Artificial Evolutionary Development

TILL STEINER

Dissertation

A thesis presented to the
Technische Fakultät of the Universität Bielefeld
in partial fulfillment of the requirements
for the degree of
Doctor rerum naturalis

October 2010

Printed on non-aging paper (DIN EN ISO 9706).

Acknowledgements

Working at the Honda Research Institute was an exciting opportunity for me. Not many PhD-students have the chance to start their scientific career as an employee of a big international enterprise. In this privileged and unusual setting, I began to explore the somewhat exotic topic of computational evolutionary development back in 2006. I was acquainted with the basics of this scientific field from my previous internship at Honda, and felt at home in the institute. Provided with exceptional supervision and funding, it was the ideal starting point for my PhD. Who else can say that they are sent to Singapore for a short research project, just because they would enjoy such an experience and agree to pursue a PhD afterwards? Not to mention being allowed several conference visits per year throughout the course of my PhD... Additionally, I had the opportunity to take a peek into the organization of an industrial research institute. I am starting to see the great value of having had this experience now. A very close and amicable contact with my supervisor not only shaped my scientific approach, but also my general image of what a good working environment should be, as well as my future professional plans.

That said, it seemed that the general concept of 'no free lunch'* (ironically, fundamental to my scientific field), did not apply to me at that time. However, a severe disadvantage showed up toward the end of my PhD in 2009: the global crisis which badly affected the financial situation of the automobile industry coincided with my negotiations for an extension of my contract. In addition, this had a huge effect on the scientific and human resource strategy of the institute, and on its general working atmosphere. To witness and to be part of a complete re-orientation of a small division within a global enterprise is, in my opinion, one of the most important economic lessons to learn. Luckily I was affected by such an event at a formative stage of my career.

The final phase of my PhD was governed by re-orientation. Despite the serious situation, the institute *was* able to offer me an extension of my contract. Nevertheless, I decided to leave Honda to search for a new experience in a smaller, medium-sized business. In retrospect, I am very grateful to the management board of the Honda Research Institute for the offer. They certainly had to exert their influence far beyond what I was aware of at the time.

Many people have supported me doing scientific work. The relation between their effort and impact is naturally non-linear. In one case, however, it is very clear. Thank you, Bernhard, for all the time you spent with me in innumerable meetings and your sympathy for the sometimes unusual problems of a PhD student. I will certainly follow your ideals on my future professional and personal path.

*somehow, every advantage has to be paid for by a disadvantage in another area

Thank you Yaochu, for teaching me the crafts of science: studies, statistical analysis, condensation of the information and publication of the results would all have been much harder without your advice.

Thank you, former colleagues at the HRI, especially Markus, Stefan, Lars, Martina, Giles, Mathias, Nils, Sven, Sven, David, and of course my 'room-mate' Lisa, for making every day a fun day at the institute! (Also, for scientific discussions, of course...)

I want to thank the University of Bielefeld for accepting this document as an external PhD thesis. Especially, I am grateful to the university part of the scientific committee: Prof.-Dr. Helge Ritter, Prof.-Dr. Tim Nattkemper, and Dr. Thomas Hermann.

I thank my family for all their support during the PhD time, and throughout my life. Kristin, thank you for all the love and having been there whenever I needed you.

Contents

1. Introduction

> *For the real amazement, if you wish to be amazed, is this process. You start out as a single cell derived from the coupling of a sperm and an egg; this divides in two, then four, then eight and so on, and at a certain stage there emerges a single cell which has as all its progeny the human brain. The mere existence of such a cell should be one of the great astonishments of the earth. People ought to be walking around all day, all through their working hours calling to each other in endless wonderment, talking of nothing except that cell.* Lewis Thomas (1979)*

In the late 19th century, a group of biologists led by Edmund Beecher Wilson and Thomas Hunt Morgan began to investigate the process of multicellular development in fertilized eggs. They quickly found this process must have an enormous complexity to give rise to creatures equipped with impressive characteristics. Among these characteristics, which even nowadays are extraordinary to the extent that they are rarely found in man made designs, are the ability to regenerate, and to be robust in reaction to errors in their DNA, that is, the building process blueprints. Furthermore, biology equips individuals with adaptivity to a wide range of environmental conditions, and the capability to be constantly improved through random mutations and selection, despite their high complexity.

In contrast, technical devices engineered by humans usually lack these characteristics, even though they are much sought after. From an engineering point of view, designing devices with more of these biological features is a challenge, since such designs would necessitate a complex, internal monitoring system, and an adaptive building substrate that allows for flexible resource allocation. At the same time, small random alterations to the construction process must not disrupt overall development, even though many processes would have to be interwoven intricately.

Fortunately, more and more details of biochemical and biophysical processes underlying development have been discovered recently. Bioinformatic tools enable scientists to integrate a vast amount of data on biological development, which in turn yield first insights into the overall organization and interplay of the functional parts of the process. This increasing insight into development is starting to allow an abstract, more technical point of view on the biological process. If it is possible to abstract biological development, and adapt these new concepts to an engineering design problem, we could gain a novel engineering paradigm, creating artifacts with capabilities beyond those of present-day designs.

*Thomas, L. (1979). "On Embryology." In *The Medusa and the Snail,* Viking Press, New York, p. 157.

Thus, we have to ask ourselves, how to mimic biological development to gain such artifacts. The young research field of Artificial Development is devoted to this question. Artificial Development scientists build computer simulations of biological growth processes and use them to gain further insight into developmental design. The target is to create novel approaches toward solving engineering problems of many different kinds, such as efficient code for computer programs, flexible mechanical design or integrated circuits.

For all these endeavors, it is crucial to abstract biological development prudently. Too few details would limit the capabilities of the method, while too many details would be too expensive to model, and too difficult to adapt to an engineering problem. Usually, computational development models are coupled to an evolutionary optimization method to autonomously find this adaptation to engineering problems, which eventually allows the development of a device with the desired features. On one hand, this is an elegant solution to achieve simulated developmental processes that go beyond human design. On the other hand, depending on the model of development, Artificial Development can result in very complex solutions. This means that analysis and understanding of the resulting design process can be extremely difficult.

This thesis is situated in the field of Artificial Development coupled to evolutionary computation. I discuss the problem of finding a *suitable* abstraction level for the developmental process in engineering design. Here, suitable refers to the capability to produce non-trivial artifacts, while keeping the developmental process and its formation comprehensible. Throughout this thesis, I distinguish between two components of development: The first component is the dynamical control of the developmental process. The second component is the cellular representation employed within development. I study both components individually, as well as coupled together as a complete Artificial Development system in an evolutionary context.

The basis for my investigation of simulated evolutionary development is an efficient implementation of the process. I have created a novel artificial development simulation environment for this purpose. It is designed to interface with a real valued evolution strategy, and comprises a gene regulatory network simulation for control of the development, 3D multi-body physics simulation, an implementation of different cellular representations, as well as chemical diffusion simulation and mechanical evaluation of the resulting designs. Using the simulation environment, I have investigated evolution of graphs for the control of stable multicellular development. Analysis of the evolution of a negative feedback motif and its role in both individual developmental time scale and evolutionary time scale gives insight into the problems arising from the use of evolution to design graphs as a representation for dynamical systems. Following this investigation, a cellular representation employing differential cellular forces is presented as the basis for structural design experiments. I have amended the biologically inspired properties of polarization and chemotaxis to this model, to increase its representational capacities. Both methods are evaluated on a mechanical stability design problem, and compared to state of the art approaches in this field. This comparison, as well as the negative feedback motif analysis, are the foundation of a criticism of graph representations. Graphs are state of the art for representing developmental control in

an evolutionary context. I propose an alternative approach toward modeling dynamics for evolutionary development. This approach is called Vector Field Embryogeny and relies on phase space modeling of dynamics, rather than graphs. The representation is thoroughly investigated in an evolutionary context and shows better performance for simple benchmark problems as compared to graph approaches. The reason for this advantage is investigated in terms of causality of the genotype- to phenotype-map.

This thesis is structured as follows: Chapter 2 introduces biological embryogenesis as paradigm for Artificial Development. Both, the general process and an evolutionary perspective on it are presented. In Chapter 3, I review related work in Artificial Development and describe the simulation environment, which is the basis for the investigations presented in Chapters 4, 5, and 6. Chapter 4 presents the investigations into graph-based dynamical representations, while Chapter 5 describes my research on Vector Field Embryogeny. In Chapter 6, I work out the details of the causality investigations for Vector Field Embryogeny and the graph-based approach. Finally, in Chapter 7, I conclude the thesis with a summary and discussion.

2. The Paradigm: Embryogenesis – The Early Development of Organisms in Biology

2.1. Introduction

Multicellular organisms evolved to survive and reproduce successfully in a complex environment. In biology, a sophisticated crossover, in which two single cells of parent individuals fuse, necessitates a process of growth to create functional, adult offspring. On the following pages, I will present an overview over the early phase of biological development. This phase is called *embryogenesis*, and I will emphasize principles, that are derived from processes found in a large number of species.

The development of an individual organism exhibits certain features typical for its species. However, besides that, biologists find a great degree of similarities, which are conserved in principle throughout many distinct species. Different embryogenies exhibit similarities on many levels, such as gene regulation, cellular activity, and organic growth, all temporally organized in similar embryonic stages, and allow for a connection between creatures as unequal as chimps and fruit flies. In the first part of the chapter, I will concentrate on the description of these biological principles.

Coupling the analysis of evolution with research on development is a relatively young field of science termed *evolutionary development*, or short, *evo-devo*. In this field, not only the historic background of developmental processes is paid attention to. Rather, the mutual influence of evolution and development is researched. Genetic mutations primarily cause alterations in the growth process of an individual. Since such a changed development must still yield a viable organism in order to transfer the genetic alteration to the next generation, only genetic changes that 'fit' into the existing development, can be passed on. By similar means, developmental processes evolved, which allow a high plasticity in phenotypic traits. Development allows evolution to 'tinker' wisely. Thus, developmental processes strongly influence the evolutionary path of species. The second part of this chapter is dedicated to describing principles found in this field of *evo-devo* in appropriate detail.

All information I am going to present in this chapter is summarized from the standard literature (e.g. [1, 2, 3, 4]), where the interested reader can find abundant details on the interplay of genes and cells during embryogenesis in the light of evolution.

2.2. Building a Multicellular Organism

2.2.1. An Overview

The development of a multicellular organism starts from a single fertilized egg cell, the *zygote*. The process of growth is organized as a series of cellular divisions and coordinated cell actions, in which the information stored on the DNA (deoxyribonucleic acid) is extracted and used to form a functional individual. Embryogenesis stands for the early phase of this development, from the zygote to the end of the embryonic stage, which is marked by birth of an individual or a transition into the larva stage.

Two abstract processes govern embryogenesis: cellular *proliferation*, and *differentiation*. Proliferation is the provision of cellular material, i.e. the creation of new cells through cell divisions, which are the basis to physically shape the organism. Cellular differentiation is the way how a cell, in comparison to others, changes its role and specializes to perform a distinct function. In this context, both the position of the differentiating cell, as well as the type it differentiates into, need to be coordinated, locally in the cellular neighborhood, as well as globally with respect to the whole embryo. To achieve this coordination, biology relies on cellular communication. Cellular communication is based on an exchange of physical and chemical signals, which can take place in a variety of ways. Section 2.2.3 is dedicated to the description of these means of cellular information exchange and processing. We will see how both, self organization and hierarchical control are employed, to create robust growth.

Leaving the cellular level and assuming a more abstract, systematic viewpoint, we observe a striking similarity in the temporal pattern of embryogenesis throughout animal species. Most animals exhibit an early development that follows a well defined sequence of embryonic stages. Each of them has characteristic features and a clearly assignable function for the successful growth of the individual.

In the following, I will give more details on the aforementioned processes and patterns of embryogenesis on different levels of abstraction. I start by presenting general background on genetics, which is the molecular basis for embryogenesis. Then, communication on the cellular level will be discussed. I will conclude with a macroscopic description of the embryonic stages and their function in a successful development.

2.2.2. Genes and Differential Gene Expression During Embryogenesis

The DNA encodes all information necessary for the construction and maintenance of the organism. Hence, it can be seen as a blueprint, which encodes a temporal and spatial sequence of events that take place during development. Each cell in a multicellular organism contains a copy of the same DNA, such that the information is present and can be processed in a parallel manner by each cell separately. The DNA is structured

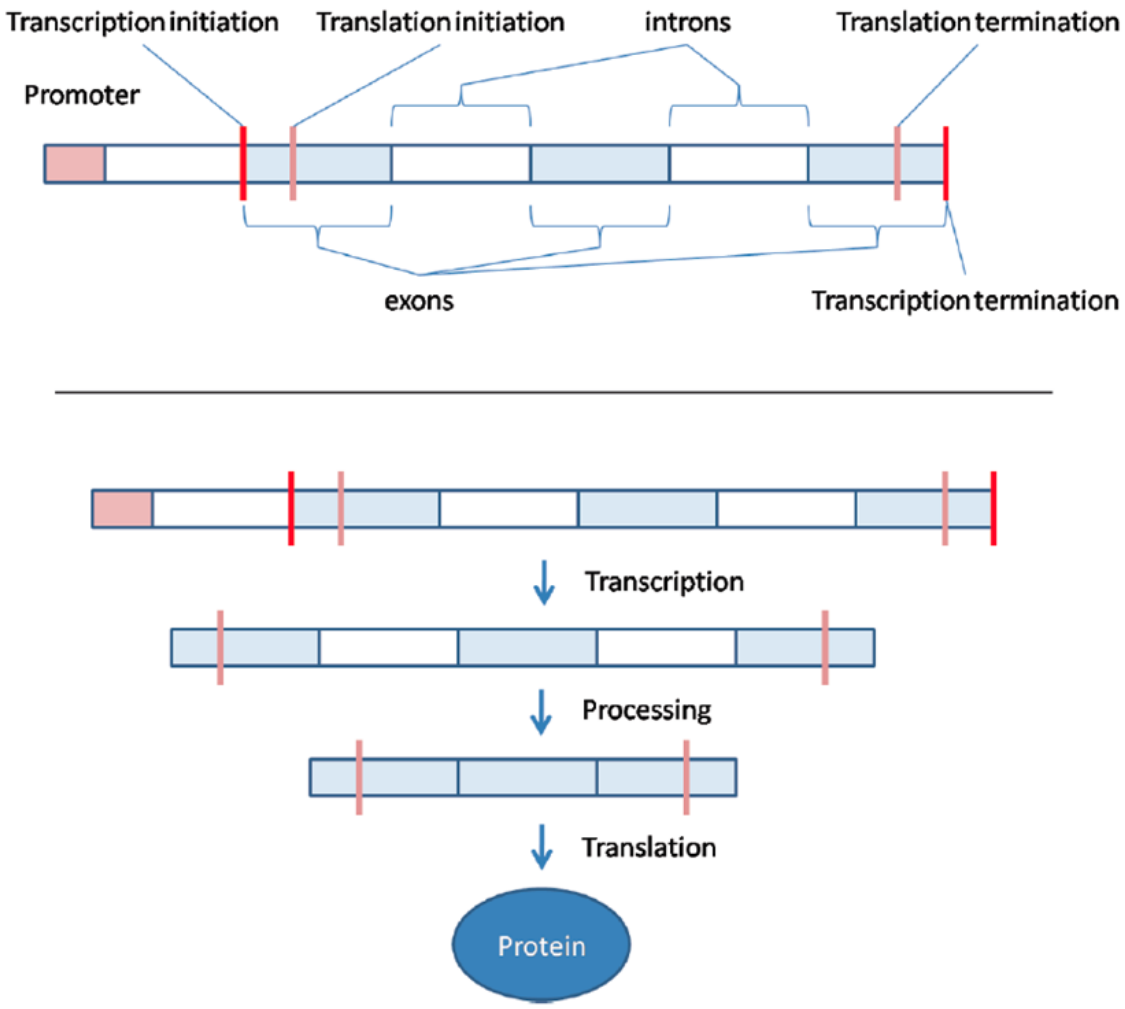


Figure 2.1.: A schematic representation of the process of translating a gene into a protein. The upper panel depicts the structure of a typical eukaryotic gene, the lower panel gives a simplified sequence of the process. More information is given in the text.

into genes, which, in an eukaryotic organism, are typically composed of five distinct functional regions (see Figure 2.1):

1. a promoter region,
2. the transcription initiation site,
3. the translation initiation site,
4. a sequence of exons and introns,
5. a translation termination site and
6. a transcription termination site.

The process of using the information encoded on the DNA starts with the transcription of genetic information (first stel in the lower panel of Figure 2.1). The promoter region is responsible for the initiation of the transcription process. The RNA polymerase, an enzyme which transcribes the genetic information on the DNA into RNA (ribonucleic acid) so that it can be further processed, binds to the DNA at the promoter region. The transcription process is then started at the transcription initiation site. The RNA polymerase shifts along the DNA and copies the information into an RNA strand known as nuclear RNA (nRNA). This strand of RNA is then further processed to remove certain regions referred to as introns, and re-align the remaining regions (exons), a process known as splicing. The resulting RNA, called messenger RNA (mRNA) is then transported to, and translated on the ribosomes. Ribosomes are complexes of RNA and proteins, which specifically detect nucleotide triplets in the mRNA and accordingly align a sequence of amino acids, which are provided by the tRNA. The sequence of amino acids then self-arranges into a three dimensional structure (folding), forming a protein. This protein serves as a basic functional unit for the cell. For example, it can become part of the cell membrane or form complexes with other proteins. It can also serve as a signal, which means that it gets involved in the transcription process of genes as explained as follows.

The RNA polymerase needs special proteins to be able to bind to the promoter region of a gene to begin transcription. These proteins are called transcription factors (TFs). There are specific DNA sites where they can bind to at the promoter. These sites can be distinguished into enhancer sites and silencer sites. Enhancer sites attract those TFs, which ease the binding of RNA polymerase to the DNA. If such a TF is present, the transcription process can be initiated. A silencer can be seen as an inverse enhancer: If the TF is present, the probability of transcription is reduced. The TFs that bind to enhancers or silencers are usually gene products themselves. Therefore, a regulatory link can exist between genes: the product of one gene enhances or silences the activity of another gene, which in turn can regulate the activity of another gene and so on. The resulting interaction network can be depicted as a graph, where genes are represented as nodes and TF-promoter interactions are depicted as the edges. The graph naturally consists of two different types of unidirectional link: activation and inhibition, depending on the TF and its binding site. A comprehensive description of gene expression and regulation can be found in [2].

The aforementioned graphs are called gene regulatory networks (GRNs). These GRNs represent dynamical systems, which control the spatial and temporal pattern of gene expression in an embryo and thereby the whole growth process. To structure an embryo, it is necessary to have cells at certain positions behave differently from other, surrounding cells. Since genes are responsible for cellular activity, this implies that the activity of genes must differ in distinct cells, a state that is termed *differential gene expression*.

Since all cells in an embryo originate from a single zygote, organized differential gene expression necessitates an initial breaking of symmetry. Otherwise, a cellular division would yield two identical daughter cells, and development would eventually result in a colony of identical cells, rather than an organism with differentiated parts. Symmetry breaking can be the result of at least three different causes in biology:

- Non-symmetric cleavage: if a cell divides in a way that one daughter cell has a different concentration of TFs than the other daughter cell, a symmetry breaking has occurred. Subsequently, genetic activation can differ in the two cells.
- Random fluctuations: small fluctuations in the intra- and extracellular concentrations of chemical substances can trigger differential gene expression in neighboring cells.
- (Initial) external conditions: defined external conditions, such as chemical gradients provided by the mother individual, or the point of entry of the sperm into the egg, can provide positional information. Chemical gradients that are produced by the mother are called *maternal gradients* and found to play a role in body axis determination, i.e. symmetry breaking.

In this light, the role of the GRN is to allow for coordinated differential gene expression once a symmetry breaking has occurred. However, a secondary function, directly related to biological evolution, can be attributed to the GRN: its complex dynamics also allow to keep development robust yet flexible toward genetic and environmental changes. Many genes of biological GRNs seem to provide to this secondary function [5]. I will present investigations into these evolutionary implications of GRNs in Chapter 4.

2.2.3. Cellular Communication

Cellular communication is a key component of coordinated multicellular development. In biology, this communication occurs in terms of *induction*: one cell sends out a signal, and induces a reaction in terms of behavioral change in other cells. Examples for such changes are e.g. an altered cell shape, or a new developmental fate. Developmental biologists distinguish between two communication roles in chemical signaling: *inducing cells* and *competent cells*. The inducing cell produces chemical signals that can be sensed and reacted to specifically by a competent cell. This implies that not all signals are reacted to equally by all cells. In fact, a cell must be in a certain state to sense the respective signal at all. The receiving cell must be competent, otherwise the signal is not interpreted.

Chemical signaling occurs at different distances in the growing embryo. Thus, three types of signal transduction are distinguished, defined by the distance their signals are allowed to travel. The spatially shortest communicative interaction is called *juxtacrine* signaling. Juxtacrine interactions occur between directly neighbored cells via proteins that are attached to the cell surfaces. On the other hand, *paracrine* interactions take place between cells that have a short distance. It is achieved via short range diffusion of TF-like chemicals. Finally, *endocrine* interaction describes the communication through hormones that are transported through the blood and thus span over longer distances.

In summary, the different kinds of communication allow for coordinated cell actions both, in localized regions, and across the embryo. Induction allows directed flow of information, competence renders selective interpretation of this information possible.

From a systematic point of view, the coordination of development can be achieved with two distinct methods. During biological development, both methods are employed. The first possibility to coordinate a structuring process is known as self organization. In a group of similar cells, all cells follow the same set of instructions and are both, competent, and inducers at the same time. Mutual influence leads to cellular differentiation, e.g., a stable spatial gene expression pattern. Kondo [6] investigated how the stripe pattern of a marine angelfish grows and rearranges itself during development through self organization. He presents a computational model based on a Turing system [7], composed of equal basic units, which can create growing patterns with striking similarities to the biological example. The second way to coordinate growth on a cellular level is hierarchical control of differentiation. A subset of a group of cells acquires induction capabilities and induces organized cellular differentiation in surrounding cells. These *organizer cells* exist in biological development throughout embryogenesis. Interestingly, it seems that early organizer cells are predefined by maternal gradients, and divide into new organizer cells, which then coordinate development in later embryonic stages. The most prominent biological example is the *Spemann's Organizer* in vertebrates [8]. It is responsible for the induction of embryonic axis specification.

Pattern formation in systems of coupled units is an active field of research. Abstract computational models exist for the study of their dynamical behavior, e.g. reaction diffusion equations, cellular automata, etc. The model discussed in this thesis comprises simulated chemical diffusion to achieve cellular coordination. Since the cells are the medium that reacts to these diffusing chemicals, the model potentially allows for pattern creation abilities similar to a simple reaction diffusion model.

2.2.4. The Four Stages of Embryogenesis

Throughout the animal kingdom, embryogenesis consists of four typical stages:

- *Fertilization*
  Fertilization takes place as two gametes combine to form the *zygote*, the fertilized egg cell. The genetic material of two organisms is recombined and the fertilization process triggers development to start. Primary body axes are defined in this stage, usually by maternal gradients, produced by the mother individual.

- *Cleavage*
  The zygote transforms through multiple, rapid cellular divisions into a coarsely structured heap of cells. From this early stage on, organizer cells are created, which have the ability to induce further developmental processes in their neighboring cells. From experiments with model organisms, it is known that most cells in this stage do not have a pre-determined final function, i.e., they do not have a defined cellular fate. Rather, cells are pluripotent, thus can take a multitude of roles assigned to them by chemical signals available at their relative position in the embryo. In this stage, embryos are robust: a loss of cells usually still results in a complete individual after embryogenesis. However, a loss of organizer cells results in an incomplete embryo that terminates development in the final part of cleavage, the blastula stage.

  The blastula is a hollow ball of cells, containing enough cellular material and enough space on the inside for the following stage of gastrulation to proceed.

- *Gastrulation*
  While cleavage results in a relatively simply shaped blastula, gastrulation is the phase where physical structure is created in the embryo. Cells and tissues start to move in a coordinated fashion. While patterns of gastrulation vary between different species, only a few types of cellular movement can be observed, and are described in [1] as follows

  - Invagination: The infolding of a region of cells, much like the indenting of a soft rubber ball when it is poked.
  - Involution: The inturning or inward movement of an expanding outer layer so that it spreads over the internal surface of the remaining external cells.
  - Ingression: The migration of individual cells from the surface layer into the interior of the embryo. The cells migrate independently.
  - Delamination: The splitting of one cellular sheet into two more or less parallel sheets. The result is the formation of a new sheet of cells.
  - Epiboly: The movement of cell-sheets that spread as a unit, rather than individually, to enclose the deeper layers of the embryo.

- *Organogenesis*
  After gastrulation, the embryo is coarsely structured. Functional parts of the organism are separated. The following stage of organogenesis is characterized by local cellular interaction to form functional organs such that hatching or birth of a juvenile or larval stage of the organism ends embryogenesis.

When we consider the conserved order of embryonic stages above, and recollect the systematic coordination mechanisms of development described in the previous section, we might wonder whether a typical communication mode can be attributed to certain stages. Developmental biologists have historically coined the terms *regulative* for a self organizing, and *mosaic* for a hierarchical, modular development. They believed that embryos follow one of these strategies exclusively. As stated by Raff [4], recent

understanding shows that all embryogenies reach a point in development, where a modularization and simultaneously an irreversible differentiation have taken place, such that a removal of parts can not be compensated for. Raff argues that this point is only reached later in those species whose development was termed regulative. However, even in mosaic embryos, self-organization is still active in later stages of development, and can even be found in the grown individual, e.g. as wound healing. Nowadays, it is accepted that both coordination mechanisms can be found in all stages throughout embryogenesis.

2.3. The Evolutionary Perspective on Embryogenesis

The fossil record indicates that multicellular organisms emerged from single cellular organisms under selective pressure. One speculation for an evolutionary advantage of multicellularity is that sticking together to increase size of individuals could hinder smaller predators to feed on such an individual. As a consequence, cells at the core of such a heap of cells were completely engulfed by other cells, which made the intake of nutrition from the environment virtually impossible, except by mutual exchange of chemicals between inner and outer cells. In this way, cooperation evolved [9].

An effective share of labor is a factor for the success of multicellular development. An example are the cyanobacteria, which form colonies organized in filaments. Their normal mode of operation at daytime is photosynthesis to gain energy. At nighttime, nitrogen fixation is used for protein synthesis. Both processes are exclusive: the oxygen resulting from photosynthesis would hinder nitrogen fixation to occur in the same cell. Under environmental stress, single cells in regular distances along the filamentuous colonies differentiate into heterocysts; specialized cells that are equipped with a solid cell membrane such that oxygen cannot enter them, which allows them to perform nitrogen fixation even at daytime. The remaining cells continue with photosynthesis. Through channels in the cell membranes, neighboring cells exchange synthesized proteins and energy gained from photosynthesis, resulting in an efficient cooperation between the cells of a colony [9].

Development evolved as a successful means toward survival. At the same time, these early developmental processes created the foundation for future evolutionary steps. Nowadays, it is known that development plays a central role in the evolutionary change of multicellular organisms. I will discuss how developmental mechanisms and their change contribute to Darwins idea of "descent with modification" [10]. In this context, we will see that developmental modules are a prerequisite for evolution to act successfully on multicellularity [4]. Per definition, modules are units that are coherent within themselves and also parts of larger units [1].

In 2006, Wesseler published her paper "Eukaryotic Transposable Elements: Teaching Old Genomes New Tricks" [11], on the use of genes and parts of regulatory networks as modules. Many of these old genes – new tricks relations can be observed in development, as described in the following.

- **Dissociation – Heterochrony and Allometry:** Dissociation refers to the parallel nature of development. It is sometimes possible to separately influence or change a developmental feature by genetic mutation, without altering the development of others. Particularly, it is possible to influence the temporal characteristics of modules genetically: modular processes are activated at variable times during embryogenesis, and run at different speeds relative to each other. The concept of heterochrony and allometry describe such relative differences in timing and speed: Klingeberg states that "Heterochrony is defined as evolutionary change in rates and timing of developmental processes" [12]. An example is given in [1]: The eyes of birds and lizards initiate growth earlier in development than mammals. Thus, they are proportionately larger in later, comparable developmental stages. Further, Klingenberg notes that "Allometry is the pattern of covariation among several morphological traits", where a typical example is the central toe of a horse, which grows 1.4 times faster than the lateral toes. In this way, an otherwise normal foot development creates the typical horse hoof shape.
- **Duplication and Divergence:** Duplication and divergence describes the way in which modules are replicated, and can subsequently be altered, without changing the original module and thus without losing the original functionality (as proposed by Susumu Ohno [13]). Typical examples are the Hox genes. These genes are expressed along the anterior posterior axis in all vertebrates. Further, they are homologous to the homeotic selector genes, which have a similar expression pattern in drosophila fruit flies. There exist 4 copies of the Hox genes in mammals. They are very likely results of duplication events. From gene knockout experiments, it is known that
  1. different sets of Hox genes are necessary for regional specification of the embryo, and
  2. duplicated Hox genes have acquired new functionality. More specifically, they are responsible for the development of different subsets of organs in similar regions.
- **Co-Option:** When pre-existing units are redeployed for a novel purpose, developmental biologists speak of co-option. Examples are leg-structures, that were intended for walking and have transformed into wings, or fish-jaw bones, that became parts of the mammalian ear. The implications for evolution are twofold: Firstly, some developmental processes might be more flexible than others, and thus allow for easier adaptation, which in turn results in their selection in a changing environment. Secondly, evolutionary history matters! A phenotype results from a process that is the accumulation of changes to the development of its ancestors. Therefore, if development relies on a certain process, and this process has certain constraints, the extend of possible evolutionary change might be limited, which in turn influences future phenotypes and thereby future evolutionary steps.

Generally, evolutionary constraints come in various flavors. For example, they are *physical*, such that diffusion, hydraulics, etc. influence development. Or they can

be *morphogenetic*, where the developmental process itself does not allow for certain changes. For example, the middle digit of all vertebrates is always longer than the surrounding digits. Finally, constraints can be *phyletic*, meaning that the development of a part of the embryo cannot be changed without affecting other parts. This is related to a non-complete modularization.

2.4. Summary

During early development, a single fertilized egg cell transforms into an embryo with distinguishable, functional organs. This structure building is achieved by cellular proliferation and differentiation, with cellular processes controlled by information stored on the DNA. The DNA is transcribed and translated into proteins, which are the main building blocks of cellular material and cellular communication. The DNA transcription is thereby triggered by special proteins, the transcription factors. Since proteins result from DNA, mutual interaction between genes can result. This mutual interaction can be represented as a directed graph, which is called gene regulatory network. On the cellular level, diffusion allows chemical signaling to occur, which results in localized cellular interaction. Thus the embryo can grow, through both, cellular self organization, and hierarchical control between cells during embryogenesis.

Biological evolution resulted in incremental design of the process of embryogenesis. That is the reason for many growth patterns, genes, and pathways to be preserved in different species. However, often these parts are adapted and used in different context, with different trigger processes and targets. Nevertheless, the embryonic building process seems to be code-efficient. There is reuse of modules on a genetic level and the use of environmental information, e.g. Hox genes, temperature influence, etc. Because of the cellular nature, and the initial single cell and consecutive divisions, the process has an inherent "coarse to fine" character over time. Consequently, the structuring in the developmental stages is coarse to fine. Note that the developmental building process is robust: in many stages, almost any cell can be replaced and/or removed, with the big exception being the organizer cells, as described above. Apart from being robust, the building process still is evolvable, i.e. nature has found a representation that can adapt its building blocks to novel environmental influences to achieve ongoing individual reproduction.

3. Simulation of Development

3.1. Introduction

The simulation of development for scientific purposes has mainly two different roots: Firstly, biologists aim for an understanding of biological development through simulation models, and secondly, computer scientists try to harness developmental representations for optimization purposes. Independent from the reason to simulate development, models exist on many different abstraction levels, from detailed microscopic biochemical and biophysical modeling to abstract mathematical formulations for pattern creation.

Biologists usually want to model microscopic processes in great detail to gain knowledge about the development of the simulated organism. Therefore, models should have predictive power for a small subset of dynamical behavior of the organism. In this context, computer models should enhance or replace traditional biological *in vivo* or *in vitro* experiments since they are easy to handle: experiments that have been performed once, can be repeated exactly – without much effort. All data are available, and undesirable external conditions, such as changing temperature or lighting conditions, do not exist. Artificial evolutionary development is a computer process that necessitates coupled simulations of multi-body mechanical interaction, chemical reaction-diffusion, and analog signal processing. Probably because of this complexity, computational modeling is still a less frequently used method among developmental biologists. An early example is the work by Furusawa and Kaneko [14], who investigate the emergence of multicellular development in a simple computer simulation environment. Since the fossil record does not provide sufficient information on this transition, computer models can help to fill this gap or at least provide novel hypotheses and viewpoints, which can subsequently be investigated. The authors simulate spherical cells, diffusing chemicals for cellular communication and cellular adhesion to find that multicellularity emerges when nonlinear dynamics with oscillations, differential adhesion, and cell-cell interaction through media is present, without any need for natural selection processes. Similarly, Hogeweg [15] shows in her computational model that morphogenesis results directly from differential cell adhesion, cell-cell signaling, cell growth and programmed cell death (apoptosis), and can yield biologically interesting morphologies when an evolutionary process toward increased celltype-diversification is simulated. Järvinen et al. [16] create a model for tooth-growth in mice, which accurately replicates the biological growth process. Recently, Cickovski et al. [17] have created a software framework for the simulation of biological developmental processes, which uses a modular organization to combine simulations on transcription-, cellular- and chemical environment-level. This tool could help to further advance the use of computer simulations in developmental biology.

In this chapter, I want to focus on the more abstract models used by computer scientists, who target the investigation of systemic principles for engineering purposes such as optimization. Particularly, points of interest are:

1. scalability/high complexity: The ability to perform a successful optimization process in a high-dimensional problem domain is evident in biology: evolution has found ways to create complex, viable organisms in a natural environment. Thus, computational models of development try to achieve similar features by encoding a building process, rather than the final artifact in the object variables of an evolutionary algorithm.

2. adaptability/robustness: Biological organisms function robustly in diverse environmental conditions, e.g., diverse mechanisms shield the body from changes in external temperature, skin color changes according to lighting conditions, and organism locomotor systems adapt to their environment by muscle and bone growth, which is mediated by strain. A constant control and adaptation of tissue production and change is necessary for robust functionality. Genetic mechanisms are responsible for this developmental aspect of biological organisms.

3. self repair: Upon injury, biological individuals show a high capacity to re-generate. In early stages of development, even the loss of half the embryo will not impede the growth of the organism. Computer models try to capture and analyze this feature of development to transfer self-repairing ability to technical artifacts.

Since the simulation in this context is not intended to precisely reflect biological development, but rather to exhibit similar features, simplifications at different levels are made. Throughout this thesis, for the purpose of grouping different ways of abstracting development, I will distinguish between the simulation of *control mechanisms* and the simulation of *phenotypic mechanisms*. Control mechanisms are those parts of simulation, that mimic the representation and decoding of the DNA for the control of development, including the dynamics of this process. For example, the genotype representation is part of these control mechanisms, as well as the simulation of gene-interaction such as mutual expression control. In contrast, phenotypic mechanisms are the simulated cellular environment and physics, such as shape and structure of single cells, and cellular interaction via chemical diffusion and mechanical forces. In biology, this separation is slightly counterintuitive: For example, proteins play a role in both processes, since they are used for the control of gene expression as well as for building up cells and mediating physical and chemical cellular interactions. However, in computer simulations, separating between these mechanisms seems natural, since programs usually distinguish between these two aspects, e.g. employing different simulations for gene regulation proteins than for physical cell structure simulation.

