Where to Look Next? Proto-object Based Priority in a TVA-based Model of Visual Attention

Thesis submitted for obtaining the academic degree

Doctor of natural sciences (Dr. rer. nat.)

Marco Wischnewski

submitted to
the Faculty of Technology, Bielefeld University, on October 18th, 2011.

Contents

Chapter 1

Introduction

1.1 Scope and contributions of the thesis

"Where to look next ?" is a central function of visual saliency computations and attention selection. The difficulty lies in *capacity limitations* of the primate visual system in terms of object recognition and visuo-motor control (Schneider 1995) - limitations that call for *selective mechanisms* able to prioritize chunks of the fixated scene, possibly containing the best candidates for further processing. As human and non-human primates as well as artificial systems share this problem of limited resources, attention modeling has become essential to explain data of visual search and object recognition (Treisman and Gelade 1980; Bundesen 1990; Schneider 1995; Wolfe and Horowitz 2004; Torralba, Oliva, Castelhano, and Henderson 2006) as well as for the synthesis of computer vision or robotic gaze orienting systems (Driscoll, Peters, and Cave 1998; Breazeal and Scassellati 1999; Steil, Heidemann, Jockusch, Rae, Jungclaus, and Ritter 2001; Nagai, Hosoda, Morita, and Asada 2003; Ruesch, Lopes, Bernardino, Hornstein, Santos-Victor, and Pfeifer 2008). Many of the artificial systems have been thereby inspired by the human attentional system and tried to replicate a similar function at different degrees of biological and psychological plausibility (Frintrop, Rome, and Christensen 2010). However, none of the existing models come even close to the apparent ease with which humans integrate bottom-up and top-down control of selective processing, e.g., for efficient visual search informed by task and context. There are three basic kinds of factors determining the outcome of attentional processing that are heavily investigated in human and non-human primate vision and, consequently, are also subject to computational modeling: *bottom-up visual feature maps*,

visual proto-objects, and *top-down task-based control*.

The modeling of *bottom-up* feature processing is inspired by the architecture of the ventral pathway (Rousselet, Thorpe, and Fabre-Thorpe 2004; Nassi and Callaway 2009; Freeman and Ziemba 2011; Kravitz, Saleem, Baker, and Mishkin 2011) and usually leads to the production of a saliency map from the weighted combination of different feature maps, reflecting the retinotopic structure of the input and considering single dimension conspicuousness at each location (Treisman and Gelade 1980; Koch and Ullman 1985). Established basic visual low-level features like intensity, color, and orientation (Wolfe and Horowitz 2004) are known to play a relevant role in different aspects of visual processing that refer to segmentation, visual search, and figure-ground discrimination (Nothdurft 1993). Many respective computational models for low-level features have been developed in this direction (Itti, Koch, and Niebur 1998; Driscoll, Peters, and Cave 1998; Breazeal and Scassellati 1999; Steil, Heidemann, Jockusch, Rae, Jungclaus, and Ritter 2001; Itti and Koch 2001) and have aimed at reproducing selected facets of human and non-human primate feature processing to some extent, see (Frintrop, Rome, and Christensen 2010) for a review. The representation as stack of basic feature maps has been refined and improved during the last years (Ruesch, Lopes, Bernardino, Hornstein, Santos-Victor, and Pfeifer 2008; Nagai 2009; Park, Shin, and Lee 2010; Walther, Itti, Riesenhuber, Poggio, and Koch 2010). Nevertheless, the account of attentional selection, when it comes to computational modeling, is mostly pixel-wise and bases on low-level features.

Attentional selection in everyday tasks is often *object-based*: We look for something. We want to grasp or manipulate an object, or to navigate an environment while avoiding obstacles. This object-based account of attention has been recently substantiated by growing experimental evidence from highly controlled laboratory studies (Scholl 2001; Bundesen and Habekost 2008; Naber, Carlson, and Einhäuser 2011), and it has also been picked up in some recent object-based computational approaches, such as (Walther and Koch 2006; Orabona, Metta, and Sandini 2008; Sun, Fisher, Wang, and Gomes 2008). They share the idea to bind regions on the feature-map level to proto-objects based on color/edge based segmentation or extraction of coherent regions in one feature channel, respectively, and partially refer to Gestalt ideas for segmentation of such regions (Orabona, Metta, and Sandini 2008). However, none of these approaches use proto-object based mid-level features for further processing. The latter is realized in the saliency model of (Aziz and Mertsching 2008), where regions have mid-level features such as size, symmetry, orientation and eccentricity. Beyond this, the predecessor of this model (Wischnewski, Belardinelli, Schneider, and Steil 2010) already integrates proto-objects in an

overall computational architecture for top-down control of attention.

Based on growing evidence for *task-dependent control* of covert and overt attention (Deubel and Schneider 1996; Bundesen and Habekost 2008; Land and Tatler 2009; Land 2009), computational bottom-up models have been extended, mostly by changing the weighting of features (Steil, Heidemann, Jockusch, Rae, Jungclaus, and Ritter 2001; Moren, Ude, Koene, and Cheng 2008) or by ex-post modification of the saliency map (Navalpakkam and Itti 2006; Navalpakkam and Itti 2010). Tsotsos' Selective Tuning Model (Tsotsos, Culhane, Winky, Lai, Davis, and Nuflo 1995) also implements a connectionist form of top-down biasing by enhancing target features and inhibiting distractor features. However, this kind of weighting can account only for simple preferences of basic visual feature channels over others ("look for red!") and fails to model more complex tasks. On the other hand, within the psychological literature there is the well established Theory of Visual Attention (TVA, (Bundesen 1990; Bundesen, Habekost, and Kyllingsbæk 2005; Bundesen and Habekost 2008; Bundesen, Habekost, and Kyllingsbæk 2011)) that is capable of explaining a large range of behavioral and neurophysiological data on covert visual attention by means of a relatively simple mathematical model. TVA provides a psychologically plausible and elegant way to combine top-down control of priorities for certain features or categories and bottom-up computed visual information. Importantly, TVA assumes that visual units or *proto-objects* have already been formed when attentional control is computed. In other words, TVA implies an object-based account of visual attention and therewith implicitly includes feature processing on the level of mid-level features. Surprisingly, except for (Wischnewski, Belardinelli, Schneider, and Steil 2010), TVA has not yet been included in any computational attention model, nor has it been subjected to stand-alone computational modeling.

An essential property of primate physiology is the *inhomogeneous* distributed density of photoreceptors and receptive fields in visual processing (Vincent, Troscianko, and Gilchrist 2007; Weber and Triesch 2009). The density is highest at the center of the visual field, called fovea, and decreases with increasing distance. This not only applies to retinal processing but also for subsequent processing stages like V1 (Watson 1983). Without spatial inhomogeneity there would be no need for task-based overt attentional shifts, e.g. saccadic eye movements, within the visual field because no additional information would be gained. Some models follow this inhomogeneity approach (Orabona, Metta, and Sandini 2008; Sun, Fisher, Wang, and Gomes 2008; Wischnewski, Belardinelli, Schneider, and Steil 2010) by blurring the input image using different techniques, e.g. log-polar (Sandini and Tagliasco 1980), but none of them use a spatial

inhomogeneous pixel grid for further processing, e.g. for object segmentation.

Finally, object- and task-based models have to find a way for a suitable task definition in order to search for one (or more) *learned* natural objects. So far, only the model of (Wischnewski, Belardinelli, Schneider, and Steil 2010) realizes a TVA compliant task definition based on mid-level features: For each feature dimension, a Gaussian describes the feature value of the target object (mean) and the accuracy of search (variance). But as this model does not include any learning stage, feature values have to be set by hand. Other object-based models waive segmentation and define target templates based on a set of low-level features (Zelinsky 2008; Hwang, Higgins, and Pomplun 2009; Elazary and Itti 2010). None of the existing models implements a learning of multiple proto-object representations which is appropriate for two reasons. First, when modeling inhomogeneous processing, mid-level feature representations of natural objects differ with regard to the object's position within the visual field. Empirical findings show that humans learn to associate these representations (Cox, Meier, Oertelt, and DiCarlo 2005). Second, because proto-objects are only coarsely segmented representations of objects, some objects are mapped by more than one proto-object where each can have strong different feature values, e.g. if an object is segmented in a small red and a big blue region.

In this thesis, a computational model of visual attention is presented that is centered around *proto-objects* to integrate all discussed factors of priority control: bottom-up low-level features organized in a spatial inhomogeneous feature map; mid-level feature computation to gain proto-object features like color, size, orientation etc.; and task-dependent priority computation through TVA basing on learned proto-object representations. The computation of proto-objects is the key step in this respect: Proto-objects represent discrete units of attention labeled by the mid-level features computed within their boundaries and by their position and extension in the field of view, and they provide the input for the TVA stage. The model also uses these mid-level features to learn proto-object representations of natural objects. By implementing a classification approach, it can assign different mid-level representations to one natural object, so tasks can be defined on the level of "search for the coffee mug" instead of being defined by specified (sets of) feature values that describe the target object. Having defined the task by choosing a target object according to the weight equation of TVA (Bundesen 1990), an attentional weight (attentional priority) is computed for each proto-object and stored in an retinotopic attentional priority map. These weights depend on bottom-up influences, such as the sensory evidence for visual features, and on top-down influences, such as the current task, and determine the degree of priority in perceptual processing. In the model, the assumption is added that these weights

also determine where-to-look-next (Wischnewski, Belardinelli, Schneider, and Steil 2010; Carbone and Schneider 2010): The proto-object with the highest attentional weight receives the highest priority in perceptual processing and, therewith, most likely becomes the target for the next saccade or camera shift (Schneider 1995). The combination of mid-level features and object classification makes it possible to realize both a task-based search for natural objects and an improved computation of the saccadic landing position, as saccades likely land close to an object's center (Foulsham and Underwood 2009; Nuthmann and Henderson 2010) and not on a "salient pixel".

1.2 Thesis outline

At first , in Ch. 2, an overview is given regarding the aspects of cognitive neuroscience of visual attention and eye-movement control which are relevant to the computational model presented in this thesis. In addition to the introduction of important psychological terms, it is explained why the model focuses on certain mechanisms that guide visual attention and why TVA builds the core of the model. Afterwards, in Ch. 3 it is shown whether and how these mechanisms are realized in established computational models in comparison to this model. At the end of the chapter, the model's global flow diagram is presented which includes all these mechanisms in a coherent architecture (Fig. 3.1).

Starting with Ch. 4 up to Ch. 9, both model's processing streams are described in detail. The first stream (green, see Fig. 1.1), is used to learn proto-object representations of natural objects. For this, object examples pass the first three stages: building an inhomogeneous low-level feature map (Ch. 4), segmentation of proto-objects (Ch. 5), and computing the proto-objects' mid-level features (Ch. 6). The resulting feature values are implicitly stored in the weights of a classification network (Ch. 7). The second stream (red) realizes a complete saccadic cycle. First, an input image also passes the first three stages to compute its proto-objects. Then, based on a given search task ("search for object x") and the learned object representations, the TVA weight equation is used to assign an attentional weight to each proto-object of the input image (Ch. 8). Finally, in Ch. 9, the saccade's target object is selected by a winner-takes-all mechanism considering the positions of the last fixations (inhibition of return). After a saccade has been executed, the model can compute the next one by starting again with building an inhomogeneous low-level feature map.

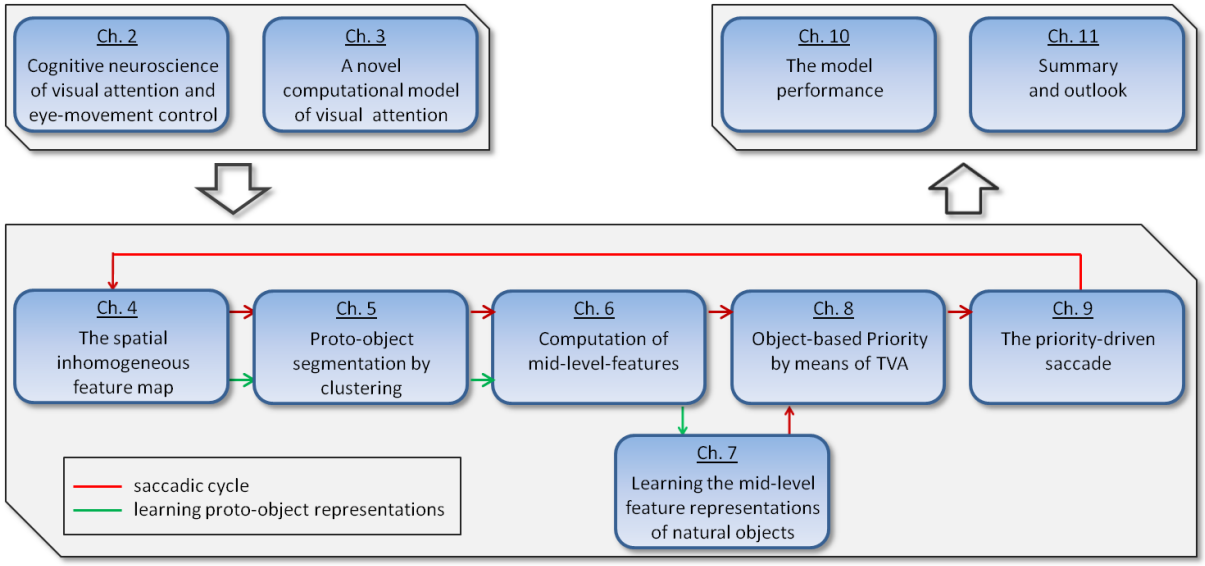

Figure 1.1: *Thesis outline, see Sec. 1.2 for details.*

The penultimate Ch. 10 explains how the performance of the model can be TVA-like adjusted to satisfy the proto-object approach. Such an adjustment for a given set of objects is illustrated by an example. In Ch. 11, after a concise summary, useful model extensions are presented and discussed.

There are two appendixes: The first gives an overview of the notation used in the thesis (App. A). The second shows an image library (COIL), consisting of 100 different natural objects, which is used in most examples (App. B).

Chapter 2

Cognitive neuroscience of visual attention and eye-movement control

2.1 Introduction

In order to build a computational model, first, the subject of modeling has to be limited. On this account, in this chapter an overview is given about those aspects of visual attention and eye-movement control which are relevant for this thesis' model. Additionally, different psychological terms are explained and it is argued why the model focuses on a certain kind of visual attention, selected mechanisms that guide visual attention, and one special psychological attention model.

2.2 Spatial inhomogeneous visual processing

As neuronal resources are basically limited, e.g. the number of receptors/neurons as well as the number of connections between them, cognitive processes like vision underlie a *capacity limitation* (Bundesen 1990; Schneider 1995; Bundesen, Habekost, and Kyllingsbæk 2005). Accordingly, this limitation also affects the *spatial resolution* of the visual processing path, that is, the spatial resolution that is available for sampling (via photoreceptors) and processing (e.g. via receptive fields) of the visual environment. A measure for spatial resolution is the density of receptors/receptive fields as well as size and spatial frequency of receptive fields.

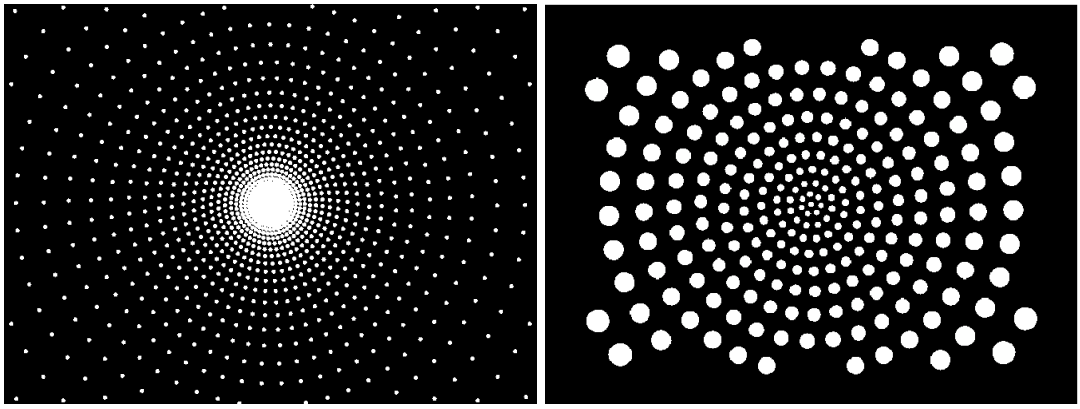

Figure 2.1: *Retinal density of photoreceptors depending on eccentricity (left). The density is highest at the visual center, called fovea, and decreases with increasing eccentricity. In receptive fields of subsequent processing stages, additionally, size concurrently increases (right).*

An increase of resolution corresponds to (a) an increase of density and frequency and (b) a decrease of size.

It is a characteristic of the primate visual system that, at least in the early processing stages, the spatial resolution differs depending on the position within the visual field (Watson 1983; Van Essen and Anderson 1995; Weber and Triesch 2009). In these stages the spatial resolution is highest at the center of the visual field and decreases with increasing angle of eccentricity (see Fig. 2.1). This means that visual processing underlies *spatial inhomogeneity*.

As a result, resolution-dependent performances like object recognition are highest at the center of the visual field, the so-called fovea. So, e.g., Carrasco et al. have shown that in a target detection task (*conjunction search*, see (Treisman, Sykes, and Gelade 1977) for details) the *search efficiency*, measured by the number of errors and reaction time, decreases the more peripherally a target is located, which they called the *eccentricity effect* (Carrasco, Evert, Chang, and Katz 1995). An additional effect was found regarding an object identification task: Participants had to identify letters at different eccentricities, see e.g. (Klein, Berry, Briand, D'Entremont, and Farmer 1990). Again, the performance, that is the percentage of correct identifications, decreased with increasing angle of eccentricity. Furthermore, if an eye-movement has to be made under time pressure to one of two nearby peripheral objects, it tends to land in-between these objects (center of gravity), which is called the *global effect* (Findlay 1982; Vitu 2008). Although, for certain tasks the rare case arises that even the foveal performance collapses. An example is the so-called c*entral performance drop*, see e.g. (Kehrer 1989).

In sum, in general the performance of the visual systems in primates decreases with increasing angle of eccentricity. The reason for this is the peripheral decrease of spatial resolution.

2.3 Overt attention

The spatial inhomogeneity in primate visual processing can be described as a trade-off. On one hand, the concentration of resources in the foveal region allows, by comparison with a spatial homogeneous processing, a region-limited, sophisticated mapping of the environment. On the other hand, peripheral mapping is much less accurate. The latter is a problematic point because regions or objects in the periphery can be highly relevant, e.g. a dangerous animal, but are more difficult to identify. In primates, the 'evolutionary solution' is to quickly bring a potentially important object to the foveal region. This can be realized by a *saccadic eye movement* or by moving the head or other parts of the body, which is, however, significantly slower. This is called *overt attention*: Action-relevant regions or objects are brought to the fovea to gain maximal visual processing resources. Additionally, attention can be shifted covertly, without any physical movement (Findlay and Gilchrist 2003). Moreover, an obligatory coupling exists between covert attention and saccades (Deubel and Schneider 1996) depending on task difficulty (McPeek, Maljkovic, and Nakayama 1999). This thesis focuses solely on *overt attention by eye movements*.

The notion that visual attention guides visual perception, depending on action-relevance, is called *selection-for-action* (Allport 1987; Neumann 1987; Schneider 1995; Humphreys, Yoon, Kumar, Lestou, Kitadono, Roberts, and Riddoch 2010). Accordingly, a main requirement of the visual system is to implement an appropriate frequency of eye movements that ensures a sufficiently fast shift of overt attention to focus on task-relevant objects but also leaves enough time for cognitive processes, like object classification, in order to assess the task-relevance of foveally as well as peripherally located objects. In humans, the frequency of saccadic eye movements in general lies between 3 and 4 movements per second. The duration of a typical saccade is around 30 ms and is followed by a fixation of 300 ms. During a saccade, the visual system is 'blind' to visual input (Henderson 2007). There are different types of saccades (Hutton 2008), and one has to additionallydistinguish between saccades and pursuit (Krauzlis and Chukoskie 2009). This thesis concentrates entirely on so-called *prosaccades* where a visual system directly saccades to a given search target.

2.4 Saliency vs Priority

If visual attention guides visual perception, e.g. by means of eye movements, then what guides visual attention? One aspect of this question lies in the distinction between saliency and priority. *Saliency-driven attention*, also called stimulus-driven or exogenous or bottom-up controlled attention, signifies that attention is guided by local contrasts in the visual input stream, see e.g. (Treisman and Gelade 1980; Koch and Ullman 1985; Itti and Koch 2000; Henderson 2003; Lingyun, Tong, and Cottrell 2007; Elazary and Itti 2008). Corresponding models claim that the strength of these local contrasts is entered in a retinotopic *saliency map* (Koch and Ullman 1985; Itti and Koch 2000; Elazary and Itti 2008). Then, the most salient region, the region where contrast is highest, is likely to be the next saccade target. On the other hand, *priority-driven attention*, also called task-driven or endogenous or top-down controlled attention, means that attention is guided by task-relevant features or objects, see e.g. (Wolfe 1994; Navalpakkam and Itti 2005; Rothkopf, Ballard, and Hayhoe 2007; Mozer and Baldwin 2008; Bundesen and Habekost 2008; Cutsuridis 2009; Land 2009), and global scene gist (Torralba, Oliva, Castelhano, and Henderson 2006). Analogous to salient attention, the features' or objects' strength of task-relevance are entered in a retinotopic *priority map* (Bundesen 1990; Bundesen, Habekost, and Kyllingsbæk 2005). Then the highest-priority location in the map is likely to be the next saccade target (Carbone and Schneider 2010; Wischnewski, Belardinelli, Schneider, and Steil 2010).

There are only a few cases where an explicit task plays no role in shifting attention, e.g. in the *pop-out effect* (Treisman and Gelade 1980) or free-viewing of natural scenes. Another case is the task-based viewing of natural scenes: Saliency models are also able to reliably predict fixation points in a task-based scene viewing, especially for the first saccades. But it was shown that this is possibly only a correlation caused by scene knowledge (Foulsham and Underwood 2011). Although the influence of saliency cannot always be completely suppressed (Theeuwes 2004), empirical findings have shown that, if priority exists, then it generally dominates over saliency (Zelinsky, Zhang, Yu, Chen, and Samaras 2006; Peters and Itti 2007; Henderson, Brockmole, Castelhano, and Mack 2007; Einhäuser and Perona 2008; Ballard and Hayhoe 2009). So, at least for natural scenes, saliency hardly influences visual attention. On this account, this thesis only focuses on *priority-driven attention*.

2.5 Low-level features vs object-based features

Another aspect regarding the question "what guides visual attention?" concerns the distinction between feature-based and object-based attention. *Low-level feature-based attention* is based on the idea that local features or local feature contrasts(e.g. for color, orientation, intensity, or motion) guide attention, see e.g. (Treisman and Gelade 1980; Itti and Koch 2000; Wolfe and Horowitz 2004). So, the search for something red can be realized by prioritizing red regions, and the pop-out effect can be explained by the strength of local contrasts in one or more feature dimensions. On the other hand, in the case of *object-based attention*, attention is guided by features which describe properties of objects, e.g. size, color, or orientation (Scholl 2001; Bundesen and Habekost 2008). So object-based attention is also based on features, but these features are more complex: They are built on the basis of an object's low-level features. Depending on the level of complexity, object-based features can be called *mid-level features* (Wischnewski, Belardinelli, Schneider, and Steil 2010) or even *high-level features* (e.g. used for object recognition). It was shown that, in general, visual attention is guided by objects, see e.g. (Scholl 2001; Rothkopf, Ballard, and Hayhoe 2007; Zelinsky 2008; Bundesen and Habekost 2008; Land 2009; Nuthmann and Henderson 2010; Naber, Carlson, and Einhäuser 2011). Accordingly, this thesis concentrates on *object-based attention*.

2.6 From low-level features to proto-objects

The high frequency of saccades in primates leads to a corresponding limitation in time to identify the perceived objects (see Fig. 2.2). The less time is available, the less accurate is object segmentation (Findlay 1982) and the lower is the feature complexity of object representation (Rousselet, Thorpe, and Fabre-Thorpe 2004; Nassi and Callaway 2009). The result is that it is impossible for the visual system to perform object recognition within one saccadic cycle. Instead, the primates' visual system has to deal with a lower level of object representation regarding overt attention. These *preattentive object representations* are the result of *mid-level vision* (H.S. Scholte 2009), and their features are accordingly called *mid-level features* (Wischnewski, Belardinelli, Schneider, and Steil 2010) or even *moderately complex features* (Tanaka 2003). There are many names for mid-level feature object representations, e.g. *proto-objects* (Rensink 2000), *object files* (Kahneman, Treisman, and Gibbs 1992), *preconceptual objects* (Pylyshyn 2001), or *perceptual units* (Bundesen and Habekost 2008). In this thesis, the term *proto-object*

is used.

In mid-level vision, a proto-object is defined as (1) a spatial region coming from surface segmentation (H.S. Scholte 2009) and as (2) a set of mid-level features which describe the region's properties, like mean color, size, etc. (Nassi and Callaway 2009; Wischnewski, Belardinelli, Schneider, and Steil 2010). Thus, at first the segmentation has to be performed. Then the low-level features within a proto-object's region, e.g. local color or orientation features, are used to compute its mid-level features, e.g. mean color or texture. After this computation, the visual system has to *bind* the features (Humphreys and Riddoch 2006) in order to build a feature set which is then assigned to the associated region. Although not all features are equally able to guide attention (Wolfe and Horowitz 2004).

In sum, the set of formed proto-objects is a quickly built coarse object representation of the visual environment. Proto-objects are more than an "object-less" low-level feature representations, but they do not reach the level of object recognition. This imprecision of representation implies that proto-objects have a "candidate" status: The visual system does not have an absolute certainty about whether a proto-object really represents the natural object the system is searching for, but, on the level of proto-objects, it can distinguish between more or less appropriate candidates regarding a given task.

2.7 Visual search: where to look next?

Assuming the visual system has already formed proto-objects from the input stream and the task is to search for object o, which proto-object will then be the next saccade's target? For a proto-object, the probability of being the next saccade target increases as thesimilarity between it and o increases. This is called the *similarity effect* (Findlay and Gilchrist 2003). As proto-objects are represented by mid-level features, the similarity comparison between a proto-object and o has to be realized on that level. This implies that the visual system has to have access to a mid-level feature representation of object o. There is one psychological model which provides such a similarity computation based on mid-level features: the "*Theory of Visual Attention*" (TVA) (Bundesen 1990).

In TVA each proto-object obtains an *attentional weight* depending on feature similarity which is called *filtering*. For each feature dimension, the similarity is separately computed by the η-function. In terms of TVA, feature similarity is called *sensory evidence*. Furthermore, each

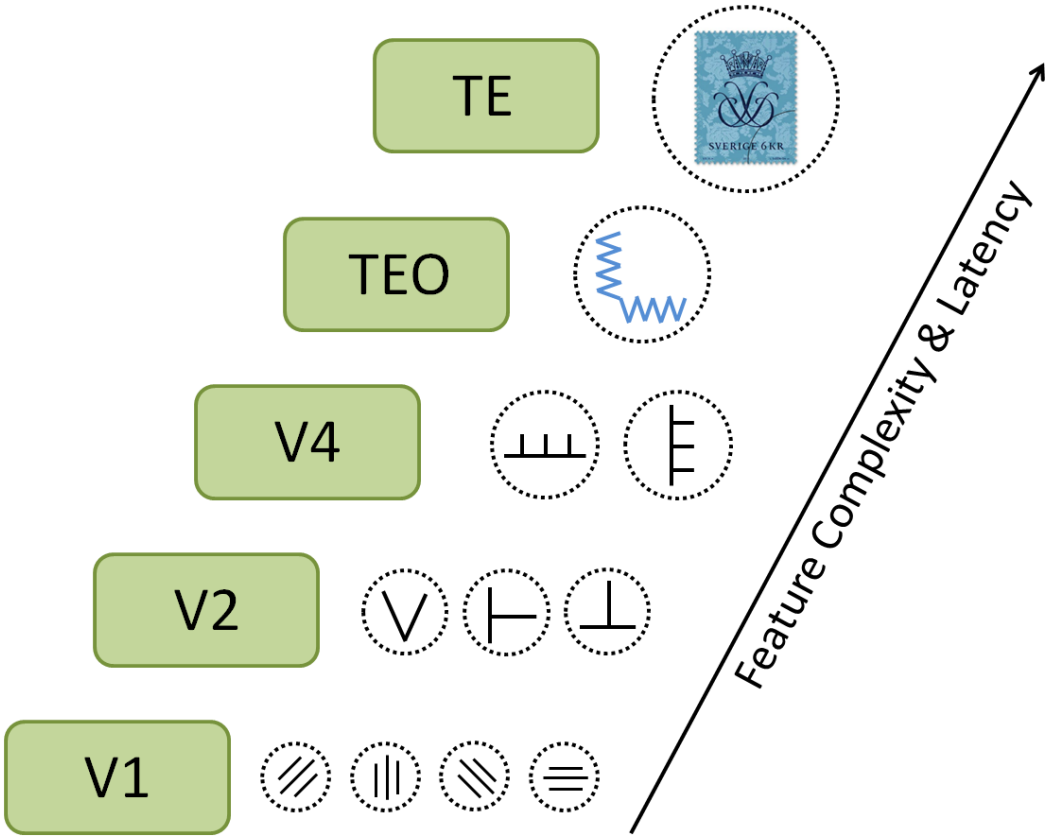

Figure 2.2: *Schematic illustration of the ventral pathway according to (Rousselet, Thorpe, and Fabre-Thorpe 2004). Here processing is illustrated from the level of the primary visual cortex (V1) up to the inferio-temporal cortex (TE). For each level, one or more feature examples are shown which the associated receptive fields (RFs) are able to detect, e.g. RFs for orientation in V1 or RFs for simple geometric shapes in V2 etc. Each processing level uses the outputs of the RFs of the preceding levels, thus, with each additional processing level both feature complexity and processing latency increases, too. Additionally, as indicated in the figure, also the size of the RFs increases.*

feature dimension can be weighted by pertinence π. These pertinence values are defined by task. Then for each proto-object the attentional weight is built by the sum of all pertinence weighted η-values. So, e.g. if a system searches for a blue object, pertinence for the feature dimension "blue" is set high and pertinence for all other dimensions is set to zero. The result would be that only those proto-objects that have feature "blue" would obtain a high attentional weight. In general terms: The higher the feature similarity for dimensions with $\pi > 0$, the higher the attentional weight.

In TVA all stages up to the computation of attentional weights work in *parallel,* which is in compliance with empirical findings, see e.g. (Rousselet, Thorpe, and Fabre-Thorpe 2004; Nassi and Callaway 2009). Moreover, TVA claims that, due to the visual system's capacity limitation, the sum of all proto-objects' weights is constant. This means weights are absolute and not relative. So, the computation of attentional weights can be interpreted as a task-based competition for neuronal resources. A similar approach is followed by the "bias competition" model, which additionally includes saliency (DeSimone and Duncan 1995). Finally, in TVA the attentional weights are stored in a retinotopic *priority map*. There is empirical evidence that in humans the strength of these weights correlates with saccadic latencies (Carbone and Schneider 2010). The higher the weight, the shorter the latency. Based on this finding, in this thesis it is assumed that these attentional weights also determine the target position of the next saccade: The higher the weight, the higher the probability of being the next saccade target. Following this assumption, the question "where to look next?" can be answered: Most likely to the location of that proto-object which achieved the highest attentional weight.

There are two more effects that also strongly influence saccadic behavior. The first one is called *proximity effect* (Findlay and Gilchrist 2003): More foveally located objects are more likely to be the next saccade's target. In terms of TVA, this would corresponds to a general peripheral decrease of attentional weights independent of feature similarity and pertinence. Such an eccentricity-dependent weight modification is not yet part of the original TVA, although it was already added by the predecessors of the model presented in this thesis (Wischnewski, Steil, Kehrer, and Schneider 2009; Wischnewski, Belardinelli, Schneider, and Steil 2010). The second effect concerns the problem of creating an efficient search strategy. If visual search produces a series of saccades, a so-called *scanpath*, then it is important to avoid *saccadic oscillations*. Saccadic oscillations would keep attention imprisoned to a few highly prioritized locations in space without the chance to escape, even if none of the perceived objects were identified as search targets. Moreover, if the currently fixated object becomes the winning ob-

ject again, that is, it serves as the next saccade target, then the system produces a *saccadic standstill*. The solution, in terms of TVA, is to attenuate the attentional weight of the last fixated objects. Furthermore, the attenuation of all attenuated objects itself decreases within each saccadic cycle. The result is that the last fixated object is the most attenuated. The neuropsychological mechanism that realizes such a suppression of saccadic oscillation is called *inhibition of return* (IOR) (Klein 2000). IOR is a component of many models, e.g. (Zelinsky 2008), but not of TVA.

