spherical intersections. In contrast to the shape collision method, atomic intersections are completely avoided, but molecules may overlap or extend into the interior of a molecule.
3. In addition, both methods may be combined.

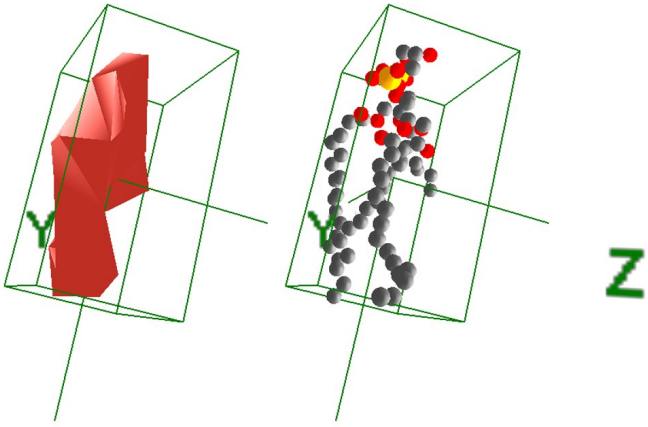


Figure 99: The two collision modes used in CELLmicrocosmos 2.2 MembraneEditor They are geometrically shown on Cardiolipin (cdn_exp): Shape (left) and atom collision (right) (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

The shape-based visualization, reflecting exactly the shapes used in method 1, enables the visualization of quite complex membrane models, for example, large membrane patches or membranes containing very large or a large number of proteins. Of course, MembraneEditor provides also standard visualization modes like “covalent” or “van der Waals radii” and “Balls and Sticks”.

Because it is not a computationally simple task to visualize a large number of atoms, MembraneEditor is supplied with an external version of the Jmol viewer, which can be used to preview membranes externally [00a, Herr07].

5.2.2.4 Membrane Model: Stacks and Microdomains

Meets Requirement R2.II,III,IV

The standard shape of a membrane model patch is rectangular. The size is defined in Ångstrom. To enable the combination of different membrane stacks, multiple mono layer or double layer stacks can be modeled. In addition, it is possible to define microdomains by drawing spherical, rectangular or irregular shapes into the membrane layer (see Section 2.2.5). These microdomains feature a custom lipid composition which has to be defined. They can be also used to generate membranes with irregular shapes. The mobility of the lipid during the packing process using the algorithms of MembraneEditor is restricted to the concerned microdomain.

5.2.2.5 Lipid Packing Algorithms and the Two-and-a-half-dimensional Knapsack Problem

Meets Requirement R2.VI

Now that the different preconditions for the lipid packing have been discussed, finally the problem-solution approaches to the LPP have to be discussed: the *lipid packing algorithms (LPA)*. As mentioned before, all algorithms discussed in this section focus on the outer shapes of the stiff molecules – intermolecular atomic repositioning is omitted. Each introduced algorithm features a number of properties which can be defined by the user. Normally this is done prior starting an algorithm. In some cases it may also be possible to change special settings during the algorithm's runtime. In any case, it is possible to run all algorithms discussed here with and without visualizations, the latter case leads normally to a faster computation time.

In this section a number of standard lipid packing algorithms are discussed which have been implemented for MembraneEditor by Tim Dingersen during his student assistantship supervised by the author of this work.[58] These algorithms will be compared later to the plugin algorithm The Wanderer, discussed and developed in the context of this work (Section 5.2.2.6 and Subchapter 6.2).

The simplest algorithm discussed here is the *Linear Placing (LP)*. It places lipids row by row, starting from the top-left position and terminating at the bottom-right position. It adds molecules successively without reallocation and without knowledge of the succeeding items a priori [Dyck90]. Its packing behavior is similar to those of a 2D-KP (see Section 3.2.6). The reason is that not the molecular shape but rather the bounding box – represented by the width and depth of its boundary area – is used during the placing process as collision indicators. The LP is not a full-value optimization algorithm, and therefore it is also not 2D-KP. If the new lipid to place cannot be positioned at a special position, because it collides with another molecule or the membrane border is reached, it is skipped and the lipid is placed onto the next free position. Because this algorithm lacks any optimization steps and relies on the bounding boxes, best packing is achieved using a uniform lipid orientation.

The *Random Placing (RP)* algorithm uses a combined random placing and orientation process to create a well-filled bilayer. A relatively high lipid packing density (D_l, see Section 3.2.7) is achieved by a cyclic process of shaking all lipids iteratively and approaching the nearest protein or, alternatively, the membrane center by using the collision-detection of MembraneEditor. The free space which may be created by this method is used to place new lipids onto the membrane. The algorithm needs to be stopped manually if the D_l of the membrane layer meets the expectations of the user. The integration of approaching procedures shows that this algorithm is an optimization algorithm in contrast to LP. The optimum is defined by OPT_LPP_MIN_AREA (see Section 3.2.4).

58 For a detailed discussion of the different lipid packing algorithms the reader is referred to the publication [SDGS11]. “Standard” lipid packing algorithms are those algorithms directly installed in conjunction with MembraneEditor.

In addition, there is a special version of RP called the *Distributor*. It integrates the RP algorithm as the basic filling process. But instead of limiting the number of runs by the manual user-invoked termination, the Distributor runs until the predefined number of lipids – D_l respectively – is reached. The absolute number of lipids can be defined as the termination requirement or the area per lipid to reach. This algorithm can also be used to alter membranes generated with other algorithms. Therefore, the regarding optimization class is OPT_LPP_DEF_AREA. Of course, it is possible that too ambitious number of lipids is not reachable because the membrane layer is already crowded. In this case, the optimum is defined by OPT_LPP_MIN_AREA.

The third algorithm in the family of random placing algorithms discussed here is the *Advanced Random Placing (AP)*. This algorithm is also finite, but in contrast with the Distributor it is not possible to define a concrete D_l. Another unique characteristic is the subdivision of the membrane patch into smaller parts during computation. For all these sub-problems the RP method is again applied. Additional user-definable properties are the range for the random orientation, shifting and tilting. Each of those parameters affects the runtime. Because of its finite character combined with multiple trial-and-error random placing attempts, trying to find local optima, AP represents a classical greedy algorithm. Again, the optimum is defined by OPT_LPP_MIN_AREA.

Finally, the nature of these KP approaches has to be examined. In Section 3.2.6 the 2D-KP was described as the temporarily classification of the LPP. But is this the correct description for this algorithm?

The lipid is moved on the two-dimensional lipid area only in the X and Z direction. If the LP algorithm would feature an optimization function, this algorithm would belong to the class of 2D-KP. But during the collision-detection – used by all random-placing-based algorithms discussed here – the bounding boxes are not relevant, in contrast to LP. The collisions of the whole outer molecular, irregular shape have to be computed. For this reason, all random-placing-based algorithms previously introduced have to deal with geometrical as well as combinatorial heterogeneous problems. A large number of packing algorithms take advantage of the assignment of many different items to a few types of differing shapes [Dyck90]. Because different instances of the same lipid type feature all the same shape, this also applies to membranes in MembraneEditor, but only as long as a membrane contains only one single lipid type. However, even in this case the irregularity of shapes leads to the requirement to compute every single collision separately. Therefore the RP is not a pure 2D-KP approach. On the contrary the RP is also not 3D-KP, because this problem class would be appropriate for the description of packing processes in different interrelated layers. An alternative is needed.

Therefore RP and the following algorithms should be considered as 2.5D-KP.

The three properties of a 2.5D-KP are as follows:

1. the movement is restricted to a single two-dimensional container,
2. the shape of the items is three-dimensional and
3. irregular.

The approach of using a half dimension as an intermediate step in special approaches is not new. Dyckhoff introduced a one-and-a-half-dimensional packing problem describing the combination of a one-dimensional problem with continuous measurements [DyKr85]. Packing approaches differing from the KP sometimes also feature two-and-a-half-dimensional definitions. Cagan, Shimada, and Yin introduced such an approach which describes the usage of different consistent layers, where the items of different layers are not packed on top of each other [CaSY02]. But in the knapsack context, this type of packing is already subsumed in the definition of the MKP (see Section 3.2.6).

In Section 4.3.2 a quite different application of a two-and-a-half-dimensional approach was discussed in the context of 2.5D network modeling tools.

5.2.2.6 A high-density-generating Plugin Algorithm: The Wanderer

Meets Requirement R2.VI

The Wanderer (TW) is a plugin-algorithm (next Section 5.2.2.7) which is – in contrast to previously described algorithms – not part of the regular MembraneEditor release (as of version 2.2.1_2). It was developed by the author of this work to generate high lipid densities within an appropriate time frame.

Figure 100 shows a very simplified notation of TW. The main method – Lines 9 to 34 – runs until the user stops the process or until it turns out that the last lipid cannot be placed. This is the case if the last row – row_{last} – is reached (Line 27) and the placing attempts *pa* reached the maximum (pa_{max} in Line 26). The placing starts with the new lipid l_{new} in the first row (Line 7) and iterates each time the row_{act} (Line 30) if the pa_{max} was reached and the lipid is not wandering around row_{last} (Line 26). Moreover, it can be seen in Line 31 that col_{act} is reset to 1. This variable is usually used in Line 21 to save the actual X-position of the last placed lipid if the actual computed position is valid. If the position is not valid – for example, this is the case, if l_{new} intersects the membrane borders or intersects with other molecules – it is removed, ignoring the actual position in Line 23. In Line 10 l_{new} is generated based on row_{act} and col_{act}. The latter variable is used as a rough starting point to place l_{new} at the position occupied by the previously placed lipid.

But how is the placement of the lipids computed? Obviously, it will usually happen that l_{new} is placed on top of the previously placed lipid and they are intersecting. In addition, it may happen that the lipid is placed too close to an outer membrane border. Now the Lines 11 and 12 cause l_{new} to wander for freedom. First it moves to the eastern border. Immediately if l_{new} has reached a vacant place, the process *wanderForFreedomEast()* finishes. The next function *wanderForFreedomSouth()* in Line 12 causes now l_{new} to wander south. But this function is only important in case the previous wanderForFreedomEast() function is not successful. This may happen if l_{new} is trapped between a previously placed lipid and the eastern border. In this case, it just wanders with the wanderForFreedomSouth() function to the south and will stay there at the next free position.

With this functionality it is already possible to generate a membrane layer filled with lipids. But of course, this is not an optimal placement, because it is a quite linear methodology. Sometimes there might be holes in the membrane or complete rows might be skipped, because of some unfavorable constellations.

```
01  placing attempts   pa;
02  maximal placing attempts   pa_max;
03  actual row to place lipid   row_act;
04  actual column to place lipid   col_act;
05  last row∈membrane   row_last;
06  new lipid to place   l_new;
07  row_act:=1;
08  col_act:=1;

09  while(user_did_not_abort()) do {
10        l_new:=generateNewLipid(row_act, col_act);
11        l_new. pos :=wanderForFreedomEast();
12        l_new. pos:=wanderForFreedomSouth();
13        l_new. backup:=l_new. pos;
14        if(wanderForFellow) do{
15              l_new. pos:=wanderForFellowWest();
16              l_new. pos:=wanderForFellowNorth();
17              if(wanderDancing) do l_new. pos:=wanderForOptimalRotationWestNorth();
18              if(l_new. pos.isNotValid()) do l_new. pos:=l_new. backup;
19        }
20        if(l_new. pos.isValid()) do {
21              col_act:=l_new. pos.x;
22        }else do {
23              l_new. remove();
24        }
25        pa:=pa+1;
26        if(pa≥pa_max){
27              if(row_act=row_last) do {
28                    break();
29              }else do {
30                    row_act:=row_act+1;
31                    col_act:=1;
32              }
33        }
34  }
```

Figure 100: The Wanderer algorithm strongly simplified

Therefore the optional *wanderForFellow* functions were implemented which are activated in Line 14. After l_{new} was placed on a free position, the following two optimization functions in Lines 15 and 16 lead l_{new} into the opposite directions: first west (Line 15) and then north (Line 16). This means, the lipid movement is stopped as soon as it touches another lipid or a border.

The last optimization step is initiated if the wanderForFellow process has finished and *wanderDancing* is activated. A lipid is dancing, if it turns along its Y-axis and tries to come even closer to the neighboring molecule by wandering west-north. Finally, if there are

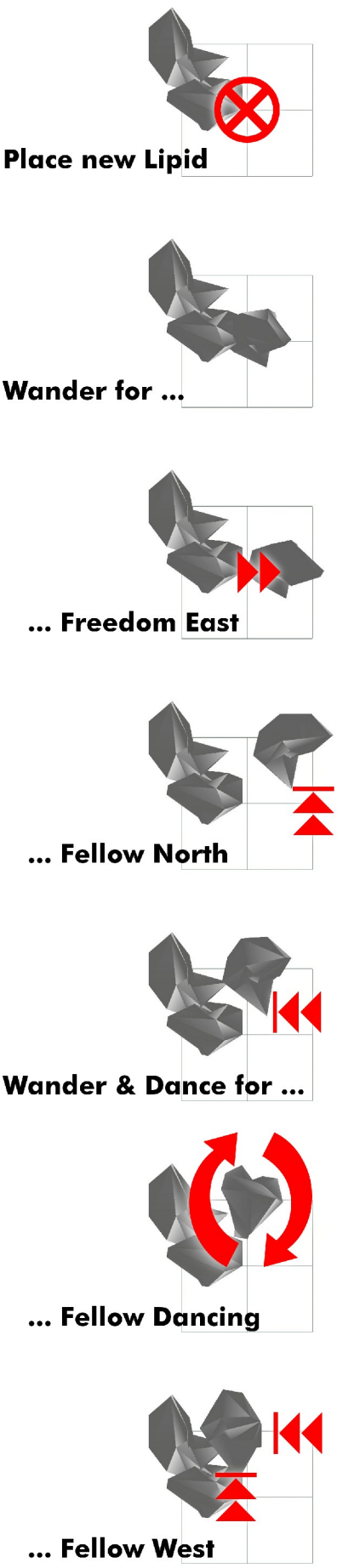


Figure 101: The Wanderer Process

problems with the optimization – the wanderForFellow – process, the lipid position is reset in Line 18 by restoring the old position stored in l_{new}.backup in Line 13.

Figure 101 shows the typical procedure of a wandering run. In this case, the starting point is a membrane which contains two lipids already placed by TW. First, a new lipid is generated and placed (see also Figure 100, Line 10). Obviously this new lipid is positioned on top of the previously placed lipid – they are intersecting. Now the wandering process starts. The new lipid moves east for a free position (Line 11). Next, it would move south (Line 12) if no free position is found. But because the lipid already found a free position, this method is skipped. Now the lipid should move west to find a fellow (Line 15). But again, this procedure is not needed in this case, because the lipid is already positioned directly besides the previously placed lipid (because the lipid did not wander south). Therefore, the next method is invoked, causing the lipid searching for a fellow in the north (Line 16). The lipid stops when meeting the border in the north. Now the final process is started: the lipid wanders and dances (Line 17). The rotation process is associated with a movement to the west and north. The latter cardinal direction does not cause a visual difference in Figure 101, because the lipid already touches the northern border. The process stops when the new lipid moved very close to the lipid located in the north-western corner of the layer.

Combining these methodologies, a quite dense packing is possible as will be shown in Section 6.2.1.1. Obviously it is also a greedy algorithm. Its characteristic of placing lipids row by row is similar to the Linear Placing algorithm, but the difference is, of course, the subsequent correction/optimization of the placing process. Therefore this algorithm belongs again to the OPT_LPP_MIN_AREA class.

5.2.2.7 The Plugin-Interface

Meets Requirement R2.XIV

The algorithm The Wanderer was introduced as a plugin-algorithm. But what exactly is a plugin in the context of MembraneEditor?

At the beginning of the development of MembraneEditor it was planned to provide users with the possibility of using

custom algorithms with this software package [Schn08]. For this purpose, the plugin-interfaces was integrated. This interface enables the import of algorithms during runtime. Because of this property users are able to change their custom algorithms and test them in MembraneEditor without the need to restart the program. In addition, it is not necessary to understand the complex structure of MembraneEditor. The users can just focus on the documentation of the plugin-interface. Using the stand-alone version of MembraneEditor enables the direct integration of JAVA files. Moreover, precompiled JAR files may be imported. The web-start-version of MembraneEditor is restricted to the import of precompiled JAR file for technical reasons.

The plugin-interface enables the access to all required methods to manipulate lipids and proteins including, for example, their identifiers, positions, orientations, membrane-layers, microdomains and single atoms.

The user needs a single jar called MembranePackingAlgorithm.jar which can be downloaded from the website http://Cm2.CELLmicrocosmos.org. The included MembranePackingAlgorithm class has to be extended by a new JAVA class which implements the corresponding methods. These methods can be used for integration with custom algorithms. In Appendix 9.7 the source code of a very simple MembranePackingAlgorithm is shown.

This work also discusses a number of MPA plugin-algorithms and -tools for application cases in Section 6.2.2: the *Molecule Boxifier* and the *Dimension Lister*.

5.2.2.8 Seed-based random number generation

Meets Requirement R2.XV

For the professional generation of membranes it is inevitable to provide users the possibility to reproduce their results. Therefore MembraneEditor was improved by adding seed-based random number generation. For each algorithm run a seed can be defined. Using this seed in conjunction with the same lipid models, lipid densities and the same properties, the computational production of a membrane model can be repeated.

5.2.2.9 Reverse-Parsing

Meets Requirement R2.XIII

As mentioned before, MembraneEditor is able to export membrane models to the PDB format. The exported membranes can be used for further analysis or recalculation with other programs like MD simulation programs. As long as the quantitative structure of a membrane is not changed[59], it can be re-imported into MembraneEditor. For this purpose, the original XML structure must be accessible to MembraneEditor. There are two options to ensure this:

1. The XML/Cm2 file is saved separately to enable this file as the base for the reverse-parsing process.

59 This means, the number of lipids as well as proteins may not be changed. But the position as well as the atomic conformation of the different molecules may differ.

2. The XML/Cm2 file is saved in the REMARK section of the PDB file (Appendix 9.6). Of course, the external program which edits the PDB file must preserve the REMARK section; otherwise the information will be lost. In this case, method 1 can be used.

After the membrane model is re-imported into MembraneEditor, all changes regarding the atomic structure are visualized when using the atom visualization modes. But the shape-based visualization for single molecules will not be changed: molecules of the same type will always have the same shape. For this reason it is not reasonable to use the shape collision mode if the reimported membrane model should be used for further computations. Only the atom collision mode should be used for this purpose (Section 5.2.2.3).

5.2.2.10 Lipid Packing Density and Statistics

Meets Requirement R2.XI

As mentioned before, an appropriate property to judge the quality of the packing process is the lipid packing density, D_l (Section 3.2.7). A discussion regarding the large variety of computational methods to measure the area per lipid is found at Alwarawrah, Dai and Huang [AlDH10].

In this work, a simple method is used: D_l is computed by dividing the layer area by the total number of lipids (average lipid density). The according number is permanently shown in the properties window.

In addition, MembraneEditor provides different statistics to analyze the packing quality:

- the quantitative ratio for lipids (#),
- the absolute number of lipids (|#|),
- the atomic mass ratio of lipids and proteins (u) and
- the absolute atomic mass of lipids and proteins (|u|).

Each of these statistics can be visualized for the extracellular and/or intracellular membrane layer.

5.2.3 Implementation Details

The CELLmicrocosmos 2.2 MembraneEditor (CmME) was originally developed in Java 5. Now it is a Java 6 application. For 3D visualization the Java 3D version 1.5.1 library was used. It is available as a stand-alone application for downloading and as a Web Start application to start directly from the browser. The whole project, the documentation, the forum, links to the publications and more is found at:

http://Cm2.CELLmicrocosmos.org

The software package is released on GNU General Public License/GPL version 3. It was officially released since version 2.2.1.

Please refer to the Appendix 9.17 for information regarding the implementation work and to Appendix 9.15 for the version information. The included third-party libraries and their licenses are shown in Table 12.

Name	Version	Usage	License	More Info
Infonode Docking Windows	1.6.1	Manages the different windows in the main GUI	GPL	http://www.infonode.net
j3d-core, vecmath	1.5.2	Displaying and picking of the 3D objects in the Membrane View	GPL 2 (with CLASSPATH exception)	http://www.j3d.org
j3d-core-utils	1.5.2	Picking and Navigation	BSD (without advertising clause)	http://www.j3d.org
JDOM	1.1.1	Loading and saving of XML files	Apache License (without acknowledgment clause)	http://www.jdom.org
JFreeChart JCommon	1.0.13 1.0.16	Visualization of the resulting percentaged lipid distributions and lipid/protein weights	LGPL 2.1 (or later)	http://www.jfree.org/jfreechart/
Jmol	12.0.40	External previewing of the membrane or single molecules	LGPL 2.1 (or later)	http://jmol.sourceforge.net
Substance	4.0	The L&F of the GUI	BSD (without advertising clause)	http://substance.dev.java.net
Tools	from JDK 1.6.0_24	Compiling of java files	GPL 2 (with linking exception)	http://openjdk.java.net

Table 12: Program packages included in the CELLmicrocosmos 2.2.1 MembraneEditor

5.3 [FUN] Functional Modeling: Integrating Metabolic Networks into a Virtual Cell

In Subchapter 5.1 the methods and implementation of the mesoscopic level were discussed. This was needed to create the Virtual Cell environment which will be correlated with the functional level in this subchapter. The functional level is represented in this work by protein-related pathways, and more precisely, by metabolic networks. The biological basics were already discussed in Subchapter 2.3. Many preconditions which will be important in this subchapter were found in the technical basics: the data sources (Section 3.3.2) as well as the Network Mapping Problem (Section 3.3.4).

This subchapter will now cover all aspects which have been implemented in the CELLmicrocosmos 4.2 PathwayIntegration. First, the requirements will be listed (Section 5.3.1), followed by the methods addressing the identified aspects (Section 5.3.2) and finishing with a short overview of the implementation (Section 5.3.3).

5.3.1 Requirements

R3.I Import of Cell Models

The cell models known from CellEditor must be imported to allow the correlation with networks (Section 5.1.2.2).

R3.II Access to a Data Warehouse

Data warehouses represent an appropriate collection of the information needed for proteomic localization (Section 3.3.3.1). Access to such an approach should be provided by the application.

R3.III Import of Network Structures

Due to the fact that this work does not aim at the generation of another conventional Bioinformatics network reconstruction and analysis tool, there must be a possibility to import data containing networks generated with external applications like those discussed in Section 4.3.1.

R3.IV Differentiation of Localization Terms

It must be possible to subdivide different localization terms which might reflect a cell component, its special segments or multiple different component types (Section 5.1.2.4).

R3.V Import of Localization Data

It should be possible to import localization data into the application directly (Section 5.3.2.3.2).

R3.VI Access to Localization References

Localizations are normally associated with one or multiple publications where the evidence was reported. There should be an Internet link – where possible – to the according publication(s).

R3.VII Localization Visualization

An appropriate, quantitative and qualitative visualization of the localization is needed.

R3.VIII Combination of different Networks

In the light of cell biology which is described by a large number of different protein-related networks it is inevitable to enable the import and combination of different networks.

R3.IX Coloring Methods for the Network

Different networks have to be visually distinguishable. But human color perception is limited to a number of colors, especially in combination with a three-dimensional visualization. Therefore appropriate coloring methods have to be provided.

R3.X Coloring Methods for the Localizations

Besides the visual differentiation of networks, it should be also possible to support the localization elaboration by colors (as also shown in GEOMI Section 4.3.3.3).

R3.XI Node Distribution Algorithms

Two-dimensional as well as three-dimensional visualization of networks is normally always associated with different layout algorithms. Therefore, a number of layout algorithms are needed which help to improve their representation (Section 3.3.4.2).

R3.XII Node Mapping Functions

In Subchapter 5.1 the generation of a Virtual Cell was discussed. This three-dimensional construction has to be associated with the localized network. For this purpose, node mapping functions are needed, placing the nodes at their temporary or final destination (Section 3.3.4.3).

R3.XIII Interactivity

The complexity of the proteomic localization of networks in a cell has to be explorable by the user. A static representation is normally only useful if the according amount of data is very small or a subset of a large network is shown. Therefore the application has to provide a possibility for user interaction (Section 4.3.3.4).

R3.XIV 3D Navigation

In contrast to a two-dimensional navigation which was perfected over a decade ago by tools such as Adobe® Photoshop®, the movement in three-dimensional space is a quite complex task to take. A Virtual Cell environment correlated with a network needs a specialized navigation.

R3.XV Shading for the Network

The spatial structure of a network in three-dimensional space is emphasized by shading effect.

R3.XVI 3D Stereoscopy

Whereas the scientific community has been using 3D Stereoscopy since more than a decade, the advent of 3D Stereoscopy techniques in the living rooms of today's media recipients started around 2010. These techniques should be also applicable to the Virtual Cell associated with a network (see also Sections 4.1.1.2 and 4.3.3.2).

R3.XVII Desktop and VR-ready

The application has to be compatible to general notebooks, desktop PCs, but also with powerful visualization environments like powerwalls.

R3.XVIII Reproducibility

Results should be reproducible.

5.3.2 Methods

Figure 102 shows the modeling pipeline at the functional level which will be discussed in the following sections. The complete modeling pipeline is illustrated in Figure 7.

5.3.2.1 Cell Modeling as Basis

Meets Requirement R1.I

Prior to correlating a Virtual Cell with a biological network, the cell model has to be created or imported. The different methods were already discussed in detail in Subchapter 5.1. Therefore the focus of the following sections will be on the creation and localization of the biological network.

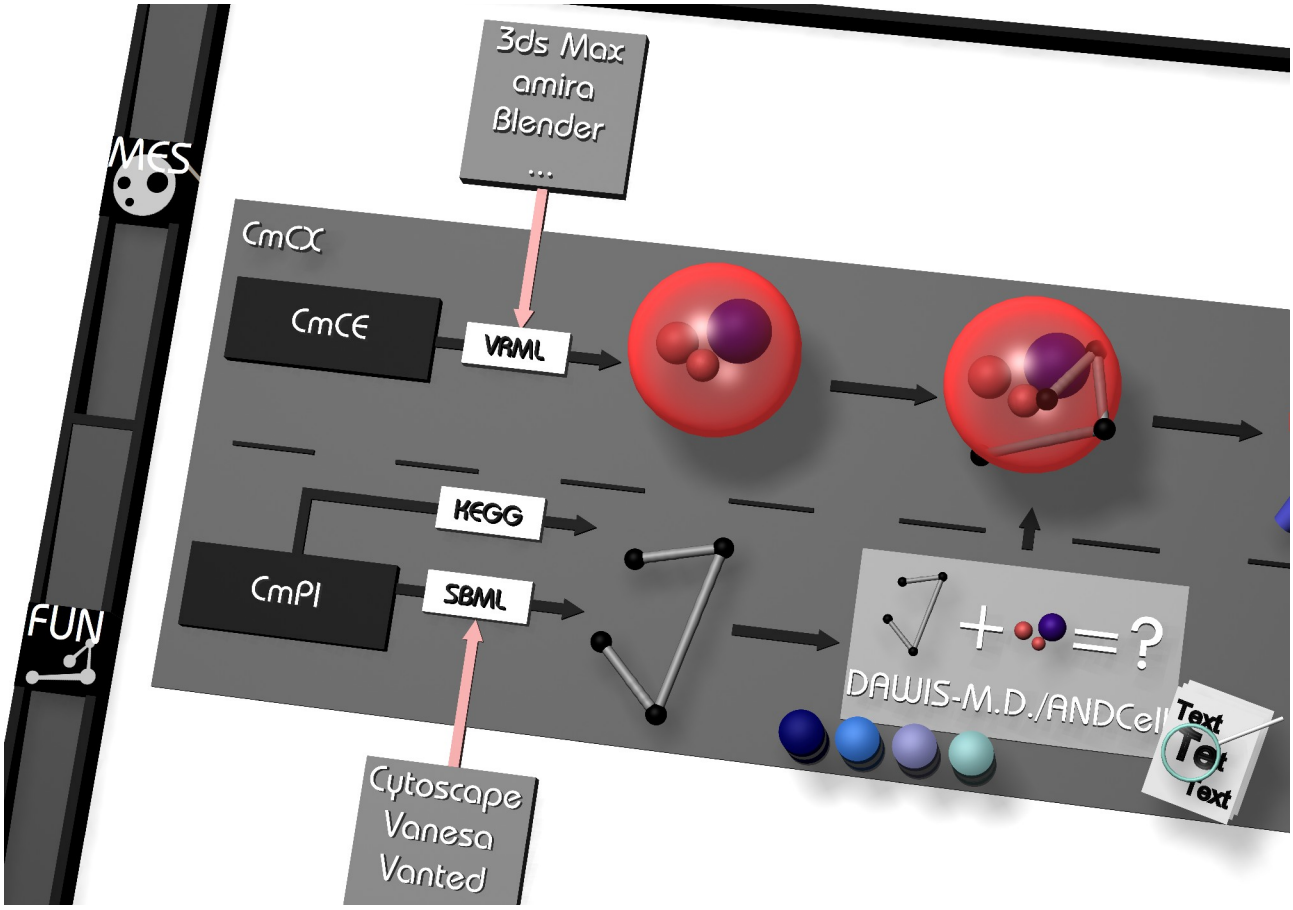


Figure 102: The CELLmicrocosmos modeling pipeline at the functional level

5.3.2.2 Network Reconstruction

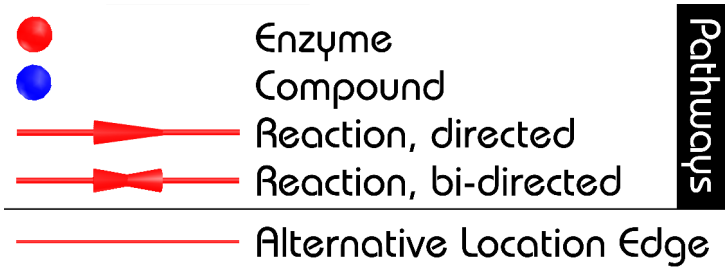


Figure 103: The graphical elements in 3D of CmPI

5.3.2.2.1 Metabolic Pathways from KEGG

The initial idea of CmPI was the localization of metabolic pathways. The base for the pathway generation in this approach is the KEGG database discussed in Section 3.3.2.1.1. Those pathways can directly be loaded from the data warehouse DAWIS-M.D. - introduced in Section 3.3.3.1 – into CmPI. Figure 104 shows the initial stage of CmPI: in the first step, the organism is chosen in the *Pathway Download and Creation Section*, here: “Homo sapiens (hsa)”. In the second step, the KEGG pathway is chosen, in this case the “citrate cycle (00020)”. By pressing the button “KEGG Pathway” it is downloaded from DAWIS-M.D. and afterwards appears in the *Pathway Table*.

The network structure and visualization is quite intuitive:

- *enzymes:* red-colored spheres,
- *compounds (products and substrates):* blue-colored spheres, and
- *reactions:* the reactions are simple cylinder-shaped lines, the reaction direction is marked by a cone (directed) or two cones (bi-directed) in the center of the line.

Figure 103 shows the different graphical elements used to visualize a (metabolic) pathway. The coloring of the edges indicates the associated pathway and will be discussed in Section 5.3.2.8.[60]

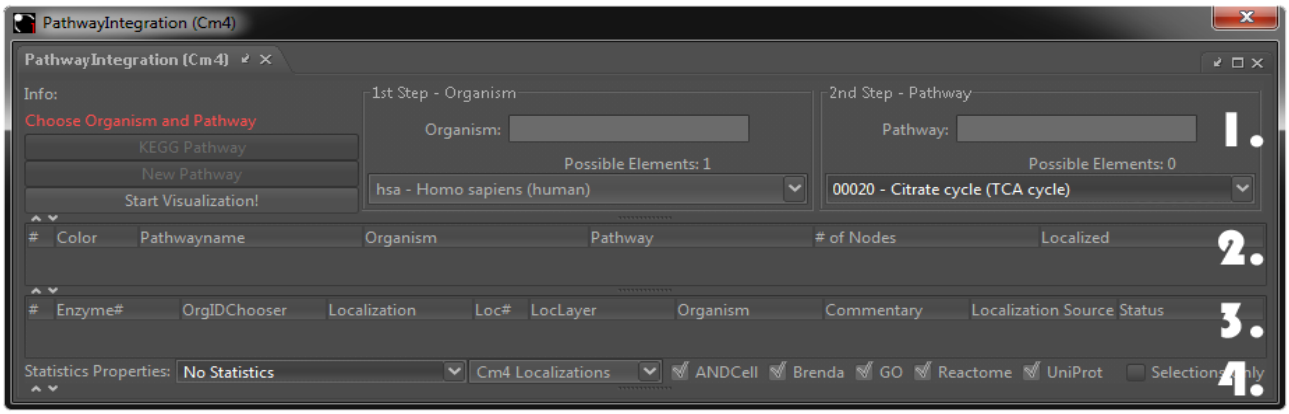


Figure 104: The Main Window of CmPI
1. The Pathway Download and Creation Section; 2. The Pathway Table; 3. The Localization Table; 4. The Localization Charts[61]

5.3.2.2.2 Pathway Import via SBML/AND

CmPI was extended by import capabilities. Therefore protein-related data like protein-protein interaction networks, protein signaling or regulatory networks can also be loaded into the application.

For this purpose, two XML-based formats are supported. First, the *Systems Biology Markup Language (SBML)* is based on XML [HFSB03]. A SBML parser was implemented, supporting SBML Level 2 Version 3 and 4 (.sbml-extension) [Özgü10]. Many applications are compatible by providing this format, for example: Vanesa [JKTH10], Carmen [SVTB10] or the Reactome web site. Secondly, the import of associative networks generated with ANDVisio (.and-extension) is also implemented (Section 3.3.3.2).

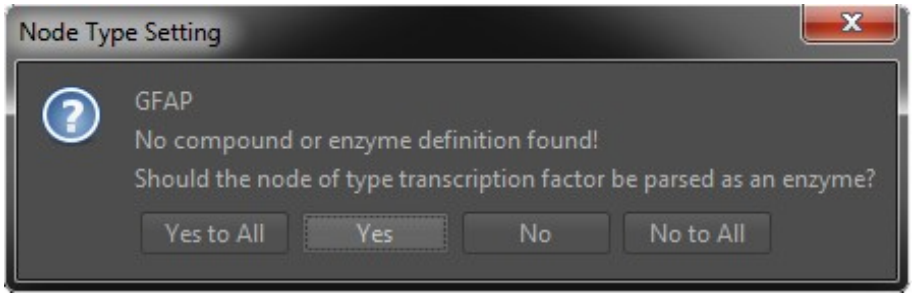


Figure 105: Node Type Setting in CmPI
The user can choose if an imported node should be defined as an enzyme

In some cases it is not possible for CmPI to verify whether an imported element is an enzyme. For example, ANDVisio defines the types of nodes when storing a pathway. Therefore, CmPI can automatically assign the type to loaded elements. But most SBML-exports like those from VANESA do not define the type of node. In these cases, the user has to choose the correct node type (Figure 105). However, CmPI also tries to automatically typify a protein node by using the EC-format (N.N.N.N, where N is a number, see Section 3.3.2.2.2).

60 The “Alternative Location Edge” will be discussed in Section 5.3.2.7.
61 The subsequent state of CmPI – after downloading and localizing the selected pathway hsa00020 – is discussed in the Application Section 6.3.1.1.

5.3.2.2.3 2D Viewer

Meets Requirements R3.III,XIII

In order to analyze the networks before mapping them to the cell model, a very simple 2D Viewer was integrated. It can be used to add, remove or rename nodes. Also new edges can be drawn. Of course, this tool cannot/is not intended to compete with 2D network editors introduced in Section 4.3.1. For this purpose, the previously mentioned import capabilities should be used.

The main intention of the 2D Viewer is to provide an additional overview of the currently selected pathway. This overview can also be used to navigate through the cell environment by double-clicking onto nodes of interest. It also follows the focus+context paradigm introduced by the related approach in Section 4.3.3.1; during the exploration of the network in 3D, for example, if two interacting nodes are focused, the 2D Viewer can be used to keep the context – the complete network – visible.

5.3.2.3 Protein Localization

Meets Requirements R3.II,IV,V,VI,VII,XIII

5.3.2.3.1 The Databases

Figure 106: Color codes used for different localization databases accessible by CmPI

In Section 3.3.2.2 the different databases used for the localization were introduced: Reactome, BRENDA, UniProt, Gene Ontology, which are contained in DAWIS-M.D., and the separate ANDCell database. Figure 106 shows the color coding of the localizations based on the coloring of the logos. The characteristics of these databases were discussed in Section 3.3.2.2.

Only those pathway elements previously defined as proteins can be localized. If a KEGG pathway is downloaded, the localization process starts immediately afterwards. In the other case, if the pathway is imported from an XML-file (Section 5.3.2.2.2), the user has to start the process from the pathway table.

5.3.2.3.2 Localization Import via AND

Because ANDVisio (Section 3.3.3.2) is able to create networks containing protein-localization associations, CmPI can extract the localization information from an AND-file. This was particularly an important feature during the beginning of the project when it was not possible to directly access the ANDCell database.

5.3.2.3.3 The Mapping Table

The problem of synonyms for localization was first introduced when the need for an appropriate vocabulary was discussed as the initial idea for The Gene Ontology database (Section 3.3.2.2.4). Especially when querying different databases for localizations, this problem persists. Therefore, CmPI uses an internal mapping table to assign appropriate localizations.

An important base to enable sub-sub-cellular localization of proteins is the fact that CmCX features multi-layered cell components (Section 5.1.2.4). Figure 90 showed the different layers of a mitochondrion model. Table 13 shows exemplarily three *Localization Terms* mapped to each cell component layer of a mitochondrion. A Localization Term is a word, a sentence fragment or a complete sentence acquired from the database. Obviously it is not possible to define an exact localization for each of these terms. For example, the term "mitochondria" is unspecific, but it of course matches a cell component. The Localization Term "mitochondrion" is eponymous to with the *Cm4 Localization* which is a unique cell component in the context of CmPI. In this case the term "mitochondrion" is mapped onto the mitochondrial default layer which is the outer mitochondrial membrane. A membrane layer inside a cell component is defined as a distinct *Cm4 Membrane Localization*.

Cm4 Localization	Cm4 Membrane Localization	Localization Term
Mitochondrion	Matrix	mitochondrial matrix
		mitochondrion matrix
		protein import into mitochondrial matrix
	Inner Membrane	intrinsic to mitochondrial inner membrane
		mitochondrial inner membrane
		mitochondrial respiratory chain complex 1
	Intermembrane Space	mitochondrial intermembrane space
		mitochondrial lumen
		protein import into mitochondrial intermemb...
	Outer Membrane	mitochondrial outer membrane
		mitochondrion (!)
		mitochondrion localization
	Cloud	mitochondrial cloud
		mitochondrial envelope
		protein targeting to mitochondrion

Table 13: Localization Terms mapped to distinct Cm4 Cell Component Layers

During the parsing process it often happens that still not all Localization Terms are included in the Mapping Table. In this case, the unknown terms are listed in the console of CmPI. The user can analyze this list and extend the Mapping Table. For these terms the *Unknown* cell component exists: all protein nodes which were not localized are mapped onto this cell component.

5.3.2.3.4 The Localization Table and Localization Referencing

Figure 104 shows the different elements in the main window of CmPI. The *Localization Table* shows all proteins in the pathway table and their potential localizations as well as the selected one. The last mentioned selection is those localization which will be visualized in the cellular environment.

A very important feature is the listing of Localization references. If a Localization is found in the database of an .AND-file, the results are combined with information about the sources. These sources depend highly on the providing database.

In case of ANDCell, the Localization Source contains the text fragment the Localization is based on together with the link to the underlying PubMed entry using the website: “http://www.ncbi.nlm.nih.gov/pubmed/”.

Full publication information like the authors and the title of the work as well as – if available – the link to the PubMed abstract (one Localization is often based on multiple reference information) is provided by BRENDA.

Results of the GO database contain the link to the correlated GO-term, using the GO-associated website of the EBI: ”http://www.ebi.ac.uk/QuickGO/“.

In case the ID of Reactome is given, a link to the original Reactome website is provided, directly accessing the underlying Localization Term in the pathway visualization of the website “http://www.reactome.org”.

In case a full UniProt-KB entry is shown, a full publication list – similar to BRENDA – is parsed. But this might also be based on a UniProt-Annotation. In this case the link to the UniProt on-line resource is provided: ”http://www.uniprot.org”.

Additional information regarding the proteins as well as compounds can be derived from links to the KEGG database in case a KEGG pathway was loaded: “http://www.genome.jp”.

5.3.2.3.5 The Localization Charts

The last row in the main window of CmPI (Figure 104) shows the *(Subcellular) Localization Charts*. This element can be used to visualize, analyze and assign localizations to a single protein, to a set of selected proteins as well as to complete pathways.

There are five different types of Localization Charts thus far, which can be selected by the corresponding drop-down list:

- *Localizations:* This category sums up all potential localizations found for the selected protein(s). It is important to mention that a database might contain multiple hits for the same localization. This is the fact if different localization terms for the same localization were found.
- *Localizations: colored by database:* Using the colors of Figure 106, the chart shows the database hosting the acquired information used for the localization mapping. Therefore this category might be taken into account during the analysis of the data quality.
- *Proteins: ordered by proteins:* An overview of all selected protein(s) colored by localizations, sorted by protein names. A detailed discussion of the applied color codes is given in Section 5.1.2.3. For the coloring in this chart the same colors are used as for the SphereCell Components.
- *Proteins: ordered by pathways:* An overview of all proteins colored by localizations, sorted by pathways. This type of sorting is reasonable if multiple pathways were loaded into CmPI. Otherwise, the sorting will be the same as using the category “Proteins: ordered by proteins”.

- *Protein Co-Localizations:* This chart uses a nearest-neighbor-search method to determine similar localizations of adjacent proteins. The different localization sources are summed up. This category is also subdivided into *ordered by proteins* and *ordered by pathways*.

A second drop-down box on the right side of the chart's category chooser can be used to change between the previously discussed three localization categories:

- *Cm4 Localizations*,
- *Cm4 Membrane Localizations* and
- *Localization Terms*.

Therefore, there are 18 (6X3) potential combinations.

In addition, it is possible to filter the results by database. Only the selected databases are taken into account. Moreover, there is an option to show only the actually chosen localizations of the selected protein(s). This option can be used to visualize the actual state of localizations.

The Localization Charts are interactive. If the user clicks on a statistic bar, all interrelated rows in the Localization Table are selected. And if the user makes a double-click on this bar, the localization actually chosen is assigned to all selected nodes. Therefore it is possible to assign a localization to a set of proteins in a fast and convenient way.

Obviously, the Localization Charts dynamically affect the Localization Table. But this is also the fact in the opposite direction. Selecting a pathway or multiple pathway(s) in the Pathway Table, or selecting a single protein or multiple ones in the Localization Table updates the charts accordingly. For the selection of multiple elements, the common mouse/keyboard combinations of “left mouse click” and “Shift” or “Ctrl” are used.

5.3.2.4 Correlating Networks with Cell Components

After discussing the localization of proteins, the question arises how the proteins are correlated with the cell components. In Section 3.3.4 this problem was defined as the Network Mapping Problem.

Figure 107 illustrates the two basic process of placing nodes in the cell environment focusing at a mitochondrion. First, the nodes are distributed with a layout algorithm (which will be introduced in the following sections). The layout process is usually based on a simple shape – a sphere – as shown in Figure 107.1. The layout algorithms first distribute the nodes onto the surface of the sphere based on specific rules. And then in the second step, many algorithms redraw the network based on the connecting edges between the nodes. The distance from the center of the sphere to the node on the surface of the sphere is therefore equal. But as discussed before, most cell components contain different irregular shapes. Therefore it is not sufficient to place the node only at a distinct distance from the cell components center. The solution will be discussed in the following sections.

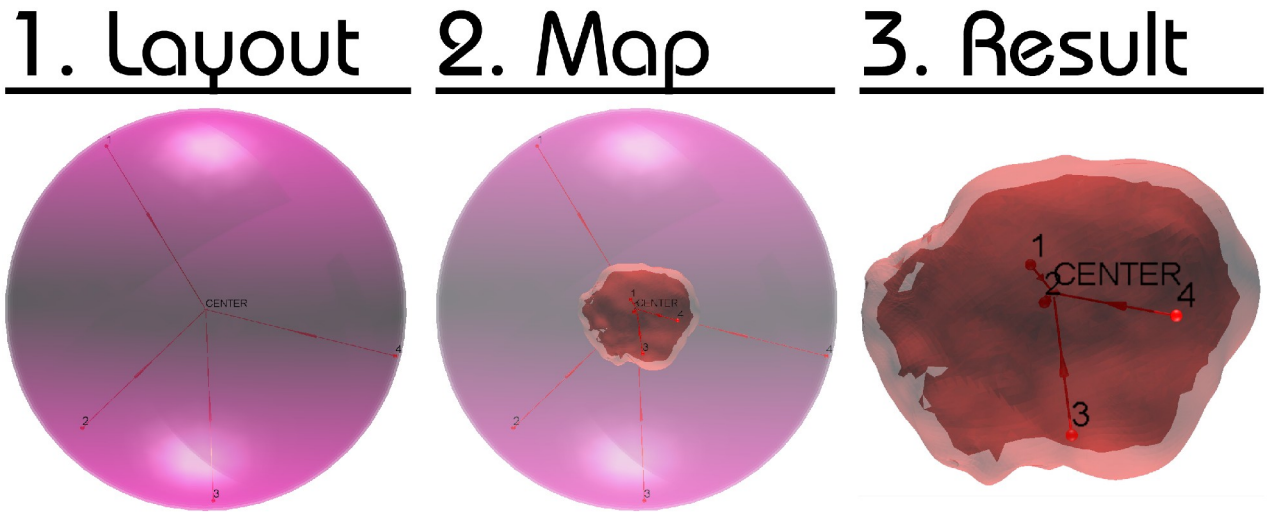


Figure 107: The Node Mapping Process

5.3.2.5 Network Layouts

Meets Requirements R3.XI,XVIII

In the preceding section the layout process was already mentioned. Now two different layouts are introduced sequentially. It will be shown that the second algorithm – the ISOM layout (Section 5.3.2.5.2) – uses the following approach for the initial node placement.[62]

5.3.2.5.1 UUUSphere

First, the focus should be laid on the nodes placement, ignoring the connecting characteristics of the edges. This optimization problem was already defined in Section 3.3.4.1 as OPT_NMP_MAX_AREA.

The *UUUSphere* layout features three basic principles. The first "U" abbreviates the *unit sphere*. This is a sphere with a constant radius of 1 which is the base for the solution for many trivial as well as complex mathematical problems. The second "U" is an abbreviation for *unique random numbers*. Their impact on the reproducibility of scientific computations was already discussed in the context of CmME (Section 5.2.2.8). Finally, the third "U" stands for *uniform distribution* and will be explained in the following paragraphs: the *Uniform Unique Unit Sphere layout.*

The surface of a sphere consists of an unlimited number of points, vectors which all have the same length starting from the origin of the sphere. The problem is now to find discrete points which can be taken into account for solving OPT_NMP_MAX_AREA.

Figure 109 shows two different options available for the UUUSphere layout. The first approach is based on *Rakhmanov, Saff and Zhou (RS&Z)* and lays a virtual spiral onto the surface of a sphere (Figures 109.1a/b). This spiral is the path along which the nodes are distributed. For this purpose a generalization of spiral points is used [RaSZ94].

Figure 108 shows the Java/Java 3D implementation found in the UUUSphere layout. The *nodeNumber* in Figure 108 is the actual number of the node to place. The *nodeNumberMax*

62 CmPI also contains additional network layouts. This section focuses only on layouts implemented by the author.

is the maximal number of nodes to place onto the sphere. Obviously, this method is invoked for every node to place and unique positions are applied. The constant *3.6* is a value derived by numerical experiments by Rakhmanov, Saff and Zhou, creating a nearly regular hexagonal spherical net.[63]

```
01 private Vector3f getUniformVector(int nodeNumber, int nodeNumberMax) {
02    double s  = 3.6/Math.sqrt(nodeNumberMax);
03    double dz = 2.0/nodeNumberMax;
04    double x = 0;
05    double y = 0;
06    double z = (1 - dz/2) - nodeNumber * dz;
07    double r = Math.sqrt(1 - z * z);
08    double longitude = nodeNumber * s / r;
09    x = Math.cos(longitude) * r;
10    y = Math.sin(longitude) * r;
11    return new Vector3f((float) x,(float) y,(float) z);
12 }
```

Figure 108: The UUUSphere Layout Approach based on Rakhmanov, Saff and Zhou

The geometrical interpretation is discussed in Saff and Kujlaars [SaKu97]. The sphere is cut into *nodeNumberMax* horizontal circles of latitude. This variable is one of three components related to polar coordinates. The second component is very similar to the longitude of a sphere and the third one is the radius. Because the base sphere used during this computation is a unit sphere with a radius of 1, the radius is implicitly multiplied (Elenzil 2012; Bourke 1996).

Between a neighboring pair of circles – along the longitude – there is always a distance of 2/nodeNumberMax. Each latitude contains exactly one spiral point and the first and the last circle consist exclusively of one node, the north pole and the south pole. In Figure 109.1a, the south pole is Node 0, and the north pole is Node 20. By combining the previously stated information with the path shown in Figure 109.1a, the methodology of the algorithm becomes clear: starting at the south pole, the algorithm proceeds to the next circle of latitude, traveling along its meridian – the outer circle – for the exact distance given by the constant geodesic *s*:

$$s = \frac{3.6}{\sqrt{nodeNumberMax}} \tag{19}$$

The term *geodesic* should be already known to the reader. It was used in the Introduction (Subchapter 1.1) where the Geodesic Domes as well as the discovery of the C_{60} buckminsterfullerene were mentioned. But what are the geometrical implications of this term? First, the term should be defined by citing the "Buckminster Fuller Reader":

"The shortest distance between two points on a curved surface. A geodesic on a sphere is an arc of a great circle, a great circle being a circle on a sphere whose plane passes through the center of the sphere." [Full70, p.369]

63 The interested reader may consult the mentioned publication for an extensive discussion of this topic and the underlying theorems [RaSZ94]. The implementation is partly based on an article in the "CGAfaq" [Comp11].

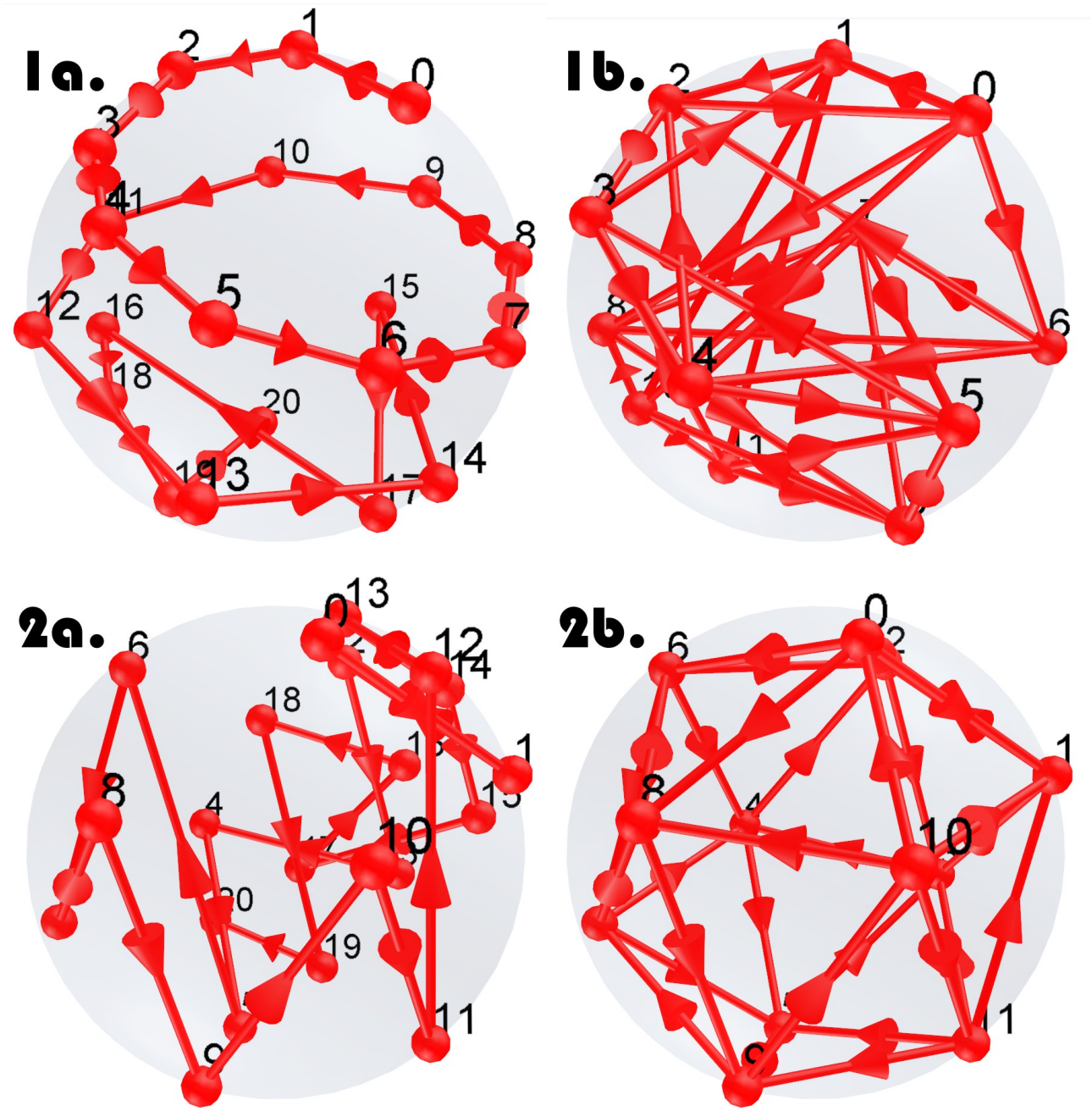


Figure 109: Two uniform distribution layouts used in the UUUSphere Layout 1. Rakhmanov, Saff and Zhou; 2. Geodesic; a. 21 nodes iteratively connected; b. 12 interconnected nodes forming a 1-Icosahedron (Figure 110.2)

Obviously, this term is describing the variable s defined before.

Referring now to Figure 109.2, the second option of the UUUSphere Layout should be explained, which can be understood as a directly reminiscent of the C_{60} buckminsterfullerene. Already the latter structure is a shape approaching the one´s of a sphere. More precisely, it is a *truncated 1-icosahedron*. Figure 110.1 shows the removal of one segment from the 1-icosahedron: one of the characteristic pentagons on the surface of the object is created. To generate a complete C_{60}-buckminsterfullerene-like shape, this step has to be repeated for all 12 vertices.

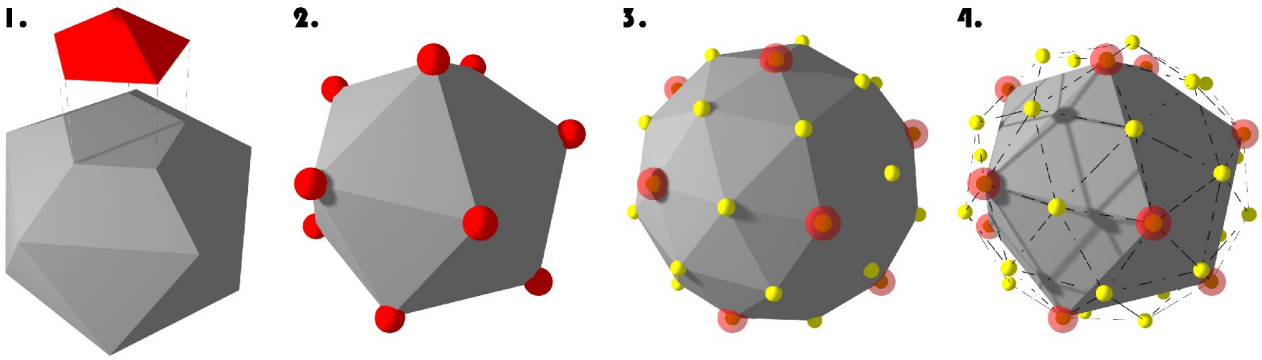


Figure 110: Icosahedra
1. A 1-icosahedron is truncated at one vertex and a C_{60}-characteristic pentagon appears; 2. A 1-icosahedron with one sphere at each vertex; 3. A 2-icosahedron with one sphere at each vertex; 4. A 1-icosahedron overlaid with a 2-icosahedron with one sphere at each vertex point (red: 12) and edge point (green: 42)

But for this layout, the regular 1-icosahedron will be sufficient. Figure 110.2 shows a 1-icosahedron with spheres attaches to each of the 12 vertex points. The advantage in comparison to a sphere is obvious: a number of discrete points already exist which could be applied to the placement of nodes onto a sphere. But using this geometry-based method, it is also possible to use more points as base for the placing process. The 1-icosahedron contains 12 vertices and 20 surfaces. In Figure 110.3, a *2-icosahedron* is shown which already features 42 vertices and 80 faces. The red nodes are placed on the original positions also known from the 1-icosahedron, and the green nodes show the additional available positions. But how does this work?

The *t-icosahedron* always contains $20N^2$ geodesic facets and $10N^2+2$ geodesic vertices where *N* is the tessellation frequency. Examining Figure 110.4, the intention of N emerges: in this case (1-icosahedron to 2-icosahedron) it subdivides each of the (triangular) geodesic facets into four sub-triangles. And these four triangles are defined by six distinct points.[64]

But which of these aspects is now interesting for solving OPT_NMP_MAX_AREA? An important difference from the RS&Z method is the equal distance between each of these vertices. In addition, each of them lies on the surface of a virtual sphere and they could be used as the basis to extrapolate the facets to a sphere (which is indeed done in Section 5.3.2.6). But the main benefit of this approach is the tessellation frequency which can be used to subdivide the surface into an arbitrary number of vertices. And these vertices can be used to position nodes onto the surface of a sphere. And exactly this approach – based on a work by Gregory D. Durgin – is shown in Figure 109.2 [Durg98].[65]

But another important aspect of these optimization approaches should be now depicted by referring to Figure 109. The RS&Z approach obviously shows good results during the generation of sequentially connected nodes generating a linear path (Figure 109.1a). But

64 Another interesting characteristic of the t-icosahedron is the fact that while the vertices of the original, the 1-icosahedron are surrounded by pentagonal patterns, the additional vertices generated by tessellation are surrounded by hexagonal patterns.

65 A profound discussion of the principles of geodesic sphere construction can be found in the stated master thesis of Gregory D. Durgin [Durg98]. The layout shown in Figure 109.4 is partly based on the implementation work of Max Downey [Bour96].

using RS&Z in conjunction with a network-like structure, the result is visually not very convincing (Figure 109.1b). For the Geodesic approach, the opposite is the case. Figure 109.2a shows a path chaotically laid out. All nodes feature a unique position, but the logical order of the connections is hard to recognize. The reason is that this layout does not follow a spiral along the outer circles of the sphere. Instead, it sequentially visits all vertices of each triangular facets and continues then with the neighboring triangle, starting at a distinct top facet and proceeding to a bottom facet. But on the other side, this methodology creates a perfect 1-icosahedron as can be seen in 109.2b.

Obviously, each network configuration needs special algorithms to solve OPT_NMP_MAX_AREA. Of course, usually the identifiers of nodes are not in any relation to the sequence of their interconnections; in contrast with Figure 109.1a. Therefore, the sequential placement of nodes will normally be of no advantage for the placing of nodes with respect to their connecting edges. The improvement of the nodes placement with respect to their connection will be the topic of the next section.

