A Dynamical Systems Approach to Visual Attention Based on Saliency Maps

Timothy Randall Rost

Master of Science
School of Informatics
University of Edinburgh
2004

Abstract

Saliency maps provide a biologically plausible model of visual attention based on parallel preattentive features. The goal of past research with saliency maps often is to find regions of interest in a scene under various conditions or top-down effects. Recent publications suggest learning the significance of preattentive feature from visual scanpaths. Our research implements a computational model of saliency maps based on dynamical systems and then proposes a method of recovering feature weights from points of focal attention. Performance of the learning model is evaluated by comparing learnt focal attention to the training data. Finally, suggestions are made for improving the learning system during future research.

Acknowledgements

I would like to thank my supervisor Sethu Vijayakumar for providing insight and guidance throughout this research. Thank you to family and friends who supported my decision to leave a career in semiconductors and pursue robotics. And finally a special thanks to Katrina Wendel for always encouraging me to follow my goals and ambitions.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Timothy Randall Rost*)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

By existing in the physical world, we as humans are subject to vast amounts of sensory input. Indeed the unified theory of information describes the environment as composed purely of information, only some of which is returned by one of the four senses. This is apparent when comparing capabilities of identical sensors across species. Considering a flower, the ultraviolet colours seen by an insect create a vastly different picture from what humans observe. Since viewing the same world leads to different understandings, the senses can be considered biological filters selectively sampling from the seemingly infinite pool of available information. But even with these natural filters there remains an excessive amount of complex data to fully analyse every bit in detail.

The visual system transforms incoming light into a descriptive representation of the environment. However, the resulting image is excessively complex to fully model and effectively process. A common example is a treed nature scene. Each tree has many branches and leaves, and in turn each of these is defined by a combination of shape, colour, and texture. Completely representing every scene detail as an internal model is arguably impossible. Instead the image is further simplified to contain only the most relevant information, where relevancy is often task-based. For instance, walking through a wooded area requires only spatial tree location while allowing the leaves to be ignored (Parkhurst, 2002).

Attention directs focus to only a portion of the available information and has long been compared to a spotlight illuminating a small area with high resolution (Eriksen and Hoffman, 1972; Treisman and Gelade, 1980). In addition to spatially selecting the most interesting regions, attention controls information flow through the visual neural pathway and subsequently shifts focus between important parts of the scene (Tsotsos et al., 1995). Within selective attention there are two mechanisms to decide what constitutes interesting and relevant: top-down and bottom-up control.

Top-down control volitionally directs attention in order to complete some goal. Often associated with visual search, top-down control requires varying degrees of complexity according to the task. Single feature recognition triggers an automatic response because it is an essentially free product of the visual processing system. Targets based on a conjunction of features require a serial search of responses and possibly interaction with higher level cortical functions for object recognition (Treisman and Gelade, 1980). Top-down visual attention has historically been difficult to model and implement in a system because it is subjectively based on the task and individual.

The second attentional mechanism, bottom-up control, has a more analytical foundation. Scenes are decomposed into a set of fundamental visual features such as hue, intensity, orientation, and motion. The features, also known as cues, correspond directly to capabilities found within the first level of visual processing and result in a measure of interest for locations in the scene. This measure is known as saliency. Unlike top-down, bottom-up control is a subconscious operation where the points of highest saliency "pop-out" of the background and capture focal attention (Treisman and Gelade, 1980; Braun and Julesz, 1998).

The key difference between the two schemes is that of conscious versus subconscious control, but in reality focal attention is most often a balance between bottom-up and top-down mechanisms. Even in a demanding visual search the bottom-up system remains aware of changes in the scene that could signal danger or prompt a change of task. An example is attention captured by a bright flash of light in the peripheral vision.

One preferred model of bottom-up control is found in saliency maps. First proposed by Christof Koch and Shimon Ullman (Koch and Ullman, 1984, 1985), saliency maps provide a biologically plausible model of attentional selection based on visual features.

1.1 Project

1.1.1 Research Focus

It is understood that not all features contribute equally to attentional selection and that some elements are more heavily weighted. In past work with saliency maps, feature weighting has been ignored, chosen to detect specific target, or roughly approximated from image data. The goal of this project is to create a biologically plausible visual attentional system based on bottom-up saliency maps and learn the relative weighting of features from eye tracking data. By examining the way features combine in saliency maps, we hope to contribute to a better understanding of the biological foundation of visual attention.

1.1.2 Thesis Overview

Chapter 1 provides an introduction to attention and the motivation for undertaking a study of saliency maps and feature weights.

Chapter 2 discusses key background research in focal attention and presents a theoretical review of saliency maps.

Chapter 3 explores the computational attention model developed for this project. Included are methods, parameter selection, and observations.

Chapter 4 looks at a method of learning saliency map feature weights with the least squares regression algorithm.

Chapter 5 details the eye tracking hardware, software implementation, and open source libraries use in the project.

Chapter 6 presents experiments where feature weights are learnt from points of focal attention. The chapter analyses results for each experiment.

Chapter 7 relates experimental findings to the computational model and learning system. Opportunities for future research are mentioned.

Chapter 8 concludes the paper by summarising the results and detailing the next step in continuing the research.

Chapter 2

Background

2.1 Introduction

This chapter provides an historical overview of several key studies in attention. Characteristics of attention are defined followed by three theories leading to the saliency map model implemented for this research. The principles of the model are next discussed, ending with four approaches to construct the saliency map from elementary features.

2.2 Highlights of Attentional Research

Psychologists, cognitive scientists, and engineers extensively studied focal attention during the twentieth century. Sigmund Freud stated in his influential work *The Interpretation of Dreams* that the state of consciousness depends on the function of focused attention (Freud, 1915). The importance of attention was later reiterated during the late 1950s when the psychologist Schachtel published in his work *Metamorphosis* that focal awareness is the highest form of consciousness and the basis of perception (Schachtel, 1959).

2.2.1 Attention Characteristics

Studying children at play led Schachtel to believe that understanding reality is not a biological need but rather a result of expressing interest in the environment. From this research he defined several characteristics of attention: attention is directional, attention is placed on a particular object, thought, or feeling, and attention excludes from consciousness anything not of focus (Schachtel, 1959). The first point was popularised by the spotlight analogy; however, the third point later came under question with some studies showing support (Posner, 1980) and others suggesting that targets of unary features can be identified even with attention directed elsewhere (Treisman and Gelade, 1980).

Schachtel further stated that focal attention implies mentally taking hold of an object and working to fully understand it from many sides. To accomplish this, attention is directed not in a single sustained act but rather through several shorter approaches where each examines a different aspect or relation (Schachtel, 1959). Work on mentally understanding objects was a prelude to later studies about the importance of visual attention for object analysis and recognition.

2.2.2 Filter Theory of Selective Attention

Working on object identification and classification during the early 1960s, Minsky (1961) noted how passive classification becomes less adequate with complicated problems. He further reasoned that visual identification requires full attention to segmented parts of an image. Concerned with the same problem, Broadbent (1958) published the filter theory of selective attention. The theory proposes a two stage framework of visual processing. To solve problems of information complexity, the first stage computes simple features in parallel across the entire field. The second stage involves higher level analysis such as object recognition. As the second stage is much more computationally intensive, it receives only a portion of the available data from the first stage. Selecting a data subset is handled by an attention mechanism and any remaining information is discarded (Broadbent, 1958).