In the following section, I will present developmental models employed in computer science, particularly in the context of evolutionary computation. I will group them based on the two features mentioned above. A critical evaluation of the scientific field will conclude the presentation. Subsequently, I will describe the features of the model used in the studies presented in this thesis.

3.2. Related Work

The research field of computational evolutionary development is still young. Nevertheless, some influential work marks the corner-stones of Artificial Development: The first (and perhaps most cited) research work published on development for evolutionary optimization purposes is related to the field of artificial neural networks. In 1990, Kitano [18] used matrix rewriting, a method to incrementally create a connection matrix of a neural network and claimed inherent modular organization of resulting connection matrices as well as better scalability to higher dimensional problems as compared to a direct coding approach. Eight years later, both claims were disputed [19]. Especially, scalability was shown to only hold for certain initializations of the direct coding algorithm. In 1994, Sims [20] presented a simulation of virtual creatures that performed several tasks such as swimming and reaching for an object. Both, morphology and control were co-evolved. Morphology was created by following a developmental process, that assembled blocks to form a body. Unfortunately, few information exists on the implementation details. Nevertheless, the impact of this work was high, since a development-like process was first shown to be able to create artificial creatures with biology-like features and behaviors, such as undulatory swimming. One year earlier, Fleischer presented a model for cellular development that had a wealth of simulation mechanisms included [21]: 2D physical simulation of spherical cells, reaction-diffusion, cellular communication using chemical signaling, chemotaxis, etc. The computational requirements were high, and experiments had thus been of very basic nature. For example, manually coded cellular behavior yielded cellular assemblies that resembled biological tissue. In the following years, a few conference contributions on Artificial Development were published, and after 1998, an increasing number of articles on evolutionary development in an artificial life context appeared. The first taxonomy paper of the field [22] was published in 2003 and an edited book from the same year comprised most of the state of the art modeling and analysis approaches [23].

3.2.1. Dynamics of Development – Control Mechanisms

The cornerstones presented above give a rough overview over the timeline of the young field of evolutionary development. In the following, I will give an overview over different approaches toward control of development, using a schematic grouping into grammatical approaches, cellular automata, simulated gene regulatory network dynamics and alternative approaches.

Control via Grammatical Approaches

Grammatical approaches can be traced back to L-systems, originally proposed by Lindenmayer [24]. L-systems are grammatical rewriting approaches, which iteratively replace single elements of a string with a sub-string by executing *production rules* (see Figure 3.1). Lindenmayer originally used L-systems to represent plant development in biology. While L-Systems have the advantage of producing genotype to phenotype

maps that are relatively easy to understand, results are always fractal in nature, and thus the design space is limited.

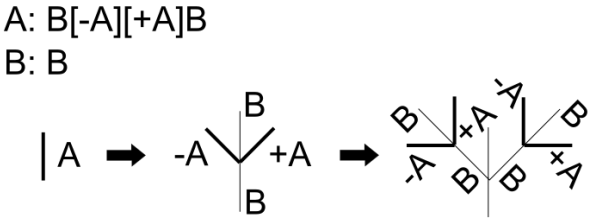


Figure 3.1.: A schematic example of an L-System, as presented in [22]. A denotes the symbol that is rewritten in this example, denoted by the bold line in the plots. + and - denote positive and negative angles for the graphical output. The production rules are stated in the upper left corner of the figure.

However, L-Systems can be extended by different means. For example, parameterizing the rules allows for multiple outcomes of the execution of a single rule. For the same purpose, rules can be implemented to be context sensitive (see [22] for an overview). Several grammatical approaches for simulating embryogenesis have been proposed in different contexts, such as neuroevolution[25, 26], evolutionary hardware [27], and evolution of morphologies [28, 29].

Recently, the focus of the community seems to have shifted away from purely grammatical approaches, probably due to the restriction of design space mentioned earlier. Also, L-systems seem to be too abstract to capture complex biological development, even though results show a striking similarity to plant morphologies.

Control via Cellular Automata

Cellular automata [30] consist of a set of rules and a spatially discretized distribution of cells. During execution, the cell state of all cells is updated according to rules stored in a rule table, similar to L-systems. Taking into account the states of neighboring cells, the new state of a cell is defined in an update step (synchronous or asynchronous). The main difference to L-Systems is that an element is not replaced by other elements according to its own type only. Instead, the actual configuration of a defined neighborhood determines the choice of rule from the rule table (see Figure 3.2). Thus, similar to biology, all cells have access to the same rule table (the DNA in biology), and perform differential actions according to their positional information.

Cellular automata are mostly researched on 2D lattices, where the spatial neighborhood defines the decision configuration for rule choice. Problem domains that have been researched are thus often related to 2D pixel images. For example, cellular automata can be used to create 2D microstructural patterns, to be employed in material science [31]. More indirectly, the growth of 2D truss structures can be modeled based on cellular

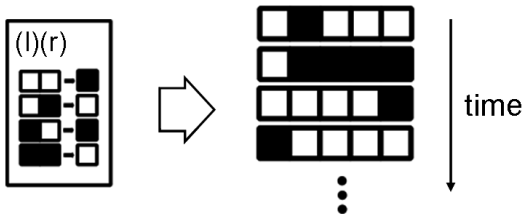


Figure 3.2.: This example shows a graphical representation of a 1D binary cellular automaton. White color represents the value 0 and black color represents the value 1. The table in the left panel of the figure gives the rules for cellular updates, depending on the two neighboring cell states to the left (l) and the right (r). The figure shows a few steps of cellular dynamics according to the rule table, with periodic boundary conditions applied.

automata [32]. Resulting genomic representations are able to grow trusses in a variety of environments without being adapted or re-evolved. More abstract scientific questions can also be tackled evolving cellular automata. The construction of self-healing and stable cellular arrangements through evolution of 3D cellular automata is possible [33]. A direct selection for stability in the evolutionary algorithm is necessary, and less than 20% of experiments evolve the desired function. Analysis reveals that successful solutions employ two different strategies: homeostasis, and constant apoptosis with cell regeneration. In the field of evolvable hardware, cellular automata can be used to instruct the building of an n-bit adder and n-bit parity calculator [34]. Recently, Harding and Miller compared direct encoding and cellular automata on a series of different Kolmogorov-complexity 2D patterns [35]. They find that the direct map outperforms their cellular automaton implementation for most experiments. Also, they find that scaling, attributed to developmental processes, is not achieved in their model. They infer that direct maps are reliable in evolutionary computation, while features typically attributed to development, are not inherent for naive implementations and may thus need more investiagion.

Control via Simulated Gene Regulatory Network Dynamics

Approaches that belong to this category are based on simulation of biological gene transcription and translation processes. As described in Section 2.2.2, biological organism development is controlled by genes that interact through proteins and form regulatory networks. Such networks consist of nodes (the genes) and connections (the regulatory links between genes), and the structure of a gene regulatory network is encoded on the DNA. In simulated gene regulatory network dynamics, a string of symbols is employed as representative for the DNA (e.g. binary values, characters, integer or double precision numbers). To create a structuring of the information on the simulated DNA into genes, two different strategies are usually employed. Firstly, a gene can be predefined as a region of fixed size and position on the string. In that way, the DNA would contain a sequence of sub-strings, usually with equal lengths, each defining the properties and connectivities of one gene. The second implementation possibility is that a short

sequence of elements from the used alphabet can denote beginning and end of a gene, such that the region between such markers contain the property information of one gene. The elements inside a gene usually define whether and how one gene regulates the expression of other genes, and cause a cell to perform a cellular action, such as division, inter-cellular signaling, etc. Gene expression order is either randomized or follows a defined sequence, iterated over several time steps to produce a time course of gene activations, the gene regulatory network dynamics.

Several researchers have employed gene regulatory network-based methods for the creation of controllers for simulated and physical agents. For example, early realizations were employed in the development of artificial neural networks for robot control [36]. In this work, gene regulatory networks evolve to create 2D multicellular arrangements and growth of dendrites is simulated to connect the cells. The resulting network is used to control a robot for the tasks of corridor following and object avoidance. Similarly, simulated gene regulatory networks can be evolved to control development of complete agents, including the growth of both, morphology and control. The approach has been shown to be able to evolve individuals from scratch that can follow a curved line in a simulated environment [37]. For locomotion, Eggenberger-Hotz developed a gene regulatory network model based on structural units and regulatory units [38]. While he focuses his descriptions on the cellular representation of the model, he describes an application to the evolution of locomotion, where no neural network is needed since the simulated gene regulatory network controls cellular movement directly. Reisinger and Miikkulainen compare a gene regulatory network based development with the popular augmenting topology method NEAT [39] and direct coding toward creating neural networks that are able to play othello [40]. While NEAT outperforms direct encoding, the gene regulatory network method outperforms NEAT.

Another topic of interest that employs gene regulatory network simulations is the investigation of specific features of a developmental representation, such as robustness and scalability. In 2005, Bentley has shown that graceful degradation can be observed in software which realizes its function through a simulated developmental process, when the binary files are corrupted after compilation, before execution [41]. Bowers investigated scalability, robustness, and modularity in growth processes toward 2D target patterns [42] and showed that for simple targets, all these factors evolve. A thorough analysis of scalability in an evolutionary developmental system using gene regulatory network simulation was presented by Federici and Downing [43]. The gene regulatory network in this model is simulated as recurrent neural network. The authors show scalability for 2D patterns, and find that development outperforms a direct encoding when the task is scaled to higher dimensions.

A number of articles discuss the dynamics of evolving gene regulatory network simulations. For example, Reil investigates transitions between attractors of gene regulatory network dynamics, and shows evolvability of gene regulatory networks toward complex limit cycles [44]. Knabe investigates the evolution of a gene regulatory network model toward limit cycle dynamics, and cellular differentiation [45]. He finds that evolved graph motifs in his artificial system are not as important as might be expected from the work of Alon [46]. Banzhaf presents a gene regulatory network model that is not

coupled to cellular development [47]. He describes dynamical behavior of networks resulting from a randomly chosen DNA, and shows simple evolvability toward a target concentration of proteins.

From a bio-inspired artificial life perspective, Geard and Wiles use a recurrent neural network as a representation for a gene regulatory network to evolve cell lineages similar to the C. elegans organism. However, networks have a fixed, predefined size and topology [48]. Psujek and Beer investigate the evolutionary perspective of a genotype to phenotype map that develops graph structures in general [49]. They evaluate the effect of random mutations for random gene regulatory networks and find 2 biases: a global bias (some phenotypes cannot be represented at all) and a local bias (the distribution of offspring networks depends on the parent position in genotype space, independent from the phenotype).

Alternative Approaches

There are a few implementations that cannot clearly be attributed to either of the three categories above, being a combination of them, or completely alternative.

For example, Sekanina and Bidlo developed a method that evolve an operator to transform a small functional sorting network into larger sorting network. Since this operator is applied recursively, arbitrarily large sorting networks can be evolved [50]. Nolfi and Parisi develop a semi-direct encoding of an ANN for controlling an agent in a pixel grid environment. All positions and connections of neurons are encoded in a chromosome. Still, the approach is generative, since axon growth, which is genetically defined, takes place in time, and thus results in an artificial neural network that grows incrementally over time [51].

Based on the cartesian genetic programming technique (CGP) originally proposed by Miller [52], two approaches extend the idea by introducing a developmental perspective [53, 54]. Instead of a gene regulatory network, the representation of developmental control is a structure similar to a digital electrical circuit. This structure is used to control cellular growth according to local information. Harding et al. investigate a self modifying variant of CGP [55]. Upon execution, the program can duplicate, delete, move and change functional parts of its own instructions. Therefore, the program creates a development-like, self modifying process. Results show that software can be grown to calculate squares of numbers without using multiplication or division, a result which cannot be achieved without self modification.

Stanley proposes a method to create graphs, which complexify through evolution (neuro-evolution of augmenting topologies, NEAT). In the context of artificial neural networks [39], his method is able to solve different tasks such as pole balancing and virtual agent control. The focus lies on the evolution of a genotype that completely describes the phenotype, and not a building process for the phenotype. Thus, the method is generative, but not developmental. Still, extending the approach with complex pattern producing networks (CPPNs), Stanley achieves solutions with attributes that are typically developmental, such as symmetry and repetition with variation in the creation of 2D images [56].

3.2.2. Cellular Simulation – Phenotypic Mechanisms

This section will give an overview over different approaches toward the simulation of phenotypic development. Again, I will use a schematic grouping into different approaches: grid based, non-grid based and complex cells.

Grid Based

Cells in grid based environments maintain positions on vertices of a structured grid (Figure 3.3, first panel). At each discrete time-step of developmental simulation, they occupy discrete positions with a defined cellular neighborhood. Note that this does not prohibit cellular movement: in several models, cells are allowed to move from one vertex to an adjacent one. Most simulation environments employ cartesian grids. For example, Miller and Banzhaf [53] introduced pixel based Artificial Development of 2D patterns through subsequent cellular divisions. Since it is computationally easy to arrange cells on a regular lattice, many models rely on a similar representation [57, 58, 43, 59, 60, 61, 62, 63, 64, 65]. Particularly, Bentley and Kumar introduced a pixel based simulation to create tessellating tiles by employing periodical boundaries [66]. Rieffel and Pollack devise a method to develop morphology not through subsequent cellular divisions, but through a single motile agent that leaves a 'material'-trail on the pixel grid while moving. Hogeweg [15] groups multiple pixels to macro-cells, while developmental instructions are still executed for each pixel alone. This allows for a great flexibility in the creation of shapes. Finally, Astor and Adami [67] simulate development on a hexagonal lattice.

Grid based models have the advantage that they allow fast computation of cellular positions and resulting cellular interactions. The disadvantage is encountered when cellular growth via division is simulated: if a cell is instructed to divide into an adjacent vertex of the grid, which is already occupied by another cell, a conflict results which in general is treated with one of the following solutions: 1. overwriting the adjacent cell, 2. not executing the division, 3. implementation of a mechanism that simulates physical pushing of adjacent cells onto available vertices. Also, rectangular artifacts may be encountered in resulting phenotypes.

Non-Grid Based

This group of cellular simulation allows for cells to be freely arranged in space, not adhering to a position on a grid, but rather determining their position relative to the remaining cells. In general, cells in this group are motile and can change their location by simulated mechanics. Different possible implementations exist, e.g. cells can be represented by spheres, but also by polygons (see Figure 3.3, center and right panel).

Models in 2D and 3D usually rely on a simulation of springs that dynamically displace cells according to each others position [14, 68, 38, 42, 69]. In contrast, Fleischer and

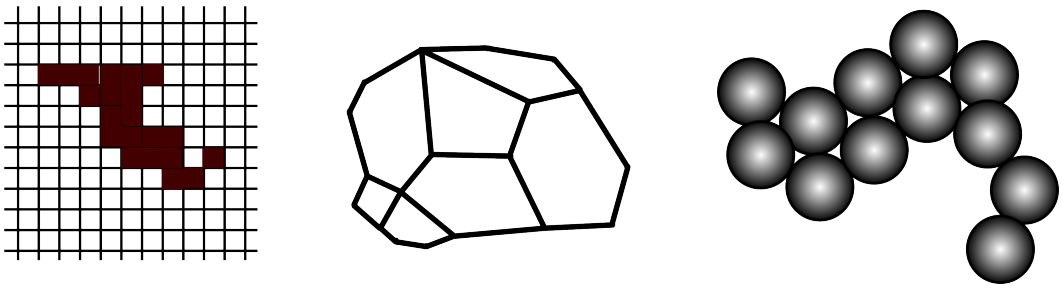


Figure 3.3.: The figure shows three possible representations of cells. The left panel shows cells positioned in a cartesian grid; cells are denoted by dark squares. The center panel depicts polygonal cells, where linear cell walls are connected and usually represent springs for physics simulations. The right panel shows an assembly of simple spherical cells.

Barr developed a model that relies on multiple rigid body interactions for cellular displacement in 2D [21].

In general, the advantage of non-grid based cellular simulations for simulated embryogeny is that cells possess the ability to move around freely, such that no artifacts from a grid-based cellular organization will influence the resulting development and/or final shape. Especially, the aforementioned collision of cells through division can elegantly be handled by simulating such inter-cellular mechanics as springs. However, the computational cost for such simulations is usually high compared to grid-based methods.

Complex Cells

This group of models simulates cells which have complex features, such as hinges, or mechanical elasticity, or are composed of several sub-parts. For example, for the modeling of development of artificial creatures, Sims employed cells that are rectangular blocks in 3D and represent body parts of mobile agents [20]. In the final morphology, these blocks are interconnected by actuated joints. Similar work has followed, e.g. Spector et al. use so called division blocks to represent agents [70] and Komosinski and Rotaru-Varga [71] use a similar implementation and call it the framesticks model [72].

Using a representation for cells that is directly linked to the calculation of desired properties of resulting designs has been employed by several researchers in the mechanics domain. For example, Kawamura and Ohmori describe a developmental model, that consists of placing truss elements in a defined design space to grow truss structures [73] which can be evaluated using finite element calculation. Similarly, Devert et al. simulate development with trusses, that form a regular mesh in the beginning of simulation [74]. During the developmental process, truss lengths and strengths are varied and thus reshape the initial truss structure toward optimal designs. Yogev et al. model hexahedral cells in 3D, which can grow and divide and thus form complex structures for the creation of lightweight, stable designs [75]. Since these cubes are basic elements for finite element calculations, the growing design can directly be evaluated toward mechanical stability.

A unique implementation of development is presented by Taura and Nagasaka [76]. Instead of multiple cells, a single spherical medium represents morphology. Changes to this medium are simulated by moving points on its surface, and interpreting point density as extrusion strength for transformation of the medium. The initially spherical medium is then deformed to represent a complex shape.

Downing has a more abstract notion of cells [77]. He employs a hierarchical formulation for neural development, where cells are placed inside neuron groups, and a number of neuron groups make up neuromeres. Neuron groups can change an abstract size and thereby influence neuron connectivity. His cells have no physical representation, but rather a connection pattern that results.

3.2.3. Discussion of Related Work

From the publication dates of major contributions to the field, we can see that evolutionary development is still a relatively young research field. The main motivation to study development in an evolutionary context from a computer science perspective is to achieve optimization results, that exhibit

- scalability, i.e., the possibility to extend the approaches to high dimensional problem domains,
- robustness, i.e., the ability of evolved solutions to function under small condition variations, such as noise on or small fluctuation of the environmental constraints and
- adaptivity, i.e., the ability of one evolved developmental process to produce different, adapted artifacts, when faced with modified boundary conditions.

Altogether, researchers expect to gain an evolvable representation of complex artifacts.

In the field of evolutionary development, there is no state of the art algorithm, and no generally accepted simulation environment is shared by larger groups of scientists. Even for well defined fields of optimization, such as topology optimization for stable, lightweight structures, it is not clear yet whether evolutionary development yields functional solutions, and how these would compare to standard optimization approaches.

Generally, we can observe that most developmental models need much effort for their implementation. The work to create a developmental model is highly interdisciplinary: cell physics and chemistry simulations, as well as efficient implementations of parallel computing, together with research and abstraction of biological phenomena need to be performed. Hence the multitude of implementations that can be found, and the strong focus on description of implementations, as compared to a rather underrepresented analysis of the models.

Research on evolutionary developmental systems has mainly concentrated on the phenotypic side. Evaluation of e.g. phenotypic robustness and scalability is relatively often performed for the models. Only few exceptions exist, that investigate developmental

models also in genotypic aspects, as well as in the important genotype to phenotype relation [49, 78].

For evolutionary development research, it would generally be sensible to look more at problem domains that are well known and that have known state of the art solutions. Then, new approaches can be compared to standard methods and advantages or drawbacks could be set into context. Also, a thorough analysis of the evolution of developmental models is hardly found in the literature. Not only the evolved solutions, but also the process that lead to them must be investigated at gene regulatory network dynamics level which underlies development.

The choice of level of abstraction for simulations is divers, since it depends on the specific problem domain or even the problem that evolutionary development is applied to. Thus, the assessment of an abstraction level can only be done in relation to a class of problems that it is applied to.

3.3. The Simulation Environment

In this section, I will describe the methodology for simulations of multicellular development within an evolutionary computation framework, which is the basis for the investigations presented in this thesis.

Abstractly, the developmental process can be seen as a genotype-to-phenotype mapping. In this interpretation, it is a module that is compatible with optimization techniques that rely on point wise evaluations of genotypes to asses their quality. In this context, development represents the decoding of the genotype into an artifact in design space, the phenotype. Thus, it is possible to describe development detached from the optimization method. It can be seen as a black-box that has a vector of object variables as input, and returns a spatial distribution of cells ready for evaluation. In a similar way as for the previous section, I have structured the following description of the developmental model into two parts: 'control mechanisms', dealing with the simulation of a dynamical system for the control of the cellular growth and 'phenotypic mechanisms', which comprehends all model features related to physical environment and cellular representation.

3.3.1. Control Mechanisms

Graph Based Control

Graphs are state of the art for representing dynamical systems. An example is the group of continuous time recurrent neural networks [79], employed in areas as diverse as pattern recognition, time series prediction and robot control. Inspired by biological development, mutual gene transcription control is represented as a graph, the gene regulatory network. In contrast to artificial neural networks, the nodes of gene regulatory networks usually have slightly more complex activation functions. Also, since

gene regulatory networks are usually part of multicellular simulations, i.e. they are embedded in a spatial context, the activity function of the nodes contains a diffusive term, which distributes signals spatially in the given simulation environment. Finally, the general modus operandi in gene regulatory network models is to influence graphs by evolutionary change in both, structure and connection weights, while usually the weights of a neural network are the focus of adaptation. Structural adaptation can of course also be done in neural networks, and the scientific field of neuro-evolution [80] is thus related to research in artificial embryogenesis.

The model presented in this section relies on the graph-paradigm as well. Inspired by the model presented by Eggenberger-Hotz [81], the graph-structure results from the evaluation of a vector of double precision values, called the virtual DNA (vDNA), of which an identical copy is available for translation to all cells in an individual's development. This vDNA is grouped into regulatory subunits (RUs) and structural subunits (SUs), subvectors of a defined size, which are initially lined up in a random order. This choice of vDNA structure is inspired by the natural organization of the genetic material in biology: RUs correspond to TF-binding sites at the promoter, and SUs to exons (see Chapter 2). A functional unit of this vDNA, called a gene, is composed of a group of SUs and its preceding RUs. The values inside the SUs encode actions that a cell should perform. This includes the generation of gene activation signals, while the RUs determine the activity of a gene, based on these signals. The actions and signaling encoded in a gene will only be performed if it is active.

An illustrative example of a genome with three genes is given in Fig. 3.4. Note that the RUs behind the last SU and the SUs in front of the first RU are not taken into account for the developmental process, since they do not compose a proper gene, lacking SUs or RUs respectively.

In the model, simulated transcription factors (TFs) take part in gene regulation. The detailed process is presented below, so far, it is important to note that each simulated TF consists of a spatial distribution of concentrations, discretized on a grid, and an affinity value. This affinity value can be seen as an identification number, and is used to determine whether a TF alters the activation of a given RU of a gene. Also, each TF has a defined decay rate and diffusion constant.

In the following, I will give details about the key components of gene expression and regulation in the model.

- **Structural subunits:** A SU encodes the action to be performed, and contains the parameters that specify the action. Possible actions include cell division, production of a diffusing chemical signal (a TF for cell-cell signaling and gene activity regulation), and determination of cell-cell adhesion forces.

  Formally, a SU is a vector $\mathbf{x}$ with five components $x_i \in [0..1]$, $i = 1, \ldots, 5$. x_1 is used to determine the type t of action encoded by the SU:

$$t = \begin{cases} 1 & \forall\, \mathbf{x}: \quad 0 \leq x_1 < \frac{1}{3} \\ 2 & \forall\, \mathbf{x}: \quad \frac{1}{3} \leq x_1 < \frac{2}{3} \\ 3 & \forall\, \mathbf{x}: \quad \frac{2}{3} \leq x_1 < 1 \end{cases}$$

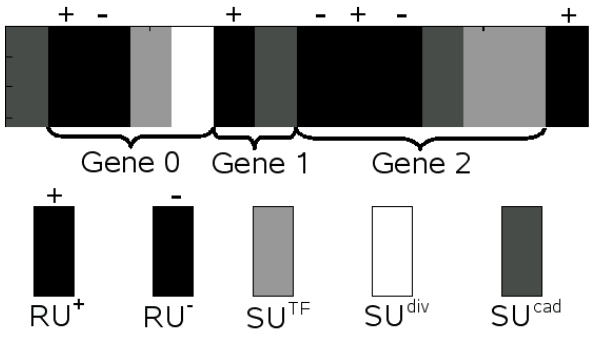


Figure 3.4.: An illustrative vDNA with three genes, each consisting of one or more structural subunits (SUs) and regulatory subunits (RUs). Two different kinds of RUs exist: inhibitor (RU^-) and activator (RU^+). A SU coding for the production of a transcription factor is denoted by SU^{TF}, a SU coding for a division by SU^{div} and a Cadherin producing SU by SU^{cad}.

If $t = 1$, cell division is encoded and x_2 is used to determine the division angle, while the values x_3 to x_5 are ignored. If $t = 2$, the production of a TF is encoded and x_2 represents an affinity value assigned to the TF (aff_{TF}), x_3 defines the amount of TF to be released, x_4 is a diffusion constant, and x_5 the decay rate.

In the case of $t = 3$, the simulated production of adhesive molecules is encoded by the gene and the type of adhesive molecule is determined by x_2. In my model, cells containing the same type of adhesive molecule will adhere to each other. Note that for $t \neq 2$, not all x_i are used, but they are still kept as a part of the SU. Therefore, mutation affects them without being subject to selection pressure. If a mutation results in $t = 3$, all five values are required again as parameters for the production of a TF. It can be speculated that this intermediate random-walk of the redundant parameters has an effect on the evolution process.

- **Regulatory subunits:** Two types of RUs are used in the model, which either increase (activate) or decrease (inhibit) the expression of a gene. RUs can 'sense' the presence of certain types of TFs in the vicinity of the cell. If the label of a TF is affine to a label associated with the RU, and if the concentration of the TF lies above a threshold, an activity value is determined for each RU. All activating (= positive sign) and inhibiting (= negative sign) activity values belonging to the same gene are summed up to determine the overall activity of the gene.

  More formally, a RU consists of a vector $\mathbf{y}$ with three components $y_i \in [0..1]$, $i = 1, \ldots, 3$. y_1 codes for an affinity parameter, which is used to determine the affinity between that RU and the surrounding TFs. If the affinity decision variable γ, calculated by

$$\gamma = 0.2 - |\text{aff}_{TF} - y_1|$$

  is greater than 0, the TF and the RU are affine to each other.

  Let M be the number of RUs belonging to a certain gene. Let L_j be the number

of TFs that are affine to RU_j, the jth RU of the gene, and y_i^j the ith entry of the vector $\mathbf{y}$ of RU_j. We first determine the indices k_j, which denote the TFs that have a concentration c that is greater than y_2^j:

$$k_j \quad = \quad \{k \in \{1, .., L_j\} : c_k > y_2^j\}.$$

Here, y_2^j can be seen as a threshold of RU_j. The partial activity a_{RUj} for the RU is given by

$$a_{RUj} \quad = \quad \sum_{k_j} c_{k_j} - y_2^j$$

A sum over all partial activities, scaled by y_3 yields

$$\alpha \quad = \quad \sum_{j=1}^{M} a_{RUj} \cdot (2 \cdot y_3^j - 1).$$

y_3^j can be interpreted as the sign of RU_j, because the term in brackets is negative for $y_3 < 0.5$. The overall activity A of the gene is finally determined by

$$A \quad = \quad \frac{2}{1 + \exp(-20 \cdot f \cdot \alpha)} - 1,$$

where f denotes the slope of the nonlinear function and is encoded in the vDNA for every gene. If A is greater than zero, the gene is active.

Note that for the multicellular simulation, every cell in the developmental process determines its cellular actions through evaluation of gene activity from the same vDNA in every developmental time step. Since TFs have a spatial distribution, positional information in terms of different TF-concentration at different locations can lead to differential cellular behavior, even though cells use the same vDNA. A sequence diagram of the developmental process is given in Appendix B.

Vector Field Based Control

In the following, I will motivate and describe an alternative approach toward controlling simulated development, termed Vector Field Embryogeny. In biology, as described in Chapter 2, TFs possess the ability to regulate the expression of genes by binding to the respective promoter regions on the DNA and thereby influencing genetic transcription processes. In this way, mutual interaction between genes occurs by means of their products, which eventually results in complex gene regulatory networks. gene regulatory networks are nonlinear systems that create complex patterns of gene activation. Nonlinear systems can typically be characterized by the following features [82]:

1. multiple isolated equilibria
2. limit cycles

3. subharmonic, harmonic, or almost periodic oscillations
4. chaos
5. multiple modes of behavior

Note that according to Khalil [82], *behavior* in item 5 refers to the set of dynamical features given in items one to four. Items 1 and 2 are abundant in the dynamics of gene regulatory networks. For example, multiple isolated equilibria can account for cell differentiation in biological organisms [83], and many different inter- and intra-cellular processes are represented by limit cycles [84], where probably the most prominent representatives are the circadian rhythms [85, 86]. Item 5 is the most prominent feature of life; the ability to adapt to different external conditions by switching between modes of operation can be found in virtually all organisms. Recently, it has been observed how a biological gene regulatory network dynamically changes its modes of operation, when environmental conditions are altered [87]. Items 3 and 4 are observed in computational models for biological gene regulatory networks [88]. Under certain conditions, simulated circadian clock genes exhibit chaotic and birhythmic behavior. However, it is argued that the smallness of the parameter range in which this occurs makes it unlikely to occur in biology. Also, known arrhythmic biological mutants of the circadian clocks seem to result from a severe structural change in the underlying network, rather than from normal mode of operation under certain environmental conditions [89].

In this light, the dynamical behavior of gene regulatory networks seems to account for the flexibility and robustness of biological organisms. Therefore, the most common approach to realize an artificial system with these features is to model the interplay between a number of genes to create regulatory networks. The natural representation of these networks is a directed graph. Each node of such a graph represents a state variable of the system, and the links indicate modes of interaction between nodes with connection weights and more or less complex activation functions.

Here, I propose to shift evolutionary focus from the structure and weights of the network to the dynamics that such a network would create, i.e., to its system phase space. Figure 3.5 illustrates the approach: I enable mutation operators to directly create and shape the system phase space (direct manipulation), instead of doing so indirectly via graph manipulation. This allows a more causal relationship between mutation and resulting changes in system dynamics. Direct shaping of the phase space is inspired by a method known as vector field editing [90] and will be described in the following:

In computer graphics, the vector field editing method is used for creating texture alignments and extracting analytical information about given graphical representations of vector fields [90, 91, 92]. To be able to apply this method to regulatory systems for artificial embryogeny, the formulation of an Artificial Development-system must be viewed in an abstract way. The following considerations are presented using a two-dimensional version of the system for clarity and visualization purposes. Note that the method extends to D dimensions by applying the respective D-dimensional geometrical operations.

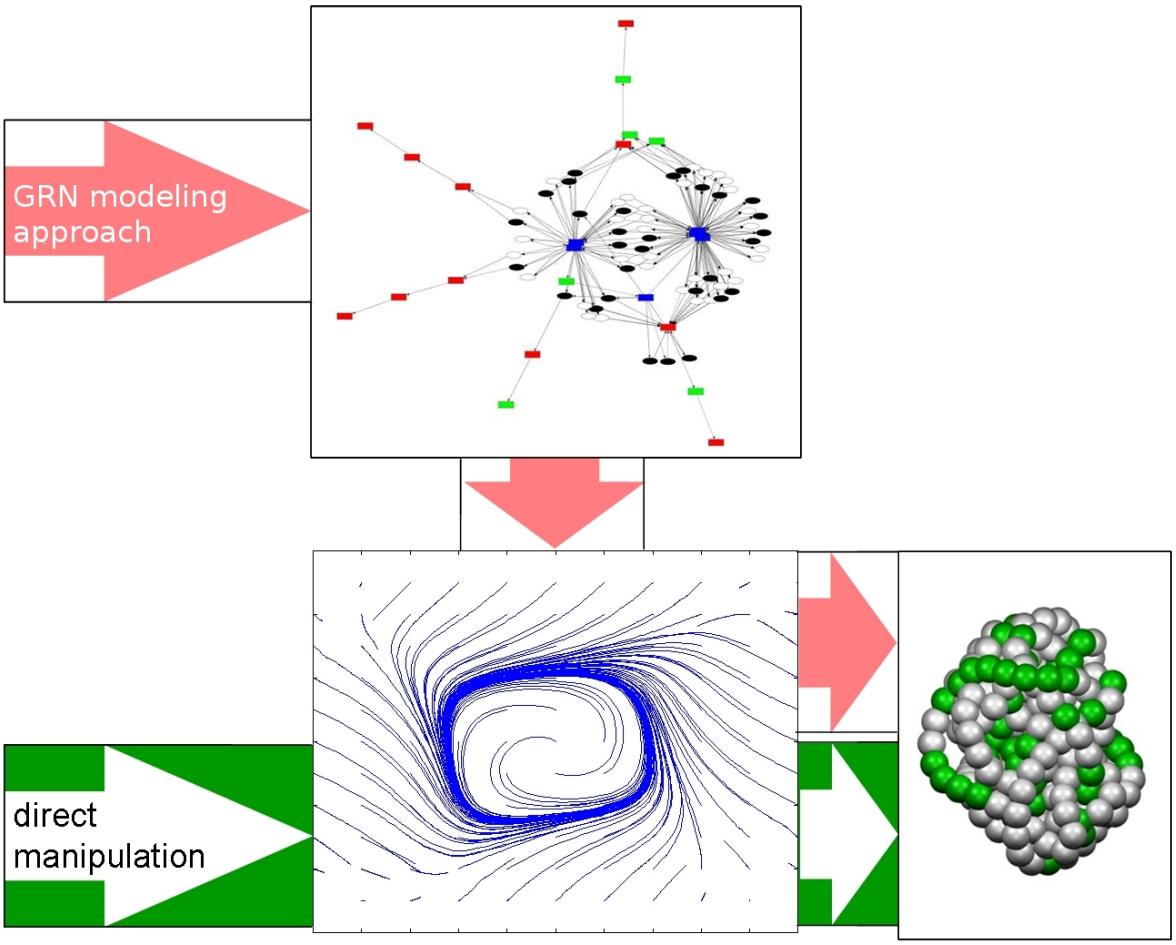


Figure 3.5.: Two different approaches toward evolving control of development. The usual approach for evolving developmental processes consists of manipulating a regulatory network, which then creates dynamical system properties that control the developmental process (upper arrow). The approach presented here omits the network representation by directly manipulating the system phase space, i.e., the dynamical behavior of the system, to evolve a control for development (lower arrow).