5.3.2.5.2 ISOM

Good solutions for the placing of nodes were found, solving OPT_NMP_MAX_AREA, but still the algorithms implemented in UUUSphere are not sufficient to meet the prerequisites of a full-value layout process. In Section 3.3.4.2 the appropriate optimum definition was given as OPT_NMP_ENERGY.

Now the methodology of the ISOM – the Inverted Self-Organizing Maps – layout from Bernd Meyer should be examined.

Figure 111 shows the complete ISOM algorithm. Lines 01-07 list the different relevant variables. The algorithm runs a predefined number of epochs t_{max}. In Line 18 the variable t is iterated. The epoch interval $t_{interval}$ is checked in Line 19: each time (or: epoch) the next interval is reached, the radius r is decreased by 1. The maximum r_{max} and minimum r_{min} for the radius are defined in Lines 03 and 04. But what is the meaning of this radius?

In Line 15 the radius is used to filter the neighboring nodes of node w – this means only those nodes connected to w by an edge – according to their reciprocal distances d. The radius r is therefore a cutoff which is constant during an epoch and which decreases if the next epoch is reached.

But of course, the methodology is not the same. First, a step back should be made to analyze the initial settings. In Line 10 the method *random_vector()* is invoked. It creates an initial three-dimensional position in space located on the surface of a unit sphere. The nodes are randomly distributed, ignoring collision detection or optimal packing approaches like those implemented in the UUUSphere layout.

In Line 13 a random position is created again. But in this case the random position is a place on the sphere which should guarantee an equal distribution on the surface of the sphere. And this is exactly the task discussed in Section 5.3.2.5.1. Each of the four approaches included in the UUUSphere layout can be used here. These three-dimensional positions are now the poles attracting the nodes which were initially positioned onto simple random positions.

In Line 14 the node *w* is chosen which is closest to the pole chosen in Line 13. The Lines 15 to 17 iterate now through all interconnected neighbors of this node. Each of the neighbor nodes is assigned a new position in Line 16. The normalization ensures the repositioning onto the surface of the unit sphere.

Now the core functions located in Line 12 and 16 of the algorithm have to be examined. The algorithm is based on the idea of Teuvo Kohonen's *Self-Organizing Maps (SOM)* [Koho90]. The SOM was derived from neural networks. The elements of a neural network, the neurons, compete in terms of their response to external input signals, the so-called stimuli. In a neural network usually only one winner neuron is chosen which responds best for such a stimulus. A special characteristic of neurons is that all elements of the network are connected to each other. Therefore, the neuron with the best response to the stimulus inhibits the other interconnected neurons which may cause a deactivation of all other nodes. This procedure is called: *competitive dynamics* (see also: Molecular Dynamics in Section 3.2.3).

```
01  maximal epochs  t_max;
02  epochs interval  t_interval;
03  maximal radius  r_max;
04  minimal radius  r_min;
05  maximal adaption  α_max;
06  minimal adaption  α_min;
07  cooling factor  c;

08  epoch  t:=1;
09  radius  r:=r_max;

10  forall v∈V do v.pos:=random_vector();

11  while(t≤t_max) do {
12        adaption  α:=max(α_min, e^(−c(t/t_max))·α_max);
13        i:=uniform_random_vector();
14        w:=v∈V such that s(v.pos, i) is minimal;
15        for w∧all successors w_k of w with d(w, w_k)≤r do{
16              w_k.pos:=normalize(w_k.pos−2^(d(w, w_k)) α(w_k.pos−i));
17        }
18        t:=t+1;
19        if t mod t_interval=0 ∧ r< r_min do r:=r−1
20  }
```

Figure 111: The ISOM Algorithm applied to a Unit Sphere [Meye98]

But another important factor is the connection strength between different nodes. These are the so-called weights. The intention of neural networks is of course the execution of a learning process. If a stimulus is now presented to a neural network, there will usually be one winner selected. In case of *supervised learning*, there is a teacher who decides if the result of the stimulation matches the expectations. Usually this will be not the case and the

weights of the network will be adjusted. Now the stimulation process is repeated, a winner will be selected and the new results will be applied to the weights of the network, until the desired configuration is achieved. In the case of an *unsupervised network* there is no teacher and the network has to find its optimization criteria itself.

And exactly this is also the case for SOM. But in contrast to the *competitive learning* discussed before, SOMs are optimized for *soft learning*. This means that the weights of neighboring nodes of the winner will be increased as well during the learning process. Until now, the responses of the nodes to the input stimuli had to be computed to elect the winning nodes. But Kohonen also introduced an alternative option: instead of the input stimuli, the smallest distance of the node to the input vector is calculated.

The procedure of the SOM starts with the generation of a single stimulus vector (Line 13). Then the node with the smallest distance to this stimulus is selected, the winning node w (Line 14). The node may be represented by coordinates in two-dimensional as well as three-dimensional space. And this n-dimensional position vector will be normalized in this example. Therefore the length for all vectors – which are those representing node w as well as its successors – is 1. Now the weights of the selected node as well as its neighbor nodes are adapted. And it is exactly this process which is shown in Line 16. The weights w_k representing the three-dimensional vector of the neighbor nodes in the certain radius r and the winning node w are adapted in relation to their distance to w. The function shown in Line 16 was optimized based on Meyer's experimental results with his ISOM layout. But the *adaption factor* $e^{-c(t/t_{max})}$ is based on the original SOM layout. The usage of the exponential function enables a fast adaption during the first epochs and a slower adaption during the last epochs. The cooling factor c can be used to decelerate or accelerate the adaption process.

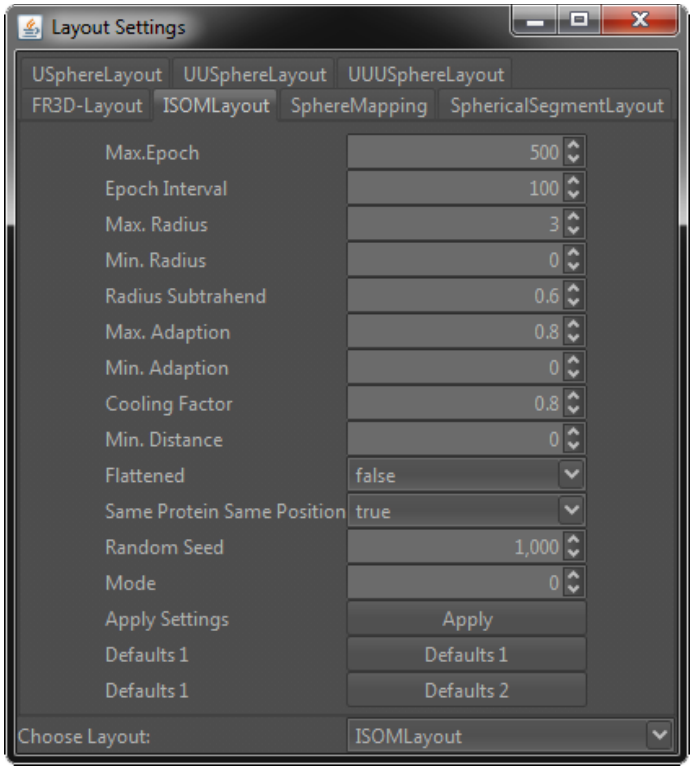


Figure 112: The Window of the ISOM layout in CmPI showing the different properties

Now that the algorithm is explained, the final question remains why ISOM is the inverted process of SOM. First, the input is changed: the stimuli were substituted by the network topology – the connections between the nodes. The second aspect regards the weight space: in SOM, the weight space is used to tune the output of the map. In ISOM, the weight space is the output [Meye98].

The variables listed in Lines 1-7 of Figure 111 can be changed in CmPI using the Layout Settings dialog shown in Figure 112.

5.3.2.6 Node Mapping

Meets Requirements R3.IV,VII,XII

The nodes have been distributed on the surface of the sphere, the layout was applied, and now the final problem described in Section 3.3.4.3 has to be solved: OPT_NMP_POS. Therefore, the second step maps the nodes onto the correct position of the cell component.

In the case of a SphereCellComponent (Section 5.1.2.2), this process is very simple, because it is sufficient to scale the distance between the center of the sphere and the pre-positioned node. But if complex, irregular shapes are used like those shown in Figure 107.2-3, the placing process is more complicated. For this purpose, a *Pick Ray* is sent starting at the center of the cell component to be directed to the actual position of the node. The Pick Ray is a Java 3D component which contains an ordered list of all shape collisions (Section 3.4.3.2). The list created by a Pick Ray for a node in a crowded cell model may contain the following entries in conjunction with the respective point coordinates:

0. mitochondrial matrix,
1. mitochondrial matrix,
2. mitochondrial inner membrane,
3. mitochondrial intermembrane space,
4. mitochondrial outer membrane,
5. mitochondrial cloud,
6. cytosol,
7. cell membrane, and
8. extracellular space.

It is important to see that there might always be redundant entries like entry 0 and 1, because the shapes may contain folded and/or cave-like structures. In the case of the nodes in Figure 107, the mapping is as follows:

1. mitochondrial matrix,
2. mitochondrial inner membrane,
3. mitochondrial intermembrane space,
4. mitochondrial outer membrane, and
5. mitochondrial cloud.

The “CENTER” node is located in the origin of the mitochondrion.

The Pick Ray is analyzed and in case of Node “1” the first list entry, “Mitochondrial Matrix”, is selected and the pre-positioned node is shifted from the surface of the sphere onto the picked point at the shape of the matrix.

But as shown in Figure 103 there are different types of nodes in the cell environment. CmPI is focusing on the localization of proteins, therefore only the proteins are placed by this algorithms. But what is happening to the compounds?

The compounds are always placed in relation to the proteins. There are different rules concerning the compound-placement:

- if two proteins correlated with a compound are located at the same cell component, the compound is placed in the center position of these nodes;
- if the proteins are located at different locations throughout the cell environment, the compound is placed in an average point between the different nodes; and
- if the compound node has only one parent node, it is placed in the closer neighborhood of this point.

5.3.2.7 Correlating Networks

Meets Requirement R3.VIII

In CmPI, the correlation of networks is based on the localization of proteins. In the last section the layout process of interconnected proteins was discussed. In Figure 112 the options of the ISOM layout were shown. One option is called “Same Protein Same Position”. If this option is selected, proteins of the same type are placed onto the position, if they are located a) at the same cell component and b) at the same cell component membrane layer.

In Figure 103 the “Alternative Localization Edge” is shown. This is an alternative way to visualize localizations. The protein at the actual localization is correlated with all alternative protein positions at other available Cell Components.

5.3.2.8 Coloring Methods for Networks

Meets Requirement R3.IX

The coloring methods with focus on the Cell Components were extensively discussed in Section 5.1.2.3. The same methods can also be applied to the networks in CmPI.

The coloring of a network is exclusively applied to the edges (Figure 103). If there is a set of different pathways, there are two basic options. If there is no sequential order of the pathways, then the “Contrast Color Codes” are the best choice (Section 5.1.2.3.1). But if there is a special order associated with a set of pathways, the method “Color Scales” should be chosen (Section 5.1.2.3.2). In addition, the user has the option to directly change the color of a single network.

5.3.2.9 Coloring Methods for Nodes

Meets Requirement R3.X

In Figure 103 the coloring for nodes was shown; the proteins are usually colored in red and the compounds in blue.

But a few words should be directed to alternative differentiation approaches. One option would be to distinguish nodes by using different types of shapes. But the problem is that usually particular complex shapes need more computational performance than simple spheres. The other option, the texture, has a big disadvantage: it is hard to recognize differences from a distant point. And naturally the navigation through a three-dimensional space is accompanied by the continuous changing of distances. Therefore, coloring outperforms the use of textures in this context. In addition, it was shown that coloring often

(but not always) outperforms shapes as well as textures in terms of fast differentiation perception [Gree10].[66]

Of course, the node colors in CmPI can also be changed by the user. Another interesting option is the coloring of nodes according to the actual node localization. A similar methodology is already known from GEOMI (Section 4.3.3.3).

In the case of CmPI, the actual selected color scheme of those discussed in Section 5.1.2.3.3 will be chosen. For example, if the color scheme "Contrast Colors" was chosen, a protein node localized at the mitochondrion will be colored in "Mallow" (see Figure 89, Section 5.1.2.3.1 and Appendix 9.14):

5.3.2.10 CELLmicrocosmos Navigation and the NodeDetails Window

Meets Requirements R3.XIII,XIV

The navigation for CmPI was drastically extended to meet the requirements of a network-correlated cell environment [SKSH10]: a *6DOF-navigation* (featuring 6 degrees of freedom). For this purpose, three different navigation modes has been implemented which altogether are referred to as the *CELLmicrocosmos Navigation*:

- Floating Mode
- Flight Mode
- Orbit Mode

In the *Floating Mode*, the mouse can be used for selecting objects. Navigation is only possible by using the keyboard. Regular keyboard commands can be used which are also known from game-associated environments like the WADS-keys, the NUM-block-keys and the arrow keys.

The *Flight Mode* is toggled by holding the Shift-key. In this mode, complete navigation is possible by using the mouse. Left Clicking moves forward, Right Clicking backwards. The Center Mouse Button or the Shift key toggle the Flight Mode. The movement of the mouse changes the orientation. During movement, the mouse wheel can be used to change the speed. This is especially important if the user wants to follow, for example, a pathway on a low scale inside the mitochondrion – in this case he needs a low speed – and then wants to travel in a fast way to another cell component like the nucleus.

The *Orbit Mode* is based on the standard Java 3D "Orbit Behavior" (Section 3.4.3.2) which is also used in CmME. Cell Components and all nodes can be focused by a simple double click (with the left mouse button) and the movement is centered around the selected point by holding the left mouse button. With the right button the viewing direction can be changed and with the centered mouse button the distance between the object and the view port can be changed. The Orbit Mode can also be used to jump from node to node: if the left mouse button is triple-clicked, the user flies into the direction of the selected node until a certain distance to this node is reached.

66 In the cited work the textures were represented by faces [Gree10].

In the Floating and Orbit Modes a right click toggles a pop-up which can be used to toggle different additional options.

The CELLmicrocosmos Navigation has another important function supporting the selection of objects in the background which are blocked by other objects in the foreground. If the user holds the left mouse button clicked down and moves the mouse wheel, all objects lying under the mouse pointer can be browsed by scrolling. This is done by making all objects in the *Cell Universe* – the Java-3D-based environment where the cell components are located – transparent and only solidly showing the active object.

A high level of interactivity is also guaranteed by integrating all other view into the navigation. As mentioned before, the 2D View (Section 5.3.2.2.3) can be used to navigate through the cell environment, but it is also possible to double-click onto proteins in the Localization Table (Section 5.3.2.3.4) or to jump from node to node by selecting them in the *NodeDetails* window which shows many details about the currently chosen node: name, synonyms, associated pathway, current localization and – especially interesting for the navigation – the neighbor nodes.

5.3.2.11 3D Stereoscopy and Shading

Meets Requirements R3.XV,XVI,XVII

It was already mentioned in Section 5.1.2.3.2 that the shading is an important aspect for three-dimensional visualizations to accentuate the spatial relationships inside a cell. The nodes and pathways are also shaded in order to improve the depth perception. But another very important technique supporting this context is 3D-stereoscopic visualization.

The basic idea of 3D Stereoscopy is the emulation of human visual perception. The homo sapiens owns two eyes, therefore an object is rendered from two perspectives. Even though the first stereoscopic devices were already invented in the early 19th century, 3D Stereoscopy was in the last decades mainly intended for professional applications like medical domains [BHHI09, Brew56] The commercialization during the last few years enabled the use of stereoscopic devises in the exhibition, movie as well as gaming industries. Today, projectors, televisions, PC monitors, portable game consoles and even notebooks are available using this technology.

The applications of the CELLmicrocosmos project have supported 3D Stereoscopy compatible with professional devices like NVIDIA® Quadro® graphic cards for years. In addition, first tests on a consumer notebook have been presented in 11/2011 and 03/2012 at two opportunities during poster presentations [CiOS11, SoZh12]. Especially when navigating through three-dimensional networks, the advantages of 3D-stereoscopic viewing are obvious [WaFr96, WaMi05].[67]

Figure 113 shows a stereoscopic camera modeled in 3ds Max. The top view shows a cell environment: in its center there is the cell model which is surrounded by a circle. This circle is a camera path, along which the cameras may be moved. Two cameras are attached: the

67 Of course, there are a number of positive as well as negative aspects to be discussed in the context of 3D Stereoscopy. The interested reader is referred to different interesting publications on this topic [HWSB99, Lipa07, TsAW08, WaRM95].

red one is the camera for the right eye and the green one for the left eye. In the middle of Figure 113 the projections of both cameras are shown. The figure at the bottom shows the front view of the cell environment. The local coordinate axes on the left side, located inside the cell model, represent the camera target. The lenses of the cameras are always pointing onto this position. But especially interesting in terms of stereo viewing is the second local coordinate system located between the camera target and the camera itself. This element represents the focal distance. Objects lying between the focal distance and the camera will appear to move outside the stereo device, for example, a 3d-stereoscopic monitor. These objects seem to move towards the spectator. On the contrary, objects lying between the focal distance and the camera target will lie beyond the screen. Therefore, the distance perception can be regulated by manipulating this slider.[68]

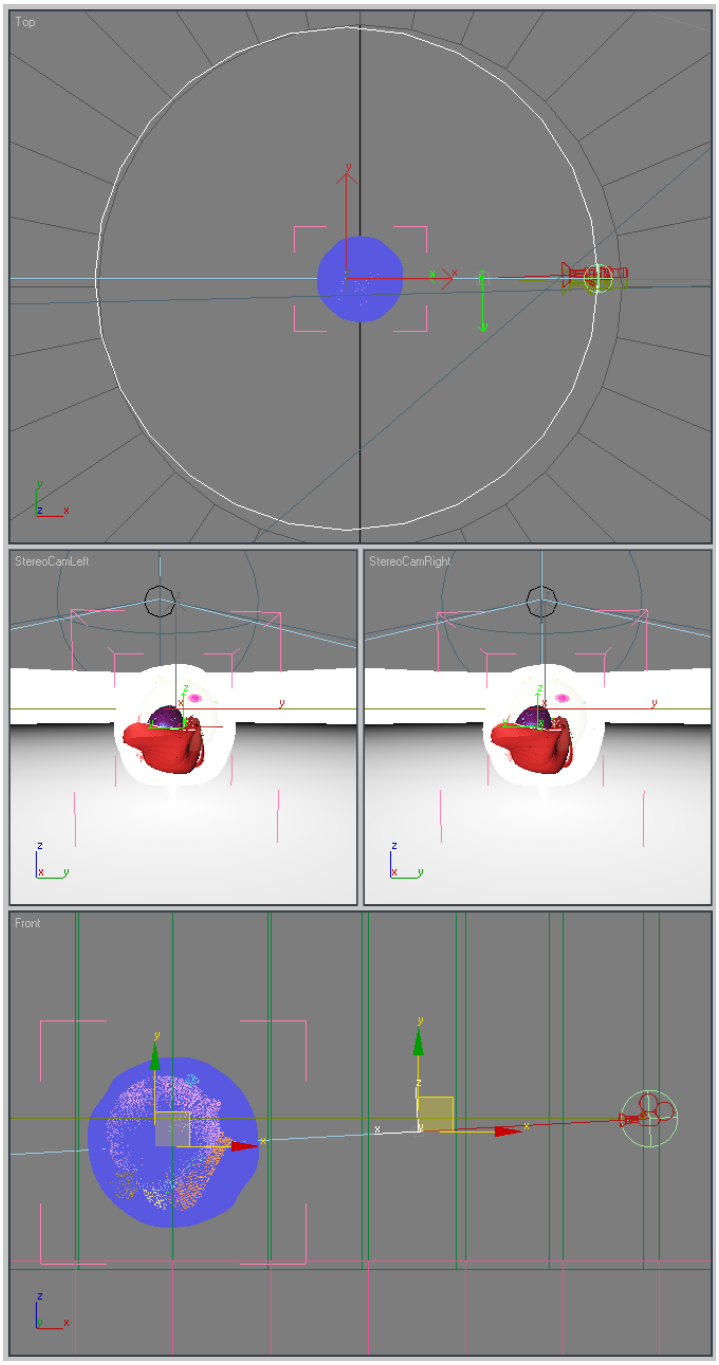


Figure 113: A 3D-stereoscopic Camera Cell Environment in 3ds Max

Of course, this method can be used for animations. But in the context of CmPI, the problem arises that the stereoscopic perception must be regulated during the navigation process. The reason is the large differences in distances: in the beginning, the user usually sees the whole cell model. But when the user navigates to a cell component or a protein node and (looking again at Figure 113) the problem is apparent; the focal distance has to be adjusted.

For this purpose the focal distance is indirectly adjusted automatically. This is done in CmPI by sending a Pick Ray from the center of the view port into the viewing direction[69]. The first position obtained by collision detection with an object – a cell component or node – will be used to recalculate the distance between both eyes; in Figure 113 this is the distance between the red and green camera. Because of performance issues, the Pick Ray cannot send continuously towards the view port center. Therefore, the eye distance is only updated if the navigation process is stopped.

68 This approach is an extended version of the technique introduced in an on-line tutorial by Elliott Smith [Smit11].

In CmME another approach is used: because the Orbit Behavior is used, there is continuously a focus point around which the navigation is performed across the membrane. This focus is used to permanently recalculate eye distances.

5.3.2.12 Export Capabilities

Meets Requirement R3.III

The import of data into CmPI has been discussed before (Sections 5.3.2.2.2 and 5.3.2.3.2). Another important feature is the export of data. The pathways in correlation with the cell environment can be saved in different formats.

5.3.2.12.1 XML (.Cm4)

The native format was already discussed in Section 5.1.2.7. For CmPI, the Cm3-file is extended by Cm4-tags. Therefore, the Cm4-file optionally combines Cm3-tags containing information concerning the cell model configuration, but it is mandatory that it contains all information needed to rebuild one or different pathways, with all localization information, sources and actual settings. Of course, if Cm3-information are included, also the packaging with external VRML97 files is done.

5.3.2.12.2 SBML

In CmPI there are additionally export capabilities using SBML level 2 version 3 and 4. The SBML export creates files similar to those of the Cm4-format, partly compatible with external tools. Of course, those tools can usually only read the pathway-related information [Özgü10].

5.3.2.12.3 VRML97

The VRML97 format was introduced in Section 3.1.3.4. It was already mentioned that this format is used to import cell component models. But additionally, it is possible to export the whole cell as one model, correlated with pathways. The actual settings and layouts of the pathways are taken into account during the saving process.

5.3.3 Implementation Details

The CELLmicrocosmos 4.2 PathwayIntegration (CmPI) was once developed based on the CELLmicrocosmos 3.1 CellEditor (CmCE), whose capabilities were discussed in Subchapter 5.1. Today CmPI is integrated in the CELLmicrocosmos 1.1 CellExplorer. The implementation information can be found in Section 5.1.3.

The project is located at:

http://Cm4.CELLmicrocosmos.org

Please refer to the Appendix 9.17 for information regarding the implementation work and to Appendix 9.15 for the version information. The included third-party libraries and their licenses are listed in Table 14.

69 The Pick Ray from Java 3D was already used for the Node Mapping (Section 5.3.2.6).

Name	Version	Usage	License	More Info
jung-algorithmns	2.0	2D Viewer	BSD License	http://jung.sourceforge.net
jung-api	2.0			
jung-graph-impl	2.0			
jung-io	2.0			
jungvisualization	2.0			
collections-generic			Apache License 2.0	
axiom-api	1.2.7	Communication with Databases via MySQL or WebService Connection	Apache License 2.0	
axiom-impl	1.2.7		Apache License 2.0	
axis-adb	1.4		Apache License 2.0	
axis-kernel	1.4		Apache License 2.0	
backport-util-concurrent	3.1		CPL/EPL	
commons-httpclient	3.1		Apache License 2.0	
commons-codec	3.1		Apache License 2.0	
commons-logging	11.1		Apache License 2.0	
concurrent			Public Domain	http://g.oswego.edu/dl/classes/EDU/oswego/cs/dl/util/concurrent/intro.html
log4j	1.2.15		Apache License 2.0	
neethi	2.0.4		Apache License 2.0	
mysql-connector-java	5.15		GPL	
wsdl4j	1.6.2		CPL/EPL	
JFreeChart	1.0.13	Localization Charts	LGPL 2.1 (or later)	http://www.jfree.org/jfreechart/
XmlSchema	1.4.2	XML import and export	Apache License 2.0	

Table 14: The program packages included in the CELLmicrocosmos 1.1 CellExplorer relevant for this subchapter (CELLmicrocosmos 4.2 PathwayIntegration)

5.3.3.1 Integration of DAWIS-M.D. and ANDCell

DAWIS-M.D. as well as ANDCell have been already introduced (Sections 3.3.3.1 and 3.3.3.2). Here, two methods should be shortly explained which are used by CmPI to connect to these databases.

The first method is a direct connection. In this case, MySQL-queries are sent directly to DAWIS-M.D. and/or ANDCell and the results can be directly received and computed. The data exchange is very fast, but this method can only be used in the institution where the system is installed (in our case the Bio-/Medical Informatics Department).

The alternative method enables the communication between the different systems by using an Internet connection. An asynchronous web service is used to which the application connects on-line. The web service administrates the communication between DAWIS-M.D. and the application. Figure 114 shows the workflow between the different systems.

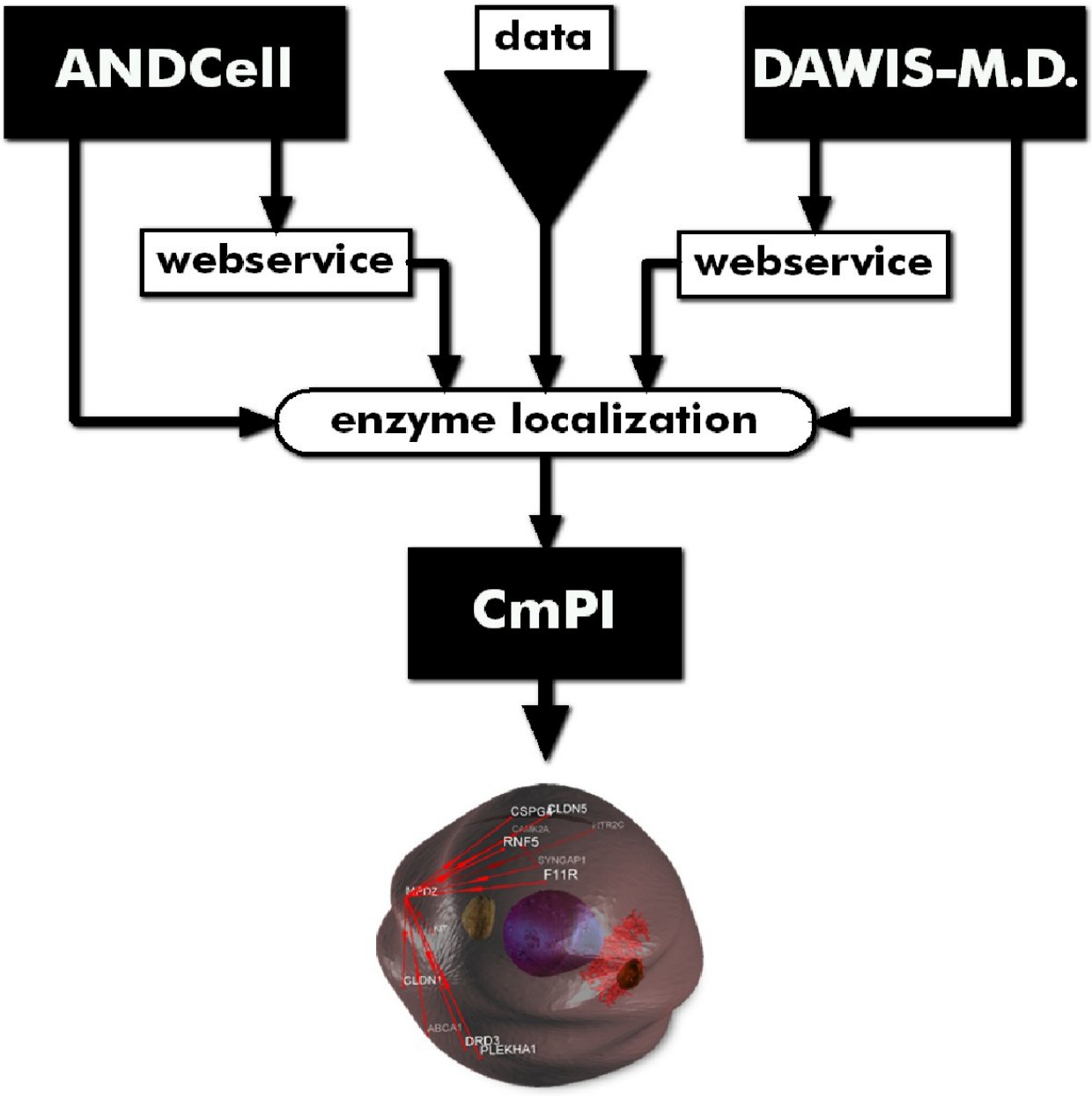


Figure 114: The Workflow of CmPI, showing also the different connection types between CmPI and ANDCell/DAWIS-M.D.

5.4 [MES+MOL+FUN] Integrative Modeling: Combining Mesoscopic, Molecular and Functional Modeling

The preceding subchapters introduced all methods needed to model a cell environment, to generate membrane models as well as the creation and mapping of protein-related networks. Now the final question should be evaluated: how is it possible to link these three levels – the mesoscopic, the molecular and the functional – into one application?

5.4.1 Requirements

R4.I Combination of CellExplorer and MembraneEditor

There must be an optional tool which can be used to combine CellExplorer as well as MembraneEditor.

R4.II Membrane Mapping

Resulting membrane models have to be shown in correlation with the Cell Components in CellExplorer. Techniques have to be developed to do this.

5.4.2 Methods

The overview of the integrative level was already shown in Figure 7. A closer perspective is provided by Figure 115.

5.4.2.1 CellExplorer

In the implementation sections 5.1.3 and 5.3.3 the term CellExplorer was already mentioned. This project is already an integrative approach, combining CellEditor with PathwayIntegration. The name was changed when the focus of the major application was made on the exploration of existing cell models. Especially the implementation of the CELLmicrocosmos Navigation discussed in Section 5.3.2.10 was an important step in this direction. Therefore, CellExplorer already unites the mesoscopic level with the functional level. For example, it is possible to correlate a cell model generated with CellEditor with a metabolic pathway reconstructed with PathwayIntegration.

In the preceding sections it will be explained in which way these two levels are finally combined with the molecular level.

5.4.2.2 CmX mRNA

Meets Requirement R4.I

The CmX is a small application which combines three programs:

- CellExplorer,
- MembraneEditor, and
- Jmol (which is included as a Stand-alone application in MembraneEditor package).

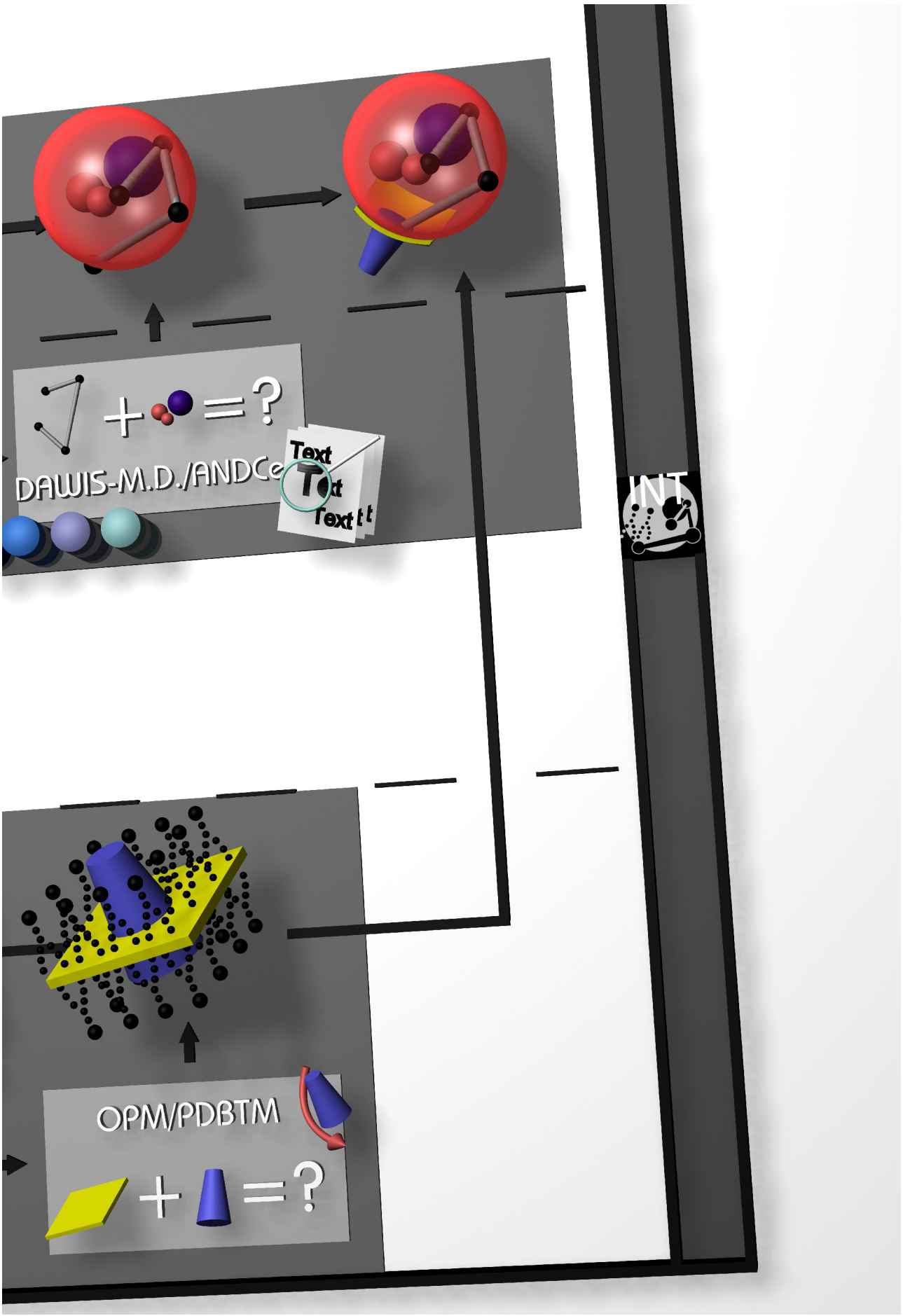


Figure 115: The CELLmicrocosmos modeling pipeline at the integrative level

CmX contains an abstract class called the MRNA. It is the only class contained in the package cmX_mrna. The complete code is shown in Figure 116. As its name suggests, this class is used to enable communication processes between CellExplorer and MembraneEditor. The functionality will be explained in the following paragraphs describing the workflow. The abstract MRNA class is also found in MembraneEditor as well as CellExplorer. The implementation is located in the CmXMRNA file part of CmX. Of course, CmX is not a stand-alone application: it needs the complete CellExplorer as well as MembraneEditor projects.

The correlation of a Cell Component with a membrane in PDB-format is saved in the Cm3-file discussed in Section 5.1.2.7. In addition, all PDB-files are packaged with the Cm3-file using the same method as for the different VRML97-files.

Figure 117 shows a potential workflow using CmX. On the left side of the image, the methods associated with CellExplorer are shown, on the right MembraneEditor and in the center the communication processes administrated by CmX.

1. The CmX application is started. It can be used to open separate instances of CellExplorer, MembraneEditor or Jmol. In this example, CellExplorer is opened. The main window of CmX administrates all different program instances, also in case CellExplorer is now used to open MembraneEditor.
2. In CellExplorer, a Cm4-file is loaded or a cell is generated using CellEditor and correlated with metabolic pathways using PathwayIntegration (Sections 5.1.2.5, 5.1.2.7, 5.3.2.4 and 5.3.2.5). In this example it is important that a KEGG pathway is downloaded which contains information about the PDB files associated with the different EC numbers. Now the layer of the Cell Component is chosen which should be attached to a new membrane. In this case, the inner membrane of the mitochondrion is chosen.
3. An instance of MembraneEditor is opened and the user is now able to generate a new membrane model using the Lipid Packing Algorithms (Section 5.2.2.2.1). The code in Figure 116 shows the underlying function which opens MembraneEditor: “editMembrane”.
4. After the membrane is generated, a protein should be placed inside the bilayer. For this purpose, the NodeDetails window of PathwayIntegration provides a list of PDB IDs associated with the chosen EC number (Section 5.3.2.10). By clicking on “MembraneEditor (Cm2) Search”,
5. a new window is opened, asking the user if a new instance of MembraneEditor should be opened or a running instance should be used. Because the protein should be placed into the previously generated membrane, the running instance is chosen. The code in Figure 116 shows the relevant method: “performSearchInDownloadAPDB”.

6. Then, the search is started using the “Download a PDB” window in MembraneEditor (Section 5.2.2.2.2 and Figure 97).
7. A membrane protein is chosen featuring alignment information and the placed into the membrane bilayer removing intersecting lipids (Section 5.2.2.2.3).
8. Then the membrane is saved in PDB-format and immediately attached to the Cell Component in CellExplorer. The method “refreshMembranesInCm1” provides the underlying functionality in the code of Figure 116.
9. Now the user can navigate to the according layer of the Cell Component. A texture rendered from the associated membrane is shown. Moreover, the user can perform a right-click with the mouse onto the layer and select “Show Inner Membrane”. Then the membrane model is three-dimensionally mapped onto the surface of the Cell Component layer (Section 5.4.2.3).
10. Another option in the menu toggled by the right-click is to open the actual membrane in Jmol.
11. A Jmol instance is opened and the according PDB model is shown. Jmol is packaged with MembraneEditor (Section 5.2.2.3). The “openMembraneInJmol” method in the code of Figure 116 provides this functionality.

```
package cmX_mrna;

/**
 * The messangerRNA:
 * This class connects Cm1 CellExplorer and Cm2 MembraneEditor
 * by using methods of the CmX Package.
 *
 * @author Bjoern Sommer bjoern@CELLmicrocosmos.org
 *
 */
public abstract class MRNA {

	public void editMembrane(String absolutePath){
	}

	public void openMembraneInJmol(String absolutePath) {
	}

	public void refreshMembranesInCm1(String absolutePath, String cm1TempName2Edit) {
	}

	public void performSearchInDownloadAPDB(final String searchString,
	                                        final String searchTitle) {
	}

}
```

Figure 116: The complete MRNA.java code of CmX

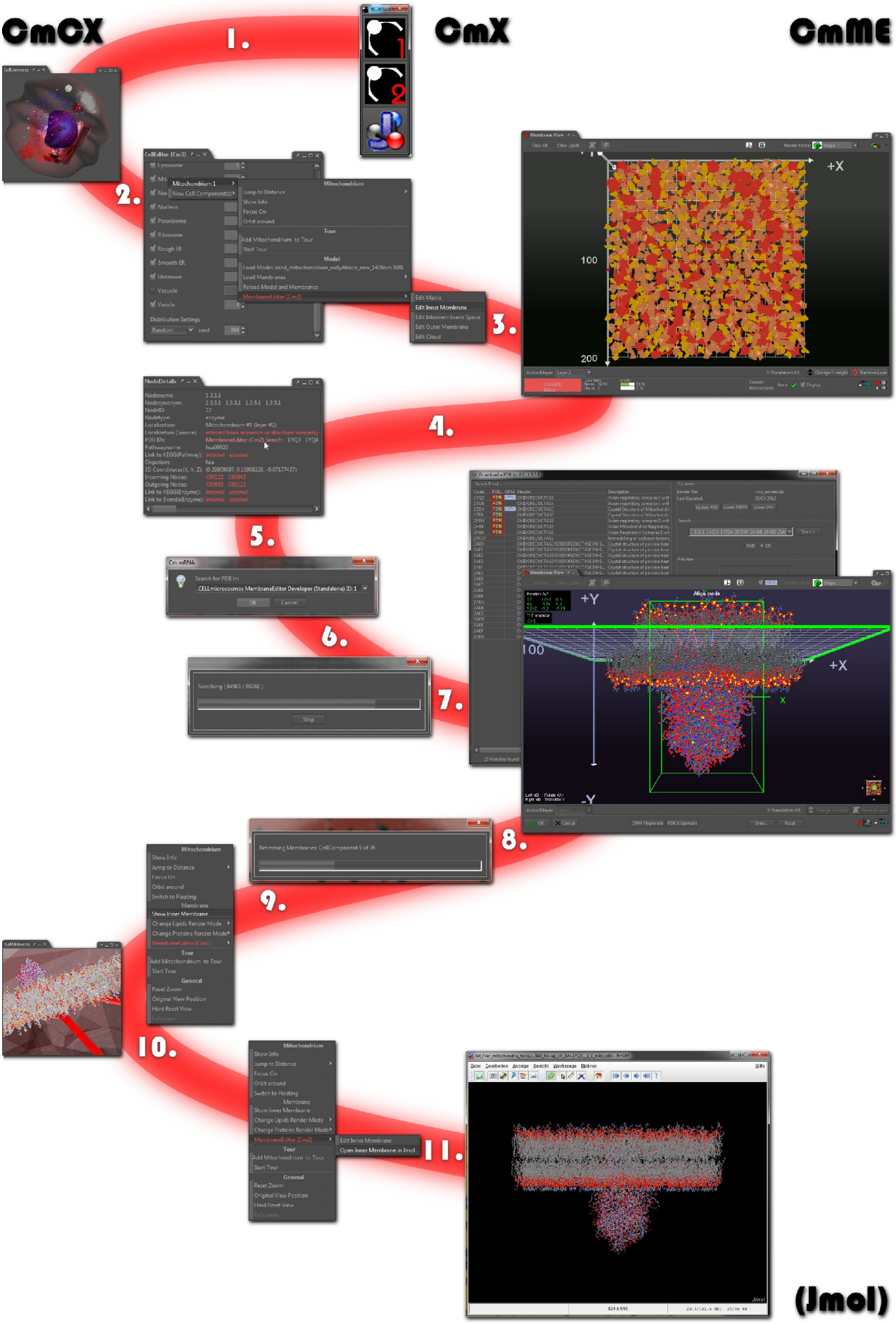


Figure 117: Overview of a potential CmX Workflow combining CmCX, CmME and Jmol

5.4.2.3 Membrane Mapping

Meets Requirement R4.II

In step 9 of the preceding section the mapping of a molecular membrane onto a cell component was shortly mentioned. How does the mapping work?

The first approach of mapping a membrane onto cell components was already implemented by Marion Zysik during her diploma thesis. Originally, CellEditor discussed in Section 5.1.2.5 was mainly intended to generate cell models correlated with PDB-models only for the purpose of showing them in Marion's application; the “pdbFiles”-tag in the Cm3-file were saved exactly for this purpose (Section 5.1.2.7). But her application was developed in C++ using the VTK packages for visualization; therefore it was not included for future development [Zysi07].

In the previous section the workflow of mapping a membrane onto a cell component was already described. There are two methods to map a PDB membrane generated with MembraneEditor onto a cell component in CellExplorer:

1. The PDB-membrane is rendered from the top perspective as a two-dimensional Java 3D texture and mapped onto the complete shape of the associated cell component layer.
2. The PDB-membrane is three-dimensionally mapped onto a distinct position of the active cell component layer.

The texture mapping is done using the usual Java 3D methods. Therefore, only the second method should be described in the next paragraphs.

After the user has fallen short of a predefined distance, the pop-up shown in Figure 117.9 can be invoked. If the option “Show membrane X” was chosen, the clicked position is used as the center point to place the membrane. It is no problem to get the position of the membrane. The Java 3D Pick Ray, which was also used in Section 5.3.2.4, contains the information of the intersecting point. But it is complicated to adjust the alignment of the membrane. For this purpose, another Java 3D picking object is used: the *Pick Cylinder*. When clicking the right-mouse-button, all points lying in a virtual circle around the mouse cursor are stored in a list. These points, together with the positioning point found by the Pick ray, are used to generate a plane. And this plane is used to compute a surface normal. And based on this surface normal the membrane is adjusted. This method does not always compute an optimal normal, but it is featuring an appropriate performance.

Because the membranes are regularly rectangular-shaped, a special method was used to generate circular membranes whose outer borders integrate better into the cellular environment: the rectangular membrane is surrounded by a circle containing the same lipid composition like the rectangle. In addition, two outer circular areas contain 50% and 25% of the same lipid composition.

5.4.3 Implementation Details

The CELLmicrocosmos X.0 mRNA (CmX) was originally developed in Java 6. This small tool depends on CellExplorer (Section 5.1.3) and MembraneEditor (Section 5.2.3) project which have to be found in the class path of CmX. This program is intended as a first approach, therefore it is an alpha-version (version X.0) and there is no Java Web Start version presently available.

Please refer to the Appendix 9.17 for information regarding the implementation work and to Appendix 9.15 for the version information. The included third-party library and its license are listed in Table 15.

It includes the following libraries:

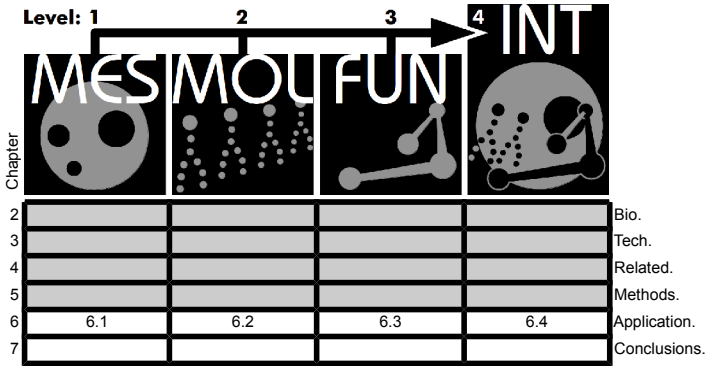
Name	Version	Usage	License	More Info
Substance	4.0	The L&F of the GUI	BSD (without advertising clause)	http://substance.dev.java.net

Table 15: The program packages included in the CELLmicrocosmos X.0 mRNA

6 Application Cases

All relevant methods have been introduced to generate cell and membrane models as well as protein-based networks. Now the different application cases will be presented.

Which steps are needed to generate a simple cell model? How complex is the task of creating a membrane model? And what about the quality of the different membrane packing algorithms? And then, how is it possible to localize a metabolic pathway or other protein sets in the cellular environment? How can they be visualized? And finally the integrative approach will be introduced, addressing the question how can the different levels be integrated into one single model?



6.1 [MES] Cm3 CellEditor

Subchapter 6.1 will discuss different application cases of methods introduced in Subchapter 5.1. CellEditor can be used to model eukaryotic cells like animal cells (Section 6.1.1) and plant cells (Section 6.1.2). In addition, the creation of bacterial cells will also be briefly addressed (Section 6.1.3). A very high grade of abstraction is provided by the SphereCell (Section 6.1.4). The import (Section 6.1.5) as well as export (Section 6.1.6) capabilities will be covered, showing how to handle externally generated models. Finally, it will be shown in which way the generated cell models can be used for exhibitions (Section 6.1.7) and education (Section 6.1.8).

6.1.1 Construction of an Animal Cell

In Section 5.1.2.5 the methods to create cell component distributions by using CellEditor were discussed. Using these methods, two eukaryotic cells featuring ABS_LEV_2 were created. The cell component coloring in this case mainly follows the H&E coloring method.

6.1.1.1 Construction of an Animal Comparison Cell

The first cell model is shown in Figure 118. Obviously this is not a typical cell component distribution as it could be expected from a cell. Here, the “Comparison”-distribution mode was chosen. The first row shows three mitochondrion models, the second row, three golgi apparati, and the third row three vesicles. The number of cell components was defined by using the cell editor (Figure 91), and the distance between the compared cell components

was set by using the distribution options (Figure 92). The fact that the third vesicle seems not to be visible is the result of the transparency settings. In the Virtual Cell environment, the vesicle would appear again if the position of the observer would be changed.

Of course, these cell components could be substituted by other cell components of the same type. Therefore, different cell component models of the same type could be compared. Another application case for comparison cells will be discussed in Chapter 6.3.2.4.

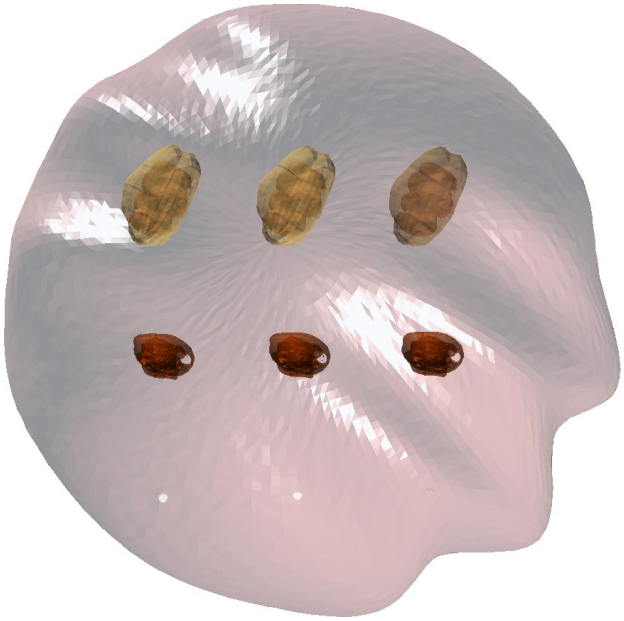


Figure 118: An abstract animal cell created in comparison mode by CellEditor
Top: 3 mitochondria; center: 3 golgi apparati; 3 vesicles

6.1.1.2 Construction of a regular Animal Cell

Of course, the previously discussed distribution mode is not appropriate for creating cell models with a cytological reasonable composition. For this purpose, the "Random" mode is used with a random-seed of 995 (Figure 92). Figure 120.1 shows the result. The cell contains the following models: cell membrane:1; cytosol: 1; extracellular matrix: 1; golgi apparati: 3; lysosome: 1; mitochondrion: 1; nucleolus: 1; nucleus: 1; peroxisomes: 2; ribosomes: 5; rough ERs: 2; smooth ER: 1; vesicles: 4.

Figure 119: CellEditor: a detail of the eukaroytic cell
The complete cell is shown in Figure 120.1, in the center: the ribosome

The nucleus is found close to the center, surrounded by the endoplasmic reticulum (ER); the rough ER close to the nucleus, the smooth ER adjacent to the rough ER (the tubular structure at the top in Figure 120.1). The golgi apparati are also found on the left-side of the rough ER. The mitochondrion is located on the right side of the cell and the small green structure close to it is a peroxisome. The small vesicles distributed throughout the cell can be seen at a few positions – the white structures close to the smooth and rough ER. The small dark particle located in the upper section of the cell model is a lysosome close to the cell membrane which could be about to leave the cell environment. Naturally, the ribosomes are too small to be visible in the resolution provided by Figure 120.1. Here, the need for a navigable

virtual environment becomes apparent, because by moving through the Virtual Cell, also small cell components can be explored. Figure 120 centers the small and very abstract ribosome. Beneath it in the background: the peroxisome. The red structure is the rough ER. The small red spheres represent also ribosomes, but they are part of the rough ER. The whole group of these ribosomes represent one layer of the rough ER. As discussed before, the surface of this cell component is studded with ribosomes (Section 2.1.3.6).

All cell components included were discussed, except two: the cytosol and the extracellular matrix. Both are invisible, because they represent the liquid around the inner cell components and the one around the cell membrane. It will be later discussed, for which application cases these structures are needed (Section 6.3.1.6).[70]

6.1.2 Construction of a Plant Cell

Figure 120.2 shows a plant cell (ABS_LEV_2) containing the following cell component models: cell membrane:1; cell wall: 1; chloroplasts: 2; cytosol: 1; extracellular matrix: 1; golgi apparati: 3; lysosome: 1; mitochondrion: 1; nucleolus: 1; nucleus: 1; peroxisomes: 2; ribosomes: 5; rough ERs: 2; smooth ER: 1; vacuoles:2; vesicles: 3. Obviously, a number of new cell components were added which should be briefly introduced. The large globular structure is the vacuole. Here the feature of CellEditor was used to integrate two different vacuole models. In addition, the vesicles are larger than those found in the previously discussed eukaryotic cell. In front of the large vacuole: the chloroplast. Moreover, another cell membrane model was integrated surrounded by a double-layered cell wall.

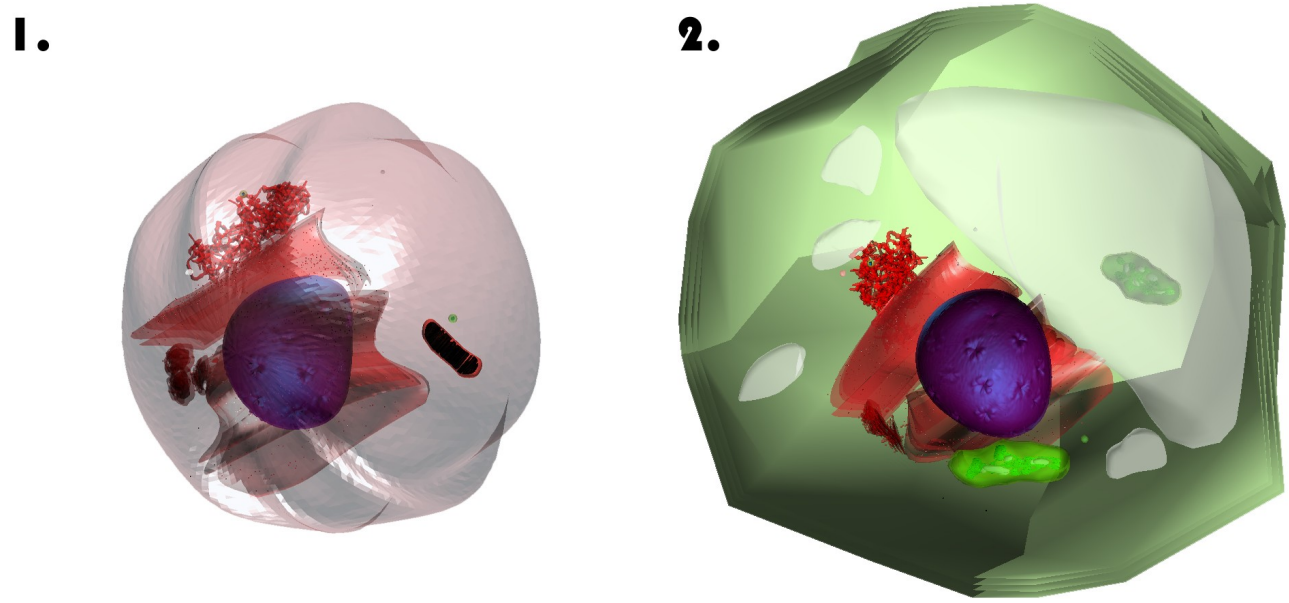


Figure 120: Two eukaryotic cells in CellEditor
1. an animal cell; 2. a plant cell; all shown cell components apply to abstraction level 2 (Section 5.1.2.1)[71]

70 The discussed Cm3-models are: comparison_mito_golgi_vesicle_3each[Cm3] and Tierische_Zelle__1_3[Cm3].

71 The majority of this cell component models has been modeled by Björn Sommer. Some models have been

6.1.3 Construction of a Bacterial Cell

As discussed in Section 5.1.2.5, the modeling of a prokaryotic cell (ABS_LEV_2) is also possible by using CellEditor. Figure 121 shows this cell model including all appropriate cell components available, listed from the innermost to the outer shape: storage granule, nucleoid, plasmid, ribosomes and the membrane-associated components: cytosol, cytoplasmic membrane, cell wall and capsule.

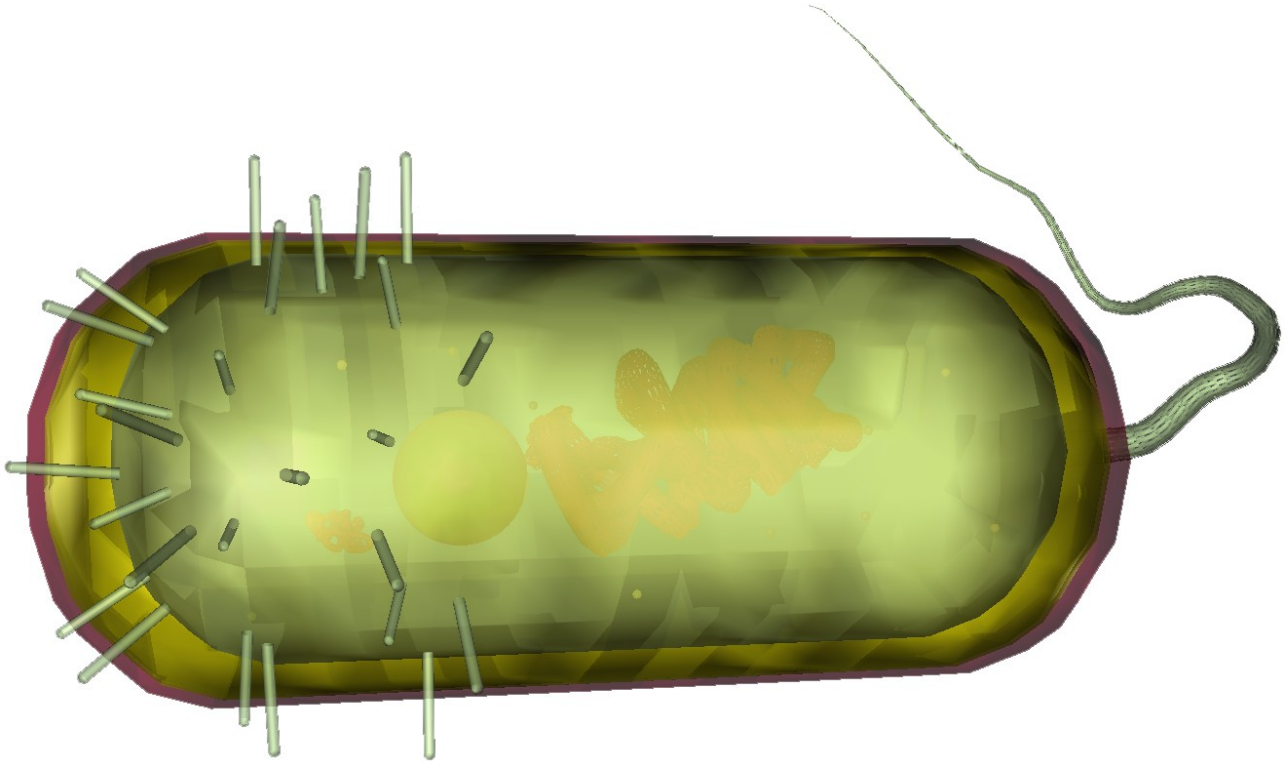


Figure 121: The prokaryotic cell in CellEditor
[72]

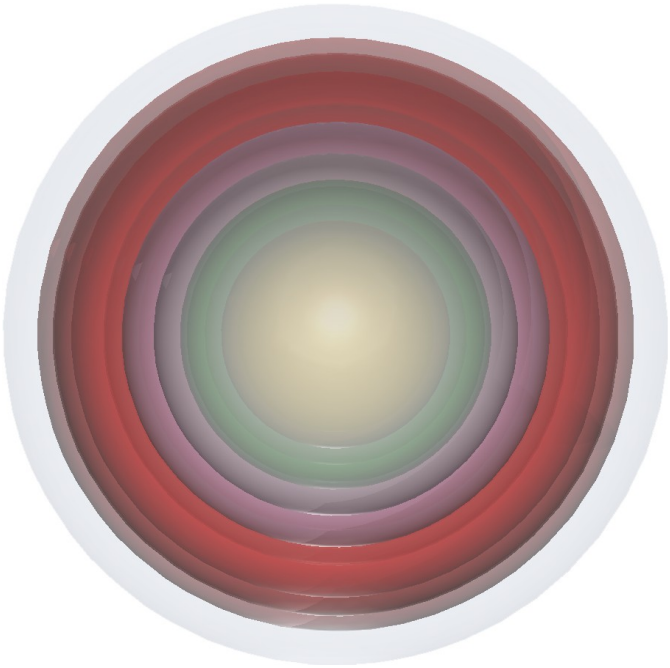


Figure 122: CellEditor: the SphereCell of a Eukaryote

The membrane-associated cell components are all centered in the origin of the cell with the same rotation. A random distribution would not be sensible, because all components depend on each other in their position as can be seen in Figure 121.[73]

6.1.4 Construction of a SphereCell

Especially the section concerning the coloring discussed aspects of the SphereCell (Section 5.1.2.3). This kind of

created by Florian Heißenberg (see also Appendix 9.17.3).