Even if it is fixed which proto-object serves as next saccade target, the question of the exact target position within the visual field still remains. Findings have shown that humans tend to saccade on the center of objects, similar to a Gaussian distribution (Foulsham and Underwood 2009; Nuthmann and Henderson 2010). If a saccade is larger than a 20° angle of sight, the human visual system produces an *undershoot* of about 10% followed by a very fast correction saccade. It is assumed that both saccades are part of a single executed gaze shift (Land and Tatler 2009).

This thesis' model is based upon TVA, because, although incomplete in some fields, it is the most suitable psychological framework for a computational model of visual attention regarding task- and object-based eye-movement control.

2.8 Summary

In this chapter, the role of attention in eye movements, which is called *overt attention*, was explained: Since cognitive systems are capacity-limited, processing resources have to be focused on task-relevant information. In vision, the visual system has to decide which object in the visual field is most likely relevant with respect to a given task, e.g. taking a cup. The peripheral, low spatial resolution (spatial limitation) as well as the high frequency of saccadic eye movements (temporal limitation) makes it impossible to realize object recognition for all objects within one saccadic cycle. According to this, the visual system computes coarse mid-level feature representations of a scene's objects, the so-call *proto-objects*. Mid-level features are, e.g. an object's size, mean color, etc... Then the visual system selects that proto-object which on the level of mid-level features best meets the target object. This implies that the visual system has already learned a mid-level feature representation of the object it searches for (target template). Afterwards, an eye movement brings the selected object into the fovea, the

location on the retina providing the highest spatial resolution. Therewith the maximum potential amount of resources is assigned to the selected object and it can be decided if this object is the object the system was searching for. But there are additional mechanisms that influence the choice of the saccadic targets: *Inhibition of return* avoids saccadic oscillations respectively standstillsdo not know what this means perhaps “avoids saccadic oscillations altogether” ?and the *proximity effect* shows that more foveally located objects are preferred.

It was shown that there are two essential distinctions with regard to the question ”what *does* guide visual attention ?”: object-based (e.g. size or mean color) vs. low-level feature based attention (local contrasts) and task-based (e.g. search for a certain object or low-level feature) vs. saliency-based attention (salient objects or low-level features). For this thesis’ model, it was argued to focus on *task-based and object-based control of visual attention* because this combination represents the common and usual case when primates overtly shift attention in natural scenes.

Finally, it was argued to chose *TVA*, the ”Theory of Visual Attention“ (Bundesen 1990), as the model’s foundation because there are convincing empirical findings, up to the neurophysiological level (Bundesen, Habekost, and Kyllingsbæk 2005), that support this theory, and because it provides a framework that is explicitly task- and object-based.

Chapter 3

A novel computational model of visual attention

3.1 Introduction

There are numerous computational models that aim to model visual attention. So far, no model has reached, even to some extent, the level of modeling every known detail, e.g. with regard to findings in the primate visual system. This is already quantitatively impossible. Moreover, often compromises have to be made in order to ensure a model's applicability, e.g. for real-time capability.

Accordingly, the model presented in this thesis also deals with these limitations and therefore focuses strictly on those aspects which are most relevant for modeling primates, such as visual attention, considering the aim to simulate the whole processing path from the sensory level (image perception) up the action level (saccadic eye-movement). Following this guideline, the model's core properties are derived from the psychological findings presented in Ch. 2. So, it includes spatial inhomogeneous processing (see Sec. 3.2), an object-based (see Sec. 3.3) and task-based control of attention, learned target templates, and TVA-based target selection (see Sec. 3.4).

In this chapter, these properties are explained from a computational viewpoint. It is described how this model implements them in comparison with other models. Furthermore, the global model structure is presented which embeds all these properties in a coherent way (see Sec. 3.5).

It is shown that in detail (e.g. object segmentation on spatial inhomogeneously distributed pixels), but even more from a global perspective, that is the integration of essential psychological findings in a novel coherent model structure, this model realizes a unique (neuro)psychological motivated approach of modeling visual attention.

3.2 Spatial inhomogeneous processing

As shown in the previous chapter, eye movements and eccentricity-dependent spatial resolution are inseparably linked to each other. In other words, computational models that have not implemented spatial inhomogeneous processing, see e.g. (Itti and Koch 2000; Navalpakkam and Itti 2005; Elazary and Itti 2010), lack the cause for overt attentional shifts, because the spatial resolution (and therefore the resources of the visual system) is equally distributed over the whole visual field. But even if this conceptual error is ignored, these models are not able to explain the dependency of segmentation and identification performance on eccentricity.

There are only a few models which have implemented spatial inhomogeneous processing. An early approach is called *log-polar* (Sandini and Tagliasco 1980) and is used e.g. in (Orabona, Metta, and Sandini 2008; Sun, Fisher, Wang, and Gomes 2008). The log-polar method transforms pixel positions from the Cartesian plane (x, y) to the log-polar plane (ρ, Θ) where ρ equals the distance to the point of origin, which equals the center of gaze, and Θ equals the associated angle. The number of pixels remains constant but, due to the logarithmic transformation, the density of pixels decreases with increasing eccentricity. A similar approach was used by (Vincent, Troscianko, and Gilchrist 2007). Here, however, the pixel density decreases as a *power law* and the pixel positions underlie *jitter*. Another model comes from (Geisler and Perry 2002), e.g. used in (Zelinsky 2008). Here the image is *blurred* by first building a multiresolution pyramid (Burt and Adelson 1983) with eccentricity-dependent low-pass filtering and subsequent interpolation of the pyramid's layers. For this thesis' attention model, another approach is made use of: the *Watson model* (Watson 1983), e.g. used in (Kehrer and Meinecke 2003; Wischnewski, Steil, Kehrer, and Schneider 2009; Wischnewski, Belardinelli, Schneider, and Steil 2010). The Watson model provides a biologically motivated model not only for the eccentricity-dependent spatial density of receptive fields but also for the eccentricity-dependent changes of the receptive fields' size and frequency: Density and frequency decreases whereas size increases with increasing angle of eccentricity. The scaling of density, frequency, and size equally depends on scaling factor s which *grows linearly* with increasing eccentricity.

In contrast to most attention models that make use of spatial inhomogeneous processing only on the retinal level, in this thesis' model, the whole processing path, up to the segmentation of proto-objects, is realized inhomogeneously [1]. Thus, e.g. the proximity effect (see previous chapter) can be made plausible: Peripherally located objects are represented by less pixels and neurons, respectively, and thereby obtain a relative lower attentional weight.

3.3 Proto-objects

Many attentional models are based on the saliency model of (Koch and Ullman 1985), e.g. (Itti and Koch 2000; Walther and Koch 2006). In these models, attentional weights, which correspond to the strength of local contrasts, are assigned to single pixels in the input image; that is, they work *pixel-based*. As a result, they do not provide proto-object representations, but there are nevertheless two ways object-based attention has been realized using the pixel-based approach. First, in (Walther and Koch 2006), the computation of the saliency map is followed by an object segmentation stage where only one proto-object is segmented around the most salient pixel. The result is a contiguous region of high saliency. Object-based saliency could then be computed by region-based averaging. By inhibiting the proto-object's region, the segmentation process can be repeated to obtain more than one proto-object. A weak point of the model is the serial segmentation of proto-objects, which does not match empirical findings that prove processing is parallel (Bundesen and Habekost 2008). Second, in (Zelinsky 2008) and (Elazary and Itti 2010), proto-objects are defined by a one-pixel low-level feature vector (see Sec. 3.4 for details). Then each pixel's feature vector of the feature map is compared to this feature vector. In Zelinsky's *target acquisition model* (TAM), the comparison is realized by computing the correlation between these vectors, which results in a *target map of visual similarity*. On the contrary, in Elazary and Itti's model, *SalBayes*, the comparison is realized by a Bayesian approach. For each feature in each feature map, the model computes the probability that this feature represents the target. The result is a set of *probability maps* which are then ummed up to a single saliency map (as task-based, priority map would be the more suitable term). Both TAM and SalBayes models do not provide a segmentation of proto-objects. So, they have not implemented object-based attention in the original sense regarding TVA (Bundesen 1990): where an attentional weight is assigned to each proto-object of the sensory input. Both models, on

[1] Download the model software at `http://www.uni-bielefeld.de/psychologie/ae/Ae01/IIP`

the contrary, do use a pixel-based object representation (low-level feature vector) to compute pixel-based attentional weights. Moreover, especially SalBayes is unable to detect an object's center and therewith not capable of saccading on this. This is because SalBayes uses the most salient pixel of an object to compute the target feature vector which can be located anywhere in the object. TAM always uses the central pixel to compute the target feature vector, which makes this approach more stable with regard to this problem.

Other models make use of a segmentation stage to obtain proto-object representations of natural objects, e.g. (Orabona, Metta, and Sandini 2008; Sun, Fisher, Wang, and Gomes 2008; Wischnewski, Belardinelli, Schneider, and Steil 2010). In Orabona et al.'s model, at first, an edge map is computed, and then a watershed transformation (Smet and Pires 2000) is used to fill all spaces between these edges. This results in homogeneous color regions. A similar approach was realized by Wischnewski et al.: Proto-objects are the result of a bottom-up and top-down process within a Gaussian pyramid to identically label pixels that belong to the same homogeneous color region, see also (Forssén 2004). Sun et al. use the *EDISON* [2] approach for segmentation (P. Meer 2001) which also uses a preceding edge detection stage. Additionally, they implemented a hierarchical grouping of proto-objects in conformity with (Haxhimusa and Kropatsch 2003).

In sum, pixel-based models are able to assign an attentional weight to each pixel, where that weight depends on a low-level feature vector representing the target object. So, they deliver a measure for local similarity regarding the target object, but they do not provide any information about non-target objects. On the other hand, object-based models yield an object-based scene representation. By segmentation, they provide information about which proto-objects are in the scene and, thereby, are able to compute a set of mid-level features for each of them: size, mean color, orientation etc. As shown in the previous chapter, the latter approach better suits empirical findings in visual attention.

Accordingly, the model of this thesis follows the object-based approach. Its proto-object segmentation is built on its predecessor (Wischnewski, Belardinelli, Schneider, and Steil 2010), but the segmentation stage has been extended by two new properties. First, the segmentation is computed using spatial inhomogeneous distributed pixels, which is unique so far: Although many object-based models also apply some kind of retinal transformation, also called *foveation*, they all map the blurred result on a standard image pixel grid before starting the subsequent segmentation. Second, a classification stage (see Sec. 3.4 for details) makes it possible to merge

[2] `http://coewww.rutgers.edu/riul/research/code/EDISON/index.html`

proto-objects that belong to the same natural object. If, e.g. a ball consists of a red and a blue half, then these proto-objects can be merged into a new one. This is very useful to model the property of humans to saccade on the center of objects, see e.g. (Foulsham and Underwood 2009). Most other models fail at this point: They could only shift attention to one of the halves.

3.4 Task-dependency by means of TVA and learning of object representations

When aiming at modeling task- and object-based visual attention, a target object representation (target template) is needed in some form or another. The existing solutions so far can be roughly divided into three groups.

The first solution is an implicit object representation as can be found in (Navalpakkam and Itti 2005). They use a standard pixel-based approach to compute a saliency map. Task-dependency is then realized by a weighing of feature maps. The better a feature map's feature, e.g. a certain orientation feature, fits the target, called *relevance of feature f*, the higher the map's weight. In a later model, they have shown that sometimes search results can be improved even by enhancing non-target features (Navalpakkam and Itti 2007).

The second solution is an explicit low-level feature object representation as used, e.g. in TAM (Zelinsky 2008), SalBayes (Elazary and Itti 2010), or the model of (Hwang, Higgins, and Pomplun 2009). In the TAM a 72-dimensional low-level feature vector is computed which serves as a target object representation. The dimensionality is obtained by combining two filter types (Gaussian derivative filters of first and second order), three color channels (red-green, blue-yellow, black-white), three spatial scales (7, 15, and 31 pixels), and four orientations ($0°$, $45°$, $90°$, and $135°$). The filters' center is always the target object's center. A big disadvantage of this approach is that the target object size is limited in both directions to a maximum spatial filter scale of 31 pixels: Significantly bigger or smaller objects are not accurately mapped by the target feature vector. Furthermore, due to the filters' centrality, non-central salient object regions, e.g. a big red blob at an object's top area, are barely mapped. SalBayes follows the same approach by computing a 42-dimensional low-level feature vector. Here, seven features are computed (the same three color and four orientation features as in TAM) for six spatial scales. One main difference regarding TAM is that the filters' center is chosen by a maximum of saliency: The pixel of the target object that returns the highest saliency value serves as the

center for the feature vector computation. Thus, this approach is more flexible but, on the other hand, suffers from the fact that local contrasts often do not represent the nature of an object. If, e.g. a big, red object has a barely detectable green area, then SalBayes would learn this object as a "red/green contrast area" and therefore searchs for such contrasts to find this object instead of going the more plausible way and searching for a big red one. In the model of Hwang et al., a set of eight low-level features (intensity, contrast, gradient, frequency, orientation, and three color values in the DKL color space (Shapiro 2008)) is computed for each pixel of a 64x64 pixel sized target and stored in eight-feature histograms. This representation allows a more general description of objects because features are not computed around only one central or most salient pixel, but it shares two essential disadvantages with TAM: Target objects have to have a predefined size in pixels and a square-like shape. The models try to compensate for the latter by cutting a target image representation out of the search image. So even pixels that do not belong to the object the system searches for, e.g. the pixels in the target image's corner if the target object is circle-like, are learned to be part of the target object. Therefore, the same target object, presented at another position in the input image, would be more difficult to find since background pixels around the target object are then different.

In both solutions, attentional weights are computed pixel-wise, either by summing weighted feature maps or by a pixel-wise low-level feature vector/histogram comparison. Then the saccade's target location can be the most prioritized pixel or, when using subsequent low-pass filtering or object segmentation, the center of the most prioritized region.

The third solution realizes the idea that attention is based on features of segmented proto-objects, see e.g. (Sun 2003; Aziz and Mertsching 2008; Wischnewski, Belardinelli, Schneider, and Steil 2010). Aziz and Mertsching's model computes an object-based saliency map with regard to five mid-level features: contrast in color, eccentricity, orientation, symmetry, and size. As it is saliency based, this approach is purely bottom-up, but could easily be extended by a weighting mechanism to implement top-down control. Sun introduces so-called *weighting coefficients* to realize top-down control on different levels of feature complexity, but provides no computational implementation. In Wischnewski et al.'s model, an object-based priority map is computed based on size, orientation of the main principle axis, shape (relation between both principle axes), mean color, mean intensity, mean orientation (texture), motion energy, and motion direction. These mid-level features are weighted by means of the TVA *weight equation* (Bundesen 1990). As in TAM and SalBayes, the target object is defined by a feature vector, but here consisting of *mid-level features*. Additionally, for each feature dimension, a variance

is specified to determine the search accuracy. The feature vector's values of the target and the corresponding variance values are used to define a Gaussian. Then, according to TVA, for each proto-object in the visual input, the similarity between the proto-object and the target object is computed. This similarity is called *sensory evidence* and is computed separately for each feature dimension using the associated Gaussian. Additionally, the importance of feature dimensions is controlled by top-down *pertinence*. A proto-object's attentional weight is then computed by summing up the pertinence-weighted sensory evidence over all feature dimensions. A weak point of the model is that target feature values, their variance, and their pertinence value have to be set by hand, so there exists no mechanism for learning a set of target templates.

Concerning task-dependency and learned object representations, the model presented in this thesis takes some ideas of the described models, especially from (Wischnewski, Belardinelli, Schneider, and Steil 2010), but it also implements new components to meet the given requirements. First, compared to the Wischnewski et al. 2010 model, the feature space was extended by new shape features, which makes it possible to distinguish objects based on local pixel distribution, e.g. "+" and "x". Furthermore, the motion features were removed as this part was completely developed by A. Belardinelli (Belardinelli, Schneider, and Steil 2010). The same applies to the local orientation features, as they are very expensive to compute, but contribute only little to the system's performance regarding natural objects. Second, and this is the most important point, the model now learns mid-level feature representations from examples. Moreover, the model is able to assign different mid-level feature representations to one natural object by using a neural network for classification. This is indispensable for two reasons: One natural object can consist of two or more different proto-objects (e.g. a small red and a big blue region) and the proto-object representations change depending on eccentricity. There is evidence that humans learn to associate foveal and peripheral representations of one object (Cox, Meier, Oertelt, and DiCarlo 2005). The classification approach makes it possible to search for more than one object in parallel, where each object has its own preference (search for any cup but preferably the black one). Third, a modified version of the TVA weight equation builds the core of the model. A new interpretation of the TVA features integrates the classification approach into TVA and thereby adds the capability of adjusting performance regarding the ability to distinguish targets (objects the system searches for) from distractors (non-relevant objects). This is called *target-distractor discriminability* or simply α (Bundesen and Habekost 2008).

3.5 The model's global architecture

In the following, the model's global structure, corresponding to the flow diagram shown in Fig. 3.1, is illustrated. First, on the sensory level, an image is provided by any source, e.g. a camera of a robot. Then the model computes an inhomogeneous feature map, which simulates a retinal transformation (Ch. 4). The resulting feature values for color and intensity serve as input for the segmentation stage in order to obtain the regions of proto-objects (Ch. 5). Having the segmented regions, the feature map's color, intensity, and position values within a proto-object's region are used to compute the associated mid-level features (Ch. 6).

Then the TVA weight equation is used to assign an attentional weight to each proto-object (Ch. 8). For this, three components are necessary: (1) a learned mid-level feature object representation of each potential target object (Ch. 7), (2) a set of pertinence values to determine which object(s) the system searches for, and (3) a mid-level feature representation of the input image's proto-objects (see above). The TVA weight equation computes the attentional weights based on feature similarity between sensory proto-objects and the target object(s). The higher the similarity and the target object's pertinence, the higher the proto-object's attentional weight. Afterwards, more peripherally located objects obtain a relatively lower weight by the inhomogeneity factor (proximity effect).

Additionally, each proto-object obtains an identity value which represents the object class that the proto-object likely belongs to, e.g. black cup or mobile phone (from here on Ch. 9). This value is used to merge proto-objects that have a high probability of belonging to the same natural object in the input image. The resulting proto-objects are stored in a retinotopic attention priority map (APM). The inhibition of return (IOR) mechanism reduces the attentional weights of the last fixated positions in the APM to avoid saccadic oscillations. Then the IOR map is attenuated as a whole in order to make it possible to successively re-fixate these objects. Afterwards, corresponding to the attentional weights in the APM, a winner-takes-all (WTA) mechanism determines the next saccade target. The higher the weight, the more likely a proto-object becomes the next saccadic target. At the position of the winning proto-object, a new inhibited region is created on the IOR map. Then, on the action level, the system executes the saccade, e.g. by means of a robotic head. Finally, the APM is cleared, which implies that the system is memory-less. Finally, a new saccadic cycle can be started, either with the same or a new target object.

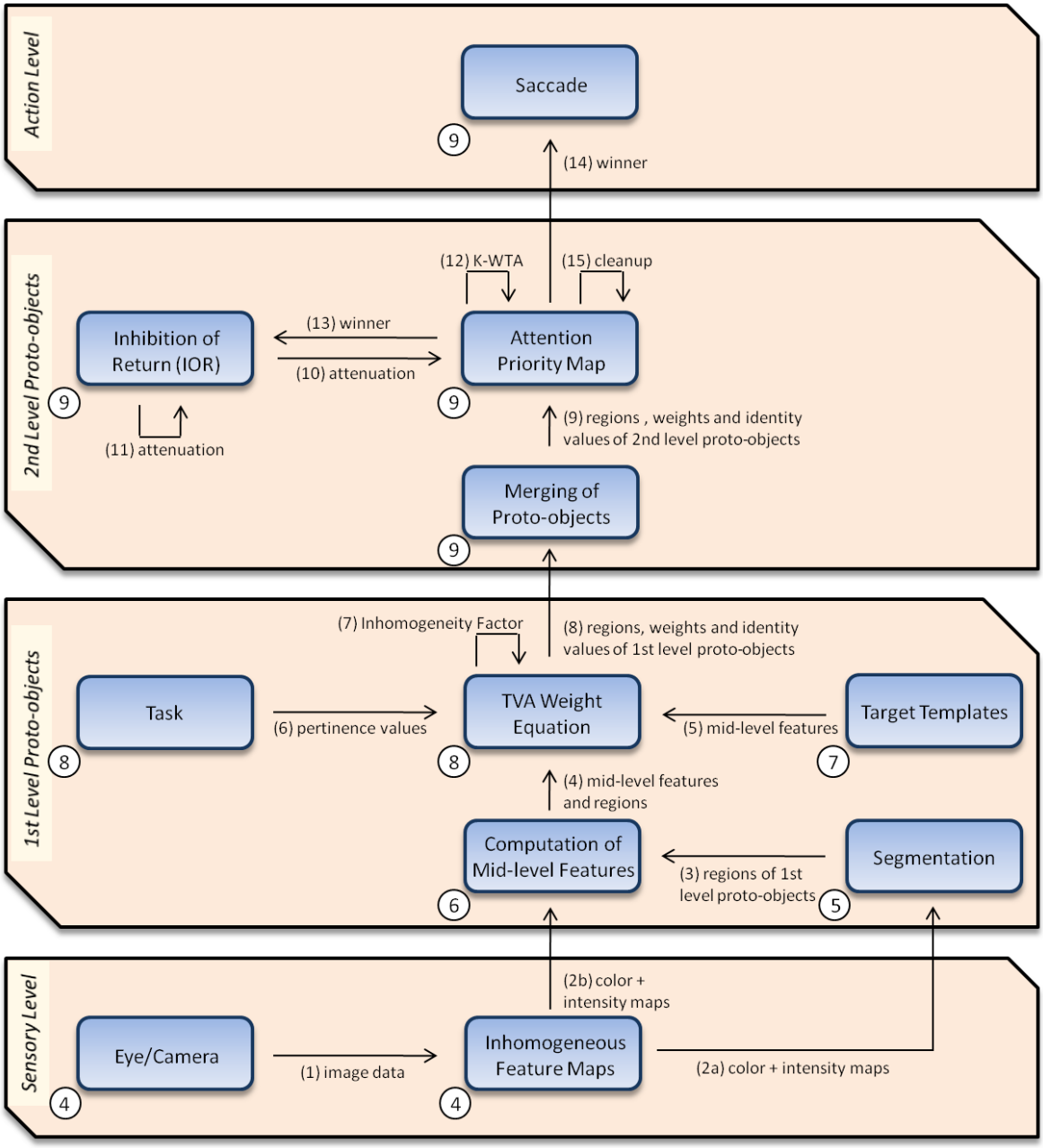

Figure 3.1: *Flow diagram of the model's global structure, see Sec. 3.5 for details. The numbers in the circles denote the corresponding chapter.*

3.6 Summary

In this chapter, a novel model of visual attention was presented, which completely meets the requirement to implement the essential psychological properties of visual attention as presented in Ch. 2. In this model, visual search is based on *learned object representations,* which the model has learned from examples. In the model, learning is realized on the level of *mid-level features*. This feature level allows an *object-based visual search,* as mid-level features do not represent feature values belonging to one pixel (e.g. color, local orientation), but feature values belonging to an object (e.g. size, mean color, shape etc.).

In contrast to other computational models, in this model the whole processing path up to the segmentation of proto-objects , which clusters homogeneous regions in the color and intensity space, is subjected to *spatial inhomogeneous processing.* As a result, the same natural object can be mapped by different values for the same mid-level feature, depending on the location within the visual field. Additionally, natural objects are represented by various proto-objects if they consist of more than one homogeneous color/intensity region. This means that the model has to learn that different sets of mid-level features belong to the same natural object. This has been realized by a feed-forward neural network which serves as a *classifier*.

Based on this classification system, an innovative *merging mechanism* makes it possible to merge proto-objects that belong to the same natural object in the sensory input. Thus, in contrast to other models, this model is able to identify regions that are more complex than e.g. simple colored bars as one object, which is a precondition to handle more than just a small model-compliant subset of natural objects. Furthermore, by merging, the model shows an improved capability to human-like saccade on the center of natural objects.

The probability of a proto-object to become the next saccade's target depends on its attentional weight computed by a *modified TVA weight equation* (Bundesen 1990). That is, the equation is embedded in the classification approach based on the level of mid-level features. The more similar a proto-object to the object the system searches for, the higher the attentional weight and thus the probability. In order to avoid saccadic oscillations, the model has implemented an *inhibition of return* mechanism which attenuates the attentional weight of the last fixated objects. Additionally, the inhomogeneity factor simulates the *proximity effect* in a biologically plausible way by referring back to the spatial inhomogeneous pixel distribution.

In the last section of the chapter, the global architecture of the model was presented in the

form of a flow diagram. The whole path, starting with the sensory level up to the action level, was illustrated. The novel combination and implementation of essential psychological and biological properties results in a unique overall functionality, which no visual attention model has reached so far.

Chapter 4

The spatial inhomogeneous low-level feature map

4.1 Introduction

The usual first processing step in computational models with regard to vision is to build low-level feature maps based on a given input image. Typical low-level features are color, intensity, orientation, depth (in stereo vision), motion (e.g. optical flow) etc. Feature maps, or even one feature map consisting of feature vectors, are built for one or more spatial scales, often in a form of a Gaussian pyramid, see e.g. appendix A in (Walther 2006). These maps then serve as input for subsequent computations, like object segmentation, or for finding the most salient regions. In this thesis' model, only color (red-green, blue-yellow) and intensity (black-white) features are computed and stored in a three-dimensional feature map (see Sec. 4.3). The orientation features typically computed (by so-called Gabor filters, see (Gabor 1946)) are left out because their computation is very expensive and it has been shown that their influence on the model's performance is rather poor.

An important property of the feature map used here (see Sec. 4.2) is its spatial inhomogeneity regarding the findings presented in Ch. 2. Usually, computational models use a spatial homogeneous image-pixel grid, even if they apply some kind of foveation, see e.g. (Zelinsky 2008). Spatial homogeneity means that there is a row- and column-based pixel grid, where the distance between pixels in both dimensions (or the size of pixels if extension is being assigned

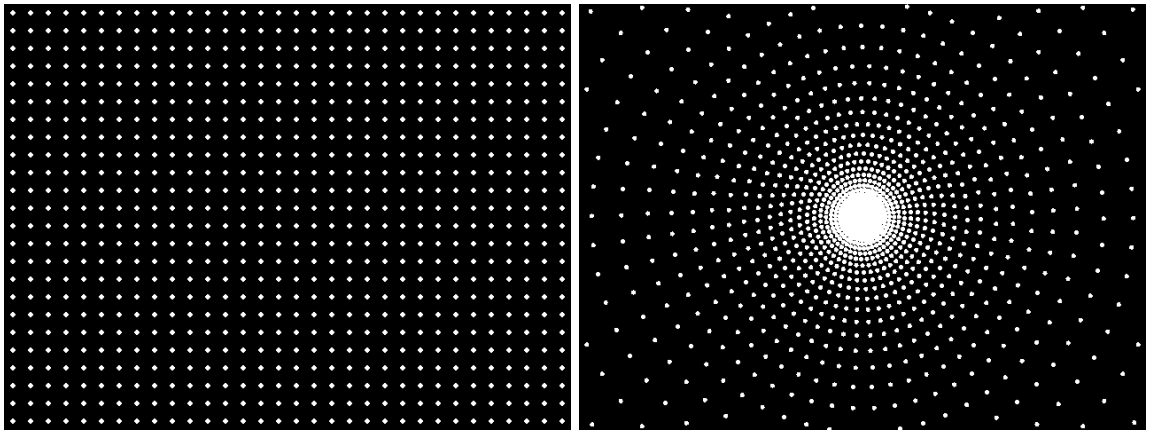

Figure 4.1: *Left: spatial homogeneous pixel grid as typically used for image representations and in image processing. Right: a biologically motivated variant of a spatial, inhomogeneous pixel grid. In general, pixels can be distributed arbitrarily.*

to them) is always identical. A corresponding feature map would have the structure of a *spatial, homogeneous pixel grid*. In contrast, on a *spatial inhomogeneous pixel grid* pixels can be arbitrarily distributed (and spatial homogeneity can be regarded as a special case of spatial inhomogeneity) (see Fig. 4.1).

The biologically motivated structure of the model's spatial inhomogeneous feature map is based on the Watson model (Watson 1983) explained in Sec. 4.2. Basically, the pixel density decreases with increasing distance to the center of the visual field. This center can be located at any position in the input image. The complete processing path of the model consistently underlies this structure of spatial inhomogeneity. The computed feature map serves as input for the segmentation stage (see Ch. 5) as well as for the computation of the proto-objects' mid-level features (see Ch. 6).

4.2 The spatial inhomogeneous pixel grid

According to the Watson model, the spatial, inhomogeneous pixel grid is built as follows: At first, two parameters have to be defined, k and f_{map}. Parameter k determines how strongly the angle of eccentricity influences the peripheral decrease of pixel density. The higher k, the stronger the decrease (see Fig. 4.2). In the model, k has a constant value of 0.4. This comes

from findings in the human visual system, where k is estimated at around this value (Watson 1983). After the value of k was fixated, the relative pixel density for each angle of eccentricity can be obtained by the scaling factor s (see Eq. 4.1).

$$s = 1 + e * k \tag{4.1}$$

Here e denotes the angle of eccentricity. In the visual center, where $e = 0$, s equals 1. The higher e, the higher s and the lower the pixel density. Given two angles of eccentricity e_1 and e_2, with $e_1 < e_2$, and their corresponding scaling factors s_1 and s_2, the density ratio can be computed by $r = \frac{s_2}{s_1}$. This means that at e_1, the pixel density is r-times higher than at e_2.

The second parameter f_{map} determines the overall pixel density (see Fig. 4.2). In the Watson model, this parameter reflects the maximal frequency as well as the spatial density of one layer of Gabor filters in V1 and is limited to a range of 0.25 to 32. An increase of f_{map} yields a proportional increase of density. As the feature maps should map the retinal receptor outputs that, in turn, serve as input for subsequent processing stages like V1, the density has to be doubled to cope with the Nyquist-Theorem. Accordingly, for foveation, the range of f_{map} is 0.5 to 64.

The choice of f_{map} reflects how strongly the system is under time pressure. High time pressure leads to a low f_{map}-value and therefore to a low pixel density. If time pressure is absent, f_{map} can be chosen maximally. Additionally, the parameter f_{map} also influences both the spatial filter density and the filters' size of all subsequent processing stages. So, in the model, higher time pressure leads to a coarser mapping of the environment, which corresponds to existing empirical findings, see e.g. (Lee and Mumford 2003; Deco and Heinke 2007).

The next step is to build the pixel grid based on k and f_{map}. The first pixel is positioned at the center of the visual field. Afterwards, all other pixels are positioned on concentric rings around this central pixel. Thereby the distance d of an inner ring (or the central pixel) to the next outer ring is determined by the scaling factor s_{ring} of the inner ring (see Eq. 4.2).

$$d = \frac{3s_{ring}}{4f_{map}} = \frac{3(1 + e_{ring} * k)}{4f_{map}} \tag{4.2}$$

Additionally, d defines the distance between the pixels and, therefore, implicitly the number of pixels on the outer ring . As d growths with each added ring, both the distance between rings and the distance between pixels on the rings increases with increasing angle of eccentricity. In

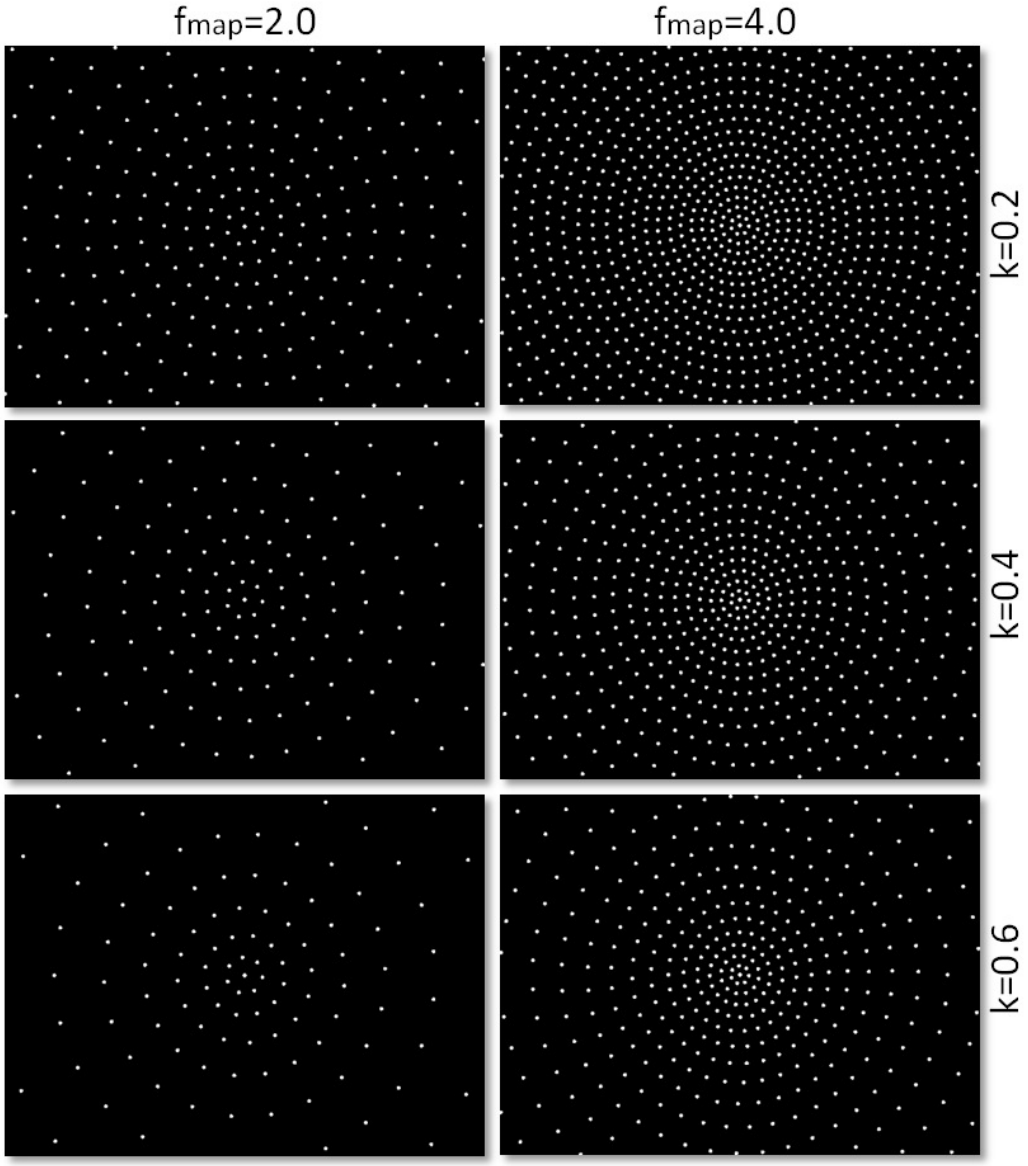

Figure 4.2: *The influence of k and f_{map} on the pixel distribution. Each image was computed with $\theta = 10$, that is, the images' sizes in x-direction equal 10 degree visual angle. f_{map} determines the overall resolution, whereas k determines the influence of eccentricity. As k is fixed to a value of 0.4, the middle row represents potential parameter sets.*

the end, as many rings as necessary can be added to cover a desired area of the visual field.