Consider an arbitrary simulated gene regulatory network inside a cell, with two genes of interest (Figure 3.6). I denote the state (i.e., activation level) of these two genes by x_1 and x_2 respectively, and together as the vector $\mathbf{X} = (x_1, x_2)$. The temporal behavior of any deterministic simulation of a regulatory network containing these two genes can now be described with respect to $\mathbf{X}$ by the differential equation

$$\mathrm{d}\mathbf{X}/\mathrm{d}t = \mathbf{F}(\mathbf{X}, \boldsymbol{\lambda}, t), \tag{3.1}$$

where $\mathbf{F}$ is a vector field and $\boldsymbol{\lambda}$ is a vector of parameters. The time dependency of $\mathbf{F}$ can result from different external influences: For example, a change in environmental conditions could be sensed by the cell and induce a different mode of operation, or a communication signal, such as a diffusing agent from neighboring cells, could alter

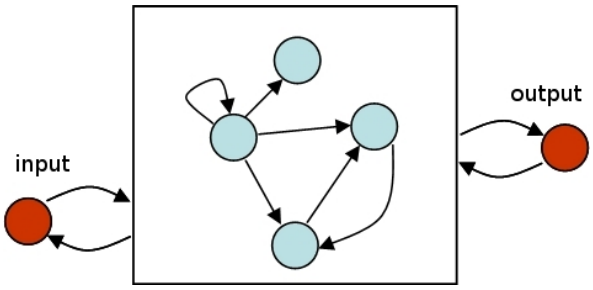


Figure 3.6.: Schematic representation of a gene regulatory network with two observable system variables, e.g. one input and one output. Observable variables have a dynamical behavior depending on the gene regulatory network they are attached to.

the dynamics of a cell's gene regulatory network. Investigating these alterations of phase spaces during development is an exciting task. However, in this thesis, I will focus on isolated cells in constant environmental conditions, such that $\mathbf{F} = \mathbf{F}(\mathbf{X}, \boldsymbol{\lambda})$. Hence, $\mathbf{F}$ describes a time independent, two dimensional vector for each system state $\mathbf{X}$, which represents the direction and magnitude of change in time, whenever the system reaches the state $\mathbf{X}$. An example vector field and a possible resulting system trajectory are given in Figure 3.7. This kind of representation is known as the phase space plot of a system [93].

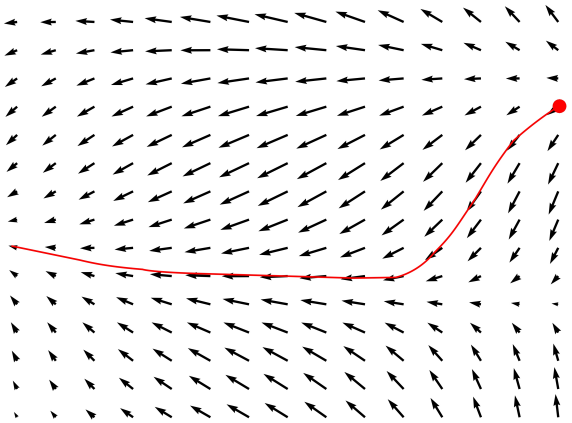


Figure 3.7.: A two dimensional vector field can be interpreted as system phase space. The vector field gives the magnitude and angle of change of the system at every system state. A possible initial system state and the system trajectory which would result from the vector field are highlighted.

Vector field editing relies on creating and changing a vector field by superposition and adaptation of basic field elements $\mathbf{E}_i(\mathbf{X}, \boldsymbol{\lambda}_i)$. The vector field for any system state $\mathbf{X}$ is then given by the superposition of these elements:

$$\mathbf{F}(\mathbf{X}, \boldsymbol{\lambda}) = \sum_i \mathbf{E}_i(\mathbf{X}, \boldsymbol{\lambda}_i). \tag{3.2}$$

To employ vector field editing for control of Artificial Development, I need to define basic field elements that are suitable to create a desired system phase space. Typical elements are proposed in [90] and [91] and can be grouped into singular elements and regular elements. Singular elements are those which create a singularity in the vector field (i.e., a source or a sink) while regular elements do not contain a singularity in their description, and thus generally change the vector field without creating singularities. Two examples are depicted in Figure 3.8.

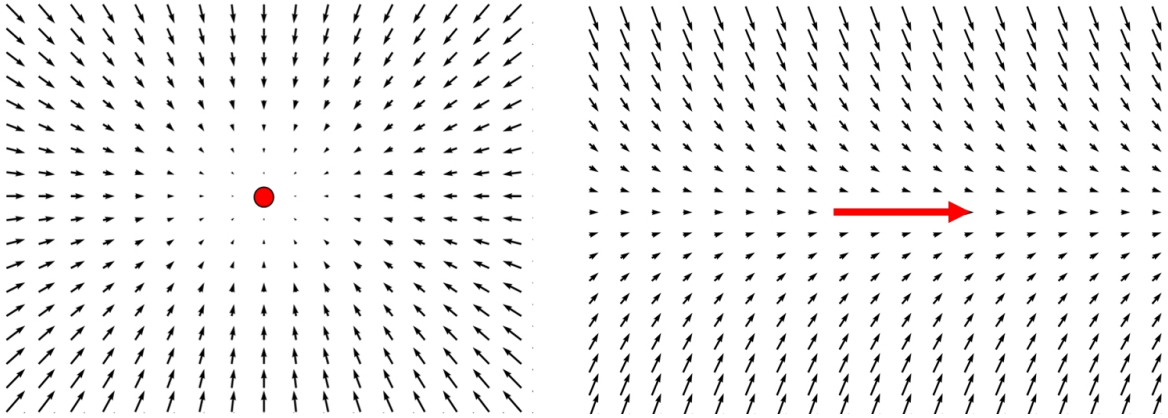


Figure 3.8.: Vector field embryogeny relies on basic field elements. Two basic field elements are employed: a singular element is depicted on the left panel, and a regular element (attachment element) is depicted on the right panel. Point and arrow mark the center and center line of the elements respectively.

In the framework, I adopt the regular element formulation given in [90] and use a simplified version of singular elements. The regular element I use is called attachment element and is depicted in the right panel of Figure 3.8. It creates a flow of surrounding system states toward an attachment line at the center of the phase space. The mathematical formulation to create such an element, where the attachment line is oriented along an arbitrary angle $\theta \in [0, 2\pi]$ is given by

$$\mathbf{A}(x_1, x_2) \quad = \quad \left(\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} - cP(x_1, x_2) \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix} \right). \tag{3.3}$$

Here, $P(x_1, x_2) = -\sin\theta(x_1 - u_1) + \cos\theta(x_2 - u_2)$ and c is a parameter describing the speed with which the flow is attracted to the line and $\mathbf{U} = (u_1, u_2)$ is the center position of the element. Note that for negative c, system states will diverge from the line instead of converging to it. To spatially limit the element's influence for superposition, this attachment element is multiplied by a Gaussian kernel $B(x_1, x_2)$ of width 2σ and center $\mathbf{U}$: $B(x_1, x_2) = e^{-((x_1-u_1)^2+(x_2-u_2)^2)/2\sigma}$. Therefore, the complete formulation of the attachment element is given by

$$\mathbf{V}^R(x_1, x_2) = B(x_1, x_2) \cdot \mathbf{A}(x_1, x_2). \tag{3.4}$$

We create a singular element by applying

$$\mathbf{V}^S(x_1, x_2) = \begin{cases} (\mathbf{U} - \mathbf{X})/\sigma & \text{if } r < \sigma \\ (2/r - 1/\sigma) \cdot (\mathbf{U} - \mathbf{X}) & \text{if } \sigma \leq r < 2\sigma. \end{cases} \tag{3.5}$$

The variable $r := \|\mathbf{X} - \mathbf{U}\|_2$ describes the distance of the system state $\mathbf{X}$ to the center $\mathbf{U}$ of the singular element. The width of the element is denoted by σ. Formulation (3.5) is a coarse piecewise linear approximation of $\mathbf{V}(x_1, x_2) = B(x_1, x_2) \cdot (\mathbf{U} - \mathbf{X})$. I use it, since it is more efficient in computer simulations.

A superposition of η field elements, each weighted by a factor α_i, yields an arbitrarily complex vector field, which can be interpreted as system phase space:

$$\mathbf{F}(x_1, x_2) = \sum_{i=1}^{\eta} \alpha_i \mathbf{V}_i(x_1, x_2), \tag{3.6}$$

where $\alpha_i \mathbf{V}_i(x_1, x_2)$ corresponds to $\mathbf{E}_i(\mathbf{X}, \boldsymbol{\lambda}_i)$ in Equation (3.2), with $\boldsymbol{\lambda}$ consisting of all $\mathbf{U}_i$,σ_i,α_i of all field elements, and additionally θ_i and c_i of the regular elements. Thus, the vector field described in Equation (3.6) constitutes the right hand side of the differential equation

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \mathbf{F}(\mathbf{X}, \boldsymbol{\lambda}), \tag{3.7}$$

which is integrated from $t = 0$ to $t = t_{\max}$ to yield a trajectory of the dynamical system.

The process is implemented in C++. The according sequence diagram is given in Appendix B.

3.3.2. Phenotypic Mechanisms

In this section, I describe the chemical and physical model of cellular interaction developed for the simulations. Both, biological inspiration and computational feasibility are taken into account for modeling. Therefore, the model is simple enough to be computed quickly, which makes it usable in evolutionary computation. At the same time, major developmental mechanisms are simulated, which maintains a certain biological plausibility.

Firstly, I present a simple cellular model, consisting of spheres in 3D space. After that, an extension to the model will be shown, which adds polarization and chemotaxis to the cells. This extension leads to a computationally more expensive calculation; however, it also increases the potential of the developmental process to produce a large variety of coordinated spatial cellular distributions. In a third section, I will outline the way chemical diffusion is simulated.

Physical Cells

The 'sphere cell'-model is inspired by the cellular model presented by [21]. In this model, spherical cells are modeled in 2D or 3D and are defined by a mass, a center

point and a radius. The motion of such a cell is influenced by three groups of factors: firstly, its own properties, secondly, the properties of neighboring cells, and thirdly the boundary of the calculation area, as described in the following.

For each pair of cells, a force is calculated that these two cells exert on each other: $F_i = \Psi(d, r)$. Here, the index i iterates over all pairs of cells in the calculation domain, d is the Euclidean distance between the two centers of the cells, r is the cells radius, and Ψ is a nonlinear function which realizes the following cellular behavior: cells push each other apart, if their distance is less than the sum of their radii (i.e., if they overlap). Additionally, cells exert an adhesive force toward each other, if they do not overlap and are closer than three times their radii. In all other cases, cells do not influence the motion of each other. More formally, Ψ is defined as follows:

$$\Psi(d,r) = \begin{cases} -9 \cdot (d-2r)^2 & \forall\ d < r_1 + r_2 \\ 0.3 \cdot (d-2r)^2 & \forall\ r_1 + r_2 \leq d < 3(r_1+r_2)/2 \\ 0.3 \cdot e^{\left(\frac{d-3r}{2}\right)^2} & \forall\ 3(r_1+r_2)/2 \leq d < 3(r_1+r_2)/2 \\ 0 & \text{else.} \end{cases} \tag{3.8}$$

The constant values are chosen empirically for maximal smoothness of the curve. Figure 3.9 shows $\Psi(d)$ for $r = 1$. The total force acting on a single cell in one developmental time step is the superposition of all F_i: $F_{\text{tot}} = \sum_i F_i$.

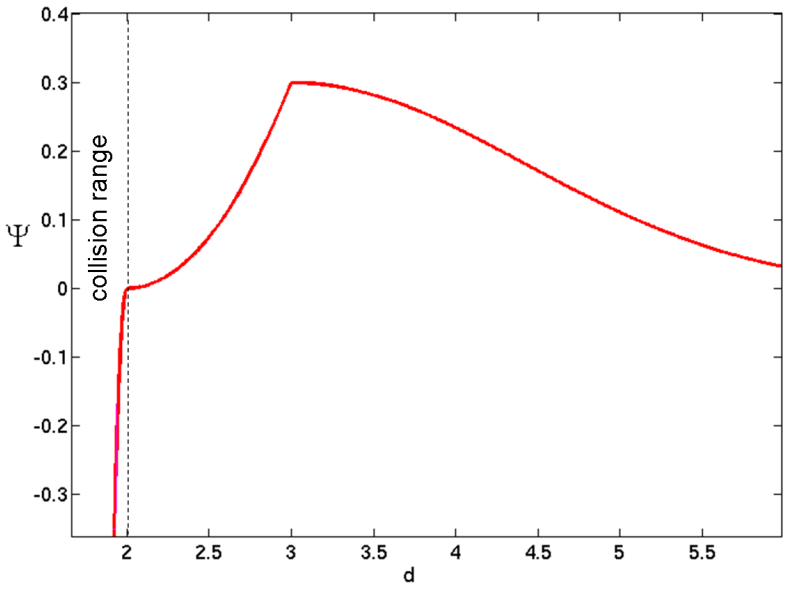


Figure 3.9.: The nonlinear force function that is used to calculate forces between cells. The force Ψ depends on the distance d between the cell centers, and is depicted for a constant cell radius of $r = 1$.

This formulation also allows to introduce differential cell adhesion. Genetically controlled, cells can exert a different attracting or even repelling force on certain cells, usually defined by mutual comparison of cell types for two cells, denoted here as T^1

and T^2. In this case, a genetically encoded factor $\gamma_d(T^1, T^2)$ is evaluated, and multiplied to the second and third equation of 3.8, which become

$$\Psi(d,r) = \begin{cases} \gamma_d(T^1,T^2) \cdot 0.3 \cdot (d-2r)^2 & \forall\ r_1 + r_2 \leq d < 3(r_1+r_2)/2 \\ \gamma_d(T^1,T^2) \cdot 0.3 \cdot e^{\left(\frac{d-3r}{2}\right)^2} & \forall\ 3(r_1+r_2)/2 \leq d < 3(r_1+r_2)/2. \end{cases} \tag{3.9}$$

One iteration step is simulated by discretization and numerical integration of the equation of motion, using the Euler forward method for a damped mechanical system.

The 'polarized sphere cell'-model is an extension to the spherical cell implementation Here, the mutual interaction between cells is implemented as adhesion between simulated charges on the surface of the spherical cells (Figure 3.10). A nonlinear force depending on the sign-difference and the distance of two interacting charges is calculated $F = c_1 \cdot c_2 \cdot e^{\frac{-d^2}{\sigma^2}}$. In this case, sigma is chosen to be $2r$. On overlap, cells behave similar to the 'sphere cell'-model, i.e. they push each other apart as described in the first term of Equation 3.8. Furthermore, cells in the 'polarized sphere cell'-model possess the ability to perform chemotaxis, i.e. to actively move along chemical gradients. This feature is simulated by calculating the local gradient of one of the transcription factor distributions, and adding a genetically scaled force into that direction to the total force F^p_{tot}: $F = F_{\text{tot}} + \alpha \cdot F_u$. Here, F_u is a unit length vector pointing into the direction of the gradient, and α is a genetically determined scalar value belonging to a cell. This value is encoded in a structural unit, and is set if this structural unit is expressed in a cell.

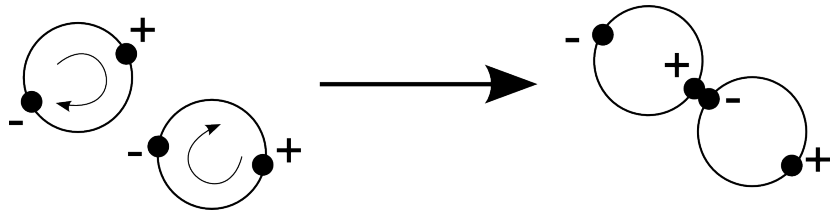


Figure 3.10.: An illustration of the 'polarized sphere cell'-model. Here, two cells with sample charges on their surface are depicted. The charges exert a force on each other, leading to both, cell adhesion and adjustment of cell orientation.

Spatial Communication

If in the model a gene is active which codes for creation of a TF, a value inside the respective SU determines the TF externality η. This externality factor determines the amount of the produced TF that is released into the environment for diffusion: $c_{rel} = \eta \cdot c_{tot}$, while $c_{int} = c_{tot} \cdot (1-\eta)$ is the amount of produced TF that is kept inside the cell. Here, c_{tot} is the total concentration of TF produced by the SU. A released

TF is subject to diffusion and allows chemical signaling with neighboring cells. The production and diffusion of chemicals is simulated on a regular cartesian 3D mesh in the following way: Firstly, a cell that releases a TF calculates the distance to its 8 nearest diffusion-grid vertices. Secondly, according to the calculated distances, the released concentration value c_{rel} is distributed onto the respective vertices. Finally, a diffusion simulation is iterated for one step, using the TF-defined values for the diffusion constant D and decay rate C_d: $\frac{\mathrm{dc}}{\mathrm{dt}} = -C_d c + D\nabla^2 c$. A standard finite difference method is used to solve the equation.

3.3.3. Evolution Strategy

The advantage of using continuous variables in the models described above lies in the possibility of employing a standard (μ, λ) evolution strategy [94] for the optimization of developmental processes. Of course, more advanced implementations of real-valued evolutionary optimization techniques [95] can be used as well. However, because of its simplicity, a (μ, λ)- evolution strategy allows for a focus on the investigation of the developmental models. For the gene regulatory network model presented in Section 3.3.1, I have altered the standard evolution strategy slightly to suit the structure of the genome. Since the relative position of RUs and SUs plays a major role in developmental control, evolution should possess the ability to change these positions. Inspired by biological gene duplication and transposition, I amended the evolution strategy mutation operator with a duplication probability P_d and a transposition probability P_t. Upon mutation, each individual undergoes gene transposition and gene translation with these probabilities, as described in the following:

If a transposition event takes place, two random 'cut'-position markers are determined for a genome. Since the SUs and RUs are the basic units of the genome, such cut positions can only be placed in between two of these basic units; single basic units cannot be fragmented. The genetic region between the two markers is then cut out of the genome, and re-inserted at a third random position. In the case of a duplication event, a similar procedure as for a transposition event is followed, except that the region between the two markers is not cut out, but copied to the new location.

This procedure allows for a dynamical chromosome size, since duplication automatically alters the length of a genome. In my implementation, the gene duplication and transposition mutation replaces the crossover operator for the gene regulatory network model.

4. Graph-based Development

4.1. Introduction

As seen in chapter 3, control of development is achieved using a dynamical system. From a biological perspective, it is straight forward to assume a graph as a representation, inspired by natural gene regulatory networks. However, an engineering and computer science perspective allows us to question if such a representation is optimal with respect to the evolutionary context in which this dynamical system is employed.

Therefore, in this chapter, I will evaluate a graph based approach toward the simulation of developmental control in an evolutionary framework. Firstly, advantages and disadvantages of the graph based method described in Section 3.3.1 will be studied. Secondly, I will exemplify the application of the framework to an engineering design problem. Finally, I will discuss critical aspects of graph evolution, which motivate the alternative dynamics-representation vector field embryogeny.

Parts of the results presented in this chapter have been published in [96, 97, 98, 99, 100].

4.2. Evolving Dynamical Motifs

4.2.1. Introduction

Graph representations of dynamical systems are state of the art in computational intelligence. They are inspired by biology, where both, neural networks as well as gene regulatory networks naturally compose directed graphs. Biological evolution acts on such dynamics representations which naturally leads to the thought that graphs are advantageous for evolutionary processes.

Here, I want to take a slightly different point of view: the graph representation of gene regulation is the only dynamics representation available to biological evolution. We will see in the following that even though biological evolution has employed such dynamics representations, graph structures are sub-optimal for standard evolutionary computation, since they possess a non-trivial relation between structure and dynamics.

The evolutionary process creates networks by small, random changes. We can expect that an evolved network is an accumulation of these changes, where the temporal aspect of evolution plays a role: the benefit resulting from a mutation can generally only be assessed with respect to the remaining network. In the following experiment, we will

see that graph structure can be linked to evolutionary history. At the same time, we will observe that some structural elements are selected for rather unintuitive reasons, such that secondary features of the evolutionary process, which are not related to a fitness function, are reflected in the evolved network structure.

4.2.2. Experimental Setup

Based on the gene regulatory network model described in Section 3.3.1, this experiment serves to investigate the evolution of a gene regulatory network that controls multicellular development toward a stable cellular shape. To this end, the 'sphere cell'-model is employed in a two dimensional environment. Here, stable development means that the growth process must reach a state where cells do not move or divide anymore, i.e., the concentration of the TFs must either have decayed to a value below all activation thresholds, or reached a stable value, which indicates that no further change in gene activity will occur. Finally, the finite number of cells that make up the individual should be located inside a predefined diamond shape centered on the grid.

The evolution of finite growth is formulated as a minimization problem. The fitness f is given by the following equation:

$$\begin{aligned} \eta_i &= \begin{cases} -1 & \forall \ \|\mathbf{p}_i\|_1 \leq 5 \\ 1 & \forall \ \|\mathbf{p}_i\|_1 > 5 \end{cases}, \\ f &= \sum_{i=1}^{N} \eta_i, \end{aligned}$$

where $\mathbf{p}_i$ is a two-element vector containing the position (x_i, y_i) of the i-th cell of the individual in the last time step, N is the total number of cells, and $\| \cdot \|_1$ denotes the 1-norm. In other words, the fitness is expressed by the number of cells outside the diamond shape around the center of the calculation area, minus the number of the cells inside the diamond shape. If the constraints are violated, i.e., if the cells touch the border of the simulation area, or if the growth process does not reach a stable state within a maximum of T_D developmental time steps, a penalty term of +700 is added to the fitness function.

In the experiment, the parent and offspring population sizes are set to 400, and 2000 respectively. The probability of a gene duplication or transpositionis given by $p_m = 0.1$. If such an event takes place, either one is chosen with probabilitiy $p_1 = 0.5$. The maximum number of developmental time steps is set to $T_D = 100$. Development is initialized with a single cell in the center of the calculation domain, and a predefined TF to trigger development. The TF has concentration 0.5 which is constant over both, space and time.

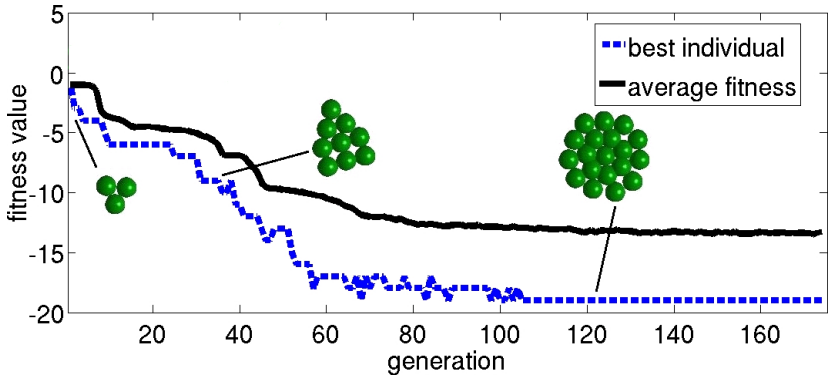


Figure 4.1.: The best (dashed line) and average (solid line) fitness of a typical evolutionary run. The shape of the best individual after convergence to a stable state is shown for three different generations. The average fitness is computed from those individuals only, that do not violate constraints.

4.2.3. Results

The result of a typical evolutionary run is presented in Figure 4.1, where the fitness of the best individual and the average fitness are plotted. It can be seen from the figure that the population stagnates from time to time, before an innovation is found, which leads to a significant fitness increase. Much wider plateaus have also been observed. Successful individuals exhibit the non-trivial behavior to grow toward a stable state during their development. This means that their shape and final state of the gene regulatory network remain constant after a certain developmental time step. Thus, the fitness of an individual is not coupled to a certain evaluation time step as it is usually the case in simulated evolutionary development, but rather to the stable individual that is reached after the developmental process has converged.

To have a closer look at the evolved genetic interactions, Figure 4.2 depicts the static interactions of the gene regulatory network belonging to an individual which results from generation 43 of the evolutionary run. The static interactions can be directly derived from the vDNA of an individual in the following way: An arrow from a SU to a RU denotes that the RU takes part in the activation calculation for that SU. This is determined by the position of the RUs relative to the SUs inside the vDNA. An arrow from a TF-coding SU to a RU denotes, that the label of the respective TF is affine to the label associated with the RU. Therefore, if that TF is produced, it will act on the RU if its concentration exceeds its threshold. If a gene consists of more than one SU, SUs are grouped together (directly adjacent, or with an arrow directly between them). This kind of representation can be useful for an overview over possible interactions, although the generally high number of interactions makes it hard to analyze them in detail. However, the major drawback of this visualization method is that it does not become clear, which interactions really become activated during development. The reason is that the interaction between a TF-coding gene and a RU depends on thresholds and

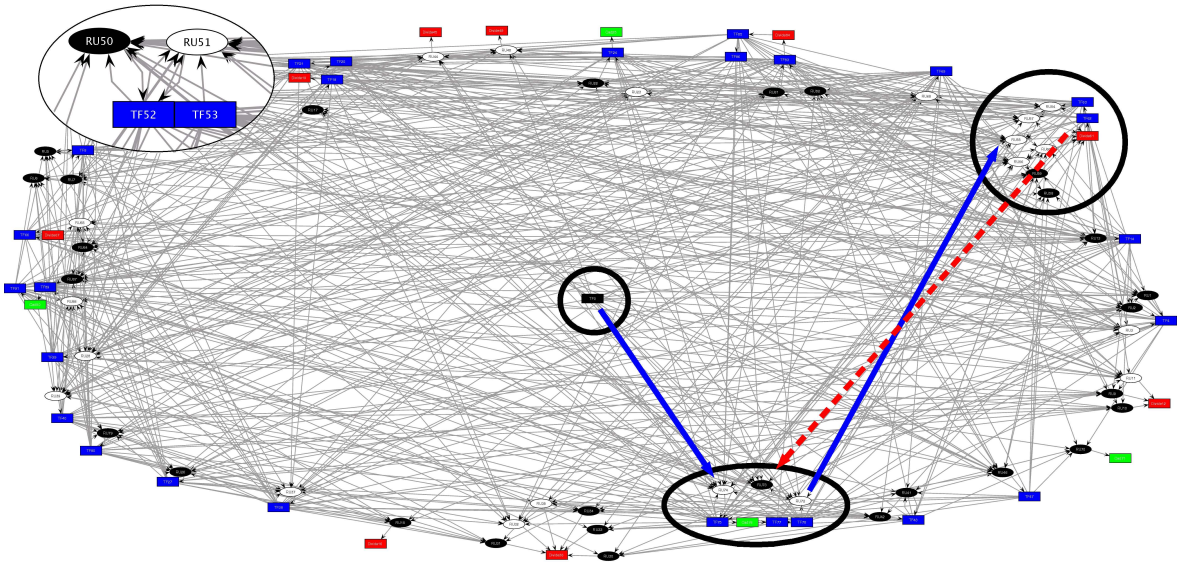


Figure 4.2.: The static interaction network of an individual from generation 43. The pre-diffused TF is placed in the center of the network. A close-up on one gene is depicted in the upper left corner: the gene consists of an inhibitory RU (black ellipses), an excitatory RU (white ellipses) and two TF-coding SUs (blue rectangles). Two interacting genes and the pre-diffused TF are emphasized by bold circles. A positive interaction (solid arrow) from the pre-diffused TF to the lower gene denotes an excitatory connection, which could be the starting point of a negative feedback loop between the two marked genes (the dashed arrow denotes a negative interaction). Note however, that the analysis of the dynamical gene regulatory network reveals that this feedback is not used, because the concentrations of the TFs do not exceed the threshold values.

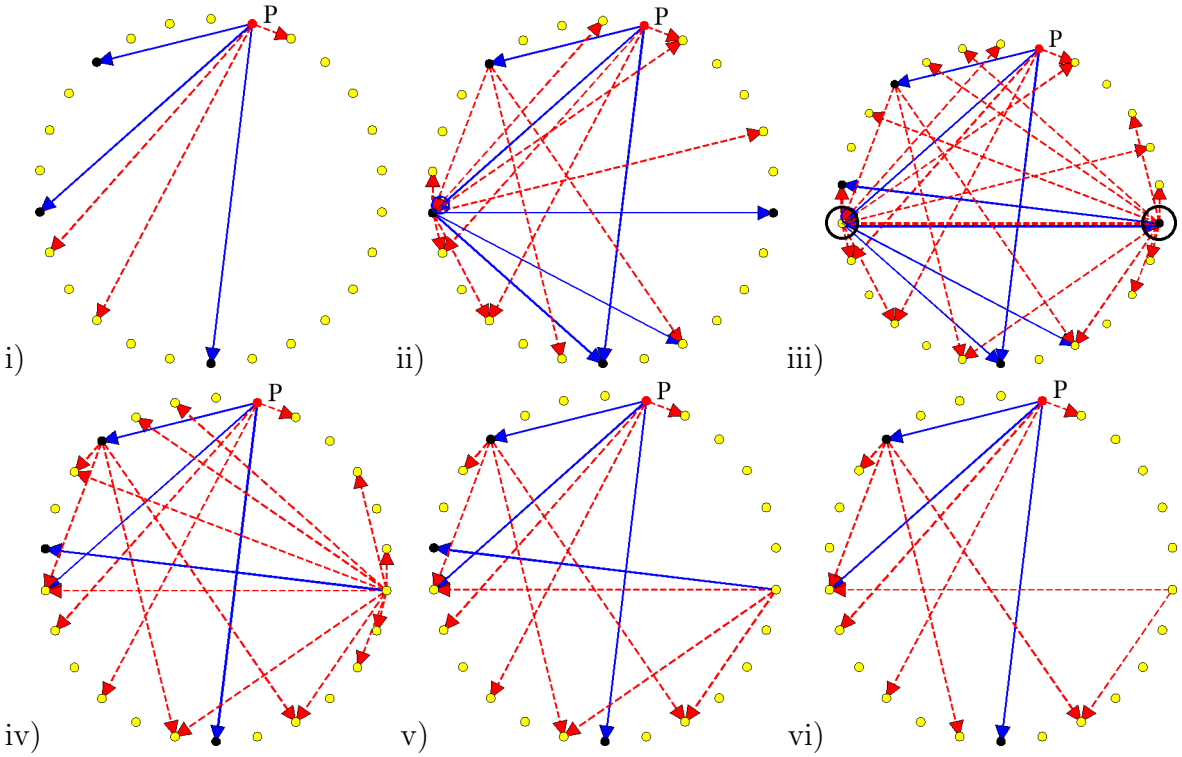


Figure 4.3.: A time series of interactions inside the dynamical gene regulatory network. Each gene is depicted as a small circle. The red point denotes the pre-diffused TF. Active genes are marked as filled circles. The interactions between the genes are either inhibitory (red, dashed arrows) or excitatory (blue, solid arrows). In iii) we highlighted two genes that form a negative feedback loop with an excitatory interaction from left to right and an inhibitory interaction in the opposite direction. Each Figure represents the state of the gene regulatory network in one time step. Note that the static condition for this individual is not yet reached after time step vi).

the concentration of the TF. The concentration depends on the position of the cell which the gene belongs to, the expression rate of the TF and the actual developmental time step. In Figure 4.2, I highlighted a negative feedback loop, which is only one among many (in fact, the closeup reveals a direct negative feedback where the gene acts on its own inhibitory RU). However, the dynamical gene regulatory network analysis described in the following reveals that none of these negative feedbacks are used during the development of the respective individual. Therefore, to get an insight into the real interactions, the missing information on TF-concentrations and time steps need to be included. In Figure 4.3, I depict a time series of network interactions as they take place in the first cell of an individual. Genes are represented by points and arranged in a circle. Since information about TF concentrations in the vicinity of the cell can be obtained for every time step, the real interactions between genes can be shown. Each time step, the interactions are updated according to the changing TF concentrations. The top solid point in Figure 4.3i) denotes the pre-diffused TF and therefore, exhibits initial interactions. From there, gene activation and inhibition can be tracked in each successive time step, from Figure 4.3i) to Figure 4.3vi). Note that this dynamical

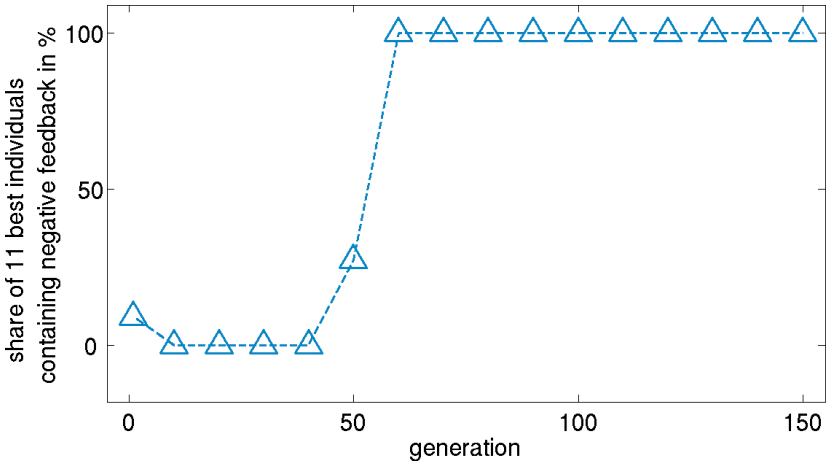


Figure 4.4.: The triangles mark the share of the 11 best individuals which possess one or more negative feedback-loops. The analysis is performed at every 10th generation.

representation of the gene regulatory network can differ from cell to cell. For the experiments, I checked that all cells of one individual reach the point where the gene regulatory network converged to the same stable state. Therefore, the analysis is performed only for the first cell of an individual.

I use the information provided by the dynamical gene regulatory network for negative feedback analysis. A negative feedback can be defined as a closed loop in the directed graph for which the product of all signs is negative. In every developmental time step, I search for such negative feedback loops in the gene regulatory network. In this way, the number of negative feedback loops for all developmental time steps in one individual can be determined. By comparison, we can eliminate the occurrence of the same loop in successive time steps and thus find the number of unique inhibitory feedbacks used throughout the developmental process.

The curve in Figure 4.4 shows the emergence of feedback during the evolutionary run. Since the analysis is computationally expensive, I choose to test the 11 best individuals of each tenth generation for feedback. The curve shows clearly, that negative feedback starts to prevail between the 40th and 60th generation. After generation 60, all 11 best individuals contain feedback loops. On closer inspection, it is possible to track the first occurrence of feedback back to the best individual of generation 44, whose dynamical gene regulatory network is depicted in Figure 4.3. The negative feedback is visible in Figure 4.3iii): an excitatory connection from the highlighted gene on the left side to the highlighted gene on the right side, and an inhibitory connection in the opposite direction.

We may assume that the negative feedback stabilizes the development of individuals against mutation. With a negative feedback loop a TF is possibly self limiting. If the concentration increases beyond a defined threshold, the TF can decrease its own production. If the concentration decreases, the level of self influence is reduced resulting in a stable state. In comparison, a positive feedback loop could only cause a TF to

increase its own production continuously without reaching a stable state. In general, a negative feedback loop in control engineering is a comparison between reference values and output values. Based on the difference the controller designed for this problem can minimize the deviation between reference and output values. Therefore, the system is stabilized toward that target value. In the case presented here, the stabilizing effect is similar except that we do not pre-define a reference value. Instead such a value is system inherent. Therefore, a mutation may cause reference values to change, but as long as the negative feedback loop is not destroyed, a system can maintain its ability to stabilize. One possible effect is that offspring of individuals with negative feedback will be less sensitive to mutations, i.e., fewer lethal mutations will occur. Here lethal means that individuals will not grow at all or will not reach a stable state after the maximum number of allowed developmental time steps. In both cases, individuals are penalized, and not taken into account for further selection. Thus, the number of feasible offspring from an ancestor containing negative feedback loops is higher than the number of feasible offspring from an ancestor without negative feedback. If the fitness of individuals containing feedback is not worse than the fitness of those without feedback, the probability that a genome with feedback is passed on during evolution increases. To verify this hypothesis, I perform a simple mutation experiment with four different individuals: The best individual from generation 44 which uses feedback, its direct ancestor from generation 43 which has no feedback (see the static gene regulatory network in Figure 4.2), the best individual at the end of the evolutionary run and a modified version of the best individual from generation 44. The modification consists of removing the gene from the vDNA that causes the negative feedback (marked in Figure 4.3iii), right circle). Note that such modified individuals still exhibit a stable, finite growth process, thus none of them violate the constraints. The four individuals are mutated 50 times each, for every sample point. Mutation is carried out by adding a random number generated from a zero-mean normal distribution with given standard deviation σ to each value of the vDNA. Thereafter, I count the number of individuals that still produce stable growth without violating the constraints and denote them as successful. Note that feasible individuals with lower fitness than the unmodified ones are also among them. Figure 4.5 shows the results of this experiment.