2.2.3 Information Processing Approach

Early studies regarded attentional selection as a form of energy allocation (Solley and Murphy, 1960). During the mid 1960s, Ulrich Neisser (1966) instead proposed an information processing stance where attention is defined as an “allotment of analysing mechanisms to a limited region of the field.” Following a dual-layer architecture outlined by Broadbent, Neisser organised preattentive processes that operate globally on an image into a single layer of parallel operations. Preattentive processes, he reasoned, directly control two general classifications of movement: focal redirection based on extracted cues and guided movements such as walking.

2.2.4 Feature Integration Theory

Feature integration theory developed by Anne Treisman further outlined the importance and method of preattentive feature extraction. According to the theory, simple features are automatically found in parallel across the entire visual field and separated by colour, orientation, spatial frequency, intensity, and direction of motion. Importantly, extraction of simple features is relatively unaffected by the presence of detractors. The separate representations are later combined with the resulting salient locations attended in serial (Treisman and Gelade, 1980).

A notable component of the theory is that focal attention is required to detect any objects defined by a conjunction of properties. Treisman proposed that focal attention is the method for merging features into singular objects, and that without either attention or top-down criteria, incorrect combinations can form illusionary results (Treisman and Gelade, 1980).

The same year a paper by Michael Posner explained how eye saccade movements can be directed by preattentive results prior to full awareness of the stimuli. In experiments, subjects exhibited saccadic eye movement toward a region yet were unable to describe the contents. A comparison of orienting from memory and to external stimuli noted that while eye saccades can be driven by input, search movements are driven by an internal search plan (Posner, 1980). One form of search plan is a feature weight model directing visual attention.

2.3 Saliency Model

Koch and Ullman (1984, 1985) developed saliency maps to provide an explanation of bottom-up attentional shift while leveraging the existing two-stage model. Figure 2.1 provides a schematic view of the model. Similar to feature integration theory, the visual scene is first decomposed along parallel paths into topographic feature maps. Called *early representations* in the first model descriptions, these provide spatial information about hue, intensity, and edge orientation. Some implementations also mention motion as a basic feature (Koch and Ullman, 1984; Hikosaka et al., 1996) but the majority concentrate on static images.

2.3.1 Colour-Opponents

In our visual system, the retina contains three types of cones sensitive to long, medium, and short wavelengths. Named L, M, and S respectively, the lateral geniculate nucleus (LGN) combines information from the cones into colour-opponents between specific cone pairs. Two classes of chromatic opponency neurons exist in the retina and LGN. Red-green opponent neurons respond to differences between long and medium cones while blue-yellow neurons compute the difference between short cones and luminance. A third type of neuron determines luminance by summing long and medium cones (Engel et al., 1997; Schluppeck and Engel, 2002). Opponency is mathematically summarised below. Experiments with functional magnetic resonance imaging showed that the response of red-green and blue-yellow colour-opponents is much greater than that of luminance stimuli (Engel et al., 1997). The results were later confirmed by additional studies into neural signals (Schluppeck and Engel, 2002).

$$\begin{array}{ll}
\text{red-green opponents:} & L-M \\
\text{blue-yellow opponents:} & S-(L+M) \\
\text{intensity opponents:} & L+M
\end{array}$$

2.3.2 Centre-Surround

Neurons in the visual cortex strongly depend on both the stimuli and its surrounding region. It is not so much the stimuli itself but rather its contrast with the surround that

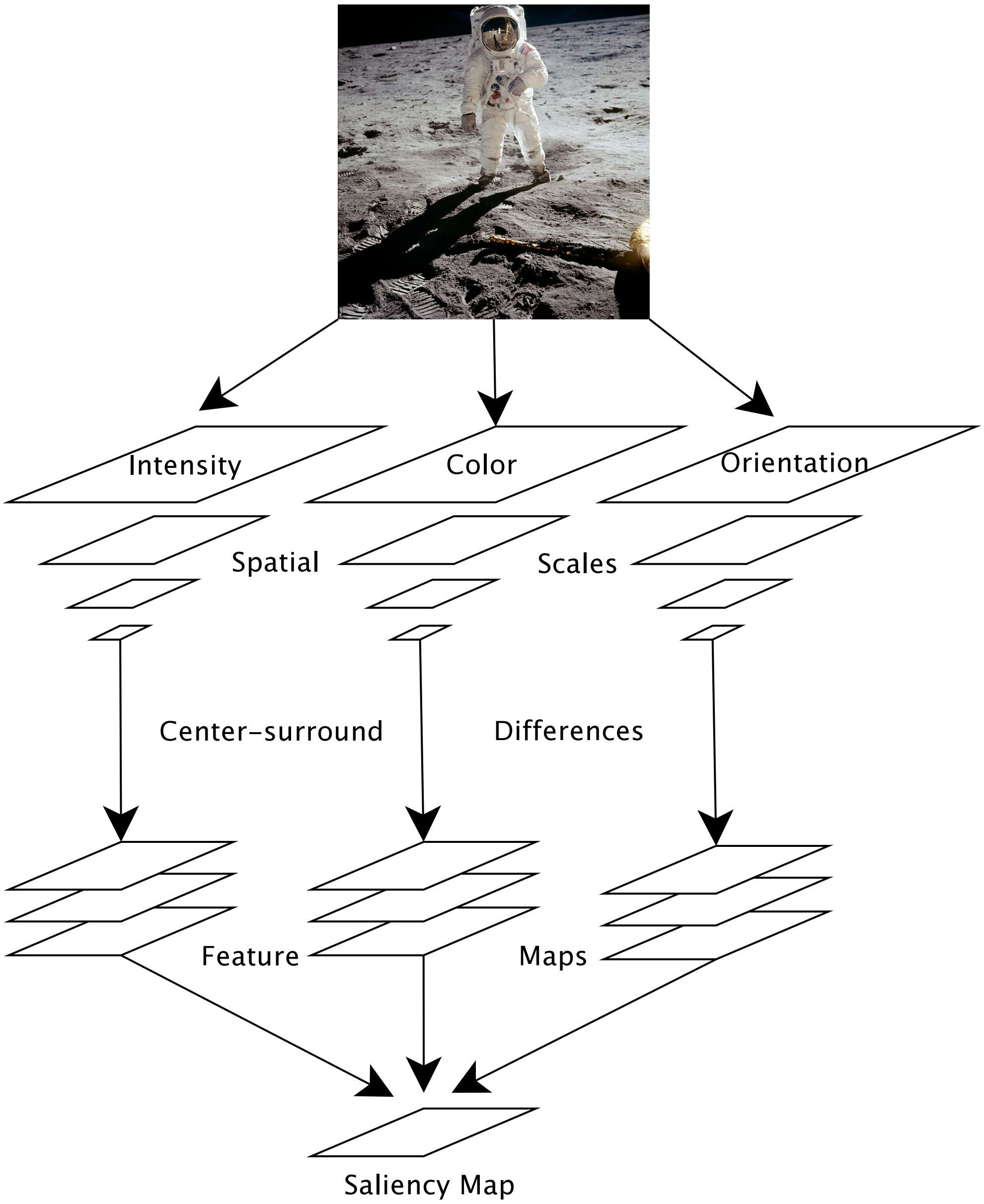


Figure 2.1: Schematic model of the saliency map architecture.