4.3 Color and intensity

An image from any source, e.g. a robotic camera, with any pixel resolution can serve as input for the feature map computation. The image's size in visual angle has to be specified for this computation. This is realized by parameter θ, which describes the image's size in x-direction. As the pixel-to-visual-angle ratio is identical in x- and y-direction, the image's size in y-direction can directly be derived from θ (see Fig. 4.3).

Normally, color values are specified according to the 3-dimensional RGB color space. To fit the approach of human-like modeling, these RGB-values are transformed into the physiological RG/BY/BW color space (Walther and Koch 2006). Thus, the model computes two color values (RG and BY) and one intensity value (BW). In this model, in contrast to Walther's, all color dimensions have the same range of $[0..1]$ to avoid an unwanted weighting of dimensions in the subsequent segmentation stage. The transformation, given that the vectors of RGB-values have a range of $[0..1]^3$, works as follows:

$$bw = \frac{r+g+b}{3} \tag{4.3}$$

$$rg = \frac{1}{2}\left[\frac{r-g}{\max(r,g,b)} + 1\right] \tag{4.4}$$

$$by = \frac{1}{2}\left[\frac{b-\min(r,g)}{\max(r,g,b)} + 1\right] \tag{4.5}$$

After the pixel grid of the feature map was fixed by the parameters k, f_{map}, and θ, each pixel of the map gets its corresponding feature vector (rg, by, bw) from the pixel in the input image which is spatially located closest to it (see Fig. 4.4).

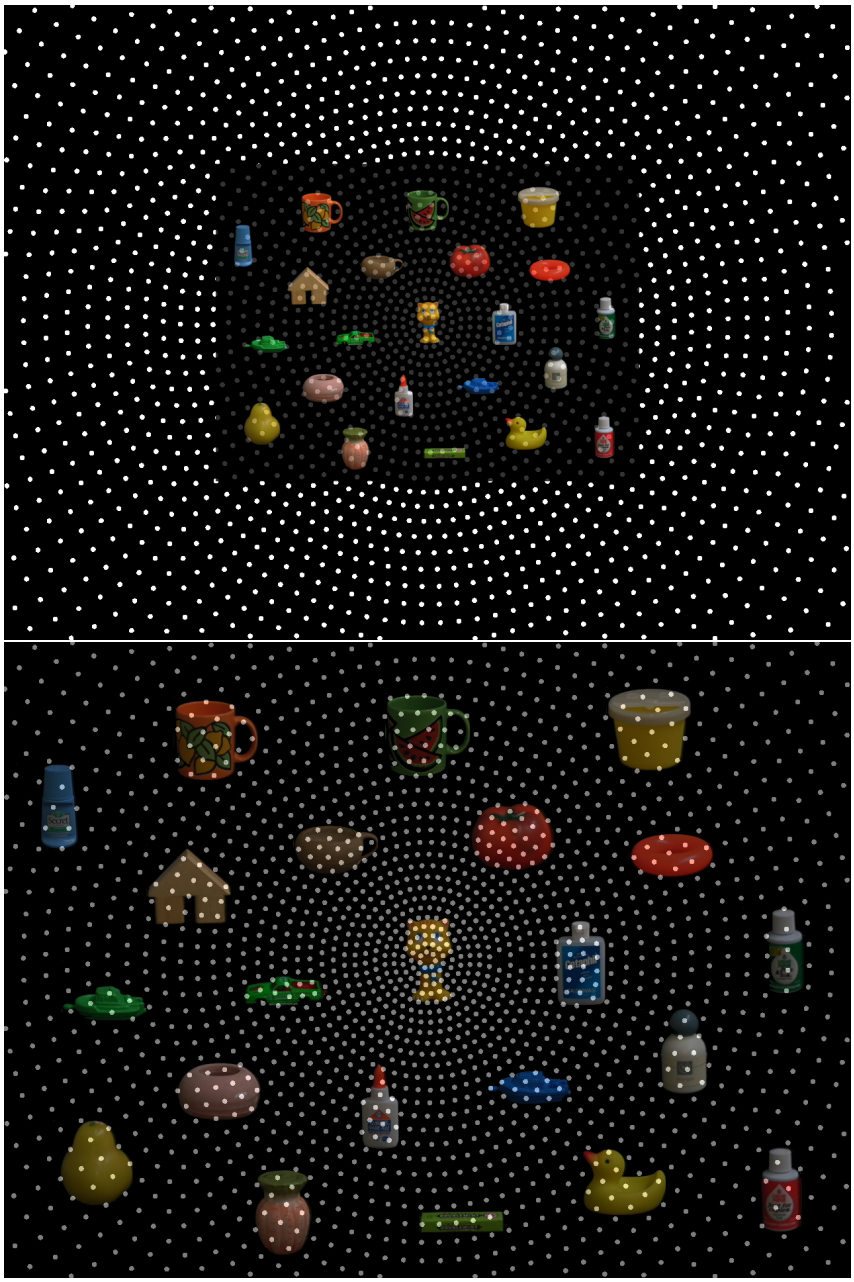

Figure 4.3: *The influence of θ. In both examples the same pixel grid is used with $k = 0.4$ and $f_{map} = 8.0$. The range of the shown areas equals 10 degree visual angle in x-direction. In the first example, the value of θ equals 5.0 (top). In the second example θ equals 10.0 (bottom), so the full area shown is covered. An increase of θ can be interpreted in two ways: objects are either closer or bigger. For robotic applications, θ can be gained from the cameras' properties.*

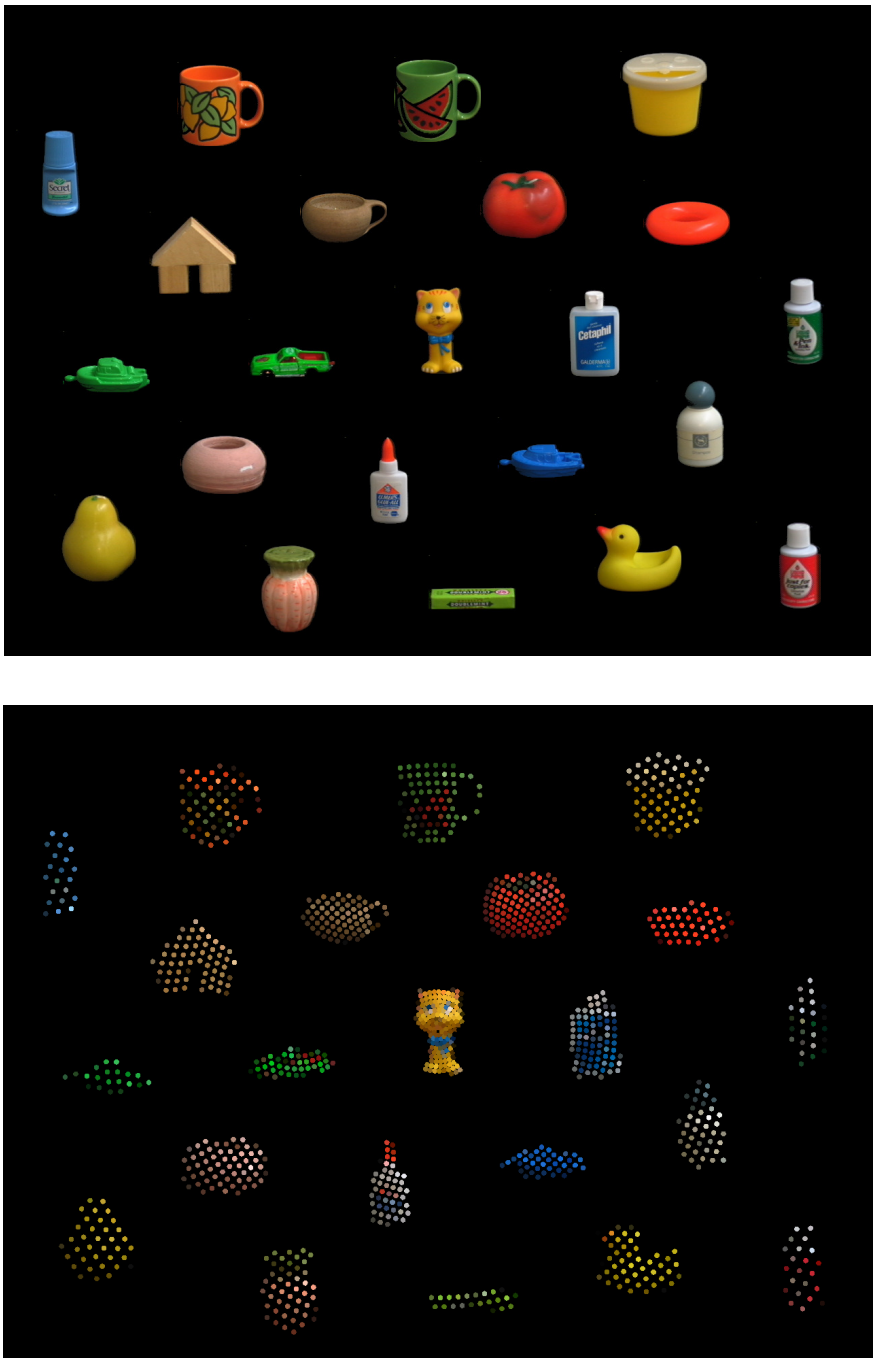

Figure 4.4: *An example feature map (bottom) with* $k = 0.4$*,* $f_{map} = 16.0$*, and* $\theta = 10.0$*. As can be seen, more foveally located objects are represented by more pixels. Object images (top) come from the COIL, see App. B.*

4.4 Summary

Motivated by findings in primates, all stages of the model are subjected to spatial inhomogeneous processing. In this first stage, the building of the low-level feature map, spatial inhomogeneity refers to the density of retinal receptors. With increasing angle of eccentricity, the density of retinal receptors decreases. This eccentricity-dependent scaling is simulated by the scaling factor s (see Eq. 4.1), which comes from the Watson model (Watson 1983).

The feature map can be computed for different processing times. The more the system is under time pressure, the lower is the obtained spatial resolution. If time pressure does not exist, the highest possible resolution is chosen.

The feature map contains feature values for color and intensity, where color is subdivided into two dimensions corresponding to the physiological RG and BY color space dimensions (Walther and Koch 2006).

Chapter 5

Proto-object segmentation by clustering

5.1 Introduction

Motivated by the findings described in Ch. 2, this model aims at assigning a set of mid-level features to each proto-object. In order to realize this approach, a segmentation stage is needed to obtain regions of proto-objects. That is, after segmentation each proto-object is associated with a set of pixels in the low-level feature map. Then, each proto-object's mid-level features can be computed based on its associated pixel set (see Ch. 6).

As proto-objects are quickly-built, coarse representations of natural objects, a suitable segmentation has to meet these requirements. Such an approach can be found in (Forssén 2004). Here, the image is segmented by clustering homogeneous color/intensity regions. The model used here adopts this clustering concept, but substantially expands on it (see Sec. 5.2). The result is the first clustering algorithm that is able to completely work on a spatial inhomogeneous pixel grid. As a result, and in accordance with psychological findings, segmentation results depend on eccentricity. So, e.g., it can be shown why small objects peripherally disappear or why saccades into the periphery tend to land on two objects' center of gravity (*global effect*, (Findlay 1982)) (see Sec. 5.4). Additionally, it is shown that this clustering approach is robust against parameter variation (see Sec. 5.3).

Another problem concerns *figure-ground segmentation*. In this model, this corresponds to the question of which cluster belongs to an object and which to the background. For this, the model implements a simple filtering stage, where potential background clusters, but also potential

artefacts, are removed (see Sec. 5.2.5).

5.2 A clustering algorithm for spatial inhomogeneously arranged data

5.2.1 The Gaussian pyramid

The basic idea of Forssn's clustering approach is to identically label adjacent pixels that have similar values in the RG/BY/BW-space to obtain a set of homogeneous regions in the input image. To this end, the model makes use of a Gaussian pyramid. The pyramid consists of a certain number of layers, each computed by a set of Gaussian filters. For the segmentation process, these filters are defined by their position $\mu = (x, y)$ within the visual field and their standard deviation σ (see Eq. 5.1).

$$G(\mu, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{1}{2}\left[\frac{x^2}{\sigma^2} + \frac{y^2}{\sigma^2}\right]\right) \tag{5.1}$$

The pyramid is constructed as follows: The filters' positions are computed just as the pixels' positions of the feature map. However, the general spatial density, defined by parameter f_{map} (see Sec. 4.2), is halved from layer to layer. For the n-th layer, parameter f_n is computed by

$$f_n = \frac{f_{map}}{2n} \tag{5.2}$$

The standard deviation σ of each filter depends on parameter f_n (layer-dependent) and parameter s (eccentricity-dependent) (see Eq. 5.3).

$$\sigma = \frac{3s}{8f_n} \tag{5.3}$$

The latter parameter grows with increasing eccentricity (see Sec. 4.2). Thus, the standard deviation increases both with increasing eccentricity and each added filter layer (see Fig. 5.1).

As the filter density decreases for each added layer, the last layer, which builds the top of the pyramid, only consists of one central filter. The total number of layers varies depending on

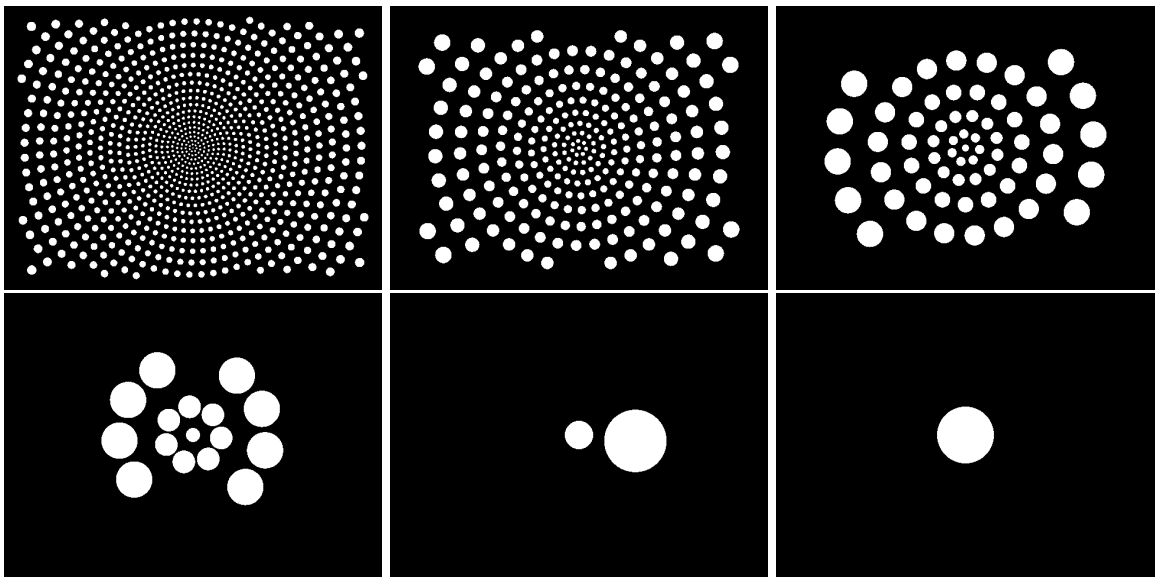

Figure 5.1: *Structure of the Gaussian pyramid. Here the topmost six layers are shown.*

parameters k and f_{map}, but also on θ (see Sec. 4.2), which defines the size of the visual field represented by the input image.

In the following, a coarse overview is given on how the segmentation algorithm works.

- (1) Assign a binary confidence value $c \in \{0, 1\}$ to each filter in the Gaussian pyramid. The confidence value reflects the degree of homogeneity of the pixels (in the RG/BY/BW-space) within a filter's area. Depending on the predefined parameter c_{thr}, a value of 1 is assigned if a threshold is exceeded, which is only the case for sufficiently homogeneous areas (see Sec. 5.2.2).

- (2) Assign a label $l \in \mathbb{N}$ to each filter with $c = 1$, where each label represents one proto-object. Labels are propagated from layer n to layer $n - 1$ and finally to the feature map (see Sec. 5.2.3 for details). All pixels on the feature map that have the same label belong to the same proto-object. Such a set of equally labeled pixels is called a *region* of a proto-object.

- (3) As the stages (1) and (2) yield an over-segmentation, an additional merging stage is necessary (see Sec. 5.2.4).

- (4) Finally, a proto-object is deleted if the number of pixels that determine its region is

too small (likely artefacts) or too big (likely part of the background) (see Sec. 5.2.5).

While stage (1) is realized as a bottom-up process, stage (2) builds potential regions of objects in a subsequent top-down process. This temporal order of bottom-up and top-down processing corresponds to empirical findings in humans, see e.g. (Roelfsema 2006).

In contrast to Forssn's model, filter density as well as filter size in this model depend on the angle of eccentricity. This affects all four processing stages just listed. As a consequence, the resulting segmentation itself depends on eccentricity. In the following, the four processing stages are described in detail. For each stage, it is explained how eccentricity influences the result.

5.2.2 The computation of confidence values

The first step in the segmentation process is to assign a binary confidence value $c \in \{0, 1\}$ to each filter in the Gaussian pyramid. The term *confidence* is taken from Forssn's work (Forssén 2004) and describes whether the RG/BY/BW-values of the pixels in a filter's area are sufficiently homogeneous. If this is the case, a value of 1 is assigned to c;otherwise is assigned.

The computation of confidence values for a filter layer is subdivided into two steps. At first, the filter responses of the Gaussian filters are computed. For each filter layer n, the input data comes from the filter output of the underlying layer $n-1$, respectively, for the undermost layer from the feature map. Thus, it involves a bottom-up process. Based on the computation of filter responses, in a second step, the confidence values can be computed. In the following, it is described how the model realizes both steps. The computation is identical for all filters in the pyramid.

At first, in order to compute the filter responses, for each filter a set of pixels P is built, which contains all pixels that are located within a radius of 2σ in the underlying layer. In the pyramid layers, the pixels' positions and values correspond to the filters' positions and responses. Then, the filter response $\bar{f} \in [0..1]^3$ of each filter is computed by an optimization technique (see Eq. 5.4).

$$\arg\min_{\bar{f}} \sum_{p \in P} p_w p_c(\| \bar{f}_p - \bar{f} \|) \tag{5.4}$$

Here p_w equals the Gaussian value at the position of pixel p. As a result, p_w realizes a pixel

weighting, where those pixels that lie closer to the filter's center μ receive a higher weighting. p_c represents the confidence value of pixel p. Thus, only those pixels whose confidence value equals 1 influence the computation of $\bar{f}$ Logically, the confidence values of the feature map's pixels have to be set to 1. The term $\parallel \bar{f}_p - \bar{f} \parallel$ represents the Euclidean distance between $\bar{f}_p$ of pixel p and $\bar{f}$ in the RG/BY/BW-space. In sum, this means that the aim of the optimization process is to minimize the sum of weighted Euclidean distances for the subset of pixels with $p_c = 1$.

The optimization is realized by an *iterative* technique that combines two approaches, *iterated re-weighted least square* (IRLS) (Zhang 1995; Stewart 1999) and *successive outlier rejection* (SOR), which assigns an *outlier value* p_l to each pixel. The outlier value realizes a pixel weighting, just as p_w. But while p_w depends on p's location, p_l depends on p's distance p_d to $\bar{f}_i$ in the RG/BY/BW-space (see Eq. 5.5). Here $\bar{f}_i$ corresponds to the $\bar{f}$-value in the i-th iteration step.

$$p_d = \parallel \bar{f}_p - \bar{f}_i \parallel \tag{5.5}$$

According to IRLS, the higher the distance, the lower p_l. According to SOR, if the distance exceeds the value of parameter d_{max1}, p_l is set to zero (see Eq. 5.6). The computation of p_l-values corresponds to the bi-weight error norm (Zhang 1995).

$$p_l(p_d) = \begin{cases} (1 - (\frac{p_d}{d_{max1}})^2)^2, & \text{if } \mid p_d \mid < d_{max1} \\ 0, & \text{else} \end{cases} \tag{5.6}$$

Given the p_l-values, the next $\bar{f}_i$ value can be computed (see Eq. 5.7 and 5.8).

$$\bar{f}_i = \mathcal{N} \sum_{p \in P} p_w p_c p_l \bar{f}_p \tag{5.7}$$

$$\frac{1}{\mathcal{N}} = \sum_{p \in P} p_w p_c p_l \tag{5.8}$$

As the computation of p_l values is a precondition to compute the $\bar{f}_i$ value, all p_l values are set to 1 for the first iteration. Then, during each subsequent iteration, first the new p_l values and then the new $\bar{f}_i$ value is computed. The remaining parameters $\bar{f}_p$, p_c, and p_w are constant across all iterations. The total number of iterations is determined by parameter i_{num}. The result of

the IRLS approach is that the higher p_d is, the less the corresponding pixel p influences the out-coming $\bar{f}$. Additionally, the SOR method ensures that p_d-values that are too high lead to a complete rejection of outliers with $p_l = 0$.

After the last iteration with $\bar{f} \equiv \bar{f}_{i_{num}}$, in a second step, the confidence value c can be computed for each filter (see Eq. 5.9).

$$c = \begin{cases} 1, & \text{if } \sum\limits_{p \in P} p_w p_c p_l \geq c_{min} \sum\limits_{p \in P} p_w \\ 0, & \text{else} \end{cases} \tag{5.9}$$

The probability that a confidence value of 1 is assigned to a filter decreases

- with a decreasing number of pixels, which themselves have a confidence value p_c of 1. This avoids that, within the Gaussian pyramid, a confidence value of 1 is assigned to a filter although the filters of the underlying layer located in the filter's area mostly have a confidence value of 0, which in turn speaks for an inhomogeneous region.
- with an increasing number of pixels that have a low p_l-value.
- if many pixels have a high p_l value and p_c value of 1, but the p_w-values of those pixels are very low.

The parameter c_{min} allows a fine-tuning of stage (1). The effect of varying c_{min} as well as other parameters is depicted in Sec. 5.3. Fig. 5.2 illustrates the computation of confidence values.

Due to the peripherally increasing filter size, homogeneous regions in the RG/BY/BW-space that are located more peripherally have to be respectively larger to obtain a confidence value of 1 (see Fig. 5.3). Thus, the computation of the confidence values and to that effect also the segmentation strongly depends on the angle of eccentricity (see next section for details).

5.2.3 Homogeneous regions by label propagation

After a confidence value $c \in \{0, 1\}$ was assigned to each filter of the Gaussian pyramid in the bottom-up process, the model assigns a label $l \in \mathbb{N}$ to each filter of the pyramid and to each pixel of the feature map with $c = 1$ in a top-down process. At first, each l value is initialized

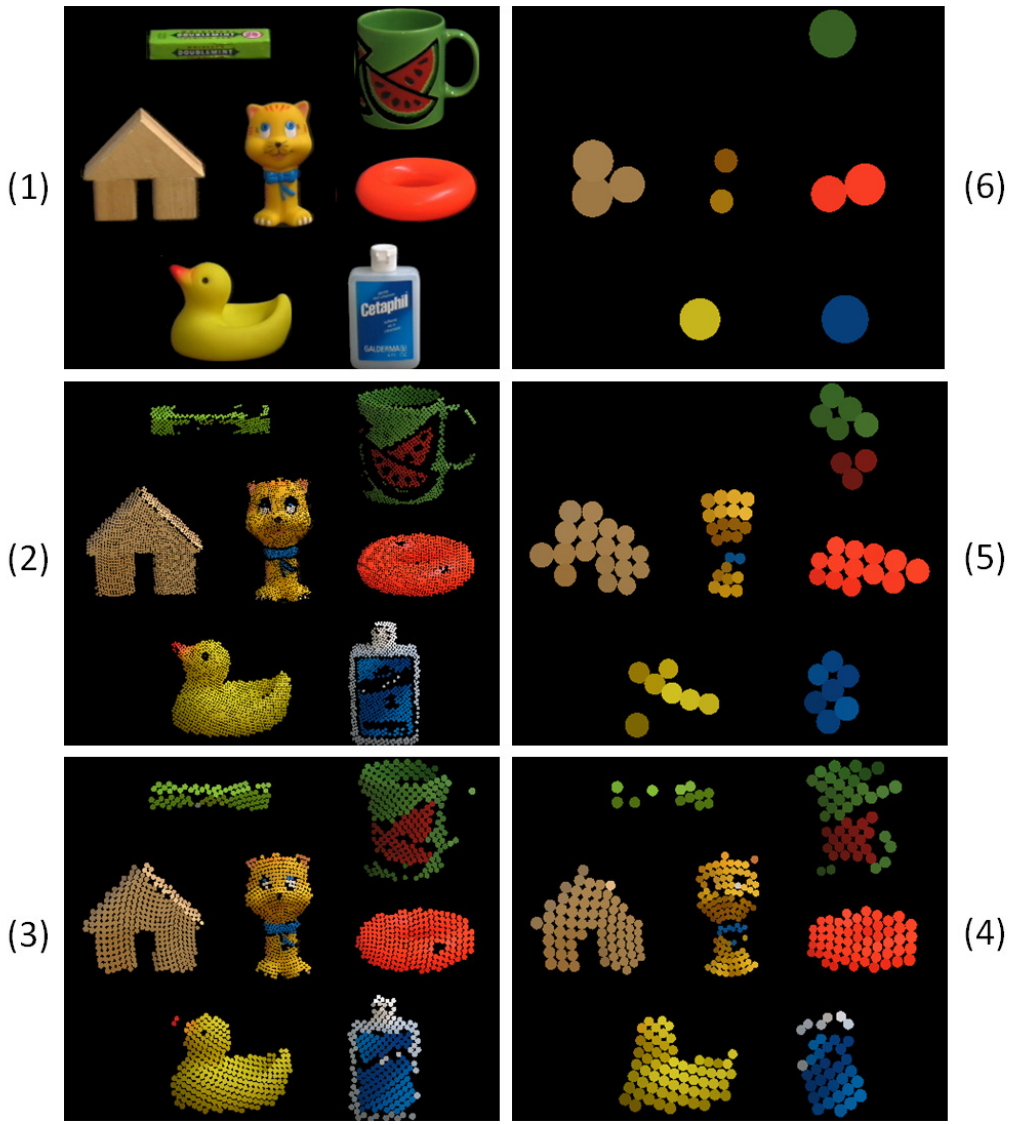

Figure 5.2: *Computation of confidence values. Within the Gaussian pyramid, only those filters obtain a confidence value of* 1 *whose input data is sufficiently homogeneous in the RG/BY/BW-space. In the figure, only filters with* $c = 1$ *were drawn into the images. (1) Input image. (2)-(6) Layers of Gaussian pyramid. Higher layers do not obtain any filters with* $c = 1$.

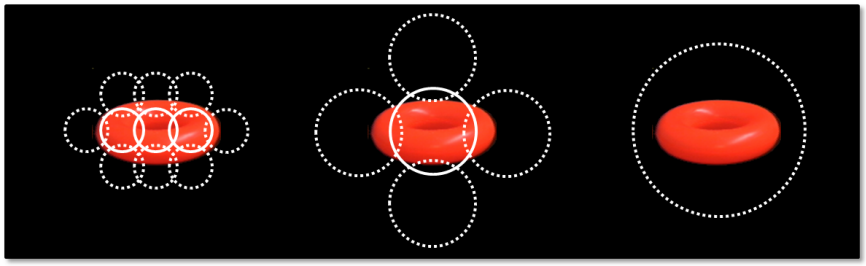

Figure 5.3: *Schematic illustration of how confidence values depend on eccentricity. The circles mark the margins of the Gaussian filters whose standard deviation always equals 2σ. A filter's circle is drawn with a solid line if it represents a region of the object with a confidence value of 1. The latter means that the area under the filter has to be sufficiently homogeneous in the RG/BY/BW-space. Left: three filters fulfill these criteria. Middle: angle of eccentricity was doubled, which in turn doubles the filters' size in both dimensions. Now only one filter is left. Right: Again, the angle of eccentricity was doubled. Now the object is too small for being represented by any filters. As a result, the filter over the object with $c = 0$ cannot inherit any label from the filter layer above and therefore cannot pass on any labels to any pixels in the feature map (see Sec. 5.2.3). This means the object cannot be represented by a proto-object.*

with 0. Then, starting with the topmost layer n and $l = 1$, the model executes both steps listed below and then proceeds with layer $n - 1$ until the undermost layer is reached.

- If a layer's filter with $c = 1$ is not assigned to a label, it gets the label l and l is increased by 1. Thus, if m filters of a layer meet this criterion, the model assigns m new labels. Such an assignment is skipped for the feature map's pixels, as this would produce proto-objects that only consist of one pixel and, therefore, wouldn't survive the filtering stage (see Sec. 5.2.5).

- Each filter F_{upp} propagates its label to those filters/pixels F_{low} of the lower layer/feature map that meet the following three criteria:

  - The center of F_{low} is located within the filter's area of F_{upp}. Similarly, in the bottom up process, the area's radius of F_{upp} equals 2σ.
  - F_{low} has a confidence value of 1.
  - The Euclidean distance of the filter/pixel value in the RG/BY/BW-space has to be less than d_{max1} (see Eq. 5.10), where $\bar{f}_{upp}$ equals the filter response of F_{upp} and

$\bar{f}_{low}$ equals the filter response of F_{low}. Parameter d_{max1} controls the measure of inhomogeneity that is allowed within a proto-object (see Sec. 5.3 for details). Higher values allow a higher degree of inhomogeneity.

If two or more filters could propagate their label to a filter/pixel in the lower layer/feature map, then the filter that produces the smallest Euclidean distance, in accordance with equation 5.10, wins.

$$\| \bar{f}_{upp} - \bar{f}_{low} \| \leq d_{max1} \tag{5.10}$$

The result of the label propagation process is that the feature map consists of two classes of pixels. Those pixels that still have a l-value of zero are not part of any proto-object, whereas those pixels with $l > 0$ belong to the proto-object with label l. This means that each proto-object consists of a set of pixels that have the same l-value. Such a set of pixels is called a region of a proto-object. The total number of proto-objects equals the number of assigned labels. Figure 5.4 shows an example.

In the simplest case, one natural object is mapped by one proto-object. However, there are three further possibilities. First, if a natural object isstructured too inhomogeneously in the RG/BY/BW-space, the model does not produce a proto-object that represents this natural object. At this point, the model reaches its limit, as those natural objects cannot be mapped and therefore cannot be the target of a saccade. Second, a natural object can be mapped by multiple proto-objects. This happens if a natural object consists of multiple homogeneous regions in the RG/BY/BW-space. How the model handles this case is part of a later processing stage (see Sec. 9.2). Third, two or more natural objects can be mapped by only one proto-object. This happens if nearby natural objects have regions with similar values in the RG/BY/BW-space, so that a filter from the upper layer can propagate its label to all those regions. The latter case is comprehensively described in Sec. 5.4. Examples for the remaining three cases are shown in Fig. 5.5.

