Anticipating Intentions as Gestalt Formation: A Model Based on Neural Competition

Martin Meier, Robert Haschke and Helge J. Ritter [*]

Neuroinformatics Group
Bielefeld University, 33501 Bielefeld, Germany
{mmeier,rhaschke,helge}@techfak.uni-bielefeld.de

Abstract. Anticipating the intentions of others is a key ability for cognitive interaction that is still not well understood and poorly replicated in artificial systems, such as robots. In this contribution we explore a neural model of Gestalt formation as a potential approach to intention anticipation. The idea is to view the already recognizable part of an ongoing action, together with the underlying intention, as a "Gestalt", which has to be completed when only the recognizable action part is given as an available fragment. To test this idea, we extend a previously developed model of competing neural layers for Gestalt formation by a "hallucination mechanism" that constructs the most likely completion of a given action fragment. We show that the resulting model can successfully anticipate cooperative moves of a human player in a two-person interaction scenario.

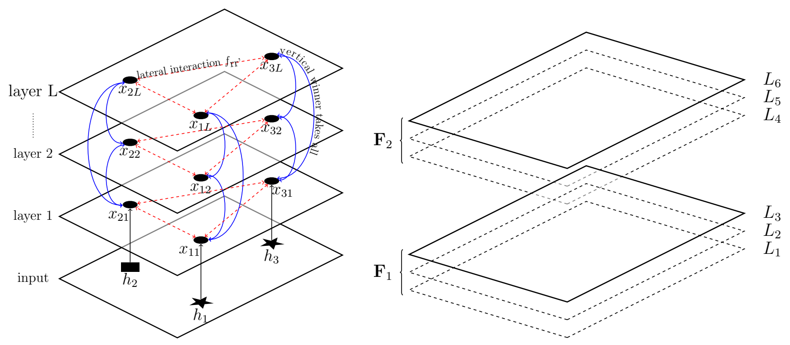
1 Introduction

When humans try to solve a task cooperatively, they can adapt their behavior to each other without explicit negotiation. This ability has many facets, one of them being the intuitive anticipation of the intentions of their collaboration partners. As an example, one can look at the task waiters perform when laying out dishes for a dinner. If one waiter starts to lay out the dishes, another one will most likely start to place the cutlery, because the dishes are taken care of and the cutlery is needed to complete the task.

This ability of "seeing" what the other will do to complete my fragmentary action has some resemblance to the completion of a fragmentary "Gestalt", viewing the fragmentary action as the incomplete pattern for which a "good continuation" is sought. This suggests to apply models for Gestalt formation to map the given input into a dynamics that completes the fragmentary actions towards a "good Gestalt", such as, e.g. "cooperation". The driving dynamics itself would then be in the role of the "intention" that completes the action.

To explore this idea in a concrete fashion, we apply the Competitive Layer Model [11]. The CLM has been proven feasible for a large set of segmentation

[*] This work has been conducted within and funded by the German collaborative research center "SFB 673: Alignment in Communication" granted by DFG.



(a) CLM (b) Multiple layer classes

Fig. 1. The left image shows the Competitive Layer Model. At the bottom are three inputs $v_{1\cdots 3}$ and their corresponding neurons $x_{1\cdots 3}$ within each layer $L = 1 \ldots \alpha$. The right image shows an example for a CLM with six layers and two different classes of interaction functions. The layers $L_1, \ldots, L_3$ respond to interaction function $\mathbf{F}_1$ and the layers $L_4, \ldots, L_6$ respond to interaction function $\mathbf{F}_2$.

and grouping problems in the image processing domain [9,14] and for automatic task segmentation [10].

The CLM allows the grouping of features based on the *Laws of Gestalt Theory* [6], a theory from the field of cognitive psychology, which tries to describe how humans perceive complex scenes. As a result of a combination of excitatory and inhibitory couplings of neurons, similar features form reinforcing groups of attractors, while simultaneously suppressing other, less similar features.

However, a human can not only correctly group features, but also imagine well fitting completions. Hence, we apply the approach presented in [7], with which it is possible to evaluate the compatibility of previously unknown features with respect to a CLM grouping result. The presented approach is extended with a technique to automatically find well fitting completions.

The recognition of intentions is of particular interest in the field of human-robot interaction and has been approached with a variety of techniques. In [5], the recognition of intention was done based on Hidden Markov Models, whereas [3] integrates Markov Models, Bayesian Networks and machine learning techniques in a hierarchical model to achieve this goal. Encoding a set of possible intentions in a Finite State Machine and learning of the transition probabilities has been done in [1].