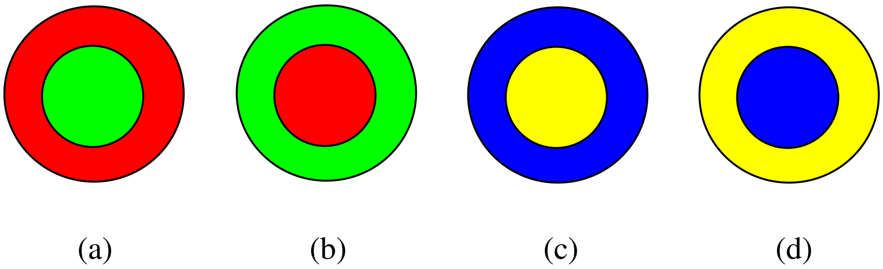


Figure 2.2: Colour opponent cells (a) red centre / green surround, (b) green centre / red surround, (c) yellow centre / blue surround, and (d) blue centre / yellow surround.

elicits the greatest response. But while high contrast between centre and surround has a reinforcing effect, low contrast inhibits response (Levitt and Lund, 1997). The retina, LGN, and primary visual cortex act in concert to detect regions that stand out from the surround (Itti et al., 1998). Figure 2.2 illustrates the centre-surround representation of red-green and blue-yellow colour opponents.

Gaussian pyramids provide a natural way of implementing a centre-surround process for colour-opponents and intensity. Each pyramid level represents an increasingly low-pass filtered sample of the feature map (Burt and Adelson, 1983) The cross-spatial difference between coarse and fine scales identifies scene locations that stand out strongly from the surrounding area (Itti et al., 1998). Centre-surround maps are created for each feature at multiple spatial scales.

2.3.3 Combining Features

With centre-surround differences representing salient locations for each feature, the results are then combined into a single global saliency map. Combination is inherently difficult as features represent non-comparable criteria. For instance, there is not an obvious quantitative relationship between units of red hue and motion. Four general methods of combination have been proposed and below each is discussed in turn. Itti and Koch (1999a) provide illustrative examples depicting results for each method.

2.3.3.1 Fixed Range Summation

A simple but naïve approach is to linearly sum the feature maps without preconceptions about the relative importance of each cue (Koch and Ullman, 1984). This disregards human studies showing that features contribute to varying degrees depending on the task (Francolini and Egeth, 1979; Folk et al., 1992; Wolfe, 1994). Reliance on the task however requires integration of top-down knowledge into the system while the traditional saliency map model is purely bottom-up. There are other problems with fixed range summation as well. Taking into account the multiresolution centre-surround for hue and intensity, the large number of opponent maps can effectively overwhelm other highly saliency locations present in only a few feature maps. This is the case for motion, considered to be a strong determinant of visual attention but directed by a single saliency map. Instead, the feature maps are summed across scale and normalised as a single conspicuity map for each feature. The conspicuity maps define the final saliency map(Itti et al., 1998; Funk, 2004).

2.3.3.2 Supervised Learning

When using saliency maps to detect a specific target, the fixed range summation technique is inappropriate as some features are more important than others. A gradient learning method compares response inside and outside of the target area. Features where the saliency is greater inside of the target region than outside contribute a greater proportion to the overall attention and are weighted higher accordingly. The weighted features are then summed into a single saliency map. A mathematical algorithm for learning the weight $w(M)$ of a feature map M contains three steps (Itti and Koch, 1999a).

1. Find the maximum and minimum saliency values M_{glob} and m_{glob} over the entire visual field.

2. For the target region, find the internal and external maximum saliency values M_{in} and M_{out} respectively.

3. An iterative learning routine increases the weights of feature maps having higher saliency inside of the target region. For other features the weights decay to a

small value. Given a learning rate $\eta > 0$

$$w(M) \leftarrow w(M) + \eta \frac{M_{in} - M_{out}}{M_{glob} - m_{glob}} \tag{2.1}$$

While this technique learns all feature weights simultaneously, they are target specific and can not be directly applied to other applications.

2.3.3.3 Content-based Weighting

One method of implementing an unsupervised learning scheme is by replicating intramodal competition. The goal is to lower weighting of feature maps having numerous peaks of similar saliency while increasing the response of those features having only a few strongly salient locations. All features are then combined into a single saliency map with weighted summation. Itti and Koch (1999a) present an algorithm for this approach:

1. Normalise the feature maps for relative strength comparison.
2. Find the global maximum M and average of all local maxima $\bar{m}$.
3. Multiply the feature map by $(M - \bar{m})^2$

Multiplying normalised feature maps by the squared difference between global maximum and average local maxima enhances those maps where active regions stand out strongly. The result is a weighting based on content rather than training data (Itti et al., 1998). While following the neurological principle of intramodal competition, it is not a biologically plausible implementation because average saliency requires fully interconnected neural pathways.

2.3.3.4 Iterative Local Competition

The primary visual cortex is organised with short localised connections and longer cortical connections extending 6-8 millimeters. Amari (1977) covered a biologically plausible model of intramodal competition using local neural connections to provide strong excitation and the longer connections for broad inhibition. To simulate this arrangement, feature maps are convolved by a two-dimensional difference of Gaussian

(DoG) kernel defined by equation 2.2 and illustrated in figure 2.3. Iteratively convolving normalise feature maps, as shown by equation 2.3, enhances salient locations that stand out strongly while inhibiting areas of little or uniform saliency (Itti and Koch, 1999a). Convolution introduces dynamic competition within the feature maps so that only areas of highest saliency remain. Feature maps are linearly summed to create a final saliency map.

$$DoG(x,y) = \frac{c_{exc}^2}{2\pi\sigma_{exc}^2} e^{-\frac{x^2+y^2}{2\sigma_{exc}^2}} - \frac{c_{inh}^2}{2\pi\sigma_{inh}^2} e^{-\frac{x^2+y^2}{2\sigma_{inh}^2}} \tag{2.2}$$

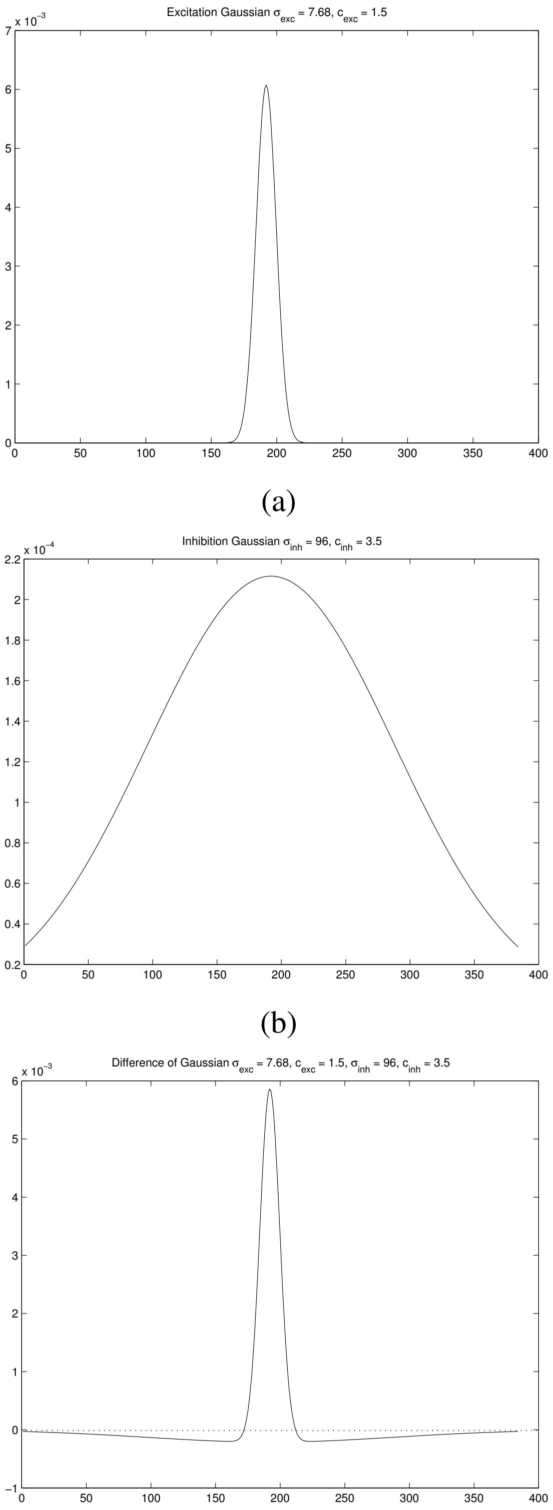
$$M \leftarrow \mid M + M * DoG - C_{inh} \mid_{\geq 0} \tag{2.3}$$

The DoG kernel consists of an inhibition Gaussian function with a large variance σ_{inh} subtracted from a excitation Gaussian of smaller variance σ_{exc}. The convolution variable C_{inh} contributes a system decay for the case when inhibition otherwise balances excitation. The literature suggests $C_{inh} = 0.2$.