In the previous section, it was deduced that homogeneous regions in the RG/BY/BW-space that are located more peripherally have to be larger to obtain a confidence value of 1. As a confidence value of 1 is a precondition to obtain a label $l > 0$ that, in turn, marks the affiliation with a certain proto-object, in the periphery, homogeneous regions that are too small are not mapped by a proto-object. Additionally, the larger a filter, the greater the maximal distance between two filters/pixels of the lower layer/feature map that can inherit a filter's label. As a

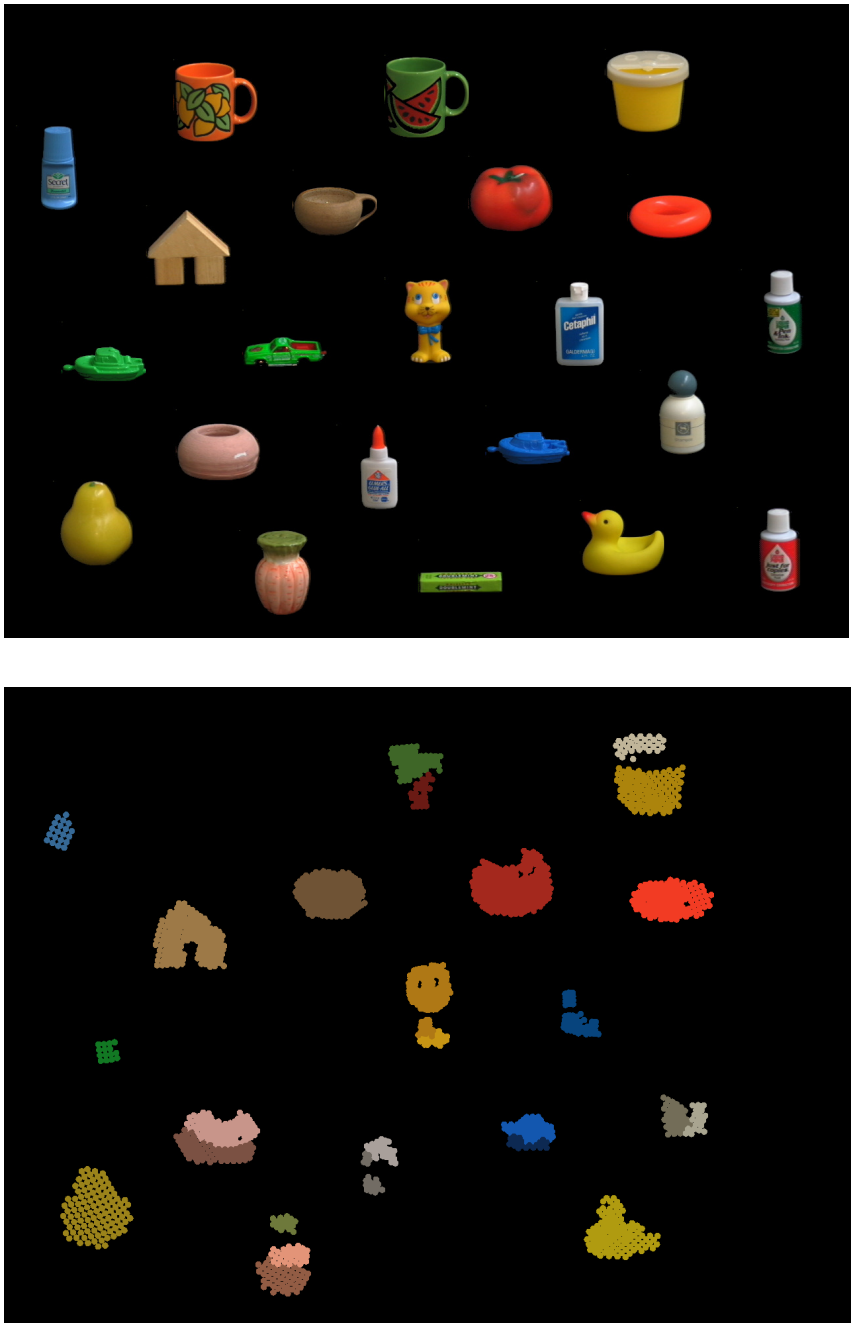

Figure 5.4: *Regions as equally labeled pixels in the feature map with* $k = 0.4$, $f_{map} = 32.0$, $\theta = 8.0$, *and* $p_{min} = 15$. *The remaining parameters correspond to their standard values, see Tab. 5.1. In order to better distinguish nearby regions, they are colored by their mean color.*

Figure 5.5: *Regions of objects. The examples come from Fig. 5.4. Top row: object images. Middle row: feature map representations. Bottom row: resulting regions. Left column: As the feature map representation shows, the cup is too inhomogeneously structured in color, so the model is not able to find a large enough homogeneous color region. A more foveal representation of the cup could lead to a different result, as it would be mapped by more pixels (see Fig. 5.7). Middle column: Almost the whole duck is represented by one single region. Only the beak gets lost. Right column: The cup is represented by two regions, as it consists of two large enough homogeneous color regions.*

consequence, the maximum distance between two different, natural objects that can inherit the same label, and therefore would be mapped by only one proto-object, increases with increasing angle of eccentricity.

5.2.4 Merging of regions

Sometimes in the first two processing stages (confidence value computation and label propagation), a homogeneous region in the input image is divided into multiple regions and therefore multiple proto-objects. This is called over-segmentation. On this account, the model implements a subsequent merging stage. Two regions m and n are merged if

- the Euclidean distance between the mean values of both regions ($\bar{f}_n$ and $\bar{f}_m$) in the RG/BY/BW-space is less than d_{max2} (see Eq. 5.11) and
- both regions sufficiently overlap (see Eq. 5.12).

While the first criterion is easy to compute, the second one requires a more complex computation. At first, it has to be determined to what degree regions mutually overlap. To do this, the model makes use of an adjacency matrix $\bar{A}$. The value $\bar{A}_{mn}$ reflects the degree of overlapping of region m and n. All matrix elements are initialized with 0. To compute the overlapping values, the model again makes use of the undermost filter layer in the Gaussian pyramid. For each filter, a list is created that contains all label values with $l > 0$ of those pixels of the feature map that are located within the filter's scope of 2σ. During this process, a certain label value can appear only once in one list (see Fig. 5.6).

Then, for each possible combination (m, n) with $m < n$ of labels that are elements of a list, the value of $\bar{A}_{mn}$ is increased by 1. Finally, each matrix element $\bar{A}_{mn}$ with $m < n$ represents the total number of filters in whose scope at least one pixel of region m and one pixel of region n are located. In a next step, for each possible combination (m, n) with $m < n$, the model merges the regions m and n if $\bar{A}_{mn}$ exceeds threshold m_{thr} (see Eq. 5.12). As the merging process produces new regions with new mean values in the RG/BY/BW-space, the merging procedure is repeated until no more merges occur.

$$\| \bar{f}_n - \bar{f}_m \| \leq d_{max2} \tag{5.11}$$

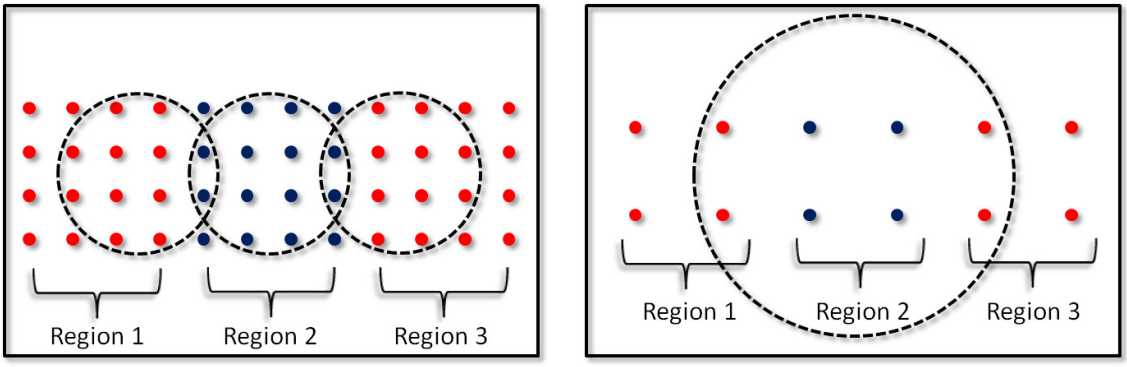

Figure 5.6: *Computation of adjacency matrix $\bar{A}$. Left: In the left filter's scope, pixels of region* 1 *and* 2 *can be found. Correspondingly, its list of regions would be* $\{1, 2\}$ *and* $\bar{A}_{12}$ *would be increased by* 1*. The middle filter does not influence* $\bar{A}$ *because its list only consists of one region. In analogy to the left filter, the right one would increase* $\bar{A}_{23}$ *by* 1*. Importantly, although regions* 1 *and* 3 *have very similar color values, they cannot be merged because the filters' scope is too small. Right: A doubling of eccentricity roughly leads to a quadrupling of the filters' scope and a quartering of the pixels' density. Given the same sensory input regions* 1 *and* 3 *are now in the scope of one filter. In the case shown here, the filter's list would be* $\{1, 2, 3\}$ *and* $\bar{A}_{12}$*,* $\bar{A}_{13}$*, and* $\bar{A}_{23}$ *would be increased by* 1*. So, depending on parameter* m_{thr}*, see Eq. 5.12, regions* 1 *and* 3 *could potentially be merged.*

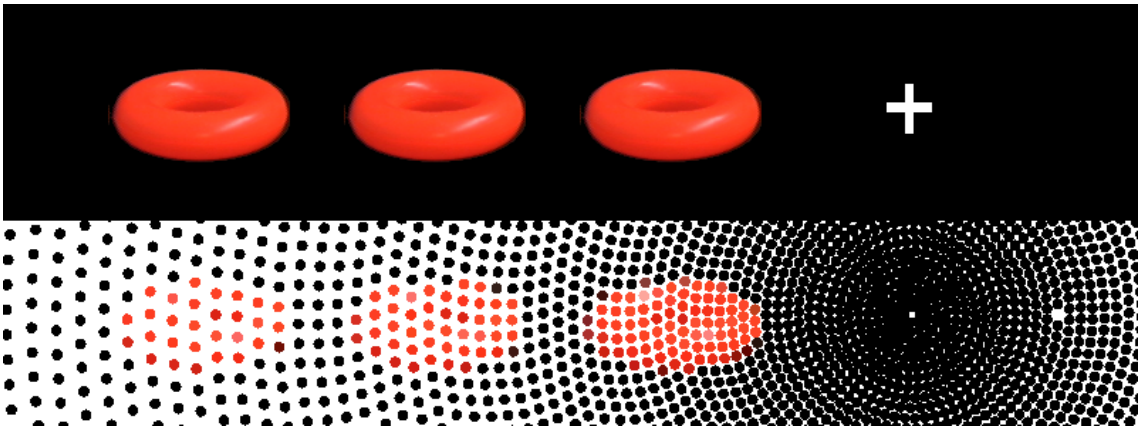

Figure 5.7: *An object displayed at three different eccentricities (top). The cross marks the visual center. The more peripherally the object is located, the lower the number of pixels* o_p *it consist of (bottom).*

$$m_{thr} < a_{n,m} \tag{5.12}$$

In this case, the eccentricityagain affects the result. An element in the adjacency matrix is only increased by 1 if pixels of different regions are located in a filter's scope of 2σ. As this scope increases with increasing angle of eccentricity, the greatest possible distance between two regions that can be merged increases too (see Fig. 5.6). This results in more merges and therefore a coarser mapping of natural objects in the periphery (see Sec. 5.4).

5.2.5 Filtering of regions

Finally, the model filters out regions that are too small (likely artefacts) or too big (likely background clusters) (see Eq. 5.13). Here o_p denotes the number of pixels of a proto-object's region with p_{min} being the smallest and p_{max} the greatest possible number of pixels a region is allowed to consist of.

$$p_{min} <= o_p <= p_{max} \tag{5.13}$$

Due to the fact that the pixel's density decreases with increasing angle of eccentricity, the model shows the effect that the region of the same natural object consists of fewer pixels if it is more peripherally located (see Fig. 5.7). As a result, small natural objects whose number of pixels o_p in the fovea does not exceed p_{min} by much are filtered out above an angle of eccentricity with $o_p < p_{min}$. The same applies, in reverse, to large natural objects. If the number of pixels o_p in the periphery does not fall much below p_{max}, objects are filtered out below an angle of eccentricity with $o_p > p_{max}$. Thus, large natural objects more likely survive in the periphery whereas small natural objects more likely survive in the foveal area. But this, of course, depends strongly on how a natural object was segmented. The deciding criterion is the number of pixels in a region. Thus, even large natural objects can survive in the foveal or parafoveal area e.g. if they were split into multiple smaller regions. Then, a proto-object represents not the whole natural object, but a part of it.

This last processing stage is not part of Forssn's segmentation algorithm. Appending this stage fulfills two essential purposes. The parameter p_{min} eliminates segmentation-caused artefacts that mostly occur at the borders of natural objects. Moreover, p_{max} realized a coarse solution for

the well-known problem to separate objects from the background, the so-called figure-ground segmentation, see e.g. (Denecke, Wersing, Steil, and Körner 2009).

In sum, the model considerably augments the capabilities of Forssn's original segmentation algorithm. First, the segmentation result strongly depends on the angle of eccentricity. Second, an new appended filtering stage eliminates proto-objects that likely represent artefacts or (parts of) the background. Fig. 5.8 illustrates the different results of both models by means of an example.

5.3 Parameters, variation, and robustness

This section provides an overview of all segmentation parameters (see Tab. 5.1). For each parameter the standard value as well as its function is shown. Thereby k, f_{map}, and θ are global model parameters as they also influence earlier and/or later processing stages.

There are no "perfect" standard values, so each value has to be understood as a compromise. If, e.g., p_{max} is chosen to be low, bigger natural objects tend to be identified as background and therefore would be filtered out. On the other hand, if p_{max} is chosen to be high, parts of the background may become proto-objects. Thus, the value of p_{max} as well as all other values of parameters of the segmentation stage were chosen to minimize unwanted results. In the end, even after being optimized, the model still sometimes produces errors regarding segmentation, merging, and filtering, but so do biological systems too.

Another point is that one could argue that it is not plausible to assume that the parameters are constant (except for parameter f_{map}, which simulates time pressure). E.g. p_{min} could be lowered if the system searches for small objects or d_{max1} could be lowered if the system searches for an object within a scene where objects have similar colors. So far such a task- or scene-dependent parameter assignment is not part of the model, but could be a promising approach for future research.

Another important point is parameter robustness. Robustness means that small changes of parameter values should cause only small changes in the results. To this end, Figs. 5.9, 5.10, 5.11, and 5.12 depict an appropriate variation of the most influential segmentation parameters. The input image uses the same seven central objects as used in Fig. 5.4, but moved closer together and with $\theta = 3.5$. If possible, the figures show a combined variation of one stage's parameters to illustrate how combined parameter variation within one processing stage influences the re-

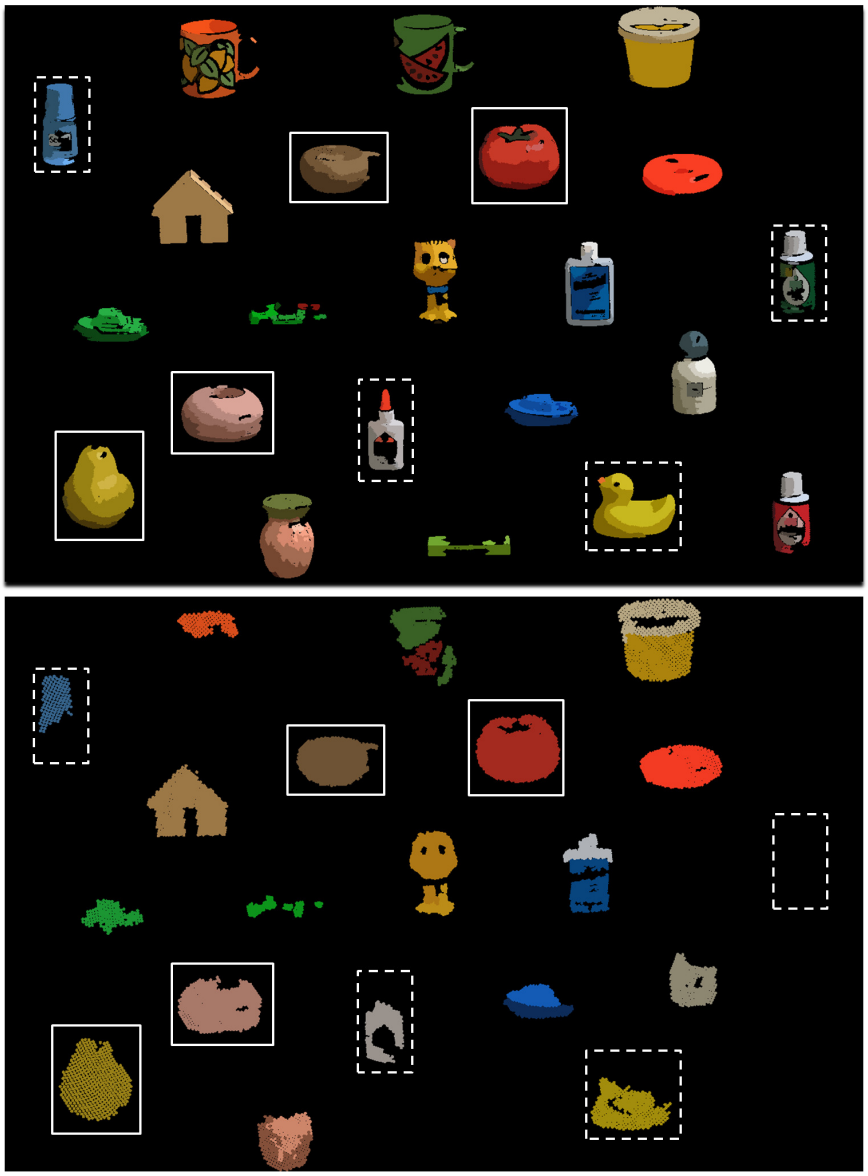

Figure 5.8: *Comparison of segmentation results using standard parameters. Top: Forssn's model. Bottom: the thesis' model with* $\theta = 8.0$*. The input image is the same as in Fig. 5.4. The results mainly differ in two points. First, in Forssn's model, fairly homogeneously colored areas within an object are segmented into considerably more regions (see e.g. solid frames). These regions poorly represent objects as a whole and the actual shape of objects, respectively. Second, in the thesis' model, segmentation results depend on eccentricity: With increasing eccentricity objects are mapped increasingly more coarsely, so more and more details get lost, and finally disappear (see e.g. dashed frames). See Sec. 5.2 for more detailed information.*

stage	parameter	value	function
confidence value	d_{max1}	0.15	determines the homogeneity of proto-objects
	c_{min}	0.65	minimum filter respondse for a confidence value of 1
	i_{num}	3	determines the number of iteration steps
label propagation	d_{max1}	0.15	see above
merging	d_{max2}	0.15	merging threshold in the RG/BY/BW space
	m_{thr}	5	the minimum number of common filters
filtering	p_{min}	100	smallest possible size of a proto-object's region
	p_{max}	10000	greatest possible size of a proto-object's region
global	f_{map}	64.0	determines overall pixel/filter density
	k	0.4	determines the influence of eccentricity, fixed value
	θ	none	size of input image in visual angle in x-direction

Table 5.1: *Segmentation parameters and their standard values.*

sult and to generally show that even combined parameter variation does not lead to non-robust results. In the figures, the central image always shows the result for standard parameter values.

5.4 The "Global Effect"

It is a characteristic property of the model that the inhomogeneity of the low-level feature map as well as the Gaussian pyramid yield an eccentricity-dependent implicit merging of natural objects. This means that with decreasing spatial resolution, adjacent objects tend to become indistinguishable. This fusion effect appears most strongly in the periphery of the visual field. The cause for the fusion's eccentricity-dependency is found in two properties of the segmentation:

- The maximum distance between two different natural objects that can be mapped by only one region increases with increasing angle of eccentricity (see Sec. 5.2.2 and 5.2.3).
- The greatest possible distance between two regions that can be merged increases with increasing angle of eccentricity. (see Sec. 5.2.4).

This peripheral fusion effect simulates the so-called *global effect* (Findlay 1982) in eye movement control: Saccadic eye movements to two nearby objects in the periphery tend to land on

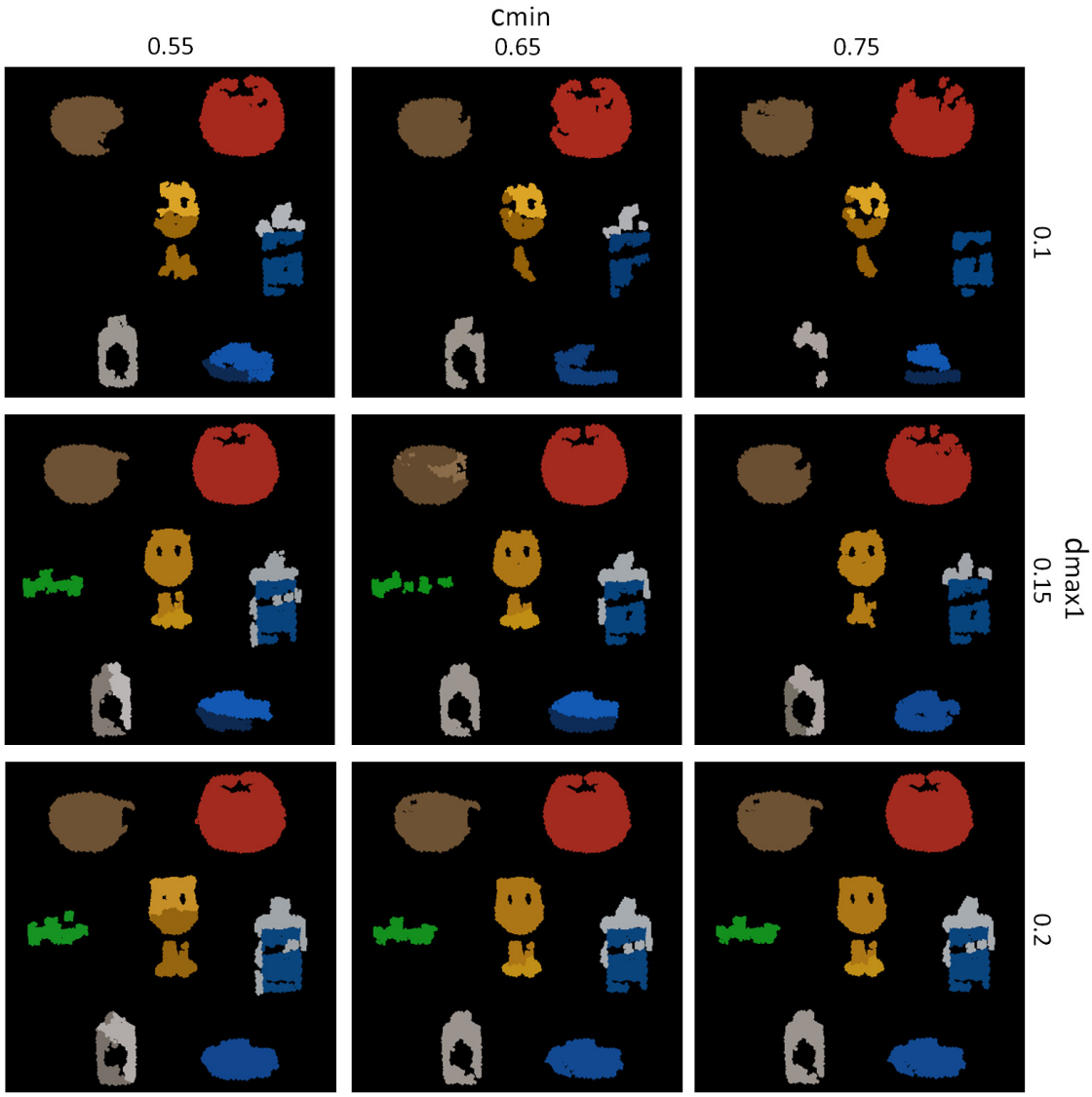

Figure 5.9: *Variation of parameters c_{min} and d_{max1}. A decrease of c_{min} as well as an increase of d_{max1} allows less homogeneous regions to become part of a proto-object. According to this, the combination of $c_{min} = 0.75$ and $d_{max1} = 0.1$ only produces proto-objects for highly homogeneous regions (right top), whereas the combination of $c_{min} = 0.55$ and $d_{max1} = 0.2$ is least sensitive with regard to inhomogeneity (left bottom).*

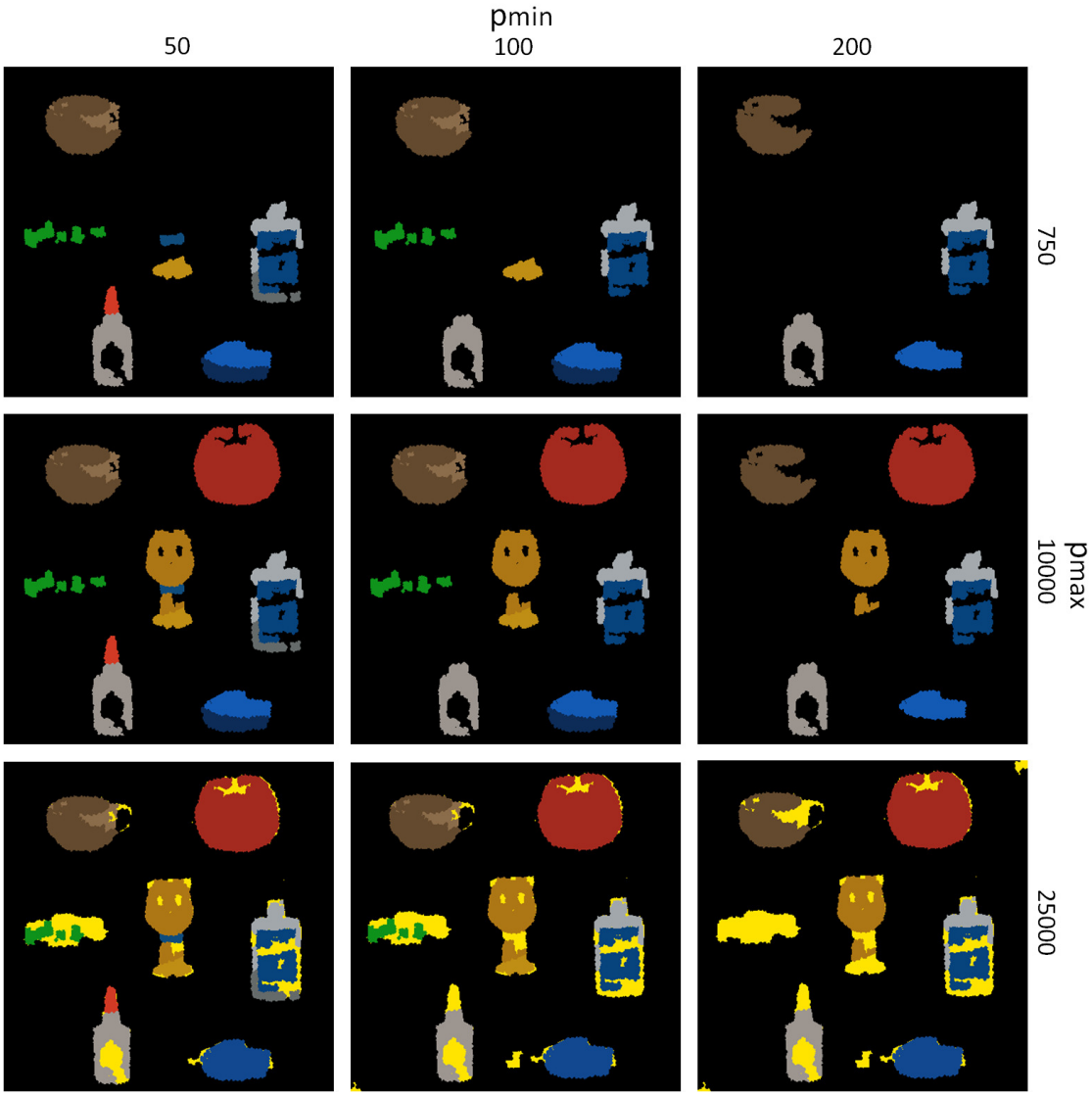

Figure 5.10: *Variation of parameters p_{min} and p_{max}. Both parameters frame the interval regarding the number of pixels a region is allowed to consist of. So the combination of $p_{min} = 50$ and $p_{max} = 25000$ produces the largest number of regions (left bottom), whereas the smallest interval with $p_{min} = 200$ and $p_{max} = 750$ produces the smallest number of regions (right top). In the lower row with $p_{max} = 25000$, the black background itself became a region. For that reason, in these examples the color yellow is used to mark areas in the image that do not belong to any object's region.*

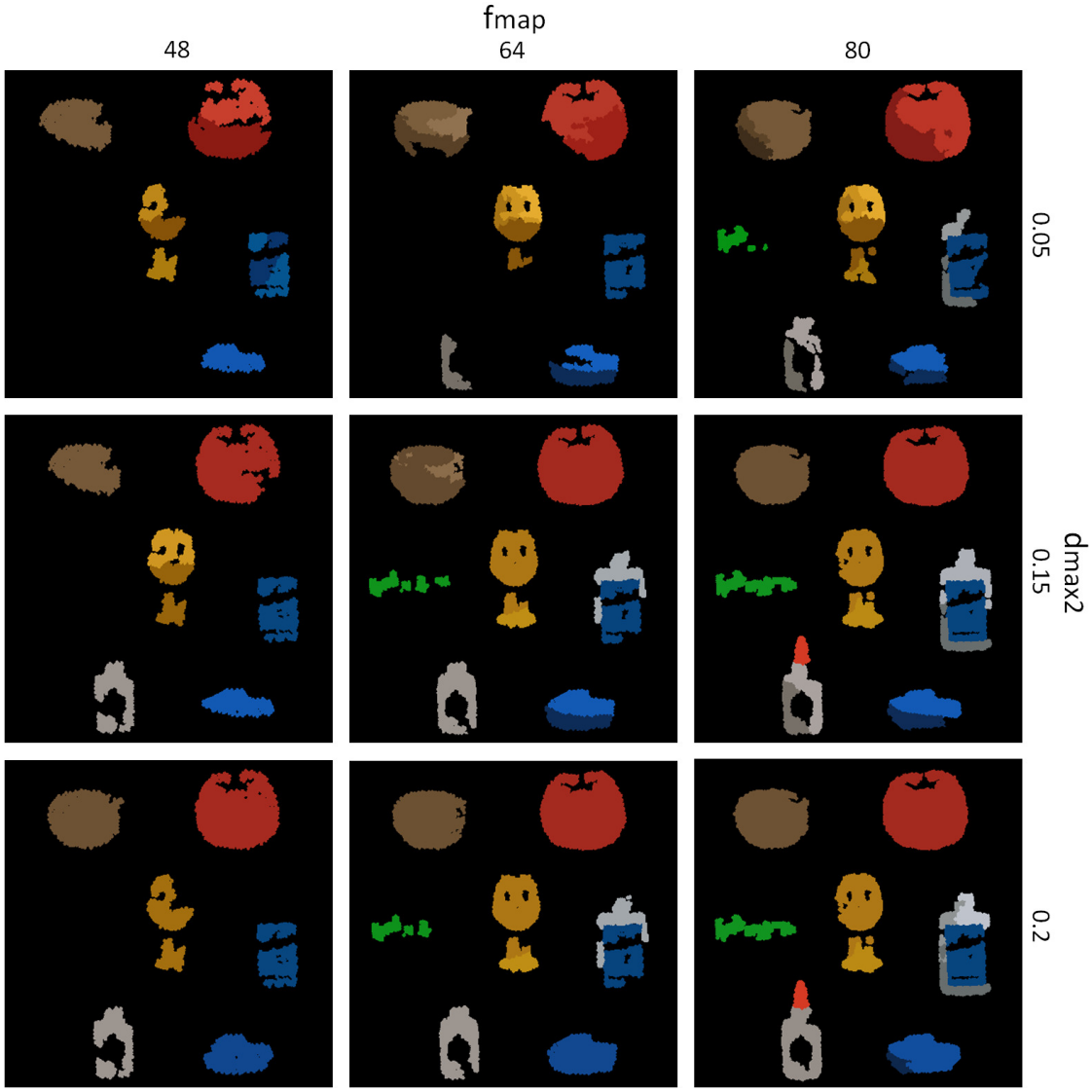

Figure 5.11: *Variation of parameters f_{map} and d_{max2}. f_{map} reflects the overall resolution, so objects can be mapped in greater detail by increasing this parameter. This is because smaller regions consist of more pixels and are thereby more likely to exceed the lower filtering threshold p_{min} (e.g. the red cap of the bottle). d_{max2} defines a merging threshold: Two regions are merged if their distance in the RG/BY/BW-space does not exceed d_{max2};, so an increase of d_{max2} increases the number of merges. An appropriate value significantly improves the mapping of objects by reducing over-segmentation. The upper row with $d_{max2} = 0.05$ shows such an over segmentation.*

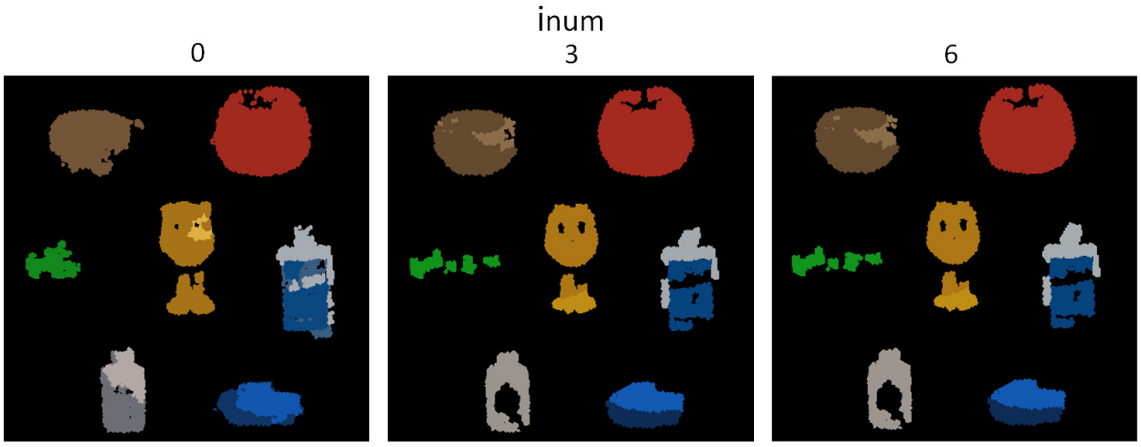

Figure 5.12: *Variation of parameter i_{num}. This parameter reduces noise in the filter computations, so the higher i_{num}, the less noisy pixels influence the result. Left: with $i_{num} = 0$ the algorithm tends to produce (a) proto-objects for non-existing colors (right, bottle) and (b) inaccurate borders between two regions (bottom, blue boat). Center: standard value with $i_{num} = 3$. Right: Higher values, here with $i_{num} = 6$, hardly influence the result anymore.*

the center of gravity of these objects. In the model this corresponds to the gaze landing on a single proto-object that covers two or more natural objects (see e.g. (Wischnewski, Steil, Kehrer, and Schneider 2009)). In human vision the "global effect" appears only under time pressure. In the model the value of parameter f_{map} simulates to what extent the system is under time pressure because high-resolution feature maps/filter layers take longer to process: the greater the time pressure, the lower the f_{map} value and the lower both the pixel density of the inhomogeneous feature map and the filter density of the Gaussian pyramid's layer. Therefore, the probability for the occurrence of fusion depends on both the angle of eccentricity and spatial resolution. This is illustrated in Fig. 5.13.