It is clearly visible that mutations with σ smaller than 10^{-5} affect individuals without feedback much more severely than individuals containing feedback: 100% and 96% respectively of the individuals containing feedback survive, while only 62% and 50% respectively survive without feedback. At $\sigma = 10^{-4}$, feedback is still an advantage, although the percentage of successful individuals has been reduced significantly to 70%. The percentage of lethal mutations with a σ larger than 10^{-3} is similar for all individuals. This might be the result of mutation destroying the negative feedback loop, thus destroying the whole control mechanism that mainly set the different individuals apart. Note that during evolution, σ was in the range between 10^{-6} to 10^{-5} from generation 43 onwards and therefore, in a region where feedback seems to be clearly advantageous.

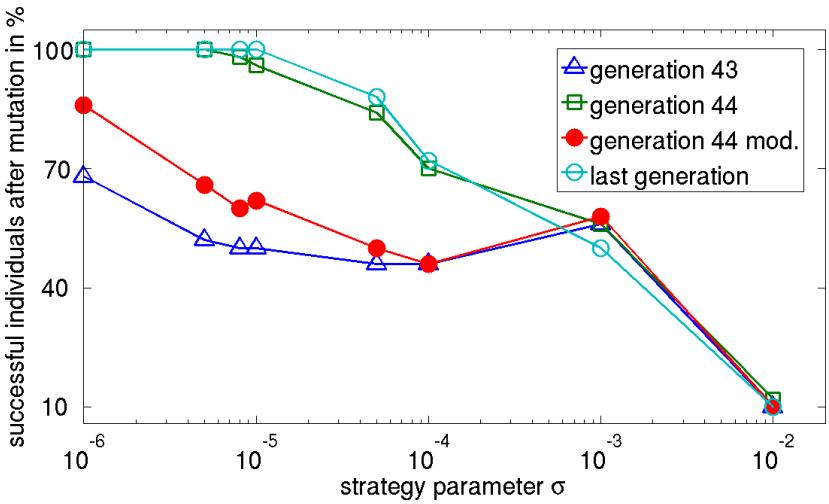


Figure 4.5.: The results of the mutation experiment: Four individuals are mutated 50 times for each strategy parameter σ. The plot shows the percentage of individuals that survived mutation.

4.2.4. Discussion

From the analysis of the experiment, we see that evolution can create graph structures with features that are not directly connected to the performance the graph was selected for. Rather, the process shapes a network that is adapted to have an advantage over other individuals in the evolutionary process, which, as we have seen in this section, includes other features such as higher robustness toward mutations. Note that robustness toward mutations is expected in certain cases to be beneficial for evolution [101].

The individuals with negative feedback show a greater robustness in the mutation experiment than those without feedback. Thus, evolutionary success is a combination of fitness and robustness against mutations. Furthermore, we have seen that static gene regulatory networks in computational models may contain interactions that are never used. Even in biology, when analyzing gene regulatory networks, there are significant structural differences between static network interactions and dynamic interactions, see e.g. [87]. The emergence of negative feedback and its persistence during evolution supports the assumption by Wagner [5], that this kind of robustness is an evolved response to stabilizing selection.

Gathering knowledge about the relation between the evolution of feedback and robustness is also important for evolutionary design, because it could tell us something about the evolvability of the system. Indeed feedback might prove to be an important tool for stabilizing certain useful processes during evolution, while deliberately avoiding feedback mechanisms might enable the evolutionary process to change features easily.

One can speculate why robustness is necessary in the special case of evolution of graphs. Note that, since mutation is the way evolution advances, robustness toward mutation may also be problematic for evolutionary computation. We can guess that robust indi-

viduals are necessary, if mutation has a high probability to yield worse fitness. Then, maintaining actual quality is a sound strategy to achieve survival in the population.

4.3. Developmental System Design

In this section, I will describe a possible application area of graph based evolutionary development, and give examples. Features such as representational power due to the growth process or complex adaptive behavior due to dynamical control render development suitable for the optimization of complex tasks. Especially the independence of genotype and phenotype dimensionalities and exploitation of design regularities could yield a method for optimization of high dimensional designs. The cellular representation allows for an easy creation of complex geometries and inner structures, as we will see below. I have chosen to demonstrate the application of evolutionary development on the field of topology optimization for mechanical stability.

I will show that the complex geometries which can be obtained by multicellular growth, renders Artificial Development suitable for topology optimization in 3D. I employ both, the 'sphere-cell' model and the 'polarized sphere-cell' model for the task of evolving stable, lightweight structures.

4.3.1. 3D Topology Optimization

The experiments presented below deal with the problem of finding a structure defined by a material distribution in a cube shaped, three dimensional design space. The structure should have maximal stability and minimal weight. Here, maximal stability means that the structure should deform as little as possible, when a load is applied to the top of it. Minimal weight is defined by the volume of the structure.

4.3.2. Experiment S: The Simplified Setup

To first show that high dimensional, functional phenotypes can arise from small genotypes in a developmental model, I use a much simplified version of a gene regulatory network, consisting of a simple 'if-then' program which interprets cellular neighborhood information in each cell. Two different cell types exist in these simulations: Cell type A denotes 'void' cells, representing space that does not contain material for stability calculation, and cell type B denotes material cells. Through division, material cells and void cells grow until the whole calculation area is filled. Cell growth, together with cell sorting due to cell-cell physical interaction, shapes inner structures. As denoted above, after the growth process has finished, the positions of the B type material cells form the design for mechanical stress calculation. A type void cells are removed since they represent holes in the structure. The resulting design is discretized into voxels for finite element calculation. Note that for all designs*, the uppermost layer of voxels is

*This includes all designs presented in this section, and the results from reference methods.

set to non-void, such that a 'top-plate' is always part of the designs. A simulated force of 700N is applied to these uppermost voxels. The top-plate allows for proper distribution of the applied force, and the resulting deformation of each voxel in the design is calculated. From this deformation, a bi-linear energy can be derived (see Appendix C), that is a measure for the stability of a structure: we can generally say that if the bilinear energy inside a structure after deformation is low, the stability is high. The minimization of the maximum over all voxels of this bilinear energy will be used as one of the objectives in the optimization runs. The second objective is the weight (mass) of the structure, which is simply given by the number of voxels representing a design.

In the simplified setup of experiment S, each cell can sense the type of its mother cell and the type of the neighboring cells of the mother cell during cellular division. The cell type of the cell that is newly produced is determined using four values in the genome: w_1, w_2, θ, and σ, plus the two inputs, i.e., mother cell type T^1 and the mean value of surrounding cell types T^o as follows:

$$\texttt{eval} = \frac{(w_1 \cdot T^1 + w_2 \cdot T^o) + 2}{4}, \tag{4.1}$$

where T^1 can take the values -1 or 1 for A-type and B-type cells, respectively. T^o is the sum of all T values of cells adjacent to the mother cell divided by their number.

Depending on the state `eval` calculated from Equation (4.1), the celltype (T^d) of the newly produced daughter cell is calculated as described by the following pseudo-code:

```
 if (σ > 0.5){
     if (eval < θ){
          T^d = -1 %produce a A-type cell
     {
     else{
          T^d = 1 %produce a B-type cell
     }
}
else{
     if (eval < θ){
          T^d = 1 %produce B-type cell
     {
     else{
          T^d = -1 %produce a A-type cell
     }
}
```

Development is initialized with a single cell in the center of the calculation area. All cells divide in every time step, yielding 2^n cells for n developmental steps. The 'sphere cell'-model employs differential cell adhesion, as described in Equation 3.9 (Section 3.3.2). The three parameters that specify differential adhesion values between all possible combinations of celltypes A and B (γ_{AA}, γ_{BB} and γ_{AB}) are also encoded in the

chromosome and are subject to evolution. Thus, in total there are seven parameters, w_1, w_2, θ, σ, γ_{AA}, γ_{BB} and γ_{AB} to be evolved.

4.3.3. Experiment C: The Complete Setup

In experiment C, the complete multicellular development framework created for evolutionary computation is investigated. For the same task as in experiment S, the gene regulatory network model described in Section 3.3.1 is employed for the control of development. Genomes are initialized with a random set of 5 SUs and 5 RUs in the beginning of evolution. Furthermore, the 'polarized sphere cell'-model is used for the production of complex cellular arrangements. Inspired by biological development, I define a chemical gradient that remains constant over developmental time to prestructure the design space.

Besides the ability to solve the topology optimization task, I want to investigate the evolution of such pre-defined chemical gradients, in combination with the evolution of the vDNA and the growth process. To achieve this I define several positions inside the cube shaped calculation area, on which the center of different Gauss-shaped gradients can be allocated. These positions are the center of the cube, the centers of all faces and the corners of the cube, totaling a number of 15.

Each of these Gauss gradient has a constant variance of $\sigma^2 = 0.5 \cdot s$ with s being the side length of the calculation area. Gaussians are scaled by $\sigma\sqrt{2\pi}$ to reach a maximum possible concentration of 1 at their center points. Additionally, the i-th gradient is scaled by a respective value h_i, directly encoded in an additional chromosome of length 15 in each individual. Finally, these gradients are superposed and inserted into the calculation area as one predefined TF (see the first panel in Figure 4.11 for an illustrative example of a resulting distribution). Equation (4.2) gives the concentration $c(\vec{x})$ at every point $\vec{x}$ in the calculation area:

$$c(\vec{x}) = \sum_i h_i \cdot e^{\left(-\frac{(\vec{x}-\vec{\mu}_i)^2}{2\sigma^2}\right)}, \tag{4.2}$$

where μ_i is the center of the i-th Gaussian.

The predefined TF is used for chemotaxis, and can also be read by the cells for gene activation. Other TFs that may be released by cells are not used for chemotaxis, and only serve for intercellular communication and gene activity calculation.

4.3.4. Fitness Function

Fitness calculation is based on two objectives: the minimization of the overall weight of the grown structure, and the maximization of the stability against a load which is applied to the top of the structure after development. When cellular development has finished, the resulting cellular design is converted into a voxel-structure and a top

plate is added (see lower right panel in Figure 4.7). The overall weight of the structure is then calculated by summing up the number of voxels that build up this resulting design. To calculate stability, the displacements, resulting from application of the evenly distributed force on each voxel of the top plate, are calculated using the Z88 FE-solver[†]. The maximal displacement describes the 'weakest' point of the design and thus, minimization of this displacement (i.e. the according bilinear energy) represents the second objective.

As evolutionary optimization method, I use the NSGA-II algorithm [102]. Different from single objective optimization algorithms, where the target is usually one optimal solution, NSGA-II produces a set of Pareto-optimal designs, i.e. in my case, designs that trade their weight off against their structural stability. For experiment S, simulated binary crossover [103] and polynomial mutation [104] have been employed to generate offspring. After the offspring population is generated, the elitist crowded non-dominated sorting is used for selecting parents for the next generation. For experiment C, I use a modified NSGA-II algorithm. Since genome sizes of different individuals may vary, it is not straight forward to employ a crossover operation. Therefore mutation of the individuals is implemented as addition of a Gaussian random variable with strategy parameter adaptation [94], without crossover. Also, I use gene duplication and transposition: with a probability of 0.1, a random number of consecutive RUs and SUs are either cut out (transposition) or copied (duplication) and then inserted into the vDNA at a new random position. Population size for all experiments is 100 and evolutions are run for 300 generations.

There are several methods designed specifically to solve topology optimization problems. Two such methods are the Solid Isotropic Material with Penalization method (SIMP) and the Evolutionary Structural Optimization method (ESO) [105]. I used both to create reference Pareto-fronts for the results.

4.3.5. Results

In experiment S, several designs with complex inner structure are found that support the applied load. An example is given in Figure 4.9 where a slice through the material cells can be seen, with the void cells removed. It is well visible that a complex arrangement of cells yields an inner structure. Remember that this structure is encoded by only 7 values, i.e. a seven dimensional genotype space. Differential cellular adhesion plays a role in creating connected structures from material cells (type B).

Experiment C comprises two different simulations: The 'predefined' simulation is an evolutionary run with a predefined, manually chosen gradient for chemotaxis. The spatial function of the concentration c is given in Equation (4.3). Note that the calculation

[†]http://www.z88.org

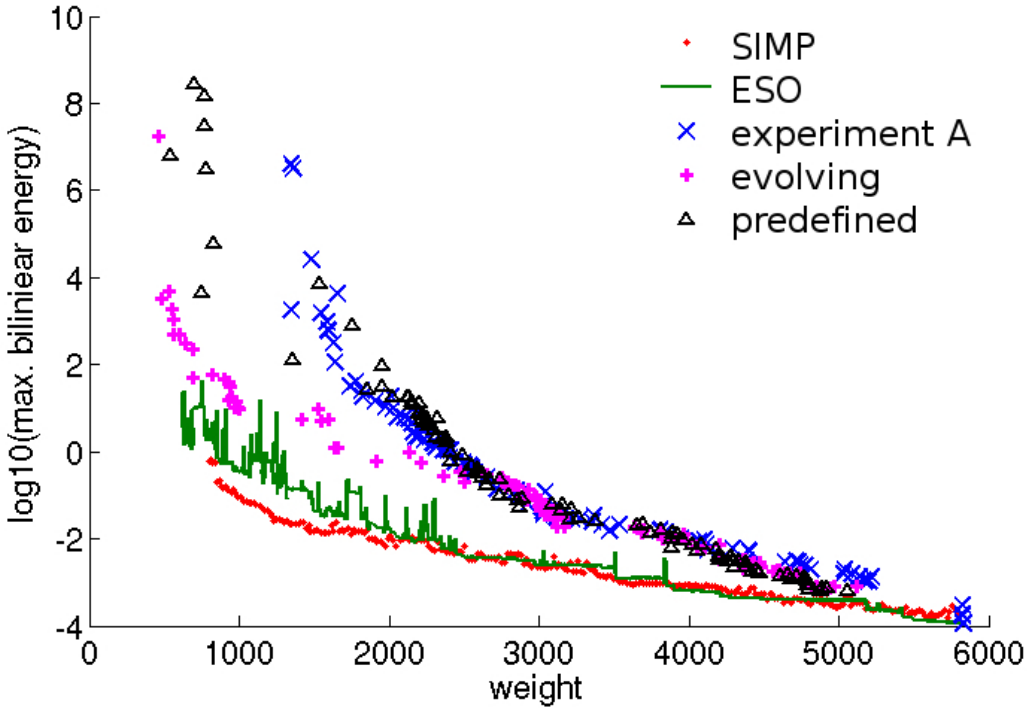


Figure 4.6.: The populations of typical evolutionary runs after 300 generations. Depicted are individuals of the model employed in experiment S, both models of experiment C (using a fixed, predefined gradient and evolving gradients), and the reference solutions from the SIMP- and ESO-algorithms. Note that the whole generation is plotted, not only individuals on the Pareto-front. Also note that for computational reasons, the cell number of the 'polarized sphere cell'-model was restricted to 500 cells, such that a maximum of 5100 voxels will not be exceeded by these data.

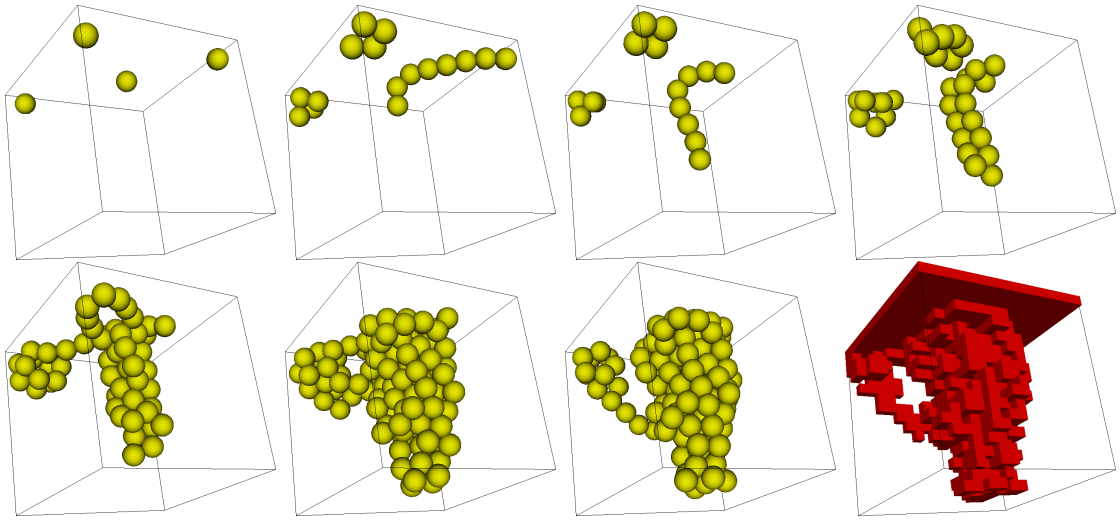


Figure 4.7.: Excerpts from a sample growth of an individual and the resulting voxel grid. Note the added top plate. Sequence starts at upper left and ends at lower right.

area extends from -8 to 8 in each spatial direction.

$$c(\vec{x}) = 1 - e^{\left(-\frac{(\vec{x}-\vec{\mu})^2}{2\sigma^2}\right)}, \quad \mu = (0, 0, -8). \tag{4.3}$$

σ^2 is chosen similar to Equation (4.2). The choice of this gradient is inspired by the shapes that typically result from the SIMP-Algorithm, where 'table-legs' are created at the corners and centers of side-boundaries (Figure 4.8(a)). Note that Figure 4.8b) gives a typical solution from the ESO algorithm, which is less regular, but also shows table leg like characteristics. Using the gradient in Equation (4.3), even without evolutionary optimization it is possible to generate a table-like structure 4.8(d). A simple chemotaxis along the manually defined gradient and continued cell division seems to be sufficient to create the four-legged table. For comparison, in Figure 4.8(c) I also depict a typical solution from the model used in experiment S. Remember that only the dark cells (type B) are used as structure, while white cells represent spacers and are removed prior to fitness calculation. By comparison, Figure 4.8(e) and (f) show an *evolved* design using the manually chosen gradient, and the voxelization of this resulting design.

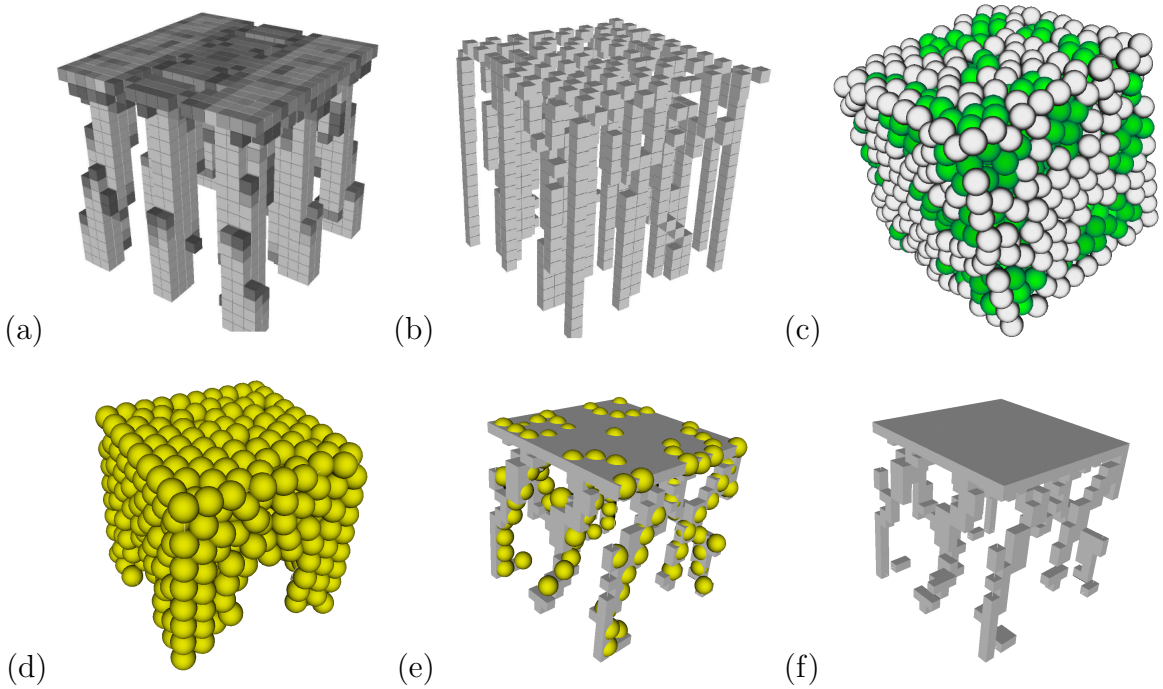


Figure 4.8.: (a) A typical solution using the SIMP algorithm. (b) A typical solution using the ESO algorithm. (c) A typical solution created by the 'sphere cell'-model, where white cells are spacer cells and removed for FE-calculation. (d) An un-evolved, hand coded genome yields this solution with motile polarized cells and a predefined gradient. (e) An evolved solution with motile polarized cells using a predefined gradient, and (f) the resulting voxel grid.

For the second part of experiment C, labeled 'evolving', the chemotaxis gradient is encoded in a chromosome and evolves as described in Section 4.3.3. It is created in

the calculation domain before the development of the individual starts, and does not change during the developmental time.

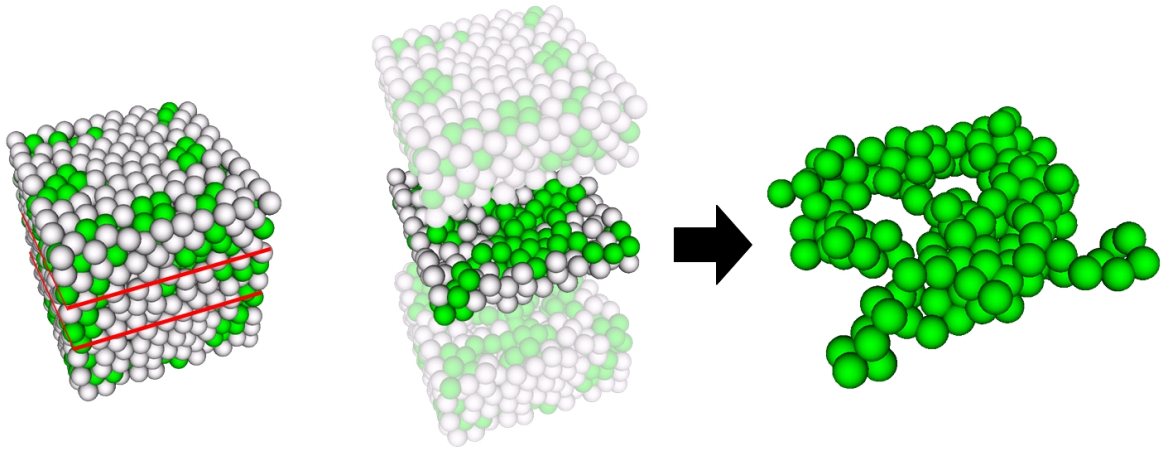


Figure 4.9.: A section through one of the Pareto-optimal solution of experiment S reveals the complex inner structure. For better visibility, the viewing angle is changed and spacer cells are removed. Also, material cells are slightly magnified on the right hand side of the Figure.

Figure 4.6 shows the the fitness of the individuals reached after 300 generations for the different models. It is clearly visible that the model of experiment S has less ability to reach optimal results, especially solutions with less than 1300 voxels are not reachable. For experiment C, a wide variety of designs emerge, which can be seen in Figure 4.10, giving sample phenotypes.

Comparing individuals with predefined and evolved gradient in Figure 4.10 (a) and 4.10 (b), we can see that in the first case, phenotypes are all built up using a common principle: Cells use the clue given by the predefined gradient and move toward the outer area of the cube. The center remains sparsely occupied by cells. Designs only differ by the number of cellular divisions and resulting 'compactness' of the design.

The evolving gradient simulations show a wider range of different designs and growth processes. Both chemotaxis and polarized adhesion can be observed during the growth process of several individuals. Heavy solutions are created by similar design processes: Cells possess equal charges, such that they repel each other. Due to the restricted calculation area, a sponge-like distribution occurs. Most light solutions are built up from a central stem, which branches toward the top of the calculation area. Figure 4.7 gives snapshots of a growth sequence for such a design.

The quality of designs using evolving gradients, especially for design weights smaller than 2500 voxels, is clearly closer to the reference curves created by the SIMP and ESO algorithms as compared to the model employed in experiment S. Interestingly, the

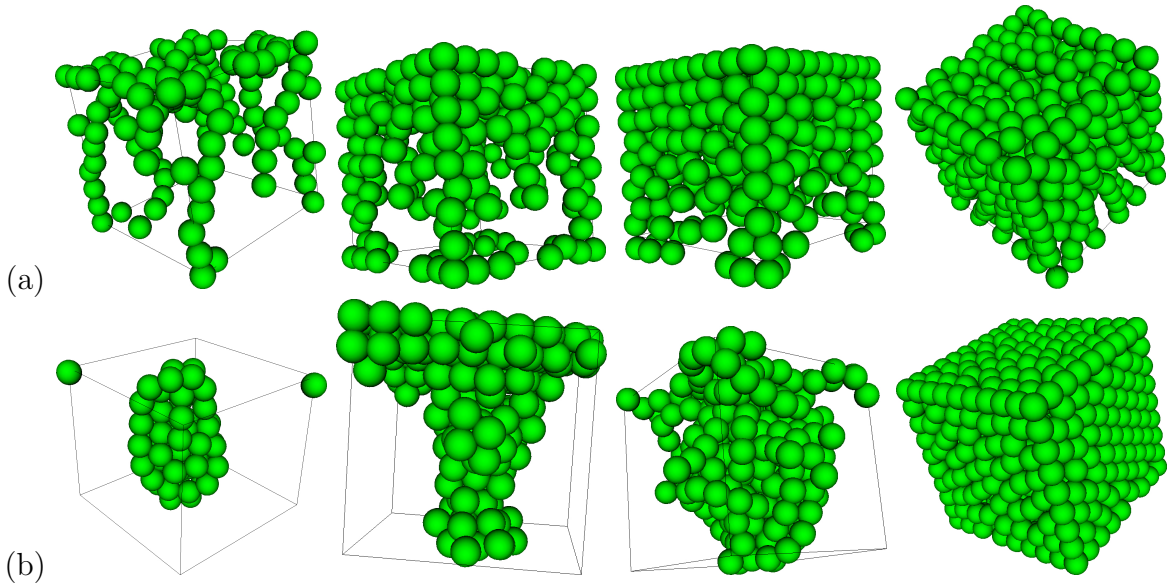


Figure 4.10.: (a) Individuals taken from generation 300 from a fixed, predefined gradient evolution run. Number of voxels for the individuals (from left to right): 755, 1361, 2027, 2866. (b) Individuals taken from generation 300 of an evolving gradient run. Number of voxels (from left to right): 758, 1420, 1665, 2331.

evolving gradient run has the ability to produce individuals that consist of less than 600 voxels and still support the load. Even though the stability of these designs is limited, the SIMP and ESO methods do not cover this design area at all. Figure 4.11 shows two more interesting phenotypes from an earlier generation of the 'evolving gradients' run, which were lost during evolution due to pareto-dominance of other solutions found.

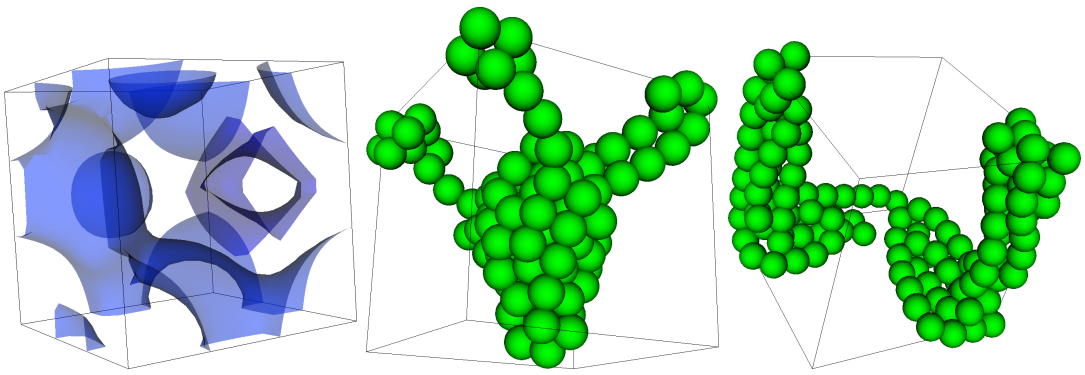


Figure 4.11.: A sample evolved gradient and two solutions that occurred during the 'evolving gradients' run are depicted to demonstrate variety in designs. The left panel shows the iso-surface of the evolved gradient, the center panel gives a tree-like structure that developed 'arms' to carry the load, the right panel depicts a suboptimal solution employing two poles that are connected at the bottom of the structure.

4.3.6. Discussion

The first experiment in this section has shown how a cellular representation using physical interaction between the cells can yield complex structures without the need of a complex genome that controls cell positioning. However, for structural stability, the approach performs worse than methods that are specifically designed for such purposes, like the SIMP algorithm.

The 'polarized sphere cell'-model approach toward structural stability shows better solutions than the simplified 'sphere cell'-model. An evolving maternal gradient for chemotaxis increases the performance. Here, we can see that a coarse pre-structuring that is more directly controlled by evolution, alleviates evolutionary success, a result that will re-occur for vector field embryogeny experiments combining allometry with hierarchy (section 5.3.7). The designs found during evolution are non-trivial. However, the structural stability for these designs is lower than for the reference approaches SIMP and ESO. Interestingly, both SIMP and ESO do not reach solutions as lightweight as found by the evolving gradient polarized cells experiments.

The considerations presented in Section 4.2, combined with the results obtained in the experiments of this section, show that evolving graphs is possible, yet non-trivial, and necessitate special attention toward such features as dynamical motifs and hierarchical formulations. Cellular models and pre-defined boundary conditions can alleviate evolutionary success. However, we can speculate that these features need to be tuned closely toward the respective optimization problem. Let us assume here, that graph representations have a tendency to generally disrupt simple ES-style mutation strategies. Then, an alternative representation of dynamics might be more suitable in the evolutionary context. In the following section, vector field embryogeny is investigated as a more suitable representation for an evolution strategy. A general evolutionary investigation of the method and a comparison to a graph based approach is presented.

5. Vector Field Embryogeny

5.1. Introduction

The experiment in Section 4.2 has shown how graph structure can result from evolutionary processes containing features that are not directly selected for. Although such graph features can yield advantages (mutation is affected so that it does not produce harmful alteration) it can also render optimization problematic: if robustness toward mutation is high, then the optimization performance is low. In the following, I will investigate vector field embryogeny as an alternative representation of a dynamical system to control development for simple developmental tasks.

In the second part of this chapter, a more biology-centered perspective on vector field embryogeny will be taken, to allow for an interface to a classical biological point of view on such principles as hierarchy, allometry and heterochrony.

Parts of the results presented in this chapter have been published in [106, 107].

5.2. Evolving Differentiation

5.2.1. Introduction

For my first investigations, I will employ vector field embryogeny for the simple but typical developmental task of cellular differentiation, and compare its performance to the gene regulatory network model presented in Section 3.3.1. Cellular differentiation is the process in which biological cells determine their future cell type in a coordinated way. Note that morphology after development consists of many different types of cells, even though all cells possess the same genetic information. This differentiation can be traced back to positional information in the early embryonic stages, where different cells experience different chemical environments due to their positions along chemical gradients that are created by the mother individual. This differentiation behavior can be interpreted as a reaction of a dynamical system that is described by the genes, to different inputs (chemical concentrations). The outcome in this interpretation is a cell type.

5.2.2. Experimental Setup

For the experiments, I set up a phase space model as described in Section 3.3.1. I will use a three dimensional system with variables x, y and z, constrained to the interval $[0, 1]$ in each dimension. Thus, $\mathbf{X} = (x, y, z)$ in Equation (3.2). This would correspond to a gene regulatory network where the state of three genes is observable during developmental time.

I then perform the following steps:

1. Determine initial states of these three variables for development (in a biological context, initial values may result from environmental signals or a maternal gradient).
2. Create a phase space in three dimensions, i.e., choose $\mathbf{V}_i(x, y, z)$.
3. Use an evolution strategy to mutate the parameters $\boldsymbol{\lambda}$ and thereby change the vector field representation of the phase space $\mathbf{F}(\mathbf{X}, \boldsymbol{\lambda})$.
4. For individual j, use the differential equation $\mathrm{d}\mathbf{X}^j/\mathrm{d}t = \mathbf{F}^j(\mathbf{X}^j, \boldsymbol{\lambda}^j)$ to create time courses of the corresponding three variables to control its development.
5. Use the evolution strategy to select fit individuals for reproduction and repeat steps 1 to 5 until a stop criterion is met.

To investigate cellular differentiation, the system state $\mathbf{X}$ is interpreted as the expression level of three genes in a certain cell of an individual. The cells that belong to the same individual share the same phase space, but have different initializations of x and y. I define z to correspond to the cell type and initialize it at $z_0 = 0.5$ for all cells, representing a non-differentiated state. The cell's environmental information is encoded in x and y and can be interpreted as maternal factors, similar to those found in the early Drosophila embryo [108]. Cells do not divide or interact. For visualization, cells are positioned on a 2D lattice, where the coordinate of a cell is chosen according to its initial state of the genes x and y. Two different resolutions are used for experiments: 2x2 and 4x4 cells. Therefore, $x \in \{0, 1\}$ and $y \in \{0, 1\}$, or $x \in \{0, 0.33, 0.67, 1\}$ and $y \in \{0, 0.33, 0.67, 1\}$ for the respective experiments.

The phase space of an individual is evolved by changing the key parameters of a fixed number of field elements. These key parameters for singular elements are $\mathbf{U} = (u_1, u_2, u_3)$, α, and σ. $\mathbf{U}$ represents the position of the element in 3D space, α is its strength and σ its width (see Section 3.3.1). For an attachment element, three additional parameters are encoded: θ, ϕ, and c. θ and ϕ are the two angles describing the direction of the element in 3D space, and c is the relative speed of attachment (see Equation (3.3)). The resulting system equations are solved for each cell by a Runge-Kutta method of order 4. The maximum simulation time is set to $t_{max} = 500s$, with a step width of $0.25s$ and 8 sub-iterations per step. I expect system states to have reached a stable state before the simulation time reaches 500 seconds. However, if this is not the case, solutions are not penalized. Simulation is terminated when either the maximum time t_{max} is exceeded or when the system state does not vary more than $\epsilon = 10^{-12}$ in two consecutive steps. A standard evolution strategy [94] is employed, with

population sizes of 15 and 100 for parent and offspring population respectively, with a single strategy parameter with step size adaptation. A more sophisticated evolution strategy could be applied ([95] gives a comprehensive overview), however, the standard version is very robust and its performance is sufficient for the purpose. The initial strategy parameter is chosen to be $\sigma_{init} = 0.1$. The fitness F is calculated by taking the squared distance between the cell types of the n cells belonging to an individual after development, and a given target vector ρ: $F = \sum_i (z_i - \rho_i)^2$. Therefore, the task is a minimization task, and optimal fitness is reached if $F = 0$. Note that for the experiments presented in the following, the maximum value for F is the number of simulated trajectories, i.e., 4 for the 2x2 and 16 for the 4x4 runs since both, ρ_i and $z_i \in \{0, 1\}$. Twenty evolutionary runs are performed per experiment.

Note also, that the adaptation of the framework to a specific developmental research problem is basically similar to setting up a graph based method. The fundamental difference lies in the evolution of the system: while both approaches possess the ability to represent complex phase spaces, vector field embryogeny is expected to create a more causal relation between mutation strength and phase space change. We will quantify this in the following chapter.