2.3.4 Attentional Selection

The result of each combination strategy is a single saliency map composed of the elementary image features. Often containing several regions of interest, a winner-takes-all (WTA) process directs attention to the location of greatest saliency. Focal selection is not permanently captured by a single point but rather it is a dynamic process where gaze shifts in serial between salient locations.

The visual system implements attentional shift by imposing an inhibition of returns at the current point of focus. This locally suppresses saliency so that a new focal point is selected by the WTA network. Figure 2.4 shows a stimuli and the resulting saliency map both before and after inhibition of returns. To maintain dynamics the temporary inhibition decays over several time steps with a study by Posner (1980) reporting that subjects returned to the original fixation point after 500ms.

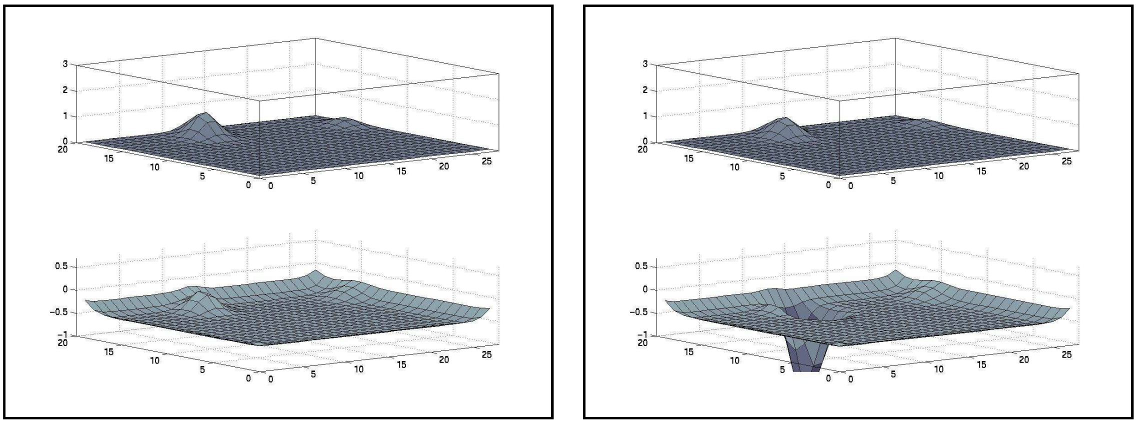


(a)

(b)

(c)

Figure 2.3: Cross section of an example (a) excitation function, (b) inhibition function, and (c) difference of Gaussian (DoG) kernel. Note the strong excitation peak and broad surrounding inhibition.



(a) (b)

Figure 2.4: Inhibition of returns before and after attentional selection.

2.4 Dynamical Systems

A dynamical system is one in which a change of state occurs over time. In a first order dynamical system the current state depends on the previous state and a velocity component computed as a scaled derivative of the input. The scaled derivative tells how much to change from the previous state.

The first order dynamical system is mathematically described by equation 2.4. If the scale parameter τ equals one then the system fully changes to the next state in one time step; with τ less than one the system changes state over successive periods. The variable $u(x,t)$ signifies the saliency map at a particular instance of time and $S(x,t)$ is an external system stimulus. The variable h represents a baseline activation level for the system. (Vijayakumar et al., 2001).

$$\tau\dot{u}(x,t) = -u(x,t) + S(x,t) + h + \sum_{x'} w(x,x')\sigma(u(x',t)) \tag{2.4}$$

Dynamical systems underlie the saliency model and provide a biologically plausible explanation for saliency competition and winner-takes-all mechanisms. While the software implements attentional selection as a maximum operator across the entire saliency map, this actually simulates a dynamical systems approach where saliency values build over time until one region surpasses an attentional threshold.

Similarly, decaying inhibition of returns is automatically handled within a dynamical systems approach. Here the τ parameter controls the rate at which inhibition decays to its zero value state.

2.5 Conclusion

The most widely accepted computational model of visual attention is that of saliency maps. Based on long standing principles found in the filter theory of selective attention and feature integration theory, saliency maps explain a biologically plausible dynamical system of attentional shift. Of the four methods for combining features into a saliency map, the more recent iterative local competition provides the most biologically sound and scalable approach. Saliency maps are inherently bottom-up though there is ongoing research into integration of top-down control.

Chapter 3

Computational Model

3.1 Introduction

The second chapter introduced the saliency map model of visual attention and several key theories leading to its development. This chapter furthers the understanding of saliency maps by undertaking a step-by-step computational implementation of the model. Illustrating the text are images generated by the completed software system.

3.2 Early Visual Features

The first stage of the computational model is creating preattentive feature maps for hue, intensity, and motion. Many previous studies of saliency maps used static images and did not consider motion; however, this implementation supports dynamic scenes captured from a video camera or MPEG file.

3.2.1 Intensity

In experiments there is often a strong correlation between intensity and attention (Koch and Ullman, 1984; Braun and Julesz, 1998; Itti and Koch, 1999b; Miau and Itti, 2001; Walther et al., 2002; VanRullen, 2003). Einhäuser and König (2003) recently published research challenging the causal role of intensity in visual attention by arguing that luminance contrast is merely an abstraction of hue. Since these findings are new and

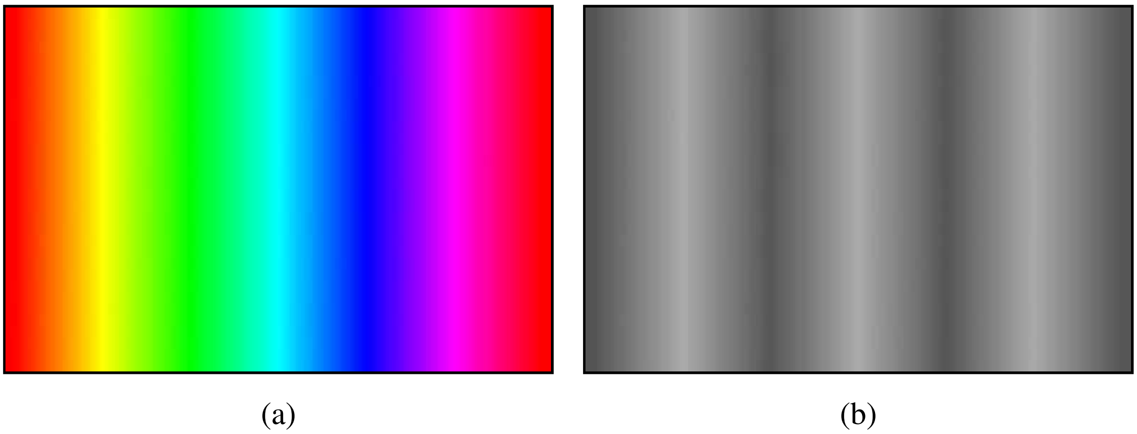


Figure 3.1: (a) A colour spectrum and (b) its intensity map. Images originate from the visual attention software.

contradict traditional understanding of visual attention, this implementation continues to use luminance as a discrete factor. A colour spectrum and its intensity image is shown in figure 3.1.