72 This cell model has been modeled completely by Nils Rothe (see also Appendix 9.17.3).

cell model is especially useful for the functional modeling, as will be discussed later (Section 6.3.1.6). The SphereCell is a construction of centered spheres; hence it is not possible to use duplicates of one cell component type. Figure 122 gives a preview of this cell, showing all eukaryotic cell components. As mentioned before, this cell model can be constructed with basic Java 3D objects; therefore there is no need for the modeler to import any VRML97-files. Of course, these simple models apply to ABS_LEV_3.

The representation of cell components by sphere objects is also known from The Interactorium (Section 4.3.3.4). But the latter mentioned software distributes the spheres throughout the cell. In context of CellEditor it proposes using cell components of ABS_LEV_2 or ABS_LEVEL_1, because they provide distinctive features.

6.1.5 VRML import of single cell components

In this section the simple workflow to create a cell component will be summarized.

On the left side, Figure 123 shows the modeling process in 3ds Max (Section 3.1.3.2). Here a mitochondrion is modeled. Instead of creating a complex shape, only a box is constructed, a base object in 3ds Max. This cell component visualization is similar to the one used in MetNetVR (Section 4.3.3.2) and would be normally used – as known from the SphereCell – for functional mapping.

Of course, this simple cell component must also match the prerequisites defined by CellExplorer; it has to include five layers (Figure 90). And as previously discussed, those are represented by five shapes, like the five base objects represented by boxes. Looking at the three 3ds Max viewports containing only the object´s two-dimensional projections in Figure 123, five boxes become clearly visible which are placed inside each other. Prior to this, each box was placed in the center of the 3D-environment and the sizes were adjusted accordingly.

Then the VRML97 export dialog of 3ds Max is invoked. Different options are possible, as shown in the center of Figure 123. Because Java 3D provides a powerful VRML97-parser, many compatible options can be chosen here. Different polygon types as triangles, quads or Ngons can be chosen, normals as well as color per vertex can be exported.

An important detail during the modeling process should be mentioned: in Figure 25 the internal structure of a mitochondrion VRML97-file is shown. It is important that the order of the different shapes always moves from the inner layer (in case of the mitochondrion the matrix) to the outer layer (the cloud). There are two possibilities for achieving this. First, the modeling process has to maintain the order of the shapes which means, the first shape modeled in 3ds Max is the matrix, etc. Alternatively, the exported VRML97-file can be edited with a regular text editor to match the structure shown in Figure 25.

Another interesting feature of CellExplorer should be mentioned: it is also possible to load a custom VRML97-model into the environment. For this purpose, the Unknown cell component can be utilized.[74]

73 The discussed model is: prokaryote_cell__1_0[Cm3]

74 Regular application cases for the Unknown cell component as well as the reason for the requirement to preserve the order of the cell component layers will be discussed in Sections 6.3.2.3 and 6.3.1.6.

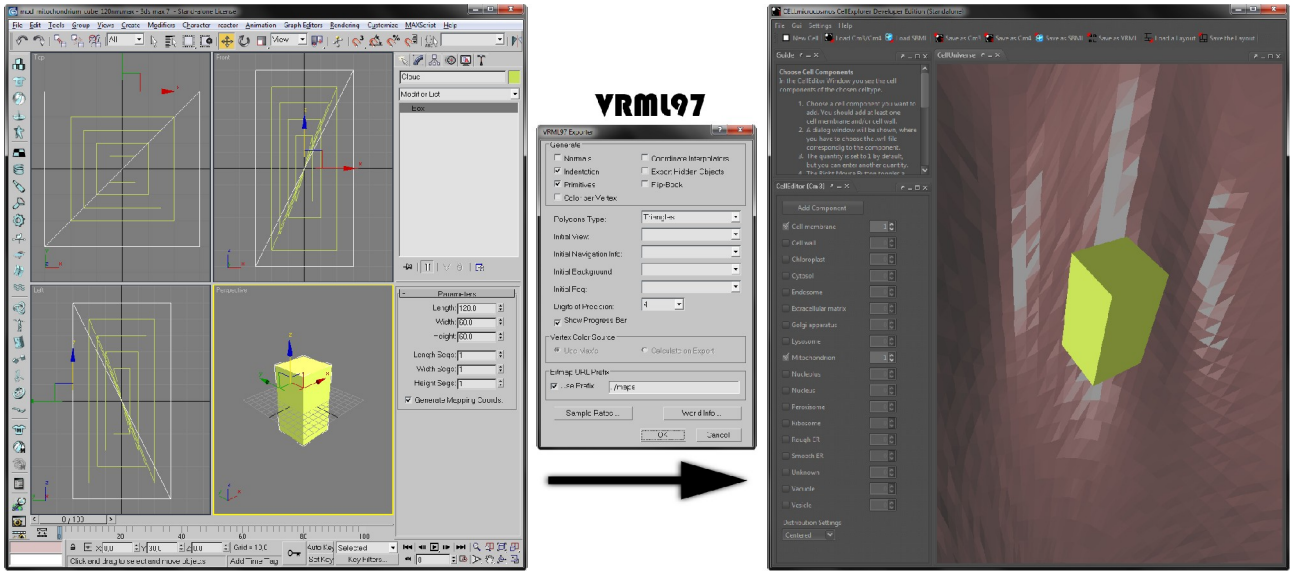


Figure 123: 3ds Max to CellEditor: a simple box-based mitochondrion
It is constructed in 3ds Max and directly exported via its VRML97-export plugin to CellExplorer

6.1.6 VRML export of the whole cell environment

CellEditor was extended by a special VRML97-export feature, enabling the export of any cell environment (Sections 5.1.2.7). Figure 124 shows the model discussed in Section 6.1.2. It was exported to the VRML97-format and visualized with the Cortona3D-browser-plugin, a well-established VRML-viewer which was used here in conjunction with Mozilla Firefox [11n].

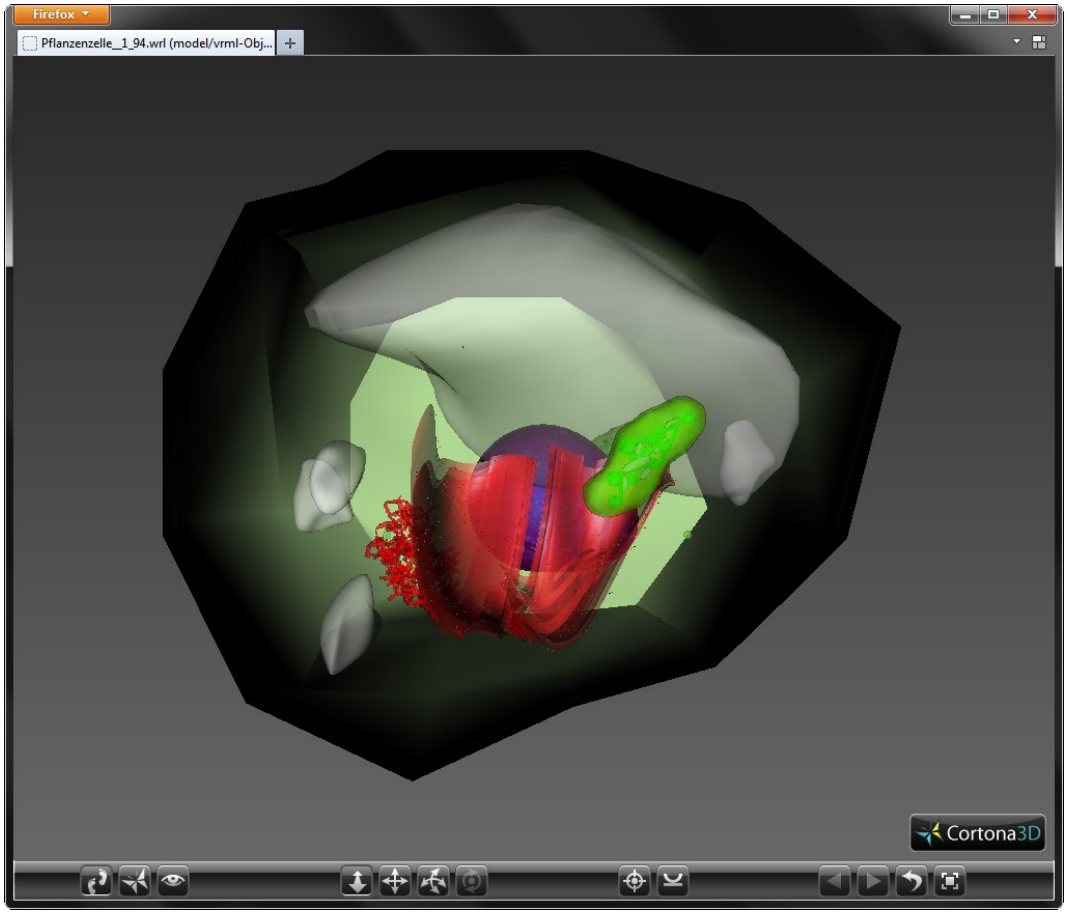


Figure 124: Cortona3D 7.0 with Firefox 9: the plant cell exported from CellExplorer

6.1.7 Cell Models for Exhibitions

Figure 125: The plant cell as it was presented at the "Grüne Woche 2010" With the Frauenhofer 3D Kiosk (imitation of the stereoscopic effect using the plant cell) (Courtesy of/Copyright © 2010 by BIOCOM Projektmanagement GmbH. Reprinted with permission)

Obviously the modularity of CellEditor is paying off. The first plant cell was designed for an exhibition of the Bundesministerium für Bildung und Forschung (BMBF) at the "Grüne Woche" in Berlin (15.-24.01.2010). It was created with CellEditor and exported to the VRML97 format (see the preceding section) to be used in conjunction with the so-called "3D Kiosk", a autostereoscopic presentation system developed by the Frauenhofer Institute [10c].[75]

6.1.8 Cell Models for Education

Another very important application case for cell models is the educational sector. CellExplorer was used as a base for a special edition called CE³; CellExplorer Educational Edition. This software was developed during the CELLmicrocosmos Cell Modeling project summer semester 2011, winter semester 2011/12 and during the master theses of Marco Civico and Roland Orlik [Civi11, Orli12].

Figure 126 shows the CE³. The Cell Universe in the center – where the plant cell is located at – was extended by a *HUD*, providing information about the navigation modi (Section

75 See also Section 5.3.2.11 for an introduction to 3D Stereoscopy.

5.3.2.10) and a two-dimensionally projected map of the cell model. On the right side the *InformationBrowser* is attached, showing texts about the active cell model. Simultaneously, many features of CellExplorer/CellEditor, especially those discussed in Subchapter 5.3, were removed or hidden to focus in CE³ only for the functions needed for the daily course work in school.

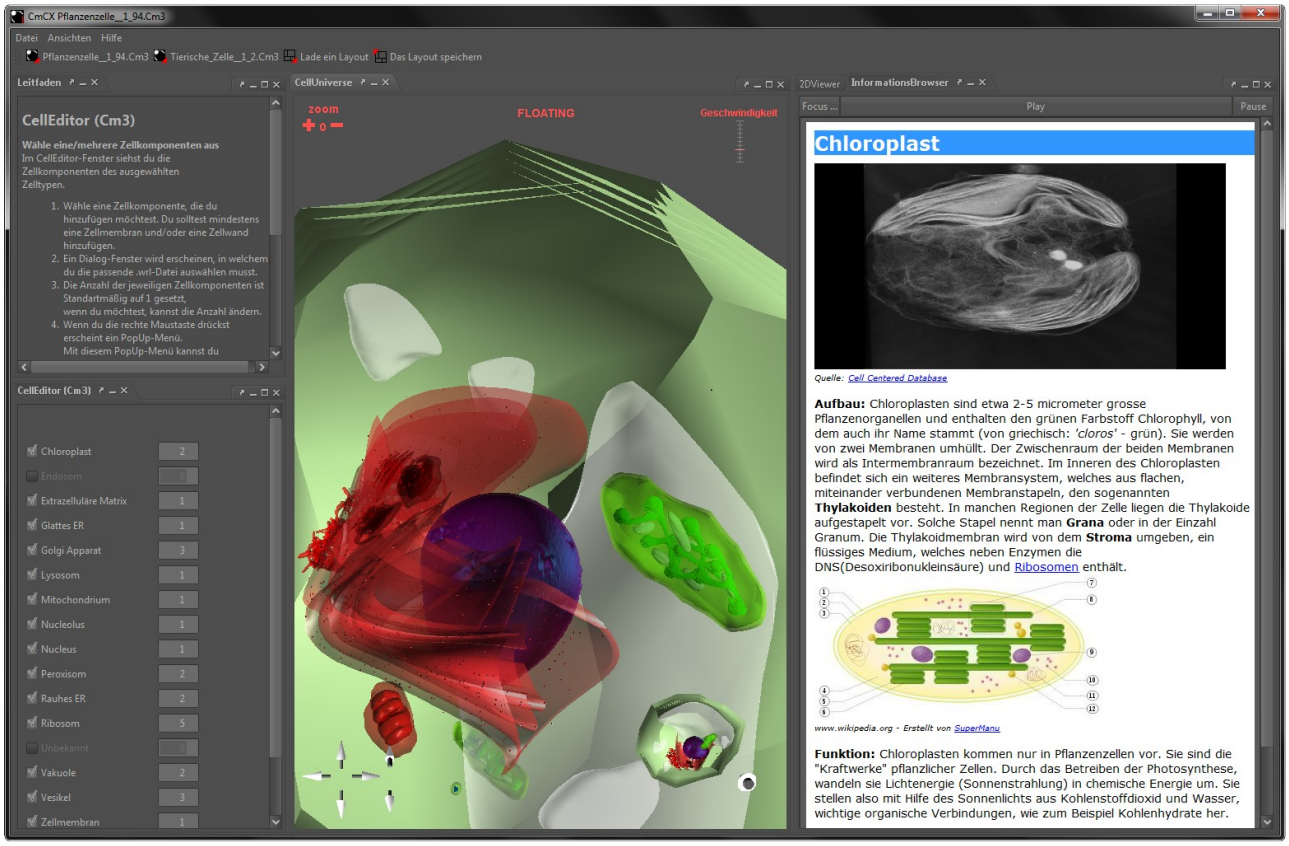


Figure 126: CE³: the plant cell in the center is accompanied by an information browser It provides German text. Here a text concerning the chloroplast is shown

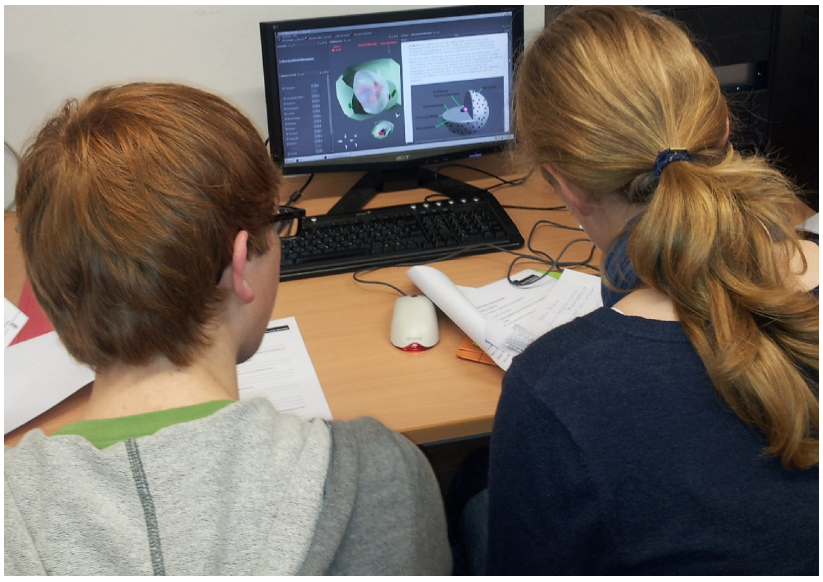


Figure 127: The plant cell used for the evaluation of CE³ At the Schloß Holte-Stukenbrock Gymnasium in autumn 2011 (Courtesy of/Copyright © 2011 by Marco Gomez Civico. Reprinted with permission)

The software was successfully applied to examinable course work during two test periods in two different schools, the *Niklas-Luhmann-Gymnasium Oerlinghausen (NLG)* in 2010 and the *Gymnasium Schloß Holte-Stukenbrock (SHS)* in 2011. For this purpose, Civico and Orlik developed questionnaires based on the current cytology curriculum for school students of the tenth grade. They contained questions to support the exploring and learning

aspects of the cell as well as evaluation questions regarding the functionality of CE³. In Figure 127 two scholars are shown working with the plant cell environment and the questionnaire.

During the two evaluation periods, the practicality of the different cell component models – exceptionally ABS_LEV_2 – has been proven. The results of NLG from 2010 are shown in Figure 128) and those of SHS from 2011 in Figure 129 [Civi11]. Obviously, the applied modeling techniques very well met the expectations created by the literature used in these classes. In addition, CE³ was an excellent support for the school students in understanding the spatial intracellular relationships. There was also positive feedback for the interplay CE³/questionnaire, the CELLmicrocosmos navigation as well as the newly-generated HUD.[76]

In addition, CE³ won a poster prize at the [Science Fair] at the <perspektive> of Bielefeld University in 2011. It was endowed with 1,000 EUR which is being currently reinvested in a student research assistant improving the project [11q].

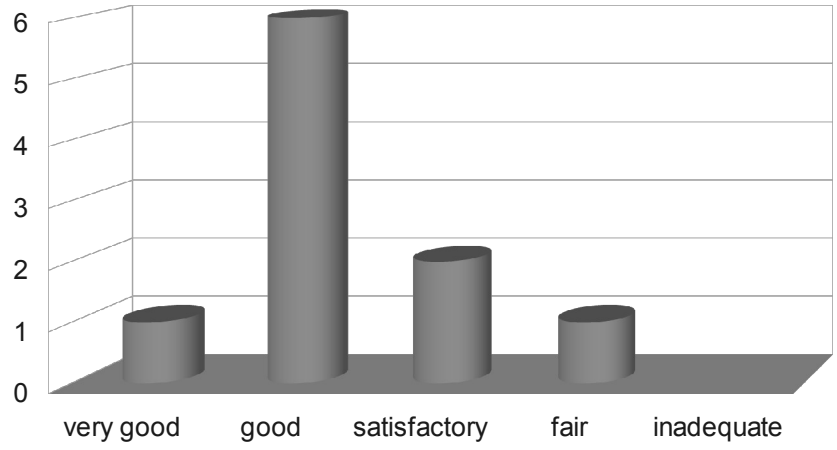


Figure 128: Test results 2010: How far could you identify the cell components in CE³?

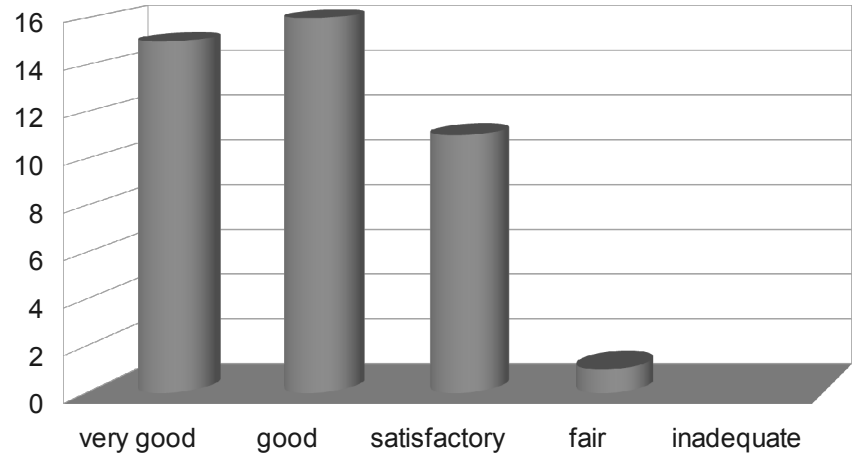


Figure 129: Test results 2011: How far could you identify the cell components in CE³?

76 An extensive discussion and an evaluation of CE³ was done in the master thesis of Marco Civico: "Entwicklung und Evaluierung der Educational Edition des CELLmicrocosmos CellExplorer" [Civi11]. The main publication is planned for 2013 after the third evaluation period is finished in 2012.

6.2 [MOL] Cm2 MembraneEditor

Based on the methods discussed in Subchapter 5.2, the following sections will discuss different membrane modeling approaches using MembraneEditor in conjunction with different Membrane Packing Algorithms (MPA) introduced in Section 5.2.2.5. The focus will be made on The Wanderer, the algorithm developed in the context of this work (Section 5.2.2.6), which will be compared with the aforementioned standard algorithms.[77]

Section 6.2.1 discusses the modeling of an inner and outer mitochondrial membrane of a rat hepatocyte taking different aspects of the realistic membrane composition into account. In Section 6.2.2 the Lipid Packing Algorithms (LPA) are utilized to simulate a typical 2D-KP. Also Protein Packing Algorithms (PPA) are taken into account by discussing the placement of a membrane-spanning as well as a single α-helix-anchored protein into the formerly created mitochondrial membrane in Section 6.2.3. The ability of the algorithms to create high density membranes is discussed in Section 6.2.4. Another unique ability of MembraneEditor is analyzed in Section 6.2.5: the generation of a cholesterol sphingomyelin lipid raft-containing plasma membrane.

6.2.1 Modeling of inner and outer mitochondrial membranes

The species- and organelle-specific percent lipid ratios needed to define the membrane composition can be derived from many references. This section discusses a membrane composition found in mitochondrial membranes, more precisely the inner and outer membrane of the mitochondrion (Sections 2.2.1.3 and 2.1.3.10). The base for the creation of this membrane is a publication by Hovius et al. [HLNK90]. This publication introduced a special method to distinguish the inner from the outer mitochondrial membrane in order to get the appropriate compositional values.

The mitochondrial membrane is a sophisticated task for the verification of distribution methods for several reasons:

- it consists of two separate bilayers with different lipid compositions,
- it contains Cardiolipin, which is a relative large lipid and which is predominantly found in the inner membrane, where it is believed to be synthesized [ScHa93], (Section 2.2.6) and
- it contains cholesterol, which is a relative small lipid (Section 2.2.1.3).

The HIC-Up database was used to obtain the following experimentally derived lipid models in PDB format (Section 3.2.2.2.1):

- *cardiolipin (CL):* cdn_exp.pdb,
- *phosphatidylcholine (PC):* psc_exp.pdb,
- *phosphatidylethanolamine (PE):* pew_exp.pdb,
- *phosphatidylinositol (PI):* pib_exp.pdb,
- *phosphatidylserine (PS):* ps2_exp.pdb, and beginning with Section 6.2.1.2
- *cholesterol (Chol):* clr_exp.pdb (Figure 130).

77 The whole chapter 6.2 is based on the publication [SDGS11].

The following three sections are concerned with analyzing if MembraneEditor is able to generate membranes featuring realistic lipid ratios using these lipid models.

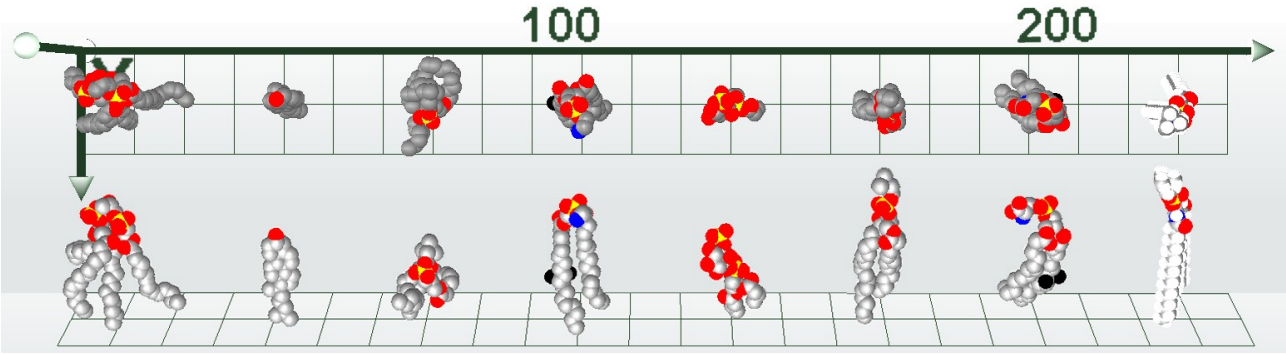


Figure 130: Mitochondrial lipid models
The mitochondrial membranes were constructed by using of eight different lipid types: cardiolipin (CL), cdn_exp; cholesterol (Chol), clr_exp; DLPC, dlp_exp; phosphatidylethanolamine (PE), pew_exp; phosphatidylinositol (PI), pib_exp; phosphatidylcholine (PC), psc_exp; phosphatidylserine (PS), ps2_exp; SM, sphingomyelin [HLNK90, LeSa68] (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

6.2.1.1 Modeling an outer mitochondrial membrane

Table 16 shows the values for the outer mitochondrial membrane as experimentally derived by Hovius et al. [HLNK90]. Summing up the percent values for the different lipid types results in an overall value of 98,9%, 1,1% less than 100%. For this reason the values of the different lipids have to be recalculated to achieve 100%. During this process it is important to stay within the limits defined by the standard deviation (SD). Because PC has a relatively large SD value of 5%, the surplus 1.1% where added to PC only. Table 17 presents the results of the computation using different algorithm of MembraneEditor. Atom collisions were used due their higher accuracy (Section 5.2.2.3).

Figure 131 provides a good overview of the quality of the different LPA. The computed values of all lipid types by every algorithm shown here lie within the limits defined by the SD values. Focusing for instance, on PS which has the lowest SD value with $1 \pm 0.3\%$, the lowest value was computed by the AP algorithm with 0.984%, the highest by the AP+ with 1.058%. Even the simplest algorithm LP does not exceed the limits with a value of 1.04%.

Table 17 shows also the sum of deviations from the initial lipid percentages. Obviously The Wanderer (TW) features the best deviation sum 0.281. Only the *Advanced Random Placing+ (AP+)*[78b] with a deviation sum of 0.288 is close to this value.

In Section 3.2.7 lipid densities were introduced which where observed under different conditions. Moreover, in Section 5.2.2.10 the incorporation of D_l into MembraneEditor was introduced. Now this factor will be evaluated by comparing the different average lipid density values indicated in Table 17.

	Outer Mitochondrial Membrane		Inner Mitochondrial Membrane	
Lipid Type	Lipid Composition (%)	Standard Deviation (%)	Lipid Composition (%)	Standard Deviation (±%)
cardiolipin	9.0	± 2.0	13.7	± 2.3
phosphatidylcholine	48.0	± 5.0	39.2	± 0.9
phosphatidylethanolamine	31.0	± 1.6	43.4	± 1.1
phosphatidylinositol	9.9	± 1.7	1.0	± 0.2
phosphatidylserine	1.0	± 0.3	0.0	± 0.0
remainder	1.1		2.7	

Table 16: The Composition of the inner and outer mitochondrial membrane As reported by Hoviius et al. [HLNK90]

	experim.%	CmME initial values (%)	Linear Placing	Random Placing[a]	Adv. Rand. Placing	Adv. Rand. Placing+[b]	The Wanderer[d]
CL	9±2	9	9.012	8.805	8.886	8.929	8.91
PC	48±5	49.1	48.7	49.161	49.171	49.074	49.121
PE	31±1.6	31	31.196	31.027	31.043	30.952	30.95
PI	9.9±1.7	9.9	10.052	9.958	9.953	9.987	9.965
PS	1±0.3	1	1.04	1.048	0.984	1.058	1.055
deviation sum[e]			0.8	0.389	0.297	0.288	0.281
Number of Lipids			282/ 295	464/ 490	425/ 419	761/ 751	851/ 855
Average Lipid Density (Å^2)			141.84/ 135.59	86.20/ 81.63	94.11/ 95.46	52.56/ 53.26	47/ 46.78
Computation Time[f]			4s [4078]	1m [60672]	13s [13343]	39m 37s [2377344]	27m 24s [1644250]

Table 17: Algorithmic lipid compositions in outer mitochondrial membrane of rat hepatocyte [HLNK90, SDGS11][78]

78 The CmME membrane compositions were tested on a 200x200 Å^2 membrane with standard algorithm values and atom collision only. The values represent the average of both layers. The seed for the random number generation was 22. Special algorithm settings were as follows: [a] stopped after 1 min, run with visualization; [b] with superb density distribution; [c] advanced initial state, automatic adding of lipids activated; [d] wander for fellow and dance option, 50 possible missteps, this algorithm is not included in CmME and can be downloaded separately; [e] based on the CmME initial values; [f] test configuration: Intel® Pentium® 4; 3 GHz (no HyperThreading); 2 GB Ram (Java Virtual Memory: 1 GB); Windows XP SP3; Java 1.6.0_21; Java 3D 1.5.2 fcs (build4); numbers in brackets indicate ms; computation without visualization (except [a]).

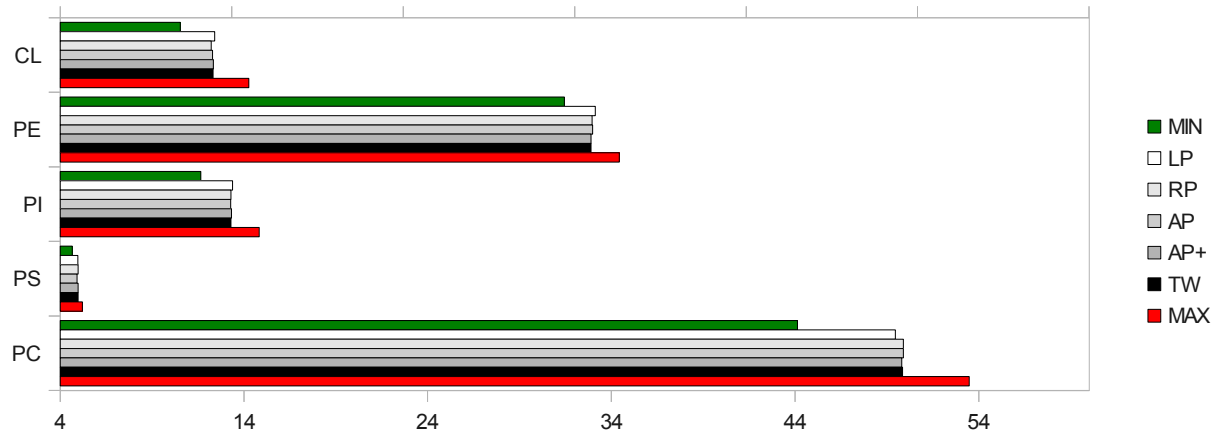


Figure 131: Comparison of initial percent values and values from CmME algorithms MIN: minimal %, MAX: maximal %, LP: Linear Placing, RP: Random Placing, AP: Advanced Random Placing, AP+: Advanced Random Placing Superb; TW: The Wanderer

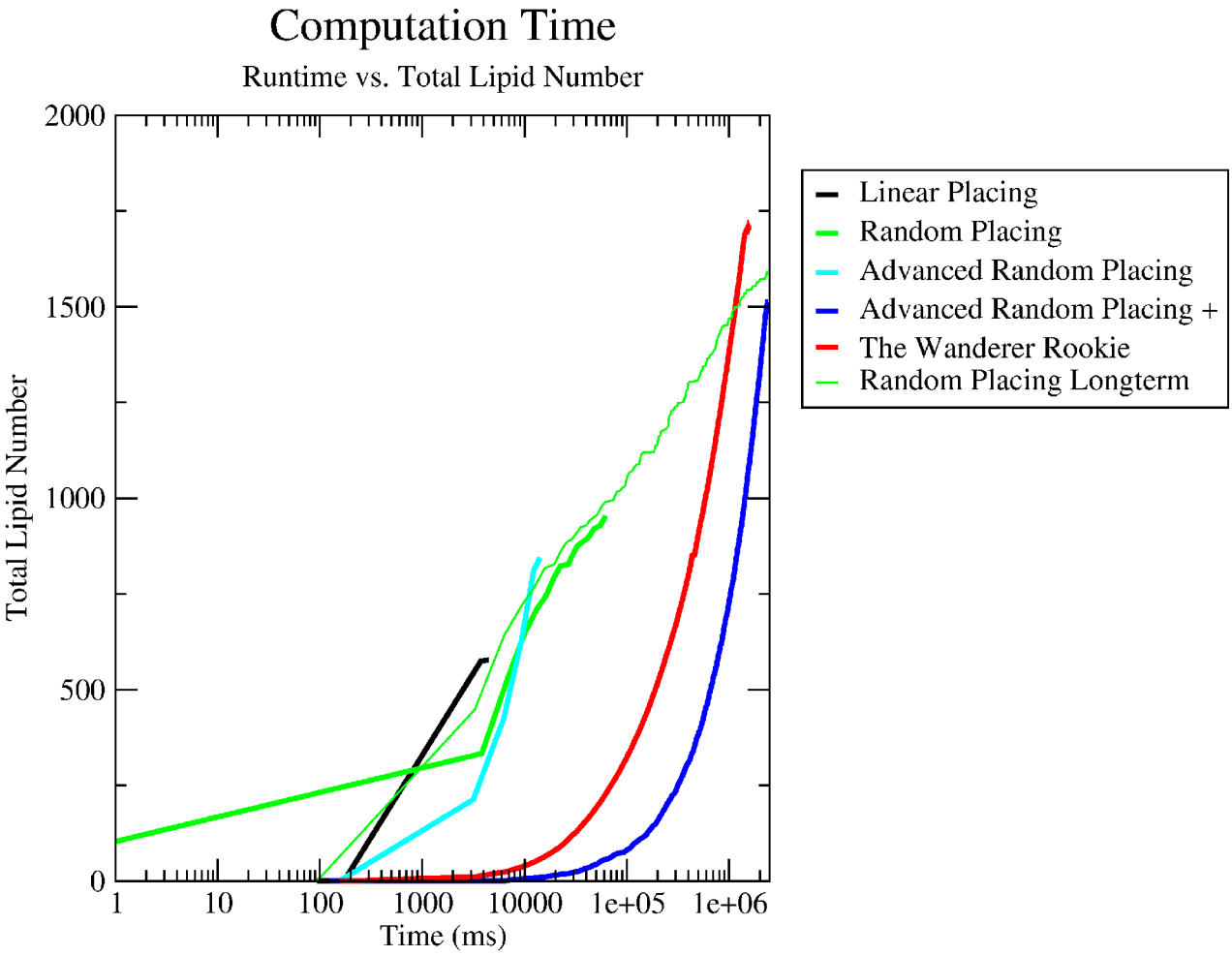


Figure 132: The computation time of the algorithms vs. the total lipid number The time is presented on a logarithmic axis. The graphic was created with xmgrace. See Table 17 for the precise values

The worst value regarding the lipid density is computed – as expected – by the Linear Placing algorithm (LP). The algorithms Random Placing (RP) and Advanced Random Placing with standard settings (AP) produced membranes with a better density of ~82 to

95 Å^2 which is still not satisfactory. But packing values close to experimental values are achievable: AP+ with an average of ~53 Å^2 and TW with ~47 Å^2, needing 40 and 27 minutes respectively.

Referring to Nagle and Tristram-Nagle [NaTr00], the area per lipid for DPPC at 20°C in the gel phase would occupy ~47 Å^2. This value was realized here by TW. At 50° in the fluid phase, an area of 56-72 Å^2 was reported. Values close to this were generated by AP+ (Figure 132).

The total number of lipids in relation to the algorithmic computation time was recorded during runtime and visualized externally with the program (Xm)grace. The results are shown in Figure 132. The time axis is scaled logarithmically. Additionally, for the values discussed in Table 17, a Random Placing algorithm was run to be comparable to the runtime of AP+ (see Figure 132, Random Placing Longterm). Obviously AP+ (but not TW) is slightly outperformed by Random Placing Longterm. The reason may be found in the method used by AP+ to subdivide the membrane into grid-like patches (Section 5.2.2.5). The RP process of redistributing the lipids is similar to equilibration or annealing processes, leading to a slightly better mixing of the overall membrane. Another interesting observation is that the graph of RP shows that this algorithm is a little bit faster than the other algorithms in the beginning. But it gets a bit slower than Random Placing Longterm in the end. The reason is that – in contrast to all other algorithms – RP was run with visualizations shown to be able to stop the algorithm exactly after one minute. Obviously the process of hiding MembraneEditor in the starting process demands some time, approx. 100-200 ns, but during the following computation time it outperforms the visualization-supported runs.

6.2.1.2 Incorporation of cholesterol

As of yet, only phospholipids have been incorporated into the membrane (Section 2.2.1.3). Now cholesterols have to be integrated. Levy and Saudner reported a value of 10% for cholesterol (Chol) in a rat mitochondrial membrane [LeSa68]. Therefore the values stated in Table 17 were reduced each by 10%, resulting in the following percent distribution: CL: 8.1, PE: 27.9, PI: 8.91, PS: 0.9, PC: 44.19 and (Chol): 10%. For the generation of the membrane TW was used (see Table 17 for settings, Table 18 for results and Figure 133 for the visualization in MembraneEditor).

To judge the quality of the incorporation of cholesterol, experimental observations should be taken into account. A well-known theory regarding the impact of cholesterol on membranes is the umbrella theory (Section 2.2.1.3) [HuFe99]. As previously discussed, it predicts that cholesterol tends to be shielded by the larger head groups of phospholipids to avoid energetically disadvantageous contact with water molecules surrounding the membrane leaflets. Another theory, which is in affirmation with the umbrella theory, is called the *cholesterol condensing effect*. It states that the surface area of a cholesterol-containing membrane is less than the sum of the individual lipid areas [AlDH10]. The average lipid density using TW decreases now from ~47 to 43.5 Å^2 due to the small size of cholesterol.

Figure 134 shows a visual analysis of the umbrella effect taking only the upper leaflet of the mitochondrial membrane into account. Table 18 shows that 92 lipids from 918 are cholesterol (10.038%).

Figure 134.1 shows the molecules in simple shape-based mode and Figure 134.2 the same molecules visualized with the complex and time-consuming van der Waals radii visualization. The amount of the visible cholesterol, the yellow areas, in both figures is nearly the same. Obviously the shape-based visualization is an appropriate alternative to atomic visualizations. In Figure 134.3 all cholesterol molecules of the upper membrane leaflet are colored yellow in covalent radii mode. Figure 134.4 is a combination of the cholesterol highlighting in Figure 134.3, yellow areas, and Figure 134.2, blue areas. It shows that 17, in particular the pure yellow areas, out of 92 lipids (approx. 15%) were completely hidden beneath the larger headgroups of the phospholipids. The atomic detail view from top and east perspective in Figure 134.6 show a single cholesterol (yellow) shielded by the surrounding phospholipids which is marked in Figure 134.4.

	initial values Outer Membrane (%)	The Wanderer	initial values Inner Membrane (%)	The Wanderer
Chol	10	10.038	10	10.044
CL	8.1	7.922	12.681	12.599
PC	44.19	44.221	36.252	36.3
PE	27.9	27.889	40.131	40.088
PI	8.91	8.953	0.936	0.969
PS	0.9	0.977	-	-
Number of Lipids		918/925		568/567
Average Lipid Density (Å^2)		43.57/43.24		70.42/70.54

Table 18: Lipid compositions with cholesterol in outer and inner mitochondrial membrane of rat hepatocyte
[HLNK90, LeSa68, SDGS11][79]

6.2.1.3 Addition of an inner mitochondrial membrane

Based on the experimental results from Hovius et al. the inner mitochondrial membrane has to be added to the membrane model [HLNK90]. 97.3% of the lipids are defined by the reported values: CL 13.7±2.3, PC 39.2±0.9, PE 43.4±1.1 and PI 1.0±0.2. To prevent the exceeding of SD values, the remainder of 2.7% was added to each of the values proportionally, resulting in: CL 14.09, PC 40.28, PE 44.59 and PI 1.04. Finally, the method of the former section was used to add cholesterol to the membrane (see Table 18). A publication by Yamaguchi et al. defines the distance between the inner and outer membrane of a mouse liver mitochondria as approximately100 Å [YLPS08]. This distance was also chosen for the mitochondrial rat hepatocyte membrane (see also Figure 136).

79 The CmME membrane compositions were tested on a 200x200 Å^2 membrane with standard algorithm values and atom collision mode only. The values represent the average of both layers. The seed for the random number generation was 22. Special algorithm settings: wander for fellow and dance options.

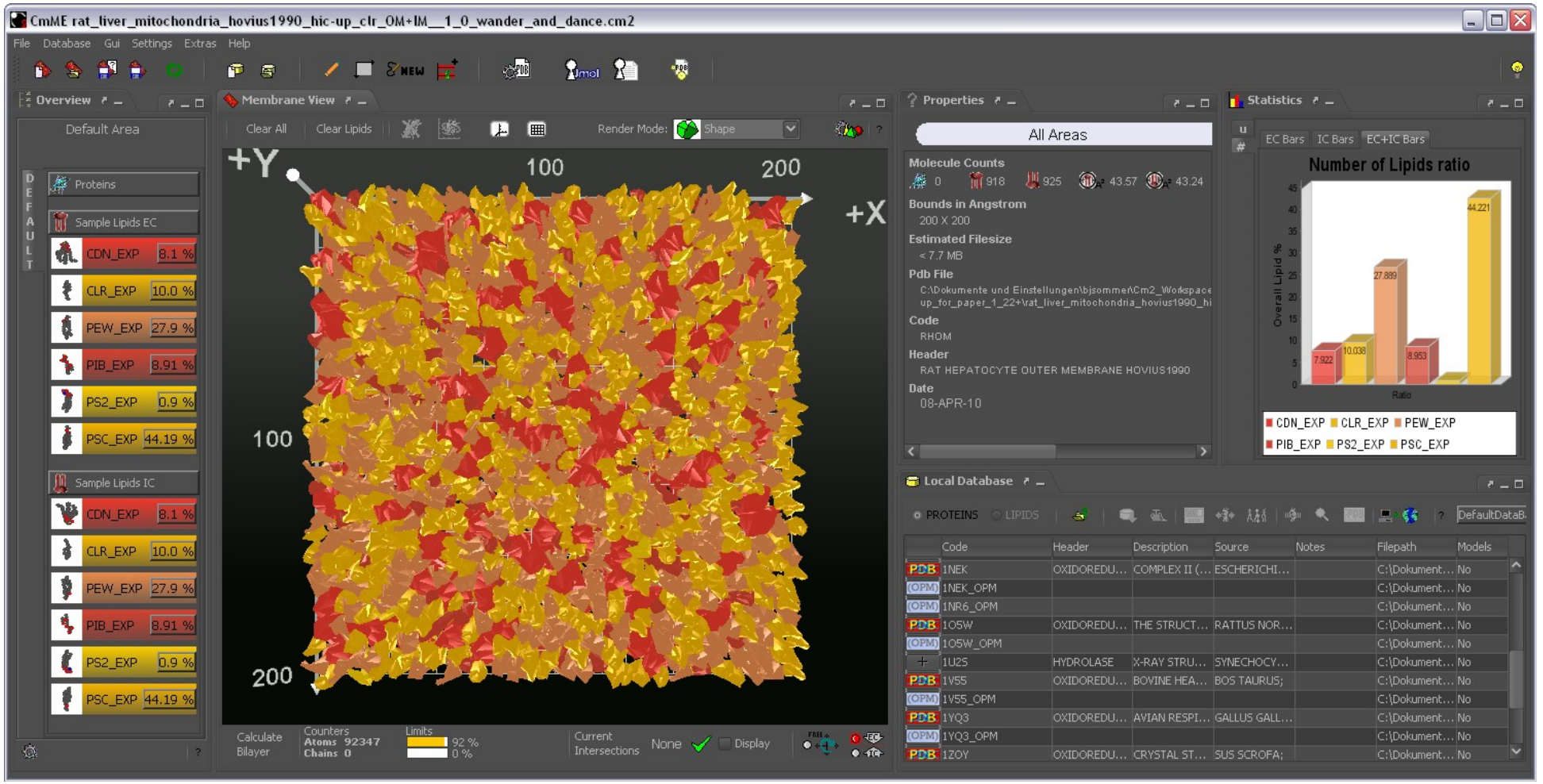


Figure 133: MembraneEditor: The GUI and the outer mitochondrial membrane
The membrane was created from experimental lipid data from the HIC-Up database. It is shown in MembraneEditor version 2.2.0 (see Table 18 for data). The six lipid types are indicated by different colors. The initial percent ratios are shown on the left side, the resulting percent ratios by the diagram on the right side. Left of this diagram, information about the actual membrane model is indicated, as are data on the average lipid density and the molecule numbers. The bottom shows the Local Database for proteins and the Membrane View in the center the 200x200 Å² membrane model in shape mode. TW was used for membrane generation (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

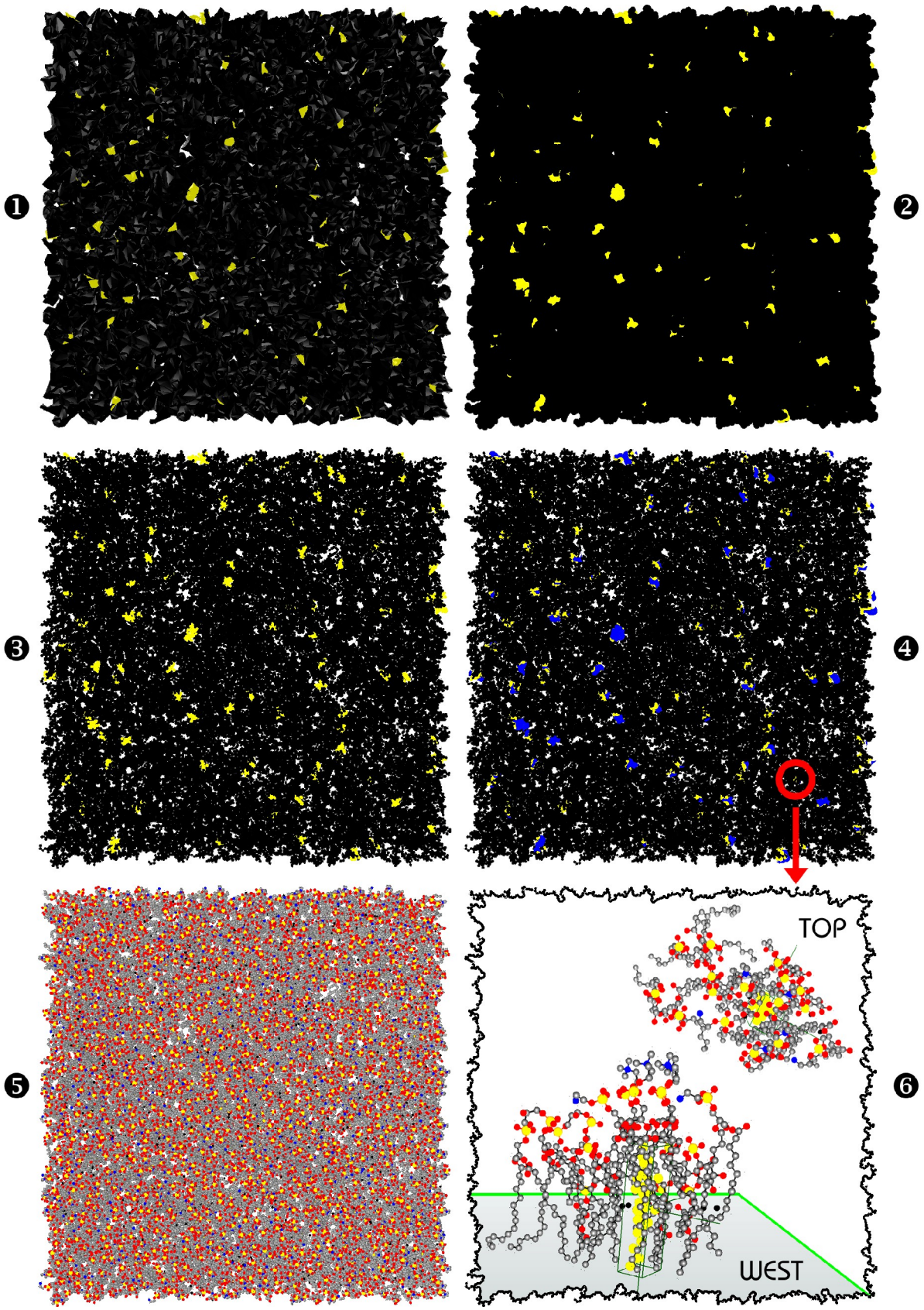


Figure 134: Different visualization modes of a membrane containing 10% cholesterol (colored yellow): 1. shapes; 2. van der Waals radii; 3. covalent radii; 4. combination of 2 (blue areas) and 3 (yellow areas): the yellow headgroups are completely shielded by the larger headgroups of other lipid types; 5. atom type-dependent coloring; 6. the molecular environment of the cholesterol marked in 5 shown from top and west perspective (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

6.2.2 Application to the Classical 2D-Knapsack-Problem

Initially the 2D-KP was defined as the appropriate term to describe LPP with *regular* three-dimensional shapes (Section 3.2.6). Then, we extended the definition to the 2.5D-KP for *irregular* three-dimensional shapes (Section 5.2.2.5) which was accountable for all application cases discussed before. In order to close the gap between both approaches, now the appliance of the classical 2D-KP to LPP should be discussed.

As an additional constraint besides the usage of regular, rectangular 2D shapes, the orientation restriction to 90° steps is introduced. This constellation is found in many of the classical two-dimensional packing and/or KP solutions [Dyck90, DyFi92, LoMM02].

The optimal solution differs from the one of the more complex 2.5D-KP and can be easily defined as the ideal membrane area ratio of 100% (OPT_LPP_2D_AREA).

Of course, the original shape of a lipid is not a box. To overcome this obstacle, a special MembraneEditor plugin-tool (Section 5.2.2.7) was developed: the *Molecule Boxifier*. This tool shifts the atoms of a lipid molecule to the edges of the lipid's bounding box. This is done for the lipids CL, PC, PE, PI and PS. Because only the outer shapes of the resulting box are relevant, shape collision is used exceptionally (Section 5.2.2.3).

Now the algorithms have to be evaluated concerning their compatibility to the restrictions stated above. The algorithms LP, AP and TW already provide the option to rotate lipids in 90° steps along the Y-axis.

Because MembraneEditor does not natively support the measuring of bounding boxes, another plugin-tool was developed, the *Dimension Lister*. It determines the membrane area ratio by summing up the rectangular areas of every lipid and comparing them to the overall size of the membrane.

The results of six different computations are shown in Table 19. The gray column indicates the only exception to the other five algorithms, the AP+ as a reference to the 2.5D-KP. As in the preceding sections it is run in superb mode without rotation restriction to 90°. It results in the second worst membrane area ratio of 51.6%. Obviously, the packing of rectangular boxes needs an orientation restriction, although the AP+ generated the second highest density in the context of the 2.5D-KP (Table 17). The AP(90°) with standard values achieved the lowest density and is outperformed by the LP with 55.26%. Remembering Section 5.2.2.5 brings back the fact that LP already uses a bounding-boxes-based, row-by-row-proceeding packing procedure. In contrast to the former sections, LP beats AP. But the best solution was achieved once again by TW(90°) with nearly 80% (Figure 135). In addition, the resulting lipid distribution percentages for TW are within the SD values stated in Table 17: CL: 8.705%, PC: 49.107%, PE: 31.027%, PI: 10.045% and PS: 1.116%. Therefore, good solutions were found for OPT_LPP_RATIO and OPT_LPP_2D_AREA.

Finally one might say that a set of algorithms developed to solve LPPs generated good solution for the classical 2D-KP, especially TW which generates again the best D_l. The only additional precondition is the restriction to 90°-rotations. What are now the consequences of these observations for the LPP, or 2.5D-KP respectively?

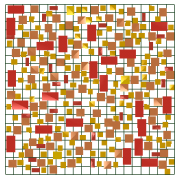
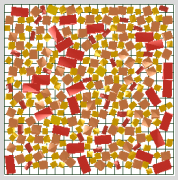
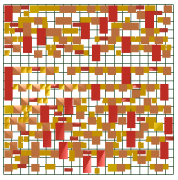
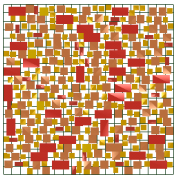
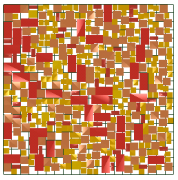
	 Adv. Rand. Placing (90°)[a]	 Adv. Rand. Placing+[bc]	 Linear Placing (90°)[d]	 Adv. Rand. Placing+ (90°)[ce]	 The Wanderer (90°)[f]
Number of Lipids	256	289	311	358	448
Average Lipid Density (Å^2)	156.25	138.40	128.61	111.73	89.28
Membrane Area Ratio (%)[g]	45.37	51.60	55.28	63.57	79.65
Computation Time[h]	2s [2078]	8m 49s [529703]	1s [1546]	6m 54s [414953]	25s [25485]

Table 19: Lipid Density values in a boxified monolayer
Based on the outer mitochondrial membrane of rat hepatocyte [HLNK90]

First, it has to be stated that the classical 2D-KP approach aims at the maximization of the membrane area ratio, while LPP tries to minimize the area per lipid. One not surprising outcome is that the 2D-KP cannot compete with the LPP in terms of the average area per lipid: 89.28 Å^2 in Table 19 face in case of TW ~47 Å^2 in Table 17. In addition, 89.28 Å^2 lies outside the experimental bounds (Section 3.2.7). This means practically that the computation of a membrane exclusively using the lipid's bounding boxes would not make sense.

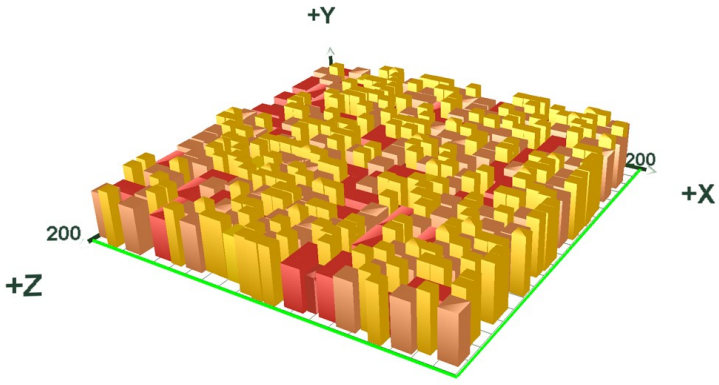


Figure 135: A classical 2D-KP membrane generated with The Wanderer
See Table 19 for details. The lipid height is not taken into account during computation, only the width and length are relevant (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

Moreover, the following assumption should be discussed: What is the impact of the 2D-KP if the bounding boxes describe exactly the shape of the lipid and they all have the same shape? For example, this may be the case if a lipid model consists only of a vertical alignment of atoms which have all the same size. In addition, the length of the membrane area should be a multiple of the lipid's bounding-box length. In this case, a membrane area ratio of 100% would be achievable. But the additional result would be an extremely high and unrealistic lipid density value.

In the previous sections it was shown that LPAs are able to achieve higher area per lipid ratios then needed for regular application cases[80]. Therefore, in this special theoretical case, the optimal solution of a 2D-KP is not needed to solve the regular LPP discussed in the previous section, because the realistic average area per lipid would be exceeded dramatically.

80 Exceptions will be discussed in Section 6.2.4.

6.2.3 Protein Placement

Different approaches to insert proteins into a membrane, addressing OPT_PPP_POS, have been introduced in Section 3.2.5. The PDBTM database was introduced in Section 3.2.2.1.4, and the OPM database in Section 3.2.2.1.5. Proteins can be dragged from the Local Database and dropped into the Membrane View (Figure 133). The justification is done automatically. Then, only one more click is needed to remove all intersecting lipids. Figure 136 shows how two proteins were semi-automatically placed into the membrane.

Cytochrome c oxidase (1V55, Section 2.2.4) is shown on the left side. It was obtained from *Bos taurus [TAYT95]*. On the right side a Monoamine oxidase (1O5W) is shown. It is a marker enzyme specifically for the outer mitochondrial membrane [HLNK90] The PDB model 1O5W was obtained for monoamine oxidase from the outer mitochondrial membrane of *Rattus norvegicus [MYYN04]* It was shown that the placement of these proteins is reliable using CmME in conjunction with the PDBTM matrix [SDGS11].

Another application case of protein packing is shown in Figure 16. All different protein assemblies were combined with a small membrane patch of 660x20 Å^2.

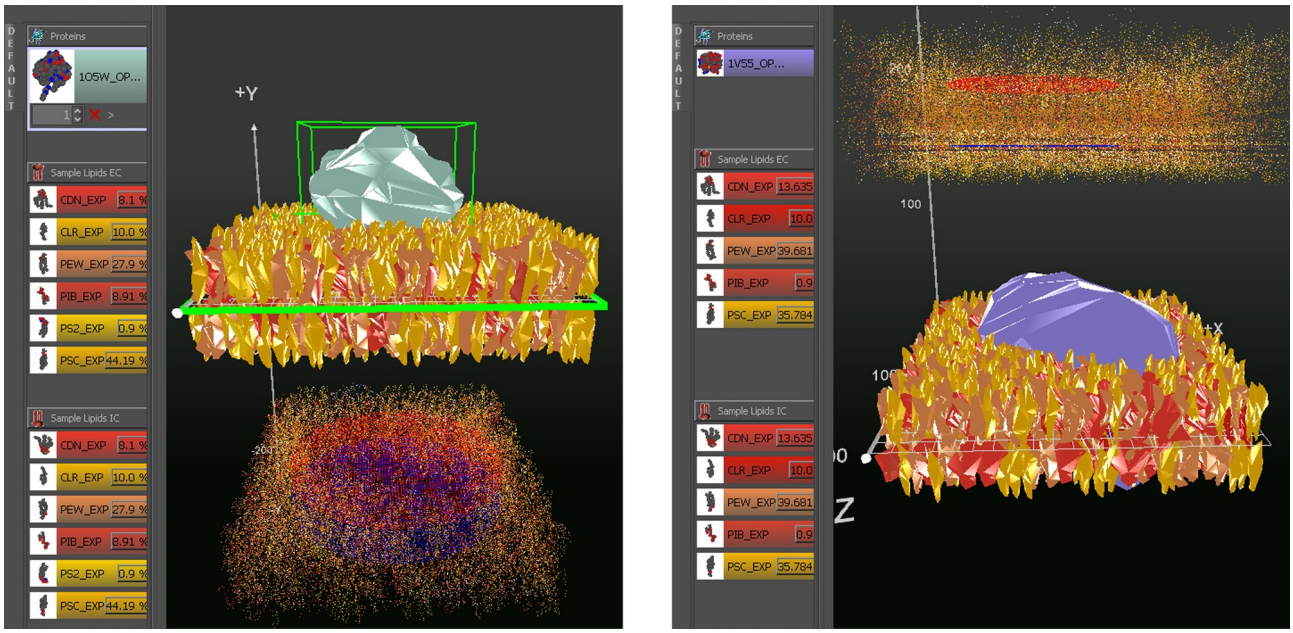


Figure 136: The outer membrane with 1O5W and the inner membrane assembled with 1V55 Left: 1O5W; right: 1V55. OPM data has been used for the semi-automatic placement. (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

6.2.4 Computation of extreme densities

In Section 6.2.1.1 it was shown that it is possible to generate good solutions using LPP for experimentally derived average values. In this section the question should be addressed, if it is also possible to generate appropriate model membranes based on extreme values. These extreme lipid densities were derived from experimental observations by Ghosh et al. [GhWT73].