5.5 Summary

The first step in building proto-objects is to find regions in the feature map that likely represent a natural object. Therefore, the term "region" denotes a set of pixels that belongs to one proto-object. As computations should be fast (see Sec. 3.6), a clustering approach was cho-

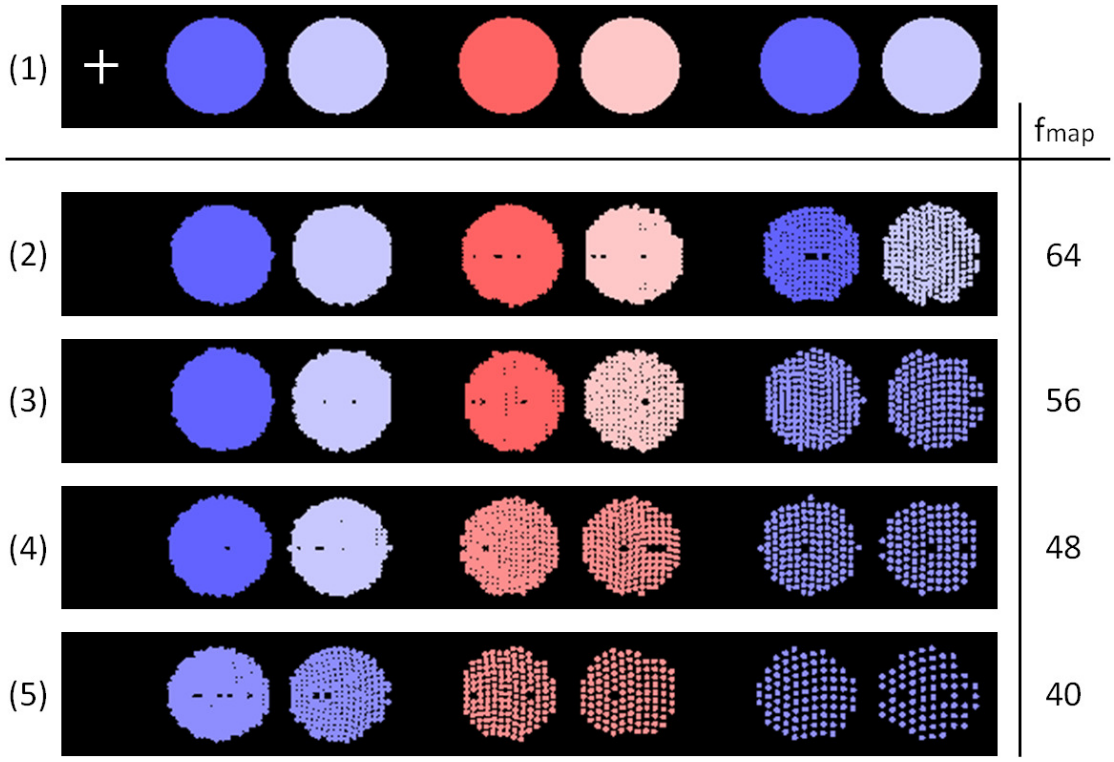

Figure 5.13: *Illustration of the global effect. Depending on resolution, two (or more) nearby objects are merged into one region and are therefore represented by only one proto-object. In this figure, if two objects are merged, they share the same color, which corresponds to the mean color of both objects. In order to allow both objects to be clearly distinguishable by color and to be merged,* d_{max2} *was set to* 0.55*. (1) Input image. The white cross marks the visual center. Three object pairs are shown at different eccentricities. The middle pair has a different color, which allows one to distinguish between a merging within a pair and a merging across pairs. This is because the resulting mean color of both cases strongly differ. (2) With* $f_{map} = 64$*, which is the standard value, each object is represented by one region. (3) If* f_{map} *equals* 56*, the peripherally located pair is merged. (4) With* $f_{map} = 48$*, the same happens to the middle pair. (5) Finally, with* $f_{map} = 40$*, all pairs are merged. As can be seen, merging only occurs within pairs.*

sen that segments homogeneous regions in the three-dimensional color (RG/BY) and intensity (BW) space (Walther, Rutishauser, Koch, and Perona 2005). Forssn published an algorithm that yields an applicable segmentation (Forssén 2004)[1]. Since, however, this algorithm only works with spatial homogeneously arranged input data, as used in standard pixel images, a complete re-implementation with several substantial modifications was done so that the segmentation algorithm is now able to handle the model's spatial, inhomogeneous feature map. Due to the spatial inhomogeneity, the segmentation result strongly differs depending on the angle of eccentricity. A new, subsequent filtering stage then removes proto-objects whose regions are too small (e.g. artefacts) or too big (e.g. parts of the background).

Because clustering is computed based on spatial inhomogeneously structured data, it can lead to an implicit merging of two or more nearby natural objects in the periphery if a low resolution has been chosen (which simulates high time pressure). If the corresponding proto-object serves as the next saccade target, the landing position would be the center of gravity of these natural objects. This property of the model simulates the psychological "global effect" (Findlay 1982).

According to the complexity of natural objects, some of them are mapped by two or more proto-objects if they consist of more than one homogeneous color/intensity region. Although, a subsequent merging stage (see Sec. 9.2) absorbs this effect to a certain extent. On the other hand, if a natural object has no homogeneous color/intensity region, then it is "invisible" to the model, so here the model reaches its limit.

Finally, it could have been shown that the clustering algorithm is robust against parameter variations. This means that small changes in parameters equally produce only small changes in the resulting regions of proto-objects.

[1] `http://www.cs.ubc.ca/~perfo/software/`

Chapter 6

Computation of mid-level features

6.1 Introduction

In order for the model to be effective in finding a certain object in the visual field with regard to a given task, it has to have the capability to sufficiently distinguish objects based on a set of features. There are different levels on which features can be computed. These levels can be roughly classified into low, mid, and high-level features and are defined as follows:

- *Low-level features* come from early vision, like local color or local orientation contrast, and need only slight computational load. Feature values are assigned to pixels, see e.g. (Itti and Koch 2000).
- *Mid-level-features* are based on low-level feature processing and allow a higher level of complexity. Computational load is moderate. Feature values can be assigned to groups of pixels, see e.g. (Wischnewski, Belardinelli, Schneider, and Steil 2010).
- *High-level features* are computed on the basis of all preceding feature levels. This corresponds, e.g. to the level of object recognition and produces a rather high computational load, see e.g. (Kirstein, Wersing, and Körner 2008)

The choice of the model's feature level relies upon two criteria. First, the level has to be sufficient to map object-based features (e.g. size or shape). Second, the high frequency of eye movements in humans yields a limitation in processing capacity and therefore the concept of merely having candidates regarding the next saccade target.

The pixel-based approach of low-level features is insufficient to map object-based features. On the other hand, high-level features produce a high computational load and allow a rather exact classification, which does not fit the concept of having candidates. Thus, the model makes use of mid-level features as they meet both criteria. Such a mid-level feature representation of a natural objects is called a *proto-object*.

The model is able to compute up to 16 different mid-level features for the categories *size*, *orientation*, *shape*, *color*, and *intensity* (see Sec. 6.3). For the feature computations, the spatial inhomogeneity has to be taken into account: More foveally located areas of a proto-object's region have a higher pixel density, corresponding to the spatial structure of the low-level feature map (see Ch. 4). The solution is to *weight each pixel* depending on its angle of eccentricity (see Sec. 6.2).

6.2 Weighted arithmetic mean by means of scaling

Each proto-object is represented by a region R which consists of a set of pixels $p \in R$. Therefore, each pixel p contains the values of 5 low-level features: two values for color (p_{rg}, p_{by}), one for intensity (p_{rg}), and two for the position (p_x, p_y). These low-level feature values serve as basis for the computation of all mid-level features.

For the computation of several mid-level features, the model makes use of an arithmetic mean. This, e.g. applies to the computation of the centroid or the mean color value of a proto-object. But due to the foveal higher density of pixels, more foveally located parts of a proto-object's region affect the result more strongly than more peripherally located parts (see Fig. 6.1).

This effect can be compensated by weighting pixels according to their angle of eccentricity. The decisive criterion to compute a pixel's weight is the relative pixel density at the pixel's angle of eccentricity. The relative pixel density exactly corresponds to the scaling factor s (see Eq. 4.1 and its explanation in Sec. 4.2). Thus, the model uses the scaling factor $s(p)$ of pixel p as the pixel's weight (see Eq. 6.1).

$$s(p) = 1 + e(p) * k = 1 + \sqrt{p_x^2 + p_y^2} * k \tag{6.1}$$

Here $e(p)$ denotes the pixel's angle of eccentricity. The effect is that more foveally located pixels obtain a lower weight then more peripherally located pixels. This is because $s(p)$ increases

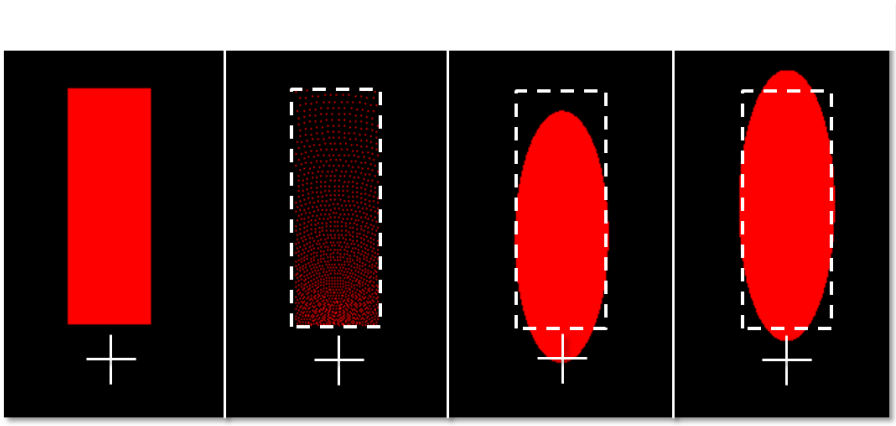

Figure 6.1: *From left to right: (a) Red rectangular object in the input image. (b) The object's region after segmentation. As can be seen, the pixel density decreases with increasing eccentricity. The white dashed frame marks the object's boundary. (c) Elliptical approximation without considering the eccentricity-dependent pixel density. The foveal higher density incorrectly "moves" the ellipse closer to the visual center (marked by the white cross). (d) Correct elliptical approximation by an eccentricity-weighted arithmetic mean.*

with increasing eccentricity. Importantly, the increase of $s(p)$ exactly andproportionally mirrors the decrease of pixel density inversely .

Hence, the exact mean values can be computed by using a *weighted* arithmetic mean. For that, each weight has to be normed by the sum $S(R)$ over all weights that belong to region R (see Eq. 6.2). Now, each non-weighted low-level feature value of pixel p can easily be transferred into a weighted value by the factor $\frac{s(p)}{S(R)}$.

$$S(R) = \sum_{p \in R} s(p) \tag{6.2}$$

In the following section, the computation of mid-level features is explained in detail.

6.3 The mid-level features

6.3.1 Color and intensity

For color and intensity, three feature values are computed: f_{rg} for red-green, f_{by} for blue-yellow, and f_{bw} for black-white (see Eq. 6.3, 6.4 and 6.5). To this end, the model makes use of the weighted arithmetic mean as described above. Each feature value has a range of $[0..1]$.

$$f_{rg} = \frac{1}{S(R)} \sum_{p \in R} p_{rg} s(p) \tag{6.3}$$

$$f_{by} = \frac{1}{S(R)} \sum_{p \in R} p_{by} s(p) \tag{6.4}$$

$$f_{bw} = \frac{1}{S(R)} \sum_{p \in R} p_{bw} s(p) \tag{6.5}$$

6.3.2 Size

The value for size is equivalent to the area of the ellipse that best approximates region R (Forssén 2004). To compute the inertia matrix that serves as the basis for the computation of the ellipse's parameters, the model makes use of the image moments coming from image processing (see Eq. 6.6). Again, the weighted arithmetic mean is used to avoid erroneous results.

$$m_{ab} = \frac{1}{S(R)} \sum_{p \in R} p_x^a p_y^b s(p) \tag{6.6}$$

First, the centroid of all pixels $p \in R$ is computed (see Eq. 6.7).

$$\bar{R_c} = \begin{pmatrix} c_x \\ c_y \end{pmatrix} = \frac{1}{m_{00}} \begin{pmatrix} m_{10} \\ m_{01} \end{pmatrix} \tag{6.7}$$

Afterwards, the model computes the inertia matrix (see Eq. 6.8).

$$\bar{I} = \begin{pmatrix} i_{00} & i_{01} \\ i_{10} & i_{11} \end{pmatrix} = \frac{1}{m_{00}} \begin{pmatrix} m_{20} & m_{11} \\ m_{11} & m_{02} \end{pmatrix} - \bar{R_c}\bar{R_c}^T \tag{6.8}$$

Finally, the area of the ellipse can be computed (see Eq. 6.9).

$$f_{area} = 4\pi\sqrt{i_{00}i_{11} - i_{01}i_{10}} \tag{6.9}$$

The range of f_{area} depends on a couple of parameters, like p_{min} and p_{max} (see Eq. 5.13), and the overall resolution, which in turn depends on parameter f_{map} (see Sec. 4.2) etc. The lower limit, which can be achieved by a foveally located object with $|R| = p_{min}$, is close to zero. The upper limit can be achieved by an object that is most peripherally located with $|R| = p_{max}$.

6.3.3 Orientation

To obtain the proto-object's orientation, the model makes use of the inertia matrix in Sec. 6.3.2. Let $\bar{e_1}$ be the eigenvector that belongs to the greater eigenvalue λ_1 of $\bar{I}$ (see Eq. 6.10). Then the orientation of this eigenvector, which is consistent with the orientation of the larger main principal axis of the ellipse, exactly corresponds to the orientation of the proto-object's ellipse.

$$\bar{e_1} = \begin{pmatrix} e_0 \\ e_1 \end{pmatrix} \tag{6.10}$$

Since the orientation of the eigenvector is ambiguous (α and $\alpha + \pi$), only the smaller value is regarded. Then this value is redoubled to cover the whole range from 0 to 2π (see Eq. 6.11).

$$\alpha = fmod(-\arctan(\frac{e_1}{e_0}) + \pi, \pi) * 2 \tag{6.11}$$

To solve the problem that geometrically similar orientations can show huge differences if their angles are being compared (e.g. $5°$ and $355°$), the model makes use of the position on the unit circle that belongs to the corresponding angle (see Eq. 6.12 and 6.13). This is illustrated in Fig. 6.2.

$$f_{orient_x} = \cos(\alpha) \tag{6.12}$$

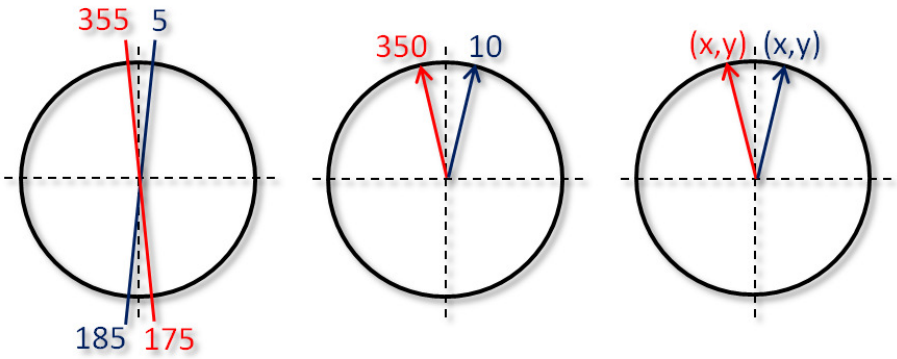

Figure 6.2: *Computation of mid-level orientation features. Left to right: (a) Main principle axes of two proto-objects (red and blue). Angles are ambiguous and, thus, there is no usable measure for feature similarity between two proto-objects. This problem can be solved in two steps. (b) First, only the doubled smaller value is used to obtain the angle of a directed vector, see Eq. 6.11. (c) Second, orientation is described by two values* $(f_{orient_x}, f_{orient_y})$*, which correspond to the intersection between the vector and the unit circle, see Eq. 6.12 and 6.13. Now feature similarity is realized by spatial distance.*

$$f_{orient_y} = \sin(\alpha) \tag{6.13}$$

In the end, two features for orientation are obtained. Compliant with the trigonometric functions, the range is $[-1..1]$.

6.3.4 Shape

In the model, three different kinds of shape features are computed. The first shape feature, f_{axes}, reflects the relation of both main principal axes (see Eq. 6.14). The length of each axis is given by the corresponding eigenvalues λ_1 and λ_2, which come from the inertia matrix (see Eq. 6.8). The range of f_{axes} is $[0..1]$. A zero value denotes a line-like proto-object, whereas a value of one denotes a circle-like proto-object. Fig. 6.3 shows some examples.

$$f_{axes} = \frac{\min(\lambda_1, \lambda_2)}{\max(\lambda_1, \lambda_2)} \tag{6.14}$$

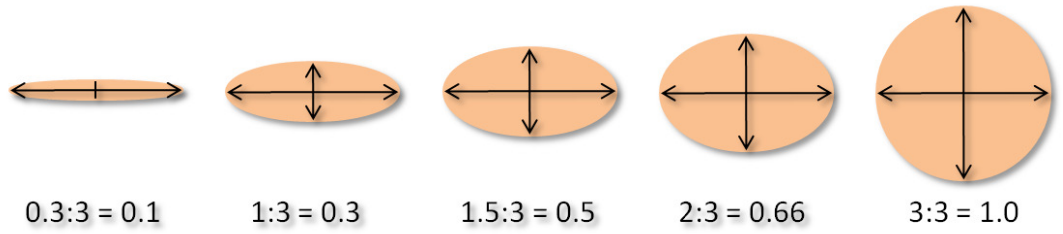

Figure 6.3: *Computation of shape feature f_{axes}. The figure shows the computation for five example proto-objects. The horizontal main principal axes always have a 3-degree visual angle, whereas the size of the vertical main principal axes is varied. Then, the feature values are computed according to Eq. 6.14.*

The second and third shape features completely differ from the previous, as they reflect the local density distribution of pixels $p \in R$. This makes it possible to distinguish natural objects that are mapped by relatively similar elliptical approximations and color/intensity values but appreciably differ in shape.

The second shape feature, f_{ring}, describes how ring-like a proto-object is. Ring-like means that the density of pixels in the center of R is lower than the average. Therefore, the number of pixels within the center has to be counted. The center's area is determined to be $\frac{1}{4}$ of the ellipse's area and to have an identical shape.

In the first step, the ellipse matrix $\bar{A}$ is computed by inverting and scaling the inertia matrix $\bar{I}$ (see Eq. 6.15)(Forssén 2004).

$$\bar{A} = \begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{pmatrix} = \frac{1}{4}\bar{I}^{-1} \tag{6.15}$$

Afterwards, the position of each pixel $p \in R$ relative to the centroid of region R is determined.

$$\bar{d}(p) = \begin{pmatrix} d_0 \\ d_1 \end{pmatrix} = \begin{pmatrix} p_x - c_x \\ p_y - c_y \end{pmatrix} \tag{6.16}$$

Now, the number of pixels lying within the center of R can be counted. Again, the spatial inhomogeneity has to be taken into account.

Figure 6.4: *Computation of shape feature f_{ring}. The more ring-like a proto-object, the higher the feature value.*

$$sum_{sr} = \sum_{p \in R} \begin{cases} s(p), & \text{if } a_{00}d_0^2 + d_1[2a_{01}d_0 + a_{11}d_1] \leq 0.25 \\ 0, & \text{else} \end{cases} \tag{6.17}$$

Finally, f_{ring} can be computed. If the center's density is equal to or higher than the average, f_{ring} equals zero. This means that the region R is not ring-like. But if the center's density is lower than the average, then a value greater than zero will be computed (see Eq. 6.18). The maximum value is 1. This is the case when the number of pixels within the center of R equals zero. Thus, the range of f_{ring} is $[0..1]$. Fig. 6.4 shows different examples.

$$f_{ring} = \begin{cases} -4(\frac{sum_{sr}}{S(R)}) + 1, & \text{if } sum_{sr} < \frac{S(R)}{4} \\ 0, & \text{else} \end{cases} \tag{6.18}$$

The third shape feature, f_{sector}, which is an 8-tupel, reflects the relation of the number of pixels between predefined angle ranges and the whole region of R. Therefore, R is divided into eight sectors (see Fig.6.5).

First, for each sector $n \in \{0, 1, 2, 3, 4, 5, 6, 7\}$ a set of pixels set_n is built that contains all pixels lying in the corresponding sector (see Eq. 6.19 and 6.20).

$$set_n = \{p \in R | f(|atan2(d_1, d_0) - \frac{1}{4}\pi n|) < \frac{1}{8}\pi\} \tag{6.19}$$

$$f(\alpha) = \begin{cases} \alpha + 2\pi, & \text{if } \alpha < -\pi \\ \alpha - 2\pi, & \text{if } \alpha > \pi \\ \alpha, & \text{else} \end{cases} \tag{6.20}$$

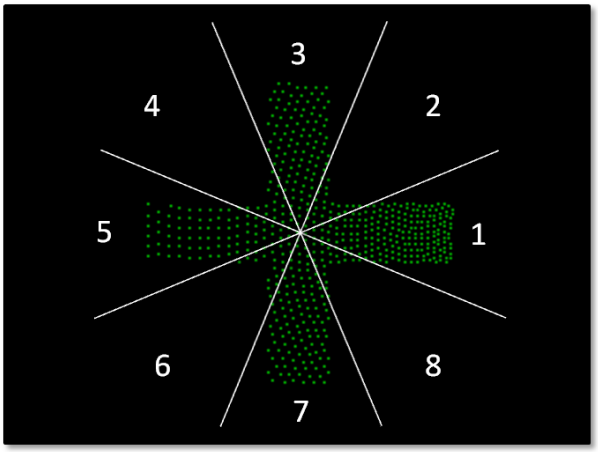

Figure 6.5: *Relative pixel density. Each proto-object is subdivided into eight sectors, where the center equals the proto-object's center of gravity. Then, the number of pixels within each sector is counted (weighted by eccentricity), see Eqs. 6.19 and 6.20. Finally, a feature is assigned to each sector by computing the relation of its number of pixels and the total number of pixels, see Eq. 6.21.*

Afterwards, considering the spatial inhomogeneous pixel density, the eight values for f_{sector} can be computed (see Eq. 6.21).

$$f_{sector_n} = \frac{4}{S(R)} \sum_{p \in set_n} s(p) \tag{6.21}$$

By multiplying with 4, f_{sector_n} equals 0.5 if the sum for sector n is exactly $\frac{1}{8}$ of $S(R)$. If set_n is empty, f_{sector_n} equals zero. A value of 1.0 is obtained if the sum for sector n is exactly $\frac{1}{4}$ of $S(R)$. Fig. 6.6 provides an example of how the model can make use of this feature to distinguish two natural objects having similar elliptical approximations and color/intensity values.

6.4 Summary

In this chapter, the computations of mid-level features was comprehensively illustrated. Altogether, the model can compute up to 16 different mid-level features for each proto-object. A core element of the mid-level-feature representation, as most features make use of it, is an

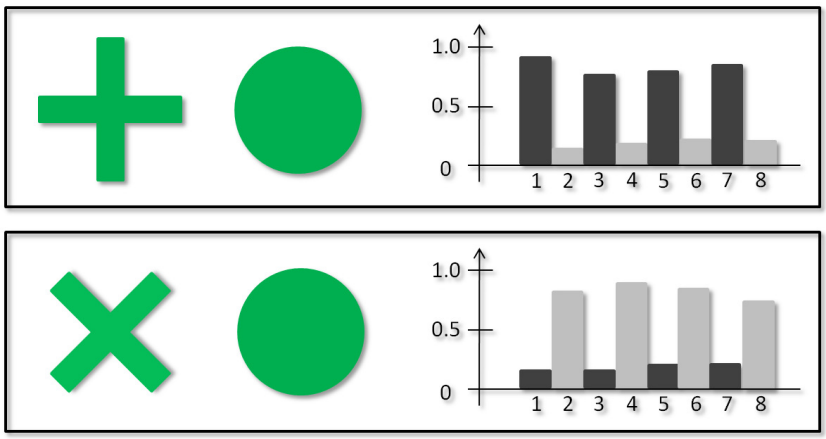

Figure 6.6: *The model's ability to distinguish objects by local pixel density. Two different objects, a "+"-like and an "x"-like (left) yield nearly identical proto-object representations (middle) with regard to color, intensity, size, orientation, and both shape features* f_{axis} *and* f_{ring}*. Nevertheless, the model is able to distinguish both objects by computing sector-wise local pixel densities (see Fig. 6.5). The result can be seen in the histograms (right), where for each sector* n *its corresponding* f_{sector_n} *value is shown.*

elliptical approximation of the proto-objects' shape. This elliptical approximation is gained by computing an inertia matrix on the basis of the set of pixels that belongs to a proto-object.

In the following, all mid-level features and their meaning are listed. The number in brackets denotes the total number of features that belong to the corresponding category.

- *Size(1)*: The size of a proto-object equals the area (in square visual angle) of the computed ellipse.
- *Orientation(2)*: The orientation features reflect the orientation of the ellipse's main principal axis.
- *Color and intensity(3)*: Two color (RG/BY) and one intensity (BW) feature represent the corresponding average value over all pixels that belong to a proto-object.
- *Shape(10)*: There are three different types of shape features. The first maps the relationship between both ellipse's principal axes. This is a measure of whether the ellipse is more circle-like or line-like. The second describes how "ring-like" a proto-object is. Ring-like means that the inner pixel density is lower than the outer. The last type consists

of eight features. The proto-object is, like a pizza, divided into eight sectors. Each feature reflects the relationship between the pixel density of the whole proto-object and its sector. The shape features that make use of pixel density make it possible to distinguish natural objects even if they have very similar values in all other feature dimensions.

Chapter 7

Learning the mid-level feature representations of natural objects

7.1 Introduction

The result of the model's previous stages is a set of proto-objects, each consisting of a set of mid-level features. When searching for a certain natural object, an associated mid-level representation (target template) of this object is needed in order to determine the *feature similarity* between it and each computed proto-object. Then, the proto-object with the highest degree of similarity is most likely to become the next saccade target.

This approach was already implemented by (Wischnewski, Belardinelli, Schneider, and Steil 2010) (see Sec. 3.4 for a detailed description and model comparison) but without the possibility to learn a set of natural objects from examples. Other models provide such an object database of target templates, but only for a foveal representation (Zelinsky 2008) or without taking spatial inhomogeneity into account at all (Elazary and Itti 2010). Findings, however, have shown that humans learn to associate foveal and peripheral representations of natural objects (Cox, Meier, Oertelt, and DiCarlo 2005), so there exists more than one template for each object. Additionally, in this model one natural object can be mapped by more than one proto-object if it consists of various homogeneous color/intensity regions.

So, when aiming at learning proto-object representations of natural objects, the problem has to be solved that one natural object cannot be represented by a single proto-object and, therefore, a

single set of mid-level features. For that, the model makes use of a feed-forward neural network for classification (see Sec. 7.2) in order to realize the computation of feature similarity: For each learned natural object, it computes the conditional probability that a proto-object's set of mid-level features represents it.

A further important point is the classification performance. By specifically restricting the learning, the neural network approach makes it possible to adjust performance based on the guideline that both an almost perfect (i. e. equaling object recognition) as well as a too poor classification are unwanted (see Sec. 7.3).

7.2 A neural network approach for classification

The model makes use of a feed-forward neural network for classification [1]. A feed-forward neural network consists of one input layer, several hidden layers, and one output layer (see Fig. 7.1). The input values of the network are the mid-level feature values of one proto-object. Thus, the dimensionality M of the input layer complies with the number of used mid-level features. The maximum value of M is 16, as this is the maximum number of mid-level features the model is able to compute. The number of hidden layers as well as their dimensionality H_1, H_2 etc. depends on the network's desired capacity. The higher the number of layers and nodes within these layers, the higher the capacity and, therefore, the network's ability to discriminate natural objects (see Sec. 7.3). The dimensionality C of the output layer complies with the number of natural objects the model is able to learn. Each node c, with $0 \leq c < C$, of the output layer represents one object class that, in turn, represents one natural object. The output layer's values serve as input for the computation of conditional probability values $p(c|F_i)$ (see Eq. 7.1).

$$p(c|F_i) = \frac{\exp(net(F_i, c))}{\sum_{k=1}^{C} \exp(net(F_i, k))} \tag{7.1}$$

Here $net(F_i, c)$ denotes the value of output node c, given a set of mid-level features F that belongs to proto-object with index i. The probability values are computed by using the so-called *softmax activation function,* which ensures that the total sum of all probabilities equals

[1] The network was implemented by using the C++ shark library (`http://shark-project.sourceforge.net/index.html`)

1 (see Eq. 7.2).

$$\sum_{c=1}^{C} \frac{\exp(net(F_i, c))}{\sum_{k=1}^{C} \exp(net(F_i, k))} = 1 \tag{7.2}$$

$p(c|F_i)$, with $0 \leq p(c|F_i) \leq 1$, is equivalent to the *conditional probability* that the proto-object with index i represents the natural object of class c. The network's design enables the model to compute the probability value $p(c|F_i)$ for each combination of class c and proto-object i.

To ensure that the network learns probability values, the model uses the *cross entropy error measure* (see Eq. 7.3 and 7.4) (Bishop 1996).

$$E = -\sum_{i=1}^{X} \sum_{c=1}^{C} \left[\delta_{c(i)c} \ln \left(\frac{\exp(net(F_i, c))}{\sum_{k=1}^{C} \exp(net(F_i, k))} \right) \right] \tag{7.3}$$

$$\delta_{c(i)c} = \begin{cases} 1, & \text{if } c(i) = c \\ 0, & \text{else} \end{cases} \tag{7.4}$$

X reflects the number of examples that are used to train the network (see Sec. 7.3) and $c(i)$ equals the class example i, with $0 \leq i < X$, which belongs to X. The use of the *Kronecker delta* (see Eq. 7.4) leads to a total error which only includes one error value per example. This value corresponds to the probability value of a correct classification. If each example was perfectly classified, this means $p(c|F_i) = 1$ if c is the correct class and thus $ln(p(c|F_i)) = ln(1) = 0$, then E would reach its global minimum of 0. As can be seen, the argument of the natural logarithm function is the output of the softmax activation function. A condition to apply the softmax activation function for classification is the implementation of the *linear activation function* for the output layer. For the preceding network's layers, the model uses the standard *Fermi activation function*.