A mathematical definition of intensity for each pixel is the average of the red (R), green (G), and blue (B) colour channels.

$$I = \frac{R+G+B}{3} \tag{3.1}$$

3.2.2 Hue

To decorrelate hue from intensity, the red, green, and blue planes are first normalised by the intensity image (I) at all points where intensity is greater than 10% of its maximum value (Itti et al., 2001). This is mathematically described by (3.2)–(3.4). Variations in hue are not perceivable under very low luminance so pixels at the lower range of intensity are excluded from normalisation. Figure 3.2 shows a colour spectrum and its extracted hue planes.

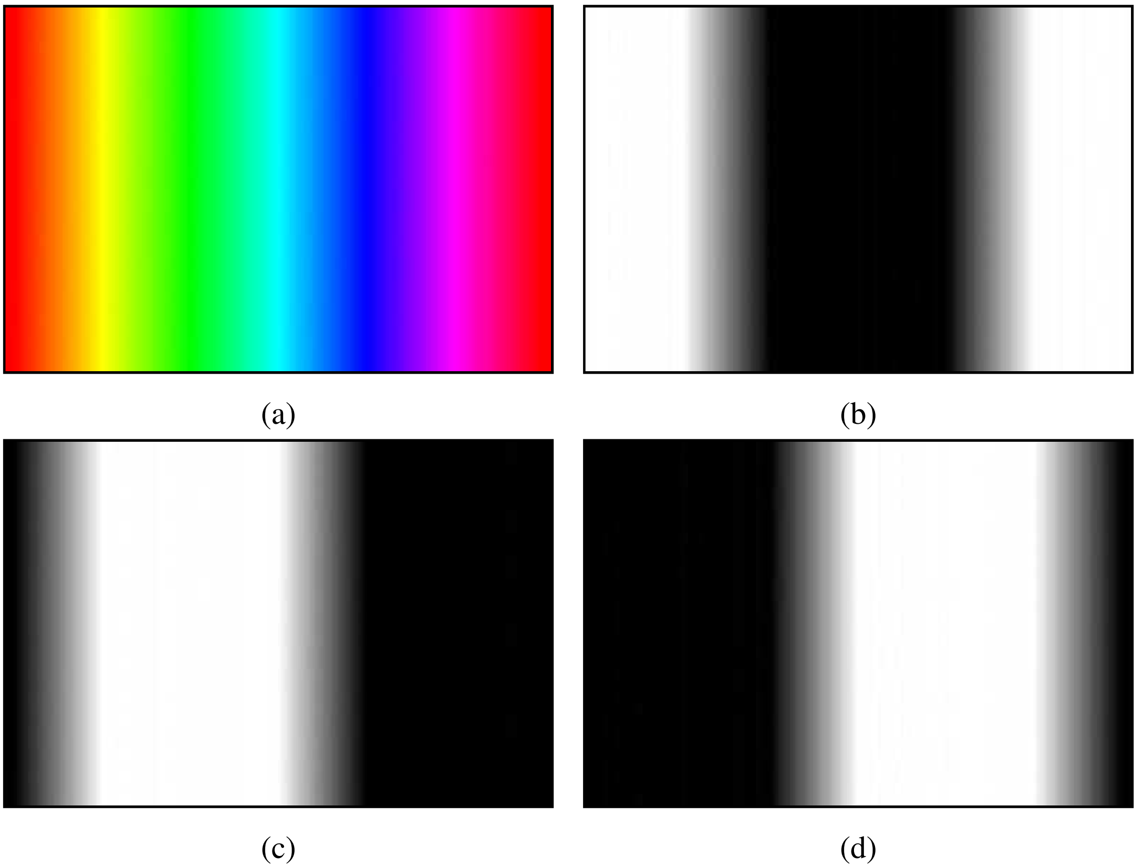


Figure 3.2: (a) A colour spectrum and its (b) red channel, (c) green channel, and (d) blue channel. Images originate from the visual attention software.

$$r = \frac{255}{3}\frac{R}{I} \tag{3.2}$$

$$g = \frac{255}{3}\frac{G}{I} \tag{3.3}$$

$$b = \frac{255}{3}\frac{B}{I} \tag{3.4}$$

Equations 3.5–3.8 create four broadly tuned colour planes with the effects of luminance removed (Itti et al., 2001). Rather than the normalised RGB planes, these hue channels are used in all further processing.

$$r' = r - \frac{g+b}{2} \tag{3.5}$$

$$g' = g - \frac{r+b}{2} \tag{3.6}$$

$$b' = b - \frac{r+g}{2} \tag{3.7}$$

$$y' = \frac{r+g}{2} - \frac{|r-g|}{2} - b \tag{3.8}$$

Equation 3.8 can be greatly simplified by comparing the three possible relations between red and green components in the numerator.

$$r > g : (r+g) - (r-g) = 2g \tag{3.9}$$

$$r < g : (r+g) - (g-r) = 2r \tag{3.10}$$

$$r == g : (r+r) - (g-g) = 2r == 2g \tag{3.11}$$

From (3.9)–(3.11), the numerator of (3.8) reduces to $2 * min(r,g)$. Equation 3.12 provides the updated derivation of the yellow channel. It is surprising to not find the simplification published in any papers.

$$y' = min(r,g) - b \tag{3.12}$$

3.2.3 Motion

A vector field measuring motion between consecutive image frames defines optical flow. With both intensity and directional information, optical flow provides a more complete motion model than simpler solutions such as the difference between frames. Dedicated hardware like that used by Vijayakumar et al. (2001) commonly implements flow calculations with a block matching method.

While optical flow yields the most complete information, this system instead relies on a motion history algorithm. A history buffer retains the difference between several successive frames. Buffered values decay over time so taking the gradient of motion history returns a reliable indicator of motion. Output from the calculation is a feature map highlighting moving objects as salient and ignoring static elements in the scene. The OpenCV library includes a robust implementation of motion history analysis.

3.3 Centre-Surround

After computing hue and intensity feature maps, the next step is to determine areas of contrast. This is accomplished with a centre-surround model of Gaussian pyramids as mentioned in the second chapter.

3.3.1 Gaussian Pyramids

Originally developed for image compression, Gaussian pyramids act as low-pass filters computing pixel values as a weighted average of localised blocks. The dimension of each pyramid level is one half that of the previous such that the interval between levels is one octave. Burt and Adelson (1983) present algorithms for Gaussian pyramids; implementations are included in the Open Computer Vision software library.

Eight-level Gaussian pyramids for the intensity and colour maps are created for centre-surround computation. Figure 3.3 shows six Gaussian pyramid levels for the broadly tuned red, green, blue, and yellow colour channels defined by equations 3.5, 3.6, 3.7, and 3.12.

3.3.2 Centre-Surround Calculation

Centre-surround is calculated as a difference across scale where scale refers to individual Gaussian pyramid levels. Because the difference is computed by matrix subtraction, all pyramid levels are first interpolated to the original image size. The centre is a pixel at scale $c \in \{2,3\}$ and the surround value is the corresponding pixel at scale $s = c + \delta$ where $\delta \in \{3,4\}$ (Itti et al., 1998). The spatial difference between two maps is often defined in literature by the operator symbol $\ominus$.