2 The Competitive Layer Model

The Competitive Layer Model *(CLM)* is a recurrent neural network which consists of $L \times N$ linear threshold neurons. These neurons are arranged in $\alpha = 1 \ldots L$ layers, where each layer holds a total number of $r = 1 \ldots N$ neurons, as depicted in Fig. 1(a). The activity of a neuron in a single layer is denoted as $x_{r\alpha}$. The dynamics is designed such, that compatible features induce high neuron activities

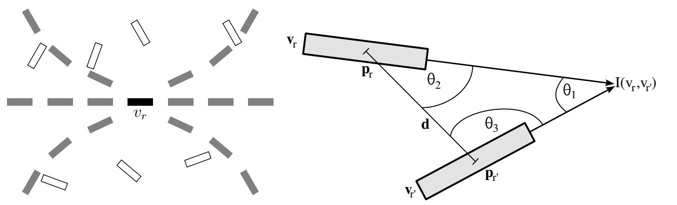


Fig. 2. An exemplary interaction function for good continuation is shown on the left. The central feature v_r gets excitatory responses from the gray shaded features and inhibitory responses from the white. The right image shows parameters for the compatibility of oriented edge elements. The used parameters are the distance between the centers of the elements, given by p_r and $p_{r'}$ and the three angles $\theta_{1,2,3}$.

within individual layers α, indicating perceptual grouping of those features. All neurons within a column r correspond to the same feature v_r.

The neurons in each layer are coupled with lateral interaction weights $f_{rr'}$, which are determined by the similarity between two features v_r and $v_{r'}$, e.g. positive values for compatible and negative values for dissimilar features. This compatibility measure needs to be explicitly specified by a symmetric interaction function

$$f_{rr'} = \mathbf{f}(v_r, v_{r'}) = \mathbf{f}(v_{r'}, v_r). \tag{1}$$

Because the interaction weights need only be computed once, they can be stored as a symmetric interaction matrix $\mathbf{F} = f_{rr'}$.

An example for an interaction function that models the Gestalt Law of good continuation is given in Fig. 2(a). Features which form a smooth path with respect to the central feature v_r create excitatory responses while other features which do not fit create inhibitory responses. To assure that an input feature v_r is only represented by an active neuron in a single layer, the corresponding neurons in each layer are coupled with a columnar *winner-takes-all (WTA)* competition.

Combining both components, the lateral interaction function and the column-wise *WTA* circuit, a linear threshold dynamics can be summarized with the following update rule:

$$\dot{x}_{r\alpha} = -x_{r\alpha} + \sigma(J(h_r - \sum_{\beta} x_{r\beta}) + \sum_{r'} f_{rr'} x_{r'\alpha}) \tag{2}$$

Here, $\sigma(x) = \max(0, x)$ is the linear threshold function, $J(h_r - \sum_\beta x_{r\beta})$ represents the columnar *WTA* with a weighting constant J, and h specifies the overall columnar activity and thus denotes the importance of a feature. Throughout this article we assume that all features are equally important, therefore $h = 1$. The lateral interaction between the features at position r and r' is computed with $\sum_{r'} f_{rr'} x_{r'\alpha}$. For a conclusive analysis of the CLM dynamics please refer to [14].

The CLM architecture as described above can only hold one interaction function at a time and therefore only respond to a specific type of feature compatibility. To circumvent this limitation, we apply the idea presented in [12] and introduce different types of layer classes. As shown in the example in Fig. 1(b), a set of layers is grouped in a layer class which responds to a class specific interaction function. In the illustration in Fig. 1(b) the layers $L_1, \ldots, L_3$ respond to the interaction function $\mathbf{F}_1$ while the layers $L_4, \ldots, L_6$ respond to $\mathbf{F}_2$.

In previous works, for example in [14], the interaction function had to be hand crafted. To gain a better generalization capability and to simplify the task of choosing a suitable interaction function, we apply the techniques presented in [13] to learn an interaction function from labeled training data.

The approach presented in [13] relies on a set of labeled training data which is used twofold. In the first step, for each feature pair $(v_r, v_{r'})$ a proximity vector $d(v_r, v_{r'})$ is calculated. These proximity vectors are subsequently clustered to yield a compact representation of the proximity space. In the simplest case, the proximity vector may simply stack the original feature vectors, i.e. $d_{rr'} = [v_r^t, v_{r'}^t]^t$. However, if more insight into the task is available, more elaborate distance vectors can be computed, expressing the similarity of two features within a multi-dimensional vector. The objective is to capture properties of typical feature pairs within the given domain. For example, Fig. 2(b) shows the proximity function employed for oriented edge features.