For comparison, I employ the gene regulatory network model (see Section 3.3.1) for the same evolutionary tasks. Here, I choose the following experimental setup, and also perform 20 runs per experiment: A genome size of 20 initial regulatory units and 20 initial structural units is employed. These are empirical values which I have used in various simulations and consider suitable for the given task. Note that through duplication, the number of units are adapted by the evolutionary process. Two time-constant pre-diffused gradients with linear distributions along the two axes are defined, which represent the x- and y-coordinate of the experiment. Cells are positioned on the x-y-plane in the same manner as in the phase space experiments. Cells do not communicate, i.e., signals do not diffuse in space. The concentration of the first genetically created transcription factor, i.e., the activity of one gene, in each cell is used for fitness evaluation, following the fitness function given above. This setup creates a task for the gene regulatory network based system, which is comparable to that of the vector field embryogeny. From experience with the gene regulatory network based system I know that good solutions are often lost through mutations. Also, the standard selection pressure is usually too high and yields early convergence to local optima. Therefore, it is necessary to slightly deviate from the standard evolution strategy employed in the vector field embryogeny framework, to achieve comparable results: Firstly, three elitists [94] are employed in the evolution, i.e., the three best individuals of a generation are carried over to the next generation without mutation. Secondly, parents population size is increased to 40 to reduce selection pressure. Finally, the 4x4 cells task is run for 200 instead of 100 generations.

5.2.3. Results

The first experiments show the feasibility of vector field embryogeny to evolve cellular differentiation. The possibility to generate an arbitrary cellular distribution in a 2x2

cell grid is investigated. To this end, three target patterns are defined: 'one point', 'half', and 'xor' (see Figure 5.1). Note that for the trivial solution of $z_i = 0.5$, i.e., no movement in phase space, a fitness value of $F = 1$ would be the result for all targets. I denote fitness values below $F = 0.0025$ as optimal. The experiments investigate the influence of field element type and field element number on the evolvability of the system: I first perform evolutionary runs that employ singular elements or regular elements exclusively. The number of elements in these experiments is varied between 2 and 6.

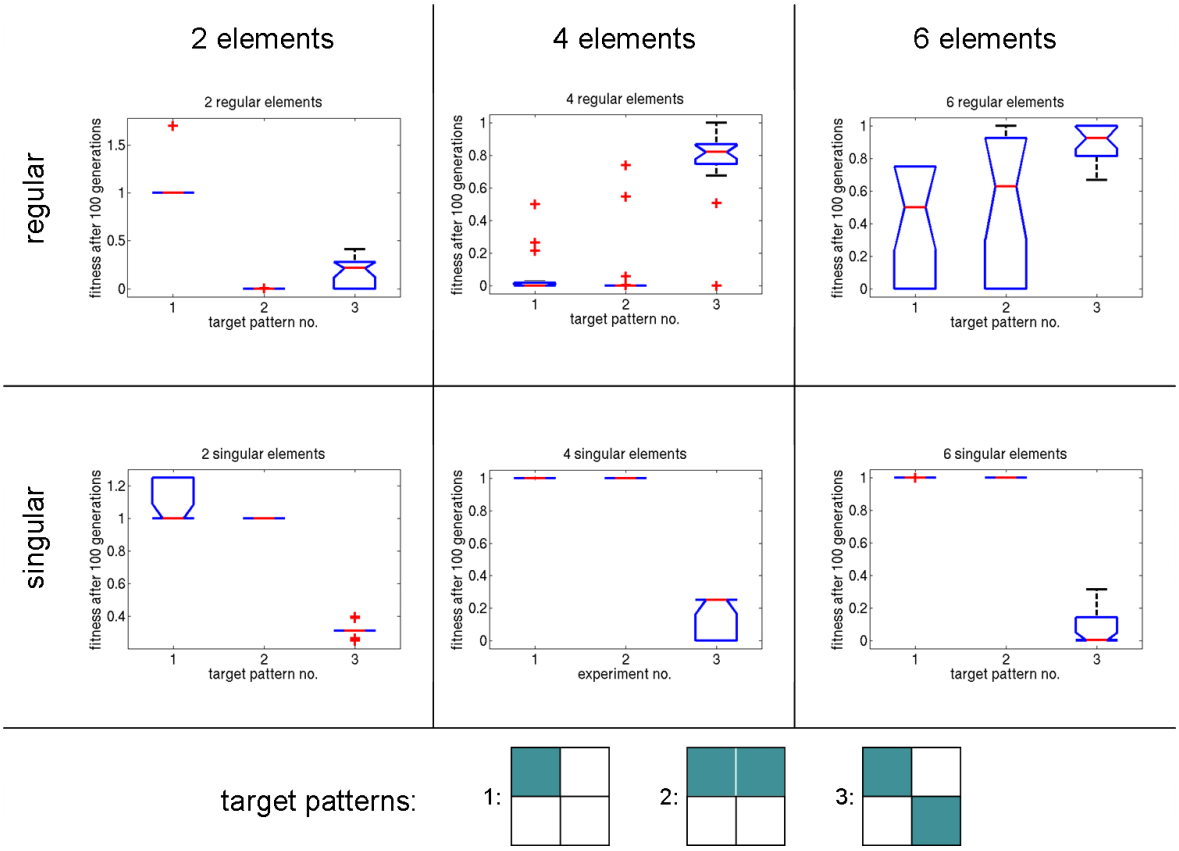


Figure 5.1.: Results from the differentiation experiment, summarized in 6 plots: experimental setups with different numbers (two, four, and six) and types (singular and regular) of basic field elements are compared. The experiments were conducted using three different target patterns: 'one point' (1), 'half' (2), and 'xor' (3).

Figure 5.1 gives the results of the experiment after 100 evolutionary generations. For the target patterns 'one point' and 'half', using 4 regular elements shows good performance, while for the target pattern 'xor', using 6 singular elements yields the best results. Interestingly, using singular elements only leads to early convergence of the 'one point' and 'half' runs, while using regular elements yields suboptimal performance for the 'xor' run. Generally, using 6 regular elements yields evolutionary runs without convergence after 100 generations, which can be seen in the variance of the fitnesses.

It seems that the exclusive strategies with one type of field element only, are suitable for certain characteristic differentiation targets only.

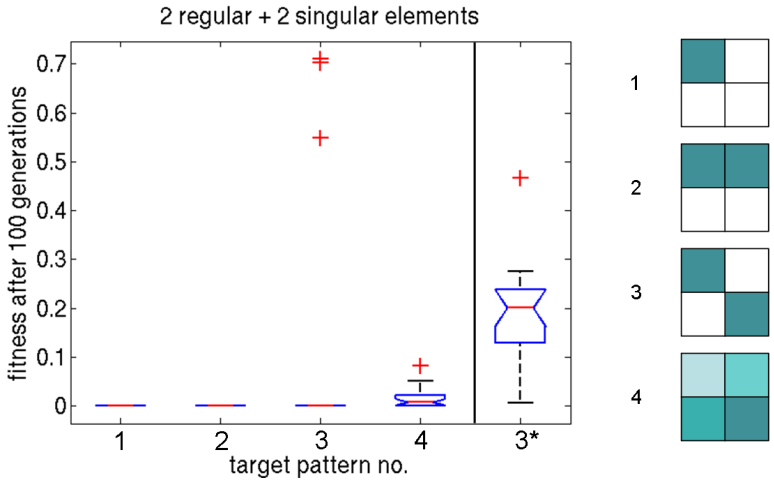


Figure 5.2.: The concluding experiment of the differentiation task. Two regular elements and two singular elements are combined, and applied to the four target patterns: 'one point' (1), 'half' (2), 'xor' (3), and 'random' (4). For comparison, the performance of a gene regulatory network based approach for the 'xor' target pattern is shown on the rightmost panel (3*) with similar evolutionary setup (see text for details).

Therefore, I create a setup with 4 field elements in total, where I combine two regular and two singular elements. Results are shown in Figure 5.2: A high fitness for all target patterns is achieved. The 'one point' and 'half' experiments are successful, while the 'xor' experiment has 3 outliers apart from all other runs reaching optimal solutions. In the rightmost column, results from the gene regulatory network based approach toward the 'xor' target pattern are shown. Additionally, results from a random target pattern run, using the same setup are presented, where each evolutionary run has a target consisting of four values independently drawn from a uniform distribution in the interval $[0, 1]$. To visualize phase space trajectories, the upper panel of Figure 5.3 depicts trajectories of a successful individual of the 'xor' run and its final differentiation pattern.

These experiments show the feasibility of the vector field embryogeny approach to cellular differentiation tasks while on average, the gene regulatory network approach converges to lower quality solutions for the 'xor' target. The setup is now changed to a more complex task in a 4x4 grid where the 'H' target pattern is used (see Figure 5.4, upper right panel). Note again, that for the trivial solution of $z_i = 0.5$, i.e., no movement in phase space, a fitness value of $F = 4$ would be the result for all targets. I denote fitness values below $F = 0.01$ as optimal. In these experiments, the number of field elements is increased to 4 regular and 4 singular elements per experiment. Results are depicted in the left column of Figure 5.4. While no run reaches global optimum,

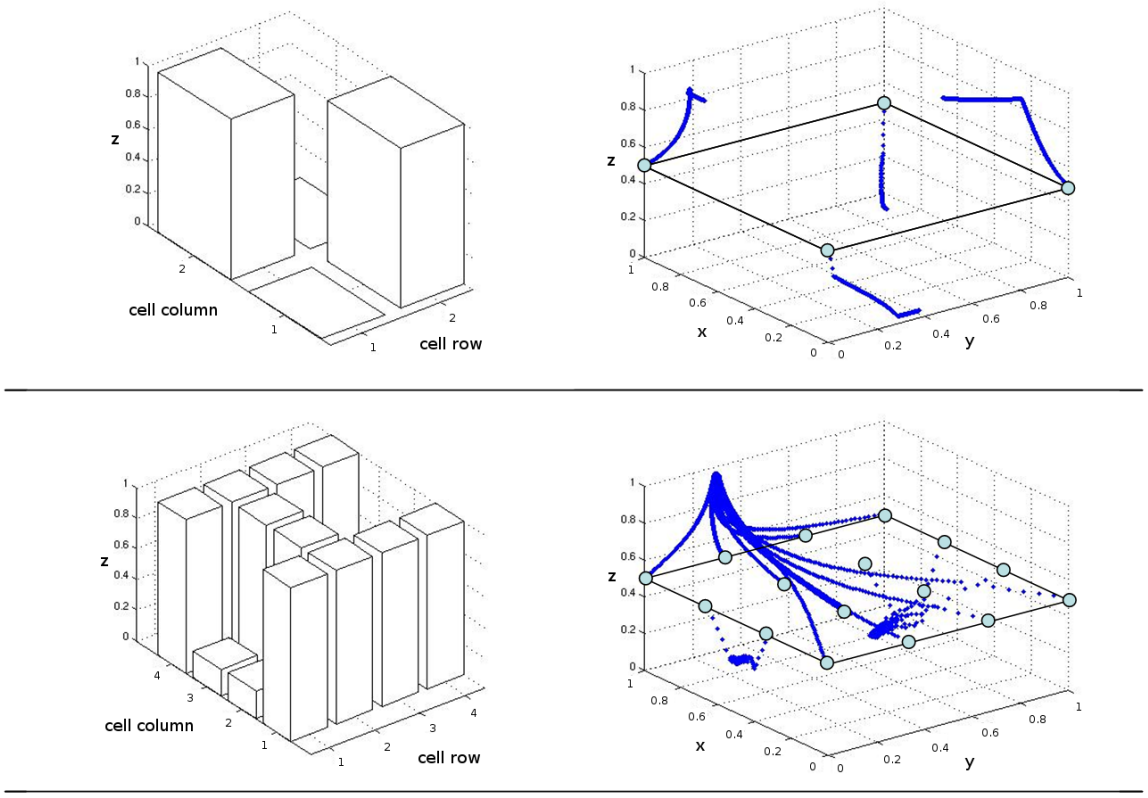


Figure 5.3.: Final cell-type distributions of successful individuals. Upper panel: The 'xor' pattern result and the corresponding evolved phase space trajectories for a run using two regular and two singular elements. Lower panel: The 'H' target pattern result of the best run using the reference setup, with the corresponding phase space trajectories.

the best individual's phenotype resembles the target and is shown in the lower panel of Figure 5.3, together with its trajectories in the phase space. In the following, I will refer to this experimental setup as the reference setup.

The right column of Figure 5.4 gives the performance of the gene regulatory network approach toward solving the same problem. Clearly, the gene regulatory network method is not able to generate the given target, despite the fact that twice the number of evolutionary generations are available. The mean fitness has a magnitude comparable to that of the two worst reference setup runs.

5.2.4. Discussion

We have seen that vector field embryogeny is suitable as an alternative representation for gene regulatory network dynamics. Although just two simple basic field elements were employed, an advantage over the gene regulatory network formulation could be seen. An interesting future research work for vector field embryogeny would be to

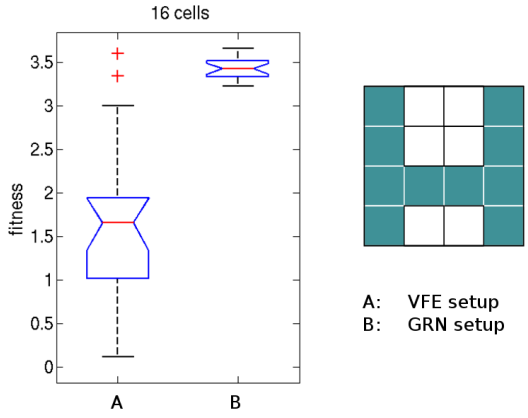


Figure 5.4.: Results from the differentiation experiment with 16 cells. The 'H'-target pattern and the results from the different setups are presented. The results of the reference setup, the allometry setup and two hierarchy setups (pre-defined and evolving weak symmetry) are depicted, as well as the result of the respective gene regulatory network based approach.

identify more useful basic elements, and assemble an element-library, possibly with grouping into different element classes for different kinds of problems. Also it would be interesting to research a complete field representation, i.e., a set of field elements that represent an arbitrary field, such that an assessable representation error exists if parts of the representation are neglected, in a way comparable to Fourier or Taylor series.

Throughout this section, I motivated vector field editing as an abstraction of biological gene regulatory networks. It is straightforward to map a network via its phase space onto a vector field. However, the opposite direction is difficult to accomplish: A given vector field in general can not easily be converted to a graph. This is mainly due to the fact that desired dynamics cannot be generated easily, e.g., by a superposition of graph-features. Therefore, the basic field elements of vector field editing are not to be seen as equivalent to network motifs [46]. Note that the search for functional sub-units of graphs that govern graph dynamics is still a debated topic [109].

Apart from evolvability, i.e., the performance in an evolutionary optimization context, the simplicity of a representation is important for analysis and easy understanding. I believe that a vector field embryogeny style spatial illustration in up to three dimensions of dynamical system properties, and especially of mutational changes in dynamics, is more intuitive than inferring system behavior changes from graph structure changes. Note that this does not necessarily decouple the observations resulting from vector field embryogeny from regulatory networks or biology: vector field embryogeny merely represents evolution of dynamical behavior on a systemic level, where both, regulatory mechanisms and evolutionary processes, are modeled abstractly, and thereby provide a different point of view on their respective biological counterparts.

Dynamics of gene regulatory networks account for many of the desirable features of organisms, such as robustness and flexibility during embryogenesis, which both make species evolvable. Many intricate regulatory interactions have been selected for in

biology to create these features. However, the exact underlying processes and driving forces are unknown. The concept of motifs has been successful to reveal structural coherence in the composition of biological regulatory networks. Still, the coupling between the embedding of a motif in a graph structure and the resulting changes of the dynamics remains unclear. The phase space approach suggested in this work concentrates on the more abstract level of the evolution of the dynamics of a regulatory system. It neglects its conceivable structural realization. This kind of explanatory level of dynamics combined with the analysis of graphs could enable us to understand the mapping between graph structure and phase space. I believe that this could hold the key to further our understanding of the evolution and function of biological regulatory networks.

5.3. Higher Level Principles of Development in Vector Field Embryogeny

5.3.1. Introduction

In the previous section, we have seen that vector field embryogeny can be used to control development in the place of a gene regulatory network formulation. From a computer science perspective, this is beneficial since standard evolutionary optimization technique can be employed using this representation. From a biology perspective, we have to ask ourselves about the suitability of such an abstract representation to model biological phenomena. In this section, the integration of abstract biological mechanisms into the vector field embryogeny framework will be discussed, where examples will be given with such features as hierarchy, heterochrony and allometry. I will refer to these as higher level principles of development, since they are abstract building blocks of biological embryogenesis. While gene regulatory networks can exhibit all these features, they are not directly included in the vector field embryogeny framework. The following section will show that it is straightforward to augment vector field embryogeny with such features, and that we can deliberately choose to include or reject these features, which allows a systematic investigation of the advantages or disadvantages that come with them.

5.3.2. Hierarchy in Vector Field Embryogeny

During biological embryogeny, organisms go through a phase of hierarchical structuring [1]. A spatial hierarchy develops over time, such that early signals in the embryo create a coarse structuring, while later signals are used to create more and more details of the final morphology. Doursat [110] has used such a hierarchy to create spatial differentiation during an artificial growth process. I want to show that it is possible to integrate a similar mechanism in vector field embryogeny.

Consider the problem formulation of cellular differentiation presented in Section 5.2, with a three dimensional phase space $n = 3$, $\mathbf{X} = (x, y, z)$, where z gives the cell type and x and y the variables which carry initial conditions. Let us assume that we simulate four cells of an individual such that four trajectories, $\mathbf{X}^0, \mathbf{X}^1, \mathbf{X}^2$ and $\mathbf{X}^3$ will be simulated in the phase space, with initial conditions $(x_0, y_0, z_0)^0 = (0, 0, 0.5)$, $(x_0, y_0, z_0)^1 = (0, 1, 0.5)$, $(x_0, y_0, z_0)^2 = (1, 0, 0.5)$ and $(x_0, y_0, z_0)^3 = (1, 1, 0.5)$, i.e., the four corners of the x-y-plane at $z = 0.5$. The common phase space represents the common genetic control of the cells of an individual. The implementation of spatial hierarchy in this three dimensional vector field embryogeny can be based on a subsequent subdivision of initial system positions on the x-y-plane, where each system state $\mathbf{X}$ has the ability to divide into four 'daughter'-system states to constitute the next hierarchical stage. Thus, if the initial stage consists of four initial system states, the second stage will contain 16 and the n-th stage 4^n system trajectories. Each hierarchical stage has its own phase space, with own evolving field elements. When a system state is subdivided, its daughters are initialized such that the 'cell type' variable, i.e., their z positions in phase space, are equal to the final 'cell type' of the mother cell, while x and y are chosen to be the corners of the x-y-plane again in the phase space of the respective hierarchical stage (see Figure 5.5). Since one hierarchical stage has only one phase space, all cells belonging to one stage share the same phase space. In approaches using multiple stages of hierarchy, this allows a reduction in parameters: if a field element can be described by v variables, and η elements are used in each of the s stages, the total number of variables to describe one individual solution is $s \cdot v \cdot \eta$, while the number of cells that can be described with this setup amounts to 4^s.

This setup also allows an explicit integration of weak symmetry constraints, i.e., for symmetry with variation. The right panel of Figure 5.10 shows an example of a lateral symmetric target pattern. The left half of the target can be reproduced by the same mechanism which creates the right half, if the underlying coordinate system is mirrored. Let the initialization of the coarse stage system be $x_0 \in \{0, 1\}$ and $y_0 \in \{0, 1\}$, and $z_0 = 0.5$, i.e., one cell in each corner of the x-y-plane, each with the same type, as depicted in Figure 5.5. The first stage of development yields two pairs of cells, each consisting of system states that reach the same type z during development (Figure 5.5, upper panel on the right). Accordingly, the four trajectories of the next stage for each cell pair will start at the same height z. To illustrate the symmetry in the resulting cell states (Figure 5.5, lower panel on the right), let us consider Figure 5.6: since the x and y coordinates are initialized equally, the fine grained solutions would be identical for those cells that have reached equal z during the first stage (Figure 5.6 upper panel). If the initial x coordinates are mirrored however, the solution will be symmetrical (Figure 5.6 lower right panel and Figure 5.5, lower panel). Note that perfect symmetry is only facilitated by this formulation, but not enforced: if system states in the first hierarchical stage converge to z values that differ between left and right, the initial states for the second hierarchical stage are distinct and can therefore yield different trajectories with different end points, which eventually results in non-symmetric patterns. Hence, in this setup we adopt the term weak symmetry constraints.

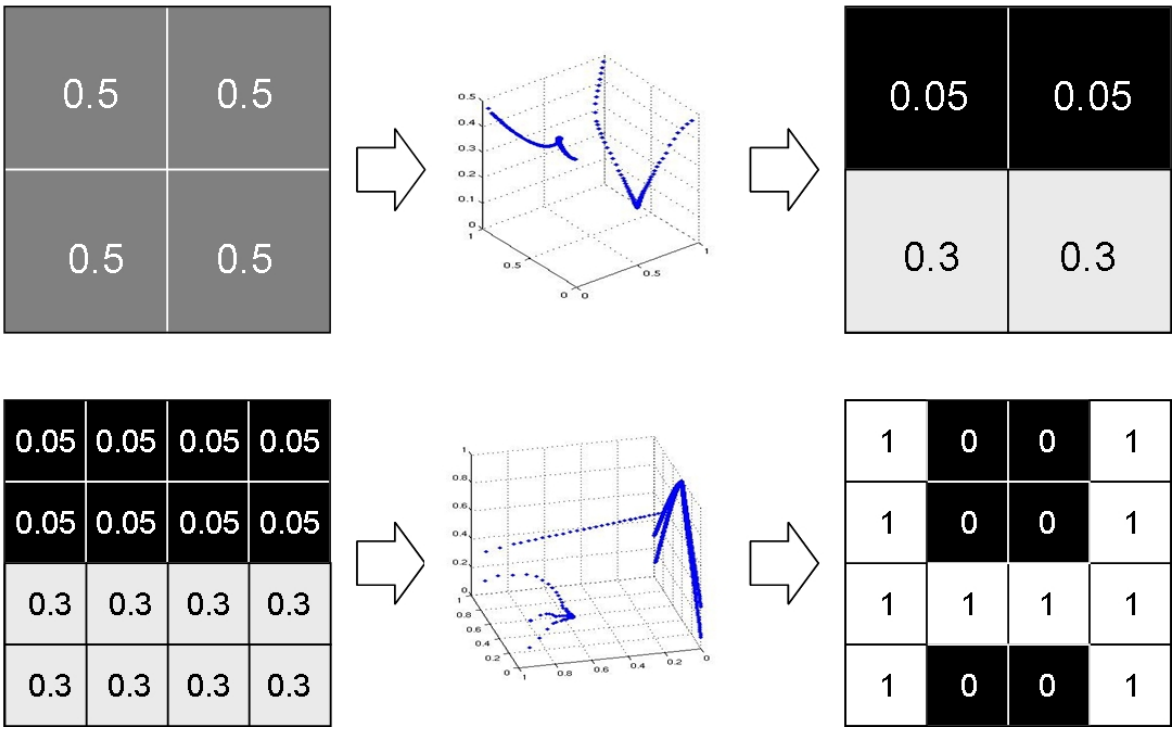


Figure 5.5.: Two hierarchical levels of development. Upper panel: four cells are initialized with cell type $z_0 = 0.5$ and $(x, y) \in \{0, 1\}$. In the 2D representations, z-positions are mapped onto the $x - y$ plane according to the cells' initial states (i.e., the lower left square gives the z-value for a cell with $(x_0, y_0) = (0, 0)$, the lower right square gives the z-value for a cell with $(x_0, y_0) = (0, 1)$ and so on). State trajectories in the phase space are given. After the system states of the first stage have reached their final position in phase space, their z-values mark the initial cell type of the four respective cells in the second, fine grained level (lower panel). These are initialized with $(x, y) \in \{0, 1\}$ again for each coarse level cell. Note the symmetric coordinates in the second level (see also Figure 5.6).

5.3.3. Heterochrony in Vector Field Embryogeny

Biologists use the terms heterochrony and allometry when describing temporal characteristics of developmental processes [12]. On the one hand, heterochrony is describing the evolutionary change in rates and timing of developmental processes. This means, an explicit view on the time axis of development is employed. On the other hand, allometry refers to time implicitly, describing covariation of size and shape of features of an embryo (see e.g. [12, 111, 112, 113]). Heterochronic change can e.g. be observed in the development of the vertebrate eye: the initiation of eye development is earlier in lizards than in mammals and are thus proportionally larger [1].

Interestingly, in biology, heterochrony seems to be under direct evolutionary influence. For example, genes of *C. elegans* have been identified which control heterochrony directly [114]: mutations of these single specific genes result in a change of relative timing of cell differentiation. Furthermore, these mutations have also been observed among natural species. Thus, observing direct evolution of heterochrony in an artificial system

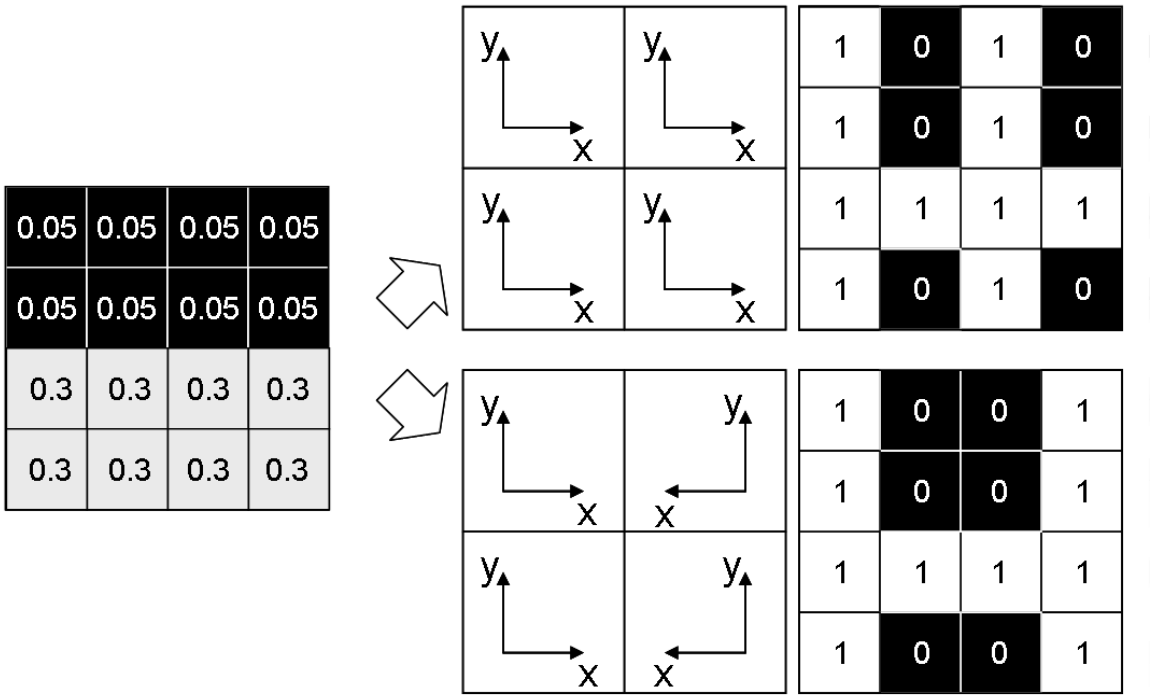


Figure 5.6.: Non-symmetrical and symmetrical setup of the experiments: Explicitly changing the coordinates of the initial state of a fine grained level allows exploitation of symmetry. The upper panel shows the resulting final cell-type distribution from the experiment presented in Figure 5.5, if the weak symmetry constraint is not employed. The lower panel shows the same result using a weak symmetry constraint.

might also allow for new insights into biological evolution.

Heterochronic changes are changes in the timing of developmental processes. In a vector field embryogeny interpretation, developmental processes are represented by basic field elements that build up the phase space. Thus, I interpret heterochrony as allowing evolution to choose a time frame during development, in which such an element is active, i.e., has influence on the system trajectory. Outside of this timeframe, the element is without influence. Therefore, I introduce heterochrony into the vector field embryogeny framework by adding a temporal dimension to each basic field element as follows: Firstly, two new variables $t_i^c \in [0,1]$ and $t_i^d \in [0,1]$ are encoded in the genome for the i-th field element. These values are scaled to fit simulation time: $t_i^{c*} = t_i^c \cdot t_{\max}$, and $t_i^{d*} = t_i^d \cdot \frac{t_{\max}}{5}$. Here, $t_{\max}$ is the maximum simulation time. t_i^{c*} specifies the temporal center of the field element, while t_i^{d*} denotes the length of the timeframe of the field element. Note that I set the maximum temporal width of an element to one fifth of the simulation time, such that a temporal separation of basic field elements is facilitated. For convenience, I define $t^+ = t_i^{c*} + \frac{t_i^{d*}}{2}$ and $t^- = t_i^{c*} - \frac{t_i^{d*}}{2}$. The values t_i^{c*} and t_i^{d*} are used to create a time dependent basic field element $\mathbf{E}_i(t) = \gamma_i(t) \cdot \mathbf{E}_i(\mathbf{X}, \boldsymbol{\lambda}_i)$, that replaces $\mathbf{E}_i(\mathbf{X}, \boldsymbol{\lambda}_i)$ in Equation (3.2). $\gamma_i(t)$ is a piecewise linear function, which has its

maximum value of 1 at $t = t_i^{c*}$ (see Figure 5.7):

$$\gamma_i(t) = \begin{cases} \frac{2(t-t_i^{c*})}{t_i^{d*}} + 1 & \text{if} \quad t^- < \quad t \quad \leq t_i^{c*} \\ \frac{2(t_i^{c*}-t)}{t_i^{d*}} + 1 & \text{if} \quad t_i^{c*} < \quad t \quad \leq t^+ \\ 0 & \text{if} \quad t \leq t^- \quad \text{or} \quad t > t^+ \end{cases} \tag{5.1}$$

This formulation allows evolution to tune both, onset and temporal length of basic field elements explicitly. The piecewise linear characteristics are chosen to ensure that

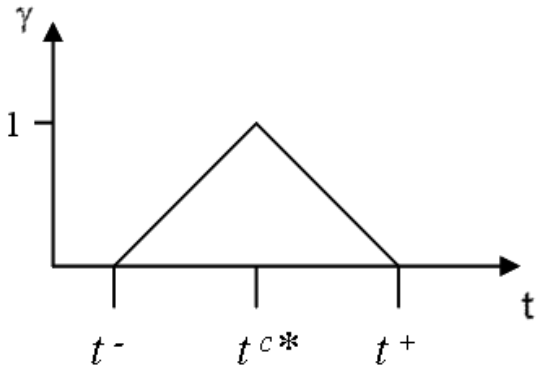


Figure 5.7.: The scaling factor γ that introduces heterochrony into the vector field embryogeny framework, has the presented piecewise linear temporal characteristics.

small mutations of t_i^{c*} and t_i^{d*} result in smooth changes of the temporal characteristics of the phase space.

For heterochrony experiments, I encode the additional parameters t_i^c and t_i^d for the i-th element in the chromosome of an individual.

5.3.4. Allometry in Vector Field Embryogeny

Allometry occurs when different parts of an organism grow at different rates in distinct species [1]. An example can be borrowed from Wolpert [115]: the central toe of a horse grows at a rate 1.4 times that of lateral toes. This allowed for evolutionary adaptation by formation of the typical shape of the horse hoof, originating from ancestral multi-toe feet. In this example, the size of a toe can be seen abstractly as a variable in the phase space of hoof development. In this light, toe development can be seen as a transient process. At a certain point in developmental time, the outcome of this process (i.e., the relative size of the toes) is a result of the evolved rates of change of the toe-size variable, achieved by a scaling of the relative speed of system dynamics of the toes. Therefore, allometry can be seen as a means to evolutionarily change the transient behavior of several microscopic parts of a system to influence its final macroscopic shape. In vector

field embryogeny, we might ask the question: What are the consequences of allowing for direct evolution of the rate of cellular processes?

We can include allometry into vector field embryogeny by evolving a speed factor for every cell-state $\mathbf{X}$, which scales the speed of the system state on its trajectory through the 3D phase space. Thus, in different individuals, $\mathbf{X}$ could reach different states in finite time, even if the phase space would be exactly the same. In practice, I encode n additional allometry variables, $\nu_i, i \in \{1, \ldots, n\}$, which are speed factors of the n cells in an individual. These variables lie in the interval $[0, 1]$, and are used to scale t_{max} for the integration of Equation (3.7), i.e., $t^{allo}_{max,i} = \nu_i \cdot t_{max}$ for the i-th cell, representing cellular phase space speed.

In the following, the three implementations described above (hierarchy, heterochrony, and allometry), will be used to augment vector field embryogeny in the task of cellular differentiation. The experiments presented in the following use the 'H'-target shape, presented in Figure 5.4 as optimization goal. I denote the basic setups without augmentation from Section 5.2.3 as baseline.

Firstly, the hierarchy setup is evaluated. Then, allometry is investigated, and I show how a combination of the two features realizes the desired optimization behavior. Finally, the heterochrony setup is applied to both, the simple 4-cell targets given in Figure 5.1 and the 'H'-target.

5.3.5. Evolving Differentiation Using Two Stage Spatial Hierarchy

To investigate the influence of spatial hierarchy, I allow for weak symmetrical boundary conditions as described above by choosing the coordinate system according to the lower panel on the right of Figure 5.6. The second hierarchy experiment uses a formulation that allows free evolution of symmetry, by encoding eight additional variables to be evolved. For each first stage cell, two of these variables (η_1, η_2) are used to determine whether the coordinate system for the respective second stage is flipped horizontally and vertically ($\eta_1 < 0.5$ and $\eta_2 < 0.5$, respectively). The number of field elements is set to 8 in total, i.e., 4 for the first and 4 for the second stage. A combination of two regular elements and two singular elements in each stage is used.

Both approaches perform significantly better than the reference setup, reaching a mean fitness of about 0.6 and 0.4 respectively (see Figure 5.10, panels 3 and 4). In Figures 5.8 and 5.9, I depict the evolution of the best individual's phenotypes coarse and fine grained stages in the symmetrical boundary conditions run from generation 5 to 100 in steps of 5. It is visible how evolution finds symmetric solutions in the coarse stage and then uses this pre-structuring in the second stage to build a perfect solution to the target matching problem. Note that asymmetric solutions are possible even though the symmetry constraint is used (generation 25) and how a symmetric solution can still be reached in the second stage even if the first stage is not symmetric (generation 60).