3.3.3 Intensity Centre-Surround Opponents

Using equation 3.13, four centre-surround feature maps are created for $c \in \{2,3\}$ and $\delta \in \{3,4\}$. Neural detectors are sensitive to either a bright centre and dark surround or conversely a dark centre with bright surround. Absolute value of the spatial difference

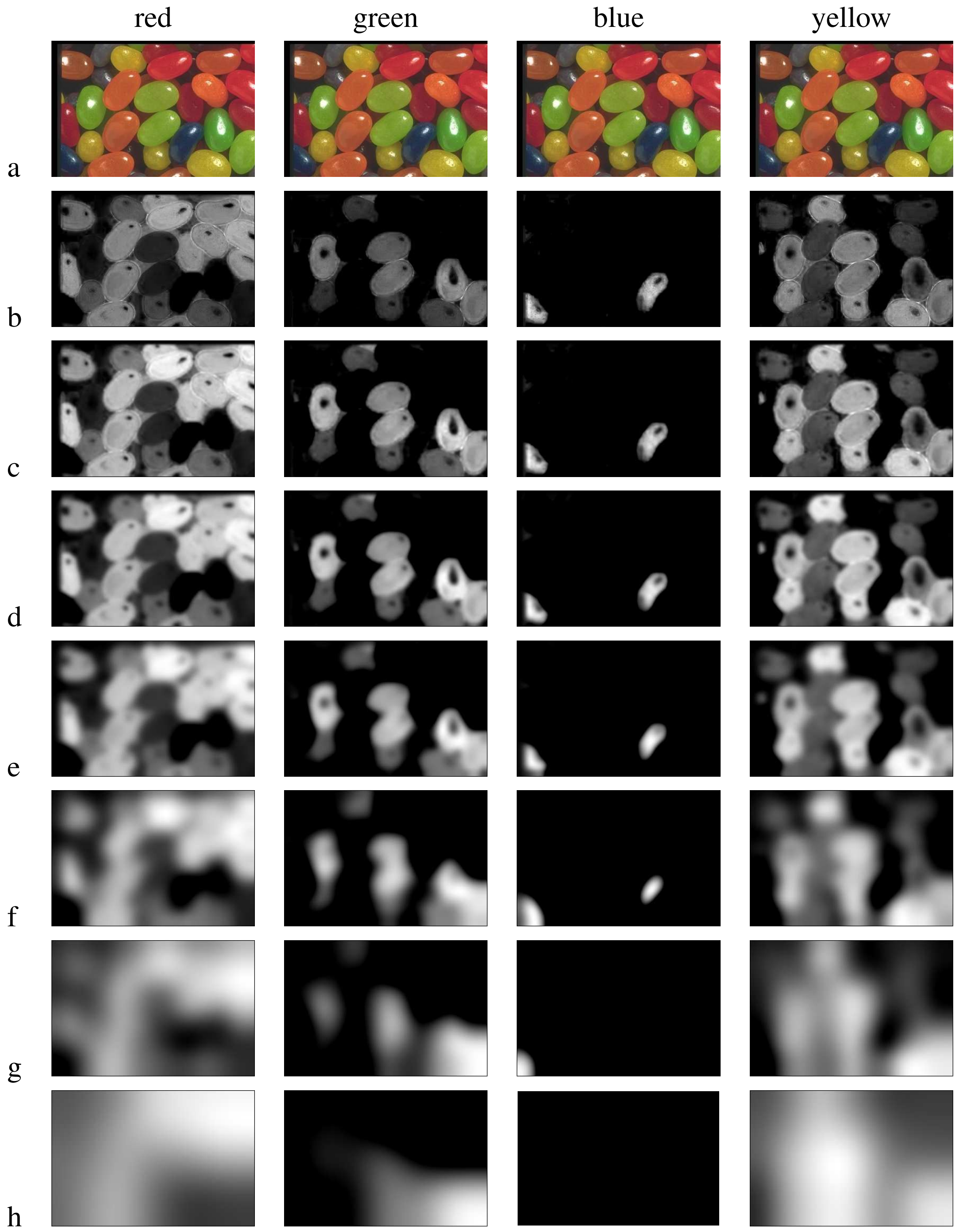


Figure 3.3: From the (a) original image the (b) red, green, blue, and yellow colour channels are created. The first six Gaussian pyramid levels (c-h) are shown. Images originate from the visual attention software.

includes both configurations in the centre-surround maps. Figure 3.4 illustrates centre-surround of intensity opponents.

$$I(c,s) = |I(c) \ominus I(s)| \tag{3.13}$$

3.3.4 Hue Centre-Surround Opponents

Red-green and blue-yellow colour-opponent maps are similarly created with the same spatial scales as intensity. Either red or green (blue or yellow) exists as the centre with the opposing colour in the surround. Centre-surround map $\mathcal{RG}(c,s)$ defined by equation 3.14 encompasses both red/green and green/red opponency while $\mathcal{BY}(c,s)$ includes blue/yellow and yellow/blue opponency in (3.15). Figures 3.5 and 3.6 show hue centre-surround opponents generated for red-green and blue-yellow respectively.

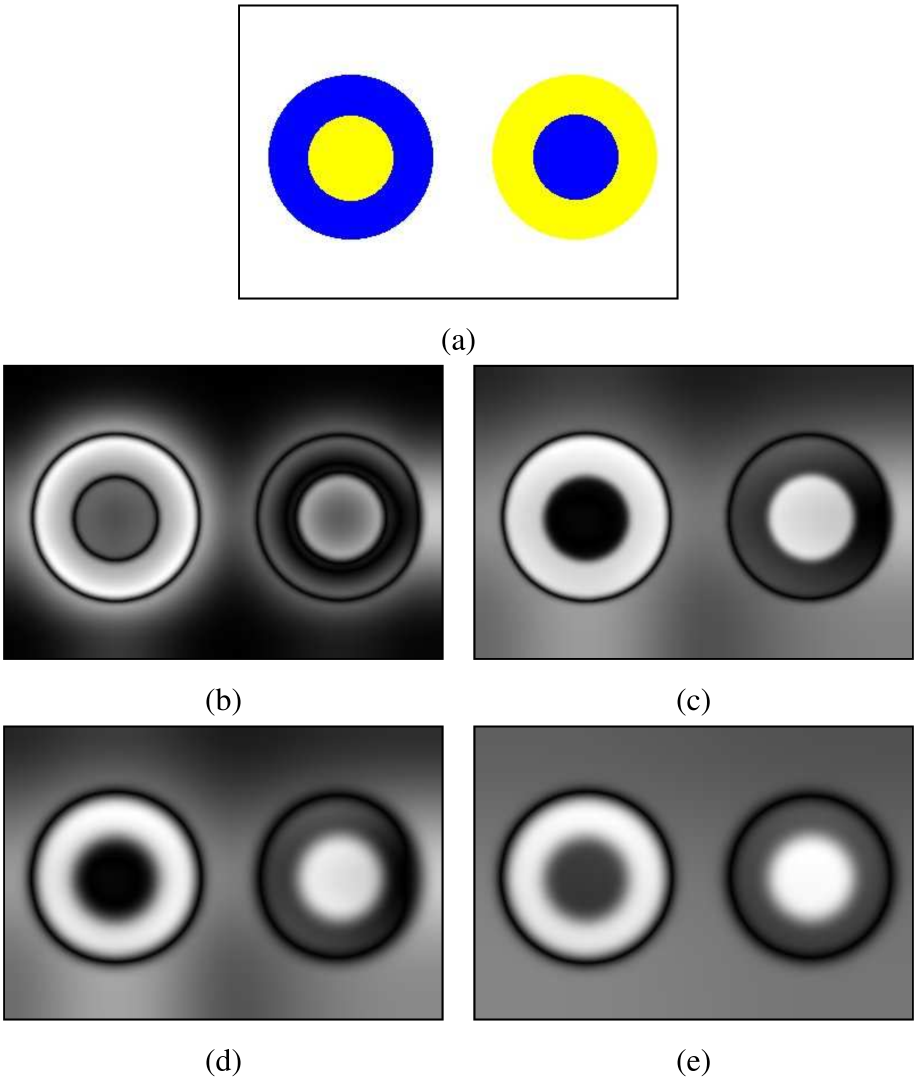
$$\mathcal{RG}(c,s) = |(R(c) - G(c)) \ominus (G(s) - R(s))| \tag{3.14}$$