The following computations were carried out on a 100x100 Å^2 membrane.

Of course, it is possible to compute low densities like those reported for PC – 134 Å^2 – with MembraneEditor. The first option is to use the RP-based Distributor (DI) and pre-define the lipid density; the second option is to alter a precomputed membrane model with the Distributor by thinning out the membrane. This also applies to the following computations.

But problems may occur when relative high lipid densities like the one stated for cholesterol – 38.31 Å^2 – are intended.

As in the preceding sections the PDB structure clr_exp was used. And once again different algorithms were used to compute the membrane model. AP achieved a value of 38.46 Å^2, AP+ a value of 22.42 Å^2, and TW a value of 16.42 Å^2 (see Table 17 for algorithm properties). Obviously it is possible to generate a membrane with the intended density of 38.31 Å^2.

But of course, cholesterol is a relatively small lipid. What happens in the opposite case, if a space-consuming lipid is taken as a base?

Under very extreme experimental conditions a value of 26 Å^2 is reported for DLPC [PeBe83]. This time, a new structure file was downloaded from the HIC-UP website: dlp_exp. Using AP+, an area per lipid of 119.04 Å^2 was realized, using TW, 131.57 Å^2. This time also the high lipid packing density algorithms obviously failed to compute an adequate placing.

Examining the structure of dlp_exp shown in Figure 130 emphasizes the reason for this placing disaster. The tails are far from being fully extended. Instead of this, the lipid is shrunken on the vertical axis and the tails are spread apart.

In such a case, if a dense packing is needed, the lipid has to be manipulated prior to importing it into MembraneEditor or internally the *Molecule Editor* has to be used to change the structure of the lipid [Ding10]. Ideally, the tails would be fully extended and all atoms aligned along the vertical axis.

Another critical limitation besides the lipid ratio is the membrane size. While the former computation took only membranes of an area of 200x200 Å^2 into account, now smaller areas are examined, using side lengths of 50 and 100 Å (Table 20). The simplest algorithm, LP, and the algorithm generating the best D_I values, TW, were used exclusively. For 50x50 Å^2, LP computed values exceeding the SD values for CL, PE, PI and PS, whereby the deviation of the last mentioned value is dramatical with +5.158 above the SD values. On the outer membrane side there were ten lipids, on the inner membrane side only eight, respectively. The extreme low number of lipids did not allow the achievement of the predefined conformation. But TW computed nearly acceptable results: only the SD values for PS were slightly exceeded. For a 100x100 Å^2 membrane, TW produced correct values, while LP failed only slightly for PS.

In summary one can say that there are three special cases which may need special solution approaches:

1. a very high lipid density,
2. a very small membrane area, or
3. an unfavorable atomic conformation.

			50x50 Å^2		100x100 Å^2		200x200 Å^2	
	experim. (%)	initial values (%)	Linear Placing	The Wanderer	Linear Placing	The Wanderer	Linear Placing	The Wanderer
CL	9±2	9	6.458	7.705	8.725	8.696	9.009	8.91
PC	48±5	49.1	45	49.444	49.273	49.104	48.705	49.121
PE	31±1.6	31	29.167	30.773	29.684	30.946	31.196	30.949
PI	9.9±1.7	9.9	12.917	9.879	10.864	10.231	10.049	9.965
PS	1±0.3	1	6.458	2.198	1.454	1.023	1.04	1.055
Number of Lipids			16/ 15	46/ 45	65/ 73	197/ 194	287/ 301	826/ 845
Average Lipid Density (Å^2)			156.25/ 166.66	54.34/ 55.55	153.84/ 136.98	50.76/ 51.54	141.84/ 135.59	47/ 46.78
Computation Time			<1s [218]	29s [29171]	<1s [641]	2m 37s [157687]	2s [2937]	28m 25s [1705766]

Table 20: Area-dependent constraints in lipid composition of model membrane [HLNK90, SDGS11][81]

Possible solutions may be:

1. Algorithms optimized for special lipid structures.
2. Preparative optimization of the atomic structure of the lipid.
3. Internal generation of a larger membrane with MembraneEditor and subsequent squeezing approaches by using e.g. minimization processes of external simulation packages.

It can be stated that again TW generates good results and also very small membranes featuring a size of 50x50 Å^2.

6.2.5 Modeling of a lipid raft-containing plasma membrane

The preceding application cases were based on experimental observations of mitochondrial rat hepatocyte membrane. Now that a lipid-raft should be modeled, the plasma membrane of a mouse hepatocyte is chosen as the starting point, because the existence of lipid rafts in mitochondria is still under controversial discussion [McNW06],[ZhBF09].

81 Lipid compositions in outer mitochondrial membrane of rat hepatocyte [HLNK90] computed on a 50x50, 100x100 and 200x200 Å^2 membrane patch. For further details see Table 17. Gray areas mark results that exceed the limits defined by %.

A cholesterol-sphingomyelin raft was constructed based on the data from Atshaves et al. [AMPG07]. Again, the membrane area was set to 200x200 Å^2 containing a raft featuring a diameter of 180 Å which lies inside the observed size for lipid rafts [PKFS00, PMPH03].

The experimental values from Atshaves et al. were taken from a wild mouse type. The adjustment of the values based on the methods described above resulted in the following values for the surrounding membrane: Chol 21%, PC 36.972%, PS 3.002%, PI 4.345%, PE 18.012%, sphingomyelin (SM) 16.669% and for the inner lipid raft: Chol: 46%, PC 21.06%, PE 11.34%, PI 4.86%, PS 5.4%, SM 11.34%.

The HIC-UP database did not provide experimental nor ideal coordinates for sphingomyelin, therefore the data was downloaded from Chemistry Molecular Models (Section 3.2.2.2.5, Figure 130). The hydrogen atoms contained in the original SM model were manually removed, because the other lipid models also do not contain any hydrogen atoms.

	Membrane[a]	1	3		2	3
	initial values surrounding Membrane (%)	 Adv. Random Placing+[b]	 Adv. Random Placing+[c]	initial values Membrane Raft (%)	 Adv. Random Placing+[b]	 Adv. Random Placing+[d]
Chol	21	21.055	21.123	46	46.013	46.029
PC	36.972	36.933	36.78	21.06	21.089	21.029
PE	18.012	17.998	18.021	11.34	11.285	11.328
PI	4.345	4.389	4.431	4.86	4.88	4.948
PS	3.002	2.909	2.806	5.4	5.359	5.339
SM	16.669	16.716	16.839	11.34	11.373	11.328
deviation sum		0.292	0.704		0.191	0.233
Number of Lipids		1012/ 1016	337/ 340		1152/ 1143	763/ 773
Average Lipid Density (Å^2)		39.52/ 39.37	43.18/ 42.80		34.72/ 34.99	33.35/ 32.91

Table 21: Membrane lipid composition with and without cholesterol-sphingomyelin raft In plasma of mouse hepatocyte [AMPG07, PKFS00] (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])[82]

82 The CmME membrane compositions were tested on a 200x200 Å^2 membrane with Adv. Rand. Placing+ and atom collision only (Table 17). The values represent the average of both layers. [a] 1. a pure bilayer featuring the plasma membrane composition, 2. a pure bilayer featuring the raft composition, 3. a mixed bilayer combining the circular raft with the surrounding membrane composition; [b] Bilayer patch of 200x200 Å^2; [c] Outer Border of the 200x200 Å^2 bilayer patch without Ø180 Å raft area; [d] Ø180 Å raft area.

Based on the initial values stated above, three 200x200 Å^2 membrane patches were computed by using AP+:

1. A pure bilayer featuring the plasma membrane composition.
2. A pure bilayer featuring the raft composition.
3. A mixed bilayer combining the circular raft with the surrounding membrane composition.

In Table 22, the average lipid densities of the different membrane patches are compared. A comparison of the columns on the right side shows that the difference in average lipid density between the pure and mixed bilayer of the raft is 1.7 Å^2. The columns on the left side show a higher difference between the pure and mixed bilayer of the surrounding membrane with a value of 3.5 Å^2. The reasons are twofold: the perimeter of the outer membrane patch has a negative impact on the area per lipid and, in addition, the outer membrane surrounding the raft has two borders.

Despite these small density-related differences, all three membrane patches behave as expected concerning the dependence of the area per lipid on the cholesterol ratio.

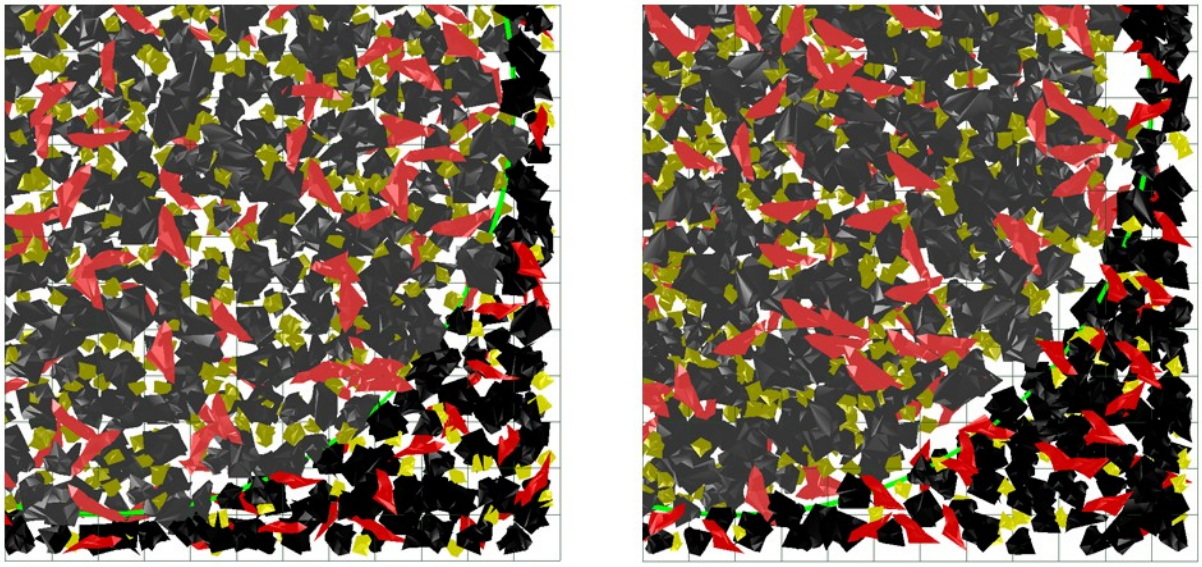


Figure 137: Comparison of a Chol/SM-Raft generated with AP+ and TW
Left: AP+; right: TW. Chol: yellow, SM: red, PC: black (Courtesy of/Copyright © 2011 by American Chemical Society. Reprinted with permission from [SDGS11])

In the previous section, TW achieved the best results. During the raft generation, AP+ was used. The reason for this choice is shown in Figure 137, were membrane patch 3 is shown twice. The left image shows the discussed approach with AP+ and on the right image, the same patch was generated with TW. Again, TW generates a higher lipid density,[83] but the vacancies along the raft borders are apparent. The north-west wandering approach of TW (Section 5.2.2.6) has obviously a bad impact on the south-east border of the raft. Therefore, the AP+ is preferable for consistent lipid distributions if a raft is involved.

83 The values for TW are not explicitly stated here, but Figure 137 already depicts the dense packing.

6.3 [FUN] Cm4 PathwayIntegration

Application cases of the Mesoscopic and molecular level were discussed. Now the underlying functional level will be focused. The relevant methods have been introduced in Subchapter 5.3 and here it will be described how they are used in the context of PathwayIntegration. The name of this tool already announces the leading question: how is it possible to spatially combine biological pathways with the cell?

6.3.1 Localizing metabolic Pathways

As discussed in Section 2.3.4, metabolic pathways are an essential entity of the cell´s function. A central pathway, the citrate cycle, was introduced in Section 2.3.5. In this section the citrate cycle will be used to correlate an animal cell model with two interrelated pathways.

6.3.1.1 Downloading the Citrate Cycle and the Glycolysis

In Section 2.3.5 it was shown that two pathways are directly correlated with the citrate cycle: the glycolysis and the oxidative decarboxylation of pyruvate. There are now two possibilities for importing such a pathway: by generating it with an external program or by directly downloading the respective KEGG-pathways from DAWIS-M.D. (Section 5.3.2.2). Of course, the last option will be chosen. Using CmPI, the pathway hsa00010, the glycolysis, and hsa00020, the citrate cycle, are downloaded. For the oxidative decarboxylation of pyruvate no special entry in the KEGG database exists, because it is part of the citrate cycle. Figure 138 shows the resulting layout for hsa00020 in the 2D Viewer of CmPI (Section 5.3.2.2.3).[84] The layout is based on the original KEGG layouts which are also known from the *KGML (KEGG Markup Language)* pathway maps which can be found at the original website of KEGG (Section 3.3.2.1.1).[85]

A difference from the original KEGG maps is the fact that each pathway can always contain only one entity of each element. The original hsa00020 map as shown in Figure 34 contains the compound C15973 twice. It is connected to the enzyme 2.3.1.61 on the bottom and 2.3.1.12 on the top. In Figure 138 there is only a single instance of compound C15973. Of course, the connections to the involved enzymes are still shown. The reason is the localization-focused view of CmPI: each instance in a pathway has a unique position.

6.3.1.2 First Localization Results

Now the localization of both pathways should be examined. In Table 4 the known localizations of these pathways were already noted. But now it should be evaluated if CmPI is also able to predict the correct localization.

84 The resulting layout of hsa00020 is found in Appendix 9.9.

85 See also Figure 34 to compare the visualization to the original picture of the Citrate Cycle known from the KEGG website.

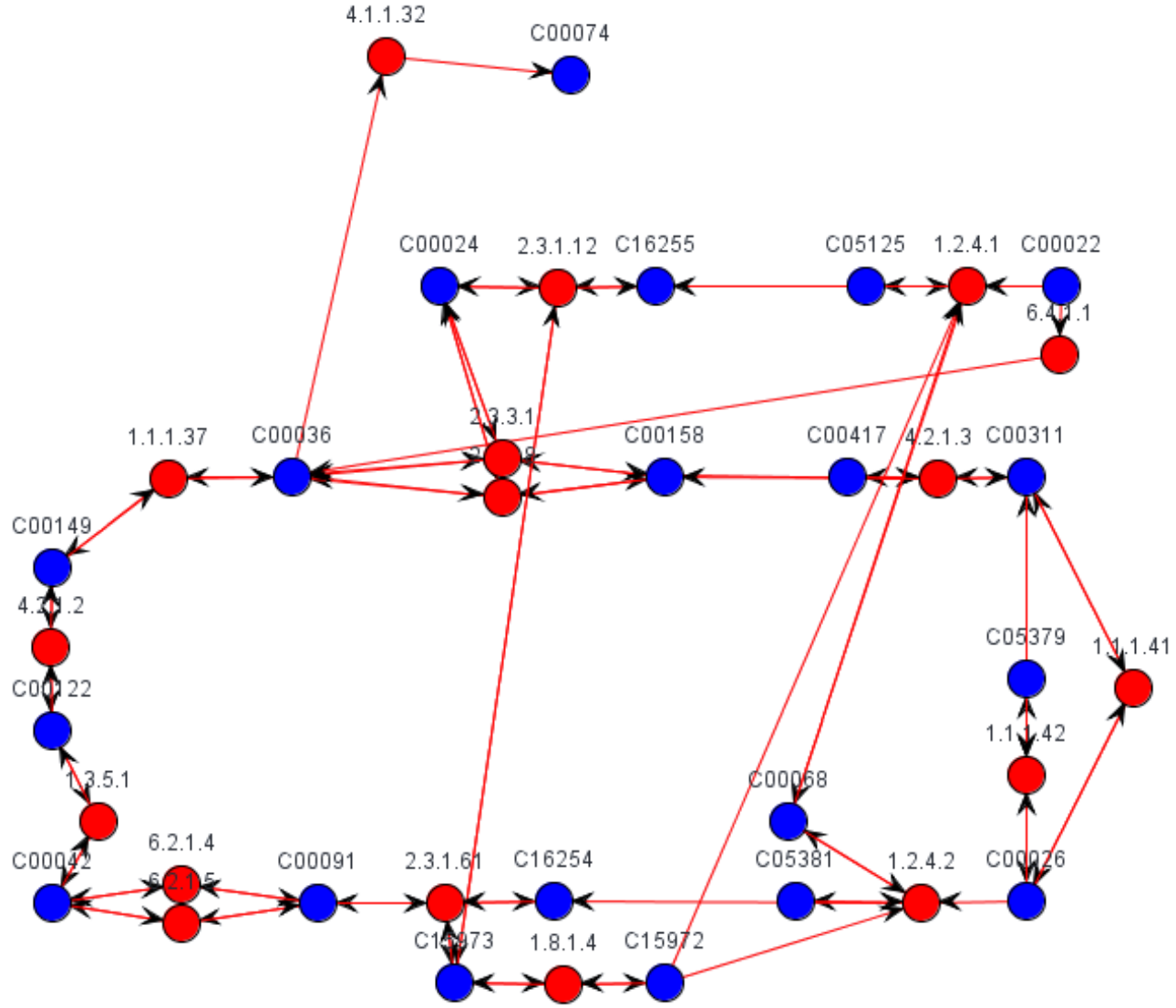


Figure 138: 2D Visualization of the Citrate Cycle (hsa00020) in CmPI

A first look at all selected localizations in the Localization Chart (Section 5.3.2.3.5) of Figure 140 and 141 show that the initially assigned localizations do not match the expectations from literature. But this is an unsurprising result, because CmPI just selects the first entry in the alphabetically ordered list of potential localizations for each protein. In addition, the KEGG pathways use the EC numbers to identify the proteins which is only a vague classification, as discussed in Section 3.3.2.2.2. Therefore six different localizations were set in the beginning. But it will be shown that only a few clicks are needed to assign the correct localizations.

Figure 139: Color codes for all cell components coded in the following figures

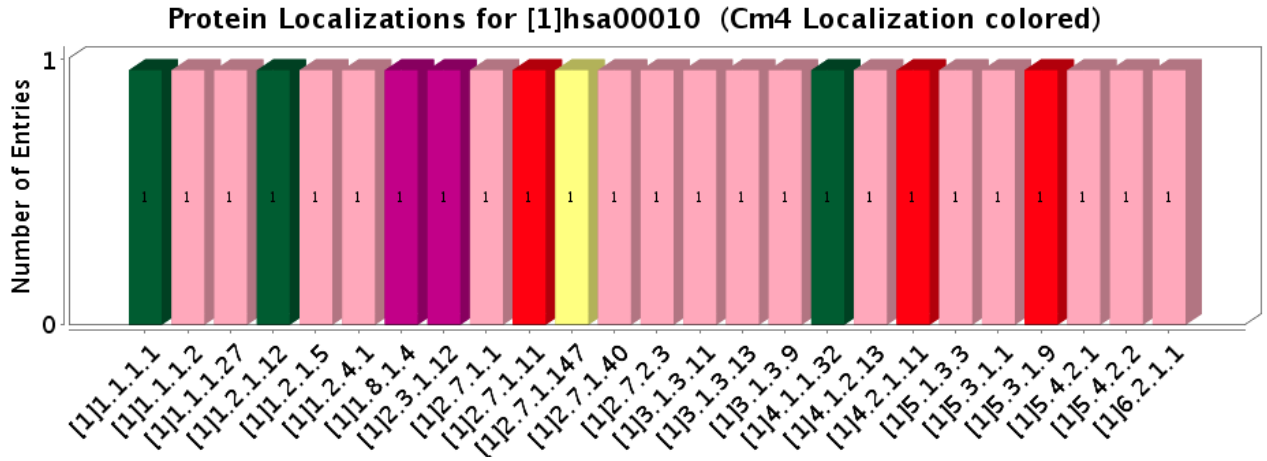


Figure 140: Initial Localization Chart, category "Protein Localizations/Cm4" for hsa00010 and hsa00020

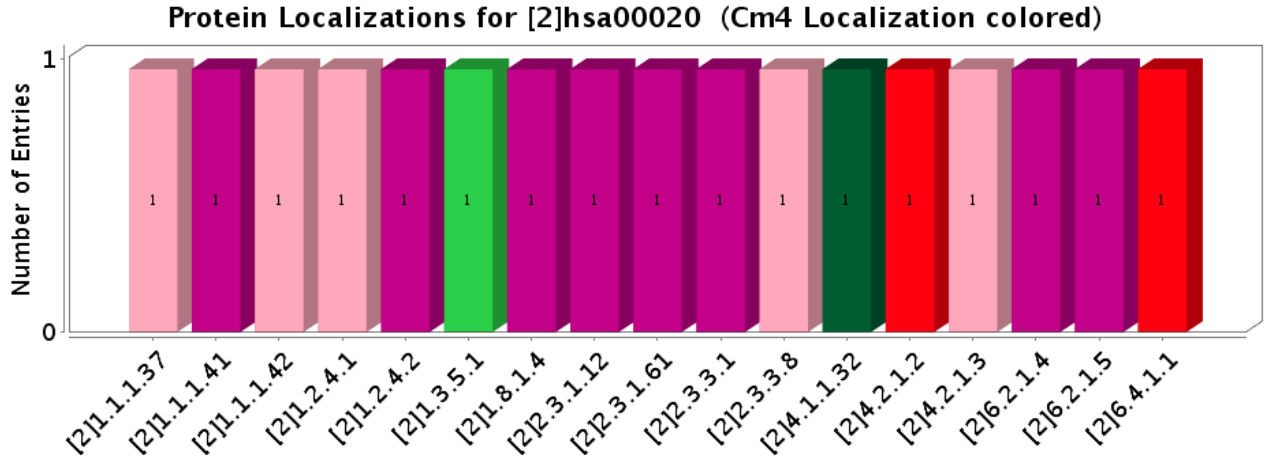


Figure 141: Initial Localization Chart, category "Protein Localizations/Cm4" for hsa00020

It should be mentioned that the applied color coding is the "Contrast Color Codes" introduced in Section 5.1.2.3.1, because the first priority during this analysis is the fast discriminability of different cell component types.

6.3.1.3 Investigating the preliminary Localizations

Now all potential localizations should be investigated. Figure 142 shows the Localization Charts for the glycolysis and Figure 143 for the citrate cycle respectively. And obviously, the initial expectations for both pathways seem to be confirmed by the images: the enzymes of hsa00010 are mainly localized at the cytosol, whereas those of the citrate cycle are concentrated at the mitochondrion. It is interestingly to note that the localization with the second highest incidence is – in both cases – the localization with the highest priority of the pathway vis-a-vis.

But still these figures contain no information if in fact all proteins can be localized at the corresponding cell components. There are different ways to find a solution. Here, a brute force method is chosen.[86]

86 The other method which uses the Localization Charts category "Protein Localizations / Cm4" is described in Section 6.3.2.2.

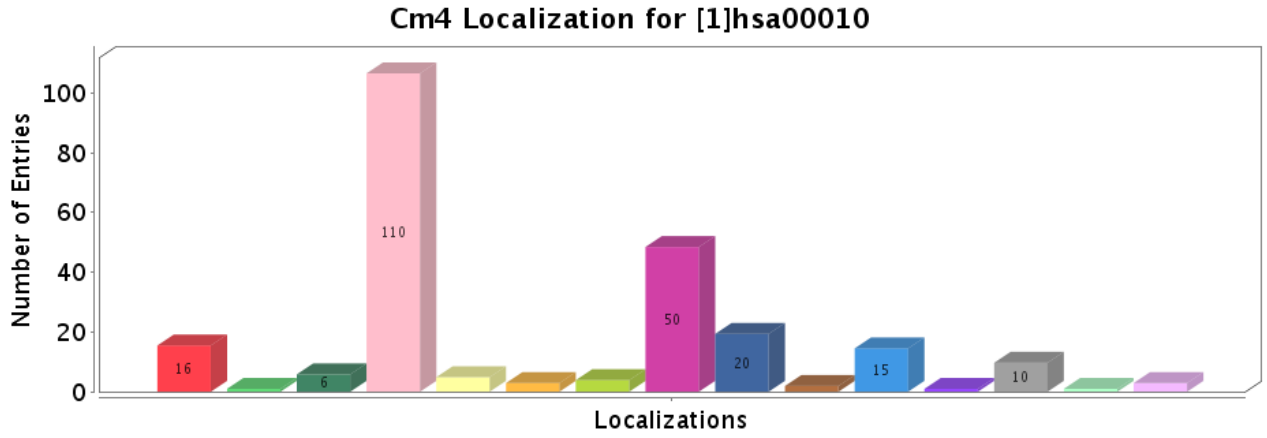


Figure 142: Initial Localization Chart, category "Localizations/Cm4" for hsa00010

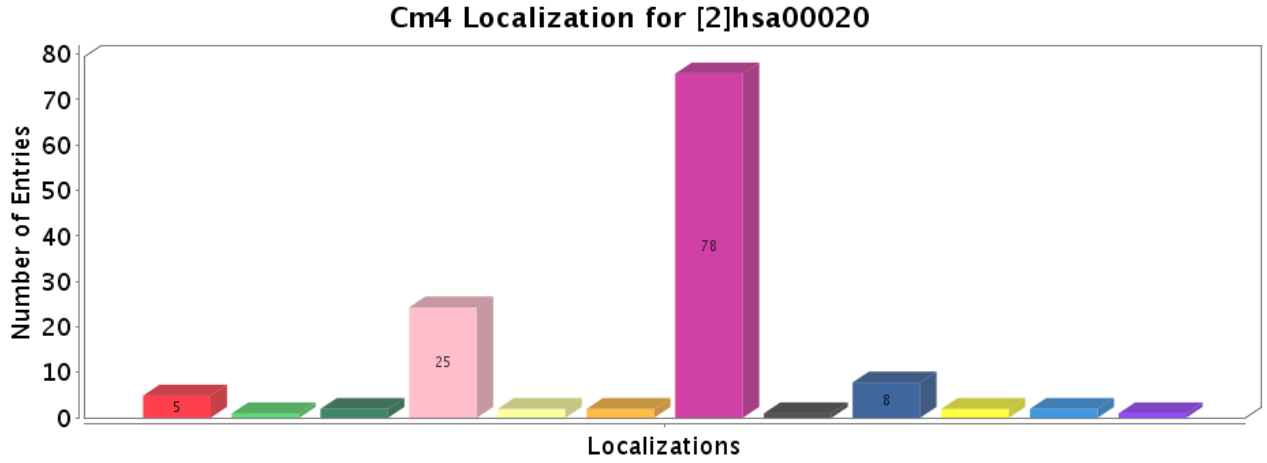


Figure 143: Initial Localization Chart, category "Localizations/Cm4" for hsa00020

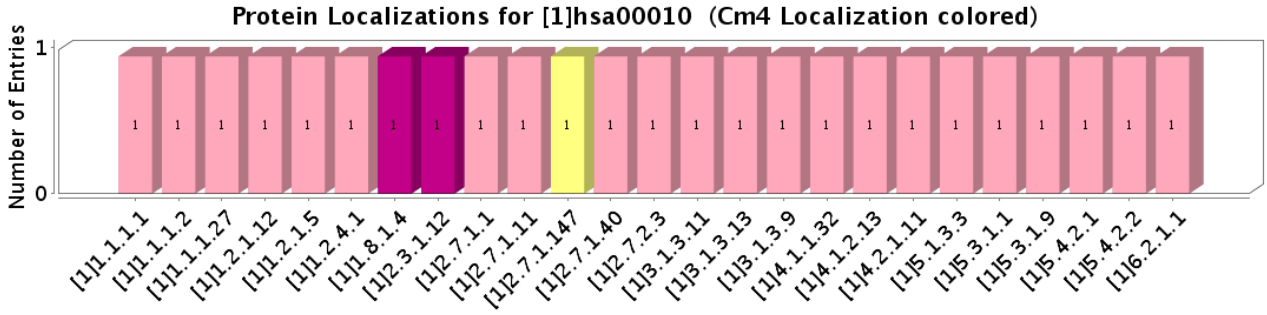


Figure 144: Localization Chart, category "Protein Localizations/Cm4" after assigning the localization "cytosol" to hsa00010

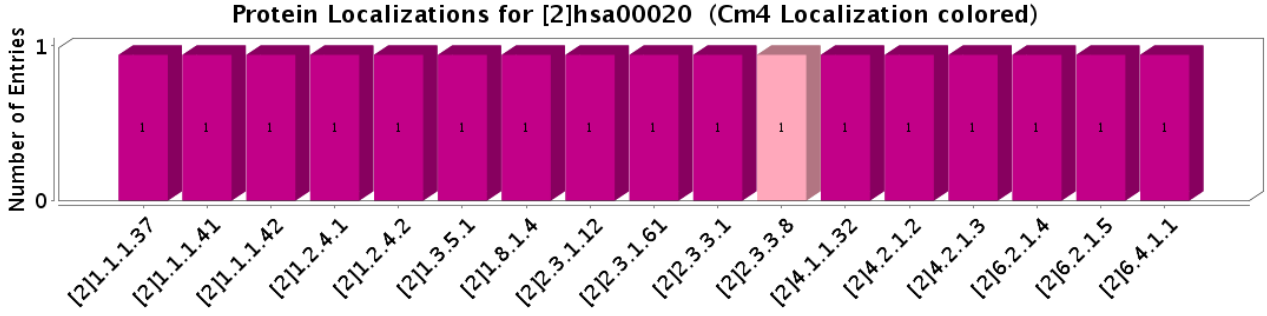


Figure 145: Localization Chart, category "Protein Localizations/Cm4" after assigning the localization "mitochondria" to hsa00020

By double-clicking on the bars representing the cytosol for hsa00010 and the mitochondrion in hsa00020, the according localization is applied to all enzymes featuring an entry for these potential localizations. Figures 144 and 145 shows the result. Obviously, the first two double-clicks were sufficient to find the correct localizations for both pathways.

For the glycolysis, 22 of 25 enzymes were localized at the cytosol and for the citrate cycle, 16 of 17 enzymes were directly assigned to the mitochondrion. Therefore, the assumption from the start of this section was verified. But it is possible to interpret even more. There are two interacting pathways, but normally, also by checking the textbook (Table 4), it is not known which proteins are involved in the transition between cytosol and the mitochondrion. By looking again at Figure 144 the proteins are directly identified. The enzymes 1.8.1.4 and 2.3.1.12 are part of both pathways and there only localization in the context of these two pathways is the mitochondrion. In addition, the enzyme 2.3.3.8 which in this example is only part of the citrate cycle, seems to be an enzyme not localized in the mitochondrion.

Now, a more accurate examination of the different localizations is needed. The Localization Terms used to map 2.3.3.8 to the cytosol is the “citrate lyase complex”. By looking at Figure 138, it can be seen that this enzyme is directly involved in the generation of the citrate (C00158). And this is processed in the citrate lyase complex which converts citrate to oxaloacetate (Table 4).

Next the accuracy of the Membrane Localization of the citrate cycle should be verified. Figure 146 shows an excerpt. Obviously, most of the enzymes were directly localized to the mitochondrial matrix, which is correct. For 1.2.4.2 and 1.3.5.1, the mitochondrial inner membrane was first selected. For the enzyme mentioned last, another option is only the outer membrane of the mitochondrion. Therefore, the inner membrane seems to be the correct option. But for 1.2.4.2, there is also a mapping to the Matrix available, which can be simply chosen by changing the entry in the Localization Table.

2	1.8.1.4	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (1/5)
2	1.2.4.2	mitochondrial inner me...	mitochondrial inner membrane	2/5 Inner Mem...	BRENDA: Reviewed (1/7)
2	2.3.1.61	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (1/5)
2	6.2.1.5	mitochondrial chromos...	mitochondrial chromosome	1/5 Matrix	ANDCell: PubMed (1/5)
2	6.2.1.4	mitochondrial chromos...	mitochondrial chromosome	1/5 Matrix	ANDCell: PubMed (1/4)
2	1.1.1.42	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (4/9)
2	1.1.1.41	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (1/8)
2	2.3.3.8	citrate lyase complex | H...	citrate lyase complex	1/1 Cytosol	GO: TAS:ProtInc (1/7)
2	1.3.5.1	mitochondrial respirator...	mitochondrial respiratory cha...	2/5 Inner Mem...	GO: ISS:UniProtKB (3/9)
2	4.2.1.2	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (4/5)
2	4.2.1.3	mitochondrial chromos...	mitochondrial chromosome	1/5 Matrix	ANDCell: PubMed (10/20)
2	2.3.3.1	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	GO: IDA:UniProtKB (1/3)
2	1.1.1.37	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (4/7)
2	6.4.1.1	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (3/6)
2	1.2.4.1	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	GO: IEA:UniProtKB-SubCell (2/12)
2	2.3.1.12	mitochondrial pyruvate ...	mitochondrial pyruvate dehy...	1/5 Matrix	GO: NAS:UniProtKB (1/7)
2	4.1.1.32	mitochondrial matrix | ...	mitochondrial matrix	1/5 Matrix	REACTOME: by G.O. (8/10)

Figure 146: An excerpt of the Localization Table showing only hsa00020

6.3.1.4 Examining an Outsider by direct Access to external Sources

For the glycolysis, there are no relevant Membrane Localizations which have to be examined. But there is one question left. The outsider enzyme 2.7.1.147 was localized at the extracellular matrix by the Localization Term “extracellular region”. The localization is provided by GO: “Inferred from electronic Annotation: UniProt-KB: SubCell”. Clicking on the link provided in the Localization Table, the user is directed to the “http://www.ebi.ac.uk/QuickGO/“ site, where links to the UniProt entry Q9BRR6 is shown, based on two references (Figure 147). A closer look at the UniProt entry reveals that the shown protein is involved in the glycolysis. Concluding these observations there are two options:

1. The enzyme 2.7.1.147 is located in the extracellular matrix during the involvement in the glycolysis.
2. There is currently no experimental proof found in the databases that this enzyme is located in the Cytoplasm – and most probably also no verification that it is in the neighborhood of the mitochondrion – but it can be predicted that this protein will be most probably found exactly there.

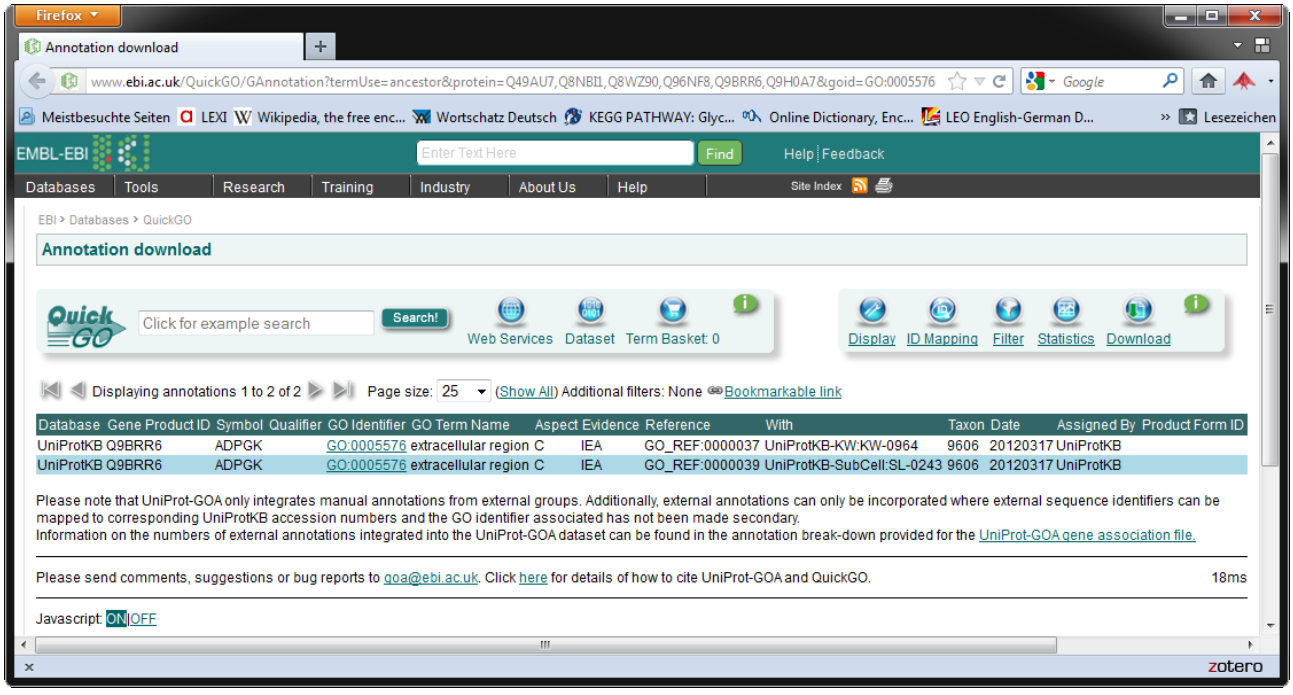


Figure 147: The Link to 2.7.1.147 from the Localization Table in CmPI
It shows additional localization information

6.3.1.5 Localization Result

Finally, it can be stated that the preliminary Localization Charts shown in Figure 144 are visually equal to the final result. Only a few changes according to the Membrane Localizations of two proteins had to be done, the other Localizations were already correct. The resulting priority list would looks like this:

Priority List based on Cm4 Membrane Localizations

1. Mitochondrial matrix
2. Mitochondrial inner membrane
3. Cytosol
4. Extracellular space

Therefore it was shown that nearly no foreknowledge would have been needed to predict the localization of these two pathways in this case.

6.3.1.6 3D Visualization

Of course, the intention of this localization process is in this case the correlation of the network with the Virtual Cell environment. Figure 148 shows a SphereCell containing only cell components relevant for the final localizations (from inside to outside): mitochondrion, cytosol, cell membrane and extracellular matrix. The applied contrast color-coding was described in Figure 89. Figure 148 shows now the different layouts discussed in Section 5.3.2.5 applied to the SphereCell.

Figure 148.1a shows the UUUSphere layout using the Rakhmanov, Saff and Zhou (RS&Z) approach and Figure 148.2a the Geodesic approach. On the right side, Figure 152.1/2b use also the RS&Z (1) and Geodesic (2) approaches, but in combination with the ISOM layout.

As previously discussed, the UUUSphere layouts solve the OPT_NMP_MAX_AREA problem. For this purpose the Geodesic layout is able to continuously extend the number of available positions by increasing the resolution of the sphere (Section 5.3.2.5.1). The disadvantage of this approach is that this layout is only completely balanced if all available positions generated by the Geodesic layout are occupied. In the case of a 1-icosahedron, the minimal number of available positions is 12. If there are only ten nodes to place, two positions will be free. The Geodesic layout places the nodes along triangular segments of the i-Icosahedron (as shown in Figure 110). Therefore, this layout features an accumulation of nodes on close positions. Moreover, because the distribution algorithm travels along the vertices of different sphere segments, the series of node placing is not as straight-forward as the RS&Z approach. The latter has the advantage of a more balanced distribution. The reason is the iterative positioning following the virtual spiral.

Obviously when comparing the two layouts, the advantage of the ISOM layouts is apparent. It has the additional feature of optimizing the positioning of the nodes with respect to the connecting edges by solving OPT_NMP_ENERGY. In opposition to this advantage, the drawback is that the uniform distribution of nodes may change to a more unfavorable state. However, the ISOM layout shows a more compact arrangement of nodes, because it tries to move interconnected nodes closer to each other.

In Figure 149 the same layouts are used with an animal cell similar to the one discussed in Section 6.1.1.2. The mitochondrion is based on the model discussed in Section 5.1.2.1.1. Because only the mitochondrion model is shown here, mainly the citrate cycle is visible. The whole context – the complete cell – can be seen in Figure 150. The difference is that all different sphere layouts have to be mapped onto the cell component´s shapes after finishing the distribution process (Section 5.3.2.4). Mostly all enzymes are located in the matrix region. But examining the position of the 1.3.5.1 in Figure 149.2b, which is localized at the inner membrane (Figure 146), it is visible that this enzyme is located at the appropriate position on the according shape.

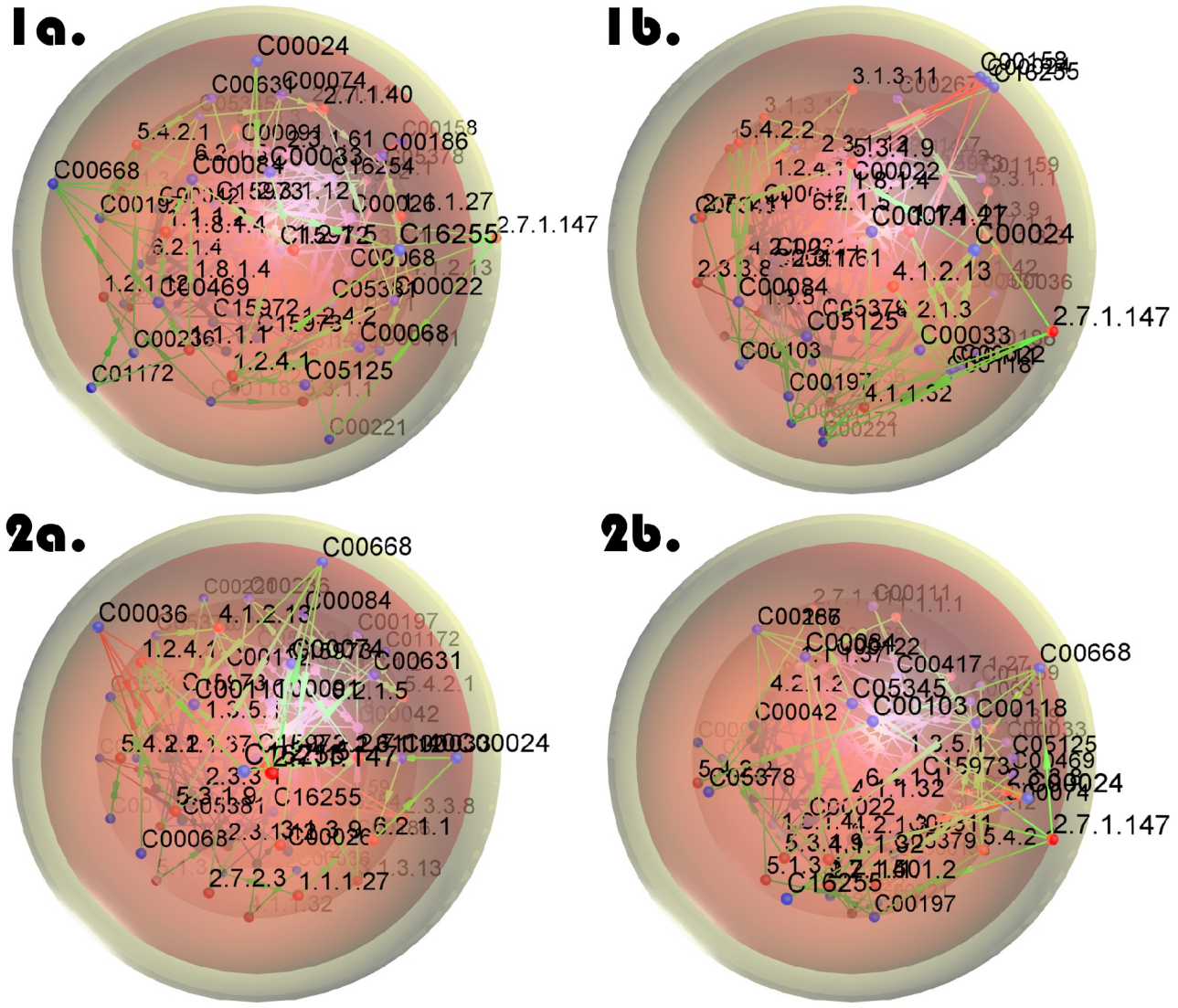


Figure 148: The citrate cycle and glycolysis mapped onto the SphereCell
1. Rakhmanov, Saff and Zhou; 2. Geodesic; a. UUUSphere Layout; b. ISOM Layout

However, the observations are similar when comparing the different layouts to the one looking at the previously discussed image. If the shape of the cell component differs extremely from the one of the surrounding distribution sphere, the mapping might lead to problematic results. For example, nodes which were initially equally distributed on the sphere surface may be mapped onto a single position at the cell component surface. Figure 150 also reveals another aspect which was mentioned before, the mapping of nodes onto the invisible structures. The cytosol and the extracellular matrix are not visible, but there are different nodes inside the cell which seem to be placed in an empty area. This is the cytosol. And the same applies to nodes outside the cell (on the top), they are mapped onto the extracellular matrix.

Finally, it has to be mentioned that the two-dimensional projections shown here are not able to compete with an interactive three-dimensional visualization where the user is able to use the different navigation methods discussed in Section 5.3.2.10.

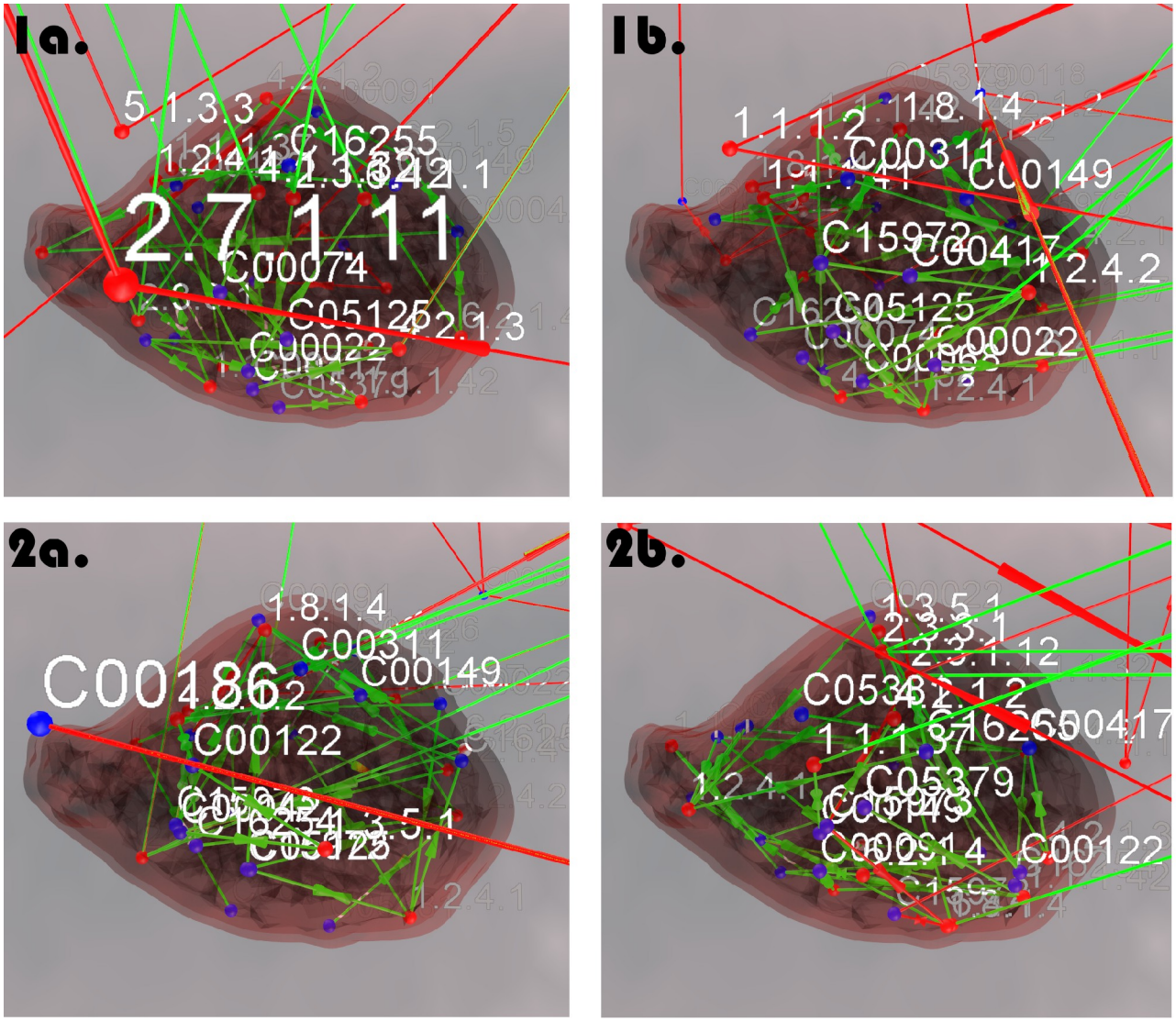


Figure 149: The citrate cycle and glycolysis mapped onto a mitochondrion model based on microscopic data
1. Rakhmanov, Saff and Zhou; 2. Geodesic; a. UUUSphere Layout; b. ISOM Layout

6.3.2 Cytological Disease Mapping

An important aspect of the subcellular localization of protein-related networks is the cytological disease mapping. Using the information generated by CmPI, disease-related protein-sets and pathways can be mapped onto the cell model. This can be especially interesting if the concrete location of a disease is unknown.

6.3.2.1 The Protein-Protein-Interaction Network Reconstruction

The first approach using CmPI to localize a disease-related pathway was published in 2010 [STKH10]. In the latter work, data integration as well as text mining techniques were discussed by comparing DAWIS-M.D. and ANDCell respectively. It was shown, that the database-related data from DAWIS-M.D. was more reliable, because databases like BRENDA and UniProt contain a huge amount of curated data. But in addition it emerged that text mining approaches like ANDCell are sometimes able to provide additional information which is not included in established databases. Because ANDCell results are not curated, it

is always needed to verify the data provided by text mining. Therefore, the option to check the underlying information by providing the link to PubMed in conjunction with the text fragment (the basis for the localization-association) is an indispensable feature of ANDCell (Section 5.3.2.3.4).[87]

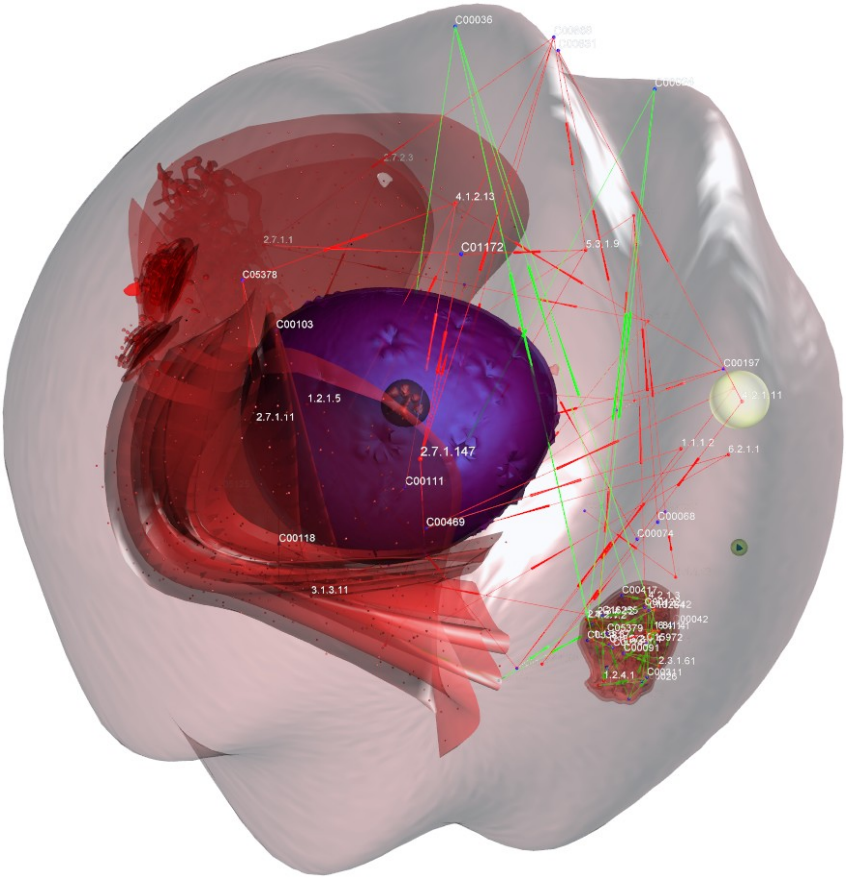


Figure 150: Whole Cell using the ISOM Layout with Geodesic Distribution Option

The following passages will deal with the data also used in the previously mentioned study. The starting point for the analysis was a distinct, experimentally identified protein: MUPP1 or MPDZ – the Multiple PDZ Domain Protein. It consists of 13 domains, featuring a large diversity of interaction proteins as well as viruses [CVPB04, LRFW05, USFL98]. This protein was found to be strongly involved in a special cardiovascular disease (CVD): the dilated cardiomyopathy (DCM). CVD is known to be one of the leading causes of death – especially in industrialized countries.

Data from a female DCM patient (age 52) which was collected during the CardioWorkBench project [CaAz07, KHTH10] was investigated by a proteomic analysis using a Clontech Am Microarray™ 500 (Lot no. 7030444, Clontech, CA, USA). MPDZ was identified as one of seven proteins assigned to the primary expression class, featuring a very high activity.

In summation, MPDZ was chosen as the appropriate protein for the former analysis of the localization quality of CmPI for following three reasons:

1. It is known that MPDZ is a relevant DCM-associated protein.
2. It is known that MPDZ shows a large diversity to proteins.
3. It is known that MDPZ is part of the Tight Junction Pathway, as can be seen in the KEGG pathway hsa04530 as well in different publications [CVPB04, LRFW05].

87 This Section is based on the aforementioned publication “Visualization and Analysis of a Cardio Vascular Disease- and MUPP1-related Biological Network combining Text Mining and Data Warehouse Approaches” and the forthcoming publication: “Subcellular Localization Charts: a new visual Methodology for the semi-automatic Localization of protein-related Data Sets” [SKDA13].

Therefore, there was quite a high probability that there will be some results which should be applicable to data integration as well as text mining approaches. And indeed, VANESA (Section 4.3.1.3) generated a network containing 12 additional proteins and ANDVisio (Section 3.3.3.2) seven additional proteins after curation (Section 5.3.2.3.4). Moreover, two of those proteins from VANESA and ANDVisio were identical. Therefore, the merged network reconstructed with CmPI contains 18 distinct proteins including MPDZ. The result can be seen in Figure 151. The ten protein nodes in the center were exclusively found by VANESA whereas the five nodes located at the left and right outer border were exclusively found by ANDVisio. The two nodes in the right center, CLDN5 and CSPG4, as well as the major protein MPDZ, were originally found by both tools.

Based on the information stated before, the localization of MPDZ could be predicted to be – also in the context of DCM – the tight junction of the cell membrane. Now it should be examined if the localizations of the other proteins also match.

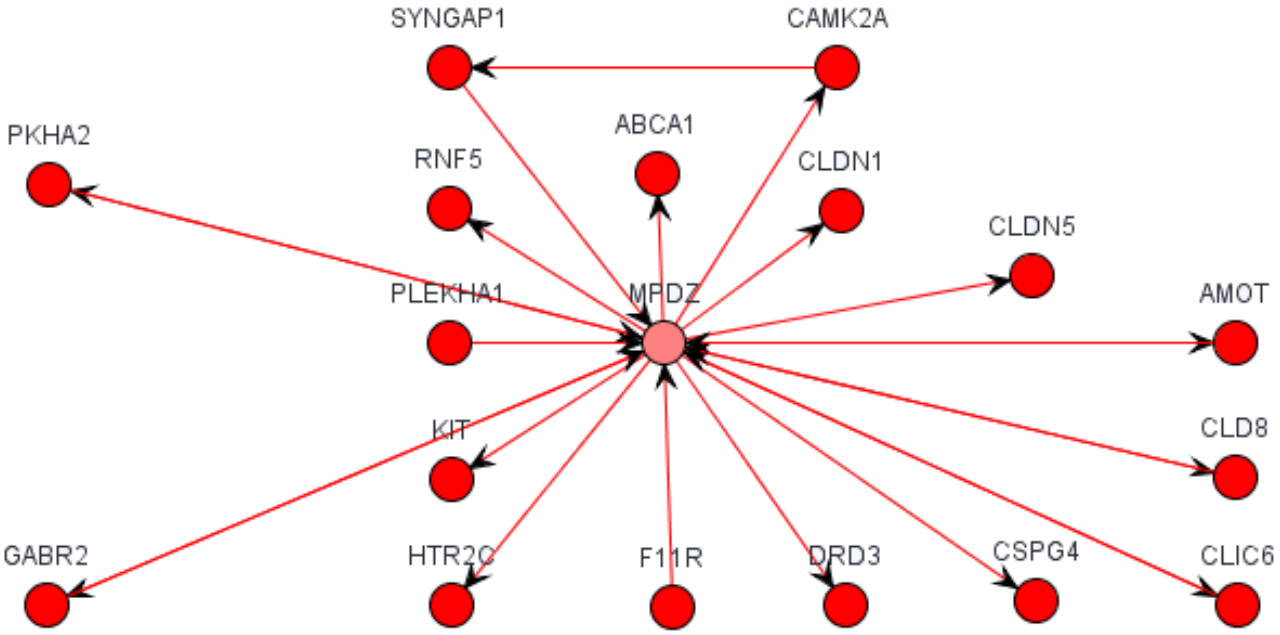


Figure 151: The MPDZ-protein-protein-interaction network based on results from VANESA and ANDVisio

6.3.2.2 First Localization Results

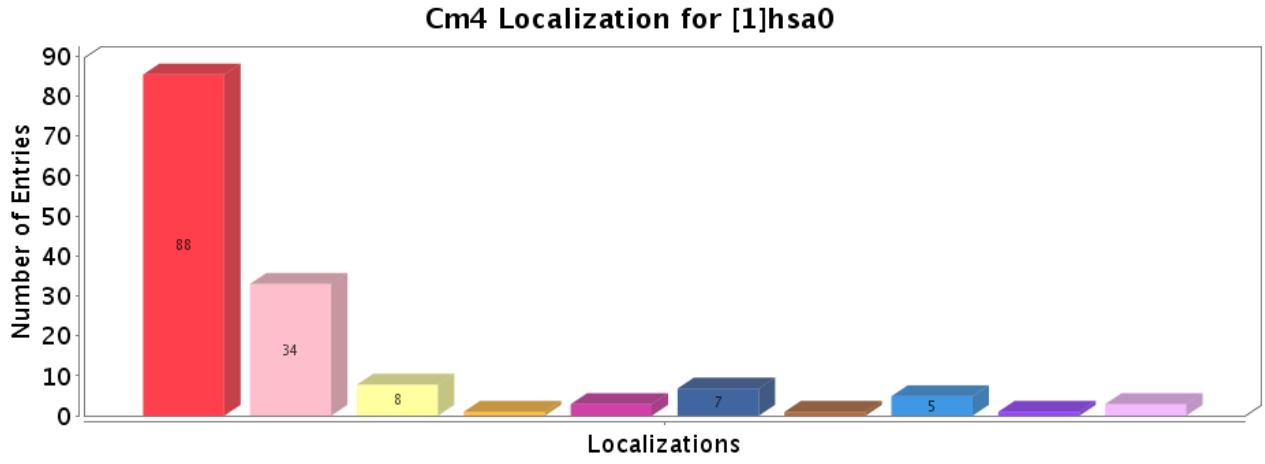


Figure 152: Cm4 Localization Overview for the complete pathway

CmPI was queried (see also Figure 104) by using the species-specific term “homo sapiens” (Section 5.3.2.3.4). The following paragraphs discuss the visual analysis protocol using the Localization Charts which were introduced in Section 5.3.2.3.5. The applied color codes are exactly those shown in Section 5.1.2.3.1: the “Contrast Color Codes”.