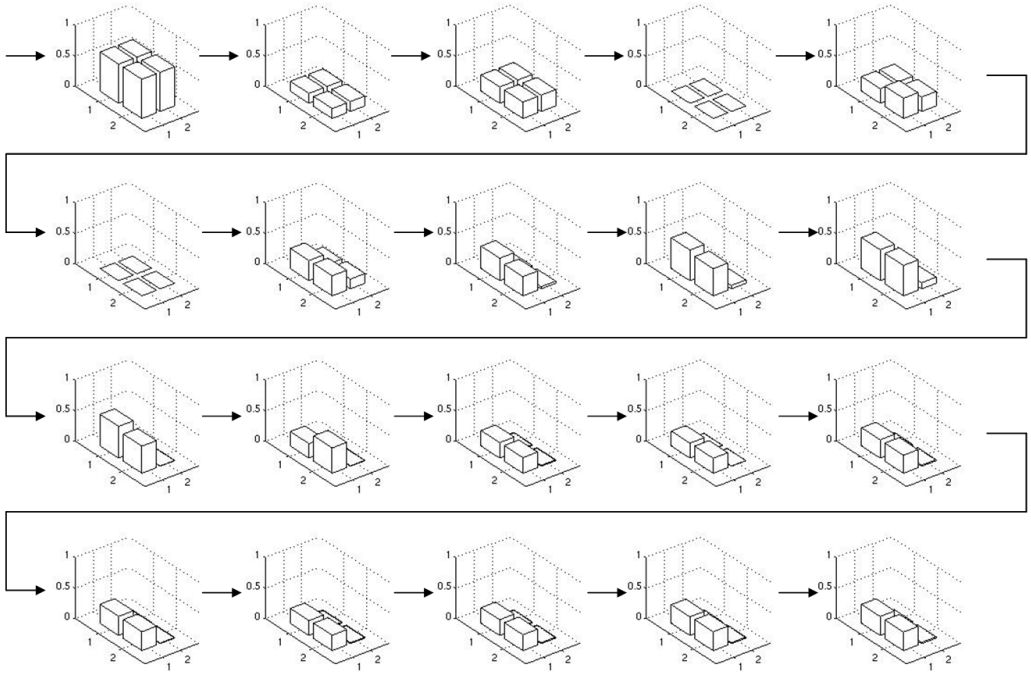


Figure 5.8.: The evolution of the coarse stage phenotype in a 2-stage hierarchy experiment. Every fifth generation from generation five to 100 is depicted. See Figure 5.9 for the fine grained stage, note that both stages evolve simultaneously.

5.3.6. Evolving differentiation with and without allometry

Column 2 in the left panel of Figure 5.10 gives the performance of the allometry setup. It shows a significant increase in the performance of the system, with a mean fitness 0.55 and smaller variance than in the reference setup. For a thorough analysis, I now concentrate on the most successful run with allometry. I depict phenotypes for generations 5 to 100 in steps of 5 (Figure 5.11). The corresponding system trajectories are given in Figure 5.12. The system seems to have one attractor for all points throughout the first generations. Around generation 35, a new, lower (i.e., $z < 0.5$) attractor is found. Prior to that, all distinct cell type- (i.e., z-) values were resulting from allometric scaling on the way to the upper attractor, i.e., by cells being in a transient state. After generation 35, the system settles for this configuration while optimizing the z-position of the new attractor. A third attractor is found around generation 50, which yields the basis vector field setup for the final solution. Until generation 100 is reached, the

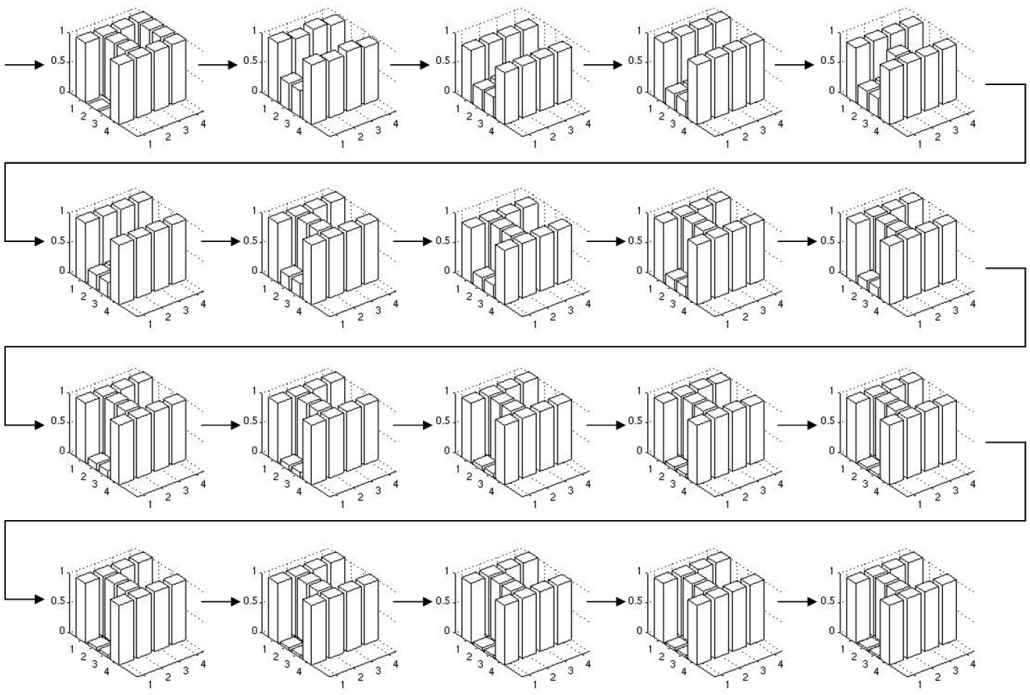


Figure 5.9.: The evolution of the fine grained stage phenotype in a 2-stage hierarchy experiment. Every fifth generation from generation five to 100 is depicted. See Figure 5.8 for the coarse stage, note that both stages evolve simultaneously.

positions of these three attractors are optimized to yield a perfect solution. In the final individual, all cells have found suitable trajectories. Additionally, I investigate the role of allometry in the developmental process of the evolving individuals: I artificially switch off allometry by setting all ν_i to 1 and repeat development for the best individuals of each generation throughout the evolutionary run (Figure 5.13). Interestingly, time plays no role for the development of an individual belonging to later generations. Indeed, after generation 60, the phenotypes of the original evolutionary run and the non-allometry run are the same (compare Figures 5.11 and 5.13). I investigated this feature in all 20 runs of the experiment and found that this holds only for the best run; all other runs produce individuals that depend on allometry. Therefore, the question remains whether the evolutionary success of the best run is directly linked to this feature.

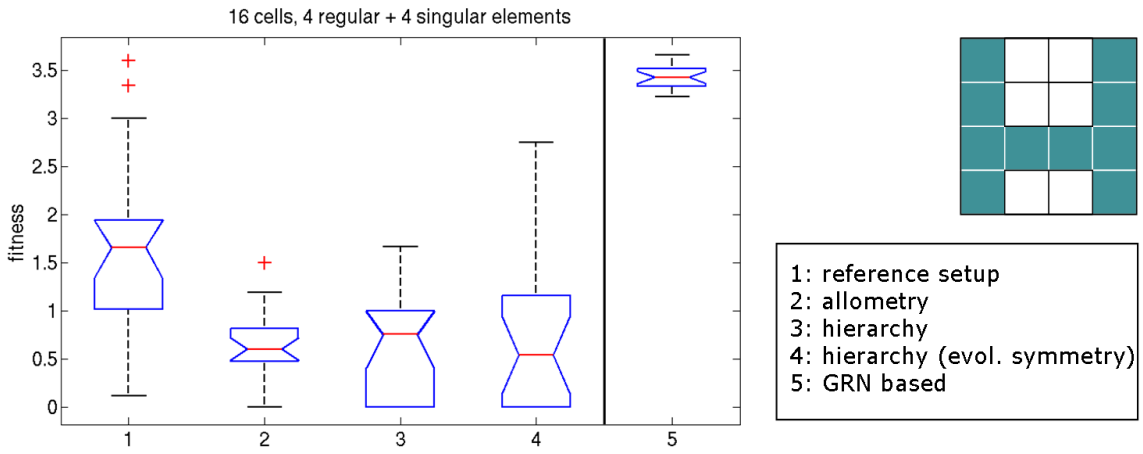


Figure 5.10.: Results from the differentiation experiment with 16 cells. The 'H'-target pattern and the results from the different setups are presented. The results of the reference setup, the allometry setup and two hierarchy setups (pre-defined and evolving weak symmetry) are depicted, as well as the result of the respective gene regulatory network based approach.

5.3.7. Evolving Differentiation Using Hierarchy and Allometry

These experiments use a combination of hierarchy and allometry. I investigate three different experimental setups: using allometry on the first hierarchical stage only, using allometry on the second hierarchical stage only, and using allometry on both hierarchical stages. For these experiments, the weak symmetrical boundary conditions apply.

The results for the three different setups are depicted in Figure 5.14. We can see that hierarchy combined with allometry on both stages reaches the optimal solution with only few outliers. Using allometry on the second stage only does not improve performance significantly. Interestingly, using allometry explicitly on the first stage gives exceptionally good results. Optimal solutions are found in 17 out of 20 runs.

5.3.8. Evolving Differentiation Using Heterochrony

Similar to the allometry implementation described above, the heterochrony setup provides an encoding of the phase-space, where the temporal dimension is used explicitly. However, the number of variables encoded in the chromosome of an individual only depends on the number of basic field elements, and not on the number of cells.

Four Cells Targets

Figure 5.1 gives the target lattices for the first differentiation experiment. The according experiments in Section 5.2.3 employing 2 regular and 2 singular basic field elements

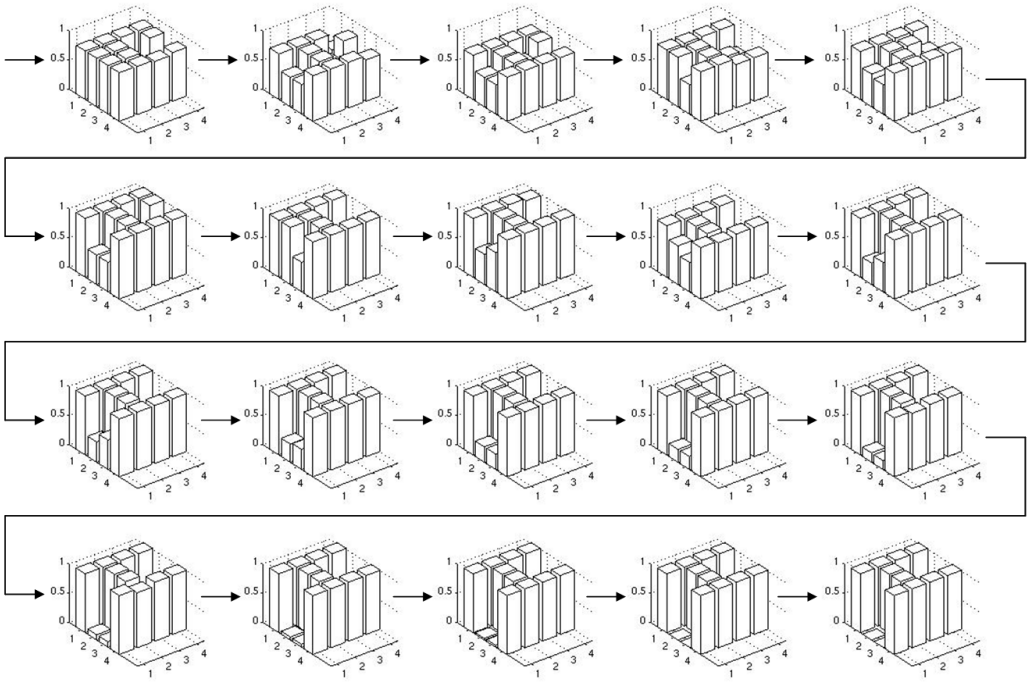


Figure 5.11.: An evolutionary perspective on phenotypes. The best phenotypes throughout an evolutionary run of the allometry experiment are depicted for every fifth generation from generation five to 100. It is visible how evolution optimizes phenotypes toward the desired target shape.

serve as baseline.

Figures 5.15 – 5.17 show the fitness distributions over evolutionary generations for the best individuals of 20 runs for each of the three experiments, employing both baseline setup (left panels) and heterochrony setup (right panels). It is clearly visible that the 'onepoint' and 'half' experiments converge quickly to an optimal solution in all runs, with no significant differences in the evolutions, except that initial populations of the heterochrony runs generally seem to have a worse fitness. Nevertheless, convergence to optimal fitness takes place for all runs in few generations.

While all except one of the heterochrony 'xor' experiments reach the optimum solution in 100 generations, we can observe slower convergence than in the other experiments. Nevertheless, vector field embryogeny seems suited for the task. It is visible that for this problem, the initial populations of the baseline and heterochrony setup have a similar fitness distribution. However, the heterochrony setup has a smoother transition in fewer generations to optimal fitness. Note that this comparison is not entirely fair: the heterochrony setup has two additional values encoded per basic field element,

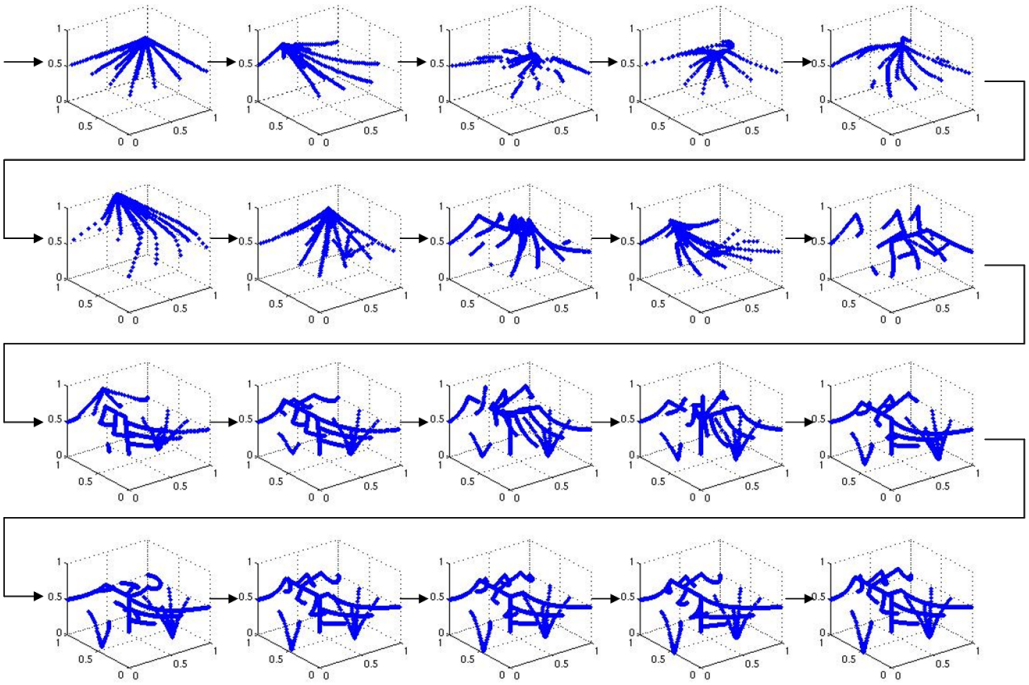


Figure 5.12.: An evolutionary perspective on phase spaces. The phase space trajectories of the individuals given in Figure 5.11 are depicted. Both, the cellular differentiation toward attractors, as well as the changes of attractor positions can be traced throughout evolution.

thus a chromosome size that is larger by 8 elements. To account for this, I repeated the baseline experiment for the 'xor' task with an increased number of basic field elements: The left panel of Figure 5.18 depicts the fitness distribution for an experiment employing 3 regular and two singular elements, i.e. the same number of free parameters as compared to the 2 regular + 2 singular element heterochrony setup. Interestingly, the performance decreases drastically. This could be related to an overfitting phenomenon. To complete the investigation, the right panel of Figure 5.18 shows the performance of an experiment employing 2 regular and 3 singular elements. It is visible that no performance increase results from the additional parameters for the evolution of the vector field. Thus, the heterochrony setup does not decrease performance of the vector field embryogeny approach for simple problems, and increases performance for the slightly more complex 'xor' problem.

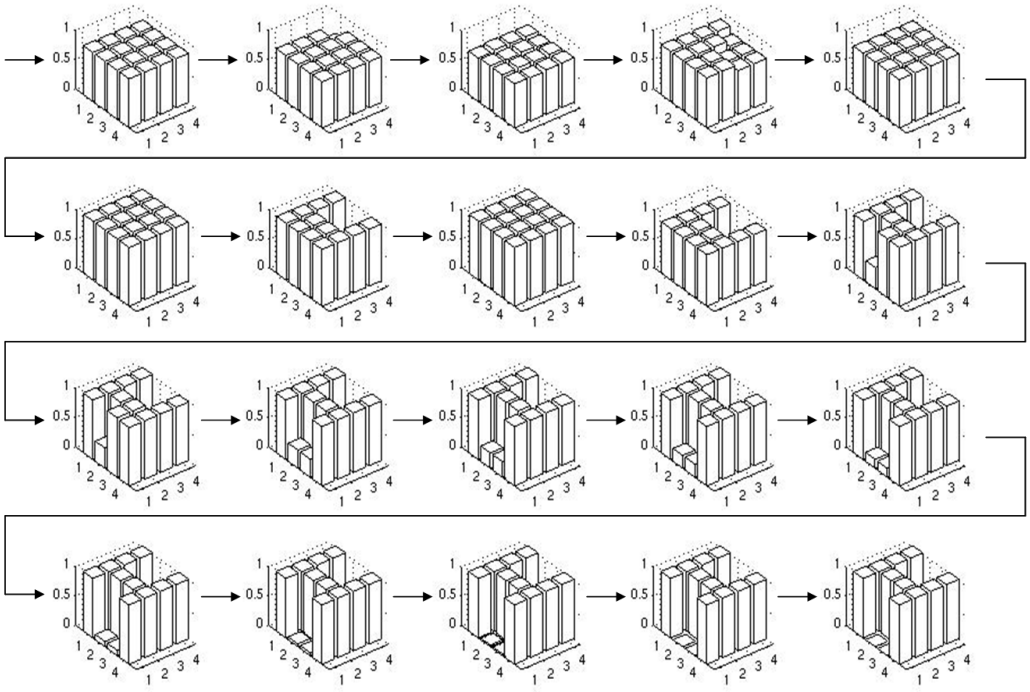


Figure 5.13.: An evolutionary perspective on modified phenotypes. The phenotypes throughout evolution are depicted, where genotype is taken from the individuals given in Figure 5.11 and allometry is switched off. The final individual is allometry-independent after evolution.

16 Cells Target

I also compare the heterochrony setup to the baseline setup on the more complex 'H'-task. To account for the increased complexity of the problem, we increased the number of basic field elements in the heterochrony setup to 4 regular + 4 singular elements. Each experiment consists of 40 runs of 200 generations. For a fair comparison, I equipped the baseline setup with 5 regular + 5 singular elements. Note that this baseline setup outperforms a 4 regular + 4 singular elements setup (results not shown). Figures 5.19 and 5.20 show the fitness distributions of the experiments. In both approaches, 10 runs converge to the optimum fitness. The heterochrony experiment has a smaller extent of the box denoting the 25th to 75th percentile, but more runs do not converge to a fitness value below 2, as compared to the convergence of all runs in Figure 5.19. Figure 5.21 shows an enlarged box plot of the distributions of the best individuals after 200 generations. Triangular markers give the 95% confidence comparison intervals, which do not overlap. Thus, it is visible that the median of the heterochrony setup is lower than the median of the baseline setup, for the given task.

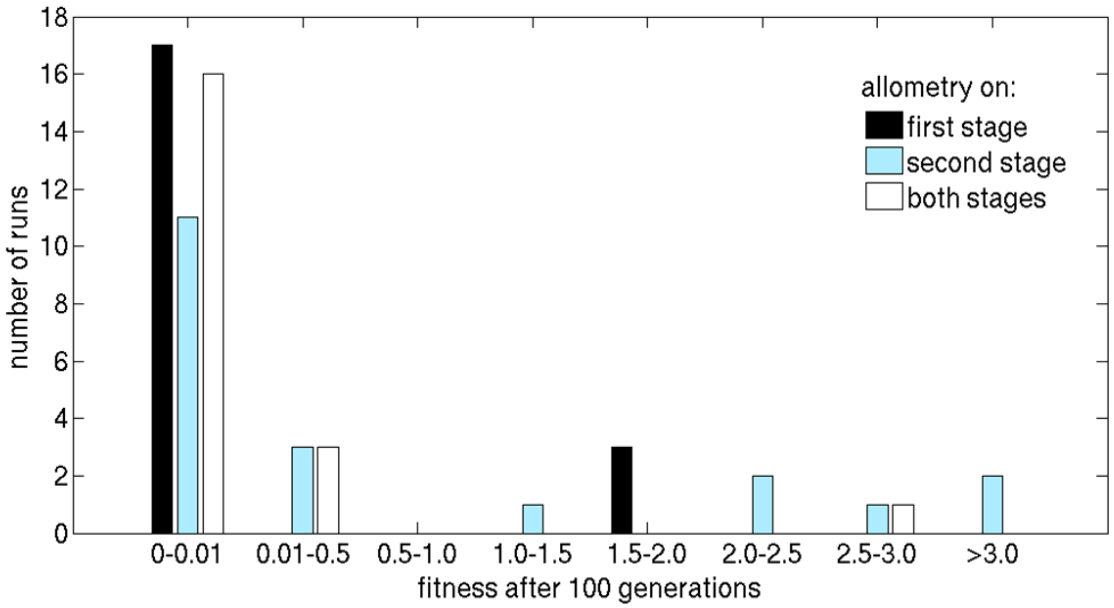


Figure 5.14.: Results of experiments combining allometry and hierarchy. Allometry is employed on the first stage only (resulting mean fitness: 0.250), on the second stage only (resulting mean fitness: 0.882), and on both stages (resulting mean fitness: 0.178), respectively. The plot shows how many of the 20 evolutionary runs reached indicated fitnesses. Note that the first bin is scaled to a small size (0.0-0.01) to account for the high quality of the solutions.

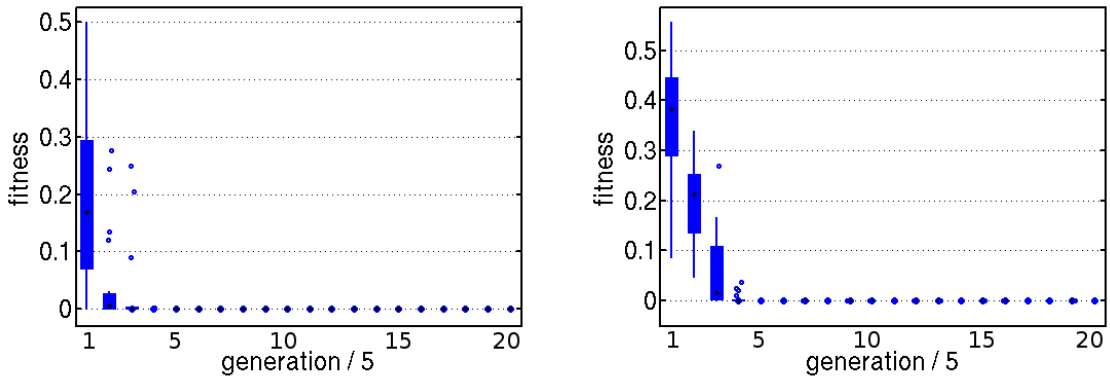


Figure 5.15.: The plots show the fitness distribution of the best individuals of 20 evolutionary runs over 100 generations for the 'onepoint' experiment (generations in steps of five). The left panel depicts the result for the baseline experiment, while the right panel shows the performance of the heterochrony experiment. Boxes in the box plot denote the 25th to 75th percentile, dots are outliers. Whiskers extend to ca. 99.3% of the data.

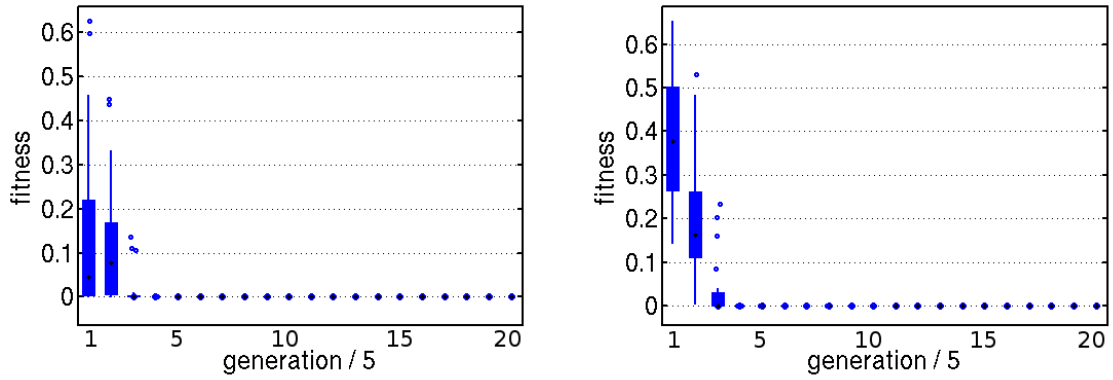


Figure 5.16.: The plots show the fitness distribution of the best individuals of 20 evolutionary runs over 100 generations for the 'half' experiment (generations in steps of five). The left panel depicts the result for the baseline experiment, while the right panel shows the performance of the heterochrony experiment.

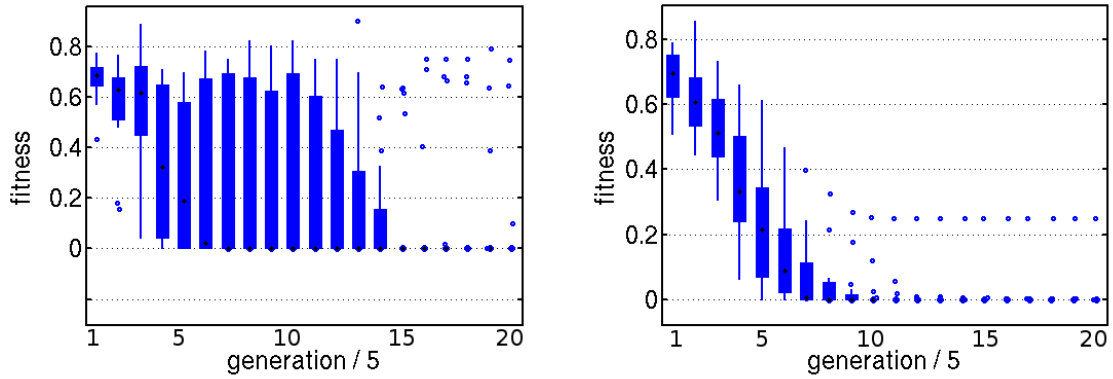


Figure 5.17.: The plots show the fitness distribution of the best individuals of 20 evolutionary runs over 100 generations for the 'xor' experiment (generations in steps of five). The left panel depicts the result for the baseline experiment, while the right panel shows the performance of the heterochrony experiment.

5.3.9. Discussion

Evolutionary changes in biological development are either spatial or temporal [1]. The temporal aspect has not been given explicit attention in the Artificial Development community, apart from defining intermediate or final stages of developmental processes, after a defined number of time steps [43]. It has even been argued that the temporal aspect can be neglected in developmental systems by replacing the developmental mapping with a CPPN (Compositional Pattern Producing Network) [56], a feed forward artificial neural network like structure with special activation functions. In biology, the significance of developmental time for evolution has long been recognized and widely studied. Time in self-organization processes, particularly in spatial pattern formation, is known to play an important role, see for example [116, 117]. The spatial aspect of

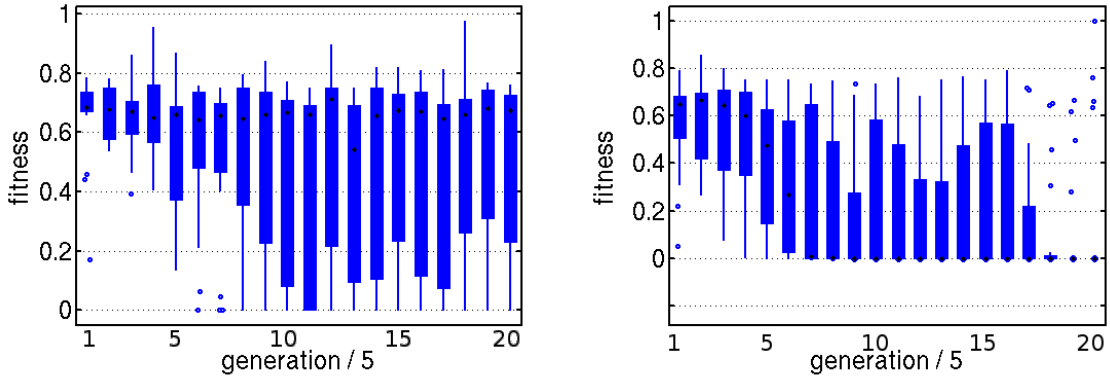


Figure 5.18.: The plots show the fitness distribution of the best individuals of 20 evolutionary runs over 100 generations for the 'half' experiment (generations in steps of five). The baseline setup is employed for both panels. The left panel depicts performance of the experiment employing three regular and two singular elements, the right panel shows fitness distributions of the experiment employing two regular and three singular elements.

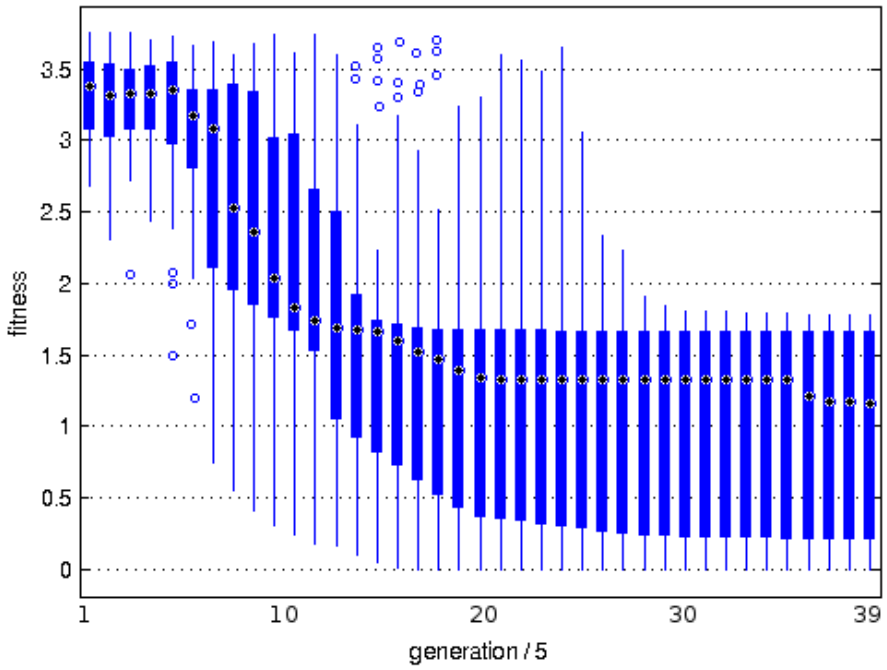


Figure 5.19.: The fitness distribution over 199 generations of 40 runs of the 16 cells baseline experiment toward the 'H' target. For box-plot attributes, see caption of Figure 5.15.

embryogenesis is a current field of research. Hierarchical structuring is e.g. employed by Doursat [110] for the creation of cellular differentiation patterns.

Formulating the vector field embryogeny-system hierarchically yields an insight into

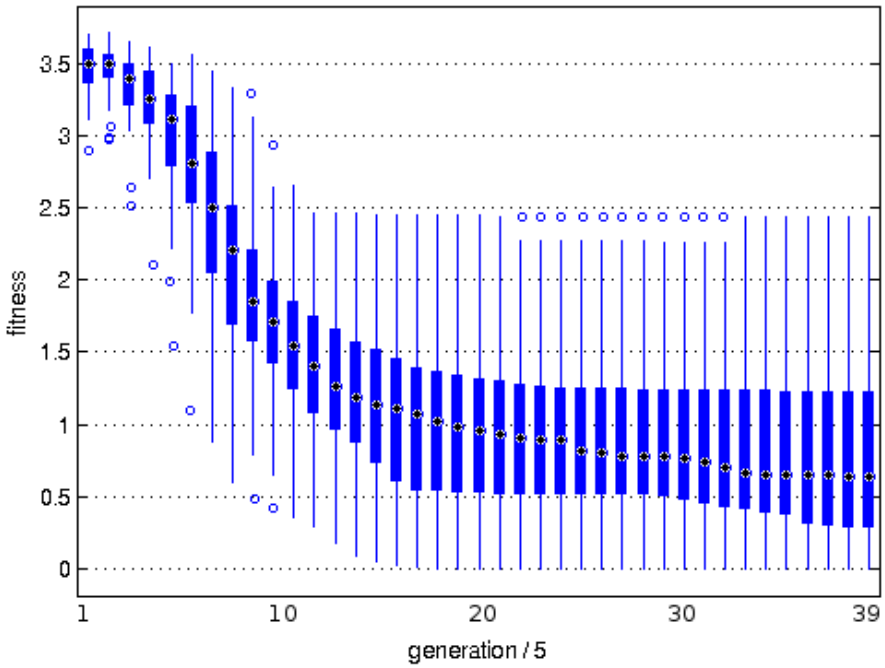


Figure 5.20.: The fitness distribution over 199 generations of 40 runs of the 16 cells heterochrony experiment toward the 'H' target. For box-plot attributes, see caption of Figure 5.15.

how a strategy might look like, with which to tackle more complex problems. The weak symmetry constraints and evolving symmetry experiments have significantly improved performance for the given task, although the evolving symmetry runs yield a relatively large variance in quality. A remaining research question is the definition of the weak symmetry, especially when more than two stages are employed: should the symmetry constraint be inherited from the coarse stage to the next stage? Should it be redefined for each stage and each quadruple of cells? If so, how can we overcome the exponential increase in parameters to encode symmetry information?

I investigated allometry for the 'H' target pattern, and found that it improves evolutionary performance significantly. The question remains however, whether the allometry setup creates a 'shortcut' for the solution of the problem, i.e., that the better performance stems mostly from directly optimizing the 16 ν_i-values and thereby rendering the evolution of the phase space trivial. One trivial (and sub-optimal) solution could be such, that the phase space consists of a single point attractor at $z = 1$, which attracts all cell states. If now ν_i of the 6 cells in the interspace of the 'H' shape evolve to be 0, the interspace would remain at initialization level $z = 0.5$. However, this would yield a fitness $F = 1.5$, a value which is higher than the fitness reached in all but one evolutionary runs. Another seemingly simple solution would be a phase space that attracts all cells to a phase space trajectory which reaches $z = 1$ at one time and $z = 0$

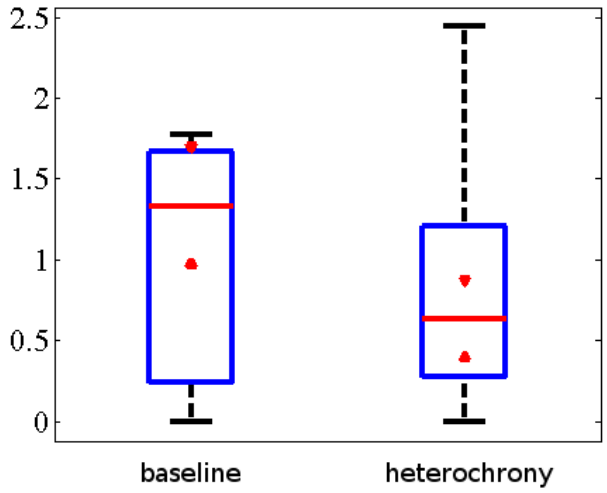


Figure 5.21.: Comparison of the baseline and heterochrony setup. Depicted are the distributions of fitnesses of the best individuals of 40 runs after 200 evolutionary generations. Boxes denote the 25th to 75th percentile. Whiskers extend to ca. 99.3% of the data. Triangular markers give the comparison intervals (see text).

at another time, and then tuning ν_i of all cells such that they stop at these two points. However, considering the basic elements we used, and the initial positioning of cell states, creating such a phase space would be extremely difficult, since the trajectories would have to cross their plane of initialization at $z = 0.5$.

The analysis of the trajectories throughout evolution shows nicely how the phase space evolves to accomplish the task. It is interesting to find a solution independent of allometry, although the evolutionary run has this feature enabled. Since allometry plays a role in the individuals in early generations, it has an effect on the course of evolution, yielding an evolutionary path to the optimal solution, which without allometry was not found in any of the reference evolutionary runs we have performed.