$$\mathcal{BY}(c,s) = |(B(c) - Y(c)) \ominus (Y(s) - B(s))| \tag{3.15}$$

3.4 Difference of Gaussian

Chapter two described several approaches to create a saliency map from multiple features. This implementation uses iterative convolution with a difference of Gaussian function because it provides a biologically plausible dynamical system of feature competition.

Normalised red-green opponent, blue-yellow opponent, intensity opponent, and motion feature maps are iteratively convolved by a large difference of Gaussian (DoG) kernel having strong local excitation and weaker broad inhibition. Application of the DoG dynamically reinforces regions of high saliency while simultaneously suppressing areas of lower or uniform saliency. Competition also eliminates noise in the feature maps. Shown in Figure 3.7, there are three bands of interaction between salient areas. Stimuli located between X_0 and X_a reinforces the X_0 value. Between X_a and X_b stimuli have an inhibitory effect while salient regions beyond X_b cause negligible change to X_0 (Amari, 1977).

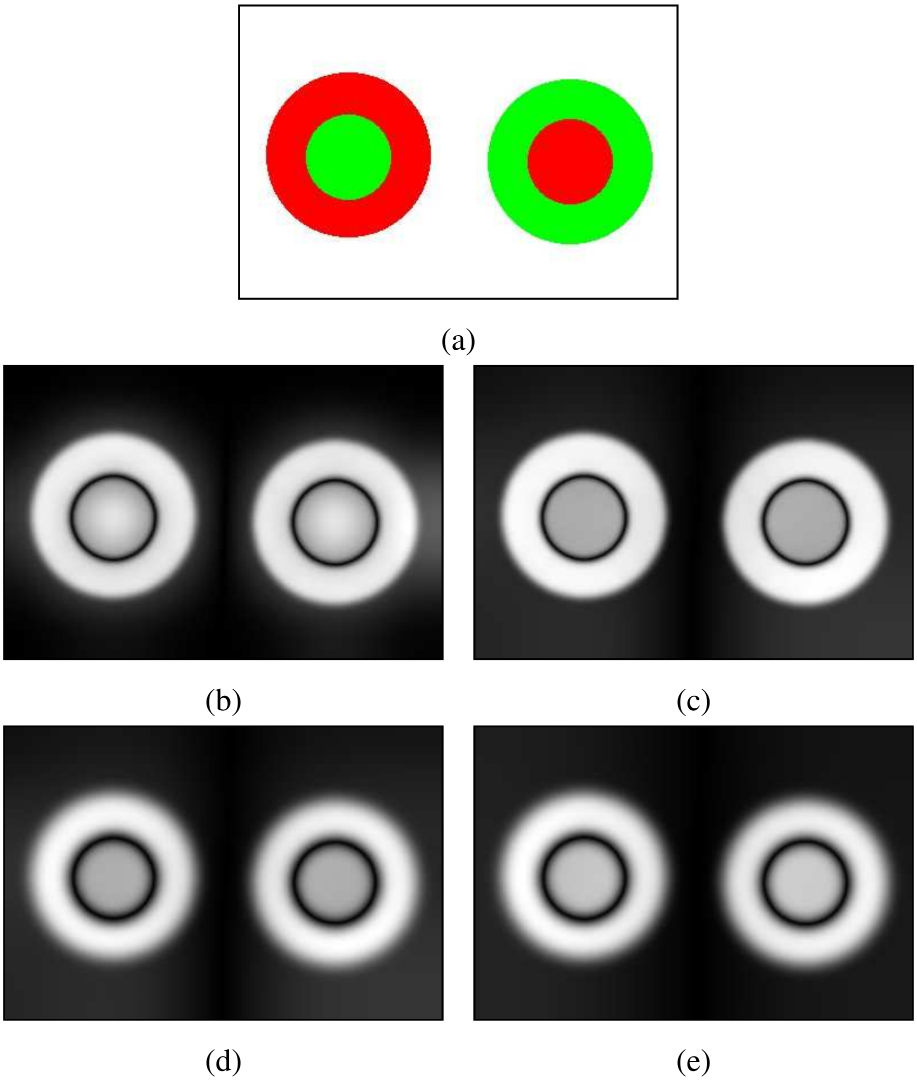


(a)

(b) (c)

(d) (e)

Figure 3.4: Intensity opponents at multiple centre-surround spatial scales: (a) original opponent image, (b) c=2 s=5, (c) c=2 s=6, (d) c=3 s=6, and (e) c=3 s=7. Images originate from the visual attention software.

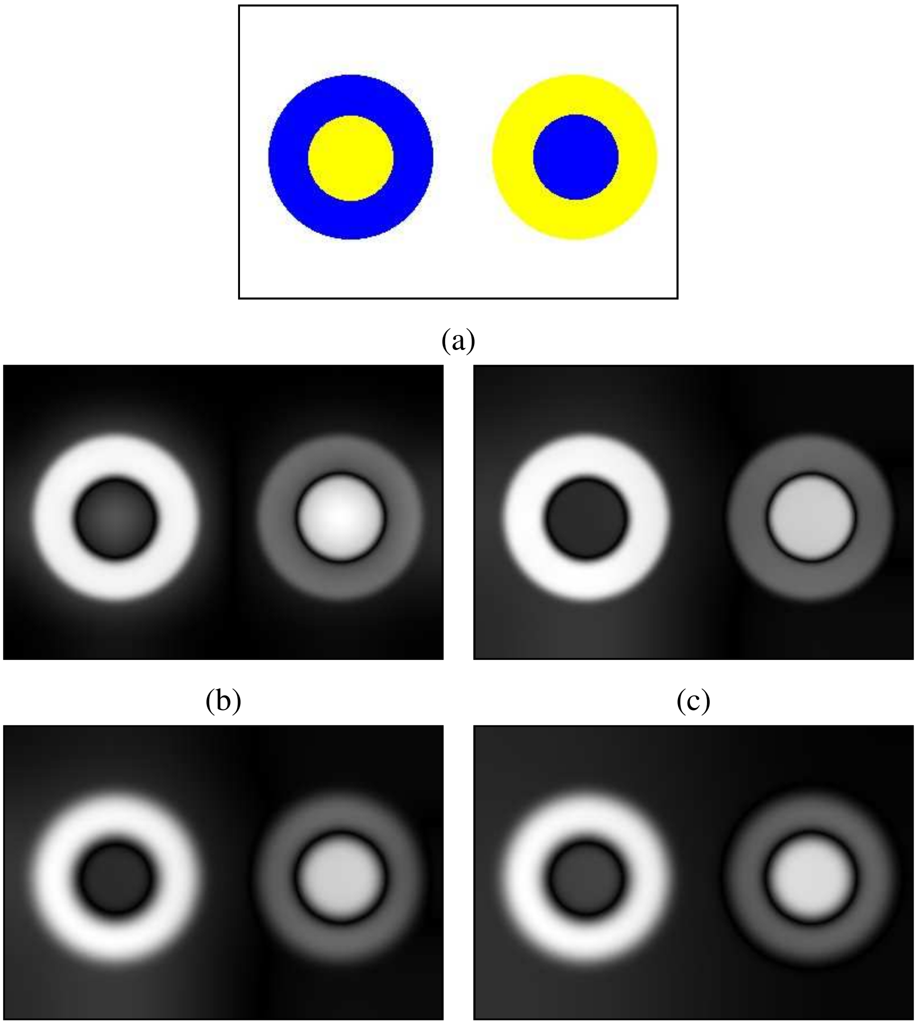


(a)

(b) (c)

(d) (e)

Figure 3.5: Colour opponents for red-green and green-red at multiple centre-surround spatial scales: (a) original opponent image, (b) c=2 s=5, (c) c=2 s=6, (d) c=3 s=6, and (e) c=3 s=7. Images originate from the visual attention software.