7.3 The training stage

To train the network, at first, a set of examples has to be created. Each example consists of $M + 1$ values: M values for the mid-level features and one value that represents the class the

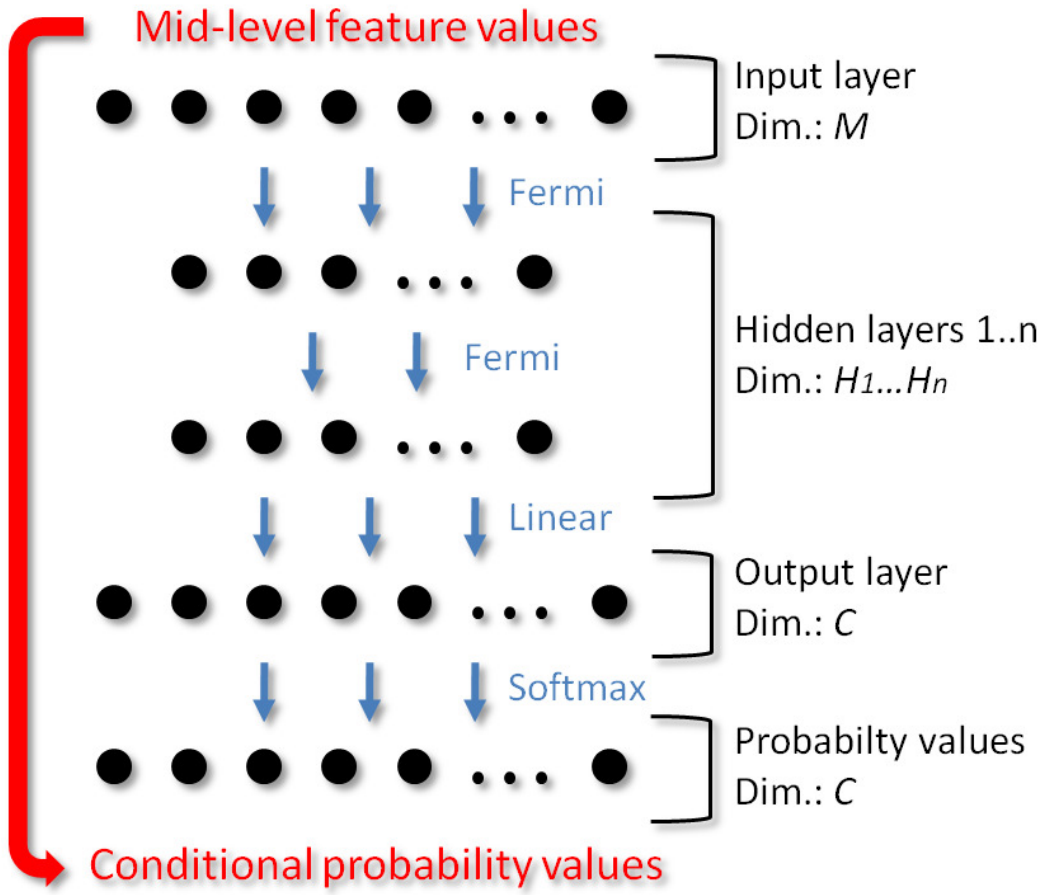

Figure 7.1: *Network scheme. M mid-level feature values of one proto-object are transformed into C conditional probability values; that is, for each object class c with $0 \leq c < C$, the probability is computed that the proto-object belongs to that class. The number of hidden layers and their dimensionality depend on the desired network performance. The activation functions were chosen to meet the aim of having probability values; that is, the sum over all values has to equal 1. Between any subsequent layers L_1 and L_2, each neuron of L_1 is connected with each neuron of L_2.*

proto-object belongs to. A natural object can yield more than one example if it is mapped by multiple proto-objects. The creation of examples takes place in four steps.

(1) First, a set of object images that should be learned has to be specified. The model is able to handle even a large number of natural objects, e.g., a set of 100 objects of the COIL-100 [2]. The total number of objects C determines the number of nodes, and thus classes, of the network's output layer.

(2) Afterwards, the (sub)set of mid-level features to be used has to be determined. One reason not to use the highest possible number of features is to reduce the computational load of both training and application of the network. This concerns the trade-off between model performance and speed. Another reason is related to the properties of the object images. If, e.g., the model is fed with black and white images, there is no reason for using color features. The total number of mid-level features M determines the number of nodes of the network's input layer.

(3) As a next step, a position grid has to be defined that determines the positions within the visual field at which each object is presented, so that for each single object, the model learns a set of position-dependent mid-level feature representations (see Fig. 7.2). The denser the position grid, the higher the accuracy of object classification, but, (due to the increase of examples) the higher the computational load that is needed to train the network.

(4) Then, for each position, each object image passes the first three stages of the model: computation of the feature map, proto-object segmentation, and mid-level feature computation. Therefore, the parameters of all these stages have to be determined beforehand.

After the examples have been created, they are used to train the network. The aim of the training is to minimize error E. The training method is called *supervised learning* because, for each example, the correct classification is known. So, if c is the correct classification of proto-object i, the value of $p(c|F_i)$ is increased by modifying the network's weights by means of the *back propagation* learning method. Within each learning step, the model first adds up the weight modifications for all examples. Then, this sum is applied to the network. This procedure is called *batch learning*. The training is subdivided into two steps.

(1) First, the network's weights are initialized randomly within a given range of $[-r..r]$. Furthermore, the β value of the Fermi function has to be determined.

(2) Then, the model's performance has to be determined. The measure of performance used

[2] (`http://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php`)

here is called *target-distractor discriminability* or just μ_α. The lower μ_α, the better the network can distinguish targets (objects the system searches for) from distractors (non-relevant objects). A comprehensive description of how to compute the μ_α values can be found in Sec. 10.2.

The value of μ_α can be approximately determined by choosing the number of hidden layers, the number of nodes within these layers, and the number of learning steps. If the network is built powerfully enough, then a μ_α value of nearly 0, that is, a nearly perfect classification would be possible for smaller sets of natural objects. But this is not desirable for two reasons. First, such a high value would be the result of over-fitting. This means that the network had perfectly learned the examples and therefore would not be sufficiently capable of correctly classifying objects that are shown at other positions within the visual field (see point (3) above). Second, this would contradict the concept of having only candidates for the next saccade and not doing object recognition. If the probability values are on average too high, then in most cases the system would find the target object with the first saccade, which is in general not the case in primates. Thus, a certain degree of faultiness is wanted, which can be realized by an appropriate μ_α value.

A biologically motivated solution to avoid the ability to classify perfectly is to limit the system's processing capacity. Following this approach, the number of hidden layers and the number of nodes within each hidden layer are so limited that the desired μ_α value is obtained after the network has been converged. This means that further learning steps do no longer significantly improve the network's performance.

In sum, the neural network assigns to each proto-object of the input stream a set of C conditional probability values. The c-th value, with $0 \leq c < C$, reflects the probability that the proto-object's mid-level features represent object class c. While it is possible to present new object examples to the network at any time, the number of learned objects cannot be dynamically increased. This would require complete new learning of all objects. The same also applies to changes in the parameters.

In general, *superordinate categories* (e.g. 'animal') cannot be mapped by the model (see (Rosch, Mervis, Gray, Johnson, and Boyes-Braem 1976; Palmer 1999)). The approach of classification by feature similarity better fits the notion of *basic-level categories* (e.g. 'bird') or even *subordinate categories* (e.g. 'a blackbird').

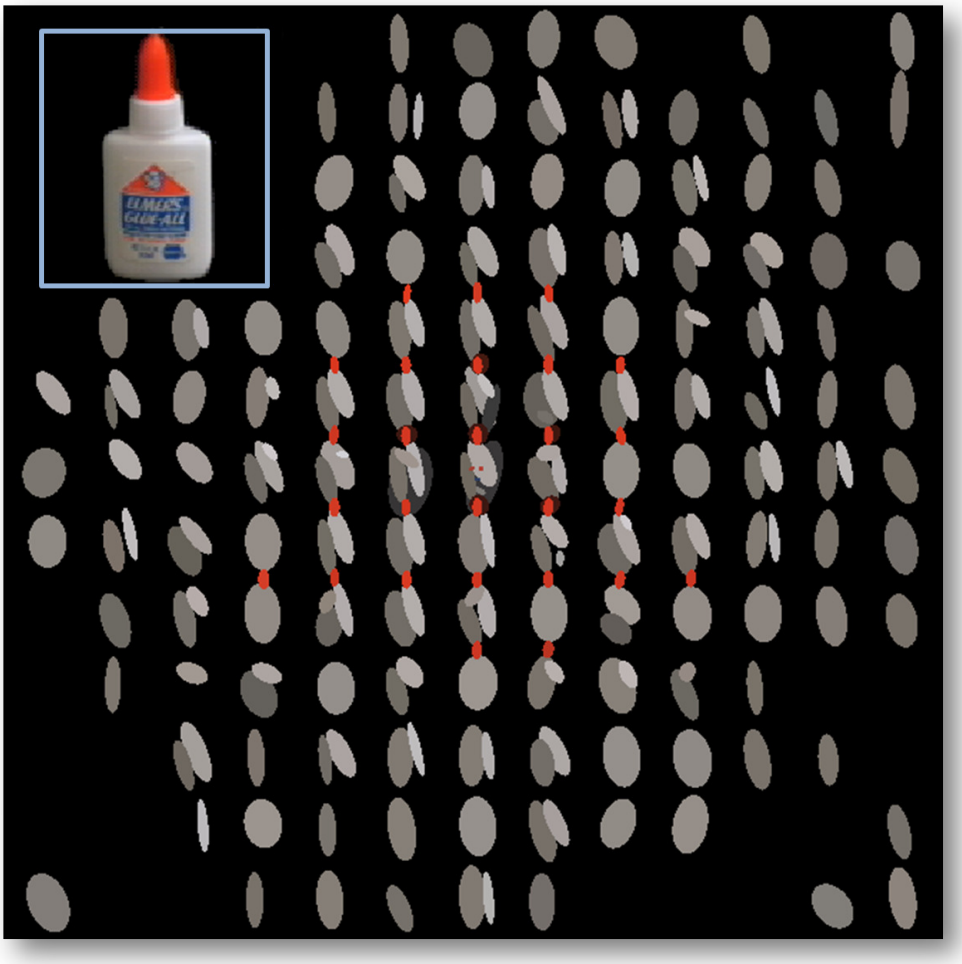

Figure 7.2: *Learning mid-level feature representations depending on eccentricity. The image shows eccentricity-dependent elliptical proto-object representations of one object from the COIL (shown in the blue frame). In the image's center, which corresponds to the foveal region, the proto-object representation is most detailed. With increasing eccentricity, smaller parts of the object disappear, e.g., the proto-object that represents the small red cap. Finally, the object itself disappears. Importantly, the neural network has to learn that all these proto-object representations belong to one single natural object.*

7.4 Summary

In this chapter, it was shown how proto-object representations can be learned on the basis of mid-level features. To do this, the problem has to be solved that one natural object cannot be represented by a single proto-object and thus a single set of mid-level features. There are two reasons for this:

- Due to the application of spatial inhomogeneity, the mid-level feature representation of a natural object strongly depends on the object's position within the visual field.
- Some natural objects are mapped by two or more proto-objects. The corresponding mid-level feature representations can differ significantly, e.g., if one part of the natural object is blue and rather line-like and another part is red and rather circle like.

The solution presented here is to make use of a feed-forward neural network for classification. The network consists of up to 16 input dimensions, one for each mid-level feature used. The number of output dimensions equals the number of natural objects and thus the number of classes that should be learned. Each output node denotes the conditional probability that a given proto-object belongs to the class represented by the node. Within the learning stage, a training set is first built from examples. This means that, for each natural object, the model computes the mid-level features for different locations within the visual field. A perfect learning of the examples, where the probability of correct classification equals nearly one, is not desirable because

- this would be the result of over-fitting. This means that the network has perfectly learned the examples. Then the network cannot generalize, which likely yields erroneous results for proto-objects that are not identical to the learned examples.
- this would contradict the concept of having candidates instead of doing object recognition. The consequence would be that the first saccade would always land on the target, which is not the case in primates.

A biologically motivated approach of restricting the network's ability to perfectly classify is to limit its capacity. Thus, the number of hidden layers and nodes within these layers is chosen so that a desired classification performance is obtained.

Using this method, several sets of images have been successfully learned. It could be shown that this learning approach not only works for simple objects or small sets of images but also for a high number of everyday objects.

Chapter 8

Object-based priority by means of TVA

8.1 Introduction

The neural network approach presented in the previous chapter makes it possible to compute the mid-level *feature similarity* for each combination of learned object class and perceived proto-objects. This is a purely *bottom-up* process since it is not important for these computations which natural object the system currently searches for. So, an additional *top-down weighting mechanism* is needed to decide, on the basis of the network's outcome, which of the proto-objects in the visual input stream will be the next saccade target.

An appropriate feature-based weighting is provided by the TVA weight equation (Bundesen 1990), see (Bundesen, Habekost, and Kyllingsbæk 2011) for an actual overview. This equation assigns an *attentional weight* to each proto-object in the visual input stream (see Sec. 2.7 and 8.2). The higher the weight, the more likely a proto-object serves as the next saccade target.

Apart from some examples, TVA makes no statement about what features the visual system uses and, moreover, how feature similarity is then concretely computed. The first implementation of the TVA weight equation (Wischnewski, Belardinelli, Schneider, and Steil 2010) (see Sec. 3.4) is exclusively based on mid-level features, and feature similarity was computed by using a task-defined Gaussian for each feature dimension: The closer the distance to the Gaussian's mean, the higher the similarity. But this does not match the classification approach. On this account, TVA features are understood as object classes for the thesis' model. So, a feature-based weighting now equals a class-based weighting. In doing so, a task definition can

be directly realized on the level of proto-objects by assigning an appropriate *pertinence value* to each class. This reinterpretation of TVA features leads to the term of a *modified TVA weight equation* (see Sec. 8.2).

The more peripherally an object is located, the less the probability that it serves as the next saccade target. This is called the *proximity effect* (see Sec. 2.7). In order to model this effect, a *homogeneity factor* is used that modifies the TVA-based attentional weights depending on eccentricity. So, more foveally located proto-objects obtain a relatively higher attentional weight (see Sec. 8.3).

8.2 The modified TVA weight equation

In the original TVA weight equation, an attentional weight $w(o)$ of proto-object o is computed as the sum over all features $f \in F$ (see Eq. 8.1).

$$w(o) = \sum_{f \in F} \eta(o, f)\pi_f \tag{8.1}$$

For each feature the η function reflects the bottom-up *sensory evidence*, i.e., to what degree "object o has feature f" (Bundesen 1990). If, e.g., f signifies the feature "color red", then $\eta(o, f)$ is highest if proto-object o has exactly feature f, which is the case if the natural object that is represented by proto-object o is red. The top-down pertinence values π_f are determined by the search task: High pertinence values are assigned to the desired features, whereas the remaining features obtain low or zero values. If the system searches for red objects, then the pertinence value of the feature "color red" would obtain a high value, whereas the pertinence values of all other color features, like "color green" etc., would obtain a low value. As a result, only those proto-objects can achieve a high attentional weight that have at least one feature with a high pertinence value.

The first computational implementation of the TVA weight equation realized the η function by a Gaussian distribution (Wischnewski, Belardinelli, Schneider, and Steil 2010). A set of search-relevant mid-level features is specified, where each feature is defined by mean and variance. The mean determines the value at which the η function is highest, whereas the variance determines the accuracy of the task: The higher the variance, the lower the accuracy. The pertinence values reflect the relative importance of features, e.g., whether the correct color is more

important than the correct size. One disadvantage of this approach is that the means, variances, and pertinence values are defined by hand. Additionally, the model has problems handling the case of one natural object being represented by more than one proto-object. If, e.g., a natural object consists of a big green and a small red part, which both are represented by different proto-objects, then the model has to leave out one proto-object or generate four target features: small, big, red, and green. But this yields the result that also small green and big red (parts of) objects would obtain the same high attentional weight as the target object. Another problematic point is the eccentricity dependency of the segmentation (see Ch. 5) and thus also of the mid-level feature values. So, e.g., nearby and similar-colored parts of natural objects tend to fuse in the periphery ("global effect", see Sec. 5.4). That is, the same natural object, if foveally located, can possibly be represented by more and smaller proto-objects than if located in the periphery. Even small changes in the angle of eccentricity can yield noticeable differences between the mid-level feature values (see Fig. 7.2 in Sec. 7.3). These examples show that it would be necessary to massively increase the number of features, so that all relevant target representations would be part of the search-relevant set of mid-level features. Therefore, there would be multiple features and thus multiple means for each feature dimension (size, orientation etc.). The consequence would be an immense increase of attentional weights for distractors, as in the example where one natural object is represented by two proto-objects. This is because many distractor feature values would equal at least one of the target feature values in one or more feature dimensions, e.g., a distractor's size value equals one of ten possible target values etc.

The problem of multiple mid-level feature representations as described above has been solved by the introduction of a neural network approach. Additionally, no mean or variance values have to be defined by hand, as they are encoded in the network's weights by learning from examples (see Ch. 7). But to make use of the network's output values, the terms of the TVA weight equation have to be reinterpreted. A TVA feature is now understood as an object class. Accordingly, the η function "object o has feature f" is replaced by the conditional probability for object class c given proto-object o's mid-level feature values. This corresponds to the sensory evidence that proto-object o belongs to object class c. Table 8.1 shows a comparison of all three computational approaches concerning the η function. Furthermore, the pertinence values π_c now also refer to the object classes. As a result, in the reinterpreted TVA weight equation, an attentional weight $w(o)$ of proto-object o is computed as the sum over all object classes c, with $0 \leq c < C$ (see Eq. 8.2), where o_F denotes the set of mid-level features of o.

Model	Formalization	Interpretation	Implementation
TVA, Bundesen 1990	$\eta(o, f)$	degree of "proto-object o has features f"	—
Wischnewski et al. 2010	$\eta(o, f)$	degree of "proto-object o has features f"	Gaussian distribution
thesis' model	$p(c\|o_F)$	probability of "proto-object o belongs to object class c"	neural network for classification

Table 8.1: *Formalization, interpretation, and implementation of the η function in different models.*

$$w(o) = \sum_{0 \leq c < C} p(c|o_F)\pi_c \tag{8.2}$$

Here C denotes the total number of object classes. The π_c values determine the model's search task. If the system searches for one or more classes (e.g., find any pen), these classes get a pertinence value of $\pi_c > 0$. Different values denote a pertinence order, e.g., find any pen but preferably the red one. The pertinence values of all other classes are set to 0. In this variant, pertinence values also have to be defined by hand if the task comprises more than one object class. To achieve a high attentional weight, a proto-object has to have high sensory evidence regarding the object class the system searches for.

8.3 The proximity effect: eccentricity-dependent weight modification

According to the biological model, the pixel density in the feature map decreases with increasing angle of eccentricity. As a result, the same proto-object is foveally mapped by a higher number of pixels than peripherally (see Fig. 5.7). If these pixels are understood as a set of excitatory neurons, then sensory evidence not only depends on feature similarity but also on the number of pixels a proto-object consists of. The higher the number of pixels, the higher the sensory evidence.

The model implements this eccentricity-dependent weight modification by a multiplicative

scaling of the η function. The scaling is realized by an inhomogeneity factor $i(o_p)$ where o_p equals the number pixels that proto-object o consists of. As the feature similarity $p(c|o_F)$ and the inhomogeneity factor $i(o_p)$ can be computed separately and, additionally, $i(o_p)$ does not depend on object class c, $i(o_p)$ can be extracted from the summation (see Eq. 8.3).

$$w(o) = \sum_{0 \leq c < C} i(o_p) p(c|o_F) \pi_c = i(o_p) \sum_{0 \leq c < C} p(c|o_F) \pi_c \tag{8.3}$$

From the segmentation stage (see Sec. 5.2.5), the domain of $i(o_p)$ is known: Parameter p_{min} specifies the smallest possible and p_{max} the highest possible value of o_p. The approach for building $i(o_p)$ is that p_{max} should give maximal output with $i(p_{max}) = 1$. The smallest possible function value, which is obtained by $i(p_{min})$, is determined by parameter i_x. That is, the value of $i(p_{min})$ is i_x times lower than the value of $i(p_{max})$ (see Eq. 8.4).

$$i(p_{min}) i_x = i(p_{max}) = 1 \tag{8.4}$$

Son i_x determines the strength of the proximity effect: The higher i_x, the more strongly $i(o_p)$ influences the attentional weights. The next step in building $i(o_p)$ is choosing the map $o_p \mapsto i(o_p)$, which was chosen to be linear, in compliance with Eq. 4.1. Given the linearity and $i(p_{max}) = 1$, the slope m (see Eq. 8.5) and thus the function values of $i(o_p)$ (see Eq. 8.6) can be computed. The standard value of i_x is 2.0. Fig. 8.1 illustrates the map.

$$m = \frac{1 - \frac{1}{i_x}}{p_{max} - p_{min}} \tag{8.5}$$

$$i(o_p) = o_p m + (1 - p_{max} m) \tag{8.6}$$

The range of $w(o)$ in Eq. 8.3 equals $0 \leq w(o) \leq 1$ as both functions, $w(o)$ in Eq. 8.2 and $i(o_p)$ in Eq. 8.6, have an upper limit of 1 and a lower limit of 0.

8.4 Summary

In this chapter, it was shown how the model uses the *TVA weight equation* to assign an attentional weight to each proto-object within the sensory input. The TVA weight equation combines

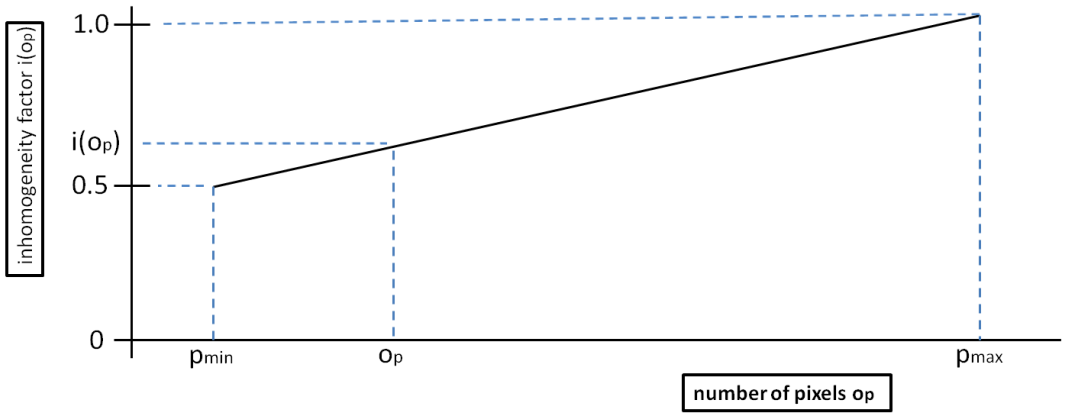

Figure 8.1: *Inhomogeneity factor depending on the number of pixels o_p, with $p_{min} \leq o_p \leq p_{max}$, that a proto-object consists of. For $o_p = p_{max}$, $i(o_p)$ always equals 1. The lowest possible value for $i(o_p)$, which is the result of $o_p = p_{min}$, is determined by factor i_x, as $i(p_{min})i_x = i(p_{max}) = 1$. The figure shows the case of the standard value, where $i_x = 2.0$, so that $i(p_{min}) = 0.5$. If, e.g., i_x equals 4.0, then $i(p_{min})$ would be 0.25. In general: The greater i_x, the steeper the slope and thus the stronger the influence of eccentricity. For a concrete proto-object, the inhomogeneity factor equals $i(o_p)$, as illustrated in the figure.*

bottom-up *sensory evidence* and top-down *pertinence*. In this model implementation, sensory evidence denotes the conditional probability that a proto-object belongs to a certain object class where each class represents one learned natural object (see Ch. 7). Because sensory evidence is computed by a neural network, the probability value of proto-object o belonging to class c is formalized as $p(c|o_F)$, where o_F signifies the proto-objects' set of mid-level features. Additionally, to define a search task, one pertinence value is assigned to each class. If the systems searches for one or more classes (e.g. find any pen), these classes get a pertinence value of $\pi_c > 0$. Different values denote a pertinence order, e.g., find any pen but preferably the red one. The pertinence values of all other classes are set to 0.

Having the values of pertinence and sensory evidence, the model is able to compute the attentional weights. According to the TVA weight equation, a sum is built for each proto-object over all classes, where the sensory evidence that the proto-object belongs to a class is multiplied with the class's pertinence (see Eq. 8.2). Thus, only those proto-objects that obtain high sensory evidence regarding the class the system searches for get a high attentional weight.

A further part of the chapter addresses the *inhomogeneity factor*. The idea is that the strength

of sensory evidence additionally depends on the number of pixels that belong to a proto-object. As these pixels are understood to simulate a set of excitatory neurons, an increase of number of pixels should also yield an increase of the sensory evidence's strength. From this it follows that a natural object gets higher sensory evidence and thus a higher attentional weight if it is presented more foveally because, due to the spatial inhomogeneity, it is represented by more pixels. The general effect is that more peripherally located objects are less likely to be chosen as the next saccade target. The inhomogeneity factor was realized as a function $i(o_p)$, whose output increases with increasing number of pixels o_p that belong to object o. Furthermore, the TVA weight equation was extended so that the original attentional weights are multiplied by $i(o_p)$ (see Eq. 8.3).

Chapter 9

The priority-driven saccade

9.1 Introduction

It is the nature of proto-objects to only coarsely map natural objects of the visual environment. As a result, some natural objects are mapped by more than one proto-object (see Sec. 5.2.3). Although this is unproblematic with regard to attentional weights, since the classification approach is applied (see Ch. 7), it strongly affects the concrete landing position of a saccade. Empirical findings have shown that humans mostly saccade close to an object's center (Foulsham and Underwood 2009; Nuthmann and Henderson 2010) (see Sec. 2.7 and 9.6). But if a natural object is divided into several proto-objects, then so far it is impossible for the model to even approximately know where this center is. A way to significantly reduce this uncertainty is to *merge* those proto-objects that likely belong to the same natural object in the visual input stream (see Sec. 9.2). Then, the system can saccade to the center of merged proto-objects, which more likely represents the center of the corresponding natural object.

After merging, as suggested in the neural interpretation of TVA (NTVA) (Bundesen, Habekost, and Kyllingsbæk 2005), proto-objects are stored along with their weights in a retinotopic *attention priority map* (APM) (see Sec. 9.3), so their relative position within this map corresponds to their relative position in the visual field. The term *priority*, in contrast to *saliency*, implies that attentional weights are built on the basis of a given task (see Sec. 2.4).

If the selection of saccade targets only depends on the proto-objects' attentional weights, then the system would likely produce so-called *saccadic oscillations*: The system repeatedly re-

fixates objects that have already been identified as not being the search target instead of saccading to candidates that have not yet been fixated. In the worst case, a *saccadic standstill*, the currently fixated object again becomes the winning object. To avoid saccadic oscillations and saccadic standstills, the model implements a simple *inhibition of return* (IOR) (Klein 2000) mechanism, which reduces the attentional weights at the last fixated positions (see Sec. 9.4) and thus for the last fixated objects. These positions are stored in an IOR map along with the strength of inhibition. With each saccade, the inhibition decreases so that the last fixated object is inhibited most strongly and re-fixations are possible after a few saccades.

Having applied IOR to the APM, the winning proto-object is selected by an object-based *winner-takes-all* (WTA) mechanism, see e.g. (Bundesen and Habekost 2008) (see Sec. 9.5): The probability of a proto-object becoming the next saccade target increases with increasing attentional weight (Carbone and Schneider 2010) (see Sec. 2.7). After selection, the saccade can be executed. The concrete landing position is modeled as a 2D-Gaussian, whose mean equals the target object's center (Nuthmann and Henderson 2010) (see Sec. 2.7 and 9.6). At this point, even some object-based models fail, e.g. (Elazary and Itti 2010).

9.2 Merging of proto-objects

9.2.1 The identity value

The model aims at saccading to the center of the natural object, which is represented by the proto-object with the highest attentional weight $w(o)$. This model property can best be realized by choosing the centroid of the corresponding proto-object (see Sec. 6.3.2) as target location. But often, a natural object is mapped by more than one proto-object (see Sec. 5.2.3). Then, the target location is best approximated by merging the corresponding proto-objects and using the centroid of the newly-created proto-object (see Fig. 9.1).

For the merging procedure, the model has to know if two or more proto-objects belong to the same natural object. If this is the case, and the natural object belongs to object class n, then the corresponding proto-objects should, in general, produce the highest probability value $p(c|o_F)$ for $c = n$. To make use of this connection, the model assigns an identity value $id(o)$ to each proto-object (see Eq. 9.1).

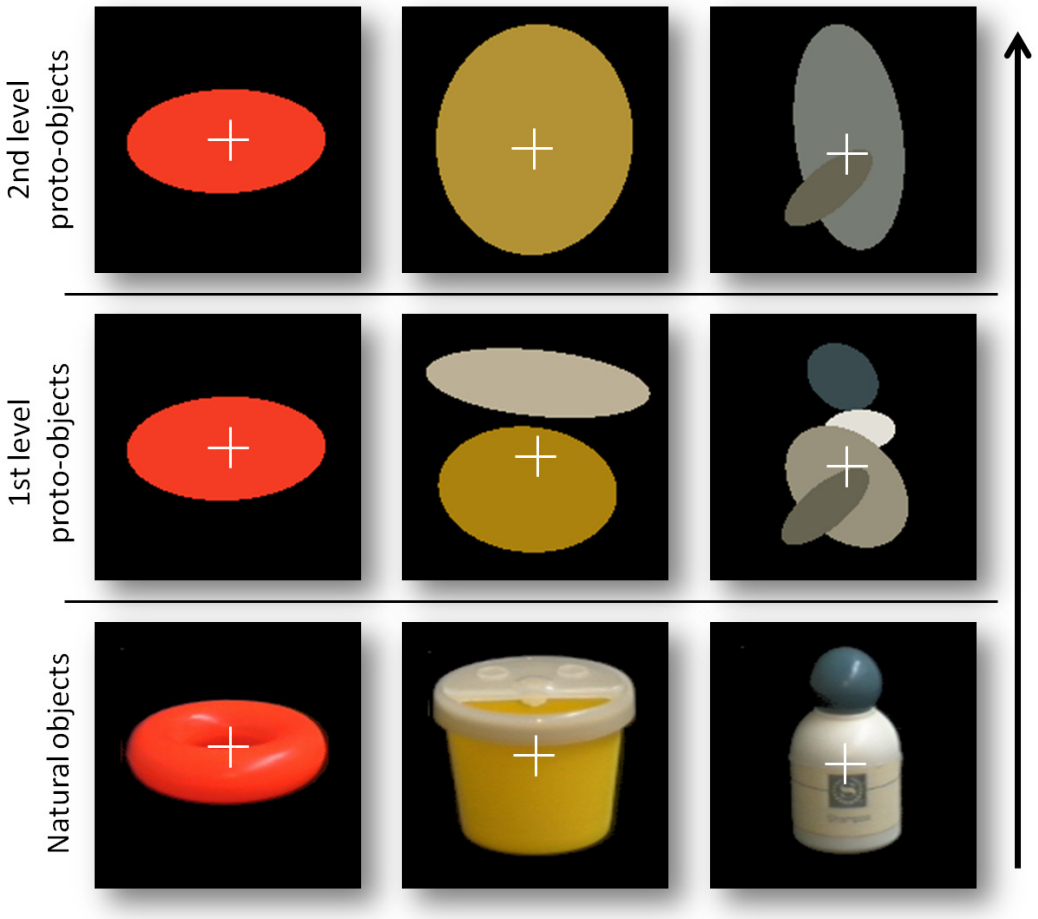

Figure 9.1: *Merging of proto-objects with* $f_{map} = 32.0$*. The white cross approximately marks the centroid of the natural object. Left column: If a natural object is represented by one proto-object, then (a) no merging occurs, and (b) the natural object's centroid is quite well mapped by the proto-object's centroid. Middle column: Here the natural object is represented by two proto-objects. As a result, the natural object's centroid is inadequately mapped because a saccade would land on one of the two proto-objects' centroids. After merging, the mapping has been significantly improved. Right column: Here, one proto-object has not been merged since its identity value (see Sec. 9.2.1) differs from the others. So, both remaining proto-objects are potential target objects for the next saccade.*

$$id(o) = \begin{cases} \max\limits_{0 \leq c < C} [p(c|o_F)], & \text{if } p(c|o_F) > m_{thr2} \\ -1, & \text{else} \end{cases} \tag{9.1}$$

This value denotes the object class that the proto-object most probably belongs to. So, if $p(c|o_F)$ is highest for $c = n$, then proto-object o most probably belongs to class n and thus $id(o)$ equals n. As a logical consequence, those proto-objects that represent the same natural object should, in general, have the same identity value. This results is the first necessary condition for merging: Two proto-objects can only be merged if an identical identity value is assigned to them.

The identity value cannot be understood as a result of object recognition; it is an assignment on the level of proto-objects and thus more prone to inaccuracies. Therefore, to avoid erroneous merges as much as possible, the merging procedure is subjected to the following condition: If the highest probability value $p(c|o_F)$ of a proto-object is equal to or less than m_{thr2}, then the identity value equals -1 (see Eq. 9.1), which means no assignment was made and this proto-object cannot be merged with any other proto-object. Thus, the merging of proto-objects can be adjusted by threshold parameter m_{thr2}.