Combining the hierarchical framework with allometry yields exceptionally good results. For allometry on the first stage, 85% of the runs converge to an optimal solution, for allometry on the second stage, only 55% converge to the optimum, and for allometry on both stages, 80% converge. Especially the case of using allometry on the first stage only is interesting, since the performance is not significantly different from employing allometry on both stages: it seems that if evolution has a more direct control of the first stage values, the second stage does not need that level of explicit control. If we use the same setup and evolve the four cell types of the first stage explicitly by direct coding (i.e., encoding z_1 to z_4 directly in the chromosome), all runs converge to the optimal solution (not shown). This finding motivates a setup where several hierarchical stages are employed, and the explicity of evolutionary control rises towards more coarse stages, such that e.g. in the four cell stage, a direct coding could be used, in the 16 cell stage a vector field with allometry, and in a subsequent, 64 cell stage a vector field

only.

Graph based artificial development-systems implicitly allow heterochrony to occur, but do not easily allow a comparison between non heterochronic and heterochronic development. vector field embryogeny allows for this comparison. The results presented here suggest that for the given problems, using heterochrony improves evolutionary success of nontrivial cellular differentiation tasks. Also, using heterochrony seems to reduce bad convergence behavior resulting from too many free variables, as seen in the 'xor' experiment. Therefore, I believe it is worthwhile to consider investigations of the temporal dimension of Artificial Development-systems in general, and in particular to exploit differential combination of modular control aspects of development along the time axis. Similar to vector field embryogeny, which allows for heterochronic as well as non heterochronic development, it would be interesting to find a way to more directly influence heterochrony in graph-based genetic representations, such as artificial gene regulatory networks. I think that this can be achieved by carefully tuning and investigating structural aspects of graphs, and may eventually result in finding motifs [46] that influence the timescale of graph dynamics explicitly. Apart from heterochrony, to investigate the links between vector field embryogeny and graph representations in general is an interesting task for future research. An approach combining an abstract, phase space centered view with a more detailed, graph centered view of gene regulatory networks could yield a better insight into the evolution of networks, especially in a biological context.

6. Evolvability of Graph- and Vector Field Embryogeny-representations

6.1. Introduction

Chapters 4 and 5 have shown features of two different representations for dynamical control of development. While on the one hand, evolving graph structures was shown to possibly incorporate secondary dynamical functions (such as negative feedback) that were not selected for, and thus possibly slow down evolutionary progress, we have seen that vector field embryogeny is generally suitable for evolution strategies to create simple dynamics.

I now want to strengthen the empirical observation that graphs are notoriously hard to evolve, while vector field embryogeny seems more suitable for an evolutionary process to succeed. An evolvable* representation needs to possess the ability to allow for an evolution strategy to proceed by its mutation operators, especially with respect to a standard strategy parameter adaptation. In this context, the causality of the genotype to phenotype map introduced in the following section can be seen as a prerequisite for evolvability.

Since mutation of the genotype, coupled to phenotype selection is the way evolution proceeds, the effect of a dynamics representation in the genotype to phenotype map is an important aspect of evolutionary success. Therefore, in this chapter I will observe in more detail how the vector field embryogeny approach and graph representations in general react toward mutations, and how this might influence an evolutionary method.

6.2. A Prerequisite: Strong Causality and the Genotype to Phenotype Map

Investigations on evolvabilty in evolution strategies were performed by Sendhoff et al. [119]. The authors found that a mutation operator should preserve neighborhood structure in corresponding evolutionary spaces, referring to the genotype and the phenotype

*Dissimilar to the use of the term evolvability in biology, where it denotes the ability of a population of organisms to generate genetic diversity and evolve through natural selection [118], I use the term with respect to the success of a standard evolution strategy.

space. They state that this is a prerequisite for a successful evolutionary search, and give the condition in terms of probabilities: $\forall g_i, g_j, g_k$:

$$\begin{aligned} \|f(g_i) - f(g_j)\| < \|f(g_i) - f(g_k)\| \\ \Leftrightarrow \\ P(g_i \to g_j) > P(g_i \to g_k) \end{aligned} \tag{6.1}$$

Here, g_i, g_j, g_k denote arbitrary genotypes while $f(*)$ is the mapping between genotype and phenotype spaces. $P(g_a \to g_b)$ is the probability to have a mutation resulting in a transition from genotype g_a to genotype g_b. In the contribution, they show that for a direct mapping, evolution strategies inherently possess strong causality, due to normal mutations with zero mean. Thus, in an evolution strategy, to maintain this causality, it would obviously be beneficial to have a high covariance between change in genotype and change in phenotype, because for maximal covariance, the monotonic relation between genotypic and phenotypic change will allow an evolution strategy to satisfy condition (6.1).

Here, I will investigate the causality change of the genotype to phenotype mapping created by both, the graph based developmental control, as well as the vector field embryogeny based developmental control. To do so, I will firstly define a phenotype representation for dynamical systems, which forms a metric space, and thus allows calculation of $\|f(g_i) - f(g_j)\|$ in condition (6.1). Subsequently, I will compare the causalities of vector field embryogeny and graph based mapping, to gain insight into evolutionary suitability of the two approaches.

6.3. A Phenotype for Dynamical Systems

It is not straight forward to define a general "phenotype" on the meta-level of dynamics. However, as we have seen in the context of development, the dynamics of a system are used to control cellular growth. Depending on the interpretation of dynamical features, system transients or trajectory endpoints (i.e. stable attractors) seem suitable for a representation of the phenotype. Since in my case, thresholds in the cellular model combined with transient behavior are used to control development, the phenotype representation should take this into account and be able to capture transient-related features and changes thereof.

6.3.1. Discrete Fields

Evaluating the vector field in the general dynamical systems equation (3.1) at n regular grid points in all m cardinal directions yields n^m vectors $\mathbf{v}_l, l \in \{1, ..., n^m\}$ that approximate the dynamical system in a discretized way. A change in any of the vectors $\mathbf{v}_l$ can be seen as a change in parts of the transients of the system, since it represents a local change in phase space, which does not necessarily result in a change of stable attractors. Note that this discrete representation can easily be gained for both, graph

and vector field embryogeny. In vector field embryogeny, the right hand side of Equation (3.2) gives $\mathbf{v}_l$ directly when evaluated at the n^m points. For graphs, initializing the nodes with the same n^m discretization points, and subsequent iteration of graph dynamics for one step yields the necessary data. The difference between each of the n^m system states after iteration and the corresponding initialization values yields a local derivative, which corresponds to the vectors $\mathbf{v}_l$.

6.3.2. Field Difference

The field difference $FD(A, B)$ between two dynamical systems A and B, represented by discrete fields $\mathbf{v}_l^A$ and $\mathbf{v}_l^B$ respectively, is calculated through the normalized sum of the euclidean distances between corresponding vectors:

$$FD(A,B) = \frac{1}{n^m} \sum_{l=1}^{n^m} \left\| \mathbf{v}_l^A - \mathbf{v}_l^B \right\|_2 \tag{6.2}$$

In the following, I will use the field difference to determine $\|f(g_i) - f(g_j)\|$ in Equation (6.1), i.e., the phenotype difference of the two individuals i and j.

6.4. Causality in Graph Based Modeling

For the analysis, only single cells with randomly initialized chromosomes are simulated. These cells have no neighbors, hence a diffusion process is superfluous in the simulations. This results in a simplification of the gene regulatory network simulation to a pure graph simulation, which could be described by a connection matrix and the nonlinear activation functions. Note that the effect of diffusion could be simulated by a negative auto-regulation of each node, if the calculation domain can be expected to have large size or open boundary conditions. Therefore, without loosing connection to the gene regulatory network model, it is possible to replace the gene regulatory network formalism described in Section 3.3.1 with the widely used paradigm of continuous time recurrent neural networks. In the neural network model, the activation a_i of the i-th neuron is calculated by $a_i = \frac{1}{1+e^{-s}}$, where s is the weighted sum of the inputs to this neuron. Weights are in the range $[-1, 1]$.

For the calculation of the field difference, three nodes of the network are selected randomly. Thus, the observed phase space is 3-dimensional, independent from the actual network size. Note that this choice represents the dynamics in the light of a developmental model, since usually not all nodes in a graph control development, but only a small subset is responsible for the interface to the cellular model. Each of the three dimensions is sampled with $n = 20$ discretization steps, thus, 8000 vectors represent each phase space. The initial activation value of each neuron is drawn from a uniform random distribution in the range $[0, 1]$.

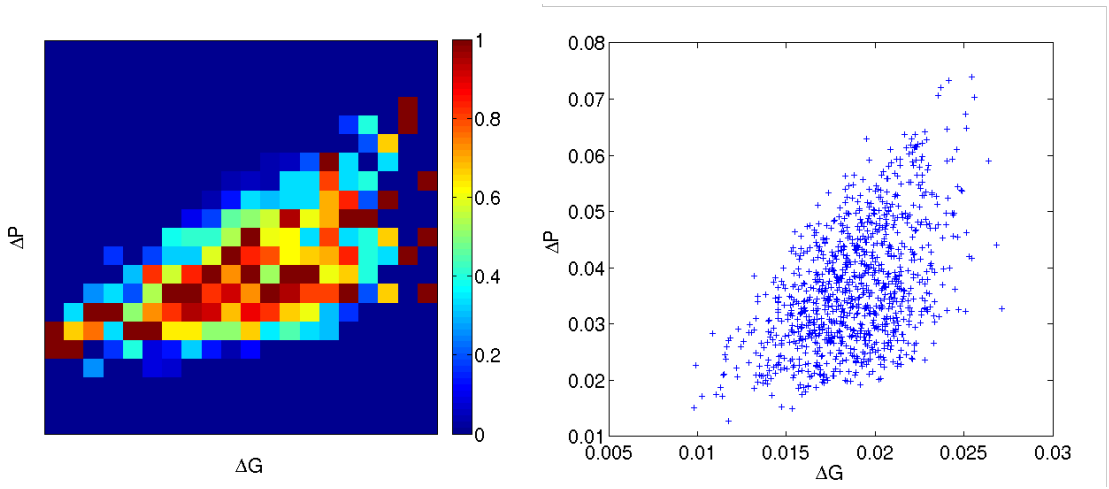


Figure 6.1.: An example for the $\Delta G \Delta P$ plots. The right panel shows the data sample that is used to create the plot in the left panel.

Firstly, I will visualize the characteristics of the different genotype to phenotype maps. To this end, I want to introduce $\Delta G \Delta P$-plots, which display the phenotypic change ΔP resulting from a number of genotypic changes ΔG. The left panel of Figure 6.1 shows a sample representation of a $\Delta G \Delta P$-plot, which is created in the following way:

1. Sample the genotype space with 100 points $g_i^0, i \in \{1, ..., 100\}$.
2. Create the discrete fields $\mathbf{v}_l$ for these 100 points, and denote them as phenotypes $p_i^0, i \in \{1, ..., 100\}$.
3. Mutate each g_i^0 1000 times by adding uniformly distributed random variables in the range [-0.1,0.1] to receive genotype $g_i^j, j \in \{1, ..., 1000\}$. This samples the genotype neighborhood in the 100 genotype locations.
4. Then, calculate new phenotypes $p_i^j, j \in \{1, ..., 1000\}$.
5. For each of these, calculate the genotype distance $\Delta G_i^j = \frac{1}{N} \left\| g_i^0 - g_i^j \right\|_2$, where N is the number of entries in the genotype vectors. Note that in the evolution strategy framework, this genotype distance correlates with the probability of transition from g_i^0 to g_i^j.
6. Calculate the phenotype distance as field difference $\Delta P_i^j = FD(p_i^0, p_i^j)$.
7. The resulting pairs $\Delta P_i^j \Delta G_i^j$ can be visualized in a histogram fashion to characterize the mapping. These plots are constructed as a 2D histogram, where the widths of the 20×20 bins are chosen to span over the range $[\min(\Delta G_i^j), \max(\Delta G_i^j)]$ along the ΔG-axis, and over the range $[0, \max(\Delta P_i^j)]$ along the ΔP-axis. In such plots, bin columns contain all ΔP-values that result from a fixed ΔG-range. Each bin column is normalized individually, such that, for each column, the highest bin has value 1. This accounts for a possibly uneven distribution of mutations along the ΔG axis.

Such plots give first information on the mutational behavior of the genotype to phenotype map. Also, because of the sampling of the genotype space, we can infer homogeneity of the mutational behavior of the genotype to phenotype map. As an example, Figures A.1 and A.2 show such plots for the mutations of the 100 points g_i^0, for graphs of size 3 and 20 respectively. The 20 nodes setup exhibits cloud-like distributions of $\Delta G \Delta P$ pairs. Independent from the mutation strength, both, small and large changes in genotype can be achieved. Interestingly, this behavior can again be observed in virtually all subplots.

From the plots alone, we can infer that the covariance between ΔG and ΔP is higher for graphs with low number of nodes than for larger graphs. Remember that for a mapping to maintain causality in ES, a distribution in the $\Delta G \Delta P$ plots that rises monotonically would be ideal. A high covariance in the data points can be chosen to represent such a behavior. Therefore, I choose the covariance between ΔG and ΔP of different representations as indicator of suitability for ES. For a systematic comparison, I subtract the mean and normalize the data for each of the 100 sets of data points that define the $\Delta G \Delta P$ plots to $[-1, 1]$. Note that normalization is sensible, since ES self-adapts mutational stepsize, such that the qualitative, and not the quantitative relation between ΔG and ΔP values are important.

Figure 6.2 gives the distributions of the covariances for the 100 g_i^0, evaluated in different sized networks. The plot verifies the speculation above: the covariance rises with smaller network sizes. Thus, for evolution strategies, we can expect that only trivial graph structures can be evolved efficiently.

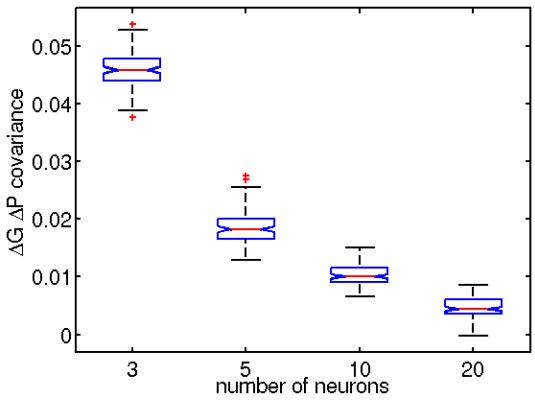


Figure 6.2.: The plot shows the distributions of covariances of $\Delta G \Delta P$-pairs for graph setups with different numbers of nodes. Each box represents the data of 100 genotype positions g_i^0, mutated for 1000 times.

6.5. Causality in Vector Field Embryogeny Based Modeling

The same investigations can now be performed for the vector field embryogeny framework. Similar to the graph experiment, Figure A.3 shows the $\Delta G \Delta P$ plots for a vector field embryogeny setup, which employs 6 singular basic field elements. Here we can observe a slightly positive slope of the distributions. The distributions over the subplots are similar, with few exceptions where the spread is larger. Again, Figure 6.3 gives the distributions of the covariances for the 100 g_i^0, evaluated for different numbers of basic field elements. We can see the similar trend as in the graph setups, that covariances

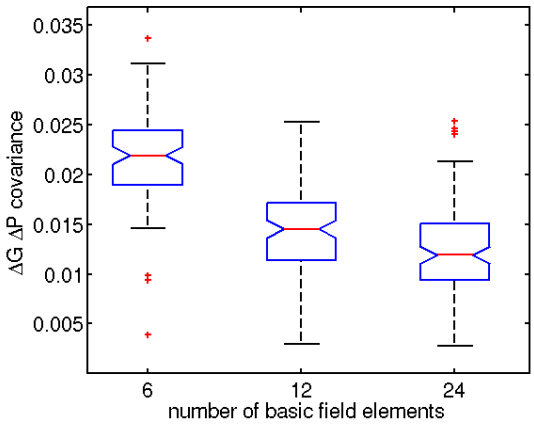


Figure 6.3.: The plot shows the distributions of covariances for vector field embryogeny setups with different numbers of nodes. Each box represents the data of 100 genotype positions g_i^0.

decrease toward larger systems.

6.6. Comparison of Graph Based and Vector Field Embryogeny Based Modeling

To be able to compare the two approaches for developmental control, causality represents but the suitability for evolutionary computation to advance. However, it is also necessary to compare systems that have the same representational power, i.e., the same capabilities to represent dynamics.

Typical dynamics features such as number of attractors and basins of attraction are interesting for the analysis of system stable states, but neglect most of the transient features, while evaluating transients yields highly variable, system dependent information. An illustrative example is the comparison of two similar phase spaces, except

that the one is scaled by a factor of two, i.e. the system is twice as fast. For the attractor based investigations, both would obviously yield the same results, while for a transient based investigation, they would be dissimilar. Therefore, we have to decide for a suitable point of view, i.e. whether the two phase spaces in this example should be ranked similar or dissimilar.

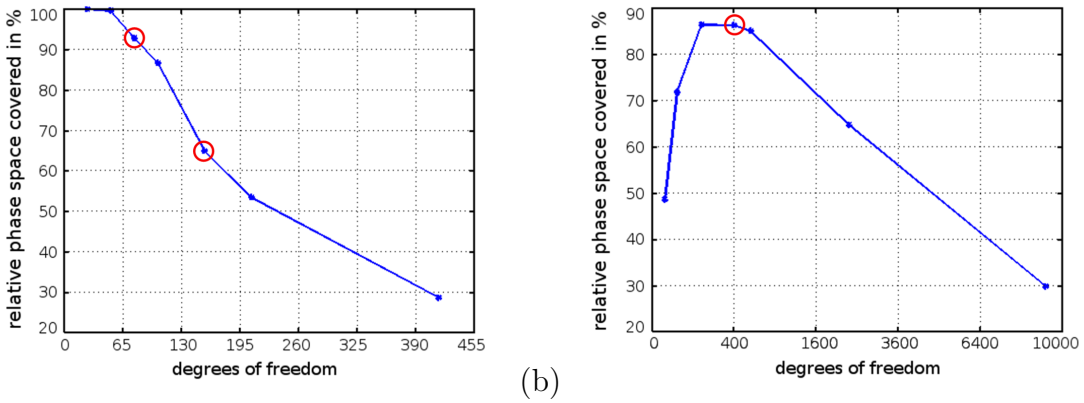
I want to develop a way of measuring "representational power", that is adapted to the use of dynamical systems formulation in developmental models. Usually, in these models transients are thresholded to yield binary function, such as triggering a cell division. Therefore, the transient behavior of the control system is important. Further, not all variables from the dynamical system serve as interface to the cellular model. We can thus assume that the transients of only a restricted number of variables from the dynamical system is of interest for the determination of representational power.

From an evolutionary developmental point of view, genetically encoded thresholds subdivide the phase-space perpendicularly to its cardinal directions. If the trajectory of the system state crosses such a threshold, an action is triggered. When a threshold exists for each cardinal direction, it is possible to constrain any 'rectangular' subvolume of the phase space with them. Therefore, the ability of a dynamical system to reach these subvolumes during transient behavior can be taken as its representational power, since cellular action that are defined to be activated in these regions can be triggered. Thus, if a dynamics representation has the ability to bring the system state to any point in the phase space, it is a powerful representation. In the following, as an indicator for this representational power, I will investigate the *covered phase space*, i.e., the area in the phase space that can be reached by the dynamical system representation.

The experiments are structured as follows: Similar to the experiments in Section 5.2, I initialize 9 system starting states $\mathbf{X}^i = (x^i, y^i, z^i), i \in \{1, ..., 9\}$ in a three dimensional phase space at $x^i \in \{0, 0.5, 1.0\}$ and $y^i \in \{0, 0.5, 1.0\}$, where all $z^i = 0.5$. This represents an intermediate sampling of the starting positions of experiments in Section 5.2. The system is initialized with a random chromosome, and from each of these starting points dynamics are iterated for 100 time steps. All $9 \cdot 100$ phase space positions visited by the system state are recorded. Then, the procedure is repeated 999 times with a new random chromosome. In this way, the resulting point-cloud of all transient system states of the 1000 random samples from the genotype space gives an estimate of the phase-space coverage of the representation. To calculate the empty phase space size, I subdivide the phase space into 10x10x10 equally sized sub-volumes, and check for each if they contain at least one of the previously recorded system states. Then, I can count these sub-volumes and get the estimate for representational power.

For vector field embryogeny, Figure 6.4(a) gives a plot of the covered phase space over the number of basic field elements, i.e., the degrees of freedom employed. Note that I chose half of the number of basic field elements to be singular, and the other half to be regular elements, similar to the experiments in Section 5.2. We can see that phase space coverage is 100% for lower degrees of freedom. Interestingly, the phase space coverage decreases steadily with higher numbers of basic elements. While at first this seems counter-intuitive, closer inspection shows us that for a large number of basic field elements, it becomes more likely to have a strong attractor close to the starting point

of the system states, such that shorter phase space trajectories occur with a higher probability. The area not covered in these cases can be found at the boundaries of the phase space (data not shown).



(a) (b)

Figure 6.4.: The phase space coverage of graph and vector field embryogeny approaches. Panel (a) shows the data for the vector field embryogeny setup, panel (b) gives the data for the graph setup. Note the abscissa values, giving the dimensionality of the search space: in (b), for a fully connected graph, the number of nodes needs is squared to give the numbers of free parameters. In (a), the number of basic field elements is multiplied by the number of parameters that encode each element. Highlighted by circles are the three setups that are used for comparison (see text for more details).

Figure 6.4(b) gives a plot of the covered phase space over different graph sizes. We can see that for a low number of nodes, less than half of the phase space is covered by the point cloud. With an increasing number of nodes, phase space coverage increases to over 87% since the representation gets richer in dynamics. however, for larger graphs, the phase space coverage decreases, where the largest network of 96 nodes covers only 30%. This effect can similarly be observed in the vector field embryogeny-appraoch. One hypthesis could be, that a larger graph possibly creates a larger number of attractors, which necessarily have a higher probability of proximity to the initial system states, similar to the observed case for the vector field embryogeny model. However, this hypothesis remains to be investigated thoroughly.

The two plots in Figure 6.4 allow us to asses representational power of the two approaches and gives us a basis for comparison of evolvability. As an example, I choose a graph with 20 nodes (400 degrees of freedom), which gives a phase space coverage of 87% (see circular highlight in Figure 6.4(b)). From the highlighted samples in Figure 6.4(a), we can see that a vector field embryogeny setup using 12 basic field elements (78 degrees of freedom) clearly has a higher phase-space coverage, while a vector field embryogeny setup using 24 basic field elements (156 degrees of freedom) clearly has a lower phase space coverage than the chosen graph sample.

Figure 6.5 shows the $\Delta G \Delta P$ covariance distributions for these three cases: a graph with 20 nodes, a vector field embryogeny setup with 24 basic elements, and a vector field

embryogeny setup with 12 basic elements. We can see that covariance values are low in general, but are significantly higher for both vector field embryogeny representations.

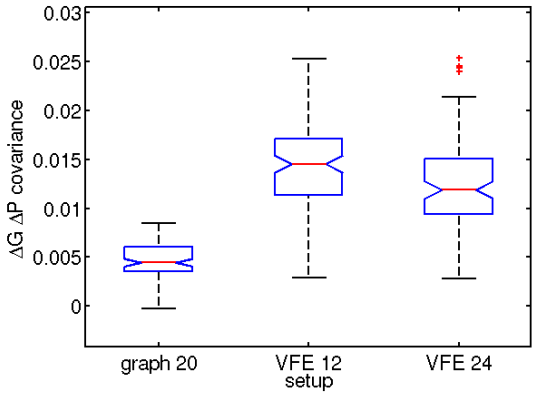


Figure 6.5.: The distributions of covariances for the comparison of graph and vector field embryogeny setups with matched numbers of nodes and basic field elements. Each box represents the data of 100 genotype positions g_i^0.

6.7. Discussion

The $\Delta G \Delta P$ plots introduced in this chapter provide a visual cue on the mutational behavior and homogeneity of the genotype to phenotype mappings. In evolution strategies, we have seen that the covariance of the $\Delta G \Delta P$ data can be used as an indicator of evolvability. The experiments have shown that for graphs, covariance decreases with increasing number of nodes. For vector field embryogeny, this trend can also be observed. This strongly motivates the use of an incremental approach toward more complex systems, where starting from a small graph/a vector field embryogeny with few basic field elements, evolution has the ability to add nodes or field elements when evolution stagnates. NEAT [39] is such an approach for graphs; similar ideas could be envisioned for vector field embryogeny.

The analysis of phase space coverage for vector field embryogeny gives insight into its general dynamic behavior: while a more complex setup with more basic field elements should potentially yield more complex dynamics, the consequences of more vector field embryogeny basic field elements are shorter trajectories. The reason for this is the probability of a strong attractor to be close to the starting points. This too would motivate an incremental approach, with careful introduction of new basic field elements. For example, a new random element could be added, with weight initialized at 0, such that it does not change phase space at first. Evolution can then tune the weight and thereby increment complexity of the representation.

For graphs, the expected gain in phase space coverage for increasingly larger networks can be observed only up to a network size of around 20 nodes. For larger networks,

phase space coverage decreases again. The reason for this is unclear so far and an interesting subject for future research. One hypothesis could be that, since larger graphs potentially have a higher number of attractors [120], the probability for a trajectory to end in such an attractor early increases, similar to the experiments in the vector field embryogeny setup.

When we choose a graph setup and a vector field embryogeny setup with similar phase space coverage, we find that the vector field embryogeny setup has a higher covariance between phenotypic changes and genotypic changes. For evolution strategies, this correlates with the higher success rate of vector field embryogeny in solving the cellular differentiation task. Another interesting fact from an evolutionary perspective is the dimensionality of the genotype space for such matched representations: while genotype space grows quadratically with the number of nodes in a fully connected graph, it grows only linearly with the number of basic field elements in vector field embryogeny. This naturally alleviates the search process in more complex domains.

7. Conclusion

Summary and Discussion

Artificial Development-approaches comparable to those presented in this thesis have become increasingly popular in evolutionary computation. The reason is a search for efficient optimization methods in complex, i.e. high dimensional problem domains [22]. Within this discipline, the developmental process is used as a genotype to phenotype map. Instead of encoding all features of the whole design directly on the genotype, a process that builds up the phenotype is encoded only. This may reduce the need for encoded information, and thus decrease the dimensionality of the search space.

The main problem of this kind of indirect mapping is the non-linear relationship between mutational changes to the genotype, and the resulting phenotypic alteration. For example, it is possible for a large mutation to result in no phenotypic change at all. At the same time, small mutations can change the developmental process substantially. Hence, strategy adaptation in the evolutionary method, which is meant to exploit knowledge gained about the problem domain during optimization can not generally succeed.

Still, the indirectness of representing the dynamics is necessary to control a developmental process. Placing a number of design points in a design space in a generative fashion must be coordinated, such that the compact genotypic description can be interpreted to produce a complete design. The experiment presented in section 4.2 of this thesis reveals that, although biology employs a graph representation for the dynamics, such an approach is less suited for computational evolution. The possibility of evolving characteristics that are not directly selected for, such as mutational robustness, poses a problem to the application of evolution as an optimization method: a graph structure evolves to decrease the effect of mutation on individual solutions. As a result, small mutations lose their effect, while large mutations still change the phenotype. However, in a rugged fitness landscape they have a high probability to yield worse fitness values. Thus, strategy adaptation tends to choose small mutational steps. Subsequently, stagnation results and an optimization might end in a local optimum prematurely.

Besides this evolvability consideration, the experiment presented in section 4.2 shows the necessity of analyzing graph dynamics more thoroughly, not, as usual, only from the static perspective. Recently, such research comes into focus in a biological context [87]. The evolutionary experiment in section 4.2 is a first step toward similar considerations in Artificial Development-systems. However, such analysis is troublesome, due to the large amount of data created. Data mining tools could facilitate this work in the future.

As we have seen in section 4.3, applying evolutionary development to engineering system design is possible. However, the aforementioned high amount of data that results from just one evolutionary run, combined with the non-trivial genotype- to dynamics-map, and the even less trivial dynamics- to phenotype-map, renders complete analysis very difficult. In accordance with the standard approach in the scientific field, I reduced the analysis to a performance based comparison between different approaches. The topology optimization task to create a Pareto-set of stable, lightweight designs represents such a benchmark, since alternative state of the art methods to solve this kind of problem exist. Both developmental models presented here, the 'sphere cell'-model and the 'polarized sphere cell'-model result in non-trivial solutions to the design problem. Even though state of the art approaches outperform the developmental models in most configurations, we can see that extremely lightweight designs can be achieved with the 'polarized sphere cell'-model only. This design domain exceeds the representational capacities of the 'simple rules' that govern the state of the art methods (SIMP, ESO). Therefore, the experiments motivate the use of a cellular encoding in more complex design problems, where such simple rules cannot be given directly, such as nonlinear problems in crash worthiness or porous material for surface-fluid interactions.

The analysis of these experiments led me to research alternative approaches to the representation of dynamical systems, that could be more suitable than graphs in the context of computational evolution. I have taken the approach to model dynamics in system phase space directly, through superposition of basic vector field elements, an approach which I named vector field embryogeny. The experiments presented in section 5.2 show that vector field embryogeny has a clear advantage over the graph-based approaches for the given sample tasks. With respect to strategy parameter adaptation in evolution strategies, the greater correlation between a genetic mutation and a change in the dynamics seems to be at least partly the reason for this advantage.

In section 5.3, I elaborate the meaning of vector field embryogeny in the context of biological science. Systematic investigations of higher level principles, such as hierarchy, heterochrony and allometry are possible when this abstraction is employed. It would be worthwhile to extend the insights into evolvability toward biological gene regulatory networks. Maybe, some genes and sub-networks of gene regulatory networks are just a low level representation of dynamical primitives, comparable to basic field elements in vector field embryogeny. Instead of observing mutations on the genetic level together with alterations to phenotypes, the change of dynamics, resulting from mutation should be monitored in biological organisms. Finding such 'biological basic field elements' would not only enhance the understanding of biological evolution, but also build a library for technical design optimization. Eventually, such a library would allow the findings from the comparison of vector field- and graph-evolution in Chapter 6 to be extended to biology. Since vector field embryogeny possesses a larger causality of the genotype- to phenotype-map than graph-representations, the vector field embryogeny-point of view could allow for a re-interpretation of mutational events and their phenotypic effect.

A natural question arising from the investigations discussed above is why biology employs gene regulatory networks, i.e., graphs as a representation for dynamics, if they are

so hard to successfully evolve. A possible answer to this is that the biologic substrate only allows such graphs as a feasible way of encoding dynamics, building on available elements such as proteins, nucleic acids, and enzymes. Still, we have to ask ourselves why biological species seem to maintain evolvability despite the graph structures of their gene regulatory networks, or vice versa, why this does not hold for evolutionary computation. We might only speculate that in contrast to biology, evolutionary algorithms employ fairly small populations and stop after extremely few generations. Needless to say this is due to the high computational expense of developmental simulations. Larger population sizes and less selection pressure in the simulations might naturally result in a larger diversity in the populations, and hence better optimization performance. Also, the biological environment is extremely rich, such that many niches exist for more 'exotic' species to survive. A successful attempt to artificially keep population diversity high through niching in graph evolution can be found in [39].

Overall, we have seen that artificial evolutionary development allows for finding non-trivial solutions for complex design problems. The proposed 'polarized sphere cell'-model imitates biological embryogeny to account for a rich space of possible cellular arrangements. The close interaction between the evolutionary process, the developmental mapping and environmental simulations necessitate a thorough adjustment of all these processes, to build up a successful overall optimization scheme. First steps are presented in this thesis. vector field embryogeny is specifically designed to be combined with a standard evolution strategy with self-adaptation to create an effective evolutionary developmental system, and shows promising results in sample evolutionary tasks.

A. $\Delta G \Delta P$ Plots

This appendix gives the $\Delta G \Delta P$ plots referenced in chapter 6. Three different setups are distinguished: a three nodes graph representation, a 20 nodes graph representation, and a 6 basic field elements VFE representation.

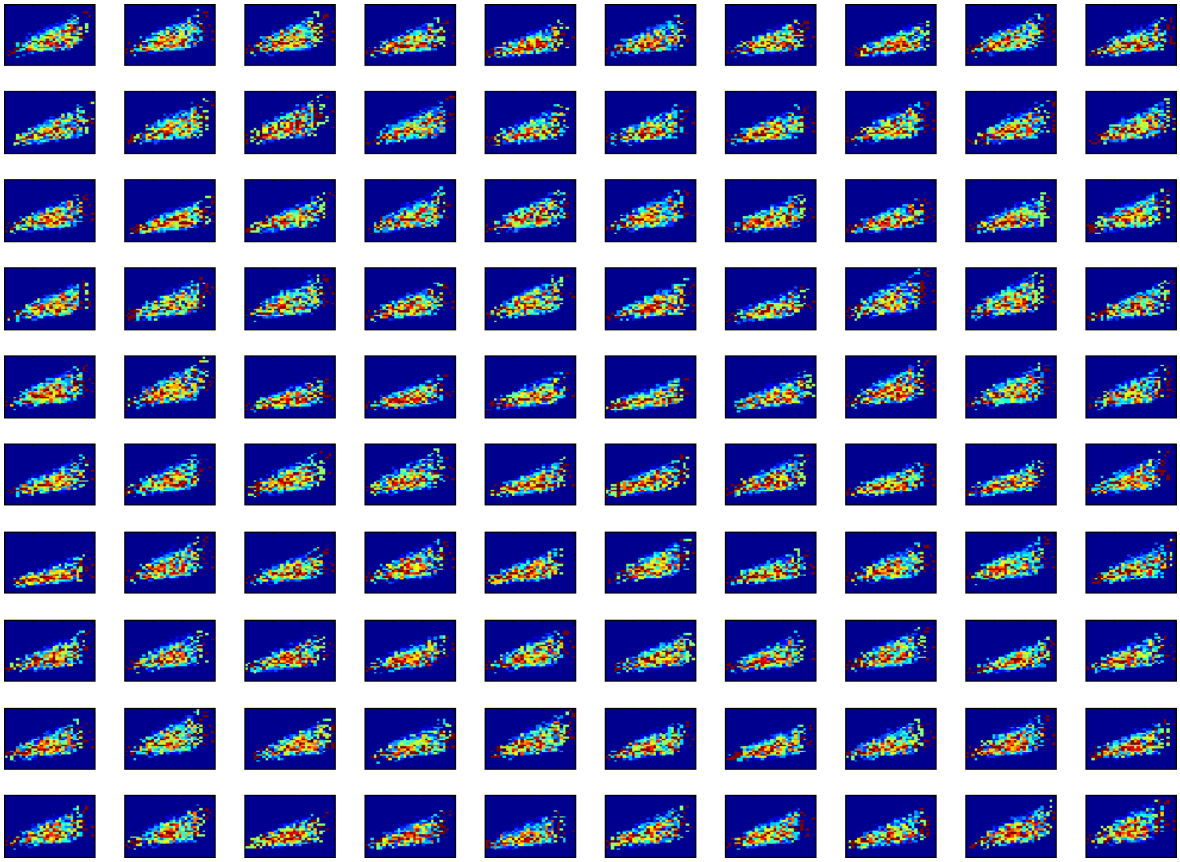


Figure A.1.: The $\Delta G \Delta P$ plots for 100 samples of the genotype space of a graph setup with three nodes are depicted. Each plot is created in the same way as the plot in Figure 6.1. In the three nodes setup, we can observe a triangular shape of the distributions for the small network size. This suggests that small mutations result in small phenotypic changes, while large mutations result in both, small and large genotypic changes. Also, we see homogeneity throughout the different subplots, which points to a genotype to phenotype map that has same mutational behavior at different genotype positions.