In a second step, the label information is used to assign positive or negative interaction weights to each cluster prototype. Prototypes mainly corresponding to feature pairs of the same group, i.e. being compatible, will get a positive weight. Contrarily, pairs of features originating from different groups will generate negative interaction weights. Exploiting the frequency of the positive and negative interaction labels, it is possible to create a set of basis interaction functions. For a detailed derivation please refer to [13].

3 "Intention Field" for Anticipating new Features

To extend the capabilities of the CLM from grouping towards amending sparse groups, we utilize an approach presented in [7]. Given a CLM which has already converged to a grouping result, it is possible to use this result to find previously unknown features which fit well to the already known features.

3.1 Quality Measurement

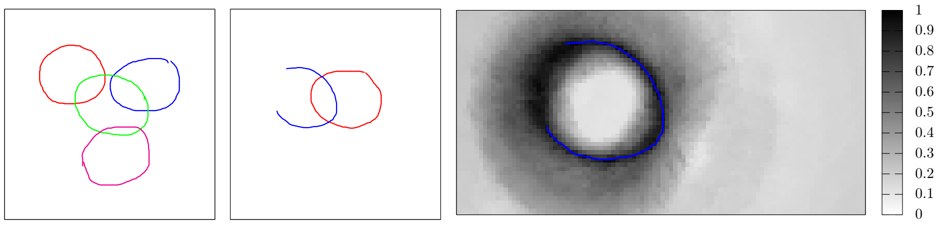
As presented in [7], it is possible to evaluate the quality Q of a new feature v_{new} by calculating its compatibility to the achieved grouping result. To this end, first an interaction vector $\mathbf{m}$ for the new feature value v_{new} has to be created

$$\mathbf{m} = (f(v_{new}, v_0), f(v_{new}, v_1), \cdots, f(v_{new}, v_r))^T \tag{3}$$

utilizing the previously learned interaction function $f_{rr'}$. The support for the new feature is calculated as

$$x_{v_{new}\alpha} = \mathbf{m}^T \cdot \boldsymbol{x}_\alpha \tag{4}$$

for all layers $L = 1 \ldots \alpha$. We will view the mapping $I : (v, v_{new}) \rightarrow x_{v_{new}}$ from the existing pattern v to the support as the "intention" $I(v, .)$ that is associated with



(a) Training data. (b) Grouping result for test data. (c) Activities from "intended" completions for the grouping result from (b).

Fig. 3. The labeled training input to learn the interaction function is shown in 3(a). The shapes are composed of edge features as depicted in Fig. 2(b). 3(b) presents the grouping result from the CLM for two test shapes. Their "intended" completions are displayed in 3(c), where the quality of the "intentions" produces a steep slope inside and outside of the circular input shapes.

the existing pattern v. In this way, we obtain a computationally clear and concise representation of "intention" as a mapping that tells how strongly the system supports different possible completions v_{new} of a given partial input. To make the computed activity for the "intended" completing feature v_{new} comparable, we apply a normalization, which shall assure that the support for such a feature is in the range from 0 to 1 in each layer.

Assume we have a CLM which holds N features and is already converged. Because of the linear threshold $\sigma(x)$ from (2), every feature has an activity greater or equal to zero. Therefore, we calculate the normalization constant M_α for each layer as:

$$M_\alpha = \sum_{r=1}^{N} x_{r\alpha} \tag{5}$$

With N being the number of features in the original input and their corresponding neural activity $x_{r\alpha}$ in the layer α. Consequently, the quality Q of an "intended" completion can be determined using (4) and (5) as

$$Q_\alpha(v_{new}) = \frac{1}{M_\alpha} \cdot \mathbf{m}^T \cdot \boldsymbol{x}_\alpha \tag{6}$$

An example of this process is shown in Fig. 3. An interaction function is learned from the training set in Fig. 3(a), where the labels are indicated by different colors. These shapes are composed of oriented edge features as shown in Fig. 2(b). The grouping result of the CLM is shown in Fig. 3(b). Here the assignment of features to the same layer is expressed by the same color. To closer examine the approach, the best fitting "intended" completions for the incomplete shape are generated. Fig. 3(c) shows the maximal quality of "intended" completions for each pixel of the image. Please note that for visualization purposes the feature space was discretized. At each pixel position the maximal activity for a total number of 36 different orientations, reaching from 0° to 175°, was calculated and is shown in Fig. 3(c).