9.2.2 Overlapping of proto-objects

The second necessary condition for merging concerns the spatial overlapping of proto-objects. For this, the model again makes use of an adjacency matrix $\bar{A}$ (see Sec. 5.2.4 for details). The value $\bar{A}_{ab}$ reflects the degree of overlapping of proto-object a and b. The higher $\bar{A}_{ab}$, the stronger the overlap. A $\bar{A}_{ab}$ value of zero signifies no overlap.

Finally, if two proto-objects a and b have the same identity value ($id(o_a) = id(o_b)$) and overlap ($\bar{A}_{ab} > 0$), then both proto-objects are merged. The merging works exactly like the merging of proto-objects described in Sec. 5.2.4. Additionally, an attentional weight has to be assigned to the new proto-object. This value equals the mean value of the merged proto-objects' weights (see Eq. 9.2).

$$w(o) = \frac{1}{j} \sum_{i=1}^{j} w(o_i) \tag{9.2}$$

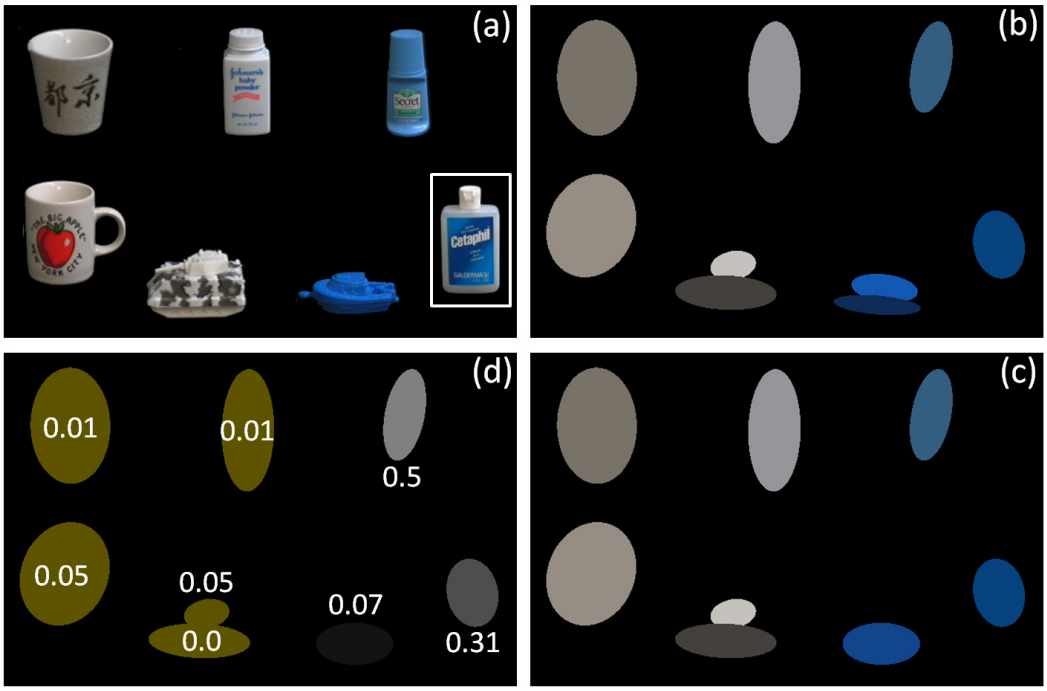

Figure 9.2: *APM example. (a) Input image. The object within the white rectangle serves as search target. (b) Resulting proto-objects. (c) Proto-objects after merging. (d) Attentional weights corresponding to the task. Intensity marks the normalized strength of weights, so the sum of weights equals* 1*. In order to visualize proto-objects with* $w(o) \leq 0.05$*, they are colored dark-yellow. As can be seen, the object the system searches for did not obtain the highest weight, so it likely will not become the first saccade's target.*

Here, $w(o)$ denotes the new proto-object's attentional weight and j the number of merged proto-objects.

9.2.3 A new level of proto-objects

The merging procedure creates a new level of proto-objects. After merging, all proto-objects, including non-merged proto-objects, are considerably more likely to represent natural objects as a whole. That is why these proto-objects are called 2nd-level proto-objects, whereas the proto-objects prior to merging are called 1st-level proto-objects (see Fig. 3.1).

2nd-level proto-objects are more accurate regarding the natural objects' shape and position

within the visual field. Moreover, they do not inherit any classification-relevant mid-level feature values from the 1st-level proto-objects. They only require a set of five geometric mid-level feature values, which map the proto-objects' position and elliptical shape: the length of both principal axes o_{λ_1} and o_{λ_2}, the orientation o_α of the longer principal axis o_{λ_1}, and the proto-object's centroids o_x and o_y (see Sec. 6.3.2 and 6.3.3 for how to compute these values). These feature values are needed for computing the next saccade's landing position (see Sec. 9.6) and an associated region for "inhibition of return" (see Sec. 9.4).

9.3 The attention priority map (APM)

At this point, the building of proto-objects is completely finished. The result is a set of proto-objects where each proto-object consists of a task-dependent attentional weight and a set of geometric feature values which map the proto-object's position and elliptical shape. As in the biological model, the proto-objects are mapped by a topographically organized *attention priority map* (APM), see (Bundesen, Habekost, and Kyllingsbæk 2005).

For each proto-object, there is one entry in the priority map. The location of each entry is determined by the corresponding proto-object's centroid, which reflects its position within the visual field. The entry's strength of priority equals its attentional weight. Fig. 9.2 shows an example. The higher the priority value on a priority map's location, the more likely the next saccade will land close to this position.

9.4 Inhibition of return (IOR)

If the model only uses the attentional weights of the APM to control saccades, this would lead to an unwanted behavior of the system. If the winner of the previous race, which is the natural object that has been brought to the fovea, again achieves the highest attentional weight, then the system would be stuck at the foveal position. This is quite likely because, given the same task, a target object would again achieve a high attentional weight. Moreover, this object would obtain an even higher attentional weight by being more foveally located (proximity effect, see Sec. 8.3 for details). A simple solution would be to restrict potential target positions to the non-foveal area of the visual field. But this approach cannot handle another unwanted behavior: the oscillation between two or more objects. Even if the system is forced to saccade from the

foveally located object o_1 to an object outside the foveal area, e.g. o_2, it is quite likely that the subsequent saccade target will again be o_1. Thus, the systems would produce an oscillating scan path between o_1 and o_2.

A typical attempt at a solution in computational models is to memorize the locations of the last n targets. At these locations in the APM, the attentional weights are decreased in compliance with the sequence of saccades. That is, the attentional weights around the last target's position are decreased the most, whereas the attentional weights around the n-last target's position are decreased the least. This method is called inhibition of return (IOR) (Klein 2000). In contrast to the human model, this approach is rather simple, but it is very effective and substantially diminishes the problems mentioned above (see (Tatler, Hayhoe, Land, and Ballard 2011) for a deeper discussion).

In this model each saccade's target is memorized as Gaussian distributions in the IOR map. A target's Gaussian is described by its geometric mid-level features: The center of the Gaussian equals the target's centroid ($g_x = o_x$ and $g_y = o_y$), the standard deviations equal the length of the principal axes ($\sigma_x = o_{\lambda_1}$ and $\sigma_y = o_{\lambda_2}$), and the orientation equals the target's orientation ($g_\alpha = o_\alpha$). Using a Gaussian implies that the decrease of a proto-object's attentional weight depends on the distance between the proto-object's centroid and the Gaussian's center. The greater the distance, the less the decrease. In order to obtain the correct function value of the Gaussian distribution, the system has to consider its orientation. For that, the proto-objects' centroids are rotated by $-g_\alpha$ around the Gaussian's center. In this way, for each Gaussian of the IOR map, the model computes a new centroid (o_{xRot}, o_{yRot}) for the proto-objects (see Eq. 9.3).

$$\begin{aligned}\begin{pmatrix} o_{xRot} \\ o_{yRot} \\ 1 \end{pmatrix} &= RT \begin{pmatrix} o_x \\ o_y \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} cos(-g_\alpha) & -sin(-g_\alpha) & 0 \\ sin(-g_\alpha) & cos(-g_\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -g_x \\ 0 & 1 & -g_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} o_x \\ o_y \\ 1 \end{pmatrix}\end{aligned} \tag{9.3}$$

Then the rotated centroid can be used to compute the function value $g(o)$ (see Eq. 9.4).

$$g(o) = \exp\left(-\frac{1}{2}\left[\frac{o_{xRot}^2}{\sigma_x^2} + \frac{o_{yRot}^2}{\sigma_y^2}\right]\right) g_f \tag{9.4}$$

The first part of $g(o)$ corresponds to a Gaussian distribution, but without the usual normalization factor $\mathcal{N} = \frac{1}{2\pi\sigma_x\sigma_y}$. This in turn normalizes the maximum value of every Gaussian to 1. The second part adds the factor g_f to the Gaussian. This factor is needed to gradually decrease the value of $g(o)$ after a saccade.

After each saccade, the Gaussian of the target object o_t is added to the IOR map with $g_f = 1$, thus $g(o_t)$ produces a maximum output of 1 at (g_x, g_y). Since the maximum value of the attentional weights $w(o)$ without consideration of the IOR map equals 1 (see Sec. 8.3), in the next race, the attentional weight of the target is near or equal to zero.

Then, with each additional saccade, g_f is decreased by parameter $\Delta g_f (g_f = g_f - \Delta g_f)$. If g_f is less or equal to 0, then the Gaussian is removed from the IOR map. As a consequence, Δg_f determines the number of saccades a Gaussian survives in the IOR map. This value is equivalent to the number of Gaussians the IOR map is maximally able to contain. The standard value of Δg_f is 0.25 which corresponds to an IOR map capacity of four memorized targets. Fig. 9.3 shows an example of what an IOR map looks like.

Now the computation of the attentional weights can be extended by the IOR map. For each proto-object, the model computes the sum of all Gaussians $g \in G$ of the IOR map. This value is subtracted from the original attentional weight (see Eq. 9.5).

$$w(o) = i(o_p) \sum_{0 \leq c < C} p(c|o_F)\pi_c - \sum_{g \in G} g(o) \tag{9.5}$$

As $w(o)$ can in principle not be negative, the lower limit for $w(o)$ equals zero. Fig. 9.4 shows an example of how the IOR map influences the proto-objects' attentional weights.

9.5 Winner-takes-all (WTA)

At this point, after the IOR map was integrated, the proto-objects have their final attentional weights. In correspondence with the TVA rate equation, the attentional weights are normalized so that their sum equals 1 (Bundesen 1990). In the model, the resulting weights reflect the probability of being the next saccade target. So, e.g., a normalized weight of 0.3 means that, with $p = 0.3$, the corresponding proto-object wins the race and thus serves as next saccade target. Thus, it is not always the proto-object with the highest attentional weight that wins

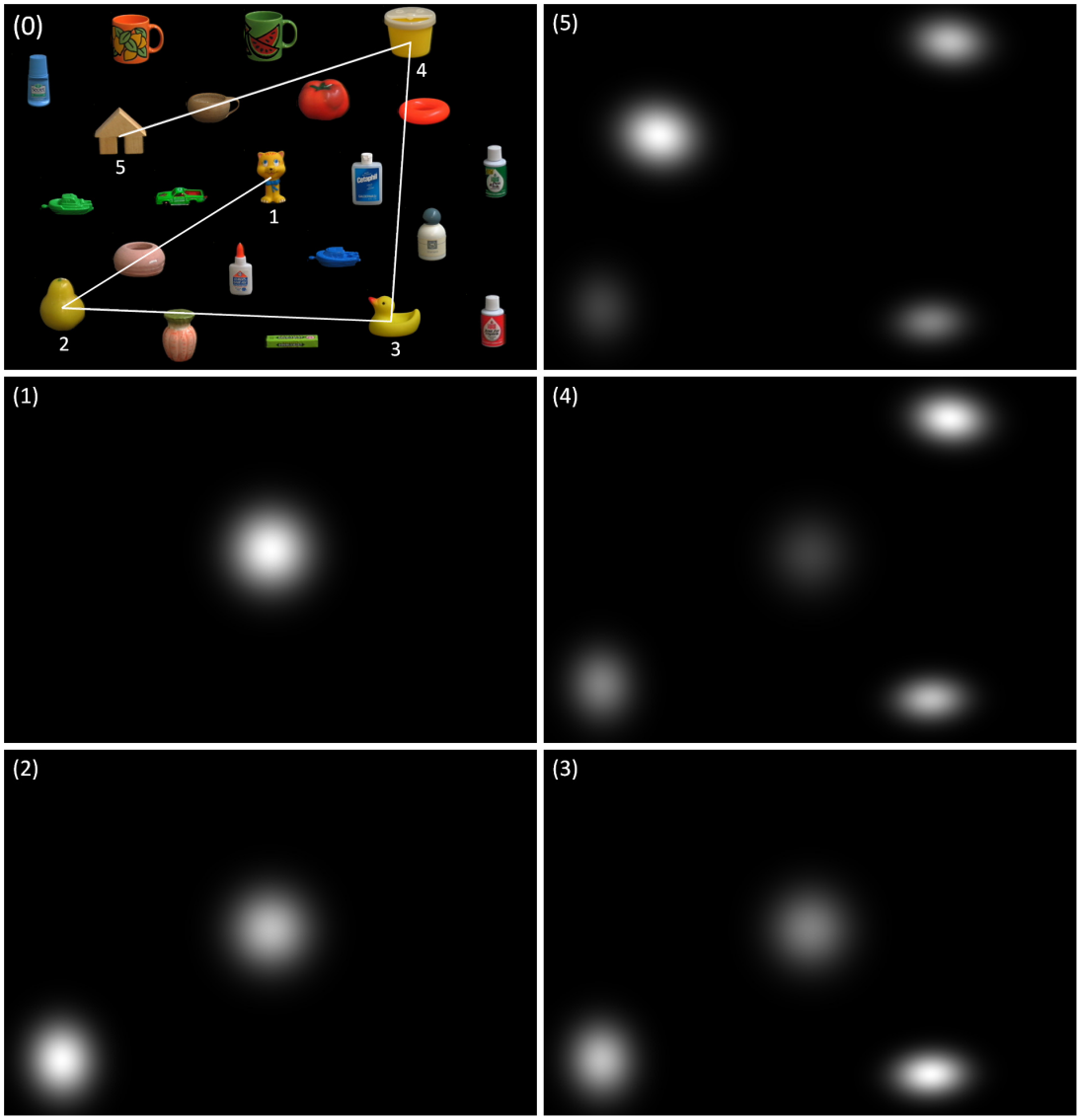

Figure 9.3: *IOR map example. (0) Input image. The task was to find the duck. The white lines denote the scanpath, and the number under the objects denotes the $n-th$ fixation point. In this example, the duck was found after two saccades. (1)-(5) IOR map after the $n-th$ fixation. So, e.g., if the duck is fixated, the system adds a duck-representing Gaussian to the IOR map. For this reason, image (3) consists of three Gaussians including the duck's one. The result is a maximal inhibition of the duck's area for the subsequent saccade. After each saccade, the IOR map as a whole is attenuated by Δg_f. With $\Delta g_f = 0.25$, an IOR map's Gaussian disappears after 4 saccades. This can be seen for the foveally located cat.*

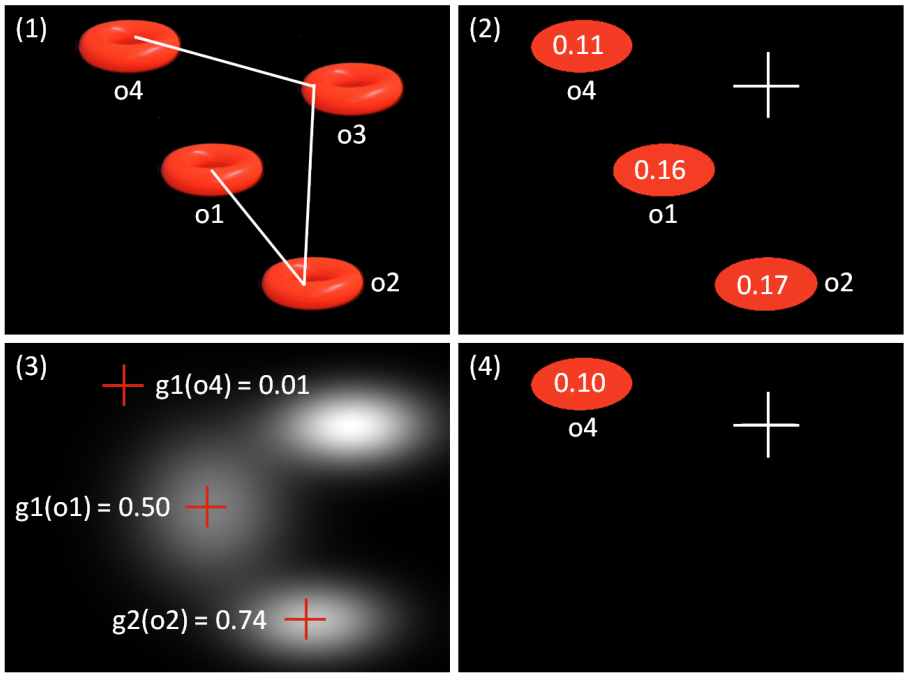

Figure 9.4: *How the IOR map influences attentional weights. (1) Four objects, o_1 to o_4, are shown. The white lines denote the scanpath. The numbering of objects corresponds to the fixation order. (2) Proto-objects and their attentional weights (without IOR) pertaining to the third saccade. The white cross marks the current fixation position. Object o_3 was filtered out as its value o_p exceeds p_{max}. At this point, based on the attentional weights, object o_2 would most likely become the next saccade target. (3) The IOR map after the second saccade. The red crosses mark the centroids of the proto-objects. There are three Gaussians, where g_2 and g_3 represent the Gaussian of the previously fixated proto-objects o_2 and o_3, whereas g_1 solely represents the Gaussian covering the foveal area of the starting position, such that g_1 has nothing to do with o_1. The value of $g_1(o_4)$ denotes how strong g_1 reduces the value of o_4. This value is very low as the distance between the centroids of g_1 and o_4 is rather large. By contrast, the value of $g_2(o_2)$ is very high as both centroids are located nearby and o_2 was the previously fixated object. As a result, the attentional weight of o_2 equals zero (lower values are not allowed). In general, when having n objects and m Gaussians, a value for each combination $g_m(o_n)$ is computed. In this example, results that are lower than 0.01 are not shown. (4) Only proto-object o_4 has survived the IOR stage with an attentional weight of 0.10. After normalizing, this value increases to 1 as there are no more proto-objects in the race.*

the race, but this is the most probable case. As there is only one winner, this method can be described as a variant of the classic *winner-takes-all* (WTA) approach.

In visual attention, the concept of probability values has been successfully applied to reporting tasks (see (Bundesen and Habekost 2008) for an extensive overview). The interesting point in reporting tasks is that attentional weights also influence the perception latency, that is, the difference between the occurrence of a visual stimulus and its memorization in visual short-term memory (VSTM). The higher the weight, the lower on average the latency. Carbone and Schneider have shown that attentional weights influence saccadic latencies in the same way (Carbone and Schneider 2010). This time aspect is not yet part of the model but would be an interesting extension.

9.6 The landing position

After a winner has been determined, a specific landing position also has to be determined. Empirical findings show that saccades generally land close to a natural object's center (Foulsham and Underwood 2009; Nuthmann and Henderson 2010), which can be coarsely mapped by a (truncated) Gaussian distribution (Nuthmann and Henderson 2010). The center of such a Gaussian approximately equals the centroid of the associated natural object, which is simulated in this model by the centroid of the corresponding proto-object. The Gaussian's standard deviations come from the length of the corresponding proto-objects' principal axes and can be scaled by parameter sd_s, so $\sigma_1 = sd_s o_{\lambda_1}$ and $\sigma_2 = sd_s o_{\lambda_2}$. The standard value of sd_s is $\frac{1}{3}$.

That way a Gaussian distribution for the landing position can be built for each target object. The next step is to obtain a random position based on this distribution. For this, the model makes use of the Box-Muller method (Box and Muller 1958). First, four independent random variables are required: p_1, p_2, z_1, and z_2. To p_1 and p_2, the model assigns a random value from the interval $[0..1]$. Then z_1 and z_2 can be computed (see Eq. 9.6 and 9.7). These values reflect the position (z_1, z_2) within the Gaussian distribution.

$$z_1 = \sqrt{-2 \ln p_1} \cos 2\pi p_2 \tag{9.6}$$

$$z_2 = \sqrt{-2 \ln p_1} \sin 2\pi p_2 \tag{9.7}$$

As z_1 and z_2 have a deviation of 1, these values have to be transformed to obtain the correct deviations o_{λ_1} and o_{λ_2} (see Eq. 9.8 and 9.9).

$$z_1 = sd_s o_{\lambda_1} z_1 = \sigma_1 z_1 \tag{9.8}$$

$$z_2 = sd_s o_{\lambda_2} z_2 = \sigma_2 z_2 \tag{9.9}$$

Finally, the landing position (pos_x, pos_y) is obtained by rotating the Gaussian by the target proto-object's orientation o_α and shifting it to the proto-object's centroid (o_x, o_y) (see Eq. 9.10).

$$\begin{aligned} \begin{pmatrix} pos_x \\ pos_y \\ 1 \end{pmatrix} &= TR \begin{pmatrix} z_1 \\ z_2 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & o_x \\ 0 & 1 & o_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} cos(o_\alpha) & -sin(o_\alpha) & 0 \\ sin(o_\alpha) & cos(o_\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ 1 \end{pmatrix} \end{aligned} \tag{9.10}$$

Now the (robotic) system can execute the saccade. After the target position has been reached, the bottom-up processing starts again from the beginning. This means that after each saccade, the attentional weights for the sensory objects are completely recomputed.

On the level of proto-objects, the system is not capable of doing object recognition; that is, the proto-object representation is not sufficient to decide if the natural object in the fovea is identical to the object the system searches for. In order to implement a "target found" criterion, the model has to be extended by a high-level object representation for object recognition. This is not part of this thesis but would be an essential aspect of a future model extension.

9.7 Summary

As various natural objects are mapped by two or more proto-objects (see Sec. 5.2.3), the model includes a processing stage which *merges* proto-objects that likely belong to the same natural object in the input stream (see Sec. 9.2). There are two merging criteria. First, both proto-objects have to be directly adjacent. Second, both proto-objects have to belong to the same object class c. The assigned class is represented by the so-called *identity value* (see Eq. 9.1).

As a result, the merging stage reduces the number of proto-objects and, by doing so, increases the model's ability to fully map a natural object. Moreover, this improves the model's ability to saccade to the center of natural objects (see Sec. 2.7)

After merging, proto-objects are stored in a retinotopic *attention priority map* (APM) (Bundesen, Habekost, and Kyllingsbæk 2005) (see Sec. 9.3) along with their weights and geometric mid-level features (see Sec. 9.2.3). So not only their weights but also their shapes and positions within the visual field are known, which is indispensable for saccading.

In order to avoid both *saccadic standstills* and *saccadic oscillations*, the model implements a simple *inhibition of return* (IOR) mechanism (see Sec. 2.7 and 9.4): The attentional weights of the n last fixated locations in the APM, and thus of the objects at these locations, are increased, which eliminates early re-fixations. So, even proto-objects with lower attention weights have the chance to become the next saccade target if the objects fixated before have been identified as not being the object the system searches for.

A subsequent *winner-takes-all* (WTA) mechanism determines the winning proto-object based on attentional weights: The higher the weight, the more likely a proto-object is the winner (see Sec. 9.5). Finally, the concrete landing position is computed using a 2D-Gaussian, whose mean and variances corresponds to the geomotric features of the target proto-object (Nuthmann and Henderson 2010) (see Sec. 9.6).

After saccading, the next cycle can be started: All computation stages (which does not include the learning of natural objects) are run again from the beginning. According to NTVA (Bundesen, Habekost, and Kyllingsbæk 2005), the priority map is memory-less, so every information constructed in the previous cycle gets lost.

Chapter 10

The model performance

10.1 Introduction

An essential characteristic of the model when computing task-relevance of sensory objects is the *performance* level: The performance must neither be too low (no sufficient discrimination) nor too high (time-consuming object recognition) (see Sec. 2.6 and 6.1). For this, in this chapter, a mechanism is presented to adjust the model's ability to distinguish *target* objects from non-relevant objects, called *distractors*, in a TVA-like fashion(see Sec. 10.2). Then, it is illustrated by examples how the performance depends on both the object class and the network's architecture (see Sec. 10.3).

10.2 Target-distractor discriminability

TVA provides a set of parameters to describe the individual-related performance in reporting tasks (Bundesen 1990; Kyllingsbæk 2006; Bundesen and Habekost 2008). One of these parameters, called α, reflects the *efficiency of selection*, that is, how well a person is able to distinguish a target object (that is, an object to be reported) from a distractor (see Eq. 10.1).

$$\alpha = \frac{w_d}{w_t} \tag{10.1}$$

An α value of zero means that a test person never reports a distractor ($w_d = 0$), whereas

an α value of 1 implies that a test person cannot distinguish between targets and distractors ($w_d = w_t$). Normally, α lies between these two values.

In TVA, α is gained from empirical findings. One of the best-known examples is illustrated in (Shibuya and Bundesen 1988). A great number of further examples can be found in (Bundesen and Habekost 2008). When modeling, it is the other way round. Then, the aim of the model is to reproduce a desired target-distractor discriminability, e.g., one coming from empirical data. In the following, it is shown how that can be realized.

In this model, the original TVA α is replaced by α_{td}, which denotes the *target-distractor discriminability* of two object classes, one target class c_t and one distractor class c_d. Furthermore, in contrast to Eq. 10.1, where weights for targets as well as distractors are assumed to be constant, the computation of α_{td} takes into account that the weights of proto-objects belonging to one object class may substantially differ. This is because each natural object is represented by a set of different proto-objects (see Sec. 7.4 for a summary) during the learning stage (see Ch. 7). On this account, for both object classes c_t and c_d, the model builds the mean weight values $\mu_{w(o_t)}$ and $\mu_{w(o_d)}$ over all class examples the neural net has learned. The set X_t consists of the examples from the training data set, which belong to any target object class c_t.Additionally, X_d contains the examples of any distractor object class c_d with $c_d \neq c_t$. Furthermore, π_{c_t} with $\pi_{c_t} > 0$ is defined as the pertinence value of target object class c_t. All other π values equal zero, which makes it possible to considerably reduce the modified TVA equation (see Eq. 10.2).

$$w(o) = i(o_p) \sum_{0 \leq c < C} p(c|o_F)\pi_c = i(o_p)p(c_t|F_o)\pi_{c_t} \tag{10.2}$$

Then α_{td} can be computed according to Eq. 10.1 (see Eq. 10.3).

$$\alpha_{td} = \frac{\mu_{w(o_d)}}{\mu_{w(o_t)}} = \frac{\frac{1}{|X_d|} \sum\limits_{o \in X_d} w(o)}{\frac{1}{|X_t|} \sum\limits_{o \in X_t} w(o)} = \frac{\frac{1}{|X_d|} \sum\limits_{o \in X_d} i(o_p)p(c_t|o_F)\pi_{c_t}}{\frac{1}{|X_t|} \sum\limits_{o \in X_t} i(o_p)p(c_t|o_F)\pi_{c_t}} = \frac{|X_t| \sum\limits_{o \in X_d} i(o_p)p(c_t|o_F)}{|X_d| \sum\limits_{o \in X_t} i(o_p)p(c_t|o_F)} \tag{10.3}$$

$p(c_t|o_F)$ with $o \in X_t$ represents the conditional probability that an example target object o belongs to target class c_t, which is a correct classification, and $p(c_t|o_F)$ with $o \in X_d$ represents the conditional probability that an example distractor object o belongs to target class c_t, which is a wrong classification. $|X_t|$ equals the cardinal number of set X_t, that is, the number of examples for the target object class. The same analogically applies to $|X_d|$.

In contrast to TVA, the α_{td} value describes the *efficiency of selection in saccades* of two object classes. When having an input image with T target objects of class c_t and D distractor objects of class c_d, $p(s_t)$ reflects the probability that the next saccade lands on one of the target objects (see Eq. 10.4).

$$p(s_t) = \frac{\mu_{w(o_t)}T}{\mu_{w(o_t)}T + \mu_{w(o_d)}D} = \frac{T}{T + \alpha_{td}D} \tag{10.4}$$

The equation ensures a TVA-compliant normalization, as described in Sec. 9.5. A disadvantage of the $p(s_t)$ value is that it is restricted to a certain target-distractor combination because other combinations can have considerably different α_{td} values. For this reason, a single α_{td} value and thus a single $p(s_t)$ value, is not valid for the whole data set of natural objects the model has learned. The solution is to build a mean α value μ_α for all target-distractor combinations of the data set (see Eq. 10.5).

$$\mu_\alpha = \frac{1}{C(C-1)} \sum_{t=0}^{C-1} \sum_{d=0, d\neq t}^{C-1} \alpha_{td} \tag{10.5}$$

C equals the total number of object classes the system has learned and $C(C-1)$ the number of target-distractor combinations. The resulting μ_α value reflects the model's overall performance in target-distractor discriminability. To achieve the aim mentioned above, that is, that the model reproduces a desired μ_α value, the capacity of the neural network has to be chosen accordingly. The higher the capacity, the lower the μ_α value. How the neural network's capacity can be adjusted is comprehensively described in Ch. 7.

Having μ_α, the mean probability that the next saccade lands on one of the target objects can be computed (see Eq. 10.6).

$$\mu_{p(s_t)} = \frac{T}{T + \mu_\alpha D} \tag{10.6}$$

It is important to realize that these mean values exclusively reflect the general model performance, which means they do not enable one to make a prediction about attentional weights or saccadic probabilities for a concrete input image. Then, many other factors (e.g., the object classes that the natural objects belong to, the pertinence values, the objects' angle of eccentricity, the IOR map etc.) determine which object the system looks to next.

Architecture	Dimensionality of hidden layer	μ_α
1	1	0.64
2	3	0.26
3	5	0.10
4	10	0.07
5	20	0.04

Table 10.1: *Five different network architectures. The higher the dimensionality (the number of nodes) of a hidden layer, the lower μ_α and thus the higher the model's performance in distinguishing targets from distractors.*

10.3 Performance examples

In this section, it is shown how performance differs globally depending on the neural network's architecture and locally depending on the object class. The given examples are based on 100 natural objects of the COIL database (see App. B), so the number of classes C equals 100. The COIL objects are 128x128 pixels in size, which corresponds to a defined size of 2 degree of visual angle in each dimension. Each object is shown in 1131 different positions (39 x 29 grid, see Fig. 7.2 as illustration) within the visual field with a distance of 0.5 degree visual angle between two positions in both dimensions. This layout covers a range of -19.5 to 19.5 degree of visual angle in x-direction and -14.5 to 14.5 degree of visual angle in y-direction.

Added up, 84519 proto-objects were computed by the model using standard parameter values (except of the overall resolution with $f_{map} = 32.0$), which averages to 845.19 proto-objects per COIL object. Then, 5 out of 16 mid-level features (f_{rg}, f_{by}, f_{bw}, f_{area}, and f_{axes}) were used to train the network, so the number of mid-level features M that were used as the network's input equals 5. The network uses one hidden layer, whose dimensionality was varied fivefold. To compare the performance of the five network architectures, learning was stopped after 100 learning steps. In Fig. 10.1 the learning progress is illustrated. Tab. 10.1 shows the resulting μ_α values.

Another interesting point is the object-related classification performance. This can be measured by building the mean value of all α_{td} values regarding one class t (see Eq. 10.7).