A first glance at all potential Cm4 Localizations listed in the category “Localizations/Cm4“ (Section 5.3.2.3.5) in Figure 152 shows that the major localization of this protein-associated network is the cell membrane. Of course, this aspect does not guarantee that all proteins can be mapped to the cell membrane. Moreover, the two other most frequent localizations, the cytosol and the extracellular matrix, are adjacent to the cell membrane. Figure 153 shows the original view of the pathway data in CmPI.

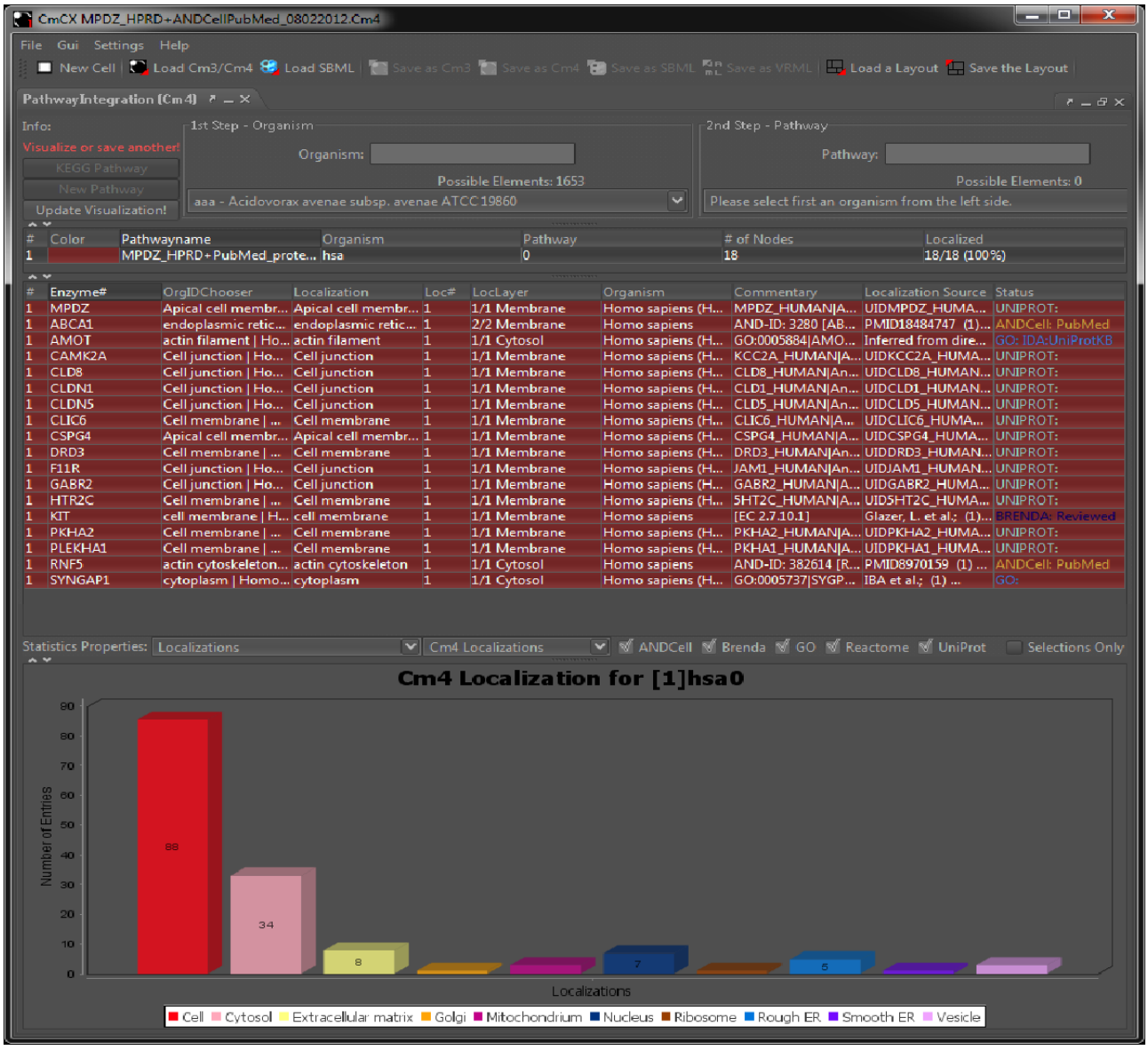


Figure 153: The CELLmicrocosmos PathwayIntegration showing the Pathway Table (top), the Localization Table (center) and the Localization Charts (bottom). The red bar on the bottom left – representing the Cm4 Localization “cell membrane” was clicked and all associated proteins were marked in the Localization Table

But already a fast glance at the category “Proteins: Cm4” in Figure 154 proves that all proteins have the cell membrane as a potential localization. PLEKHA1, RNF5 and SYNGAP1

each show only one localization result pointing to the cell membrane. But it should be mentioned that many proteins feature localizations not adjacent to the cell membrane: ABCA1, AMOT1, CAMK2A, CLD8, CSPG4, HTR2C, KIT, MPDZ, PKHA2, PLEKHA1, and RNF5.

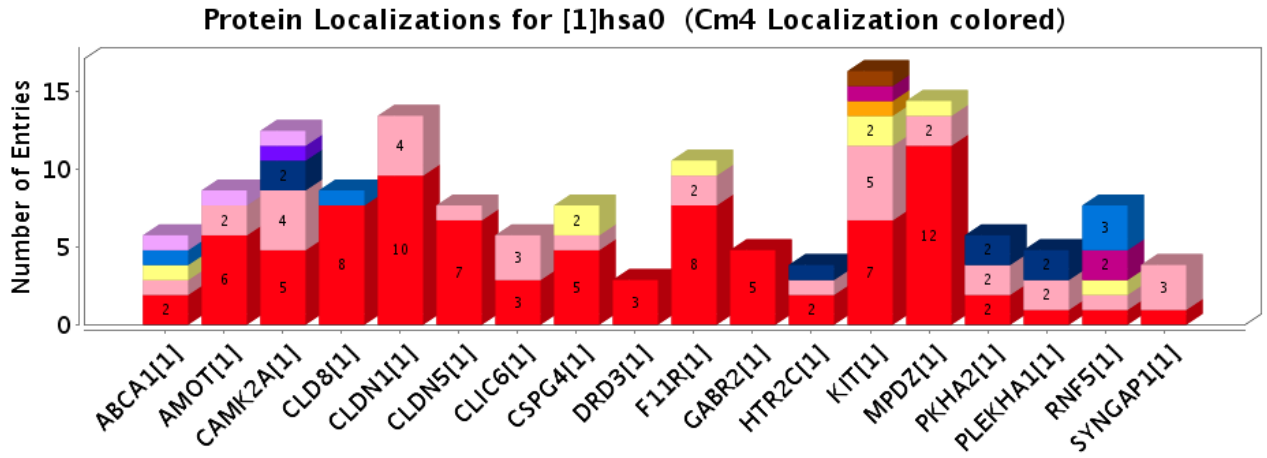


Figure 154: Protein Localizations overview for all proteins of the pathway

This is now an application case where the category “Protein Co-Localizations/Cm4” is quite interesting. Figure 155 shows the result of the nearest-neighbor-search-associated localization and obviously, the cell membrane outperforms all other cell components. Interestingly, the two localizations which were previously identified to be quantitatively relevant are also found in this figure. Most proteins are also found at the cytosol and five proteins may also be found in the extracellular matrix. However, two proteins are only found in the cell membrane.

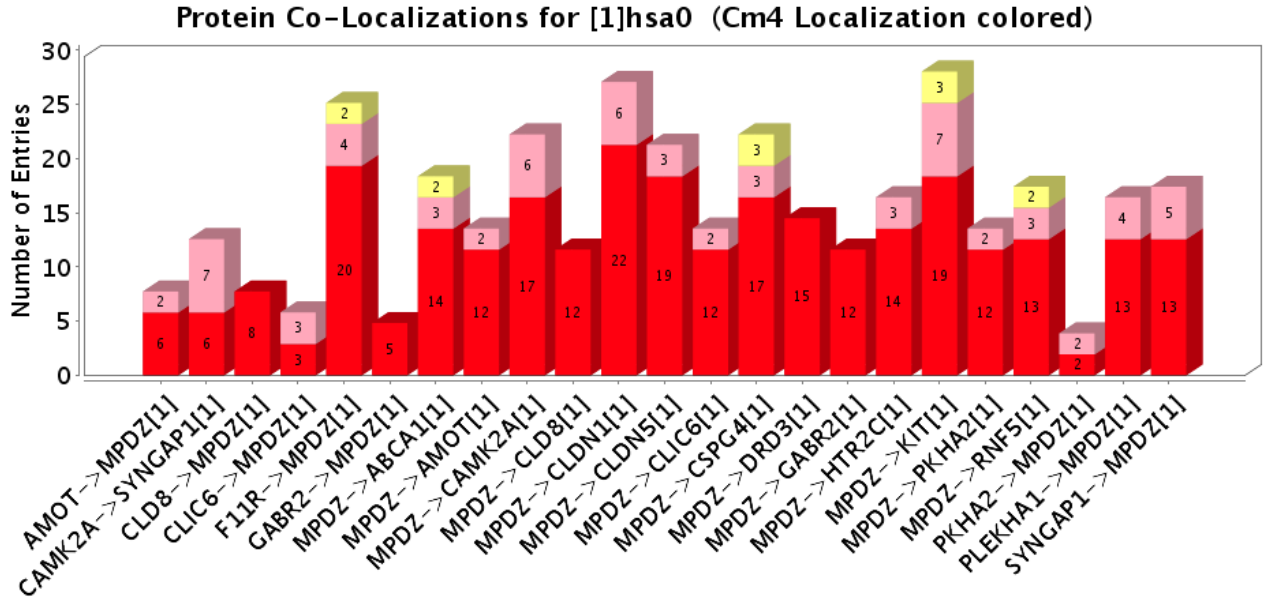


Figure 155: Protein Co-Localizations overview for all proteins of the pathway

An overview of the initially chosen localizations is shown in Figure 156. The localizations are based on the first entry in the alphabetically-ordered list in the Localization Table shown in Figure 151. They are automatically assigned after the database-parsing process was finished.

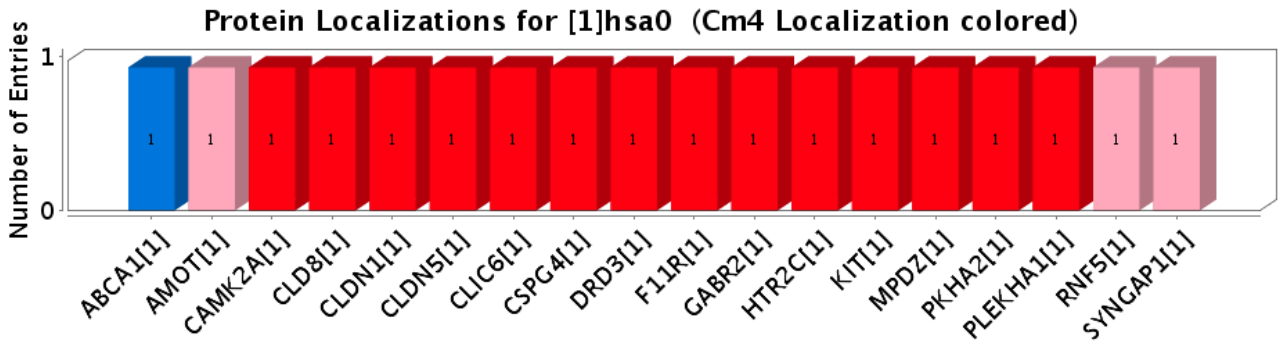


Figure 156: First chosen protein localizations of the pathway

6.3.2.3 Localization Assignment of the Pathway

Now it will be shown that only a few short steps are needed to assign appropriate localizations based on our initial observations and assumptions.

Because the cell membrane is the localization available for all proteins, it is assigned as actual localization for all proteins. For this purpose, first the whole pathway is chosen in the Pathway Table. Now all proteins are selected belonging to this pathway. In the Localization Charts the view “Localizations/Cm4” has to be chosen. Then, the chart shown in Figure 152 appears. A double-click on the bar representing the cell membrane assigns this cell component as actual localization to all proteins.

But of course category Cm4 is quite vague if different cell membrane associations should be compared. Therefore the category “Localizations/Localization Terms” is chosen for the proceeding analysis. Looking at Figure 159, by contrast, the advantage of the Cm4 mapping is obvious: many localization terms are just synonyms for the same cell component. And in terms of a Virtual Cell environment which does not explicitly integrate all available cellular entities it is indispensable to make this abstraction.

To now choose the correct distinct localizations, the favored localization should be mentioned again: the tight junction. A priority list of Localization Terms follows where the first entry has the highest priority and the number in brackets indicate the number of hits and the corresponding database:

Priority List A

1. enriched in tight junctions of epithelial cells (1, by UniProt),
2. localized at tight junctions of both epithelial and endothelial cells (1, by UniProt),
3. localized to tight junctions in all 3 segments of epididymis (1, by UniProt),
4. tight junction (by similarity) (1, by UniProt),
5. cell junction organization (4, by Reactome),
6. cell junction (10, by UniProt),
7. tight junction (20, by ANDCell), and
8. cell membrane (Cm4 Localization).

This priority list is processed by re-selecting the pathway in the Pathway Table and then assigning (again via double-click onto the according bar) the localization priorities in

reverse order, starting with priority 8 for the cell membrane. Figure 157 shows an excerpt from the resulting Localization Table after sorting alphabetically them according to the Localization Terms.

#	Enzyme#	...	Localization
1	CSPG4	...	Apical cell membrane
1	AMOT	...	Cell junction
1	CAMK2A	...	Cell junction
1	GABR2	...	Cell junction
1	CLDN5	...	Cell junction organization
1	CLIC6	...	Cell membrane
1	DRD3	...	Cell membrane
1	HTR2C	...	Cell membrane
1	PKHA2	...	Cell membrane
1	PLEKHA1	...	Cell membrane
1	MPDZ	...	Enriched at the tight junctions of epithelial cells
1	F11R	...	Localized at tight junctions of both epithelial and endothelial cells
1	CLD8	...	Localizes to tight junctions in all 3 segments of the epididymis
1	RNF5	...	Predominantly located in the plasma membrane
1	KIT	...	cell membrane
1	ABCA1	i...	integral to plasma membrane
1	SYNGAP1	i...	intrinsic to internal side of plasma membrane
1	CLDN1	...	tight junction (By similarity)

Figure 157: All selected Localization Terms of the whole pathway

Eight of 18 proteins are explicitly assigned to the tight junction complex. The remaining proteins are localized by more vague terms. Nevertheless all terms point towards the cell membrane. Because the main focus of this work is the visualization of a Virtual Cell, a three-dimensional model of an animal cell (see also Section 6.1.1.2) associated with the network is shown in Figure 158. Of course, all proteins are placed on the outer membrane shape.

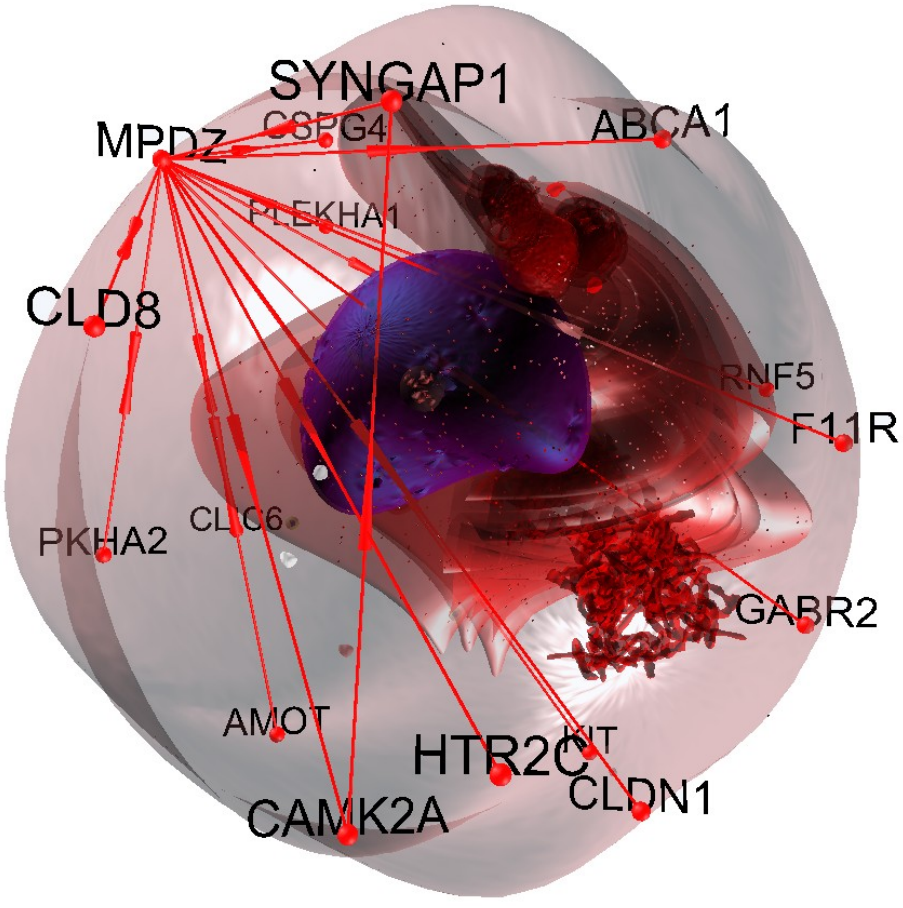


Figure 158: The animal cell shows cell components based on microscopic images associated with the network showing the disease-related protein sets

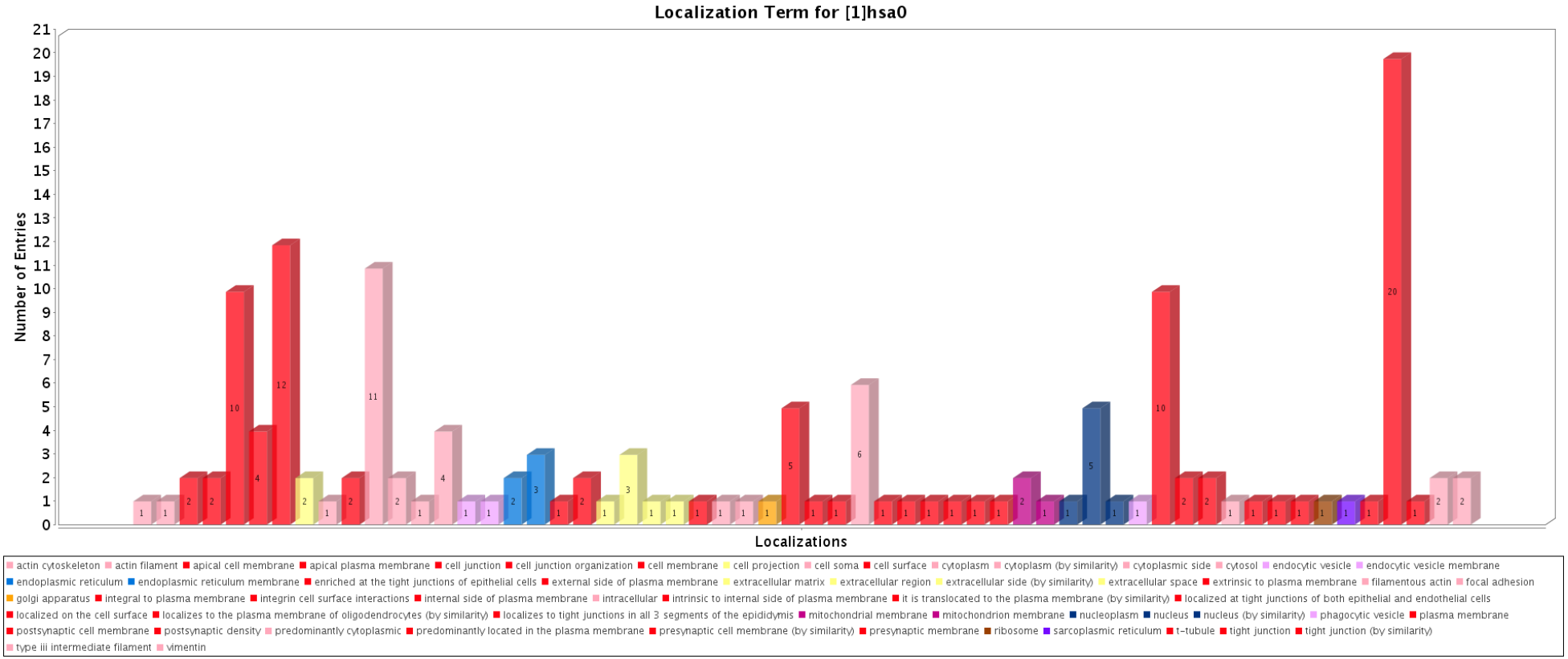


Figure 159: All Localization Terms of the whole pathway
In this case, it is not important to analyze the different terms associated with this image. Instead, this image illustrates the large variety of terms associated with distinct cell components

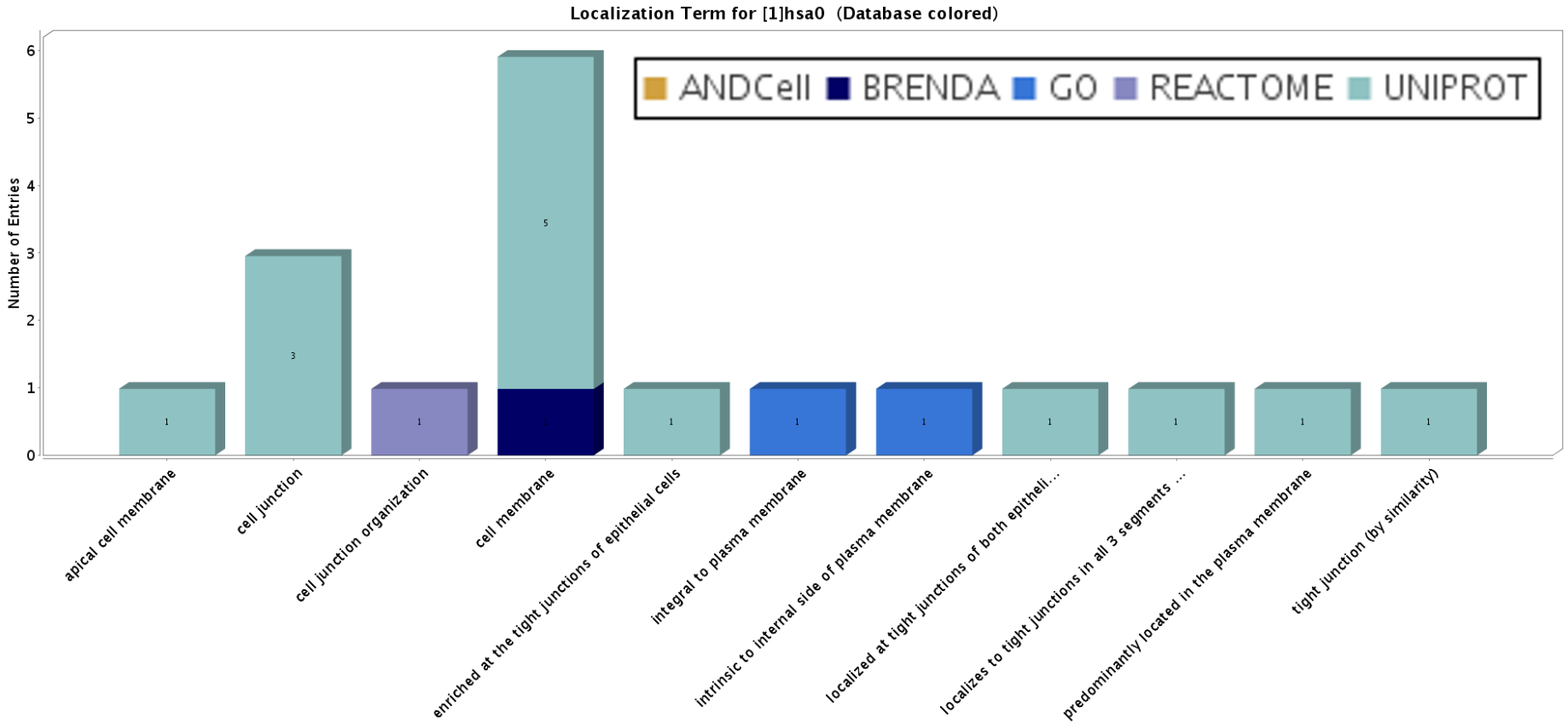


Figure 160: All actually selected Localization Terms
Representing the protein localizations in Figure 158, colored by their database sources

It was discussed before that the databases do not all feature the same reliability (Section 5.3.2.3.1). Therefore the actual localizations should be inspected regarding their accuracy. In Figure 161 the chart “Localizations: colored by database/Localization Terms” is shown. In Priority List A it can be seen that ANDCell provided most hits. But in addition, all localizations were confirmed by databases contained in DAWIS-M.D.: UniProt (14), GO (2), Reactome (1) and Brenda (1). Now the filtering options of the Localization Charts can be used to discover that ANDCell is supporting the localization to the tight junction complex for five proteins, UniProt and GO for eight proteins each, and Reactome for four proteins.

In addition, the “Co-Localization/Localization Terms” category can be chosen to evaluate the number of proteins co-localized to the tight junction complex: 11. Using this category, the 2D View should be used to visualize the pair-wise localizations.

For this purpose, first all proteins are assigned to the Unknown component. Then, Priority List B is applied to the whole pathway:

Priority List B

1. - 7. see Priority List B
8. Unknown

Figure 161 shows the result. This time (in contrast to Figure 151 where the proteins are colored according to their node type: proteins), the proteins are colored according to their localization (Section 5.3.2.9). Unknown is gray and cell membrane, as can be seen for example in the legend of Figure 152, red. Therefore it is possible to identify those proteins pair-wisely localized to the tight junction with just one glance.

The same method is now applied to the potential localizations cytosol and extracellular matrix.

Priority List C

1. cytosol
2. cell membrane

Priority List D

1. extracellular matrix
2. cell membrane

Figure 162 shows the result for the cytosol: only three proteins are not potentially found there. And Figure 163 shows the results for the extracellular matrix, where only six proteins are potentially located.

The three last figures provide a good overview of potential localizations for all proteins. These visualizations might also be utilized to make predictions regarding traveling processes between the cell membrane, the intracellular cytosol and the extracellular matrix.

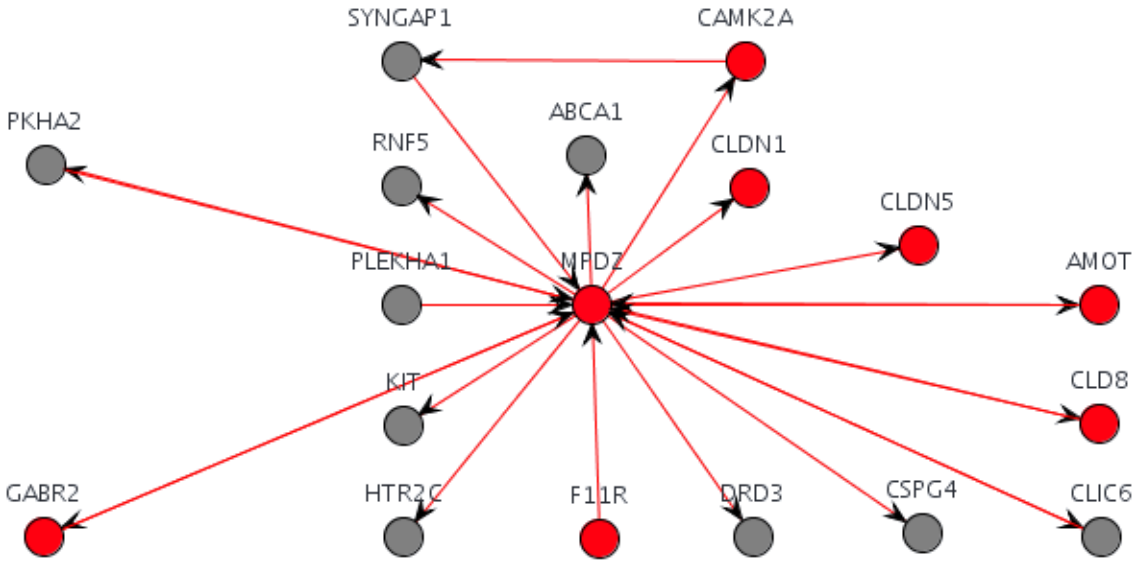


Figure 161: Priority List B applied to all proteins

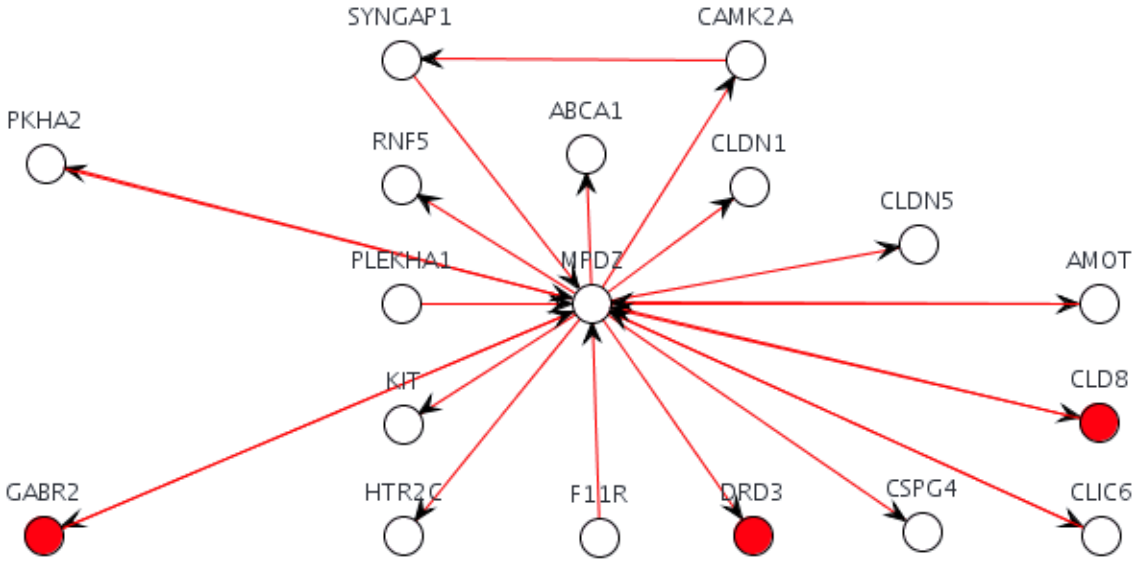


Figure 162: Priority List C applied to all proteins

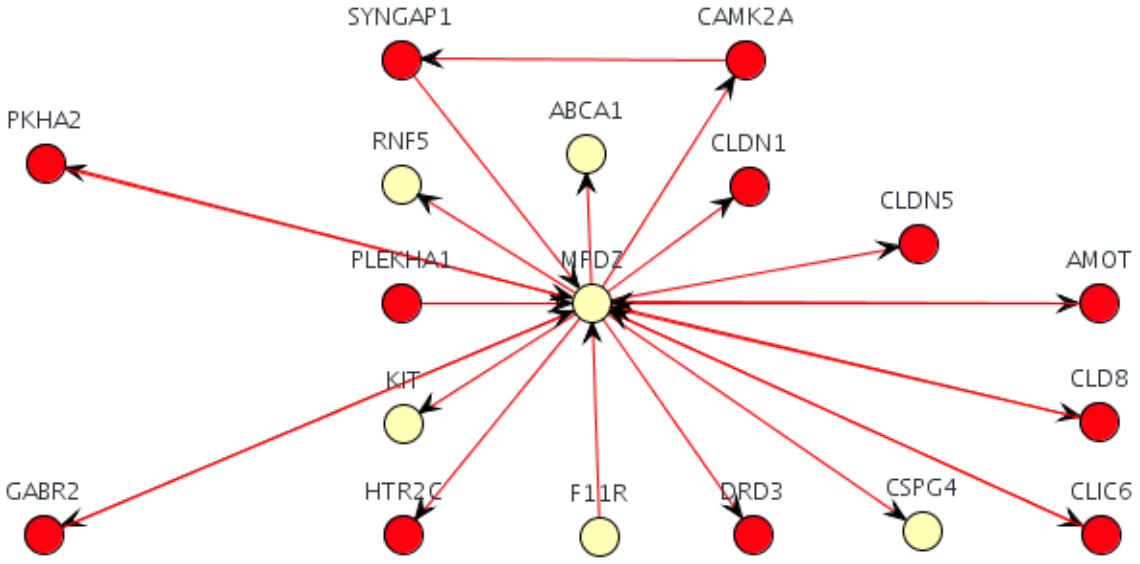


Figure 163: Priority List D applied to all proteins

6.3.2.4 Inter-species Comparison of three Pathways

In Section 6.1.1.1 a comparison cell was generated with the CellEditor. This type of cell model will be used to compare specific pathways among different species. The related approaches introduced in Section 4.3.2, for example WilmaScope, applied 2.5D visualization techniques for this purpose.

Figure 164 shows a comparison cell with three mitochondria, three chloroplasts and one (transparent) cytosol model, surrounded by the cell membrane. This cell is correlated with citrate cycles of three species acquired from KEGG: homo sapiens (hsa00020, green, left), mus musculus (mmu00020, red, center), and arabidopsis thaliana (ath00020, blue, right). The three pathways were localized using the methods discussed in the preceding sections. The image shows a parallel projection of the cell environment.

At first view it is obvious that ath00020 differs from the other two pathways, because this is found in a plant cell and therefore the citrate cycle is associated with the chloroplast. The fact that it was not possible to correlate ath00020 completely with the chloroplast shows that there was not sufficient localization information available for arabidopsis thaliana.

Finally, the two animal cell-associated pathways should be compared. Obviously, the structure of the pathways in both mitochondria slightly differ, but a number of proteins and compounds are located at the same position, for example 1.2.4.2, 6.2.1.5, and C00042. On the contrary, the positions of enzyme 2.3.3.8 differ. For hsa00020 it was localized at the cytosol, and for mmu00020, at the mitochondrion. Here, a problem of the layouts discussed in Chapter 5.3.2.5 is illustrated. In case the elements of the pathway exactly match, the structure of the pathways will be exactly the same. It would not be reasonable to use the ISOM layout for this purpose, because the force-directed layout already dramatically changes in case the localization of a single enzyme in two pathways differs. Therefore, the Geodesic layout of UUUSphere is used. Still, the 2.5D approaches outperform the comparison cell if the user wants to gain a fast overview. But if the structure of different cell models should be additionally taken into account and if it is possible to navigate in 3D space, this approach is an appropriate alternative.

Figure 165 illustrates another interesting aspect in correlation with a three-dimensional environment. This figure shows the same pathway located in the center of Figure 164 but without the mitochondrion. Obviously, the structure of the mitochondrion helps to hide the nodes in the background and to focus on the nodes closer to the user.

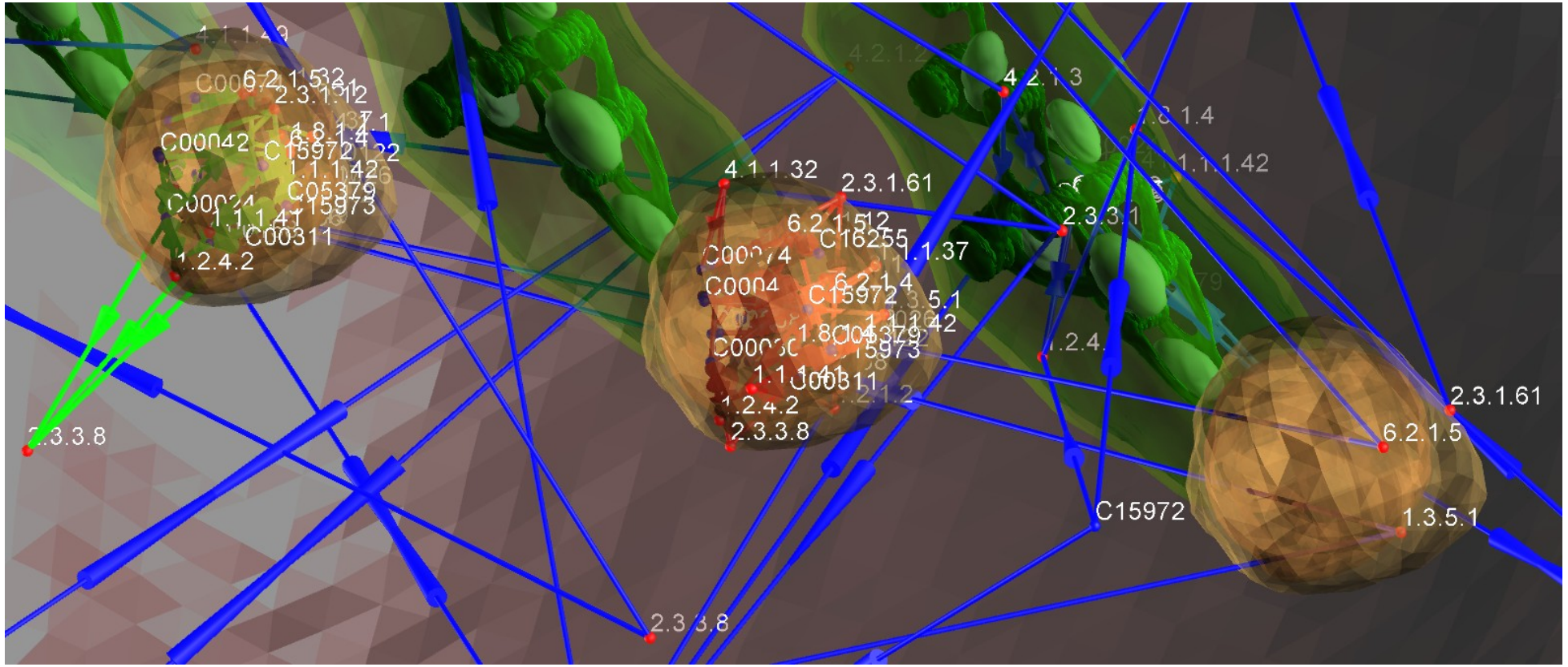


Figure 164: Comparison of citrate cycle among three species correlated with a comparison cell model Containing three mitochondria, three chloroplast and one cytosol model correlated with hsa00020 (homo sapiens, green), mmu00020(mus musculus, red), and ath00020 (arabidopsis thaliana, blue)

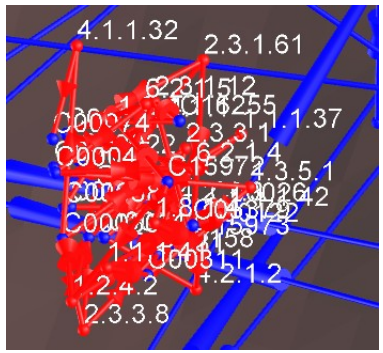


Figure 165: The pathway mmu00020 shown in the center of Figure 164 without the mitochondrion model

6.3.2.5 Proof of Concept

Another interesting aspect which should be taken into account is the fact that in 2010, during the first analysis, there was no satisfying localization found for SYNGAP1 [STKH10]. At that time, only the GO (linking in this case information from the InterPro database [HAAB09]) integration in DAWIS-M.D. provided the results that SYNGAP1 was found in the cytoplasm. This is of course indispensable for a concrete cellular localization, because cytoplasm naturally corresponds to nearly everyplace in the cell. But already at that time the CmPI-results of the SYNGAP1-associated proteins supported the theory that this protein most probably can also be localized to the cell membrane.

During the preparations of the second publication regarding this topic the localization problem of SYNGAP1 was reviewed.[87] Examining the figures of the preceding paragraphs (especially Figure 157) it is obvious that this time SYNGAP1 could be localized at the cell membrane. The term “intrinsic to internal side of membrane” (GO:0031235), in addition, shows that it is indeed located in the intracellular area which was already previously called by the term “Cytoplasm”. With this result it was proven that the localization approaches of CmPI indeed can be used to predict localizations although they are currently not available in the databases.

The prerequisite is that other proteins are available which are known to interact with the protein featuring an uncertain localization. This might be just a protein set associated by experimental data, or pathway data from databases like KEGG or Reactome, or interaction networks created by tools like VANESA, ANDVisio, Cytoscape or VANTED etc. These networks can be imported in CmPI by using its import capabilities (Section 5.3.2.2.2).

6.4 [MES+MOL+FUN] Cm1 CellExplorer and CmX mRNA

The different aspects of the three levels of Integrative Cell Modeling were now discussed in the preceding sections. The final question which should be evaluated is how these quite different but complementary approaches can be combined.

6.4.1 Integrative Cell Models for Visualization

An important application for Integrative Cell Modeling is of course the visualization of cells as it was already mentioned in Section 4.1.1.4 discussing AutoCell. Also, the models created with different CELLmicrocosmos tools can be used to create visual demanding cell visualizations.

The cover image for the Bioinformatics issue of Informatik-Spektrum in 2009 was created this way [Somm09]. Figure 166 shows one of the three proposals submitted to Informatik-Spektrum. Finally, another image was chosen for the cover, but the presented image is even more appropriate to depict the integrative approach.

The whole cell model was created using CellEditor. With PathwayIntegration, the cell was combined with the citrate cycle as well as the glycolysis from KEGG as discussed in Section 6.3.1.1. The resulting model was exported to VRML with CellEditor. On the bottom of Figure 166 an outer mitochondrial membrane is shown. This model was created with MembraneEditor and then exported to PDB. Both results were then combined, visually optimized and rendered using 3ds max. The coloring is based on the H&E scheme (Section 5.1.2.3).

The mitochondrion in the foreground is the location of the citrate cycle. On the right side of the image the transition between the citrate cycle and the glycolysis is shown. Because the glycolysis is located in the cytosol, the metabolic reactions symbolized by the edges lead from the cytosol into the mitochondrion. Phosphoenolpyruvate/C00074 is transformed during the oxidative decarboxylation (in KEGG a part of the glycolysis) to pyruvate/C00022 and takes place in the matrix of the mitochondrion (Section 2.3.5). The other paths shown at the top of the image lead out of the mitochondrion and connect the citrate cycle with other metabolic networks.

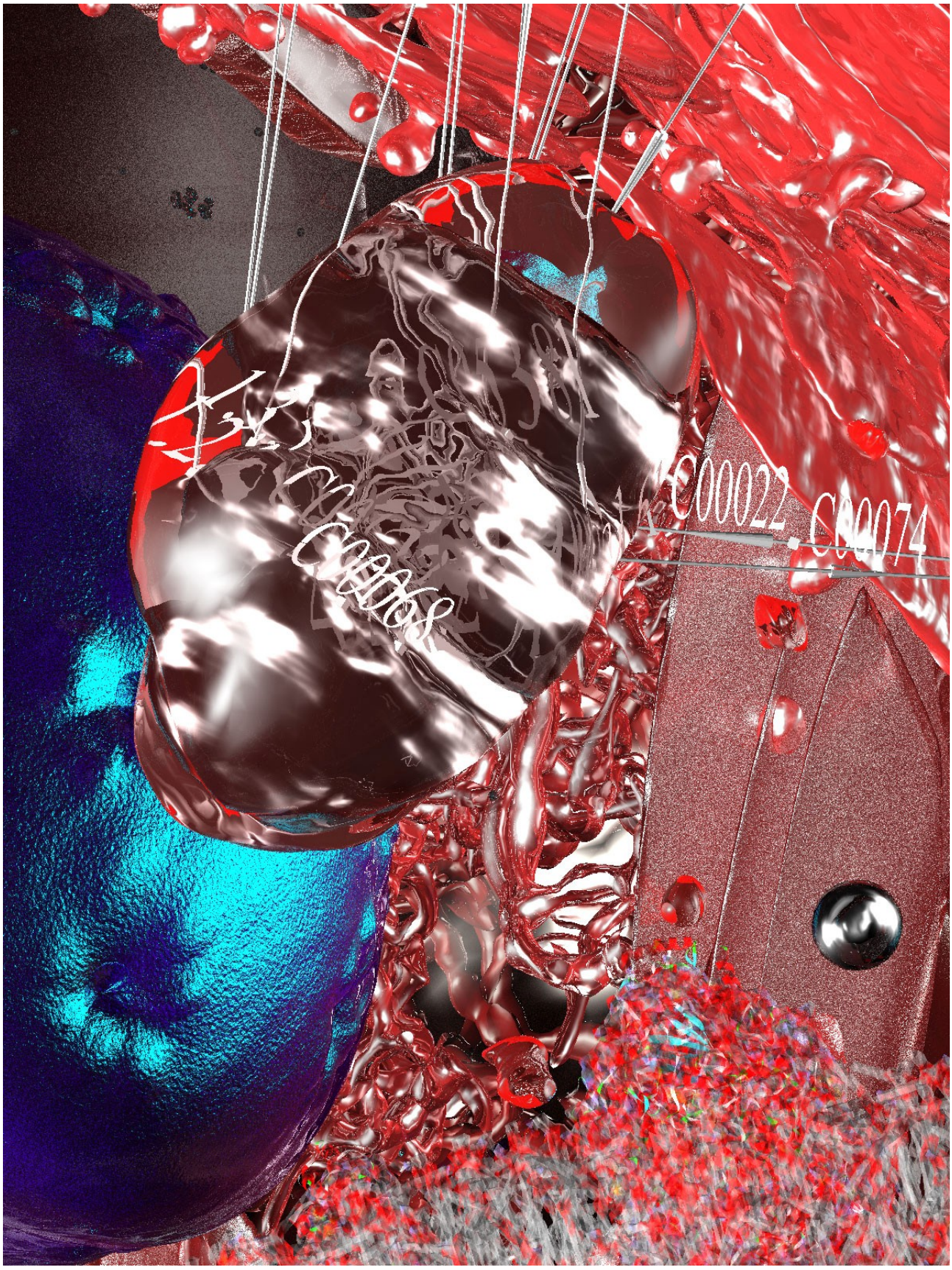


Figure 166: A proposal submitted for the cover image to Informatik Spektrum in 2009

6.4.2 An interactive Journey from the Mesoscopic to the Molecular Level

The previously discussed approach is of course a good choice to create demanding cell visualizations. But it is not possible to explorer the generated cell environment interactively; it is only a static representation. In this section it will be discussed, how CELLmicrocosmos tools can be used to explore an Integrative Cell Modeling environment interactively.

In Section 5.4.2.2 the workflow shows how a cell environment of CellExplorer is combined with PDB-membranes generated by MembraneEditor. In the according section, Figure 117 shows the methodology to combine a protein with a membrane. The shown membrane is the inner mitochondrial lipid bilayer generated in Section 6.2.1.3. But here another protein substitutes the PDB-model 1V55. The workflow image shows a cell environment correlated with the citrate cycle and the glycolysis as discussed in Section 6.3.1. The citrate cycle contains the enzyme 1.3.5.1. Using the described workflow (Figure 117), the PDB-based protein model 1YQ3 was found to belong to the EC family 1.3.5.1 (Section 3.3.2.2.2). Therefore it was downloaded from the OPM database and placed into the membrane.

Figure 167 now contains the resulting cell environment. The first image at the top shows the whole cell. Then the user navigates into the cell environment and the second image already shows that the inner mitochondrial membrane is associated with a texture based on the previously generated membrane. The third image provides a closer look and it can be seen, that also the outer membrane was already combined with a membrane model. Now, the node representing enzyme 1.3.5.1 is in the center of the image. Next, the option shown in Figure 117.9 is chosen: “Show Inner Membrane”. The PDB-membrane is mapped onto the surface of the associated Cell Component layer as can be seen in the fourth image. But also another aspect is apparent: the network is associated with the protein inside the membrane and the enzyme node is dismissed. The fifth and last image shows the atomic structure of the membrane – also the protein in the center of the membrane is visible.

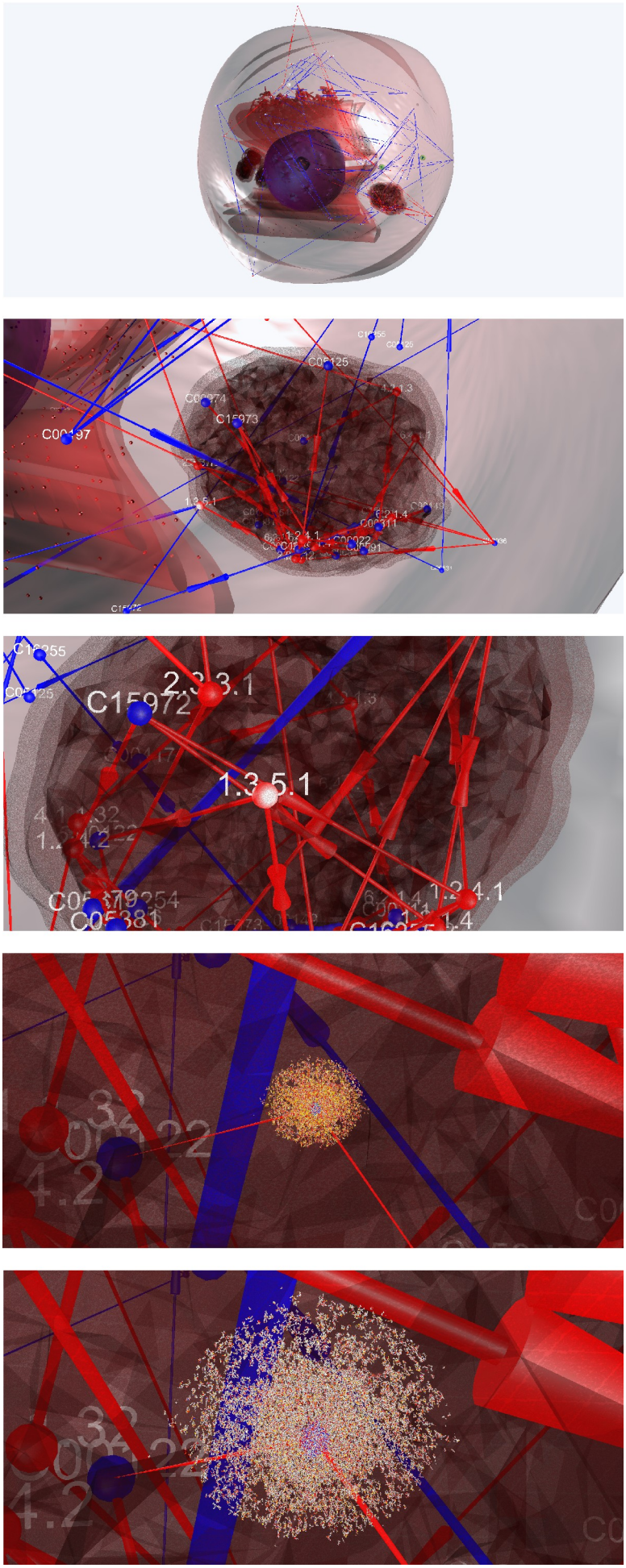


Figure 167: Journey from the Mesoscopic to the Molecular Level The PDB-based membrane modeled with MembraneEditor was associated with the inner mitochondrial membrane. In the center of the membrane the protein 1YQ3 represents the EC 1.3.5.1. The user navigates from the overview perspective (top) to the detailed atomistic view of the PDB-membrane mapped onto the surface of the mitochondrion (bottom)

6.4.3 An Animal Cell associated with publication-based Membranes

In the previous section the method was shown how to combine a single cell component with a membrane. In Section 2.2.6 different molecular membrane compositions were introduced. In this section, the membranes of a rat hepatocyte are correlated with a cell environment. For this purpose, the lipid distribution values found in Table 2 were normalized to 100%. Table 22 shows the computed values containing the following lipid types as obtained from the HIC-Up database (Section 3.2.2.2.1):

- cholesterol (Chol): clr_exp.pdb,
- diphosphatidyl glycerol (DPG): cdl_exp.pdb,
- phosphatidylcholine (PC): psc_exp.pdb,
- phosphatic acid (PA): 3ph_exp.pdb,
- sphingomyelin (SM): sphingomyelin.pdb,
- phosphatidyl glycerol (PG): aga_exp.pdb,
- phosphatidylethanolamine (PE): pev_exp.pdb,
- phosphatidylinositol (PI): pib_exp.pdb,
- phosphatidylserine (PS): ps2_exp.pdb, and
- cholesterol (Chol): clr_exp.pdb.

Figure 168 and 169 show the result: every Cell Component shown is associated with a membrane which was generated using MembraneEditor and the Advanced Random Placing Algorithm with standard settings (see also Section 5.2.2.5).

Cell Component	**Chol**	**PC**	**SM**	**PE**	**PI**	**PS**	**PG**	**DPG**	**PA**	**All**
Plasma Membrane	17.55	56.14		14.91	9.65		1.75			114
Rough ER	6,59	60.44	3.3	17.58	8.79	3.3				91
Smooth ER	9.38	51.59	11.26	19.7	6.29			1.78		106.6
Mito Inner	2.91	43.6	2.42	24.22	5.82	0.97	1.94	17.44	0.68	103.2
Mito Outer	4.25	53.02	5.3	24.39	3.18	2.12	2.65	3.71	1.38	94.3
Nuclear	10.1	55.56	3.03	20.2	7.07	3.03			1.01	99
Golgi	9.15	48.78	12.19	18.29	7.32	4.27				82
Lysosome	14.74	26.32	25.26	13.7	7.36	7.36		5.26		95

Table 22: Normalized Lipid Composition of a Rat Hepatocyte [Jain88, p.27]

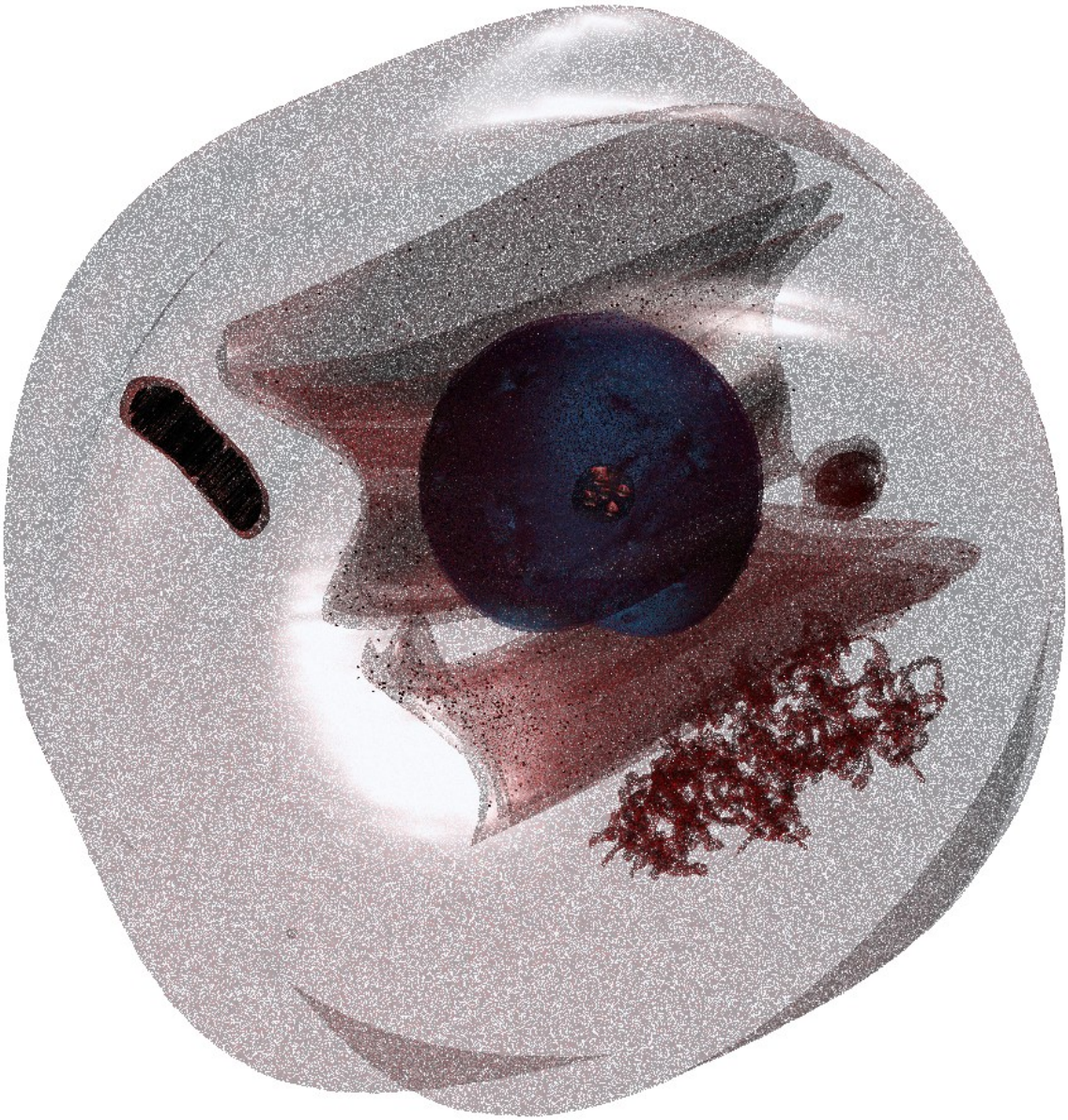


Figure 168: The Membranes of Table 22 applied to a cell environment: Mesoscopic View

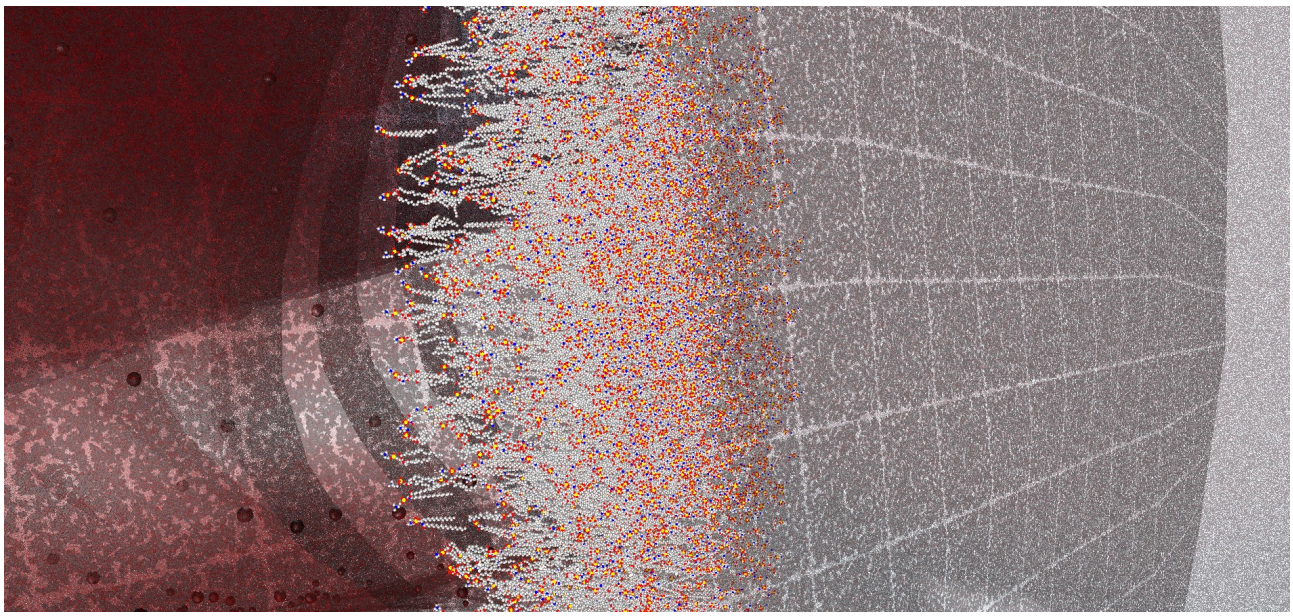
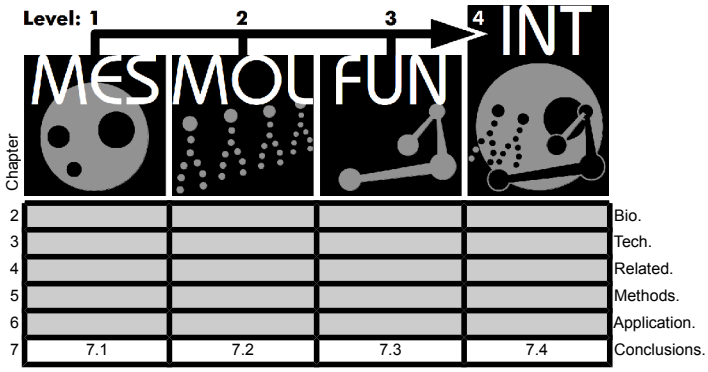


Figure 169: The Membranes of Table 22 applied to a cell environment: Molecular View

7 Conclusions and Outlook

In this work the status quo of Bioinformatics with respect to (Visual) Cell Modeling was evaluated. Different compatible modular tools were developed in the context of this work.