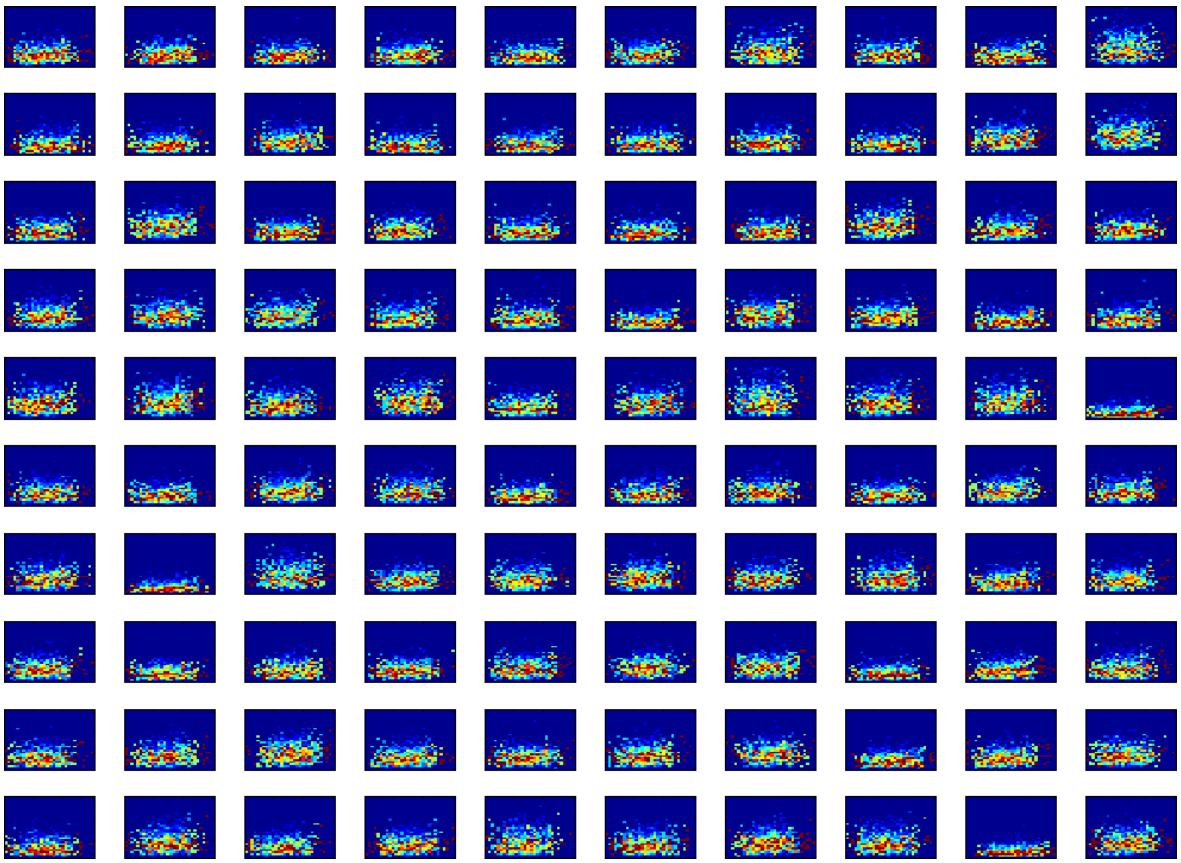


Figure A.2.: The $\Delta G \Delta P$ plots for 100 samples of the genotype space of a graph setup with twenty nodes are depicted. Each plot is created in the same way as the plot in Figure 6.1, using the same colorbar.

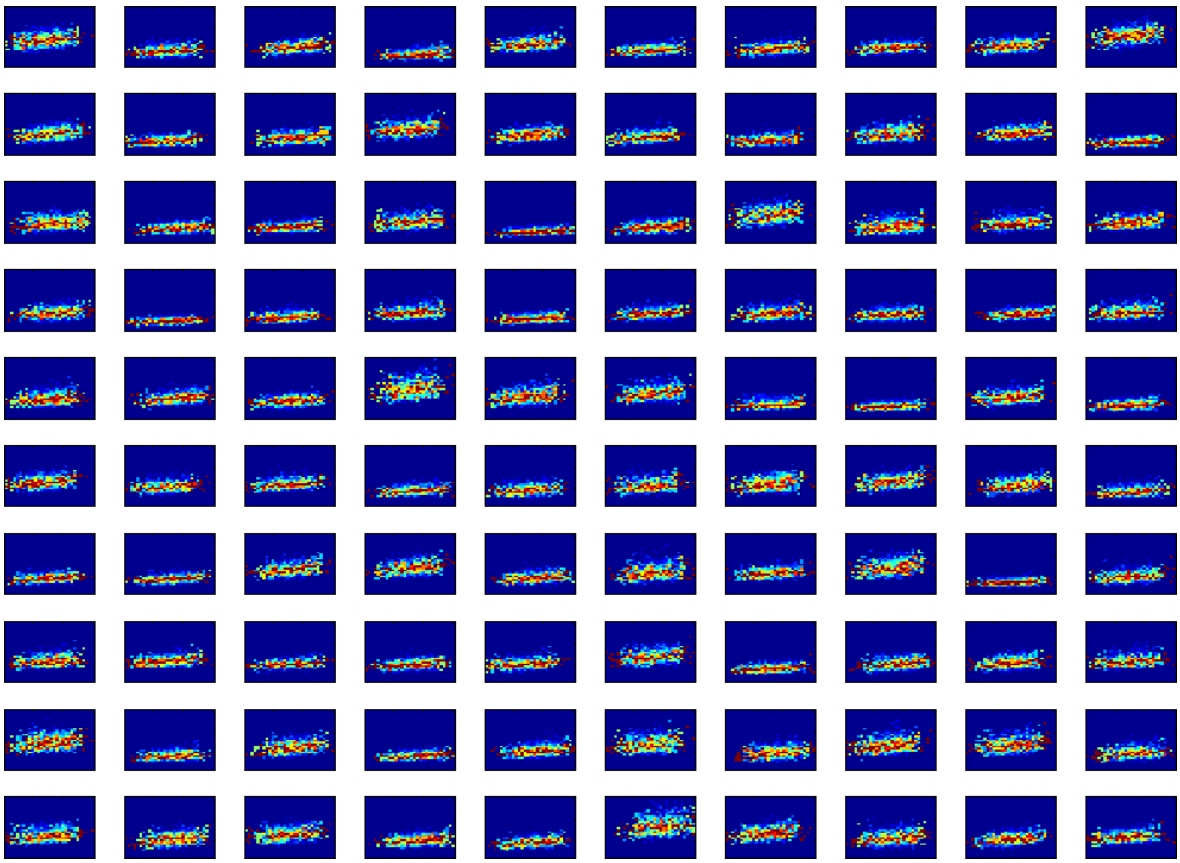


Figure A.3.: The $\Delta G \Delta P$ plots for 100 samples of the genotype space of vector field embryogeny setup with six basic elements are depicted. Each plot is created in the same way as the plot in Figure 6.1, using the same colorbar.

B. Sequence Diagrams of the Artificial Development-Simulation Environment and the Vector Field Embryogeny Simulation

The first sequence diagram depicts the implementation of development and fitness calculation in the Artificial Development-simulation. The example uses the sphere cell model, and consists of the classes

- LetItGrow: This is a placeholder class, and is used to initialize development and collect the fitness afterward. It can easily be replaced with an arbitrary evolutionary optimization technique.
- FitnessFunction: This class organizes the developmental process. It instantiates the development environment and contains the major developmental loop.
- BGene: This class transforms the unstructured vector of design variables into the sub-vector structure consisting of genes, regulatory units, and structural units, and provides helper functions such as the calculation of gene activation.
- SphereEgg: This class contains the implementation of cellular growth for the sphere cell model. Transcription factors and cells are instantiated from here.
- SphereCell: The class representing cells inside the developmental model. Cells possess a pointer to the genetic information, their own positional information, and a number of type attributes that are used to control e.g. differential adhesion or cell type.

The second sequence diagram depicts the vector field embryogeny-program for the simulation of a dynamical system. It consists of the main class DynSysMain.cpp from which the vector field embryogeny phase space model implemented in DynSys.cpp is instantiated.

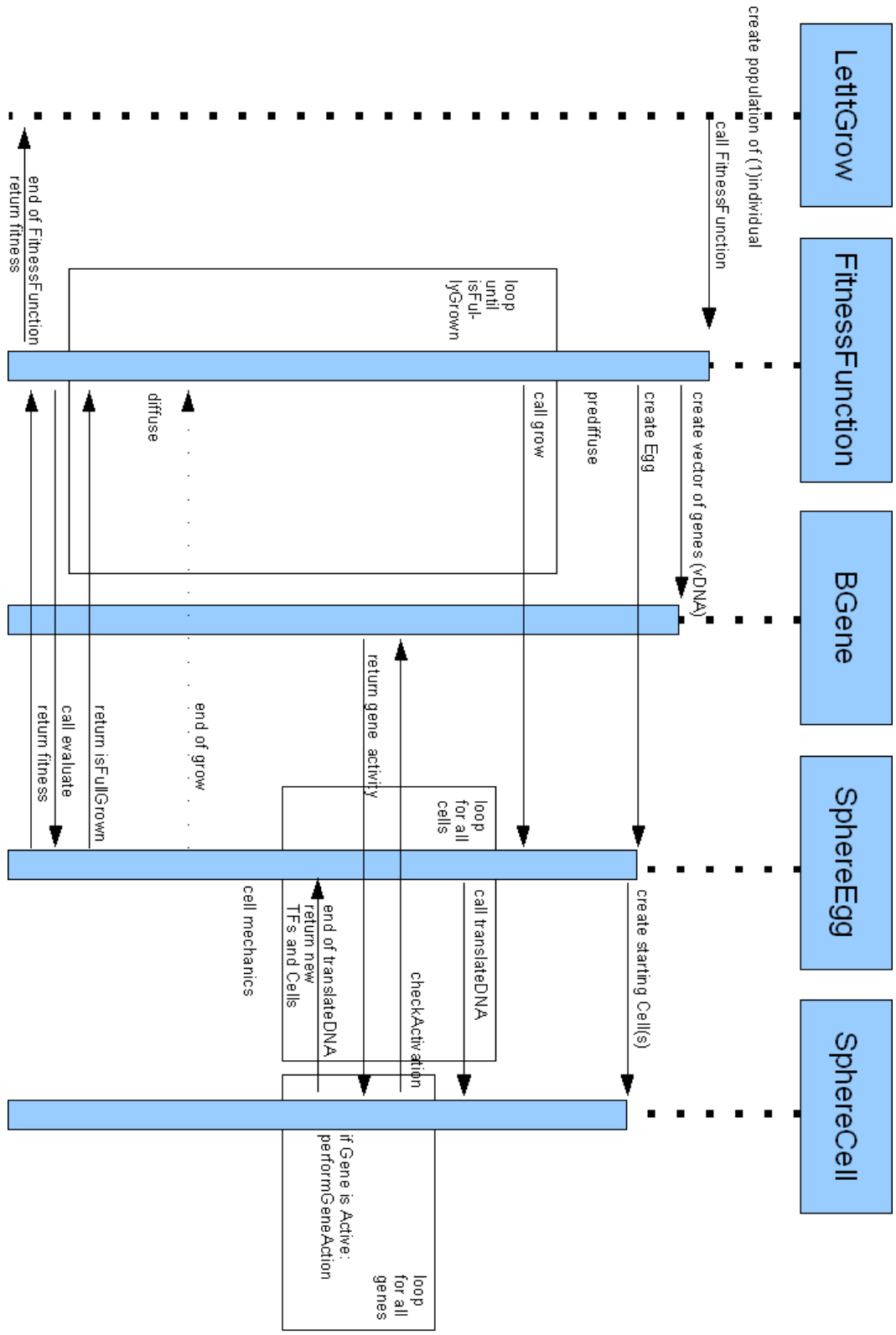


Figure B.1.: The sequence diagram of development and fitness calculation in the Artificial Development-simulation.

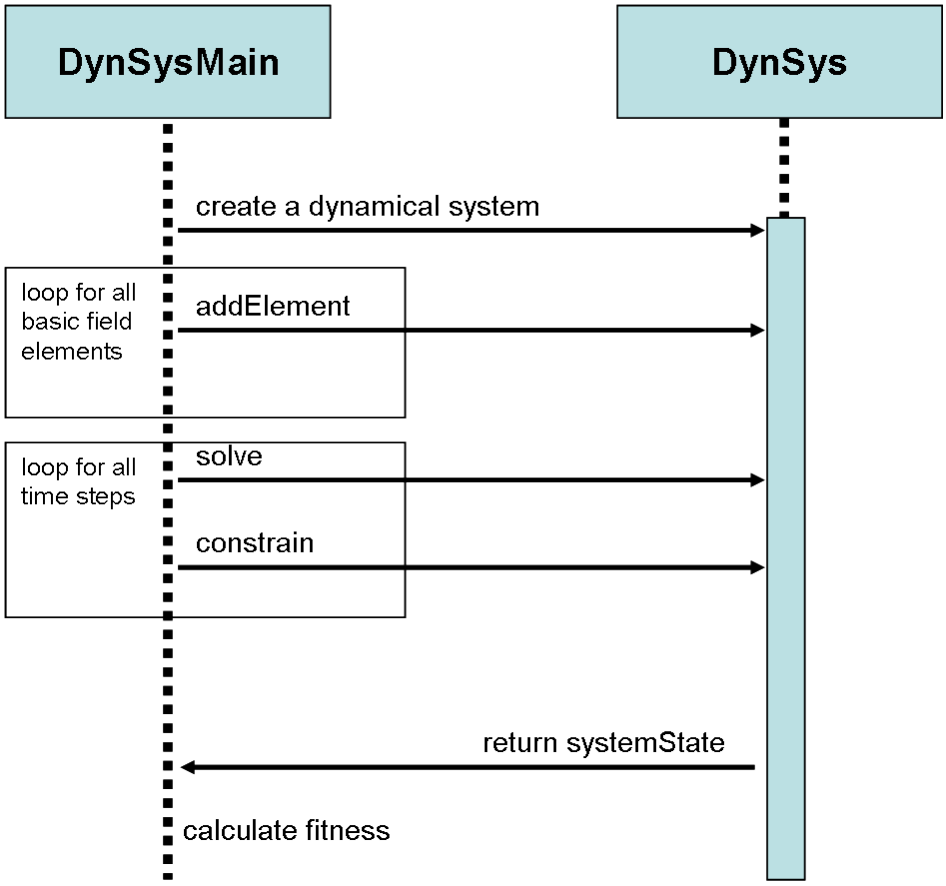


Figure B.2.: The sequence diagram of the vector field embryogeny simulation.

C. Bi-Linear Energy Calculation

In this thesis, the bi-linear energy E is used as a measure for stability of mechanical structures. in general, it is defined as

$$E = \int_{\Omega} \mathbf{u}^T \mathbf{K} \mathbf{u}. \tag{C.1}$$

In this equation, $\mathbf{u}$ denotes the displacements of the elements, while $\mathbf{K}$ is the stiffness matrix. Ω is the design domain. thus, the total bilinear energy is calculated by summing up its individual components for each finite element. Note that the bilinear energy grows with the square of the displacements. It is suitable as a measure for stability, since it includes not only the displacements alone, but also the material properties: a stiff material in a design, that is displaced heavily should have a lower quality than a 'softer' material with the same displacement. Note also that for the calculation of fitness f, only the maximal bilinear energy over all finite elements is used, which denotes the 'weakest' point of the design with respect to its load: $f = \max_{\Omega}(\mathbf{u}^T \mathbf{K} \mathbf{u})$.

List of Publications by the Author

The results presented in this thesis have partly appeared in the following peer reviewed publications:

T. Steiner, M. Olhofer, B. Sendhoff (2006). Towards Shape and Structure Optimization with Evolutionary Development. *Proceedings of the tenth international conference on the simulation and synthesis of living systems (ALIFE X, 2006).*
T. Steiner, L. Schramm, Y. Jin, B. Sendhoff (2007). Emergence of Feedback in Artificial Gene Regulatory Networks. *Proceedings of the Congress on Evolutionary Computation 2007.*
T. Steiner, Yaochu Jin, Bernhard Sendhoff (2008). A Cellular Model for the Evolutionary Development of Lightweight Material with an Inner Structure. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2008). ACM, New York.*
T. Steiner, Y. Jin, B. Sendhoff (2009). Vector Field Embryogeny. *PLoS ONE, Vol. 4(12) 2009: e8177.*
T. Steiner, Jens Trommler, Martin Brenn, Yaochu Jin, Bernhard Sendhoff (2009). Global Shape with Morphogen Gradients and Motile Polarized Cells. *Proceedings of the 2009 Congress on Evolutionary Computation, Trondheim.*
T. Steiner, Y. Jin, B. Sendhoff (2010). Evolving Heterochrony for Cellular Differentiation Using Vector Field Embryogeny. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2010).*
T. Steiner, Y. Jin, L. Schramm, B. Sendhoff (2010). Dynamic Links and Evolutionary History in Simulated Gene Regulatory Networks. In: *Handbook of Research on Computational Methodologies in Gene Regulatory Networks* (S. Das, D. Caragea, S. M. Welch, W. H. Hsu , editors). pp. 498-522, IGI Global.
T. Steiner, B. Sendhoff (2011). Evolvability of Graph- and Vector Field Embryogeny Representations. *Proceedings of the 2011 Congress on Evolutionary Computation, New Orleans.*

Bibliography

[1] Gilbert S, editor (2003) Developmental Biology. Sinauer Associates Inc.

[2] Alberts B, Johnson A, Lewis J, Raff M, Roberts K, et al. (2002) Molecular Biology of The Cell (fourth edition). Garland Science.

[3] Gerhart J, Kirschner J (1997) Cells, Embryos, and Evolution. Balckwell Science.

[4] Raff R (1996) The Shape of Life. University of Chicago Press.

[5] Wagner A (2000) Robustness against mutations in genetic networks of yeast. Nature Genetics 24: 355–361.

[6] Kondo S, Asai R (2002) A reactiondiffusion wave on the skin of the marine angelfish pomacanthus. Nature 376: 765–768.

[7] Turing A (1952) The chemical basis of morphogenesis. Philosophical Transactions of the Royal Society London B 237: 37–72.

[8] Harland R, Gerhart J (1997) Formation and function of spemann's organizer. Annual review of cell and developmental biology 13: 611–667.

[9] Bonner J (2000) First Signals. Princeton University Press.

[10] Darwin C (1859) On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life. John Murray.

[11] SR W (2006) Eukaryotic transposable elements: Teaching old genomes new tricks. In: Caporale L, editor, The Implicit Genome. pp. 138-165.

[12] Klingenberg C (1998) Heterochrony and allometry: the analysis of evolutionary change in ontogeny. Biol Rev 73: 79–123.

[13] Ohno S (1970) Evolution by gene duplication. Springer.

[14] Furusawa C, Kaneko K (1998) Emergence of multicellular organisms with dynamic differentiation and spatial pattern. Artificial Life 4: 79–93.

[15] Hogeweg P (2000) Evolving mechanisms of morphogenesis: on the interplay between differential adhesion and cell differentiation. Journal of Theoretical Biology 203: 317–333.

[16] Järvinen E, Salazar-Ciudad I, Birchmeier W, Taketo M, Jernvall J, et al. (2006) Continuous tooth generation in mouse is induced by activated epithelial wnt/β-catenin signaling. Proceedings of the National Academy of Science (PNAS) 103: 18627-18632.

[17] Cickovski T, Aras K, Swat M, Merks R, Glimm T, et al. (2007) From genes to organisms via the cell: A problem-solving environment for multicellular development. Computing in Science and Engineering 9: 50-60.

[18] Kitano H (1990) Designing neural networks using genetic algorithm with graph generation system. Complex Systems 4: 461–476.

[19] Siddiqi A, Lucas S (1998) A comparison of matrix rewriting versus direct encoding for evolving neural networks. In: Proceedings of the 1998 IEEE International Conference on Evolutionary Computation.

[20] Sims K (1994) Evolving 3d morphology and behavior by competition. In: Proceedings of the Artificial Life IV.

[21] Fleischer K, Barr A (1993) A simulation testbed for the study of multicellular development: The multiple mechanisms of morphogenesis. In: Proceedings of the Artificial Life III conference. Addison-Wesley, pp. 389–416.

[22] Stanley K, Miikkulainen R (2003) A taxonomy for artificial embryogeny. Artificial Life 9: 93–130.

[23] Kumar S, Bentley P, editors (2003) On Growth, Form and Computers. Elsevier.

[24] Lindenmayer A (1968) Mathematical models for cellular interaction in development: Parts i and ii. Journal of Theoretical Biology 18: 280–315.

[25] Gruau F, Whitley D, Pyeatt L (1996) A comparison between cellular encoding and direct encoding for genetic neural networks. In: Proceedings of the First Annual Conference on Genetic Programming.

[26] Kitano H (1998) Building complex systems using developmental process: An engineering approach. Lecture Notes In Computer Science 1478: 218-229.

[27] Haddow P, Tufte G, van Remortel P (2001) Shrinking the genotype: L-systems for ehw? In: Proceedings of the ICES 2001. volume 2210 of *Lecture Notes on Computer Science*.

[28] Hornby GS, Pollack J (2002) Creating high-level components with a generative representation for body-brain evolution. Artificial Life 8: 223–246.

[29] Hornby G (2004) Functional scalability through generative representations: the evolution of table designs. Environment and Planning B: Planning and Design 31: 569–587.

[30] Mitchell M (1998) Computation in cellular automata: A selected review. In: Gramss T, Bornholdt S, Gross M, Mitchell M, Pellizzari T, editors, Non-Standard Computation. Wiley-VCH Verlag GmbH, pp. 95–140.

[31] Basanta D, Bentley P, Miodownik M, Holm E (2003) Evolving cellular automata to grow microstructures. In: 6th European Conference on Genetic Programming (EuroGP). Springer Berlin / Heidelberg, pp. 1–10. doi:10.1007/3-540-36599-0.

[32] Kowaliw T (2007) A Good Number of Forms Fairly Beautyful. Ph.D. thesis, Department of Computer Science, Concordia University, Montreal, Canada.

[33] Basanta C, Miodownik M, Baum B (2008) The evolution of robust development and homeostasis in artificial organisms. PLoS Computational Biology .

[34] Gordon T, Bentley P (2005) Development brings scalability to hardware evolution. In: Proceedings of the 2005 NASA/DoD Conference of Evolution Hardware.

[35] Harding S, Miller J (2006) The dead state: A comparison between developmental and direct encodings. In: Proceedings of the GECCO conference.

[36] Jakobi N (1995) Harnessing morphogenesis. In: Proceedings of Information Processing in Cells and Tissues. pp. 29–41.

[37] Dellaert F, Beer R (1996) A developmental model for the evolution of complete autonomous agents. In: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior. pp. 393–401.

[38] Eggenberger-Hotz P (2004) Asymmetric cell division and its integration with other developmental processes for artificial evolutionary systems. In: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems Alife IX.

[39] Stanley K (2004) Efficient Evolution of Neural Networks through Complexification. Ph.D. thesis, Artificial Intelligence Laboratory, The University of Texas at Austin.

[40] Reisinger J, Miikkulainen R (2007) Acquiring evolvability through adaptove representations. In: Proceedings of the GECCO '07.

[41] Bentley P (2005) Investigations into graceful degradation of evolutionary developmental software. Natural Computing 4: 417–437.

[42] Bowers C (2006) Simulating Evolution with a Computational Model of Embryogeny. Ph.D. thesis, School of Computer Science, The University of Birmingham.

[43] Federici D, Downing K (2006) Evolution and development of a multicellular organism: Scalability, resileance, and neutral complexification. Artificial Life 12: 381–409.

[44] Reil T (2003) Artificial genomes as models of gene regulation. In: Kumar S, Bentley P, editors, On Growth, Form and Computers. Elsevier, pp. 256–277.

[45] Knabe J (2009) Evolvability of Computational Genetic Regulatory Networks. Ph.D. thesis, School of Computer Science, Faculty of Engineering and Information Sciences, University of Hertfordshire.

[46] Alon U (2006) An Introduction to Systems Biology - Design Principles of Biological Circuits. Chapman and Hall/CRC.

[47] Banzhaf W (2003) Artificial regulatory networks and genetic programming. In: Yu T, Riolo R, Worzel B, editors, Genetic Programming - Theory and Applications. Kluwer Academic, pp. 43–61.

[48] Geard N, Wiles J (2005) A gene network model for developing cell lineages. Artificial Life 11: 249–267.

[49] Psujek S, Beer R (2008) Developmental bias in evolution: evolutionary accessibility of phenotypes in a model evo-devo system. Evolution & Development 10: 375–390.

[50] Sekanina L, Bidlo M (2005) Evolutionary design of arbitrarily large sorting networks. Genetic Programming and Evolvable Machines 6: 319–347.

[51] Nolfi S, Parisi D (1995) Evolving artificial neural networks that develop in time. In: Proceedings of the Third European Conference on Advances in Artificial Life. Lecture Notes in Computer Science, pp. 353–367.

[52] Miller J, Thomson P (2000) Cartesian genetic programming. In: Proceedings of the 3rd European Conference on Genetic Programming.

[53] Miller J, Banzhaf W (2003) Evolving the program for a cell: From french flags to boolean circuits. In: Kumar S, Bentley P, editors, On Growth, Form and Computers. Elsevier, pp. 278 –301.

[54] Khan G, Miller J, Halliday D (2008) Developing neural structure of two agents that play checkers using cartesian genetic programming. In: Proceedings of the GECCO'08. pp. 2169–2174.

[55] Harding S, Miller J, Banzhaf W (2007) Self-modifying cartesian genetic programming. In: Proceedings of the GECCO'07.

[56] Stanley K (2007) Compositional pattern producing networks: A novel abstraction of development. Genetic Programming and Evolvable Machines 8: 131–162.

[57] Tufte G, Haddow P (2003) Building knowledge into developmental rules for circuit design. In: Proceedings of t he ICES 2003. volume 2606 of *Lecture Notes on Computer Science*, pp. 69–80.

[58] Federici D (2005) Evolving developing spiking neural networks. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC).

[59] Buck M, Nehaniv C (2006) Discrete developmental genetic regulatory networks for the evolution of cooperation. In: Developmental Systems: Papers from the 2006 AAAI Fall Symposium.

[60] Grajdeanu A, Kumar S (2006) A novel developmental system for the study of evolutionary design. In: Developmental Systems: Papers from the 2006 AAAI Fall Symposium.

[61] Devert A, Bredeche N, Schoenauer M (2007) Robust multi-cellular developmental design. In: Proceedings of the 9th annual conference on Genetic and evolutionary computation. pp. 982–989.

[62] Roggen D, Federici D, Floreano D (2007) Evolutionary morphogenesis for multicellular systems. Genetic Programming and Evolvable Machines 8: 61–96.

[63] Chavoya A, Duthen Y (2008) A cell pattern generation model based on an extended artificial regulatory network. BioSystems 94: 95–101.

[64] Cussat-Blanc S, Duthen Y (2008) From single cell to simple creature morphology and metabolism. In: Proceedings of the International Conference on Artificial Life, ALIFE XI.

[65] Harding S, Banzhaf W (2008) Artificial development. In: Würz R, editor, Organic Computing. Springer, pp. 201–220.

[66] Bentley P, Kumar S (1999) Three ways to grow designs: A comparison of evolved embryogenies for a design problem. In: Genetic and Evolutionary Computation Conference. Morgan Kaufmann, pp. 35–43.

[67] Astor JC, Adami C (2000) A developmental model for the evolution of artificial neural networks. Artificial Life 6: 189–218.

[68] Bongard J, Pfeifer R (2003) Evolving complete agents using artificial ontogeny. In: Hara F, Pfeifer R, editors, Morpho-functional Machines: The New Species (Designing Embodied Intelligence). Springer, pp. 237-258.

[69] Doursat R (2008) Programmable architectures that are complex and self-organized: From morphogenesis to engineering. In: Proceedings of the ALIFE XI: Eleventh International Conference on the Simulation and Synthesis of Living Systems. MIT Press, pp. 181–188.

[70] Spector L, Klein J, Feinstein M (2007) Division blocks and the open-ended evolution of development, form, and behavior. In: Proceedings of the GECCO'07.

[71] Komosinski M, Rotaru-Varga A (2001) Comparison of different genotype encodings for simulated three-dimensional agents. Artificial Life 4: 395–418.

[72] Framesticks website. http://www.frams.alife.pl.

[73] Kawamura H, Ohmori H (2002) Computational morphogenesis of discrete structures via genetic algorithms. Memoirs of the School of Engineering, Nagoya University 53: 28–55.

[74] Devert A, Bredeche N, Schoenauer M (2008) Artificial ontogeny for truss structure design. In: Proceedings of the SASOW, Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops. pp. 298–305.

[75] Yogev O, Shapiro A, Antonsson E (2008) Modularity and symmetry in computational embryogeny. In: Proceedings of the GECCO'08.

[76] Taura T, Nagasaka I (1999) Adaptive-growth-type 3d representation for configuration design. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 13: 171–184.

[77] Downing K (2006) A neural-group basis for evolving and developing neural networks. In: Proceedings of the AAAI-Devp.

[78] Lehre P (2006) Complexity and Geometry in Artificial Development. Ph.D. thesis, School of Methematics and Electrical Engrineering, Faculty of Information Technology, Norwegian University of Science and Technology, Trondheim, Norway.

[79] Beer R (1995) On the dynamics of small continuous-time recurrent neural networks. Adaptive Behavior 3: 469–509.

[80] Yao X, Liu Y (1997) A new evolutionary system for evolving artificial neural networks. IEEE Transactions on Nueral Networks 8: 694–713.

[81] Eggenberger P (1997) Evolving morphologies of simulated 3D organisms based on differential gene expression. In: Proceedings of the 4th European Conference on Artificial Life. pp. 205–213.

[82] Khalil H (2002) Nonlinear Systems. Prentice Hall.

[83] Ferrell Jr J (2002) Self-perpetuating states in signal transduction: positive feedback, double-negative feedback and bistability. Curr Opin Cell Biol 14: 140–148.

[84] Goldbeter A (2002) Computational approaches to cellular rhythms. Nature 420: 238–245.

[85] Hardin P, Hall J, Rosbash M (1990) Feedback of the Drosophila period gene product on circadian cycling of its messenger RNA levels. Nature 343: 536–540.

[86] Dunlap J (1999) Molecular bases for circadian clocks. Cell 96: 271–290.

[87] Luscombe NM, Babu MM, Yu H, Snyder M, Teichmann SA, et al. (2004) Genomic analysis of regulatory network dynamics reveals large topological changes. Nature 431: 308–312.

[88] Goldbeter A, Gonze D, Houart G, Leloup J, Halloy J, et al. (2001) From simple to complex oscillatory behavior in metabolic and genetic control networks. Chaos 11: 247–260.

[89] Leloup J, Goldbeter A (1999) Chaos and birhythmicity in a model for circadian oscillations of the PER and TIM proteins in Drosophila. J theor Biol 198: 445–459.

[90] Chen G, Mischaikow K, Laramee R, Pilarczyk P, Zhang E (2007) Vector field editing and periodic orbit extraction using morse decomposition. IEEE Transactions on Visualization and Computer Graphics 13: 1077–2626.

[91] Zhang E, Mischaikow K, Turk G (2006) Vector field design on surfaces. ACM Transactions on Graphics 25: 1294–1326.

[92] Praun E, Hoppe H, Webb M, Finkelstein A (2001) Real-time hatching. In: Proceedings of the ACM SIGGRAPH. ACM, pp. 581–586.

[93] Nicolis G (1995) Introduction to Nonlinear Science. Cambridge University Press.

[94] Schwefel HP (1995) Evolution and Optimum Seeking. Wiley-Interscience.

[95] Beyer H, Schwefel H (2002) Evolution Strategies – a comprehensive introduction. Natural Computing 1: 3–52.

[96] Steiner T, Olhofer M, Sendhoff B (2006) Towards shape and structure optimization with evolutionary development. In: Proceedings of the ALIFE X: Tenth International Conference on the Simulation and Synthesis of Living Systems. pp. 70–76.

[97] Steiner T, Schramm L, Jin Y, Sendhoff B (2007) Emergence of feedback in artificial gene regulatory networks. In: Proceedings of the CEC 2007: Congress on Evolutionary Computation. pp. 867–874.

[98] Steiner T, Jin Y, Sendhoff B (2008) A cellular model for the evolutionary development of lightweight material with an inner structure. In: GECCO'08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation. pp. 851–858.

[99] Steiner T, Trommler J, Brenn M, Jin Y, Sendhoff B (2009) Global shape with morphogen gradients and motile polarized cells. In: Proceedings of the 2009 Congress on Evolutionary Computation. pp. 2225-2232.

[100] Steiner T, Jin Y, Schramm L, Sendhoff B (2009) Dynamic links and evolutionary history in simulated gene regulatory networks. In: Das S, Caragea D, Welch S, Hsu WH, editors, Handbook of Research on Computational Methodologies in Gene Regulatory Networks. IGI Publishing, pp. 498-522.

[101] Jin Y, Meng Y, Sendhoff B (2009) Influence of regulation logic on the easiness of evolving sustained oscillation for gene regulatory networks. In: Proceedings of the IEEE Symposium on Artifical Life. pp. 61–68.

[102] Deb K (2001) Multi-Objective Optimization using Evolutionary Algorithms. Wiley.

[103] Deb K, Agrawal RB (1995) Simulated binary crossover for continuous search space. Complex Systems 2: 115–148.

[104] Deb K, Goyal M (1996) A combined genetic adaptive search (geneas) for engineering design. Computer Science and Informatics 4: 30–45.

[105] Bendsoe M, Sigmund O (2003) Topology Optimization - Theory, Methods and Applications. Springer.

[106] Steiner T, Jin Y, Sendhoff B (2009) Vector field embryogeny. PLoS ONE 4.

[107] Steiner T, Jin Y, Sendhoff B (2010) Evolving heterochrony for cellular differentiation using vector field embryogeny. In: Proceedings of the 12th annual conference on genetic and evolutionary computation. pp. 571–578.

[108] Nüsslein-Volhard C, Wieschaus E (1980) Mutations affecting segment number and polarity in Drosophila. Nature 287: 795–801.

[109] Ingram P, Stumpf M, Stark J (2006) Network motifs: structure does not determine function. BMC Genomics 7: 108+.

[110] Doursat R (2009) Facilitating evolutionary innovation by developmental modularity and variability. In: GECCO'09: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation. pp. 683–690.

[111] Gayon J (2000) History of the concept of allometry. American Zoologist 40: 748–758.

[112] Jeffery J, Bininda-Emonds O, Coates M, Richardson M (2002) Analyzing evolutionary patterns in amniote embryonic development. Evolution and Development 4: 292–302.

[113] Gould S (2000) Of coiled oysters and big brains: how to rescue the terminology of heterochrony, now gone astray. Evolution and Development 2: 241–248.

[114] Ambros V (1997) Heterochronic genes. In: Riddle D, Blumenthal T, Meyer B, Priess J, editors, C. elegans II. Cold Spring Harbor Laboratory Press.

[115] Wolpert L (1983) Constancy and change in the development and evolution of pattern. In: Goodwin B, Holder N, Wylie C, editors, Development and Evolution, Cambridge University Press. pp. 47–57.

[116] Haken H (1988) Information and Self-Organization. Springer.

[117] Koch A, Meinhardt H (1994) Biological pattern formation: from basic mechanisms to complex structures. Reviews of Modern Physics 66: 1481–1507.

[118] Colegrave N, Collins S (2008) Experimental evolution: experimental evolution and evolvability. Heredity 100: 464-470.

[119] Sendhoff B, Kreutz M, von Seelen W (1997) A condition for the genotype-phenotype mapping: Causality. In: Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA '97). pp. 73–80.

[120] Knoblauch A, Palm G, Sommer F (2010) Memory capacities for synaptic and structural plasticity. Neural Computation 22: 289-341.