3.2 Finding good Features

The method used to display the quality of an "intended" completion in Fig. 3(c) is not suitable for real world problems. It discretizes the feature space and is computational intractable because of the brute force data generation over the whole space.

We therefore propose to use a sampling technique which has been proven feasible for path planning and protein folding, the transition based rapidly exploring random tree (T-RRT) [4]. T-RRT extends classical RRT to find good, cost-efficient paths in the presence of a cost function defined on the configuration space, e.g. finding paths along a valley in a mountainous region. Thus it applies a transition test supplementary to the state validity checking of RRT. The transition test, as presented in [4], needs a cost function c which evaluates the cost of a newly sampled configuration q. The probability of a transition from state q_t to a new state q_{t+1} is then determined by the Boltzmann distribution:

$$p_{t,t+1} = \begin{cases} e^{-\frac{\Delta c_{t,t+1}}{K \cdot T}} & if\ \Delta c_{t,t+1} > 0 \\ 1 & else \end{cases} \tag{7}$$

where $\Delta c_{t,t+1} = \frac{c_{t+1} - c_t}{dist(c_t, c_{t+1})}$, which determines the slope of the cost along the path. K is a constant which normalizes the costs. As proposed in [4], we set $K = \frac{c_{start} + c_{goal}}{2}$. The parameter T is a temperature to allow a simulated annealing behavior. If the transition test is successful, the temperature is decreased by a factor $T = \frac{1}{a} \cdot T$. If the test fails for a given number of steps $nFail$, the temperature is increased by $T = a \cdot T$. Here we also use the same values as in [4], namely $a = 2$ and $nFail = 100$.

3.3 Integration

With the given T-RRT algorithm and the quality measurement from (6), it is easily possible to create a cost function:

$$c(v_{new}) = 1 - Q(v_{new}) \tag{8}$$

which prefers paths along high quality features, thus exploring the "intention field" towards well fitting completions. We can consider the known features of the CLM as points on a trajectory through this space. The task of the T-RRT algorithm is to connect these points and fill in the sparse regions by finding good amendments, for example to complete the shape from Fig. 3(c).

4 Evaluation

To evaluate the proposed approach, we will first test our technique in an interaction game, where the task for our system is to anticipate which figure a user has in mind and complete it accordingly. The second part of the evaluation uses artificially generated shapes and introduces an error margin to gain a quantitative measure how good the generated features of our system are.

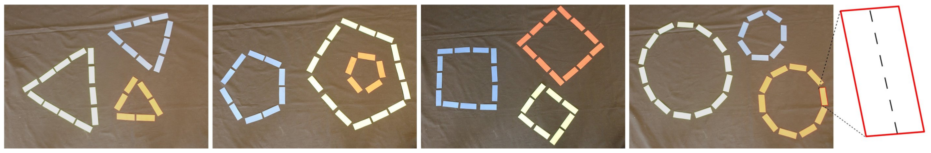


Fig. 4. Color labeled training patterns for the completion task. The rightmost part of the image shows the result of the used feature extraction. For each colored block an oriented rectangle is fitted and the longest axis of this rectangle is used as input for the CLM.

4.1 Interaction Game

The experimental evaluation consists of a cooperative figure completion task. At first we present some color labeled example figures to our system from which the interaction functions are learned. Each of the four images from Fig. 4 is used to learn the interaction function for a layer class. Each colored rectangle has the size of a wooden block from the Jenga game, as shown in Fig. 4. From these images the features are extracted using a simple color threshold and by applying a box fitting algorithm from *OpenCV*[1] to the filtered images. The longest axis of these boxes is used as an individual input feature for the CLM. The goal of the system is to anticipate which figure the human player has in mind and to complete it accordingly by suggesting a well-fitting location to place the next Jenga block using the presented approach. The actual placement of the wooden blocks was not yet realized by a robotic pick-and-place program, due to minor technical issues. Occasionally the human player may place a block by himself.