In this final chapter the achievements of the CELLmicrocosmos project should be discussed. The major focus will lie on those aspects introduced in this work. Furthermore, future outlooks are presented, including currently ongoing work. Finally, the impact of Cell Modeling on Synthetic Biology should be discussed, proposing a new direction for Cell Modeling applications.



7.1 [MES] Cell Modeling at the Mesoscopic Level

The CELLmicrocosmos approach of mesoscopic cell modeling is based on three-dimensional meshes of cell components. Four different abstraction levels for cell component models were introduced (Section 5.1.2.1).

0. This is the ideal level. Only cell models reflecting the realistic structure of a cell would be appropriate (the clone). Neither this work nor another approach is able to achieve this high level of realism.
1. 3D-microscopy-based or 3D-spectroscopy-based cell component models are directly derived from microscopic data sets (the image). For this purpose, tools like Amira were used in the context of this work to generate a mitochondrion model based on an image stack.
2. Interpretative cell component models are manually modeled by modeling tools (the allegory). These models are visually based on microscopic, usually two-dimensional images. Every cell component type introduced in this work is represented by at least one model generated with 3ds Max or Blender.
3. Abstract cell component models are applied to functional modeling (the abstraction). A correct biological topology may not be mandatory, depending on the application case. In this work, the SphereCell was introduced for this purpose.

A unique feature of CELLmicrocosmos cell component models is the sub-compartmental structure: the mitochondrial model, for example, contains an inner and outer membrane, the matrix, the intermembrane space and the cloud (Section 5.1.2.4).

The quality of the cell components was positively evaluated in the context of CellExplorer Educational Edition (CE^3) in two schools (Section 6.1.8). The future of this project will be to improve this software in order to enable the application in many secondary schools in Germany. One of the major precondition is already met by all CELLmicrocosmos tools; Java-Web-Start-compatibility (Section 5.1.3). With this feature it can normally be automatically installed by any computer connected to the Internet with operating systems such as Windows, Linux or Mac OS X.

Whereas the single three-dimensional cell component models are created with external programs, the cell environments are created by using CellEditor. It enables the modular generation of cell models which may be adjusted to the actual application case. For example, it might be necessary to show the relation of the nucleus and the endoplasmic reticulum. In this case, all other cell components might be excluded from the cell model or – alternatively – hidden.

In addition, two different cell types are available: the eukaryotic and the prokaryotic cell. During the development of CellEditor, some important changes were presented by the scientific community. While in the past there was only a differentiation between eukaryotic and prokaryotic cells, and archaea(-bacteria) were presented as a subclass of the last mentioned, today they are referred to as an independent class. Therefore, it will be a reasonable extension of CellEditor to also enable the modeling of these cell types. Prior to this extension, the prokaryotic cell type had to be used for this purpose.

A number of related approaches were introduced, but their application areas are quite different. This CellEditor should enable easy and basic modeling techniques which can be used by the bioinformatician as well as by the biologist, by the (school) student as well as the teacher. It is currently not capable of competing with the complex packing approaches of AutoFill/AutoCell, which has a completely different target group: (semi-)professional cell visualizers. It depends on existing, complex, partly commercial modeling tools like Cinema4D. The Graphite LifeExplorer which is in an early stage of development, is – like CmCE – a stand-alone program. But the target group of this program is also closer to the one of AutoFill/AutoCell than to the one of CmCE.

An important future goal is the integration of complete cell models directly derived from tomographic microscopy. For example, the CCDB contains a few of these models. Currently a program is being developed – the new version of CellEditor, 3.2 – which is improving the process of preparing such models for CellExplorer applications. For example, the generation of the different layers needed for models like the previously-mentioned mitochondrion should be automatized. In the past, every layer had to be modeled separately using programs like 3ds Max. The goal is to create only the major, visible shapes with other programs and use the new CellEditor to generate the additional needed shapes. In addition, it should enable external users to develop models compatible with CellExplorer without the need to search for structural requirements in the source code.

Another important feature of CellEditor is the export functionality which also enables viewing and editing of the generated cellular environment with other VRML97-supporting

applications. This is especially important if high quality rendered images should be created (Sections 6.1.7 and 6.4.1).

The focus of CellExplorer lies currently on a single cell. But it is already possible to add different instances of the same cell component – except the cell membrane. At the moment, the latter is unique. In the future it would be interesting to also integrate the capacity to administrate multiple cell models in one environment, to show, for example, the division of cells or cell-to-cell communication processes.

7.2 [MOL] Cell Modeling at the Molecular Level

The MembraneEditor (CmME) was shown as an appropriate and user-friendly tool to model membranes based on the PDB format. Its capabilities to generate multiple heterogeneous bilayer and/or monolayer membranes containing lipid rafts in combination with the semi-automatic protein placement makes it outstanding. Therefore, extensive modeling capabilities at the molecular level are provided as well.

For CmME, there are two main target groups. The first group is interested in modeling membranes based on published data. It was shown that the simulation of a membrane is not an obligatory condition to achieve realistic lipid densities (Sections 6.2.1.1 and 6.2.4). For this purpose different Lipid Packing Algorithms were integrated. In the context of this thesis, the algorithm achieving the highest lipid density in appropriate time (in the context of CmME) was developed: The Wanderer (Section 5.2.2.6). Even for relatively small membrane sizes[88] as well as lipid densities[89] published average values were maintained. Moreover it was shown that the Lipid Packing Problem is solved by all six introduced Lipid Packing Algorithms (LPA). For this purpose heterogeneous membranes were modeled combining experimental and ideal lipid models with the knowledge of different publications. Moreover, it was shown that efficient greedy algorithms are able to generate membrane models featuring realistic lipid ratios and density values. Another interesting fact is that the packing behavior of the cholesterol-containing membranes met the biological expectations of the umbrella model. At the moment, for many application cases the Distributor algorithm will be the first choice, because it is possible to define the absolute lipid values or the lipid density. But for densely packed membranes, The Wanderer will be the first choice. In addition, the demanding Protein Packing Problem was solved by using data from two databases: the PDBTM [Gamr09] and OPM (Section 6.2.3).

The second user group needs these LPA to generate molecular starting configurations for molecular simulations. It was already shown that CmME generates membranes which are compatible to tools like GROMACS [SDGS11]. Currently a lot of work was invested – especially in cooperation with Dr. Jens Krüger – to improve CmME to meet the needs of the GROMACS community. A special plugin is being developed, the GMX-Plugin, which should provide a bridge between CmME and GROMACS. It will be possible to run GROMACS on the local computer as well as to run it using a ssh-connection on an external computer

88 For membranes >50x50 Å² with The Wanderer and for membranes >100x100 Å² with Advanced Random Placing.

89 For phospholipids with lipid densities of approx. >50 Å².

cluster. Different tools will enable users who are not especially familiar with GROMACS to simulate membranes generated with CmME [RGKS12]. In the future it could be interesting to provide additional analysis capabilities in CmME to analyze aspects like kinks, rotations or cross-sectional areas of lipids [Träu71, WoRo96]. Another approach which might be interesting for the molecular simulation community is the integration of basic equilibration or simulation capabilities into CmME. While the regular LPP operate at the molecular level, moving the lipids only as stiff structures, this approach will also manipulate the intra-molecular configuration: the atomic structure. A first attempt was already implemented [Ding10], but the development of such Membrane Packing Algorithms (MPA) at the moment has only experimental character and will never be able – and is not intended to – compete with established simulation environments like GROMACS. In this context another interesting future perspective is the parallelization of packing algorithms. An appropriate base is LPA like the Advanced Random Placing. It was discussed that this algorithm generates different rectangular segments of the membrane iteratively. As a logical consequence each of these sub-problems could be solved by a single core or node of a multiprocessor environment or a cluster, vice versa.

In the future there might be also a third target group which may wish to use the plugin-capabilities of CmME to create custom MPAs or to generate derivatives of the published algorithms.

Packmol as well as CHARMM-GUI both provide some features which are similar to those of CmME, but both do not integrate a visualization-supported generation process and directly apply an equilibration. The latter aspect might not be applicable for all user groups. For example, the last mentioned user group usually will not be satisfied by an equilibration with external tools, because GROMACS already provides its native equilibration algorithms. In addition, CHARMM-GUI is closed source, this is another drawback for researchers who do not work with a CHARMM-based force field. Moreover, Packmol does not provide a GUI and is very demanding for users who are not familiar with the mathematical aspects of geometry. The other related approaches discussed in this work cannot compete with the broad and modular functionalities provided by CmME.

An appropriate term to describe the computational task of generating these membranes was theoretically described as the Bounded two-and-a-half-dimensional Knapsack Problem (2.5D-BKP, Section 5.2.2.5). Although CmME was optimized to solve these kinds of problems, the solution of three-dimensional packing problems is also possible. A first approach was introduced during a poster presentation at the Visualizing Biological Data Conference (VizBi) in Heidelberg in March 2012 [SoZh12]. Because of the scalability of CmME – which is especially achieved by the simplification of the atomic structures to shape-based models – it is now possible to generate ellipsoid vesicles. This capability should be extended in the future release of CmME (version 2.3). Currently, the performance of CmME is improved to enable the use of larger vesicular membranes [Unru12]. At any rate, the membrane sizes discussed by the MD community nowadays – ca. 250x250 $Å^2$ – can easily be handled by CmME as well as the limited capabilities of the PDB format (Section 3.2.2.1.1).

7.3 [FUN] Cell Modeling at the Functional Level

CellExplorer was used as a base to combine cell models with functional data. For example, this might be disease-related protein sets or metabolic pathways. At the moment, the localization capabilities of PathwayIntegration (CmPI) can only be used if protein-related data is used. For this purpose, CmPI connects to DAWIS-M.D., a data warehouse containing life-science relevant data, as well as ANDCell which provides textmining-derived data based on PubMed abstracts (Sections 3.3.3). The results achieved by querying these databases can be used to find multiple published localization results per protein. In the case of DAWIS-M.D., this data is usually reliable, because it is curated. In the case of ANDCell, the found localizations always have to be reviewed, because the text mining algorithms sometimes generate false positives which do not fit into the context. For this purpose, ANDCell always provides the sentence fragment the localization is based on together with the link to the PubMed entry.

Three different types of localizations are differentiated:

- the Localization Term is the original term derived from DAWIS-M.D. or ANDCell,
- the Cm4 Localization is the cell component which was assigned to the Localization Term by using the mapping table of CmPI, and
- the Cm4 Membrane Localization is the intra-compartmental layer contained in the mapping table.

CmPI extended the CmCX by multiple features which enable the visualization and analysis of protein localizations:

- The Localization Table lists all proteins and their actually selected localization. This table can be used to browse and select different potential localizations and to change the Cm4 (Membrane) Localization (Section 5.3.2.3.4).
- The Localization Charts provided can be used to gain a fast overview if dealing with multiple localization results. This is especially important if dealing with larger protein sets. These charts can be used to assign localizations to different selections of different localization types in a fast and easy way (Section 5.3.2.3.5).
- 2D Visualization is capable of showing the original layout of a pathway (Section 5.3.2.2.3). This may be useful if a metabolic pathway from KEGG was downloaded from DAWIS-M.D. or if a pathway was imported from a SBML-file. This tool provides a few basic edit functions as well as the option to directly navigate in the
- 3D Universe which is the original cell representation of CmCX.

The 3D Universe was extended by capabilities to correlate the cell components of CmCX with protein-related networks. For this purpose, multiple distribution algorithms have been implemented. For protein sets, the UUUSphere layout (Uniform Unique Unit Sphere, Section 5.3.2.5.1) provides different options to distribute the protein nodes equally on the surface of the cell component. For protein networks, the ISOM layout (Inverted Self-Organizing Maps, Section 5.3.2.5.2) was implemented. These algorithms operate on a unit

sphere. To place the protein nodes onto the surface of the cell components, they are basically mapped onto the point located at the surface of the associated cell component layer, lying between the position on the unit sphere and the center of the cell component.

It was shown that the Localization Charts can be used to analyze and predict pathway-dependent or protein-set-dependent localizations with a minimum of previous knowledge. For example, the publication-based localization of the citrate cycle to the mitochondrial matrix as well as the one of glycolysis to the cytosol is a simple task using Localization Charts (Section 6.3.1). Using of instead this, the Localization Table would be a time-consuming task. Using this tool, the researcher is able to analyze potential localizations in minutes whereas doing this manually by using the Internet would be – especially for large protein sets – a task of many days or even weeks. Different statistical categories were implemented for this purpose. A very useful filtering function is provided by the comparison of pair-wise localizations (Section 5.3.2.3.5).

Moreover, it was proven that a localization prediction of CmPI for the protein SYNGAP1 was used to generate a hypothesis, back in 2010, which proved now – as of 2012 – to be correct (Section 6.3.2.5).

For cell components as well as for the networks different coloring methods have been implemented (Sections 5.1.2.3 and 5.3.2.8). Two major approaches can be defined as

1. Contrast Color Codes which are reasonable for many distinct variables, and
2. Color Schemes which should be used if an internal order of the variables should be visualized.

Based on the Contrast Color Codes, a unique color was assigned to each cell component. These colors can be used in conjunction with the SphereCell, to visualize the localizations in the Localization Charts as well as in the 2D and 3D Visualization viewers.

In addition, the need for professional navigation was discussed and a 6DOF-navigation was implemented (Section 5.3.2.10).

In summation, there are the following target groups for CmPI:

1. researchers who want to analyze the localization of a set of proteins,
2. researchers who want to visualize the localization of a set of proteins in a Virtual Cell environment, and
3. in the future perhaps students who want to learn interactively about the spatial interrelationships in a cell.

A large variety of different interesting related approaches were introduced (Subchapter 4.3). It was shown that CmPI integrates some features also known from other approaches. For example, many tools also offer force-directed layouts to visualize networks in three dimensions. But there is no tool providing the option to correlate protein-related networks with various modular cell component based on database information. CmPI also provides the option to use the original two-dimensional layouts to navigate or mark a point of interest in 3D space. The latter feature is also known from HIVE. GEOMI is able to color nodes with respect to their localization, a feature which is also found in CmPI. Moreover,

the used Focus+Context paradigm was already implemented in one of the first tools in the area of 3D biological network visualization, the MNV.

The two layouts introduced in this work, the UUUSphere and ISOM layout, both have drawbacks. Future tasks will be the implementation of optimized layout algorithms which minimize the edge crossing. For example, during a student project there was already an approach successfully implemented to map the two-dimensional layouts of KEGG onto three-dimensional cell components. This will be especially an important extension for the "comparison cell"-related approaches which were introduced to support the pathway comparison among different species. Moreover, the integration of a powerful plugin-interface as known from CmME might be a reasonable future task.

But also the Localization Term mapping should be improved. At the moment, the mapping table is an integrated part of the program. It would be more convenient for the user, if this table could be edited. For example, if a new, unknown term in the database were found, the user should review this term and decide, which cell component (membrane) this term should refer to in the future. In addition, more databases providing localizations might be useful.

Increased import capabilities supporting more file formats is also an important obstacle to take. At the moment, only SBML Level 2 Version 3 and 4 and – of course – various native formats are supported. An important format for the future could be SBGN (System Biology Graphical Notation) support [NHMM09]. It would be important to enable the import of additional files which are usually used by the different target groups.

An important step to take would be the closer integration of ontologies known from The Gene Ontology or the Subcellular Anatomy Ontology (SAO) developed in the context of the CCDB [LFGC07]. At the moment, the mapping table of CmPI only maps some GO terms to the appropriate localization.

An important future perspective might be the mapping of diseases onto the cell. It was shown that this already possible with CmPI. But for future improvement a cooperation with experts in this field is indispensable. Databases like OMIM might be an interesting extension in the future [HSAB05].

The extension of the CmPI application into the fourth dimension – time – could be also a promising – but also very demanding – future task.

The stereoscopic capabilities of CmPI make it applicable to professional as well as consumer target groups (Section 5.3.2.11). Therefore, in the future this feature might be also interesting for school or bachelor students in the future to explain basic intracellular relations. For presentation purposes the use of a CAVE (CAVE Automatic Virtual Environment) could also be of interest.

7.4 [MES+MOL+FUN] Integrative Cell Modeling

The term "Cell Modeling" is – in biological context – often used in conjunction with mathematical modeling of cellular environments. Different approaches of this category have been introduced in this thesis (Section 4.1.2). To be more distinctive, these approaches were categorized as Mathematical Cell Modeling.

To judge if this word appropriately describes the work introduced here, a look into Webster's English Dictionary is sufficient. The term “modeling” is defined as follows [Read03, p.818]:

- “to make a model of”,
- “to make a model or models”, and
- “to assume the appearance of natural form”.

In addition, the term “model” has to be investigated [Read03, pp.817–818]:

- “an object, usually in miniature, representing accurately something to be made or already existing”,
- “a thing or person to be imitated or patterned after; that which is taken as an example”, and
- “that which strikingly resembles something else; an approximate copy or image”.

Because the main focus of this work is to combine different computational approaches to design a Virtual Cell environment – an approximate copy of the natural form of the cell – the term modeling is appropriate.

Two major strategies were introduced.

The top-down approach starts at the mesoscopic level: cell components are modeled and combined with cell environments. Starting with the cell membrane as the enclosing envelope, smaller components are added and distributed throughout the cell. This can start on the scale of a few thousand nanometers and go down to shapes of molecular machines like the ribosomes with the size of a few tens of nanometers. And these small shapes may be derived from atomic structures, but they also may be generated by using image stacks containing information to approximate the structure of a mitochondrion model. There are databases like the CCDB which provide such datasets, and for generating shapes based on atomic structures, the PDB can be used as well. CellEditor was presented as the appropriate tool to combine these structures at the mesoscopic level (Sections 5.1.2.1.1 and 5.1.2.1.2).

A typical problem of this mesoscopic approach was already discussed in literature, to give an example:

“Abstraction in visualization is useful to gain a global perspective; however, this is done at the risk of oversimplification.” [HoWW07, p.110]

The simplification is of course an intended feature of the top-down approach which is reflected by the Bauhaus style of the CELLmicrocosmos project: the complex, overwhelming reality of the cell is hidden beyond the membrane-based surfaces of the cell component models to depict the architectural structure of the cell.

But to address the aforementioned problem of oversimplification, the implemented bottom-up approach – the second modeling strategy – starts from the molecular structures defined by their atomic positions. Here, concrete data from publications can be used to model membranes in a relatively accurate way; “relatively”, because the knowledge about the concrete atomic structure of all cellular components is still very fragmentary. Focusing on

the complete cell, “islands of knowledge” can be created. These are small fragments on the surface of a membrane trying to create an approximate copy of a natural membrane segment. This is also a typical approach implementing the “Focus+Context” paradigm of Robinson and Flores [RoFl97]: while the view of the surrounding superstructure is always present, it is possible to focus on small but concrete aspects in a cosmos of matter. And for modeling exactly these islands of knowledge, MembraneEditor was presented as the software of choice, using databases like the PDB or HIC-UP for structural data and OPM and PDBTM for protein-alignment data (Section 3.2.2.1).

For future development, these islands of knowledge could be an interesting aspect. While the cell environment could be the (cyto-)skeleton of a whole project in which different working groups are involved, every group could work on a special island of knowledge. While the first group generates and simulates the realistic structure of a mitochondrial membrane, the second group generates the mitochondrion model based on new microscopic data sets, the third group works on a cytosol-located structure of a large protein complex and the fourth group models and analyzes a protein-related network, which is related to the previous-mentioned protein complex as well as the mitochondrial membrane. The overall visualization represented by the resulting cell environment could be the base for discussions and analyses.

While the first (top down) approach mainly combines Bioinformatics with three-dimensional modeling techniques, the second (bottom-up) approach is more related to Chemoinformatics, especially if generated membrane models are used as a base for molecular simulations.

Obviously, a bi-directed approach is used. Moreover, the different levels are correlated by the functional level in between: the pathways which were integrated using PathwayIntegration. Originally, the correlation was intended to work only at the mesoscopic level. But by using the new CmX tool, CellExplorer and MembraneEditor are interconnected and it is possible to also correlate the molecular structures with the functional level temporarily (Section 5.4.2.2). For the purpose of protein localization, the databases UniProt, Brenda, GO, Reactome and ANDCell are used, whereas metabolic pathways can be added by using KEGG.

Different packing and mapping problems were computationally solved at the Mesoscopic as well as molecular level:

- Membrane Packing Problem (Section 3.2.4) with the sub-problems:
  - Protein Packing Problem, and
  - Lipid Packing Problem;
- Network Mapping Problem (Section 3.3.4) with the sub-problems:
  - Node Distribution Problem,
  - Network Layout Problem, and
  - Node Mapping Problem.

To the knowledge of the author there are nearly no similar tools in this new category combining all these problems which should be appropriately described as “Integrative Cell Modeling”. The Interactorium and MetNetVR are tools which in part implement similar methods to visualize cellular complexes, but both environments are not intended to model cells.[90]

The only tools introduced here, which are trying to combine the mesoscopic as well as the molecular level – but not the functional level – are Graphite LifeExplorer and AutoFill/AutoCell. The latter plugin seems to have potential for becoming the state of the art in the future of modeling cells for animation purposes – for example, for making a molecular movie. Here, similar approaches as they are known from MembraneEditor are applied to whole cell environments. But as previously mentioned, the target group differs significantly from the one of the CELLmicrocosmos tools.

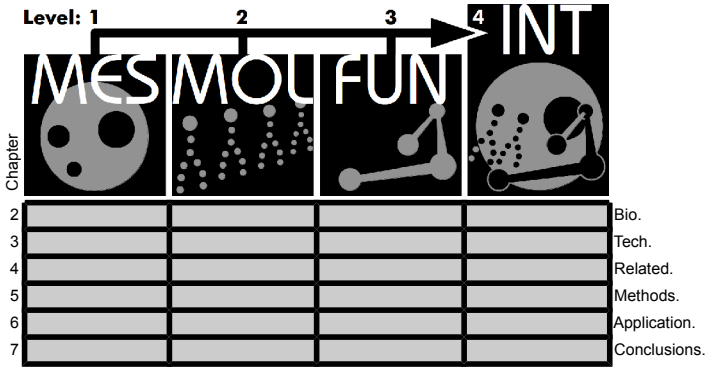
Of course, the generated cell environment in CellExplorer represents only a snapshot of the cell. Simulation results can only be integrated by importing membranes generated with MembraneEditor and simulated with tools like GROMACS (Section 5.2.2.9). Therefore, it is possible to import molecular structures whose atomic compositions are closer to their natural form.

Finally, an elementary question should be answered, when approaching the third dimension from a two-dimensional perspective: “why should a three-dimensional approach be used to visualize metabolic pathways if it is much more simple to generate and to interpret a two-dimensional image? Why merge different domains?”[91]

Whereas for many visualization approaches it will be sufficient to work with synthetic models, the chance to import realistic structures – complete cells directly generated from microscopic data – is an important future task to solve. Therefore, CELLmicrocosmos provides tools for day X, when it is possible to combine a realistic cell structure with modeled and simulated atomic structures as well as the complete intracellular metabolic network. Looking at a two-dimensional projection of a three-dimensional ISOM layout, much edge-crossing is evident. Of course, if this image is seen in a book, a two-dimensional layout would be more reasonable. But the reality of the cell is three-dimensional. And therefore the cell component models are also spatial representations. In addition, areas of interest are located in special positions in three-dimensional space. Therefore, if a realistic view of a cell should be created, making full use of the spatial perception of an individual, it is indispensable to correlate networks in three-dimensional space. And therefore it is important, to integrate features like a 6DOF-navigation and 3D Stereoscopy.

90 Anyway, a powerful feature of the Interactorium and GEOMI should be mentioned again: the grouping of molecular complexes. This is a reasonable approach which could be also interesting for the future development of CmX.

91 Similar questions are often asked by experts like David S. Goodsell in conjunction with three-dimensional visualization.



7.5 Synthetic Cell Modeling

Finally, the question should be evaluated, what the future of Cell Modeling might be. First, the category should be made clear: cell modeling is a sub-category of Systems Biology, as shown in Figure 87. The spherical representations should depict that Cell Modeling is a part of a large – theoretically unlimited – number of different approaches belonging to Systems Biology, which are not shown in this image. Systems Biology provides – usually strongly simplified – approaches relevant to understanding biological processes prior to the construction of new biological systems. And especially these biological systems are the major topic of Synthetic Biology. Two major directions are known. The minimal cell is a biological cell which contains only those elements needed for maintaining the vital functions. The protocell is mainly used for self-replication purposes and is completely synthetic, and in some cases, it may not even have to show the key features of life [PüMW11].

An important breakthrough for Synthetic Biology was the work published by Venter et al. in 2007 [LGAP07]. This group managed to transplant a genome of a *Mycoplasma mycoides* cell (Figure 2) into the envelope of a *Mycoplasma capricolum* cell. Because the genome maintained its functionality, on living organism was changed to another one.

It is a controversial question if Synthetic Biology is guided by the vision to *create* life. But it can be stated that a major objective of Synthetic Biology is to *understand* life; what are the minimal requirements for life? A common public misinterpretation of cytology is that the cell *is* the smallest unit of life. This is indeed an oversimplification which should be dissipated by Synthetic Biology in the near future.

The initial question of this subchapter can now be specified more clearly: what is the impact of Cell Modeling on Synthetic Biology? As mentioned before, Systems Biology approaches are usually used to model, predict or simulate models derived from nature. For understanding the complex interrelationships relevant for wet-lab experiments, it is indispensable to use computational methods. Molecular simulations like GROMACS or mathematical simulations like ENVIRONMENT are trying to predict biological behavior at the molecular and mesoscopic level, and vice versa. Similar to Integrative Cell Modeling, Synthetic Biology also applies top-down as well as bottom-up approaches: top-down, by starting with a living cell, removing the maximum possible genetic material while simultaneously preserving viability and bottom-up, by constructing an envelope and adding components in order to generate a biological cell [PuWe09].

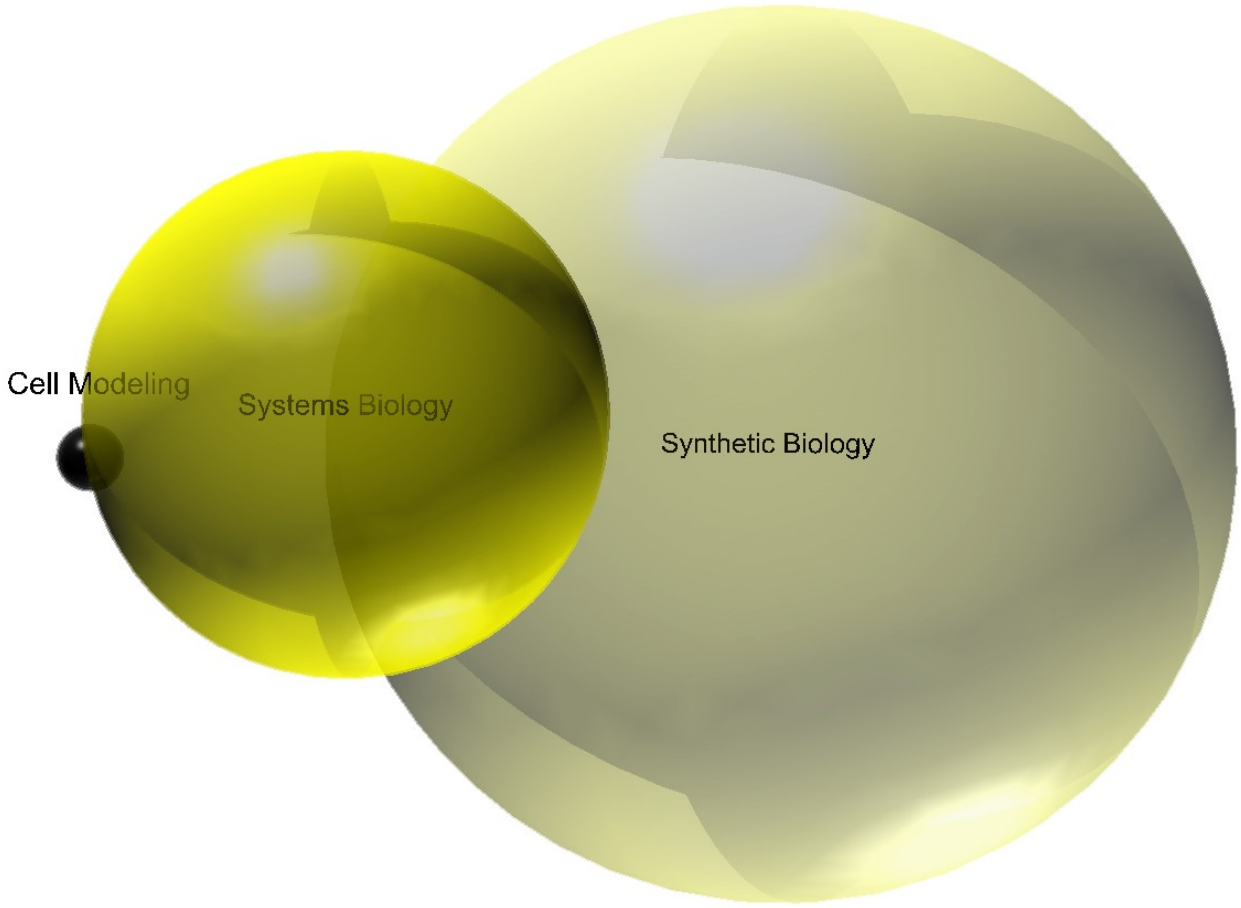


Figure 170: Interrelation of Synthetic Biology, Systems Biology and Cell Modeling in the context of this work

Another aspect for the bidirectional character of Integrative Cell Modeling was shown by the different, interdisciplinary natural paradigms used to solve the different computational problems like the Self-Organizing Maps or the Geodesic Principles.

An important aspect of Synthetic Biology is its dependence on interdisciplinary work. A large number of highly-specialized disciplines have to be combined to achieve the intended results. But while Systems Biology was usually a service provider for experimental work, the future must be a close, direct and bidirectional cooperation between the experimentalist and the theoretic researcher applying computational methods. The vision is the combination of all disciplines to be able to accurately simulate all intracellular processes on the mesoscopic, molecular and functional level *in silico* and – most importantly – to observe the same processes *in vivo*. And in order to enable an accurate simulation, an obligatory characteristic of such an environment is the use of a realistic cell model, trying to reduce the level of abstraction to a minimum.

For this purpose, the principles of Integrative Cell Modeling have to be extended by one more aspect into the fourth dimension; the time-dependent simulation which can be experimentally evaluated, generating a cyclic feedback system between in silico and in vitro/vivo experiments. Synthetic Cell Modeling.

8 References

[00a] *Jmol: an open-source Java viewer for chemical structures in 3D*. http://jmol.sourceforge.net (accessed May 3, 2010)

[00b] Google Scholar

[07] *E-Cell Simulation Environment 3D*. URL http://ecell3d.iab.keio.ac.jp/index.html. - retrieved on 11/19/2011

[10a] *ChemSite Pro*. Fairfield, USA : ChemSW, 2010

[10b] *Molecular Operating Environment (MOE)*. Montreal, Canada : Chemical Computing Group Inc., 2010

[10c] *Bundesministerium für Bildung und Forschung: Internationale Grüne Woche Virtual Market Place*. URL http://www.virtualmarket.gruenewoche.de/index.php5?id=176767&highlight=bmbf&fid=649&offset=0&Action=showCompany. - retrieved on 01/10/2012

[11a] *Home : EMDataBank.org*. URL http://www.emdatabank.org/index.html. - retrieved on 08/31/2011

[11b] *Methods - About OPM - Orientations of Proteins in Membranes (OPM) database*. URL http://opm.phar.umich.edu/about.php?subject=methods. - retrieved on 08/30/2011

[11c] *Topology definitions - About OPM - Orientations of Proteins in Membranes (OPM) database*. URL http://opm.phar.umich.edu/about.php?subject=topology. - retrieved on 08/30/2011

[11d] *HIC-Up*. URL http://xray.bmc.uu.se/hicup/. - retrieved on 08/30/2011

[11e] *Avanti Polar Lipids*. URL http://www.avantilipids.com. - retrieved on 02/17/2011

[11f] *WayBackMachine: Klotho: Biochemical Compounds Declarative Database*. URL http://web.archive.org/web/20110725074536/http://www.biocheminfo.org/klotho/. - retrieved on 08/02/2012

[11g] *GeneOntology.org*. URL http://www.webcitation.org/636UO9y6J. - retrieved on 02/25/2012. — The Gene Ontology

[11h] *Meta!Blast*. URL http://www.metablast.org/. - retrieved on 11/18/2011. — MetaBlast.org

[11i] *VCell The Virtual Cell › University of Connecticut Health Center - Know Better Care*. URL http://www.nrcam.uchc.edu/vcell_software/user_guide.html?current=four. - retrieved on 11/20/2011

[11j] *FrontPage - CompuCell3D*. URL http://compucell3d.org/. - retrieved on 11/19/2011

[11k] *CHARMM-GUI*. URL http://www.charmm-gui.org/. - retrieved on 08/04/2011

[11l] *Web of Knowledge [v5.3]*. URL http://sub3.webofknowledge.com. - retrieved on 08/07/2011

[11m] *yWorks - The Diagramming Company*. URL http://www.yworks.com/en/index.html. - retrieved on 11/21/2011

[11n] *Cortona3D software creates interactive 2D and 3D technical communications and documentation*. URL http://www.cortona3d.com/. - retrieved on 12/31/2011

[11o] *Igor Rojdestvenski: VRML in scientific visualization*. URL http://web.archive.org/web/20060221054441/http://www.patronov.net/. - retrieved on 12/31/2011

[11p] *Rendware.com - 3D animation and visualization*. URL

http://www.rendware.com/download_scape3d.html. - retrieved on 01/03/2012

[11q] *uni.aktuell: Bielefelder Nachwuchswissenschaftler ausgezeichnet*. URL http://www.webcitation.org/query?url=http%3A%2F%2Fekvv.uni-bielefeld.de%2Fblog%2Funiaktuell%2Fentry%2Fbielefelder_nachwuchswissenschaftler_ausgezeichnet&date=2012-09-01. - retrieved on 09/01/2012

[11r] *Atomic Coordinate Entry Format Version 3.3*. URL http://www.wwpdb.org/documentation/format33/v3.3.html. - retrieved on 08/29/2011

[12a] *3ds Max - 3D Modeling, Animation, and Rendering Software - Autodesk*. URL http://usa.autodesk.com/3ds-max/. - retrieved on 01/07/2012. — autodesk.com/3ds-max/

[12b] *RCSB PDB - Holdings Report*. URL http://www.pdb.org/pdb/statistics/holdings.do. - retrieved on 08/02/2012

[12c] *PubMed*. URL http://www.webcitation.org/65fs6c17R. - retrieved on 02/27/2012

[12d] *java.com*. URL http://www.java.com/de/. - retrieved on 01/08/2012

[12e] *Java 3D Parent Project — Java.net*. URL http://java3d.java.net/. - retrieved on 01/08/2012

[12f] *ALICEpedia - Graphite*. URL http://www.webcitation.org/query?url=http%3A%2F%2Falice.loria.fr%2FWIKI%2Findex.php%2FGraphite%2FGraphite&date=2012-08-09. - retrieved on 08/09/2012

[12g] *BiNoM: Biological Network Manager Cytoscape plug-in*. URL http://bioinfo-out.curie.fr/projects/binom/. - retrieved on 01/24/2012

[12h] *Nature Precedings ceased accepting new submissions*. URL http://www.webcitation.org/query?url=http%3A%2F%2Fprecedings.nature.com&date=2012-08-08. - retrieved on 08/08/2012

[ABBB00] Ashburner, M.; Ball, C. A.; Blake, J. A.; Botstein, D.; Butler, H.; Cherry, J. M.; Davis, A. P.; Dolinski, K. et al.: The Gene Ontology Consortium (2000) Gene Ontology: tool for the unification of biology. In: *Nature Genetics* 25 (2000), pp. 25–29

[ABLR83] Alberts, B.; Bray, D.; Lewis, J.; Raff, M.; Roberts, K.; Watson, J. D.: *Molecular Biology of the Cell*. 1. ed. New York, USA : Garland Publishing, Inc., 1983 — ISBN 08240728272820

[ACZL10] Andrei, R. M.; Callieri, M.; Zini, M. F.; Loni, T.; Maraziti, G.; Pan, M. C.; Zoppè, M.: BioBlender: a software for intuitive representation of surface properties of biomolecules. In: *Arxiv preprint arXiv:1009.4674* (2010)

[AJLR02] Alberts, B.; Johnson, A.; Lewis, J.; Raff, M.; Roberts, K.; Walter, P.: *Molecular Biology of the Cell*. 4. ed. : Garland Science, 2002 — ISBN 0815332181

[AKKB09] Arnold, K.; Kiefer, F.; Kopp, J.; Battey, J. N. D.; Podvinec, M.; Westbrook, J. D.; Berman, H. M.; Bordoli, L. et al.: The Protein Model Portal. In: *Journal of Structural and functional Genomics* 10 (2009), no. 1, pp. 1–8

[AlDH10] Alwarawrah, M.; Dai, J.; Huang, J.: A molecular view of the Cholesterol Condensing Effect in DOPC lipid bilayers. In: *The Journal of Physical Chemistry B* 114 (2010), no. 22, pp. 7516–7523

[AMPG07] Atshaves, B. P.; McIntosh, A. L.; Payne, H. R.; Gallegos, A. M.; Landrock, K.; Maeda, N.; Kier, A. B.; Schroeder, F.: SCP-2/SCP-x gene ablation alters lipid raft domains in primary cultured mouse hepatocytes. In: *Journal of Lipid Research* 48 (2007), no. 10, pp. 2193–2211

[Arbo08] ARBOREUS: *Cardiomyocytes | Flickr - Fotosharing!* URL http://www.flickr.com/photos/arboreus/2679987015/. - retrieved on 01/03/2012

[ArYT08] ARAKAWA, K.; YACHIE, N.; TOMITA, M.: Visualizing complex omics information - scientific visualization for genomics and systems biology. In: *BIOforum Europe 2008* 6 (2008), pp. 27–29

[Bala06] BALABIN, Ilya: *Membrane Plug-in, Version 1.1*. http://www.ks.uiuc.edu/Research/vmd/plugins/membrane/ (accessed October 1 2010), 2006

[BBMN09] BROOKS, B. R.; BROOKS III, C. L.; MACKERELL JR, A. D.; NILSSON, L.; PETRELLA, R. J.; ROUX, B.; WON, Y.; ARCHONTIS, G. et al.: CHARMM: the biomolecular simulation program. In: *Journal of Computational Chemistry* 30 (2009), no. 10, pp. 1545–1614

[BeTS06] BERG, J. M; TYMOCZKO, J. L; STRYER, L.: *Biochemistry*. 6th. ed. New York, USA : WH Freeman & Co, 2006

[BFEP04] BECK, M.; FÖRSTER, F.; ECKE, M.; PLITZKO, J. M.; MELCHIOR, F.; GERISCH, G.; BAUMEISTER, W.; MEDALIA, O.: Nuclear pore complex structure and dynamics revealed by cryoelectron tomography. In: *Science* 306 (2004), no. 5700, pp. 1387–1390

[BGHM07] BARSKY, A.; GARDY, J. L.; HANCOCK, R. E. W.; MUNZNER, T.: Cerebral: a Cytoscape plugin for layout of and interaction with biological networks using subcellular localization annotation. In: *Bioinformatics* 23 (2007), no. 8, pp. 1040–1042

[BHHI09] VAN BEURDEN, M.; VAN HOEY, G.; HATZAKIS, H.; IJSSELSTEIJN, W. A.: Stereoscopic displays in medical domains: a review of perception and performance effects, Human Vision and Electronic Imaging XIV. In: *Proceedings of the SPIE*. vol. 7240, 2009, p. 72400A

[BMHM09] BENNETT, W. F. D.; MACCALLUM, J. L.; HINNER, M. J.; MARRINK, S. J.; TIELEMAN, D. P.: Molecular view of cholesterol flip-flop and chemical potential in different membrane environments. In: *Journal of the American Chemical Society* 131 (2009), no. 35, pp. 12714–12720

[Bour96] BOURKE, P.: *Distributing Points on a Sphere*. URL http://www.webcitation.org/query?url=http%3A%2F%2Flocal.wasp.uwa.edu.au%2F%7Epbourke%2Fgeometry%2Fspherepoints%2F&date=2012-03-20. - retrieved on 03/20/2012. — http://local.wasp.uwa.edu.au/~pbourke/geometry/spherepoints/

[Bouv99] BOUVIER, D. J.: Getting Started with the Java 3D™ API, Sun MicroSystems Inc (1999)

[Bran05] BRAND, R.: *3D–Visualisierung metabolischer Pfade mittels eines kraftbasierten Layoutverfahrens*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2005

[BrBO83] BROOKS, B. R.; BRUCCOLERI, R. E.; OLAFSON, B. D.: CHARMM: a program for macromolecular energy, minimization, and dynamics calculations. In: *Journal of Computational Chemistry* 4 (1983), no. 2, pp. 187–217

[BrDS04a] BRANDES, U.; DWYER, T.; SCHREIBER, F.: Visualizing related metabolic pathways in two and a half dimensions. In: *Graph Drawing* : Springer, 2004, pp. 111–122

[BrDS04b] BRANDES, U.; DWYER, T.; SCHREIBER, F.: Visual understanding of metabolic pathways across organisms using layout in two and a half dimensions. In: *Journal of Integrative Bioinformatics* 1 (2004), no. 1

[Brew56] BREWSTER, D. S.: *The Stereoscope; its history, theory, and construction, with its application to the fine and useful arts and to education: with fifty wood engravings* : John Murray, 1856

[BrFr96] Bruss, I.; Frick, A.: Fast interactive 3-D graph visualization. In: *Graph Drawing* (1996), pp. 99–110

[Burg08] Burgan, M.: *Robert Hooke: natural philosopher and scientific explorer* : Compass Point Books, 2008 — ISBN 0756533155

[BWFG00] Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E.: The Protein Data Bank. In: *Nucleic acids research* 28 (2000), no. 1, pp. 235–242

[CaAz07] Camargo, A.; Azuaje, F.: Linking gene expression and functional network data in human heart failure. In: *PloS One* 2 (2007), no. 12, p. e1347

[CaBM97] Carey, R.; Bell, G.; Marrin, C.: Iso/iec 14772-1: 1997 virtual reality modeling language (vrml97). In: *The VRML Consortium Incorporated* 34 (1997)

[CaCa82] Carter, R. C.; Carter, E. C.: High-contrast sets of colors. In: *Applied Optics* 21 (1982), no. 16, pp. 2936–2939

[Caga94] Cagan, J.: Shape annealing solution to the constrained geometric knapsack problem. In: *Computer-Aided Design* 26 (1994), no. 10, pp. 763–770

[CaMF03] Campagna, J. A.; Miller, K. W.; Forman, S. A.: Mechanisms of actions of inhaled anesthetics. In: *New England Journal of Medicine* 348 (2003), no. 21, p. 2110

[Cant01] Cantor, R. S.: Breaking the Meyer-Overton rule: predicted effects of varying stiffness and interfacial activity on the intrinsic potency of anesthetics. In: *Biophysical Journal* 80 (2001), no. 5, pp. 2284–2297

[CaSY02] Cagan, J.; Shimada, K.; Yin, S.: A survey of computational approaches to three-dimensional layout problems. In: *Computer-Aided Design* 34 (2002), no. 8, pp. 597–611

[CBBK10] Costanzo, M.; Baryshnikova, A.; Bellay, J.; Kim, Y.; Spear, E. D.; Sevier, C. S.; Ding, H.; Koh, J. L. Y. et al.: The genetic landscape of a cell. In: *Science* 327 (2010), no. 5964, pp. 425–431

[Cens05] Census of Marine Life E&O: *Polykrikos_lebourae | Flickr - Fotosharing!* URL http://www.flickr.com/photos/55038698@N03/5224519414/. - retrieved on 01/03/2012

[CHCG05] Cickovski, T. M.; Huang, C.; Chaturvedi, R.; Glimm, T.; Hentschel, H. G. E.; Alber, M. S.; Glazier, J. A.; Newman, S. A. et al.: A framework for three-dimensional simulation of morphogenesis. In: *Computational Biology and Bioinformatics, IEEE/ACM Transactions on* 2 (2005), no. 4, pp. 273–288

[CHDW07] Call, A. B.; Hernnstadt, S.; Dickerson, J.; Wurtele, E. S.; Bassham, D.: Virtual Cell: A pedagogical convergence between game design and science education. In: *Journal of Systemics, Cybernetics and Informatics* 5 (2007), pp. 27–31

[CiOS11] Civico, M.; Orlik, R.; Sommer, B.: CE^3 – an eLearning Software for cytological education.

[Civi11] Civico, M.: *Entwicklung und Evaluierung der Educational Edition des CELLmicrocosmos CellExplorer*. Bielefeld, Germany, Bielefeld University, Master Thesis, 2011

[CKSA12] Chan, J.; Kishore, R.; Sternberg, P.; Van Auken, K.: The Gene Ontology: enhancements for 2011. In: *Nucleic Acids Research* 40 (2012), no. D1, pp. D559–D564

[CoHa07] Cooper, G. M.; Hausman, R. E.: *The cell: a molecular approach*. 4. ed. Washington, D.C., USA : ASM Press, 2007

[Comp11] comp.graphics.algorithms FAQ: *Evenly distributed points on sphere - CGAFaq*. URL http://www.webcitation.org/query?url=http%3A%2F%2Fcgafaq.info%2Fwiki%2FEvenly_distributed_points_on_sphere&date=2012-03-20. - retrieved on 03/20/2012. — http://cgafaq.info/wiki/Evenly_distributed_points_on_sphere

[Coop97] Cooper, G. M: *The cell: a molecular approach*. 1. ed. Oxford, GB : Oxford University Press, 1997

[COWH11] Croft, D.; O'Kelly, G.; Wu, G.; Haw, R.; Gillespie, M.; Matthews, L.; Caudy, M.; Garapati, P. et al.: Reactome: a database of reactions, pathways and biological processes. In: *Nucleic Acids Research* 39 (2011), no. suppl 1, p. D691

[CVPB04] Coyne, C. B.; Voelker, T.; Pichla, S. L.; Bergelson, J. M.: The coxsackievirus and adenovirus receptor interacts with the multi-PDZ domain protein-1 (MUPP-1) within the tight junction. In: *Journal of Biological Chemistry* 279 (2004), no. 46, pp. 48079–48084

[DaDa35] Danielli, J. F.; Davson, H.: A contribution to the theory of permeability of thin films. In: *Journal of Cellular and Comparative Physiology* 5 (1935), no. 4, pp. 495–508

[DaES10] Davey, N. E.; Edwards, R. J.; Shields, D. C.: Computational identification and analysis of protein short linear motifs. In: *Frontiers in Bioscience* 15 (2010), pp. 801–825

[Davi05] Davison, A.: *Killer game programming in java* : O'Reilly Media, Inc., 2005 — ISBN 0596007302

[Davi07] Davison, A.: *Pro Java 6 3D Game Development: Java 3D, JOGL, JInput, and JOAL APIs*. New York, USA : Springer, 2007 — ISBN 1590598172

[DBYS09] Dutta, S.; Burkhardt, K.; Young, J.; Swaminathan, G. J.; Matsuura, T.; Henrick, K.; Nakamura, H.; Berman, H. M.: Data deposition and annotation at the worldwide protein data bank. In: *Molecular Biotechnology* 42 (2009), no. 1, pp. 1–13

[DDGL10] Du, Z. Y.; Degrace, P.; Gresti, J.; Loreau, O.; Clouet, P.: Dissimilar properties of vaccenic versus elaidic acid in β-oxidation activities and gene regulation in rat liver cells. In: *Lipids* (2010), pp. 1–11

[DeAI08] Demenkov, P. S.; Aman, E.; Ivanisenko, V. A.: Associative Network Discovery (AND) - the computer system for automated reconstruction networks of associative knowledge about molecular-genetic interactions. In: *Computational Technologies* 13 (2008), no. 2, pp. 15–19

[DFHW02] Dunford-Shore, B.; Fabrizio, F.; Holcomb, J.; Wise, W.; Feng, B.; Sulaman, W.; Sanghi, G.; Sadekar, S. et al.: *Klotho: biochemical compounds declarative database. http://www.biocheminfo.org/klotho/ (accessed May 3, 2010)*. URL http://www.biocheminfo.org/klotho/. - retrieved on 02/16/2011

[DHAS12] Dimmer, E. C.; Huntley, R. P.; Alam-Faruque, Y.; Sawford, T.; O'Donovan, C.; Martin, M. J.; Bely, B.; Browne, P. et al.: The UniProt-GO Annotation database in 2011. In: *Nucleic Acids Research* 40 (2012), no. D1, pp. D565–D570

[Ding10] Dingersen, T.: *Die Generierung energetisch günstiger Membransegmente in CELLmicrocosmos 2.2 MembraneEditor durch die Erweiterung auf eine atomare Behandlung der Moleküle*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2010

[Durg98] Durgin, G. D.: *Advanced site-specific propagation prediction techniques*. Virginia, USA, Virginia Polytechnic Institute and State University, Master Thesis, 1998

[DwEc04] Dwyer, T.; Eckersley, P.: WilmaScope—a 3D graph visualization system. In: *Graph Drawing Software* (2004), pp. 55–75

[DYBR03] Dickerson, J. A.; Yang, Y.; Blom, K.; Reinot, A.; Lie, J.; Cruz-Neira, C.; Wurtele, E. S.: Using virtual reality to understand complex metabolic networks. In: *Atlantic symposium on computational biology and genomic information systems and technology, september* : Citeseer, 2003, pp. 950–953

[Dyck90] Dyckhoff, H.: A typology of cutting and packing problems. In: *European Journal of Operational Research* 44 (1990), no. 2, pp. 145–159

[DyFi92] Dyckhoff, H.; Finke, U.: *Cutting and packing in production and distribution: a typology and bibliography*. Heidelberg, Germany : Physica-Verlag, 1992 — ISBN 3790806307

[DyKr85] Dyckhoff, H.; Kruse, D.: Trim loss and related problems. In: *Omega* 13 (1985), no. 1, pp. 59–72

[Eade84] Eades, P.: A heuristic for graph drawing. In: *Congressus numerantium* 42 (1984), pp. 149–160

[Eade92] Eades, P.: Drawing free trees. In: *Bulletin of the Institute of Combinatorics and its Applications* (1992), no. 5, pp. 10–36

[Fabe05] Faber, C.: *Datenexploration metabolischer Pfade in 3D*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2005

[FaSS02] Faraldo-Gómez, J. D.; Smith, G. R.; Sansom, M. S.: Setting up and optimization of membrane protein simulations. In: *European Biophysics Journal* 31 (2002), no. 3, pp. 217–227

[FCMA04] Feng, Z.; Chen, L.; Maddula, H.; Akcan, O.; Oughtred, R.; Berman, H. M; Westbrook, J.: Ligand Depot: a data warehouse for ligands bound to macromolecules. In: *Bioinformatics* 20 (2004), no. 13, p. 2153

[FeJK08] Feng, Y.; Jernigan, R. L; Kloczkowski, A.: Orientational distributions of contact clusters in proteins closely resemble those of an icosahedron. In: *Proteins: Structure, Function, and Bioinformatics* 73 (2008), no. 3, pp. 730–741

[FHKS08] Fung, D. C.; Hong, S.; Koschützki, D.; Schreiber, F.; Xu, K.: 2.5 D Visualisation of Overlapping Biological Networks. In: *Journal of Integrative Bioinformatics* 5 (2008), no. 1

[FoLa07] Fourmentin, E.; Lariviere, D.: LifeExplorer: a visualization tool for molecular and synthetic biology. In: Amara, P.; Képès, F.; Bernot, G. (eds.): *Proceedings of The Evry Spring School on Modelling Complex BiologicalSystems*, 2007, p. 105

[FrBF74] Franklin, B.; Brownrigg, W.; Farish, M.: Of the stilling of waves by means of oil. Extracted from Sundry Letters between Benjamin Franklin, LL. DFRS William Brownrigg, MDFRS and the Reverend Mr. Farish. In: *Philosophical Transactions (1683-1775)* 64 (1774), pp. 445–460

[FrEd70] Frye, L. D.; Edidin, M.: The rapid intermixing of cell surface antigens after formation of mouse-human heterokaryons. In: *Journal of Cell Science* 7 (1970), no. 2, pp. 319–335

[Fric25] Fricke, H.: The electric capacity of suspensions with special reference to blood. In: *The Journal of General Physiology* 9 (1925), no. 2, pp. 137–152

[FrRe91] Fruchterman, T. M. J.; Reingold, E. M.: Graph drawing by force-directed placement. In: *Software-Practice and Experience* 21 (1991), no. 11, pp. 1129–1164

[Full70] Fuller, R. B.: *The Buckminster Fuller Reader*. London, UK : Jonathan Cape, 1970

[GaCo11] Galperin, M. Y.; Cochrane, G. R.: The 2011 Nucleic Acids Research Database Issue and the Online Molecular Miology Database Collection. In: *Nucleic Acids Research* 39 (2011), no. suppl 1, pp. D1–D6

[GaFe12] Galperin, M. Y.; Fernandez-Suarez, X. M.: The 2012 Nucleic Acids Research Database Issue and the online Molecular Biology Database Collection. In: *Nucleic Acids Research* 40 (2012), no. D1, pp. D1–8. — MEDLINE:22144685

[Gamr09] Gamroth, C.: *Erweiterung des Zellmembraneditors CELLmicrocosmos 2.1 um eine semi-automatische Proteinplatzierung unter Verwendung von PDBTM-Daten. Bachelorarbeit*. Bielefeld, Bielefeld University, Bachelor Thesis, 2009

[Garf68] Garfinkel, D.: A machine-independent language for the simulation of complex

chemical and biochemical systems. In: *Computers and Biomedical Research* 2 (1968), no. 1, pp. 31–44

[Genn89] Gennis, R. B: *Biomembranes: molecular structure and function*, *Springer advanced texts in chemistry*. New York, USA : Springer Verlag, 1989

[GhWT73] Ghosh, D.; Williams, M. A.; Tinoco, J.: The influence of lecithin structure on their monolayer behavior and interactions with cholesterol. In: *Biochimica et Biophysica Acta (BBA)-Biomembranes* 291 (1973), no. 2, pp. 351–362

[GoGr25] Gorter, E.; Grendel, F.: On bimolecular layers of lipoids on the chromocytes of the blood. In: *The Journal of Experimental Medicine* 41 (1925), no. 4, pp. 439–443

[Gree10] Green-Armytage, P.: A colour alphabet and the limits of colour coding. In: *Colour: Design & Creativity* (2010), no. 5, pp. 1–23

[GrGl92] Graner, F.; Glazier, J. A.: Simulation of biological cell sorting using a two-dimensional extended Potts model. In: *Physical Review Letters* 69 (1992), no. 13, pp. 2013–2016

[GrGr96] Grubmüller, H.; Groll, V.: *Solvate - Max-Planck-Institut für biophysikalische Chemie, Göttingen*. URL http://www.mpibpc.mpg.de/home/grubmueller/downloads/solvate/index.html. - retrieved on 08/02/2011

[Grop19] Gropius, W.: Bauhaus-Manifest (1919)

[Groß11] Gross, M.: Unordentliche Proteine. In: *Nachrichten aus der Chemie* 59 (2011), no. 2, pp. 133–134

[GrTz66] Green, D. E.; Tzagoloff, A.: Role of lipids in the structure and function of biological membranes. In: *The Journal of Lipid Research* 7 (1966), no. 5, p. 587

[HAAB09] Hunter, S.; Apweiler, R.; Attwood, T. K.; Bairoch, A.; Bateman, A.; Binns, D.; Bork, P.; Das, U. et al.: InterPro: the integrative protein signature database. In: *Nucleic Acids Research* 37 (2009), no. suppl 1, pp. D211–D215

[HaJü05] Hachul, S.; Jünger, M.: Drawing large graphs with a potential-field-based multilevel algorithm. In: *Graph Drawing* (2005), pp. 285–295

[HBLL97] Hyde, S.; Blum, Z.; Landh, T.; Lidin, S.; Ninham, B. W.; Andersson, S.; Larsson, K.: *The Language of shape: the role of curvature in condensed matter--physics, chemistry, and biology*. Amsterdam, The Netherlands : Elsevier Science Ltd, 1997 — ISBN 0444815384

[HCPP75] Hui, S. W.; Cowden, M.; Papahadjopoulos, D.; Parsons, D. F.: Electron diffraction study of hydrated phospholipid single bilayers. Effects of temperature, hydration and surface pressure of the “precursor” monolayer. In: *Biochimica et Biophysica Acta (BBA)-Biomembranes* 382 (1975), no. 3, pp. 265–275

[Herr07] Herrez, A.: *How to use Jmol to study and present molecular structures*. Morrisville, USA : Lulu Enterprises, 2007 — ISBN 1847992595

[HeUn75] Henderson, R.; Unwin, P. N. T.: Three-dimensional model of purple membrane obtained by electron microscopy. In: *Nature* 257 (1975), no. 5521, pp. 28–32

[HFSB03] Hucka, M.; Finney, A.; Sauro, H. M.; Bolouri, H.; Doyle, J. C.; Kitano, H.; Arkin, A. P.; Bornstein, B. J. et al.: The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. In: *Bioinformatics* 19 (2003), no. 4, pp. 524–531

[HKSL08] Hess, B.; Kutzner, C.; van der Spoel, D.; Lindahl, E.: Gromacs 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation. In: *J. Chem. Theory Comput* 4 (2008), no. 3, pp. 435–447

[HKTJ10] Hippe, K.; Kormeier, B.; Töpel, T.; Janowski, S.; Hofestädt, R.: DAWIS-MD-a data warehouse system for metabolic data. In: *GI Jahrestagung* 2 (2010), pp. 720–