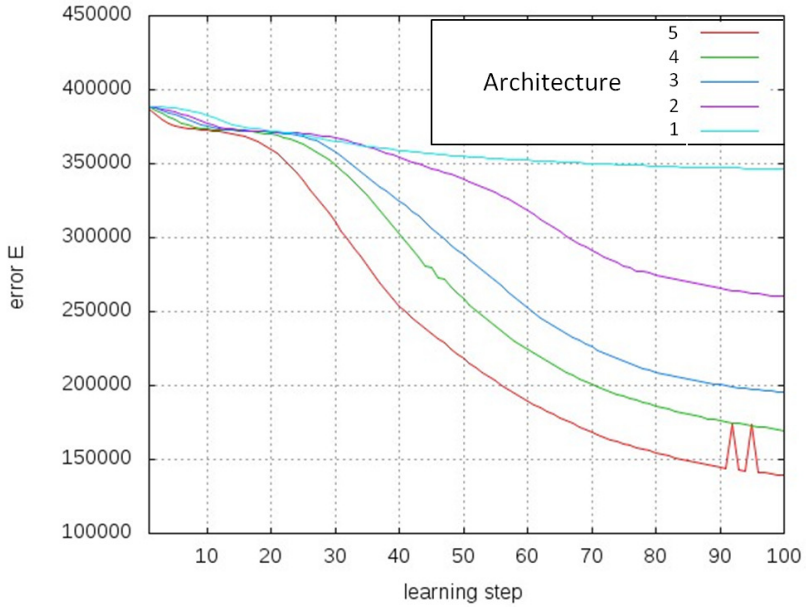

Figure 10.1: *Error E (see Eq. 7.3) depending on the network's architecture. The numbering corresponds to Tab. 10.1. The higher the dimensionality of the hidden layer, the lower E after 100 learning steps and the better the network is able to classify the examples.*

$$\mu_{\alpha_t} = \frac{1}{C-1} \sum_{d=0, d \neq t}^{C-1} \alpha_{td} \tag{10.7}$$

The resulting value is called μ_{α_t}, which denotes the model's ability to correctly classify a sensory object of class t as a member of class t. The lower μ_{α_t}, the better the classification performance. μ_{α_t} strongly differs from class to class. Fig. 10.2 illustrates this effect for all five network's architectures. The y-axis denotes the object classes, whereas the x-axis is used to illustrate the architecture-dependent μ_{α_t} value visualized by the length of the colored bars. So, e.g., for class 0 the μ_{α_t} value for architecture 1 (blue) equals 0.821, for architecture 2 (red) 0.14, for architecture 3 (yellow) 0.02 etc. So μ_{α_t} is computed based on class and architecture. Within one architecture, μ_{α_t} strongly differs depending on class t. This means that some target templates are more difficult to learn than others. Natural objects with a strong inhomogeneous color structure lead to especially unstable segmentation results and are thus more difficult to learn (see Fig. 10.3). Moreover, Fig. 10.2 illustrates the increase in performance depending on the chosen architecture. In mostly all cases, the bars become shorter by increasing the

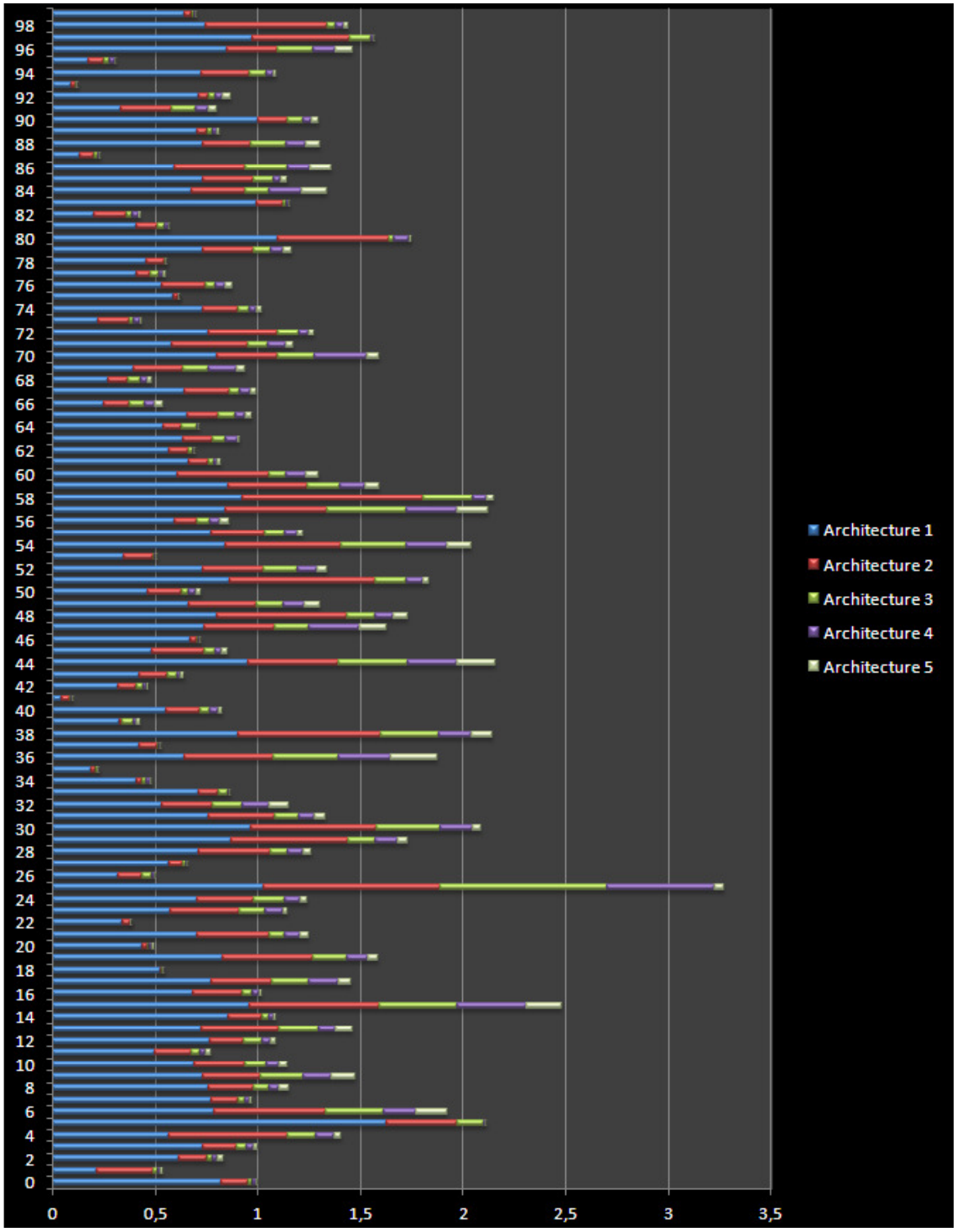

Figure 10.2: μ_{α_t} *depending on class and architecture. See continuous text for details.*

Figure 10.3: *μ_{α_t} depending on class t. Upper row: Objects consisting of large homogeneous regions in the RG/BY/BW-space are easiest to learn. Even for architecture 1 they obtain rather low $\mu_{\alpha_t} =$ values. Left to right: object class 35 with $\mu_{\alpha_t} = 0.19$, class 41 with $\mu_{\alpha_t} = 0.04$, and class 93 with $\mu_{\alpha_t} = 0.09$. Lower row: Objects that are quite inhomogeneously structured in the RG/BY/BW-space are more difficult to learn. Even for architecture 5 they obtain relatively low $\mu_{\alpha_t} =$ values. Left to right: object class 15 with $\mu_{\alpha_t} = 0.18$, class 36 with $\mu_{\alpha_t} = 0.23$, and class 44 with $\mu_{\alpha_t} = 0.19$.*

network's performance. Interestingly, the relative difference of μ_{α_t} values between classes seems to propagate from architecture to architecture: In most cases with $\mu_{\alpha_{t_1}} < \mu_{\alpha_{t_2}}$ for architecture n, the same also applies to architecture $n + 1$; so this effect is stable.

There are many possibilities to in- or decrease the model's performance. Although the dimensionality of the hidden layer in architecture 1 cannot be further reduced, performance could be decreased by using fewer mid-level features or learning steps. Conversely, an increase of μ_α can be achieved by using more mid-level features, hidden layers, learning steps, and a higher dimensionality of the hidden layers.

10.4 Summary

In this chapter, it was shown how the model's performance can be measured and adjusted using a TVA-like approach: The parameter μ_α describes the model's ability to distinguish targets from distractors, which is called *target-distractor discriminability*. When such a value is gained from empirical findings, the capacity of the model's neural network can be adjusted

to approximately reproduce it.

A variation in performance was illustrated using the COIL database (see App. B). It could be shown that performance strongly depends on the network's architecture and also differs from class to class. So, some objects are easier to correctly classify and thus easier to find in a search task.

Chapter 11

Summary and outlook

To decide "Where to look next?" is a central function of the attention system of humans, animals, and robots. In general, object-based control of attention depends on three factors: feature representations of the environment's objects (bottom-up), feature representations of potential target objects (target templates), and the task (top-down). In this thesis a novel, integrated computational model was presented, which includes all these factors in a coherent architecture based on findings and constraints from the primate visual system. The model combines spatial inhomogeneous bottom-up processing, learning of mid-level feature proto-object representations, and top-down task-dependent priority control in the form of a new computational implementation based on the "Theory of Visual Attention" (TVA, (Bundesen 1990)).

Priority control is realized on the level of proto-objects, that is, visual units that consist of mid-level features like size, shape, mean color, orientation etc. All perceived and learned proto-objects as well as the task definition serve as input to the TVA process. Tasks can be defined by choosing one or more learned objects as targets, e.g., "search for the microwave", without explicitly knowing their mid-level feature values. TVA combines this top-down and bottom-up information for computing attentional priorities; that is, for each perceived proto-object, a task-dependent attentional priority in the form of an attentional weight is computed and stored in a retinotopic attention priority map. Then, the target of the next saccade is most likely the center of gravity of the proto-object with the highest weight according to the task.

The model includes further essential mechanisms, like inhibition of return or the proximity effect, and allows a TVA-like performance adjustment to meet the concept of proto-objects. Its approach was illustrated by applying it to everyday objects and it was shown that it is robust

against parameter variations.

Due to its modular implementation, the model architecture can easily accommodate further functionality. The most important step would be to add an object recognition stage for the foveally perceived object in order to implement an "object found" status. In TAM (Zelinsky 2008) such a mechanism was integrated, but only on the proto-object level; that is, TAM computes the correlation between the low-level feature representation of the target and the central image pixel. If the correlation exceeds a given threshold, then the target is deemed to be found. But this procedure is not plausible since object identification strongly differs from the candidate approach of proto-objects. So, an additional stage is necessary that realizes genuine object recognition and thus exceeds the level of mid-level features.

There are also other interesting possibilities to extend the model. For the segmentation stage, an additional depth feature would improve the distinction between nearby and similarly colored objects which significantly differ in depth. Moreover, stereo vision would generally make it possible to determine object positions in three-dimensional space, which is important for acting, e.g., taking a cup. Furthermore, scene knowledge could shift attention to places or areas where we expect an object, e.g., a chair on the floor rather than under the ceiling. Many other aspects of visual attention could be implemented in the model.

Appendix A

Notation

Generally speaking, the listing of parameters, variables, etc. follows the chronology of the thesis. The list contains the most important identifiers. Next to the name, a short description is given. For more detailed information regarding computation, functionality, standard values, etc., check the corresponding sections.

		Feature map
k	-	parameter: determines influence of eccentricity
f_{map}	-	parameter: determines the overall spatial resolution (pixels, filters)
θ	-	parameter: size of input image in visual angle in x-direction
e	-	eccentricity in visual angle
s	-	eccentricity-dependent scaling

		Pixel features
p_x	-	position: visual angle in x-direction
p_y	-	position: visual angle in y-direction
p_{rg}	-	color: red-green
p_{by}	-	color: blue-yellow
p_{bw}	-	intensity: black-white

		Proto-object segmentation
c	-	pixel: confidence value
p_c	-	filter: a pixel's confidence value
p_w	-	filter: a pixel's weight
p_l	-	filter: a pixel's out-liner value
l	-	proto-object: label counter
o_p	-	proto-object: number of pixels composing a proto-object
d_{max1}	-	parameter: determines the homogeneity of proto-objects
c_{min}	-	parameter: minimum filter response for a confidence value of 1
i_{num}	-	parameter: determines the number of iteration steps
m_{thr}	-	parameter: the minimum number of common filters
d_{max2}	-	parameter: merging threshold in the RG/BY/BW-space
p_{min}	-	parameter: smallest possible size of a proto-object's region
p_{max}	-	parameter: greatest possible size of a proto-object's region

		Mid-level features
f_{rg}	-	color: mean red-green
f_{by}	-	color: mean blue-yellow
f_{bw}	-	intensity: mean black-white
f_{area}	-	size: area of ellipse
f_{orient_n}	-	orientation: of the greater main principle axis with $n \in \{x, y\}$
f_{axis}	-	shape: relation of both main principal axes
f_{ring}	-	shape: ring-likeness
f_{sector_n}	-	shape: local pixel density with $n \in \{0..7\}$

		Classification network
M	-	input dimension: number of mid-level features
C	-	output dimension: number of classes
X	-	number of examples
E	-	network error
$p(j\|F_i)$	-	conditional probability that feature set F_i belongs to class j

		TVA
$w(o)$	-	attentional weight of proto-object o
π_c	-	set by task: pertinence of class c
$i(o_p)$	-	inhomogeneity factor
i_x	-	parameter: strength of inhomogeneity factor

		Geometric mid-level features of proto-objects
o_x	-	position: visual angle of centroid in x-direction
o_y	-	position: visual angle of centroid in y-direction
o_α	-	orientation: of the greater main principle axis
o_{λ_1}	-	length: of the greater main principle axis
o_{λ_2}	-	length: of the smaller main principle axis

		Saccade
$id(o)$	-	identity value of proto-object o
m_{thr2}	-	parameter: merging threshold
$g(o)$	-	Gaussian for inhibition of return for winning proto-object o
sd_s	-	parameter: determines variance of the IOR Gaussians
g_f	-	decay factor with regard to inhibition of return
Δg_f	-	parameter: strength of decay after each saccade
(pos_x,pos_y)	-	saccade's landing position (x, y) in visual angle

		Performance
α_{td}	-	target-distractor discriminability regarding one target and one distractor class
μ_α	-	parameter: global target-distractor discriminability
$\mu_{p(s_t)}$	-	mean probability that the next saccade lands on a target object

Appendix B

Image Library

For this thesis the Columbia Object Image Library (COIL) was used to demonstrate the model's functionality. The library consists of 100 objects shown in different orientations. For this thesis only one orientation, the front view, was selected (see Fig. B.1). The background of the images, which was originally dark-gray, was made to be black. Otherwise, since objects' mid-level features are learned on a black background, such a rectangular dark-gray background area would be interpreted as a part of the objects.

Figure B.1: *COIL database front view.*

References

Allport, D. A. (1987). Selection for action. In H. Heuer and H. F. Sanders (Eds.), *Perspectives on Perception and Action*, pp. 395–419. Hillsdale, NJ: Lawrence Erlbaum Associates Inc.

Aziz, M. and B. Mertsching (2008). Fast and robust generation of feature maps for region-based visual attention. *Image Processing, IEEE Transactions on 17*(5), 633 –644.

Ballard, D. H. and M. M. Hayhoe (2009). Modelling the role of task in the control of gaze. *Visual Cognition 17*(6), 1185–1204.

Belardinelli, A., W. Schneider, and J. Steil (2010). Oop: Object-oriented-priority for motion saliency maps. In *Brain-inspired Cognitive Systems (BICS 2010)*, pp. 370–381.

Bishop, C. M. (1996). *Neural Networks for Pattern Recognition* (1 ed.). Oxford University Press, USA.

Box, G. E. P. and M. E. Muller (1958). A note on the generation of random normal deviates. *The Annals of Mathematical Statistics 29*(2), 610–611.

Breazeal, C. and B. Scassellati (1999). A context-dependent attention system for a social robot. In *IJCAI '99*, San Francisco, CA, USA, pp. 1146–1153. Morgan Kaufmann Publishers Inc.

Bundesen, C. (1990). A theory of visual attention. *Psychological review 97*(4), 523–547.

Bundesen, C. and T. Habekost (2008). *Principles of visual attention: Linking mind and brain*. Oxford university Press.

Bundesen, C., T. Habekost, and S. Kyllingsbæk (2005). A neural theory of visual attention: bridging cognition and neurophysiology. *Psychological review 112*(2), 291–328.

Bundesen, C., T. Habekost, and S. Kyllingsbæk (2011). A neural theory of visual attention and short-term memory (ntva). *Neuropsychologia 49*(6), 1446–57.

Burt, P. J. and E. H. Adelson (1983). The laplacian pyramid as a compact image code. *IEEE Transactions on Communications C*(4), 532–540.

Carbone, E. and W. X. Schneider (2010). Gaze is special: The control of stimulus-driven saccades is not subject to central, but visual attention limitations. *Attention, Perception, & Psychophysics 72*, 2168–2175.

Carrasco, M., D. Evert, I. Chang, and S. Katz (1995). The eccentricity effect: Target eccentricity affects performance on conjunction searches. *Perception & Psychophysics 57*(8), 1241–1261.

Cox, D. D., P. Meier, N. Oertelt, and J. J. DiCarlo (2005). 'breaking' position invariant object recognition. *Nature Neuroscience 8*, 1145–1147.

Cutsuridis, V. (2009). A cognitive model of saliency, attention, and picture scanning. *Cognitive Computation 1*(4), 292–299.

Deco, G. and D. Heinke (2007). Attention and spatial resolution: a theoretical and experimental study of visual search in hierarchical patterns. *Perception 36*(3), 335–354.

Denecke, A., H. Wersing, J. J. Steil, and E. Körner (2009). Online figureground segmentation with adaptive metrics in generalized lvq. *Neurocomputing 72*(7-9), 1470–1482.

DeSimone, R. and J. Duncan (1995). Neural mechanisms of selective visual attention. *Annual Review of Neuroscience 18*(1), 193–222.

Deubel, H. and W. X. Schneider (1996). Saccade target selection and object recognition: Evidence for a common attentional mechanism. *Vision Research 36*(12), 1827 – 1837.

Driscoll, J., R. Peters, and K. Cave (1998). A visual attention network for a humanoid robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 12–16.

Einhäuser, W. and P. Perona (2008). Objects predict fixations better than early saliency. *Journal of Vision 8*(14), 1–26.

Elazary, L. and L. Itti (2008). Interesting objects are visually salient. *Journal of Vision 8*(3), 1–15.

Elazary, L. and L. Itti (2010). A bayesian model for efficient visual search and recognition. *Vision Research 50*(14), 1338–1352.

Findlay, J. and I. Gilchrist (2003). *Active vision: the psychology of looking and seeing*. Oxford psychology series. Oxford University Press.

Findlay, J. M. (1982). Global visual processing for saccadic eye movements. *Vision Research 22*(8), 1033 – 1045.

Forssén, P.-E. (2004). *Low and Medium Level Vision using Channel Representations*. Ph. D. thesis, Linköping University, Sweden, SE-581 83 Linköping, Sweden. Dissertation No. 858, ISBN 91-7373-876-X.

Foulsham, T. and G. Underwood (2009). Does conspicuity enhance distraction? saliency and eye landing position when searching for objects. *Quarterly journal of experimental psychology (2006) 62*(6), 1088–1098.

Foulsham, T. and G. M. Underwood (2011). If visual saliency predicts search, then why? evidence from normal and gaze-contingent search tasks in natural scenes. *Cognitive Computation 3*(1), 48–63.

Freeman, J. and C. M. Ziemba (2011). Unwrapping the ventral stream. *Journal of Neuroscience 31*(7), 2349–51.

Frintrop, S., E. Rome, and H. I. Christensen (2010). Computational visual attention systems and their cognitive foundations: A survey. *ACM Trans. Appl. Percept. 7*(1), 1–39.

Gabor, D. (1946). Theory of communication. *Communication Theory 93*(3), 429–457.

Geisler, W. S. and J. S. Perry (2002). Real-time simulation of arbitrary visual fields. In A. T. Duchowski (Ed.), *Proceedings of the symposium on Eye tracking research applications ETRA 2002*, pp. 83–87. ACM Press.

Haxhimusa, Y. and W. G. Kropatsch (2003). Hierarchical image partitioning with dual graph contraction. In B. Michaelis and G. Krell (Eds.), *In Proceeding of the DAGM Symposium 2003*, Magdeburg, pp. 338–345. Springer.

Henderson, J. M. (2003). Human gaze control during real-world scene perception. *Trends in Cognitive Sciences 7*(11), 498–504.

Henderson, J. M. (2007). Regarding scenes. *Current Directions in Psychological Science 16*(4), 219–222.

Henderson, J. M., J. R. Brockmole, M. S. Castelhano, and M. Mack (2007). Visual saliency does not account for eye movements during visual search in real-world scenes. *Eye movements A window on mind and brain*, 537–562.

H.S. Scholte, V. L. (2009). Vision: Surface segmentation. In *Encyclopedia of Neuroscience*. Academic Press, Oxford.

Humphreys, G. W. and M. J. Riddoch (2006). Features, objects, action: The cognitive neuropsychology of visual object processing, 1984-2004. *Cognitive Neuropsychology 23*(1), 156–183.

Humphreys, G. W., E. Y. Yoon, S. Kumar, V. Lestou, K. Kitadono, K. L. Roberts, and M. J. Riddoch (2010). The interaction of attention and action: from seeing action to acting on perception. *British journal of psychology 101*(2), 185–206.

Hutton, S. B. (2008). Cognitive control of saccadic eye movements. *Brain and Cognition: A Hundred Years of Eye Movement Research in Psychiatry 68*(3), 327–340.

Hwang, A. D., E. C. Higgins, and M. Pomplun (2009). A model of top-down attentional control during visual search in complex scenes. *Journal of Vision 9*(5), 1–18.

Itti, L. and C. Koch (2000). A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision Research 40*(10-12), 1489–1506.

Itti, L. and C. Koch (2001). Feature combination strategies for saliency-based visual attention systems. *Journal of Electronic Imaging 10*(1), 161–169.

Itti, L., C. Koch, and E. Niebur (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*(11), 1254–1259.

Kahneman, D., A. Treisman, and B. J. Gibbs (1992). The reviewing of object files: object-specific integration of information. *Cognitive Psychology 24*(2), 175–219.

Kehrer, L. (1989). Central performance drop on perceptual segregation tasks. *Spatial Vision 4*(1), 45 – 62.

Kehrer, L. and C. Meinecke (2003). A space-variant filter model of texture segregation: parameter adjustment guided by psychophysical data. *Biological Cybernetics 88*(3), 183–200.

Kirstein, S., H. Wersing, and E. Körner (2008). A biologically motivated visual memory architecture for online learning of objects. *Neural Networks 21*(1), 65–77.

Klein, R., G. Berry, K. Briand, B. D'Entremont, and M. Farmer (1990). Letter identification declines with increasing retinal eccentricity at the same rate for normal and dyslexic readers. *Perception & Psychophysics 47*(6), 601–606.

Klein, R. M. (2000). Inhibition of return. *Trends in Cognitive Sciences 4*(4), 138–147.

Koch, C. and S. Ullman (1985). Shifts in selective visual attention: towards the underlying neural circuitry. *Human neurobiology 4*(4), 219–227.

Krauzlis, R. and L. Chukoskie (2009). Target selection for pursuit and saccades. In L. R. Squire (Ed.), *Encyclopedia of Neuroscience*, pp. 863 – 868. Oxford: Academic Press.

Kravitz, D. J., K. S. Saleem, C. I. Baker, and M. Mishkin (2011). A new neural framework for visuospatial processing. *Nature Reviews Neuroscience 12*(4), 217–230.

Kyllingsbæk, S. (2006). Modeling visual attention. *Behavior Research Methods 38*(1), 123–133.

Land, M. F. (2009). Vision, eye movements, and natural behavior. *Visual Neuroscience 26*(1), 51–62.

Land, M. F. and B. W. Tatler (2009). *Looking and acting: vision and eye movements in natural behaviour*. Oxford University Press.

Lee, T. S. and D. Mumford (2003). Hierarchical bayesian inference in the visual cortex. *Journal of the Optical Society of America A 20*(7), 1434–1448.

Lingyun, Z., M. H. Tong, and G. W. Cottrell (2007). Information attracts attention: A probabilistic account of the cross-race advantage in visual search. In *Proceedings of the 29th Annual Cognitive Science Conference*.

McPeek, R. M., V. Maljkovic, and K. Nakayama (1999). Saccades require focal attention and are facilitated by a short-term memory system. *Vision Research 39*(8), 1555–1566.

Moren, J., A. Ude, A. Koene, and G. Cheng (2008). Biologically based top-down attention modulation for humanoid interactions. *International Journal of Humanoid Robotics 5*(1), 3–24.

Mozer, M. and D. Baldwin (2008). Experience-guided search: A theory of attentional control. *Advances in Neural Information Processing Systems 20 20*, 1–8.

Naber, M., T. A. Carlson, and W. Einhäuser (2011). Perceptual benefits of objecthood. *Journal of Vision 11*(4), 1–9.

Nagai, Y. (2009). From bottom-up visual attention to robot action learning. In *Proceedings of 8 IEEE International Conference on Development and Learning*. IEEE Press.

Nagai, Y., K. Hosoda, A. Morita, and M. Asada (2003). A constructive model for the development of joint attention. *Connection Science 15*(4), 211–229.

Nassi, J. J. and E. M. Callaway (2009). Parallel processing strategies of the primate visual system. *Nature Reviews Neuroscience 10*(5), 360–72.

Navalpakkam, V. and L. Itti (2005). Modeling the influence of task on attention. *Vision Research 45*(2), 205–231.

Navalpakkam, V. and L. Itti (2006). An integrated model of top-down and bottom-up attention for optimal object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, New York, NY, pp. 2049–2056.

Navalpakkam, V. and L. Itti (2007). Search goal tunes visual features optimally. *Neuron 53*(4), 605–617.

Navalpakkam, V. and L. Itti (2010). A goal oriented attention guidance model. In *Biologically Motivated Computer Vision*, pp. 81–118. Springer.

Neumann, O. (1987). Beyond capacity: A functional view of attention. perspectives on perception and action. In H. Heuer and H. F. Sanders (Eds.), *Perspectives on Perception and Action*, pp. 361–394. Hillsdale, NJ: Lawrence Erlbaum Associates Inc.

Nothdurft, H. (1993). The role of features in preattentive vision: Comparison of orientation, motion and color cues. *Vision Research 33*(14), 1937–1958.

Nuthmann, A. and J. M. Henderson (2010). Object-based attentional selection in scene viewing. *Journal of vision 10*(8), 1–19.

Orabona, F., G. Metta, and G. Sandini (2008). A proto-object based visual attention model. In *Attention in Cognitive Systems. Theories and Systems from an Interdisciplinary Viewpoint*, pp. 198–215.

P. Meer, B. G. (2001). Edge detection with embedded confidence. *IEEE Transactions on Pattern Analysis and Machine Intelligenc 23*, 1351–1365.

Palmer, S. E. (1999). *Vision Science*. Cambridge, Mass.: MIT Press.

Park, S., J. Shin, and M. Lee (2010). Biologically inspired saliency map model for bottom-up visual attention. In *Biologically Motivated Computer Vision*, pp. 113–145. Springer.

Peters, R. J. and L. Itti (2007). Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2007)*, Number 1, pp. 1–8.

Pylyshyn, Z. W. (2001). Visual indexes, preconceptual objects, and situated vision. *Cognition 80*(1-2), 127–158.

Rensink, R. A. (2000). The dynamic representation of scenes. *Visual Cognition 7*(1), 17–42.

Roelfsema, P. R. (2006). Cortical algorithms for perceptual grouping. *Annual Review of Neuroscience 29*, 203–227.

Rosch, E., C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyes-Braem (1976). Basic objects in natural categories. *Cognitive Psychology 8*(3), 382439.

Rothkopf, C. A., D. H. Ballard, and M. M. Hayhoe (2007). Task and context determine where you look. *Journal of Vision 7*(14), 1–20.

Rousselet, G. A., S. J. Thorpe, and M. Fabre-Thorpe (2004). How parallel is visual processing in the ventral pathway? *Trends in Cognitive Sciences 8*(8), 363–370.

Ruesch, J., M. Lopes, A. Bernardino, J. Hornstein, J. Santos-Victor, and R. Pfeifer (2008). Multimodal saliency-based bottom-up attention a framework for the humanoid robot icub. In *International Conference on Robotics and Automation, Pasadena, CA, USA*, pp. 962–967.

Sandini, G. and V. Tagliasco (1980). An anthropomorphic retina-like structure for scene analysis. *Computer Graphics and Image Processing 14*(3), 365–372.

Schneider, W. X. (1995). VAM: A neuro-cognitive model for visual attention control of segmentation, object recognition, and space-based motor action. *Visual Cognition 2*(2-3), 331–376.

Scholl, B. J. (2001). Objects and attention: the state of the art. *Cognition 80*(1-2), 1–46.

Shapiro, A. G. (2008). Separating color from color contrast. *Journal of Vision 8*(1), 1–18.

Shibuya, H. and C. Bundesen (1988). Visual selection from multielement displays: Measuring and modeling effects of exposure duration. *Journal of Experimental Psychology 14*(4), 591–600.

Smet, P. D. and R. L. V. P. M. Pires (2000). Implementation and analysis of an optimized rainfalling watershed algorithm. *IS&T/SPIE's 12th Annual Symposium Electronic Imaging 2000: Science and Technology Conference: Image and Video Communications and Processing 3974*, 759–766.

Steil, J. J., G. Heidemann, J. Jockusch, R. Rae, N. Jungclaus, and H. Ritter (2001). Guiding attention for grasping tasks by gestural instruction: The gravis-robot architecture. In *Proc. IROS 2001*, pp. 1570–1577. IEEE.

Stewart, C. V. (1999). Robust parameter estimation in computer vision. *SIAM Review 41*(3), 513–537.

Sun, Y. (2003). Object-based visual attention for computer vision. *Artificial Intelligence 146*(1), 77–123.

Sun, Y., R. Fisher, F. Wang, and H. M. Gomes (2008). A computer vision model for visual-object-based attention and eye movements. *Computer Vision and Image Understanding 112*(2), 126–142.

Tanaka, K. (2003). Columns for complex visual object features in the inferotemporal cortex: clustering of cells with similar but slightly different stimulus selectivities. *Cerebral Cortex 13*(1), 90–99.

Tatler, B. W., M. M. Hayhoe, M. F. Land, and D. H. Ballard (2011). Eye guidance in natural vision: Reinterpreting salience. *Journal of Vision 11*(5), 1–23.

Theeuwes, J. (2004). Top-down search strategies cannot override attentional capture. *Psychonomic bulletin review 11*(1), 65–70.

Torralba, A., A. Oliva, M. S. Castelhano, and J. M. Henderson (2006). Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological review 113*(4), 766–786.

Treisman, A. M. and G. Gelade (1980). A feature-integration theory of attention. *Cognitive Psychology 12*(1), 97–136.

Treisman, A. M., M. Sykes, and G. Gelade (1977). *Selective Attention and Stimulus Integration*, Chapter 17, pp. 333–361. Hillsdale, NJ: Lawrence Erlbaum Associates.

Tsotsos, J. K., S. M. Culhane, W. Y. K. Winky, Y. Lai, N. Davis, and F. Nuflo (1995, October). Modeling visual attention via selective tuning. *Artificial Intelligence 78*(1-2), 507–545.

Van Essen, D. and C. Anderson (1995). Information processing strategies and pathways in the primate visual system. In S. Zornetzer, J. Davis, C. Lau, and T. McKenna (Eds.), *An Introduction to Neural and Electronic Networks*, pp. 45–76. Academic Press.

Vincent, B. T., T. Troscianko, and I. D. Gilchrist (2007). Investigating a space-variant weighted salience account of visual selection. *Vision Research 47*(13), 1809–1820.

Vitu, F. (2008). About the global effect and the critical role of retinal eccentricity: Implications for eye movements in reading. *Journal of Eye Movements Research 2*(3), 1–18.

Walther, D. (2006). *Interactions of Visual Attention and Object Recognition : Computational Modeling , Algorithms , and Psychophysics*. Ph. D. thesis, California Institute of Technology.

Walther, D., L. Itti, M. Riesenhuber, T. Poggio, and C. Koch (2010). Attentional selection for object recognitiona gentle way. In *Biologically Motivated Computer Vision*, pp. 251–267. Springer.

Walther, D. and C. Koch (2006). Modeling attention to salient proto-objects. *Neural Networks 19*(9), 1395 – 1407.

Walther, D., U. Rutishauser, C. Koch, and P. Perona (2005). Selective visual attention enables learning and recognition of multiple objects in cluttered scenes. *Computer Vision and Image Understanding 100*(1-2), 41 – 63. Special Issue on Attention and Performance in Computer Vision.

Watson, A. B. (1983). Detection and recognition of simple spatial forms. Technical report, NASA Ames Research Center.

Weber, C. and J. Triesch (2009). Implementations and implications of foveated vision. *Recent Patents on Computer Science 2*(1), 75–85.

Wischnewski, M., A. Belardinelli, W. X. Schneider, and J. J. Steil (2010). Where to look next? combining static and dynamic proto-objects in a tva-based model of visual attention. *Cognitive Computation 2*, 326–343.

Wischnewski, M., J. J. Steil, L. Kehrer, and W. X. Schneider (2009). Integrating inhomogeneous processing and proto-object formation in a computational model of visual attention. In *Human Centered Robot Systems*, pp. 93–102.

Wolfe, J. M. (1994). Guided search 2.0 a revised model of visual search. *Psychonomic Bulletin & Review 1*(2), 202–238.

Wolfe, J. M. and T. S. Horowitz (2004). What attributes guide the deployment of visual attention and how do they do it? *Nature Reviews Neuroscience 5*(6), 495–501.

Zelinsky, G. J. (2008). A theory of eye movements during target acquisition. *Psychological review 115*(4), 787–835.

Zelinsky, G. J., W. Zhang, B. Yu, X. Chen, and D. Samaras (2006). The role of top-down and bottom-up processes in guiding eye movements during visual search. *New York 18*, 1569–1576.

Zhang, Z. (1995). Parameter estimation techniques: A tutorial. Technical report, 2676, INRIA.