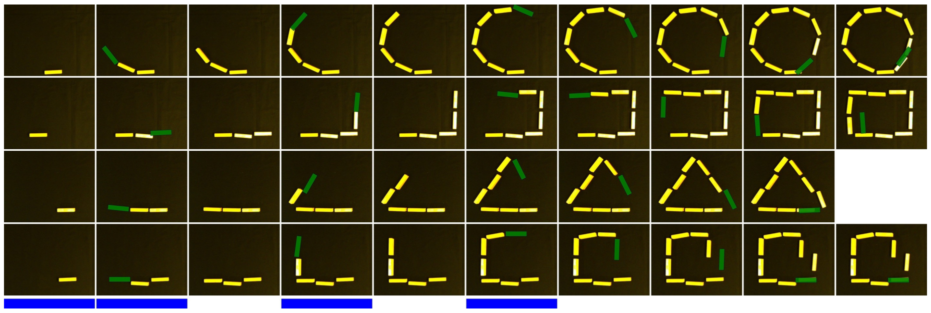


Fig. 5. Results from the cooperative shape completion task. Each of the four rows shows a sequence of the task. The color coding at the bottom of the figure indicates if it is the user's or the system's turn. A blue bar represents an action performed by the user and no bar stands for an action executed by the system. The green bars in each frame denote the anticipation of the system for the following frame. After the sixth frame the system is completing the shape autonomously.

[1] Open Source Computer Vision - http://opencv.org

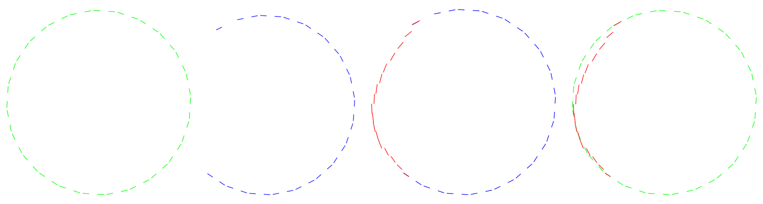

(a) Original (b) Sparse (c) “Intentions” (d) Comparison

Fig. 6. Illustration of the generation and evaluation of artificial shapes. The shape in the leftmost image serves as reference, which is used to generate a sparse shape as shown in Fig. 6(b) and as ground truth data for the error calculation. In Fig. 6(c) the completing features generated by our approach are shown in red. The rightmost image compares the completing features from Fig. 6(c) to the ground truth data. The mean mutual distance between these features is used as error margin.

When a new block is placed on the table, our system explores the neural potential in the vicinity of this block towards unoccupied regions. Because the geometry of a Jenga block is known and the task is to place one block at a time, the exploration radius is limited to the length of one block.

Four game sequences are shown in Fig. 5 corresponding to different geometric shapes to be laid out. Each row depicts a single trial which shows the sequence from left to right. The bottommost row indicates when our system or the user performed the shown action in that column. A blue label indicates that the action was performed by the user whilst no label represents that our system decided where to place the next block. Additionally, a green block in the sequence of images shows the anticipation of our system for the following frame.

The procedure was similar for all trials. The first two Jenga blocks were placed by the user. From block three to six the system and the user were taking turns. Eventually, the system should complete the figure autonomously.

The evaluation shows overall good results. Especially the circular shape in the topmost row is rapidly recognized, which can be seen by the first first anticipation of the system in the second frame. The other shapes were recognized in the fourth frame, because the lines in the previous three frames are too ambiguous. After getting a hint in the fourth frame through the user’s action, the system detects the rectangular respective triangular shape. Also the autonomous completion after the sixth frame works well in three of four shown trials. Only the rectangle in the last trial is not completed as expected. This may have the cause that the training pattern from Fig. 4 contains a shape with an edge length of two blocks, which corresponds to the anticipation of our system. Although the used learning technique generalizes over different sizes and rotations of the presented shapes, which can also be seen by the figure created in the second sequence, which is not present in the training image, the presented training shapes may have not been variant enough to fully exploit the generalization ability of the learning approach.

4.2 Artificial Data

To gain a quantitative error margin for the proposed approach, we evaluate the process with artificially generated shapes. To this end, a shape with varying size and rotation is generated and stored as ground truth data for later comparison. We will denote this complete shape as C. From this shape C, features are randomly removed to create a sparse shape, hereafter named S. Our approach is then used to close these gaps by generating completing features. The set of these features will further be denoted as H. The used error margin (9) is the mean mutual distance of generated features from H to their closest corresponding feature in the original set C. An exemplary overview of this procedure is shown in Fig. 6. The first image shows the complete shape which is used as reference. In the second image a number of features is removed to create shape C. The result of the "intention" completion can be seen in Fig. 6(c). An overlay of the "intended" completions and the original shape is illustrated in Fig. 6(d). From the features of Fig. 6(d) the error is calculated as:

$$E(C, H) = \frac{1}{|H|} \sum_{h \in H} \min_{c \in C} ||d(c, h)|| \tag{9}$$

We generated four different kinds of shapes, namely a triangle, a rectangle, a pentagon and a circle with varying sizes and orientations. From these shapes up to 40% of the feature were removed randomly in steps of 10%. For each shape and percentage of removed feature, 50 shapes are randomly generated. The generated shapes have an average of 2.8 distance between two features. In table 1 the average mutual distance over the 50 trials per shape and removed percentage is shown. There are no significant outliers in the data and given the distance

Table 1. Distance between "intended" completions and original shape.