725

[HLNK90] Hovius, R.; Lambrechts, H.; Nicolay, K.; de Kruijff, B.: Improved methods to isolate and subfractionate rat liver mitochondria. Lipid composition of the inner and outer membrane. In: *Biochimica et Biophysica Acta (BBA)-Biomembranes* 1021 (1990), no. 2, pp. 217–226

[Hoff10] Hoffman, C.: *Montréal Biosphère | Flickr - Photo Sharing!* URL http://www.flickr.com/photos/kitonlove/4982068671/in/set-72157624814426759/. - retrieved on 09/27/2012

[HoLZ08] Houidi, I.; Louati, W.; Zeghlache, D.: A distributed virtual network mapping algorithm. In: *ICC'08. IEEE International Conference on Communications, 2008.* : IEEE ICC, 2008 — ISBN 142442075X, pp. 5634–5640

[Hook65] Hooke, R.: *Micrographia, or, Some physiological descriptions of minute bodies made by magnifying glasses, with observations and inquiries thereupon: by Robert Hooke; with a preface by RT Gunther* : Dover Phoenix Editions/The Royal Society, 1665 — ISBN 0486495647

[HoWW07] Ho, E.; Webber, R.; Wilkins, M. R.: Interactive three-dimensional visualization and contextual analysis of protein interaction networks. In: *Journal of Proteome Research* 7 (2007), no. 01, pp. 104–112

[HRKV10] Hall, A.; Róg, T.; Karttunen, M.; Vattulainen, I.: Role of glycolipids in lipid rafts: a view through atomistic molecular dynamics simulations with galactosylceramide. In: *The Journal of Physical Chemistry B* 114 (2010), no. 23, pp. 7797–7807

[HSAB05] Hamosh, A.; Scott, A. F.; Amberger, J. S.; Bocchini, C. A.; McKusick, V. A.: Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. In: *Nucleic Acids Research* 33 (2005), no. suppl 1, pp. D514–D517

[HuDS96] Humphrey, W.; Dalke, A.; Schulten, K.: VMD: Visual Molecular Dynamics. In: *Journal of Molecular Graphics* 14 (1996), no. 1, pp. 33–38

[HuEl99] Hulbert, A. J.; Else, P. L: Membranes as possible pacemakers of metabolism. In: *Journal of Theoretical Biology* 199 (1999), no. 3, pp. 257–274

[HuFe99] Huang, J.; Feigenson, G. W.: A microscopic interaction model of maximum solubility of cholesterol in lipid bilayers. In: *Biophysical Journal* 76 (1999), no. 4, pp. 2142–2157

[HVTS07] Heazlewood, J. L.; Verboom, R. E.; Tonti-Filippini, J.; Small, I.; Millar, A. H.: SUBA: the Arabidopsis subcellular database. In: *Nucleic Acids Research* 35 (2007), no. suppl 1, pp. D213–D218

[HWSB99] Hubona, G. S.; Wheeler, P. N.; Shirah, G. W.; Brandt, M.: The relative contributions of stereo, lighting, and background scenes in promoting 3D depth visualization. In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 6 (1999), no. 3, pp. 214–242

[ICHC04] Izaguirre, J. A.; Chaturvedi, R.; Huang, C.; Cickovski, T.; Coffland, J.; Thomas, G.; Forgacs, G.; Alber, M. et al.: CompuCell, a multi-model framework for simulation of morphogenesis. In: *Bioinformatics* 20 (2004), no. 7, pp. 1129–1137

[IMOT09] Ishiwata, R. R.; Morioka, M. S.; Ogishima, S.; Tanaka, H.: BioCichlid: central dogma-based 3D visualization system of time-course microarray data on a hierarchical biological network. In: *Bioinformatics* 25 (2009), no. 4, pp. 543–544

[IOTI09] Inomata, K.; Ohno, A.; Tochio, H.; Isogai, S.; Tenno, T.; Nakase, I.; Takeuchi, T.; Futaki, S. et al.: High-resolution multi-dimensional NMR spectroscopy of

proteins in human cells. In: *Nature* 458 (2009), no. 7234, pp. 106–109

[IsMN76] Israelachvili, J. N.; Mitchell, D. J.; Ninham, B. W.: Theory of self-assembly of hydrocarbon amphiphiles into micelles and bilayers. In: *Journal of the Chemical Society, Faraday Transactions 2* 72 (1976), pp. 1525–1568

[IzVo06] Izvekov, S.; Voth, G. A.: Multiscale coarse-graining of mixed phospholipid/cholesterol bilayers. In: *Journal of Chemical Theory and Computation* 2 (2006), no. 3, pp. 637–648

[Jain88] Jain, Mahendra Kumar: *Introduction to biological membranes*. 2nd. ed. New York, USA, 1988 — ISBN 0471844713

[JKII08] Jo, S.; Kim, T.; Iyer, V. G.; Im, W.: CHARMM GUI: a web-based graphical user interface for CHARMM. In: *Journal of computational chemistry* 29 (2008), no. 11, pp. 1859–1865

[JKTH10] Janowski, S.; Kormeier, B.; Töpel, T.; Hippe, K.; Hofestädt, R.; Willassen, N.; Friesen, R.; Rubert, S. et al.: Modeling of cell-to-cell communication processes with Petri nets using the example of quorum sensing. In: *In Silico Biology* 10 (2010), no. 1, pp. 27–48

[JLKI09] Jo, S.; Lim, J.; Klauda, J.; Im, W.: CHARMM-GUI Membrane Builder for Mixed Bilayers and Its Application to Yeast Membranes. In: *Biophysical Journal* 97 (2009), no. 1, pp. 50–58

[John11] Johnson, G. T.: *Automated Modeling and Visualization of the Cellular Mesoscale*. La Jolla, USA, The Scribbs Research Institute, Doctorate Thesis, 2011

[JoKI07] Jo, S.; Kim, T.; Im, W.: Automated builder and database of protein/membrane complexes for molecular dynamics simulations. In: *PloS one* 2 (2007), no. 9, p. e880

[JóMa07] Jójárt, B.; Martinek, T. A.: Performance of the general amber force field in modeling aqueous POPC membrane bilayers. In: *Journal of Computational Chemistry* 28 (2007), no. 12, pp. 2051–2058

[JuKS06] Junker, B.; Klukas, C.; Schreiber, F.: VANTED: A system for advanced data analysis and visualization in the context of biological networks. In: *BMC bioinformatics* 7 (2006), no. 1, p. 109

[KaAg10] Kaystha, S.; Agarwal, S.: Greedy genetic algorithm to Bounded Knapsack Problem. In: *3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), 2010*. vol. 6 : IEEE, 2010 — ISBN 1424455375, pp. 301–305

[KaKa89] Kamada, T.; Kawai, S.: An algorithm for drawing general undirected graphs. In: *Information processing letters* 31 (1989), no. 1, pp. 7–15

[Kane12] Kanehisa, M.: *Plea to Support KEGG*. URL http://www.webcitation.org/65fqsRU11. - retrieved on 02/23/2012

[KeJu76] Kelly, K. L.; Judd, D. B.: *Color: universal language and dictionary of names* : US Dept. of Commerce, National Bureau of Standards: for sale by the Supt. of Docs., US Govt. Print. Off., 1976

[Kell65] Kelly, K. L.: Twenty-two colors of maximum contrast. In: *Color Engineering* 3 (1965), no. 6, pp. 26–27

[KePP04] Kellerer, H.; Pferschy, U.; Pisinger, D.: *Knapsack problems*. Berlin, Germany : Springer Verlag, 2004 — ISBN 3540402861

[KGSF12] Kanehisa, M.; Goto, S.; Sato, Y.; Furumichi, M.; Tanabe, M.: KEGG for integration and interpretation of large-scale molecular data sets. In: *Nucleic acids research* 40 (2012), no. D1, pp. D109–D114

[KGUB09] Kučerka, N.; Gallová, J.; Uhríková, D.; Balgavý, P.; Bulacu, M.; Marrink, S. J.; Katsaras, J.: Areas of Monounsaturated Diacylphosphatidylcholines. In:

Biophysical Journal 97 (2009), no. 7, pp. 1926–1932

[KHTH10] Kormeier, B.; Hippe, K.; Töpel, T.; Hofestädt, R.: CardioVINEdb: a data warehouse approach for integration of life science data in cardiovascular diseases. In: *Journal of Integrative Bioinformatics* 7 (2010), no. 1, p. 142

[KiKC08] Kincaid, R.; Kuchinsky, A.; Creech, M.: VistaClara: an expression browser plug-in for Cytoscape. In: *Bioinformatics* 24 (2008), no. 18, pp. 2112–2114

[KKWZ08] Knoops, K.; Kikkert, M.; van den Worm, S. H. E.; Zevenhoven-Dobbe, J. C.; van der Meer, Y.; Koster, A. J.; Mommaas, A. M.; Snijder, E. J.: SARS-coronavirus replication is supported by a reticulovesicular network of modified endoplasmic reticulum. In: *PLoS biology* 6 (2008), no. 9, p. e226

[Klei97] Kleinzeller, A.: Ernest Overton's contribution to the cell membrane concept: a centennial appreciation. In: *News in Physiological Sciences* 12 (1997), no. 1, p. 49

[KlJo98] Kleywegt, G. J.; Jones, T. A.: Databases in protein crystallography. In: *Acta Crystallographica Section D: Biological Crystallography* 54 (1998), no. 6, pp. 1119–1131

[KLRK07] Kim, H.; Ling, S. C.; Rogers, G. C.; Kural, C.; Selvin, P. R.; Rogers, S. L.; Gelfand, V. I.: Microtubule binding by dynactin is required for microtubule organization but not cargo transport. In: *The Journal of Cell Biology* 176 (2007), no. 5, pp. 641–651

[Koho90] Kohonen, T.: The self-organizing map. In: *Proceedings of the IEEE* 78 (1990), no. 9, pp. 1464–1480

[Konr02] Konrad-Zuse-Zentrum für Informationstechnik: amira user's guide: amira 3.0, Konrad-Zuse-Zentrum für Informationstechnik (2002)

[Korm10] Kormeier, B.: *Semi-automated reconstruction of biological networks based on a life science data warehouse*. Bielefeld, Germany, Bielefeld University, 2010

[KrFi08] Krüger, J.; Fischer, W. B.: Exploring the conformational space of Vpu from HIV-1: a versatile adaptable protein. In: *Journal of Computational Chemistry* 29 (2008), no. 14, pp. 2416–2424

[Krot97] Kroto, H.: Symmetry, space, stars, and C60 (Nobel lecture). In: *Angewandte Chemie International Edition in English* 36 (1997), no. 15, pp. 1578–1593

[KRPP02] Karp, P. D.; Riley, M.; Paley, S. M.; Pellegrini-Toole, A.: The Metacyc Database. In: *Nucleic Acids Research* 30 (2002), no. 1, pp. 59–61

[LaFo08] Larivière, D.; Fourmentin, E.: 3D realistic visualization of supramolecular assemblies. In: *EMC 2008 14th European Microscopy Congress 1–5 September 2008, Aachen, Germany* : Springer, 2008, pp. 175–176

[LeHe92] Levkowitz, H.; Herman, G. T.: Color scales for image data. In: *Computer Graphics and Applications, IEEE* 12 (1992), no. 1, pp. 72–80

[LeSa68] Levy, M.; Sauner, M. T.: Specificite de composition en phospholipides et en cholesterol des membranes mitochondriales. In: *Chemistry and Physics of Lipids* 2 (1968), no. 3, pp. 291–295

[Levk97] Levkowitz, H.: *Color theory and modeling for computer graphics, visualization, and multimedia applications*. Norwell, U.S.A. : Kluwer Academic Publishers, 1997 — ISBN 0792399285

[LFGC07] Larson, S. D.; Fong, L. L.; Gupta, A.; Condit, C.; Bug, W. J.; Martone, M. E.: A formal ontology of subcellular neuroanatomy. In: *Frontiers in Neuroinformatics* 1 (2007), no. 0

[LGAP07] Lartigue, C.; Glass, J. I.; Alperovich, N.; Pieper, R.; Parmar, P. P.; Hutchison III, C. A.; Smith, H. O.; Venter, J. C.: Genome transplantation in bacteria: changing one species to another. In: *Science* 317 (2007), no. 5838, pp. 632–638

[Lipa07] Lipari, N. G.: Evaluation of stereoscopy and lit shading for a counting task in knot visualization. In: *The 2007 International Conference on Computer Graphics and Virtual Reality* : Citeseer, 2007

[LiSi10] Lingwood, D.; Simons, K.: Lipid rafts as a membrane-organizing principle. In: *Science* 327 (2010), no. 5961, pp. 46–50

[LLPM06] Lomize, M. A.; Lomize, A. L.; Pogozheva, I. D.; Mosberg, H. I.: OPM: orientations of proteins in membranes database. In: *Bioinformatics* 22 (2006), no. 5, pp. 623–625

[Lok11] Lok, Corie: Biomedical illustration: From monsters to molecules. In: *Nature* 477 (2011), no. 7364, pp. 359–361

[LoMM02] Lodi, A.; Martello, S.; Monaci, M.: Two-dimensional packing problems: A survey. In: *European Journal of Operational Research* 141 (2002), no. 2, pp. 241–252

[LoMü01] Louis, D.; Müller, P.: *Jetzt lerne ich Java*. Munich, Germany : Markt+Technik Verlag/Pearson Education Deutschland GmbH, 2001 — ISBN 3827263697

[LoPM11] Lomize, A. L.; Pogozheva, I. D.; Mosberg, H. I.: Anisotropic solvent model of the lipid bilayer. 2. Energetics of insertion of small molecules, peptides, and proteins in membranes. In: *Journal of Chemical Information and Modeling* 51 (2011), no. 4, pp. 930–946

[Lord90] Lord Rayleigh: Measurements of the Amount of Oil Necessary in Order to Check the Motions of Camphor upon Water. In: *Proceedings of the Royal Society* (1890), no. 47, pp. 364–367

[LoSc01] Loew, L. M.; Schaff, J. C.: The Virtual Cell: a software environment for computational cell biology. In: *TRENDS in Biotechnology* 19 (2001), no. 10, pp. 401–406

[LóSt07] López, M.; Still, G.: Semi-infinite programming. In: *European Journal of Operational Research* 180 (2007), no. 2, pp. 491–518

[LPJM12] Lomize, M. A.; Pogozheva, I. D.; Joo, H.; Mosberg, H. I.; Lomize, A. L.: OPM database and PPM web server: resources for positioning of proteins in membranes. In: *Nucleic Acids Research* 40 (2012), no. D1, pp. D370–D376

[LPLM06] Lomize, A. L.; Pogozheva, I. D.; Lomize, M. A.; Mosberg, H. I.: Positioning of proteins in membranes: a computational approach. In: *Protein Science* 15 (2006), no. 6, pp. 1318–1333

[LRFW05] Latorre, I. J.; Roh, M. H.; Frese, K. K.; Weiss, R. S.; Margolis, B.; Javier, R. T.: Viral oncoprotein-induced mislocalization of select PDZ proteins disrupts tight junctions and causes polarity defects in epithelial cells. In: *Journal of Cell Science* 118 (2005), no. 18, pp. 4283–4293

[Luck08] Luckey, M.: *Membrane structural biology: with biochemical and biophysical foundations*. New York, USA : Cambridge University Press, 2008 — ISBN 0521856558

[Luka12] Lukat, G.: *Entwurf eines Programms zur Voronoi-Diagramm-unterstutzten Analyse von Membransimulationen*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2012

[LuVo09] Lu, L.; Voth, G. A.: Systematic coarse-graining of a multicomponent lipid bilayer. In: *The Journal of Physical Chemistry B* 113 (2009), no. 5, pp. 1501–1510

[MABM09] Martínez, L.; Andrade, R.; Birgin, E. G.; Martínez, J. M.: Packmol: A package for building initial configurations for molecular dynamics simulations. In: *Journal of computational chemistry* 30 (2009), no. 13, pp. 2157–2164

[MaMa03] Martínez, J. M.; Martínez, L.: Packing optimization for automated generation of complex system's initial configurations for molecular dynamics and docking.

In: *Journal of Computational Chemistry* 24 (2003), no. 7, pp. 819–825

[Mars96] Marsh, D.: Intrinsic curvature in normal and inverted lipid structures and in membranes. In: *Biophysical journal* 70 (1996), no. 5, pp. 2248–2255

[MaRu10] Mavelli, F.; Ruiz-Mirazo, K.: ENVIRONMENT: a computational platform to stochastically simulate reacting and self-reproducing lipid compartments. In: *Physical Biology* 7 (2010), no. 3, p. 036002

[MaTo90] Martello, S.; Toth, P.: *Knapsack problems: algorithms and computer implementations* : John Wiley & Sons, Inc., 1990 — ISBN 0471924202

[McNW06] McBride, H. M.; Neuspiel, M.; Wasiak, S.: Mitochondria: more than just a powerhouse. In: *Current Biology* 16 (2006), no. 14, pp. R551–R560

[MCNW09] McComb, T.; Cairncross, O.; Noske, A. B.; Wood, D. L. A.; Marsh, B. J.; Ragan, M. A.: Illoura™: a software tool for analysis, visualization and semantic querying of cellular and other spatial biological data. In: *Bioinformatics* 25 (2009), no. 9, pp. 1208–1210

[MeGl05] Merks, R. M. H.; Glazier, J. A.: A cell-centered approach to developmental biology. In: *Physica A: Statistical Mechanics and its Applications* 352 (2005), no. 1, pp. 113–130

[Mend93] Mendes, P.: GEPASI: a software package for modelling the dynamics, steady states and control of biochemical and other systems. In: *Computer applications in the biosciences: CABIOS* 9 (1993), no. 5, pp. 563–571

[Meye98] Meyer, B.: Self-organizing graphs—a neural network perspective of graph layout. In: *Graph Drawing* : Springer, 1998, pp. 246–262

[Meye99] Meyer, H.: Zur Theorie der Alkoholnarkose. In: *Archiv für experimentelle Pathologie und Pharmakologie* 42 (1899), pp. 109–118

[MGGC09] Matthews, L.; Gopinath, G.; Gillespie, M.; Caudy, M.; Croft, D.; De Bono, B.; Garapati, P.; Hemish, J. et al.: Reactome knowledgebase of human biological pathways and processes. In: *Nucleic Acids Research* 37 (2009), no. suppl 1, pp. D619–D622

[MGWQ02] Martone, M. E.; Gupta, A.; Wong, M.; Qian, X.; Sosinsky, G.; Ludäscher, B.; Ellisman, M. H.: A cell-centered database for electron tomographic data. In: *Journal of Structural Biology* 138 (2002), no. 1-2, pp. 145–155

[Mich12] Michal, G.: *ExPASy - Biochemical Pathways*. URL http://www.webcitation.org/query?url=http%3A%2F%2Fweb.expasy.org%2Fcgi-bin%2Fpathways%2Fshow_thumbnails.pl&date=2012-03-18. - retrieved on 03/18/2012

[Mpi11] MPI: *Research with Photons at the Storage Ring DORIS III*. URL http://hasylab.desy.de/e81/e5400/e41243/e5403/e57510/index_eng.html. - retrieved on 01/04/2012

[MRTW62] Mueller, P.; Rudin, D. O.; Ti Tien, H.; Wescott, W. C.: Reconstitution of cell membrane structure in vitro and its transformation into an excitable system. In: *Nature* 194 (1962), pp. 979–980

[MSSL02] Moraru, I. I.; Schaff, J. C.; Slepchenko, B. M.; Loew, L. M.: The Virtual Cell: an integrated modeling environment for experimental and computational cell biology. In: *Annals of the New York Academy of Sciences* 971 (2002), pp. 595–596

[MYYN04] Ma, J.; Yoshimura, M.; Yamashita, E.; Nakagawa, A.; Ito, A.; Tsukihara, T.: Structure of rat monoamine oxidase A and its specific recognitions for substrates and inhibitors. In: *Journal of Molecular Biology* 338 (2004), no. 1, pp. 103–114

[NaNe06] Nance, Jasper; nebarnix: *phase contrast blood | Flickr - Fotosharing!* URL http://www.flickr.com/photos/nebarnix/303799491/. - retrieved on

01/03/2012

[Nath04] Nathansohn, A.: Über die Regulation der Aufnahme anorganischer Salze durch die Knollen von Dahlia. In: *Jahrbücher für wissenschaftliche Botanik*. vol. 39, 1904, pp. 607–644

[NaTr00] Nagle, J. F.; Tristram-Nagle, S.: Structure of lipid bilayers. In: *Biochimica et Biophysica Acta (BBA)-Reviews on Biomembranes* 1469 (2000), no. 3, pp. 159–195

[Natt04] Nattkemper, T. W: Multivariate image analysis in biomedicine. In: *Journal of Biomedical Informatics* 37 (2004), no. 5, pp. 380–391

[Nauj95] Naujoks, G.: *Optimale Stauraumnutzung*. Wiesbaden, Germany : Gabler Verlag, 1995 — ISBN 3824461668

[NeCL08] Nelson, D. L.; Cox, M. M.; Lehninger, A. L.: *Principles of Biochemistry* : Freeman, 2008 — ISBN 1429208929

[NHMM09] Le Novère, N.; Hucka, M.; Mi, H.; Moodie, S.; Schreiber, F.; Sorokin, A.; Demir, E.; Wegner, K. et al.: The systems biology graphical notation. In: *Nature Biotechnology* 27 (2009), no. 8, pp. 735–741

[NoCa97] North, C.; Cafiso, D. S.: Contrasting membrane localization and behavior of halogenated cyclobutanes that follow or violate the Meyer-Overton hypothesis of general anesthetic potency. In: *Biophysical Journal* 72 (1997), no. 4, pp. 1754–1761

[Olea87] O'Leary, T. J.: Lateral diffusion of lipids in complex biological membranes. In: *Proceedings of the National Academy of Sciences of the United States of America* 84 (1987), no. 2, pp. 429–433

[Orli12] Orlik, R. E.: *Analyse wirtschaftlicher Potentiale und Entwicklung von Geschäftsmodellen für Open-Source-Projekte anhand der E-Learning-Software CELLmicrocosmos CE³*. Bielefeld, Germany, Bielefeld University, Master Thesis, 2012

[OrOv11] Orosz, F.; Ovádi, J.: Proteins without 3D structure: definition, detection and beyond. In: *Bioinformatics* 27 (2011), no. 11, pp. 1449–1454

[OuKZ98] Outrata, J. V.; Kočvara, M.; Zowe, J.: *Nonsmooth approach to optimization problems with equilibrium constraints: Theory, applications, and numerical results*. New York, USA : Springer, 1998 — ISBN 0792351703

[Over99] Overton, E.: Über die allgemeinen osmotischen Eigenschaften der Zelle, ihre vermutlichen Ursachen u. ihre Bedeutg f. d. Physiologie. In: *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich* 44 (1899), pp. 88–135

[Özgü10] Özgür, A.: *Implementierung von SBML-Import/Export in den CELLmicrocosmos Cell Explorer unter Verwendung von metabolischen Netzwerken und 3D-Daten*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2010

[PaCB51] Pauling, L.; Corey, R. B.; Branson, H. R.: The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain. In: *Proceedings of the National Academy of Sciences of the United States of America* 37 (1951), no. 4, pp. 205–211

[PaMa05] Pan, L.; Martín-Vide, C.: Solving multidimensional 0-1 knapsack problem by P systems with input and active membranes. In: *Journal of Parallel and Distributed Computing* 65 (2005), no. 12, pp. 1578–1584

[PaVK91] Pastor, R. W.; Venable, R. M.; Karplus, M.: Model for the structure of the lipid bilayer. In: *Proceedings of the National Academy of Sciences* 88 (1991), no. 3, pp. 892–896

[PeBe83] Peters, R.; Beck, K.: Translational diffusion in phospholipid monolayers measured by fluorescence microphotolysis. In: *Proceedings of the National*

Academy of Sciences of the United States of America 80 (1983), no. 23, pp. 7183–7187

[PESM95] Peitzsch, R. M.; Eisenberg, M.; Sharp, K. A.; McLaughlin, S.: Calculations of the electrostatic potential adjacent to model phospholipid bilayers. In: *Biophysical Journal* 68 (1995), no. 3, pp. 729–738

[Pfef77] Pfeffer, W.: *Osmotische Untersuchungen: Studien zur Zellmechanik* : W. Engelmann, Leipzig, 1877

[Pfei11] Pfeiffer, Michael: *Java 3D 1.3 für Einsteiger und Fortgeschrittene, Version 1.1*, 2011

[PGHC04] Pettersen, E. F.; Goddard, T. D.; Huang, C. C.; Couch, G. S.; Greenblatt, D. M.; Meng, E. C.; Ferrin, T. E.: UCSF Chimera—a visualization system for exploratory research and analysis. In: *Journal of Computational Chemistry* 25 (2004), no. 13, pp. 1605–1612

[PGME00] Petrache, H. I.; Grossfield, A.; MacKenzie, K. R.; Engelman, D. M.; Woolf, T. B.: Modulation of glycophorin A transmembrane helix interactions by lipid bilayers: molecular dynamics calculations1. In: *Journal of Molecular Biology* 302 (2000), no. 3, pp. 727–746

[Pike06] Pike, L. J.: Rafts defined: a report on the Keystone Symposium on Lipid Rafts and Cell Function. In: *Journal of Lipid Research* 47 (2006), no. 7, p. 1597

[PKFS00] Pralle, A.; Keller, P.; Florin, E. L; Simons, K.; Horber, J. K. H.: Sphingolipid-cholesterol rafts diffuse as small entities in the plasma membrane of mammalian cells. In: *Science's STKE* 148 (2000), no. 5, pp. 997–1008

[PMPH03] Prior, I. A; Muncke, C.; Parton, R. G; Hancock, J. F: Direct visualization of Ras proteins in spatially distinct cell surface microdomains. In: *Science's STKE* 160 (2003), no. 2, pp. 165–170

[POSS08] Pavlopoulos, G. A.; O'Donoghue, S. I.; Satagopam, V. P.; Soldatos, T. G.; Pafilis, E.; Schneider, R.: Arena 3D: visualization of biological networks in 3D. In: *BMC Systems Biology* 2 (2008), no. 1, p. 104

[PüMW11] Pühler, A.; Müller-Röber, B.; Weitze, M. D: Stellungnahme von DFG, acatech und Leopoldina (Zusammenfassung). In: *Synthetische Biologie* (2011), pp. 151–155

[PuWe09] Purnick, P. E. M.; Weiss, R.: The second wave of synthetic biology: from modules to systems. In: *Nature Reviews Molecular Cell Biology* 10 (2009), no. 6, pp. 410–422

[PVCJ04] Pandit, S. A.; Vasudevan, S.; Chiu, S. W.; Jay Mashl, R.; Jakobsson, E.; Scott, H. L.: Sphingomyelin-cholesterol domains in phospholipid membranes: atomistic simulation. In: *Biophysical journal* 87 (2004), no. 2, pp. 1092–1100

[PYDK11] Podkolodnaya, O. A.; Yarkova, E. E.; Demenkov, P. S.; Konovalova, O. S.; Ivanisenko, V. A.; Kolchanov, N. A.: Application of the ANDCell computer system to reconstruction and analysis of associative networks describing potential relationships between myopia and glaucoma. In: *Russian Journal of Genetics: Applied Research* 1 (2011), no. 1, pp. 21–28

[PZKZ09] Potamitis, C.; Zervou, M.; Katsiaras, V.; Zoumpoulakis, P.; Durdagi, S.; Papadopoulos, M. G.; Hayes, J. M.; Grdadolnik, S. G. et al.: Antihypertensive Drug Valsartan in Solution and at the AT1 Receptor: Conformational Analysis, Dynamic NMR Spectroscopy, in Silico Docking, and Molecular Dynamics Simulations. In: *Journal of Chemical Information and Modeling* 49 (2009), no. 3, pp. 726–739

[Quin88] Quincke, G.: Ueber periodische Ausbreitung an Flüssigkeitsoberflächen und dadurch hervorgerufene Bewegungserscheinungen. In: *Annalen der Physik* 271 (1888), no. 12, pp. 580–642

[RaSZ94] Rakhmanov, E. A.; Saff, E. B.; Zhou, Y. M.: Minimal discrete energy on the sphere. In: *Mathematical Research Letters* 1 (1994), no. 6, pp. 647–662

[RBBC03] Rhee, S. Y.; Beavis, W.; Berardini, T. Z.; Chen, G.; Dixon, D.; Doyle, A.; Garcia-Hernandez, M.; Huala, E. et al.: The Arabidopsis Information Resource (TAIR): a model organism database providing a centralized, curated gateway to Arabidopsis biology, research materials and community. In: *Nucleic Acids Research* 31 (2003), no. 1, pp. 224–228

[Read03] Read, A. W.: *The new international Webster's comprehensive dictionary of the English language*. 2003. ed. Köln, Germany : Typhoon International/Karl Müller Verlag, 2003 — ISBN 3898939790

[Rein11] Reinot, Andres:URL http://www.reinot.com/gallery.html. - retrieved on 01/01/2012. — Reinot.com

[RGKS12] Rubert, S.; Gamroth, C.; Krüger, J.; Sommer, B.: Grid Workflow Approach using the CELLmicrocosmos 2.2 MembraneEditor and UNICORE to commit and monitor GROMACS Jobs. In: *Proceedings of the Grid Workflow Workshop (GWW2011), Cologne, Germany, March 4, 2011.*, *CEUR-WS*. 826 (2012)

[RiBM05] Rich, A. N.; Bradshaw, J. L.; Mattingley, J. B.: A systematic, large-scale study of synaesthesia: implications for the role of early experience in lexical-colour associations. In: *Cognition* 98 (2005), no. 1, pp. 53–84

[Robe04] Robertson, M.: Reactome: clear view of a starry sky. In: *Drug Discovery Today* 9 (2004), no. 16, pp. 684–685

[Robe59] Robertson, J. D.: The ultrastructure of cell membranes and their derivatives. In: *Biochemical Society Symposium*. vol. 16, 1959 — ISBN 0067-8694, pp. 3–43

[RoFl97] Robinson, A. J.; Flores, T. P.: Novel techniques for visualising biological information. In: *Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology (ISMB-97)*. vol. 5, 1997, pp. 241–249

[Rojd03a] Rojdestvenski, I.: Metabolic pathways in three dimensions. In: *Bioinformatics* 19 (2003), no. 18, pp. 2436–2441

[Rojd03b] Rojdestvenski, I.: VRML metabolic network visualizer. In: *Computers in Biology and Medicine* 33 (2003), no. 2, pp. 169–182

[RoKS09] Rohn, H.; Klukas, C.; Schreiber, F.: Integration and Visualisation of Multimodal Biological Data. In: *Proceeding of German Conference on Bioinformatics*, 2009, pp. 105–115

[RoKS11] Rohn, H.; Klukas, C.; Schreiber, F.: Creating views on integrated multidomain data. In: *Bioinformatics* 27 (2011), no. 13, pp. 1839–1845

[Rube08] Rubert, S.: *Recherche, Berechnung & Visualisierung der Lipid-/Proteinrelationen in Mitochondrienmembranen mit dem für diese Zwecke zu erweiternden CELLmicrocosmos 2.1 Membrane-Editor. Bachelorarbeit*. Bielefeld, Germany, Bielefeld University, Bachelor Thesis, 2008

[Rube11] Rubert, S.: *Entwicklung eines CELLmicrocosmos 2.2 MembraneEditor Plugins zur Verwaltung von molekulardynamischen GROMACS-Simulationen in Cluster-Umgebungen*. Bielefeld, Germany, Bielefeld University, Master Thesis, 2011

[SaKu97] Saff, E. B.; Kuijlaars, A. B. J.: Distributing many points on a sphere. In: *The Mathematical Intelligencer* 19 (1997), no. 1, pp. 5–11

[SBEG10] Seebacher, F.; Brand, M. D.; Else, P. L.; Guderley, H.; Hulbert, A. J.; Moyes, C. D.: Plasticity of Oxidative Metabolism in Variable Climates: Molecular Mechanisms*. In: *Physiological and biochemical zoology: PBZ* (2010)

[SCCG11] Schreiber, F.; Colmsee, C.; Czauderna, T.; Grafahrend-Belau, E.; Hartmann, A.; Junker, A.; Junker, B. H.; Klapperstück, M. et al.: MetaCrop 2.0: managing and exploring information about crop plant metabolism. In: *Nucleic Acids Research* 1 (2011),

pp. 1–5

[ScHa93] Schlame, M.; Haldar, D.: Cardiolipin is synthesized on the matrix side of the inner membrane in rat liver mitochondria. In: *Journal of Biological Chemistry* 268 (1993), no. 1, pp. 74–79

[Schn08] Schneider, S.: *Entwicklung einer Schnittstelle zur Integration membranberechnender Algorithmen in den CELLmicrocosmos 2.1. Diplomarbeit*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2008

[ScML04] Schroeder, W.; Martin, K.; Lorensen, B.: *The Visualization Toolkit: an object oriented approach to 3D graphics* : Kitware Inc, 2004 — ISBN 1930934122

[SDGS11] Sommer, B.; Dingersen, T.; Gamroth, C.; Schneider, S. E.; Rubert, S.; Krüger, J.; Dietz, K. J.: CELLmicrocosmos 2.2 MembraneEditor: a modular interactive shape-based software approach to solve heterogeneous Membrane Packing Problems. In: *Journal of Chemical Information and Modeling* 5 (2011), no. 51, pp. 1165–1182

[SGCS11] Scheer, M.; Grote, A.; Chang, A.; Schomburg, I.; Munaretto, C.; Rother, M.; Söhngen, C.; Stelzer, M. et al.: BRENDA, the enzyme information system in 2011. In: *Nucleic Acids Research* 39 (2011), no. suppl 1, pp. D670–D676

[ShBS97] Shen, L.; Bassolino, D.; Stouch, T.: Transmembrane helix structure, dynamics, and interactions: multi-nanosecond molecular dynamics simulations. In: *Biophysical Journal* 73 (1997), no. 1, pp. 3–20

[ShDK08] Shinoda, W.; DeVane, R.; Klein, M. L.: Coarse-grained molecular modeling of non-ionic surfactant self-assembly. In: *Soft Matter* 4 (2008), no. 12, pp. 2454–2462

[SHFP00] Schütz, G. J.; Hesse, J.; Freudenthaler, G.; Pastushenkoo, V. P.; Knaus, H. G.; Pragl, B.; Schindler, H.: 3D mapping of individual ion channels on living cells. In: *Single Molecules* 1 (2000), no. 2, pp. 153–157

[Ship73] Shipley, G. G.: Recent X-ray diffraction studies of biological membranes and membrane components. In: Chapman, D.; Wallach, D. F. H. (eds.): *Biological Membranes*. New York, USA : Academic Press, 1973, pp. 1–86

[SHLV06] Sickmeier, M.; Hamilton, J. A.; LeGall, T.; Vacic, V.; Cortese, M. S.; Tantos, A.; Szabo, B.; Tompa, P. et al.: DisProt: the database of disordered proteins. In: *Nucleic Acids Research* 35 (2006), no. suppl 1, pp. D786–D793

[SiIk97] Simons, K.; Ikonen, E.: Functional rafts in cell membranes. In: *Nature* 387 (1997), no. 6633, pp. 569–572

[SiNi72] Singer, S. J.; Nicolson, G. L.: The fluid mosaic model of the structure of cell membranes. In: *Science* 175 (1972), no. 23, pp. 720–731

[SKDA13] Sommer, B.; Kormeier, B.; Demenkov, P. S.; Arrigo, P.; Hippe, K.; Ates, Ö.; Kochetov, A. V.; Ivanisenko, V.A.; et al.: Subcellular Localization Charts: a new visual Methodology for the semi-automatic Localization of protein-related Data Sets. In: *Journal of Bioinformartics and Computational Biology* 11 (2013), no. 1, forthcoming

[SKSH10] Sommer, B.; Künsemöller, J.; Sand, N.; Husemann, A.; Rumming, M.; Kormeier, B.: CELLmicrocosmos 4.1: an interactive approach to integrating spatially localized metabolic networks into a virtual 3D cell environment. In: Fred, A.; Filipe, J.; Gamboa, H. (eds.): *BIOINFORMATICS 2010 - Proceedings of the 1st International Conference on Bioinformatics, part of the 3rd International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2010)*, 2010 — ISBN 9789896740191, pp. 90–95

[Smit11] Smith, E.: *Creating a stereoscopic camera rig in 3ds Max*. YouTube.com, 2011

[SMOB03] Shannon, P.; Markiel, A.; Ozier, O.; Baliga, N. S.; Wang, J. T.; Ramage, D.; Amin, N.;

Schwikowski, B. et al.: Cytoscape: a software environment for integrated models of biomolecular interaction networks. In: *Genome research* 13 (2003), no. 11, pp. 2498–2504

[Somm04] Sommer, B.: *CELLmicrocosmos - Zellenvisualisierung. Bachelorarbeit.* Bielefeld, Germany, Bielefeld University, Bachelor Thesis, 2004

[Somm06] Sommer, B.: *CELLmicrocosmos - Interaktive 3D-ZellVisualisierung. Masterarbeit.* Bielefeld, Germany, Bielefeld University, Master Thesis, 2006

[Somm09] Sommer, B.: Cover. In: *Informatik-Spektrum* 32 (2009), no. 4, p. 0

[Somm13] Sommer, B.: Membrane Packing Problems – A short Review on computational Membrane Modeling Methods and Tools. *Computational and Structural Biotechnology Journal* 5 (2013), forthcoming.

[SoZh12] Sommer, B.; Zhou, Y.: The Vesicle Builder - A Plugin for the CELLmicrocosmos MembraneEditor.

[SPAS12] Secrier, M.; Pavlopoulos, G. A.; Aerts, J.; Schneider, R.: Arena3D: visualizing time-driven phenotypic differences in biological systems. In: *BMC Bioinformatics* 13 (2012), no. 1, p. 45

[Spec11] Specious Reasons: *Kale leaf brightfield 100x | Flickr - Fotosharing!* URL http://www.flickr.com/photos/28594931@N03/4249744935/. - retrieved on 01/03/2012

[SSIH09] Sakakibara, D.; Sasaki, A.; Ikeya, T.; Hamatsu, J.; Hanashima, T.; Mishima, M.; Yoshimasu, M.; Hayashi, N. et al.: Protein structure determination in living cells by in-cell NMR spectroscopy. In: *Nature* 458 (2009), no. 7234, pp. 102–105

[SSRB01] Shelley, J. C.; Shelley, M. Y.; Reeder, R. C.; Bandyopadhyay, S.; Moore, P. B.; Klein, M. L.: Simulations of phospholipids using a coarse grain model. In: *J. Phys. Chem. B* 105 (2001), no. 40, pp. 9785–9792

[STKH10] Sommer, B.; Tiys, E. S; Kormeier, B.; Hippe, K.; Janowski, S. J; Ivanisenko, T. V; Bragin, A. O; Arrigo, P. et al.: Visualization and analysis of a Cardio Vascular Disease-and MUPP1-related biological network combining text mining and data warehouse approaches. In: *Journal of Integrative Bioinformatics* 7 (2010), no. 1, p. 148

[STKI05] Sugimoto, M.; Takahashi, K.; Kitayama, T.; Ito, D.; Tomita, M.: Distributed cell biology simulations with E-Cell system. In: *Grid Computing in Life Science* (2005), pp. 20–31

[SVTB10] Schneider, J.; Vorhölter, F. J.; Trost, E.; Blom, J.; Musa, Y. R.; Neuweger, H.; Niehaus, K.; Schatschneider, S. et al.: CARMEN-comparative analysis and in silico reconstruction of organism-specific metabolic networks. In: *Genetics and Molecular Research* 9 (2010), no. 1, pp. 1660–1672

[TaCM06] Tarini, M.; Cignoni, P.; Montani, C.: Ambient occlusion and edge cueing to enhance real time molecular visualization. In: *IEEE Transactions on Visualization and Computer Graphics*, 2006, pp. 1237–1244

[TAYT95] Tsukihara, T.; Aoyama, H.; Yamashita, E.; Tomizaki, T.; Yamaguchi, H.; Shinzawa-Itoh, K.; Nakashima, R.; Yaono, R. et al.: Structures of metal sites of oxidized bovine heart cytochrome c oxidase at 2.8 A. In: *Science* 269 (1995), no. 5227, pp. 1069–1074

[TFOS09] Tompa, P.; Fuxreiter, M.; Oldfield, C. J.; Simon, I.; Dunker, A. K.; Uversky, V. N.: Close encounters of the third kind: disordered domains and the interactions of proteins. In: *Bioessays* 31 (2009), no. 3, pp. 328–335

[Thej10] TheJCB: *Intracellular trafficking | Flickr - Fotosharing!* URL http://www.flickr.com/photos/thejcb/4077864129/. - retrieved on 01/03/2012

[THTS99] Tomita, M.; Hashimoto, K.; Takahashi, K.; Shimizu, T. S.; Matsuzaki, Y.; Miyoshi, F.; Saito, K.; Tanida, S. et al.: E-CELL: software environment for whole-cell simulation. In: *Bioinformatics* 15 (1999), no. 1, pp. 72–84

[TiBe98] Tieleman, D. P.; Berendsen, H. J. C.: A molecular dynamics study of the pores formed by Escherichia coli OmpF porin in a fully hydrated palmitoyloleoylphosphatidylcholine bilayer. In: *Biophysical journal* 74 (1998), no. 6, pp. 2786–2801

[TiCD66] Tien, H. T.; Carbone, S.; Dawidowicz, E. A.: Formation of "Black" Lipid Membranes by oxidation products of cholesterol. In: *Nature* 212 (1966), pp. 718–719

[TiDi67] Tien, H. T; Diana, A. L: Some physical properties of bimolecular lipid membranes produced from new lipid solutions 215 (1967), pp. 1199–1200

[TISS03] Takahashi, K.; Ishikawa, N.; Sadamoto, Y.; Sasamoto, H.; Ohta, S.; Shiozawa, A.; Miyoshi, F.; Naito, Y. et al.: E-Cell 2: multi-platform E-Cell simulation system. In: *Bioinformatics* 19 (2003), no. 13, p. 1727

[TKKH08] Töpel, T.; Kormeier, B.; Klassen, A.; Hofestädt, R.: BioDWH: a data warehouse kit for life science data integration. In: *Journal of Integrative Bioinformatics* 5 (2008), no. 2, p. 93

[Träu71] Träuble, H.: The movement of molecules across lipid membranes: a molecular theory. In: *Journal of Membrane Biology* 4 (1971), no. 1, pp. 193–208

[Trau75] Traube, Moritz: Experimente zur physikalischen Erklärung der Bildung der Zellhaut, ihres Wachsthums durch Intussusception und des Aufwärtswachsens der Pflanzen, Teil 2/2. In: *Botanische Zeitung* 33 (1875), no. 5, pp. 65–70

[TsAW08] Tsirlin, I.; Allison, R. S.; Wilcox, L. M.: Stereoscopic transparency: constraints on the perception of multiple surfaces. In: *Journal of Vision* 8 (2008), no. 5

[TuDS05a] Tusnády, G. E.; Dosztányi, Z.; Simon, I.: PDB_TM: selection and membrane localization of transmembrane proteins in the protein data bank. In: *Nucleic Acids Research* 33 (2005), no. Database Issue, pp. D275–D278

[TuDS05b] Tusnády, G. E.; Dosztányi, Z.; Simon, I.: TMDET: web server for detecting transmembrane regions of proteins by using their 3D coordinates. In: *Bioinformatics* 21 (2005), no. 7, p. 1276

[UdKo82] Ude, J.; Koch, M.: *Die Zelle - Atlas der Ultrastruktur*. Stuttgart, Germany : Fischer, 1982

[Umhe11] UMHealthSystem: *Hodgkin's disease | Flickr - Fotosharing!* URL http://www.flickr.com/photos/umhealthsystem/5395779805/. - retrieved on 01/03/2012

[Unip10] The UniProt Consortium: The Universal Protein Resource (UniProt) in 2010. In: *Nucleic Acids Research* 38 (2010), no. suppl 1, pp. D142–D148

[Unru12] Unruh, P.: *Membrane-Mapping und Optimierung der molekularen Darstellung mit Java 3D in den CELLmicrocosmos Anwendungen*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2012

[USFL98] Ullmer, C.; Schmuck, K.; Figge, A.; Lübbert, H.: Cloning and characterization of MUPP1, a novel PDZ domain protein. In: *FEBS letters* 424 (1998), no. 1-2, pp. 63–68

[Virc58] Virchow, R.: *Die Cellularpathologie in ihrer Begründung auf physiologische und pathologische Gewebelehre*. Berlin, Germany : Verlag August Hirschwald, 1858

[VoVP06] Voet, D.; Voet, J. G.; Pratt, C. W.: *Fundamentals of biochemistry: life at the molecular level*. New York, USA : Wiley, 2006 — ISBN 0471214957

[VZHP93] Venable, R. M.; Zhang, Y.; Hardy, B. J.; Pastor, R. W.: Molecular dynamics simulations of a lipid bilayer and of hexadecane: an investigation of

membrane fluidity. In: *Science* 262 (1993), no. 5131, pp. 223–226

[WaCa11] Wanner, Gerhard; Carl Zeiss Microscopy: *Baker's yeast (Saccharomyces cerevisiae) 4 | Flickr - Photo Sharing!* URL http://www.flickr.com/photos/zeissmicro/7361893014. - retrieved on 08/26/2012

[WaFr96] Ware, C.; Franck, G.: Evaluating stereo and motion cues for visualizing information nets in three dimensions. In: *ACM Transactions on Graphics (TOG)* 15 (1996), no. 2, pp. 121–140

[WaMi05] Ware, C.; Mitchell, P.: Reevaluating stereo and motion cues for visualizing graphs in three dimensions. In: *Proceedings of the 2nd Symposium on Applied Perception in Graphics and Visualization* : ACM, 2005 — ISBN 1595931392, pp. 51–58

[Wang11] Wang, Q.: 3DScape: three dimensional visualization plug-in for Cytoscape. In: *Nature Precedings* (2011)

[WaRM95] Wann, J. P.; Rushton, S.; Mon-Williams, M.: Natural problems for stereoscopic depth perception in virtual environments. In: *Vision Research* 35 (1995), no. 19, pp. 2731–2736

[WaSt09] Wassall, S. R.; Stillwell, W.: Polyunsaturated fatty acid-cholesterol interactions: domain formation in membranes. In: *Biochimica et Biophysica Acta (BBA)-Biomembranes* 1788 (2009), no. 1, pp. 24–32

[WBDK10] Wurtele, E. S.; Bassham, D. C.; Dickerson, J.; Kabala, D. J.; Schneller, W.; Stenerson, M.; Vasanth, A.: Meta!Blast: a serious game to explore the complexities of structural and metabolic cell biology. In: *Proceedings of the ASME 2010 World Conference on Innovative Virtual Reality* : ASME, 2010

[Webb92] Webb, E. C.: *Enzyme Nomenclature - Recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the Nomenclature and Classification of Enzymes, NC-IUBMB*. San Diego, USA : Academic Press, 1992

[Wint12] Winter, MarK: *WebElements Periodic Table of the Elements*. URL http://www.webelements.com/. - retrieved on 07/29/2012. — WebElements: the periodic table on the web

[WLDZ03] Wurtele, E. S.; Li, J.; Diao, L.; Zhang, H.; Foster, C. M.; Fatland, B.; Dickerson, J.; Brown, A. et al.: MetNet: software to build and model the biogenetic lattice of Arabidopsis. In: *Comparative and Functional Genomics* 4 (2003), no. 2, pp. 239–245

[WoRo94a] Woolf, T. B.; Roux, B.: Molecular dynamics simulation of the gramicidin channel in a phospholipid bilayer. In: *Proceedings of the National Academy of Sciences* 91 (1994), no. 24, pp. 11631–11635

[WoRo94b] Woolf, T. B.; Roux, B.: Conformational flexibility of o-phosphorylcholine and o-phosphorylethanolamine: a molecular dynamics study of solvation effects. In: *Journal of the American Chemical Society* 116 (1994), no. 13, pp. 5916–5926

[WoRo96] Woolf, T. B; Roux, B.: Structure, energetics, and dynamics of lipid–protein interactions: a molecular dynamics study of the gramicidin A channel in a DMPC bilayer. In: *Proteins: Structure, Function, and Bioinformatics* 24 (1996), no. 1, pp. 92–114

[Wott96] Wottawa, M.: *Struktur und algorithmische Behandlung von paraxisorientierten dreidimensionalen Packungsproblemen*. Köln, Germany, University of Cologne, Doctorate Thesis, 1996

[WPLW09] Widjaja, Y. Y.; Pang, C. N. I.; Li, S. S.; Wilkins, M. R.; Lambert, T. D.: The Interactorium: Visualising proteins, complexes and interaction networks in a

virtual 3D cell. In: *Proteomics* 9 (2009), no. 23, pp. 5309–5315

[WSBB10] Walter, T.; Shattuck, D. W.; Baldock, R.; Bastin, M. E.; Carpenter, A. E.; Duce, S.; Ellenberg, J.; Fraser, A. et al.: Visualization of image data from cells to organisms. In: *Nature Methods* 7 (2010), no. 3 Suppl, pp. S26–41

[Yeag05] Yeagle, P.: *The structure of biological membranes*. New York, USA : CRC Press, 2005 — ISBN 0849314038

[Yeag87] Yeagle, P.: *The membranes of cells*. London, UK : Academic Press London, 1987 — ISBN 0127690409

[Yesy07] Yesylevskyy, S. O.: ProtSqueeze: simple and effective automated tool for setting up membrane protein simulations. In: *Journal of Chemical Information and Modeling* 47 (2007), no. 5, pp. 1986–1994

[YEVO08] Yetukuri, L.; Ekroos, K.; Vidal-Puig, A.; Oresic, M.: Informatics and computational strategies for the study of lipids. In: *Molecular BioSystems* 4 (2008), no. 2, pp. 121–127

[YEWC05] Yang, Y.; Engin, L.; Wurtele, E. S; Cruz-Neira, C.; Dickerson, J. A: Integration of metabolic networks and gene expression in virtual reality. In: *Bioinformatics* 21 (2005), no. 18, p. 3645

[YLPS08] Yamaguchi, R.; Lartigue, L.; Perkins, G.; Scott, R. T.; Dixit, A.; Kushnareva, Y.; Kuwana, T.; Ellisman, M. H. et al.: Opa1-mediated cristae opening is Bax/Bak and BH3 dependent, required for apoptosis, and independent of Bak oligomerization. In: *Molecular Cell* 31 (2008), no. 4, pp. 557–569

[YuTo04] Yugi, K.; Tomita, M.: A general computational model of mitochondrial metabolism in a whole organelle scale. In: *Bioinformatics* 20 (2004), no. 11, pp. 1795–1796

[YWCD06] Yang, Y.; Wurtele, E. S.; Cruz-Neira, C.; Dickerson, J. A.: Hierarchical visualization of metabolic networks using virtual reality. In: *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications* : ACM, 2006 — ISBN 1595933247, pp. 377–381

[Zami11] Zamis, T. M.: *Chemistry Molecular Models. http://www.uwsp.edu/chemistry/pdbs/ (accessed May 3, 2010)*. URL http://www.uwsp.edu/chemistry/pdbs/. - retrieved on 02/16/2011

[ZhBF09] Zheng, Y. Z.; Berg, K. B.; Foster, L. J.: Mitochondria do not contain lipid rafts, and lipid rafts do not contain mitochondrial proteins. In: *Journal of Lipid Research* 50 (2009), no. 5, pp. 988–998

[ZMHR09] Zidar, J.; Merzel, F.; Hodošček, M.; Rebolj, K.; Sepčić, K.; Maček, P.; Janežič, D.: Liquid-ordered phase formation in cholesterol/sphingomyelin bilayers: all-atom molecular dynamics simulations. In: *The Journal of Physical Chemistry B* 113 (2009), no. 48, pp. 15795–15802

[ZPAL10] Zini, M. F.; Porozov, Y.; Andrei, R. M.; Loni, T.; Caudai, C.; Zoppè, M.: BioBlender: fast and efficient all atom morphing of proteins using Blender game engine. In: *Arxiv preprint arXiv:1009.4801* (2010)

[Zysi07] Zysik, M.: *Programmieren einer Software für die Visualisierung von Zell- und Membranmodellen in VRML und PDB*. Bielefeld, Germany, Bielefeld University, Diploma Thesis, 2007

9 Appendix

9.1 Biological Abbreviations

9.1.1 Lipid Abbreviations [Genn89]

9.1.1.1 (Glycero-)Phospholipids

PL phospholipid, glycerophospholipid

9.1.1.1.1 Cardiolipin

CL cardiolipin

DPG diphosphatidyl glycerol

9.1.1.1.2 Phosphatidic Acid

PA phosphatidic acid

9.1.1.1.3 Phosphatidylcholine

PC phosphatidylcholine

DEPC dieleidoyl phosphatidylcholine

DHPC dihexanoyl phosphatidylcholine

DLPC dilauroyl phosphatidylcholine

DMPC dimyristoyl phosphatidylcholine

DOPC dioleoyl phosphatidylcholine

DPPC dipalmitoyl phosphatidylcholine

POPC 1-palmitoyl-2-oleoyl phosphatidylcholine

SOPC 1-stearol-2-oleoyl phosphatidylcholine

9.1.1.2 Phosphatidylethanolamine

PE phosphatidylethanolamine

DLPE dilauroyl phosphatidylethanolamine

DMPE dimyristoyl phosphatidylethanolamine

DOPE dioleoyl phosphatidylethanolamine

DPPE dipalmitoyl phosphatidylethanolamine

SOPE 1-stearol-2-oleoyl phosphatidylethanolamine

9.1.1.2.1 Phosphatidylglycerol

PG phosphatidylglycerol

DMPG dimyristoyl phosphatidylglycerol

POPG 1-palmitoyl-2-oleoyl phosphatidylglycerol

9.1.1.2.2 Phosphatidylinositol

GPI Glycosylphosphatidylinositol

PI phosphatidylinositol

9.1.1.2.3 Phosphatidylserine

PS phosphatidylserine

DMPS dimyristoyl phosphatidylserine

9.1.1.3 Sphingolipids

9.1.1.3.1 Sphingomyelin

SM sphingomyelin

9.1.1.4 Sterols

9.1.1.4.1 Cholesterol

Chol Cholesterol

9.1.2 Other Biological Abbreviations

ATP Adenosine Triphosphate

ER Endoplasmatic Reticulum

$FADH_2$ Flavin Adenine Dinucleotide, oxidized form

H&E hematoxylin and eosin

NAD^+ Nicotinamide Adenine Dinucleotide, oxidized form

NADH Nicotinamide Adenine Dinucleotide, reduced form

9.2 Technical Abbreviations

2D two-dimensional

2.5D two-and-a-half-dimensional

3D three-dimensional

4D four-dimensional

3ds Max Autodesk® 3d Studio Max®

6DOF 6 Degrees of Freedom

*X*D-KP *X*-Dimensional KP

AP Advanced Random Placing algorithm

ATP Adenosine Triphosphate

AVS Advanced Visual Systems: unstructured cell data

BKP Bounded KP

CAVE Cave Automatic Virtual Environment

CC Creative Commons License

CE^3 CellExplorer Educational Edition

Cm1 CELLmicrocosmos CellExplorer Project

CmCX CELLmicrocosmos CellExplorer

Cm2	CELLmicrocosmos MembraneEditor Project
CmME	CELLmicrocosmos MembraneEditor
Cm3	CELLmicrocosmos CellEditor Project
CmCE	CELLmicrocosmos CellEditor
Cm4	CELLmicrocosmos PathwayIntegration Project
CmPI	CELLmicrocosmos PathwayIntegration
D_l	lipid packing density
DICOM	Digital Imaging and Communications in Medicine
FM3	Fast Multipole Multilevel Method
FRAP	Fluorescence recovery After Photobleaching
GEOMI	Geometry for Maximum Insight
GUI	Graphical User Interface
HIC-Up	Hetero-compound Information Centre – Uppsala
HUD	Head-Up Display
JND	Just Noticeable Difference
KP	Knapsack Problem
LP	Linear Placing algorithm
LPA	Lipid Packing Algorithm
LPP	Lipid Packing Problem
MC	Monte Carlo Simulation
MD	Molecular Dynamic Simulation
MNV	Metabolic Network Visualizer
MPA	Membrane Packing Algorithm
MPP	Membrane Packing Problem
PDB	Protein Data Bank (format)
PIF	Pots Initialization Files (CompuCell3D)
POI	Point of Interest
PPA	Protein Packing Algorithm
PPP	Protein Packing Problem
RP	Random Placing algorithm
RS&Z	Rakhmanov, Saff and Zhou
SD	Standard Deviation
PLY	Stanford Polygon Format
SPP	Sphere Packing Problem
STL	Stereolithography
TIFF	Tagged Image File Format

TW	The Wanderer algorithm
UKP	Unbounded KP
VCML	Virtual Cell Markup Language
VFRAP	Virtual FRAP
VRML	Virtual Reality Markup Language
VRML97	Virtual Reality Markup Language, Version 2.0 (introduced in 1997)
XWG	native GEOMI XML format

9.3 Optimization Problem Definition

- Membrane Packing Problem (MPP)
  - Lipid Packing Problem (LPP)
    - OPT_LPP_MIN_AREA: The optimum is defined by the minimal possible area per lipid ($Å^2$) (Sections 6.2.1 and 6.2.4).
    - OPT_LPP_DEF_AREA: The optimum is defined by the user-defined area per lipid ($Å^2$) (Section 6.2.1).
    - OPT_LPP_2D_AREA: The optimum is defined by the highest possible membrane area occupied by lipids (%) (Section 6.2.2).
    - OPT_LPP_RATIO: The optimum is defined by reaching the correct lipid type ratios (%) (Sections 6.2.1, 6.2.2, 6.2.4 and 6.2.5).
    - OPT_LPP_ENERGY: The optimum is defined by achieving the energetic equilibrium state [SDGS11].
  - Protein Packing Problem (PPP)
    - OPT_PPP_POS: The optimum is defined by the correct protein positioning in relation to the bilayer (Section 6.2.3).
- Network Mapping Problem (NMP)
  - Node Distribution Problem
    - OPT_NMP_MAX_AREA: The optimum is defined by the maximal possible area per node (Section 3.3.4.1).
  - Network Layout Problem
    - OPT_NMP_ENERGY: The optimum is an equilibrated layout (Section 3.3.4.2).
  - Node Mapping Problem
    - OPT_NMP_POS: The optimum is defined by the correct network positioning in relation to the center of the cell component (Section 3.3.4.3).

9.4 Abstraction Levels

In Section 5.1.2.1, four different abstraction levels have been introduced:

- ABS_LEV_0: this is an ideal value: an abstraction level 0 would be the perfect copy of a

cell component, completely reflecting its natural shape (also: Clone),

- ABS_LEV_1: 3D-microscopy- or spectroscopy-based cell component models (also: Image),
- ABS_LEV_2: interpretative cell component models based on different two- or three-dimensional microscopic images but designed by mouse-based and script-based modeling techniques (also: Allegory), and
- ABS_LEV_3: abstract cell component models, often simple primitives (also: Abstraction).

9.5 Special Terms

amphipathic	containing both hydrophilic and hydrophobic regions
conformational	the conformation of the molecule is involved, e.g. the position of atoms is changed
hydrophilic	water-loving
hydrophobic	water-repulsing
structural	the structure of the molecule is involved, e.g. new atoms are added

9.6 The PDB Format: An Example

Because PDB is the most important format for protein, lipid as well as membrane modeling, a few remarks according the format definition should be made here using the structure file of one of the smallest PDB files available, 2ONX.pdb, which contains fibrils formed from different proteins (Figure 172).

The first line should be always the *HEADER* indicating the name of the PDB structure, the publication date and the *four-digit PDB identifier*. The *TITLE* provides an option to define a more precise name for the structure. The *COMPND* gives information about the compounds found in this PDB file. This may be a lipid or a sugar which was extracted together with the molecule of interest. Also found in this file is: additional data regarding the authors, the journal where the structure was published, keywords, the source (organism), the size of the crystal as well as the spatial deviations. Due to the fact that frequently the PDB file has to be adjusted to new developments of the PDB database, there is a revision number is given. The *REMARK* section is where all data can be included which does not fit into the other sections, for example, the complete title of a publication, details concerning the authors or experiments, contact information and also the resolution. The expectations regarding the resolution based on the applied experimental method (Section 3.2.1.1) are approved when comparing the value of 1.52 Å with the *EXPDTA* entry “X-ray diffraction”.

Based on the identifiers in the atomic section (the chain, residue and insertion identifier), it is also possible to define the secondary structure of a protein. Helices of different types (the standard is a right-handed alpha helix) as well as sheets are possible (see Section 2.2.1.4). Of course, also here the limitations stated above have to be taken into account.

In addition, it is possible to integrate different models into one PDB file. But these models have to be all of the same atomic structure, only the positions, the conformations may vary.

There are of course many other entries, but the explanation lies behind the scope of this

work. An overview can be found directly at [11r].

```
      1       2       3       4       5       6       7       8
12345678901234567890123456789012345678901234567890123456789012345678901234567890
ATOM      1  N   ASN A  1       0.452   2.495   0.430  1.00 13.01           N
```

Figure 171: The first coordinate line of 2ONX.pdb

The most important section contains the consecutively written ATOM entries. Figure 171 reveals the limitations of the PDB format. The length of a line is restricted to 80 digits. There is a strict syntax for the different values stored per line [11r]:

- 1 – 6: The *record name* "ATOM".
- 7 – 11: The integer *serial atom number*. Each atom should have a unique identifier.
- 13 – 16: The *name of the atom*. Here it is "N" for nitrogen.
- 17: The alternate location indicator has to be used if an atom was localized in different positions. These different positions can then be stored by iteration through this character.
- 18 – 20: The *name of the residue*, here "ASN" for asparagine (see Figure 172).
- 22: The *chain identifier*. Each polypeptide and/or nucleic acid has a unique identifier, a character. A chain is terminated by a "TER" line (see Figure 172). This is especially important if a PDB file contains different structures, as in a membrane complex.
- 23 – 26: The *residue sequence number*. This molecule contains four residues: two ASN residues plus two GLN (glutamine) residues (see Figure 172).
- 27: The code for the insertion of residues. This letter code is normally only used if two or more residues have the same numbering. The unique *residue identifier* is therefore the combination of the sequence number and the insertion code.
- 31 – 38: The coordinates for *X* in Å (a 8.3 real number).
- 39 – 46: The coordinates for *Y* in Å (a 8.3 real number).
- 47 – 54: The coordinates for *Z* in Å (a 8.3 real number).
- 55 – 60: The *occupancy*.
- 61 – 66: The *temperature factor*.
- 77 – 78: The *symbol of the element*.
- 79 – 80: The atomic charge.

Based on this definition, there are the following restrictions:

- There are 99,999 different unique atom identifiers possible.
- There are maximally 26 different locations per atom possible.
- There are maximally 26 different chains possible.
- There are maximally 9,999 different residue numbers.
- There are maximally 26 different insertion codes possible.
- Combining the sequence number with the insertion code, 26 multiplied with 9,999, resulting in 259,974 unique residues are possible.

- Each coordinate has to be within the boundaries of minimal -999.999 and maximal of +9,999.999 Å. The maximal box length is therefore approximately 11,000 Å.

9.7 A Simple Membrane Packing Algorithm

Figure 161 shows the main method *fillWithLipids()* of a very simple Membrane Packing Algorithm compatible to the CELLmicrocosmos 2.2 MembraneEditor.

9.8 The Enzyme Classification: An Example

The first EC number defines the class of the enzyme (Section 2.3.1 and 3.3.2.2.2):

1. Oxidoreductases,
2. Transferases,
3. Hydrolases,
4. Lyases,
5. Isomerases, and
6. Ligases.

The subsequent numbers define the subclasses. For example, the number 1.2.3.4 defines the following enzyme:

1. Oxidoreductases,
2. “Acting on the aldehyde or oxo group of donors”,
3. “With oxygen as acceptor”, and
4. oxalate oxidase.

9.9 2D Visualization of the Glycolysis in the 2D Viewer of CmPI

Figure 174 shows the Glycolysis as discussed in Section 6.3.1.1.

9.10 Coordinate Axes

The axes are defined as follows in all CELLmicrocosmos programs:

- X is the width,
- Y is the height and
- Z is the depth.

This is the typical description of coordinate systems usually known from school, undergraduate studies and research. Of course, there may be exceptions. Especially many PDB-related programs tend to interchange the Y (depth) and Z (height) coordinate. For this reason, MembraneEditor offers an option to alternate between the coordinate definitions during the PDB export.

```
HEADER    PROTEIN FIBRIL                          24-JAN-07   2ONX
TITLE     NNQQ PEPTIDE CORRESPONDING TO RESIDUES 8-11 OF YEAST PRION
COMPND    MOL_ID: 1;
SOURCE    MOL_ID: 1;
SOURCE   2 SYNTHETIC: YES
KEYWDS    STERIC ZIPPER, BETA SHEETS, PROTEIN FIBRIL
EXPDTA    X-RAY DIFFRACTION
AUTHOR    M.R.SAWAYA,S.SAMBASHIVAN,R.NELSON,M.IVANOVA,S.A.SIEVERS,
REVDAT   3   24-FEB-09 2ONX    1       VERSN
JRNL        DOI    10.1038/NATURE05695
REMARK   1
REMARK   2
REMARK   2 RESOLUTION.    1.52 ANGSTROMS.
DBREF  2ONX A    1     4  PDB    2ONX     2ONX             1      4
SEQRES   1 A    4  ASN ASN GLN GLN
CRYST1    4.854   16.014   15.546  90.00  96.91  90.00 P 1 21 1      2
ORIGX1      1.000000  0.000000  0.000000        0.00000
ORIGX2      0.000000  1.000000  0.000000        0.00000
ORIGX3      0.000000  0.000000  1.000000        0.00000
SCALE1      0.206016  0.000000  0.024967        0.00000
SCALE2      0.000000  0.062445  0.000000        0.00000
SCALE3      0.000000  0.000000  0.064796        0.00000
ATOM      1  N   ASN A   1       0.452   2.495   0.430  1.00 13.01           N
ATOM      2  CA  ASN A   1       0.395   3.556   1.478  1.00 15.49           C
ATOM      3  C   ASN A   1      -0.265   3.026   2.741  1.00 17.59           C
ATOM      4  O   ASN A   1      -1.491   3.018   2.841  1.00 22.47           O
ATOM      5  CB  ASN A   1      -0.357   4.785   0.989  1.00 23.58           C
ATOM      6  CG  ASN A   1       0.413   5.575  -0.038  1.00 33.44           C
ATOM      7  OD1 ASN A   1       1.123   6.520   0.300  1.00 40.45           O
ATOM      8  ND2 ASN A   1       0.280   5.193  -1.305  1.00 32.63           N
ATOM      9  N   ASN A   2       0.558   2.552   3.675  1.00  8.09           N
ATOM     10  CA  ASN A   2       0.140   2.007   4.983  1.00  7.76           C
ATOM     11  C   ASN A   2       0.620   2.936   6.093  1.00 10.30           C
ATOM     12  O   ASN A   2       1.782   3.318   6.098  1.00 13.35           O
ATOM     13  CB  ASN A   2       0.827   0.661   5.208  1.00  5.39           C
ATOM     14  CG  ASN A   2       0.222  -0.450   4.441  1.00 11.18           C
ATOM     15  OD1 ASN A   2      -0.957  -0.425   4.134  1.00 14.38           O
ATOM     16  ND2 ASN A   2       1.007  -1.500   4.203  1.00 11.50           N
ATOM     17  N   GLN A   3      -0.251   3.283   7.036  1.00  9.07           N
ATOM     18  CA  GLN A   3       0.168   4.035   8.225  1.00  6.77           C
ATOM     19  C   GLN A   3      -0.483   3.441   9.469  1.00  7.12           C
ATOM     20  O   GLN A   3      -1.715   3.312   9.528  1.00  7.25           O
ATOM     21  CB  GLN A   3      -0.179   5.530   8.149  1.00 10.37           C
ATOM     22  CG  GLN A   3       0.537   6.314   7.109  1.00  7.68           C
ATOM     23  CD  GLN A   3       0.053   7.762   7.086  1.00  9.06           C
ATOM     24  OE1 GLN A   3      -1.169   8.035   7.199  1.00 10.09           O
ATOM     25  NE2 GLN A   3       0.997   8.690   6.976  1.00  6.75           N
ATOM     26  N   GLN A   4       0.339   3.060  10.461  1.00 12.42           N
ATOM     27  CA  GLN A   4      -0.182   2.636  11.781  1.00 16.85           C
ATOM     28  C   GLN A   4       0.753   2.947  12.952  1.00 28.21           C
ATOM     29  O   GLN A   4       1.615   3.841  12.868  1.00 34.15           O
ATOM     30  CB  GLN A   4      -0.526   1.156  11.784  1.00 18.06           C
ATOM     31  CG  GLN A   4       0.648   0.212  11.888  1.00 12.50           C
ATOM     32  CD  GLN A   4       0.233  -1.201  11.608  1.00 16.78           C
ATOM     33  OE1 GLN A   4      -0.963  -1.494  11.482  1.00 19.98           O
ATOM     34  NE2 GLN A   4       1.209  -2.096  11.524  1.00  9.66           N
ATOM     35  OXT GLN A   4       0.645   2.343  14.027  1.00 29.30           O
TER      36      GLN A   4
MASTER      245    0    0    0    0    0    0    6   35    1    0    1
END
```

Figure 172: Excerpt of one of the smallest PDB files, 2ONX (some lines are omitted)

```
// the main method for the lipid placement; other methods provide options to create an initial GUI
// window, runtime options, supporting information for author and description, etc.; the following
// variables have to be defined before: this could be done by the programmer of algorithm or by
// the user if the programmer implemented a GUI for these values:
// boolean rotateLipids, int desiredNumberOfLipids
public void fillWithLipids() throws Exception {
    int currentCount = 0;
    // sets the actual membrane side to the intracellular one
    int mSide = MembraneAlgorithm.INTRA;
    // repeats the placement process until the algorithm is manually stopped by the user
    // or the defined number of lipids in the membrane is reached
    while (stop == false && currentCount < this.desiredNumberOfLipids) {
        // changes each time the actual membrane side to extra- or intra-cellular
        if (mSide == MembraneAlgorithm.INTRA)
            mSide = MembraneAlgorithm.EXTRA;
        else
            mSide = MembraneAlgorithm.INTRA;
        // creates a random position between point 0/0 and the membrane width (x) and depth (z)
        Point2f randomPosition = this.createRandomXZPosition(new Rectangle(0, 0,
              this.getMembraneSize().width, this.getMembraneSize().height));
        Lipid newLipid;
        // checks if there is a microdomain at the computed random position
        int domainID = this.getMicroDomainAtPoint(randomPosition);
        // if there is a microdomain at this position, the new lipid is created in the microdomain
        if (domainID > 0) {
            MicroDomain domain = this.getMicroDomainAt(domainID - 1);
            newLipid = mSide == EXTRA ? domain.createNextExtracellularLipid() :
                                        domain.createNextIntracellularLipid();
        // if there is no microdomain, the lipid is added to the default area of the membrane
        } else {
            newLipid = mSide == EXTRA ? this.createNextExtracellularLipid() :
                                        this.createNextIntracellularLipid();
        }
        // cancel the adding of this lipid, if the creation of a new lipid did not succeed
        if (newLipid == null) continue;
        // define the position of the newly created lipid as the random position already created
        newLipid.setPosition(new Point3f( randomPosition.x,
                                          newLipid.getYTranslation(), randomPosition.y));
        // GUI: if the user chooses to alter the rotation, do it randomly here
        if (this.rotateLipids)
            newLipid.rotY(new Random().nextInt(359) + new Random().nextFloat());
        // if the new lipid intersects the membrane borders or another molecule, cancel placement
        if (this.intersectsBorders(newLipid) || this.intersects(newLipid))
            continue;
        // finally, the new lipid is added to the predefined domain
        if (mSide == EXTRA)
            this.addExtracellularLipid(newLipid);
        else
            this.addIntracellularLipid(newLipid);
        // iterate the number of lipids, needed to find out if the algorithm is finished
        currentCount++;
    }
}
```

Figure 173: The source code of the fillWithLipids() function of a simple CmME MPA

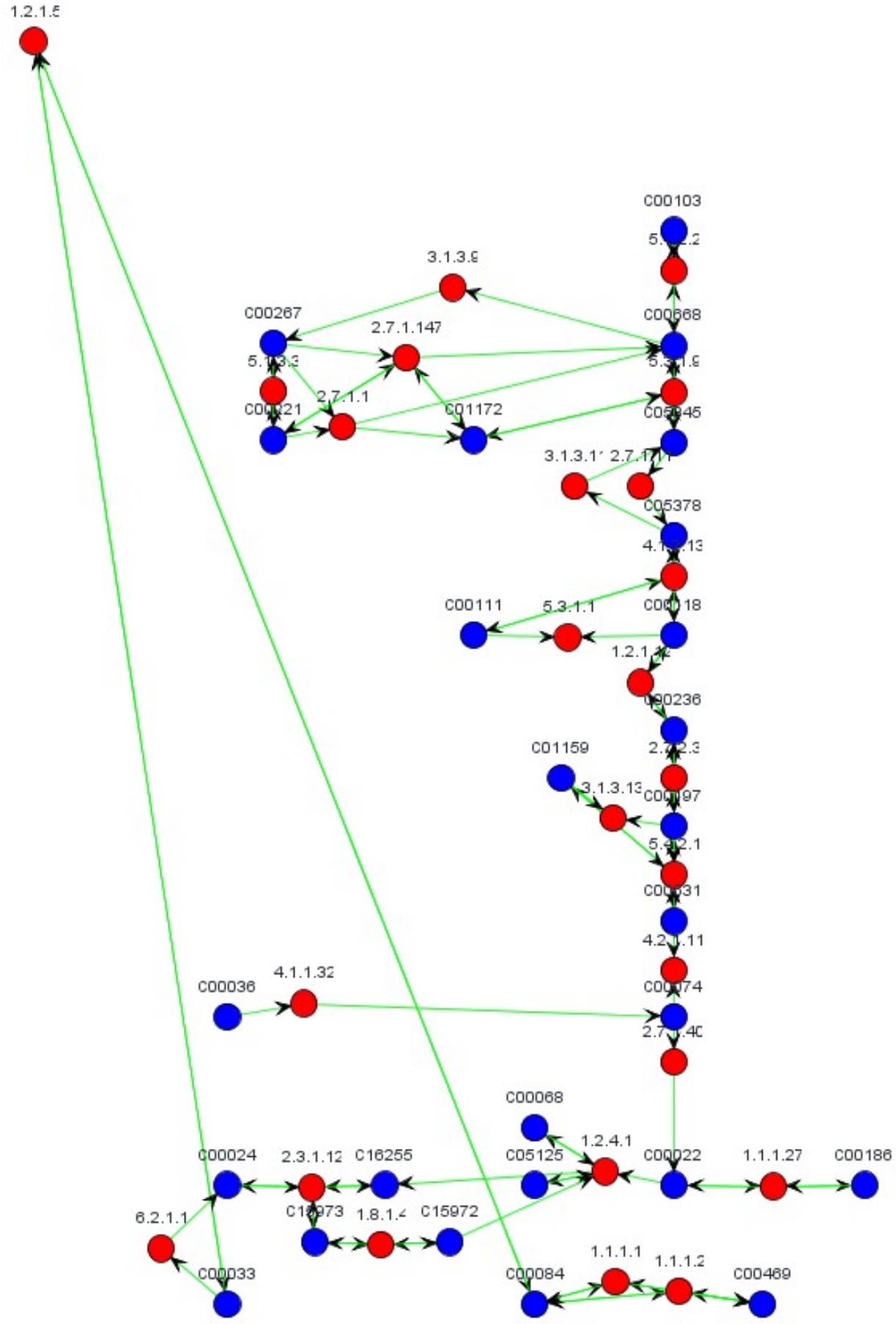


Figure 174: 2D Visualization of the Glycolysis (hsa00010) in CmPI

9.11 Units and molecular components

ATP	Adenosine 5'-triphosphate
kcal/mol	Kilocalorie per mole
kJ/mol	Kilo Joule per mole
mM	millimolar (1 mol/m^3)

9.12 Units of the cell

1 m (Meter) = 1,000 mm (Millimeter)
= 1,000,000 μm (Micrometer)
= 1*10^9 nm (Nanometer)
= 1*10^{10} Å (Ångstrom)

9.13 Common names of fatty acids

Common name	**Chain length: unsaturation**
Lauric	C12:0
Myristic	C14:0
Palmitic	C16:0
Palmitoleic	trans-9-C16:1
Stearic	C18:0
Oleic	cis-9-C18:1
Elaidic	trans-9-C18:1
Vaccenic	trans-11-C18:1
Linoleic	cis-9,cis-12-C18:2
γ-Linolenic	cis-6,cis-9,cis-12-C18:3
α-Linolenic	cis-9,cis-12,cis-15-C18:3
Arachidic	C20:0
Behenic	C22:0
Arachidonic	cis-5,cis-8,cis-11,cis-14-C20:4

Table 23: Some Fatty acids commonly found in membrane lipids.

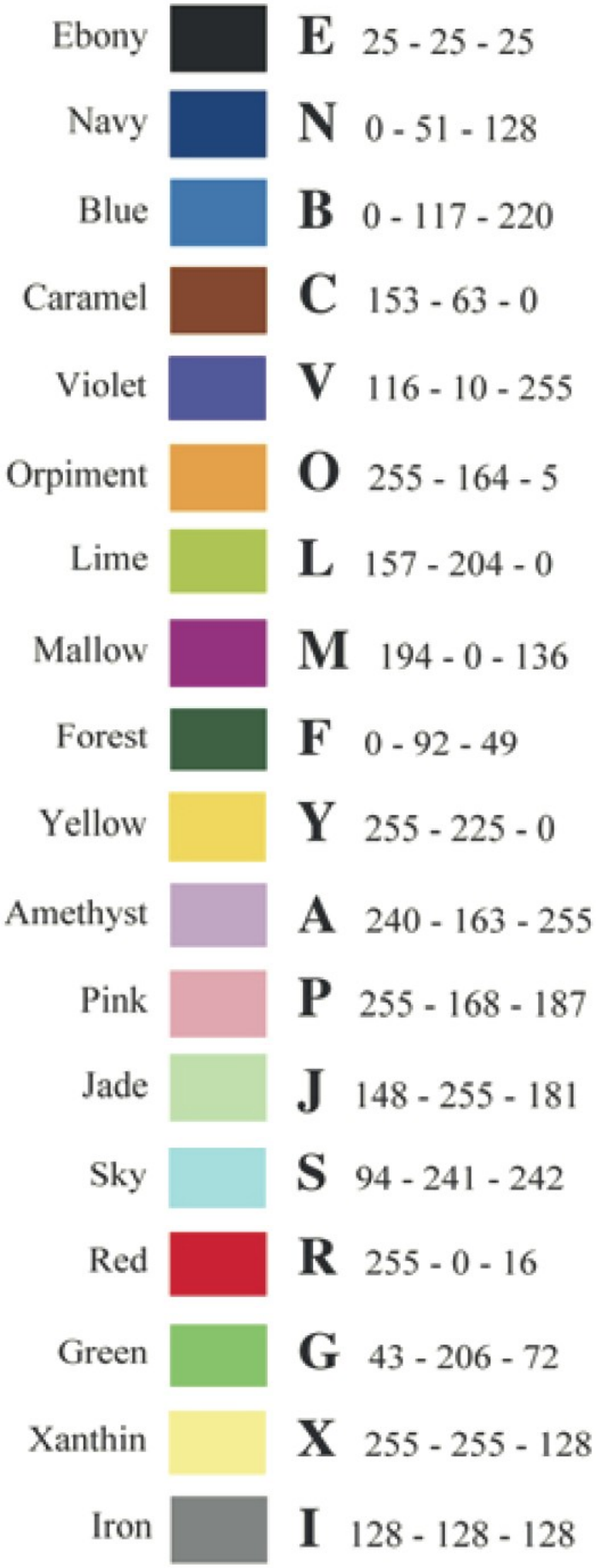
The common names are given and can be compared to the lipid names in Appendix 9.1.1.

9.14 The Color Codes derived from the Color Alphabet of Green-Armytage

The cell components are listed according to the sequential order of the SphereCell (Section 5.1.2.2). The colors are taken from the work “A Colour Alphabet and the Limits of Colour Coding” by Paul Green-Armytage [Gree10]. See also Section 5.1.2.3.1.

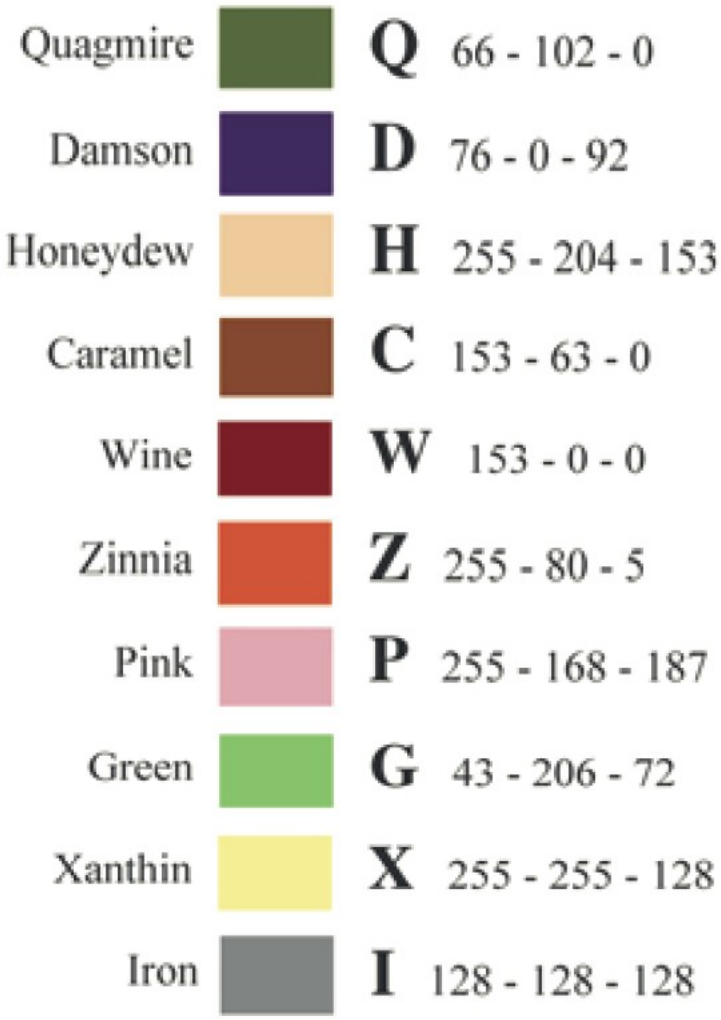
9.14.1 Color Codes for Eukayotic Cell Components

- NUCLEOLUS
- NUCLEUS
- ROUGHER
- RIBOSOME
- SMOOTHER
- GOLGI
- LYSOSOME
- MITOCHONDRION
- CHLOROPLAST
- PEROXISOME
- VESICLE
- CYTOSOL
- VACUOLAR
- ENDOSOME
- CELL
- CELLWALL
- EXTRACELLULAR
- UNKNOWN



9.14.2 Color Codes for Prokaroytic Cell Components

- STORAGEGRANULE
- NUCLEOID
- PLASMID
- RIBOSOME
- CYTOPLASMIC MEMB.
- CAPSULE
- CYTOSOL
- CELLWALL
- EXTRACELLULAR
- UNKNOWN



9.15 Versioning Information

As can be seen in Subchapter 9.16, the CELLmicrocosmos project has been underway for nearly a decade. Therefore it might be a bit confusing for external users to understand the versions of the CELLmicrocosmos releases, especially because the version of the main software packages do not reflect the chronology.

The first version number corresponds to the CELLmicrocosmos project. As of yet, there are five projects:

1. CellExplorer
2. MembraneEditor
3. CellEditor
4. PathwayIntegration

The second version number reflects the state of the project:

0. The prototype (alpha version). This software package works for its particular purpose, but the implementation style is quick and dirty, the result is unstable. This process normally takes one semester.

1. The first revision (beta version). There are two possibilities: the code of the prototype is taken as the base for the revision process or, only the idea of the prototype including the results and improvement proposals are taken into account for a complete re-implementation. This process normally takes years.
2. The second revision (first full-release). This is the first major release version. This is not a prototype, it is a product which was tested and improved during many different projects and theses and which is intended as a contribution to the scientific community. The following minor improvements are marked by the third version number.
3. The third revision: The second major release, including all improvements of the former packages, plus some new features dramatically extending the former release. This version number was not used prior to submission of this thesis.

The third version number indicates minor versions of major releases, starting from 0. The changes are found in the release notes of the according package.

The actual state of the different projects discussed in this work is: 1.1, 2.2.1, 3.1, 4.2

9.16 Projects and Participants

To realize this work, many different projects had to be created. A number of students were working at this project over the last few years. A short overview of the history is provided with the intention to thank each of the participants listed here.

Bachelor Thesis: ZellVisualisierung (Björn Sommer, WS2003/2004)

Supervisors: Dr. Dieter Lorenz, Prof. Dr. Franz Kummert

Here the initial idea for CELLmicrocosmos was created. The corporate identity and its philosophy, a number of cell components based on electron and light microscopy and an animation showing a fly through an abstract liver cell [Somm04].

Project: CELLmicrocosmos 1 (Dr. Dieter Lorenz, SS2005)

Participants: Matthias Bartneck, Sebastian Janowski, Björn Sommer

Led by Dr. Dieter Lorenz, we began to create a first Virtual Cell environment in Amira [Konr02]. The cell model created during the initial bachelor thesis was used and we began constructing a C++/Tcl/TK based Amira plugin. In addition, a first collection of cell biology based knowledge was created.

Project: CELLmicrocosmos 2 MembraneEditor (Dr. Dieter Lorenz, WS2005/2006)

Participants: Sebastian Janowski, Dominik Mertens, Jan Pieniak, Robert Schmieder, Björn Sommer, Martha Zakrzewski

An initial, very simple version of MembraneEditor was created. It was already able to export a 3D PDB membrane, but the distribution was based only on a two dimensional rectangle and the visualization was also two-dimensional.

Master Thesis: Interaktive 3D-ZellVisualisierung (Björn Sommer, SS2006)

Supervisors: Dr. Dieter Lorenz, Prof. Dr. Ralf Hofestädt

The Amira plugin developed during the Cm 1 project was extended towards a learning environment and results of the Cm2 project were integrated into the cell environment. Different animations were created giving an introduction into a liver cell. In addition, the first version of the website was created [Somm06].

Project: CELLmicrocosmos 2.1 MembraneEditor Revision (Björn Sommer, WS2006/2007)

Participants: Andrea Steinmetz, Christian Fink, Matthias Koch, Ralf Mertens, Sebastian Schneider, Tim Dingersen

The Cm 2 MembraneEditor project was completely revised. The project supported the diploma thesis of Sebastian Schneider. Tim Dingersen was mainly involved in the development of this tool. After the completion of this project, he continued to work as a co-worker on MembraneEditor.

Project: CELLmicrocosmos 3 CellEditor (Björn Sommer, SS2007)

Participants: Chan Lin, Olga Mantler, Ufuk Aydin, Xiaoliang Song

The first version of CellEditor supported the diploma thesis of Marion Zysik and should create an alternative to the expensive Amira cell environment A cell was created consisting of different cell components, which could be combined with PDB membranes for the Cm 2 project.

Project: CELLmicrocosmos 3.1 CellEditor Revision (Björn Sommer, WS2007/2008)

Participants: Christina Ander, Florian Heißenberg, Jörn Winnebald, Lukas Jelonek, Philipp Unruh, Regina Bisdorf

The revision of CellEditor restructured the program and improved the GUI drastically and many “bugs” were removed from the program.

Diploma Thesis: Programmieren einer Software für die Visualisierung von Zell- und Membranmodellen in VRML und PDB (Marion Zysik, WS2007/2008)

Supervisors: Björn Sommer, Prof. Dr. Ralf Hofestädt

The first approach was created to combine the Cm3 cell models with Cm2 PDB membranes. C++ and the VTK package was used and an autostart disk was constructed to work with the program. It was the first approach to integrate the idea of membrane mapping [Zysi07].

Diploma Thesis: Entwicklung einer Schnittstelle zur Integration membranberechnender Algorithmen in den CELLmicrocosmos 2.1 (Sebastian Schneider, WS2007/2008)

Supervisors: Björn Sommer, Prof. Dr. Ralf Hofestädt

The first Revision of MembraneEditor was used to integrate an algorithm plugin interface and to develop different algorithms to generate a membrane. The most sophisticated approach was a Simulated Annealing algorithm, which simulated a geometry-based cooling-down or a heating-up process [Schn08].

Project: CELLmicrocosmos 4 PathwayIntegration (Björn Sommer, SS2008)

Participants: Arne Husemann, Madis Rumming, Sebastian Janowski

Using the Cm 3 CellEditor, metabolic pathways were correlated with cell components, a simple node distribution algorithm was integrated and a first connection to the KEGG database (integrated in the BioDWH [TKKH08]) was realized. In addition, the GUI was extended to show node information.

Project: CELLmicrocosmos 4.1 PathwayIntegration Revision (Björn Sommer,

WS2008/2009)

Participants: Jörn Künsemöller, Norbert Sand

The Revision of the PathwayIntegration project dealt with the improvement of navigation. After analyzing different world navigation approaches, three different navigation modes were created extended by features such as a tour editor..

Bachelor Thesis: Recherche, Berechnung & Visualisierung der Lipid-/Proteinrelationen in Mitochondrienmembranen mit dem für diese Zwecke zu erweiternden CELLmicrocosmos 2.1 Membrane-Editor (Sebastian Rubert, WS2008/2009)

Supervisors: Björn Sommer, Prof. Dr. Ralf Hofestädt

This thesis analyzed the quality of the lipid/protein relationships used in MembraneEditor and improved the accuracy drastically. Two different papers containing lipid distributions were analyzed and visualized using MembraneEditor [Rube08].

Project: CELLmicrocosmos 1.1 CellExplorer (Björn Sommer, SS2009)

Participants: Marco Civico, Roland Orlik, Dominik Vahrenhorst

Going back to the idea of a CellExplorer, which once had been realized using Amira in a very simple version, this project started to correlate results coming from MembraneEditor with those of CellEditor. Here the Membrane Mapping method, once examined with the diploma thesis of Marion Zysik, was reinvoked.

Bachelor Thesis: Erweiterung des Zellmembraneditors CELLmicrocosmos 2.1 um eine semi-automatische Proteinplatzierung unter Verwendung von PDBTM-Daten (Christian Gamroth, WS2009/2010)

Supervisors: Björn Sommer, Prof. Dr. Karl-Josef Dietz

The MembraneEditor was extended with the option to automatically place proteins contained in the PDBTM database [Gamr09].

Diploma Thesis: Implementierung von SBML-Import/Export in den CELLmicrocosmos Cell Explorer unter Verwendung von metabolischen Netzwerken und 3D-Daten (Özgür Ates, WS2009/2010)

Supervisors: Björn Sommer, Prof. Dr. Ralf Hofestädt

SBML import and export support was added to the PathwayIntegration. Everything saved in .Cm4 format can also now be saved in SBML format, Level 2 Version 3 [Özgü10].

Diploma Thesis: Die Generierung energetisch günstiger Membransegmente in CELLmicrocosmos 2.2 MembraneEditor durch die Erweiterung auf eine atomare Behandlung der Moleküle (Tim Dingersen) (SS2010)

Supervisors: Björn Sommer, Prof. Dr. Karl-Josef Dietz

Tim Dingersen, who, as an undergraduate assistant, was mainly involved in the development of CmME, implemented an important feature: the atomic handling of molecules. For this purpose, a Molecule Editor was added, the Membrane Algorithm Interface was extended and a first algorithm was developed to handle a very simple atomic equilibration process [Ding10].

Project: CELLmicocosmos 3.2 Cell Modeling (SS2010, Björn Sommer)

Participants: Marco Civico, Christian Gamroth, Gunther Lukat, Christina Kropp, Roland Orlik, Jonas Osterloff, Nils Rothe, Ralf Rotzoll, Sebastian Rubert, Alexander Schäfer, Samy Slaih, Rudolf Warkentin, Pascal Witthus

This large project group was subdivided into two groups: the first group worked on the improvement of the CellExplorer to develop the CellExplorer Educational Edition (CE^3). The initial idea for this student project was conceived by Marco Civico and Roland Orlik. It resulted in the first evaluation in the Niklas Luhmann Gymnasium in Oerlinghausen at the end of 2010. The second group was lead in collaboration with Dr. Jens Krüger and began to develop the MD Edition of the MembraneEditor. The aim was to simulate membranes generated with the MembraneEditor by using GROMACS. This project led to a number of theses.

Project: CELLmicrocosmos Cell Modeling WS2010/2011 (Björn Sommer)

Participants: Christian Gamroth, Rudolf Warkentin, Pascal Witthus

In this project, 2D Mapping Layouts for the PathwayIntegration were developed, the PDB Module of the MembraneEditor was improved and the Component Builder was created which enabled the modeling of cell components compatible with the CellExplorer.

Project: CELLmicrocosmos Cell Modeling SS2011 (Björn Sommer)

Participants: Yvonne Dyck, Jonas Osterloff, Raphael-Elias Reisch

The cell component distribution algorithm was improved, the HUD was implemented and first experiments with an audio integration were made.

Master Thesis: Entwicklung eines CELLmicrocosmos 2.2 MembraneEditor Plugins zur Verwaltung von molekulardynamischen GROMACS-Simulationen in Cluster-Umgebungen (Sebastian Rubert, SS2011)

Supervisors: Björn Sommer, Dr. Jens Krüger

The GMX-Plugin was dramatically improved by Sebastian Rubert. Based on a prototype developed in CmCM SS2010, a plugin for the MembraneEditor was developed which could be used to communicate with GROMACS via ssh or UNICORE. Simulations can be externally started and the resulting data can be downloaded and analyzed using the MembraneEditor [Rube11].

Project: CELLmicrocosmos Cell Modeling WS2011/2012 (Björn Sommer)

Participants: Yan Zhou

The first version of a 3D packing algorithm was implemented, using the plugin-interface of the MembraneEditor: the VesicleBuilder.

Master Thesis: Entwicklung und Evaluierung der Educational Edition des CELLmicrocosmos CellExplorer (Marco Civico Gomez, WS2011/2012)

Supervisors: Björn Sommer, Prof. Dr. Ralf Hofestädt

Based on the first prototype of CE^3 developed in CmCM SS2010, the features were extended. For this purpose, Marco Civico was also involved in leading the CmCM SS2011 project, where some of the new features were developed. In addition, he markedly improved the TourEditor. Finally, Marco tested the software at the Gymnasium Schloß

Holte-Stukenbrock, where he is now employed as a teacher [Civi11].

Master Thesis: Analyse wirtschaftlicher Potentiale und Entwicklung von Geschäftsmodellen für Open-Source-Projekte anhand der E-Lerarning-Software CELLmicrocosmos CE^3 (Roland Peter Orlik, WS2011/2012)

Supervisors: Björn Sommer, Prof. Dr. Ralf Hofestädt

Roland Orlik reviewed different options to develop a business model based on Open Source Software. Of course, the main focus was the potential of CE^3. Different business models for this software package were developed throughout the thesis [Orli12].

Project: CELLmicrocosmos Cell Modeling SS2012 (Björn Sommer)

Participants: Christian Bender, Tobias Hoppe

A 3D-stereoscopic animation was created using Blender. A workshop was presented to the students based on a workshop presented by Christian A. Grove at the VizBi 2012 in Heidelberg. The resulting video was shown in the context of the UniVideoMagazin at CinemaxX Bielefeld using 3D-stereoscopic technique. For this opportunity, many thanks go to Paul John and Fabio Magnifico.

Diploma Thesis: Entwurf eines Programm zur Voronoi-Diagramm-unterstützten Analyse von Membransimulationen (Gunther Lukat, SS2012)

Supervisors: Björn Sommer, Dr. Jens Krüger

Gunter Lukat developed a powerful stand-alone program called APL@Voro which enables the analysis of lipid membrane models simulated with GROMACS. 2D as well as 3D diagrams which can be visualized based on Voronoi diagrams to examine the area per lipid over time [Luka12].

Diploma Thesis: Membrane-Mapping und Optimierung der molekularen Darstellung mit Java 3D in den CELLmicrocosmos Anwendungen (Philipp Unruh, SS2012)

Supervisors: Björn Sommer, Prof. Dr. Mario Botsch

Philipp Unruh improved the performance of the membrane visualization of MembraneEditor and CellExplorer by using Java-3D-internal methods. Another important aspect was the dramatic improvement of the Membrane Mapping which had been developed during the CELLmicrocosmos 1.1 project in WS2009/2010. The membranes are now mapped onto the membrane smoothly; every molecule is placed in relation to the 3D shape of the membrane [Unru12].

9.17 Implementation Work

9.17.1 Implementation Work of Björn Sommer

Cm1.1rev

- refactoring 2009
- automatic stereo adjustment
- Cell Component Coloring Methods
  - H&E Color Codes
  - Color Scales Generator

Cm2.2.1

- seed-based computation for MembranePackingAlgorithms
- Plugin Lipid Packing Algorithms
  - The Wanderer (Rookie and full version)
- Plugin tools
  - Molecule Boxifier
  - Dimension Lister
- complete OPM Integration
- 3D Stereoscopy Integration (based on Cm3.1)
- support of absolute numbers for the lipid composition, integration into the Distribution and Random Placing algorithm
- Java 7 compatibility support

Cm3.1

- cell component layer support
- custom VRML97 exporter

Cm4.1rev

- the interactive Localization Table
- cell component layer-based node mapping
- SphereCell
- shading support for networks
- coloring for networks
- Layouts
  - Distribution Algorithms
    - USphere Layout
    - UUSphere Layout
    - UUUSphere Layout
  - Layout Algorithms

    - ISOM Layout
- Layout GUI
- ANDVisio file import
- SBML Level 2 Version 4 Export and Import (based on Özgür Ates work)
- Vanesa SBML import
- Carmen SBML import
- ANDCell MySQL Querying
- DAWIS-M.D. MySQL Querying
- additional methods for 6DOF CELLmicrocosmos navigation
    - simple cell edit functionality
    - mouse wheel selection functionality
    - 3D Stereoscopy Update: updating of distance between the eyes during navigation

CmX.0

- communication between Cm1.1rev and Cm2.2
- additional methods and GUI-implementations in Cm1.1rev and Cm2.2
- using system tray implementation from Dominik Vahrenhorst

9.17.2 Implementation Work of Students and Colleagues

Cm1.1rev

- Marco Civico and Roland Orlik
    - 2nd Membrane-Mapping Approach, this time in CellExplorer
- Tim Dingersen
    - PDB-Import-Module for importing membranes from Cm2.2.1
- Jonas Osterloff
    - HUD
- Philipp Unruh
    - 3rd membrane mapping approach by parametrization of surfaces (not discussed in this thesis)
- Dominik Vahrenhorst
    - first approach to combine Cm1 and Cm2 via a system tray icon

Cm1.1 CEEE

- Marco Civico and Roland Orlik
    - concept, test scenarios, evaluation at the Niklas Luhmann Gymnasium in Oerlinghausen
- Christina Kropp

  - cell component modeling with Amira
- Nils Rothe
  - cell component modeling with Blender
- Jonas Osterloff
  - Tour Editor
- Samy Slaih and Pascal Witthus
  - Information Viewer with cell component HTML documents

Cm2.0 (this version was completely substituted by the new version)

- Sebastian Janowski and Dominik Mertens
  - PDB import/export, XML Parser
- Robert Schmieder
  - membrane generation, data model
- Jan Pieniak and Björn Sommer
  - membrane preview with Jmol integration
- Martha Zakrzewski
  - GUI

Cm2.1

- Andrea Steinmetz
  - generation of HTML documentation, biological background, testing and bug reporting
- Christian Fink and Matthias Koch
  - improvement of PDB parser, computation of Protein/Lipid Sizes
- Sebastian Schneider
  - membrane preview improvement, improvement of lipid/protein visualization
- Tim Dingersen and Ralf Mertens
  - framework programming, improvement of XML structure, GUI, Jmol integration improvement, improvement of visualization, improvement of usability

Cm2.2.1

- Sebastian Schneider
  - first version of the Plugin-algorithm interface
  - Lipid Packing Algorithm: Simulated Annealing algorithm
- Tim Dingersen
  - complete re-implementation of the Cm2.0, refactoring to Cm2.1
  - programming during the entire Cm2.1rev period of MembraneEditor development as student research assistant of Björn Sommer
  - Lipid Packing Algorithms

    - Linear Placing, Random Placing, Distributor, Advanced Random Placing
  - integration of atomic visualization as well as simple molecular simulation techniques
    - Lipid Packing Algorithm: Atom Level Minimizer
  - reverse-parsing of combined PDB/XML membranes
- Sebastian Rubert
  - implementation of lipid distribution precision improvement of up to three decimal places
- Christian Gamroth
  - PDBTM integration and improvement in 2012
  - improvement of PDB exporter, reduction of required heap space
  - 3D Stereoscopy Update: live update during navigation based on orbit behavior

Cm2.2 MD Edition (not discussed in this thesis)

- Christian Gamroth and Sebastian Rubert
  - GMX-Plugin (connecting Cm2.2 to GROMACS via ssh and UNICORE)
- André Heißmann and Ralf Rotzoll
  - all-atom lipid library and MD simulation with GROMACS (ffG45A3)
- Gunther Lukat and Alexander Schäfer
  - coarse-grained lipid library and MD simulation with GROMACS (MARTINI)

Cm2.2 Vesicle Builder (not discussed in this thesis)

- Yan Zhou
  - first 3D Lipid Packing Algorithm for Cm2.2

Cm3.0

- Chan Lin and Xiaoliang Song
  - XML, basic navigation techniques
- Olga Mantler
  - cell component distribution
- Ufuk Aydin
  - GUI programming

Cm3.1

- Christina Ander and Regina Bisdorf
  - cell component distribution, 2nd approach
- Florian Heißenberg
  - cell component modeling with Blender
- Jörn Winnebald
  - XML-based configuration

- Lukas Jelonek
  - 3D Stereoscopy integration
- Philipp Unruh
  - 1st refactoring

Cm3.1rev

- Yvonne Dyck (not discussed in this thesis)
  - improved cell component distribution algorithm
- Raphael-Elias Reisch (not discussed in this thesis)
  - audio support

Cm4.0

- Arne Husemann
  - the first node random distribution algorithm
  - the first node mapping approach
- Madis Rumming
  - the pathway table
  - Node Details window
- Sebastian Janowski
  - 1st proprietary KEGG integration from BioDWH

Cm4.1/Cm4.1rev

- Özgür Ates
  - SBML Level 2 Version 3 Export and Import
- Pavel S. Demenkov
  - ANDCell 2012 Database for PubMed-based Protein Localization
- Klaus Hippe
  - the data warehouse collection connected to CmPI: the independent project Dawis M.D.
- Benjamin Kormeier
  - the base for the data warehouse connected to CmPI: the independent project BioDWH
- Jörn Künsemöller and Norbert Sand
  - 6DOF CELLmicrocosmos navigation
- Pascal Witthus (not discussed in this thesis)
  - Layout Algorithms
    - Sphere Mapping Layout, Spherical Segment Layout (mapping of 2D layouts onto cell components)
    - FR3D layout (Fruchterman-Reingold algorithm)

Cm5/Cm3.2 (not discussed in this thesis)

- Rudolf Warkentin
  - Cm5.0/5.1: Cell Component Builder, supporting the generation of CmCX-compatible cell component VRML97 files
  - Cm3.2: the new stand-alone CellEditor (ongoing)

VipVap

- Marion Zysik
  - 1st Membrane Mapping Approach using a stand-alone software based on C++/VTK, using input files from the Cm3.0 project

9.17.3 Cell Components and their Authors

The nanometers included in the file titles refer to the maximum diameter of a cell component. High and low quality versions of the models are not listed here.

- Björn Sommer
  - Abstraction Level 1 (3ds Max, Figure 81.3, 89, 107, 109, 122, 148)
    - mod_10000nm_unknown_spheres.wrl
    - mod_1000nm_nucleus_spheres.wrl
    - mod_1500nm_granular_er_spheres.wrl
    - mod_2000nm_ribosome_spheres.wrl
    - mod_2500nm_agranular_spheres.wrl
    - mod_3000_golgi_spheres.wrl
    - mod_3500nm_lysosome_spheres.wrl
    - mod_4000nm_mitochondrium_spheres.wrl
    - mod_4500nm_chloroplast_spheres.wrl
    - mod_5000nm_peroxisome_spheres.wrl
    - mod_500nm_nucleolus_spheres.wrl
    - mod_5500nm_vesicle_spheres.wrl
    - mod_6000nm_cytosol_spheres.wrl
    - mod_6500nm_vacuole_spheres.wrl
    - mod_7000nm_endosome_spheres.wrl
    - mod_8000nm_cellmembrane_spheres.wrl
    - mod_8500nm_cellwall_spheres.wrl
    - mod_9000nm_extracellular_spheres.wrl
    - mod_ribosome_23nm.wrl
    - mod_unknown_2000nm.wrl
    - mod_unknown_600nm.wrl
  - Abstraction Level 2 (3ds Max)

- mod_agranular_e_r_3700x2000nm.wrl (Figure 93, 114, 119, 120, 124, 125, 126, 150, 158, 166, 167)
- mod_cellmembrane_18000nm.wrl
- mod_cellmembrane_27000nm.wrl
- mod_cellmembrane_9000nm.wrl (Figure 93, 118, 120, 124, 125, 126, 150, 158, 164, 166, 167)
- mod_cellmembrane_plant_13000nm.wrl (Figure 120, 124, 125, 126)
- mod_cellwall_plant_14500nm.WRL (Figure 120, 124, 125, 126)
- mod_cytosol_6000nm.wrl (Figure 149, 150, 164, 167)
- mod_endosome_180nm.WRL
- mod_extracellular_matrix_10000.WRL
- mod_extracellular_matrix_plant_17000.WRL
- mod_golgi_CTL_ImmuneSynapes_green_500nm.wrl
- mod_granular_e_r_5000nm.wrl (Figure 120, 124, 125, 126, 150, 150, 158, 166, 167)
- mod_lysosome_100nm.WRL
- mod_mitochondrium_1600x550nm.wrl (Figure 81.2, 93, 120, 124, 125, 126)
- mod_nucleolus_500nm.wrl (Figure 120, 124, 125, 126, 150)
- mod_nucleus_3500nm.wrl (Figure 79, 88.2, 93, 114, 120, 124, 125, 126, 150, 166, 167)
- mod_vacuole_1800nm.wrl (Figure 120, 124, 125, 126)
- mod_vesicle_180nm.wrl (Figure 51, 118, 120, 124, 125, 126, 158)

- Abstraction Level 3 (Amira, partly 3ds Max)
  - mod_mitochondrium_rudyAtrace_1400nm.wrl (Amira and 3ds Max, Figure 81.1, 82.2, 85, 86, 90, 107, 149, 150, 167)
  - mod_mitochondrium_segment_cone_sub_2_82slices.wrl (Figure 82.1, 83, 84)

- Florian Heißenberg (Blender)
  - Abstraction Level 2
    - mod_chloroplast_2000nm.wrl (Figure 120)
    - mod_golgi_apparatus_950nm.wrl (Figure 114, 118, 158, 164, 166, 167)
    - mod_mitochondrium_1600X1000nm.wrl (Figure 114, 118, 164)
    - mod_peroxisome_200nm.wrl (Figure 150)
    - mod_nucleus_vertexcolors_triangulated_2000nm.wrl
- Christina Kropp

  - Abstraction Level 3 (Amira, CELLmicrocosmos 5: Component Builder)
    - mod_chloroplast_phaeo1_segment_1600nm.wrl
- Nils Rothe
  - Abstraction Level 2 (Blender) (Figure 121)
    - mod_capsule_4600nm.wrl
    - mod_cellwall_prokaryote_1wall.wrl
    - mod_cellwall_prokaryote_2walls.wrl
    - mod_chloroplast_2000nm.WRL
    - mod_cytoplasmicmembrane_4000nm.wrl
    - mod_cytosol_prokaryote_3300nm.wrl
    - mod_extracellular_prokaryote_6000nm.wrl
    - mod_nucleoid_1300nm.wrl
    - mod_plasmids_1900nm.wrl
    - mod_ribosomes_prokaryote_1200nm.wrl
    - mod_storagegranule_560nm.wrl

9.18 Used Programs

For this work, the following external programs were used:

- (Discreet/Autodesk) 3ds Max 7
- (Mercury) Amira 4
- (PBSoft) ANDVisio
- Blender
- (Adobe) Photoshop
- GIMP
- GROMACS
- Linux Mint 12, Ubuntu 10.04 LTS
- Open Office 3/Libre Office 3
- (Bio Medical Informatics Department) VANESA
- (Microsoft) Windows XP, 7
- Xmgrace
- Zotero

9.19 Complete Comparison Tables

9.19.1 Cell Modeling Tools

Category	Feature	AutoPack/AutoCell[92]	DisplayCell (Amira)	LifeExplorer Project	Meta!Blast	CompuCell3D	E-Cell (3D)	ENVIRONEMENT[92]	VCell	CmCX
Availability	Plugin	X	X							
	Standalone			X	X	X	X	X		X
	Web Start								X	X
	Source Code					X	X		X	(X)
	Licenses					Open Source	GPL		MIT	(GPL3)
Computational Acceleration	Multithreading					X	X			
	Parallelization					X	X		X	
Dimensions	2D					X	X		X	X
	3D	X		X	X	X	3D		X	X
	Time (incl. 4D)				(X)	X	X		X	
Formats	AVS								X	
	CellML								X	
	DICOM					X				
	Matlab								X	
	native format	X				PIF	X		XML, VCML, VFRAP	Cm3
	PLY								X	
	SBML								X	X
	STL								X	
	TIFF					X				
	VRML	X								X
Granularity	Atomistic									X
	Molecular	X								X

92 Because these software tools are not completely available as of yet (31.07.2012), the information in this table are given with reservations and are perhaps partly fragmentary.

Category	Feature	AutoPack/AutoCell	DisplayCell (Amira)	LifeExplorer Project	Meta!Blast	CompuCell3D	E-Cell (3D)	ENVIRONEMENT[92]	VCell	CmCX
	Intra-Compartmental	X			X	X		X	X	X
	Subcellular	X			X	X			X	X
	Organisms				X					
Interactivity	3D Navigation	X		X						X
	Dynamic Tables								X	X
	Links to data sources								X	X
	focus and context								X	X
	Sorting Tables								X	X
Modeling	abstraction level, cell components	1/2	2	1	2	3			(1/2/)3[93]	1/2/3
	abstract models	X				X			X	X
	mathematical					X	X	X	X	
	microscopic models	X				X			X	X
	node distribution algorithms								2D	CmPI
Operating System	Linux	X				X	Version 1-3		X	X
	Mac OS X	X			X	X	3D		X	X
	Windows XP, VISTA, 7	X		X	X	X	Version 1-3 (executable v. 2)		X	X
Programming	Scripting	X				X	X		X	
	Scripting at Runtime								X	
Visualization	2D Mapping									CmPI

93 ABS_LEVEL_1/2 are only available via data import in VCell since version 5.1 Beta.

Category	Feature	AutoPack/AutoCell	DisplayCell (Amira)	LifeExplorer Project	Meta!Blast	CompuCell3D	E-Cell (3D)	ENVIRONEMENT[92]	VCell	CmCX
	2D Visualization					X	X		X	
	3D Visualization	X		X	X	X	X		X	X
	3D Visualization: Live-Animation			X		X				
	3D Visualization: Geometric	X		X	X		X		X	X
	3D Visualization: Volumetric									
	Graph Visualizations						X		X	
	Internal Graph Mapping									
	Marking of POI			X					X	
	Molecular Visualization	X								X
	Primary Structure View			X						X
	Secondary Structure View									
	Raytracing	X								
	Runtime Graphs								X	
	Statistics					X	X		X	X
	Stereo Support									X

Table 24: The complete comparison of all cell modeling tools discussed in this work

9.19.2 Membrane Modeling Tools

Category	Feature	Charmm GUI	ChemSW	MOE+ Script	Packmol	VMD MP	CmME
Availability	Standalone		X	X		X	X
	Command line tool				X		
	Web service						
	Website	X					
	Web Start						X
	Source Code				X	X	X
	Licenses (educational license fee)		comm. (~US$ 210)	comm. (~US$ 2,300)	GPL3	UIUC Open Source	GPL3
Computational Acceleration	Multithreading						
	Parallelization				X		
Direct Database Connections	PDB database	X		X			X
	PDBTM database						X
	OPM database	X					X
Formats	GRO					X	
	native format		X	X			X
	PDB	X	X	X	X	X	X
Libraries	Lipid library (>2 lipid types)	X	X				X
	Lipid library compatible to MD	X	X				(GMX)
Modeling	Atom-based Molecule Editor		X	X		X	X
	Bilayers	X	X	X	X	X	X
	Collision-Detection				X		X
	Counterions support	X					(GMX)
	Lipid Packing Density	X					X
	Heterogeneity	X	(X)		X		X
	Hexagonal Shape	X					
	Relative Lipid Ratio	X					X
	Monolayers		X	X	X		X
	Multilayers				X		X
	Percentaged Distribution	X	X				X

Category	Feature	Charmm GUI	ChemSW	MOE+ Script	Packmol	VMD MP	CmME
	Protocol						X
	Raft Support				X		X
	Rectangular Shape	X	X	X	X	X	X
	Reproducibility						X
	Total lipid number	X		X	X	X	X
	Water Layer Build	X	X	X	X		(GMX)
Operating System	Linux	X		X	X	X	X
	Mac OS X	X		X	X	X	X
	Windows XP, VISTA, 7	X	X	X	X	X	X
Pipelines	Ext. Simulation Package	X	X	X		X	(GMX)
	Ext. Visualization Package	X	X			X	(Jmol)
Programming	Scripting			X	X	X	X
	Scripting at Runtime					X	X
Protein Placement	Automatic Protein-Placement using OPM	X					X
	Manual Placement			X	X		X
	Semi-automatic Protein Placement						X
Simulation	Equilibration	(X)		X	X	X	(GMX)
	Minimization	(X)	X	X		X	(GMX)
	Simulation		X	X		X	(GMX)
Visualization	Atomic Structure View	X	X	X		X	X
	Graph Visualizations			X			X
	Live Distribution Visualization						X
	Secondary Structure View			X		X	(Jmol)
	Raytracing		x				
	Runtime Graphs						X
	Stereo Support			X		X	X

Table 25: The complete comparison of all membrane modeling tools discussed in this work

9.19.3 Network Modeling Tools

Category	Feature	3DScape+Cytoscape	Arena 3D	BioCichlid	GEOMI	The Interactorium,	MetNetVR	Metab. Netw. Visualizer	VANTED HIVE+	WilmaScope	CmPI
Availability	Plugin	X							X		
	Standalone			X	X	X	X[94]			X	
	Web Service										
	Website Applet							(X)[95]			
	Web Start		X						X		X
	Source Code		X	X					X	X	(X)
	Licenses	LGPL	Acad. Free	Acad .Free	LGPL	Acad .Free			GPL2	LGPL	(GPL 3)
Computational Acceleration	Multithreading										
	Parallelization										
Dimensions	2D	X			X				X		
	2.5D		X	X					X	X	
	3D	(X)			X	X	X	X	(X)	(X)	X
	Time (incl. 4D)		X	X							
Direct Database Connections	ANDCell										X
	BRENDA										X
	Gene Ontology										X
	KEGG								X		X
	OMIM										
	PDB										
	Reactome										X
	Transfac										
	Transpath										
	UniProt										X
Formats	ANDVisio										X
	CSV	X							X		X
	native format	Cytos	X		XWG				X		Cm4

94 The software MetNetVR is not downloadable, but parts of the website are still on-line (as of 31.07.2012).
95 The Metabolic Network Visualizer as well as its website is no longer available (as of 31.07.2012).

Category	Feature	3DScape+Cytoscape	Arena 3D	BioCichlid	GEOMI	The Interactorium,	MetNetVR	Metab. Netw. Visualizer	VANTED HIVE+	WilmaScope	CmPI
		cape formats									
	PDB					X					
	SBML	X	X						X		X
	VRML							X			X
Granularity	Atomistic										
	Molecular					X					
	Intra-Compartmental										X
	Subcellular	X			X	X	X	X	X		X
	Organisms								X		
Interactivity	3D Navigation	X			X	X	X		X		X
	Detail-on-Demand					X	X		X		
	Dynamic Tables										X
	Links to data sources								X		X
	Focus + Context	X						X	X		X
	Sorting Tables										X
Layouts	Clustering	X								X	(X)
	Fan						X				
	FM3	X									
	Force Directed	X	X		X	X		X	X	X	X
	GEM-3D						X				
	ISOM										X
	Radial						X		X		
	Span Tree	X									
Localizations	different localizations for one molecule				X	X					X
	datawarehouse results										X
	Experimental data localization	X							X		

Category	Feature	3DScape+Cytoscape	Arena 3D	BioCichlid	GEOMI	The Interactorium,	MetNetVR	Metab. Netw. Visualizer	VANTED HIVE+	WilmaScope	CmPI
	text mining results										X
	manual prediction										X
	re-localization										X
Modeling	cell component abstraction level	2				3	3		1		1/2/3
	abstract models	X									X
	microscopic models								X		X
	different pathways at once		X	X							X
	node distribution algorithms		X	X			X	X		X	X
Operating System	Linux	X	X		X	X		X	X	X	X
	Mac OS X	X	X	X	X	X		X	X		X
	Windows XP, VISTA, 7	X	X		X	X		X	X	X	X
Pathways	Metabolism	X	X					X	X	X	X
	Protein-Protein Interaction	X	X	X	X	X			X	X	X
	Signaling	X	X						X	X	
Programming	Scripting										
	Scripting at Runtime										
Protein Placement	Automatic Protein-Placement using OPM										
	Automatic Grouping of Protein Complexes				X	X					
	Manual Placement										
	Semi-automatic Protein Placement										
Visualization	2D Mapping										X
	2D Visualization	X							X		X
	CAVE Support						X				
	Geometric 3D	X			X		X				X

Category	Feature	3DScape+Cytoscape	Arena 3D	BioCichlid	GEOMI	The Interactorium,	MetNetVR	Metab. Netw. Visualizer	VANTED HIVE+	WilmaScope	CmPI
	Graph Visualizations	X	X	X	X	X	X	X	X	X	X
	Internal Graph Mapping										X
	Localization-dependent Node Coloring				X						X
	Marking of POI	X			X		X		X		X
	Molecular Visualization					X					
	Primary Structure View										
	Secondary Structure View					X					
	Raytracing										
	Runtime Graphs										
	Statistics								X		
	Stereo Support						X		X		X
	Volumetric 3D								X		

Table 26: The complete comparison of all network modeling tools discussed in this work