Shape	**percent removed**				**Shape**	**percent removed**			
	10%	**20%**	**30%**	**40%**		**10%**	**20%**	**30%**	**40%**
Triangle	4.77	4.99	4.64	4.62	**Rectangle**	3.39	4.71	4.83	4.70
Pentagon	3.18	4.65	3.95	4.84	**Circle**	2.93	3.59	3.88	3.75

between the generated features of 2.8, the mutual distance is small enough to say that our approach can create reasonable amendments in the absence of data. Since the mutual distance in case of the circle is overall smaller than the mutual distances of the other shapes, this suggests that our approach slightly prefers smooth continuations to corners.

5 Conclusion

We presented an approach which exploits the perceptual grouping capabilities of the Competitive Layer Model and introduces a technique which extends these grouping capabilities towards the automatic generation of new data. The approach shows reasonable error margins in the evaluation with artificial data, which indicates good completion abilities. It is further evaluated in an interaction scenario which utilizes the generative aspect of this technique to anticipate the intentions of the human partner. Although the scenario is rather simple, the results are convincing that this extension of the CLM capabilities can be useful in a broader domain and more complex scenarios, because the only part which has to be supplied by a user is a compatibility function in the feature domain. This property also eases the adaption of the presented approach in comparison to more specialized data generation techniques like, for example, [2,8]. Given the findings from [10] and the presented compatibility measurement for actions, we strive to extend the CLM based segmentation of actions towards anticipating good completing actions for cooperative human-robot tasks.

References

1. Awais, M., Henrich, D.: Proactive premature intention estimation for intuitive human-robot collaboration. In: IROS 2012. pp. 4098–4103. IEEE (2012)
2. Eslami, S.A., Heess, N., Winn, J.: The shape boltzmann machine: a strong model of object shape. In: CVPR. pp. 406–413. IEEE (2012)
3. Gehrig, D., Krauthausen, P., Rybok, L., Kuehne, H., Hanebeck, U., Schultz, T., Stiefelhagen, R.: Combined intention, activity, and motion recognition for a humanoid household robot. In: IROS, 2011. pp. 4819–4825. IEEE (2011)
4. Jaillet, L., Cortes, J., Siméon, T.: Sampling-based path planning on configuration-space costmaps. Robotics, IEEE Transactions on 26(4), 635–646 (2010)
5. Kelley, R., Nicolescu, M., Tavakkoli, A., King, C., Bebis, G.: Understanding human intentions via hidden markov models in autonomous mobile robots. In: HRI, 2008. pp. 367–374. IEEE (2008)
6. Köhler, W.: A source book of Gestalt psychology. Kegan Paul, Trench, Trubner & Company (1938)
7. Meier, M., Haschke, R., Ritter, H.: Hallucinating image features to supplement perceptual groups. In: Workshop New Challenges in Neural Computation (2011)
8. Ming, Y., Li, H., He, X.: Connected contours: A new contour completion model that respects the closure effect. In: CVPR. pp. 829–836. IEEE (2012)
9. Nattkemper, T., Wersing, H., Schubert, W., Ritter, H.: A neural network architecture for automatic segmentation of fluorescence micrographs. Neurocomputing 48(1-4), 357–367 (2002)
10. Pardowitz, M., Haschke, R., Steil, J., Ritter, H.: Gestalt-based action segmentation for robot task learning. In: Humanoids 2008. pp. 347–352. IEEE (2008)
11. Ritter, H.: A spatial approach to feature linking. In: Int. Neural Network Conference, Paris (1990)
12. Weng, S.: Data driven learning for feature binding and perceptual grouping with the Competitive Layer Model. Bielefeld University (2005)
13. Weng, S., Wersing, H., Steil, J., Ritter, H.: Learning lateral interactions for feature binding and sensory segmentation from prototypic basis interactions. Neural Networks, IEEE Transactions on 17(4), 843–862 (2006)
14. Wersing, H., Steil, J., Ritter, H.: A competitive-layer model for feature binding and sensory segmentation. Neural Computation 13(2), 357–387 (2001)