(a)

(b)

(c)

(d)

(e)

Figure 3.6: Colour opponents for blue-yellow and yellow-blue at multiple centre-surround spatial scales: (a) original opponent image, (b) c=2 s=5, (c) c=2 s=6, (d) c=3 s=6, and (e) c=3 s=7. Images originate from the visual attention software.

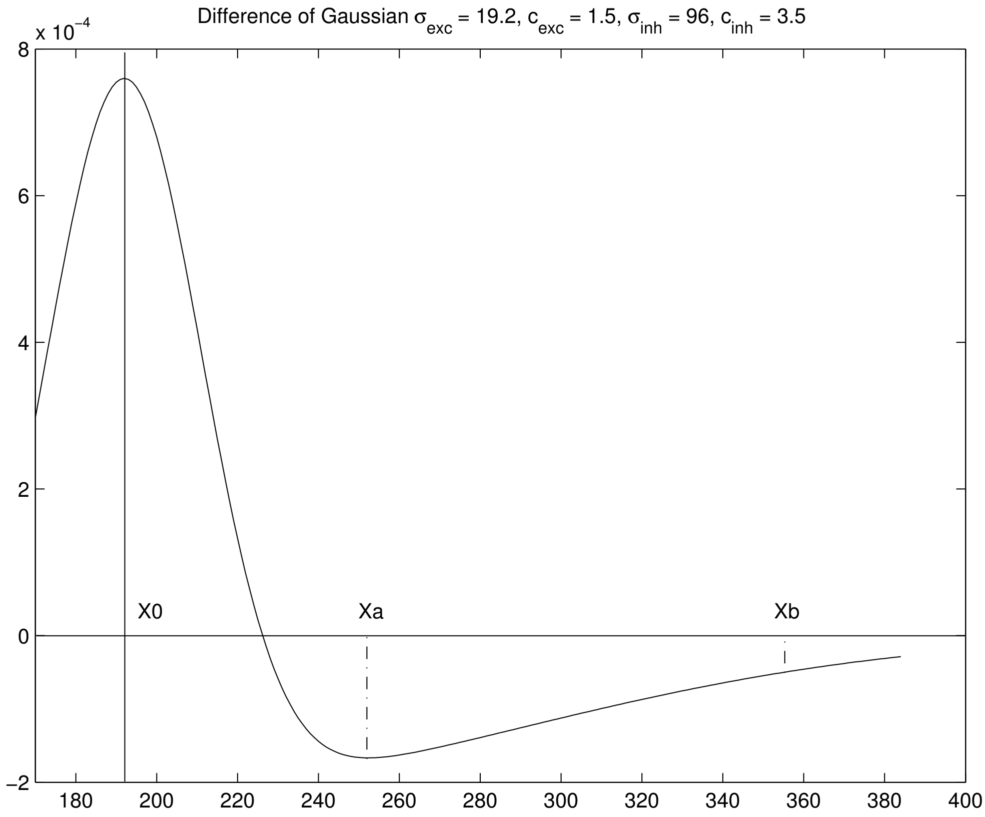


Figure 3.7: Difference of Gaussian function showing three regions of interaction. Between X0 and Xa stimuli are excitory while between Xa and Xb the same stimuli is inhibitory. Stimuli have negligible effect beyond Xb.

3.4.1 Gaussian Function

The difference of Gaussian kernel provides a strong cumulative effect over multiple iterations. Selecting excitation and inhibition Gaussian functions requires insight into the size and proximity of expected targets. Narrow excitation regions with broad, shallow inhibition are most appropriate for natural scenes where the targets are small and may be closely spaced. Intra-feature competition with such a kernel requires a large number of iterations to suppress noise and isolate salient regions; however, a greater number of features are retained. Assuming all other Gaussian parameters remain constant, a wider DoG excitation function results in a stronger inhibition response that enables fewer convolution iterations. This is best suited to large and well defined targets.

Itti and Koch (1999a) suggest an excitation function with $c_{exc} = 0.5$ and $\sigma_{exc} = 2\%$ of the image width paired with an inhibition function with $c_{inh} = 1.5$ and $\sigma_{inh} = 25\%$ of the image width. Plotted in figure 3.8, this results in a narrow excitation band of shallow inhibition. With the suggested parameter values, they found ten iterations yields the best tradeoff between computation time and effectiveness. Attempting to decrease run time, we reduced the required number of convolutions by strengthening the excitation and inhibition functions. Setting $c_{exc} = 1.5$ and $c_{inh} = 3.5$ increases the DoG kernel excitation value by a factor of ten and allows the number of convolutions to be halved from ten to five.

To test several Gaussian functions, we restricted the search space by fixing the inhibition standard deviation at $\sigma_{inh} = 25\%$. Figures 3.9, 3.10, and 3.11 show multiple convolutions of an image using $\sigma_{exc} = 2\%$ and $\sigma_{exc} = 5\%$ kernels. The first test examines response of a kernel with $c_{exc} = 1.5$, $c_{inh} = 3.5$, and $\sigma_{exc} = 2\%$. A second test retains $c_{exc} = 1.5$ and $c_{inh} = 3.5$ but increases σ_{exc} to 5% of image width. For comparison, a third test uses $c_{exc} = 0.5$ and $c_{inh} = 1.5$ with $\sigma_{exc} = 5\%$. As expected, the stronger Gaussian kernels resulted in much faster feature suppression. Most visible in the red-green opponent maps of figure 3.9, the first convolution with the stronger kernels is roughly equivalent to ten iterations with the third set of values. The blue-yellow and intensity feature maps of figures 3.10 and 3.11 show a risk of over-suppression where smaller salient features are discarded. Further tests confirmed over-suppression of small targets in the experimental scenes.

Based on the results, we selected a $\sigma_{exc} = 5\%$ curve using the original $c_{exc} = 0.5$ and $c_{inh} = 1.5$ values proposed by Itti and Koch. In scenes with distinctly salient regions and limited noise, the number of convolutions can be reduced to five without any adverse effect. Natural scenes are better suited to maintain $\sigma_{exc} = 2\%$ with ten iterations. The complete difference of Gaussian function is shown in equation 3.16. Excitation and inhibition sigma values assume an image width of 384 pixels as used in this system.

$$DoG(x,y) = \frac{(0.5)^2}{2\pi * (19.2)^2} e^{-\frac{x^2+y^2}{2*(19.2)^2}} - \frac{(1.5)^2}{2\pi * 96^2} e^{-\frac{x^2+y^2}{2*(96)^2}} \tag{3.16}$$

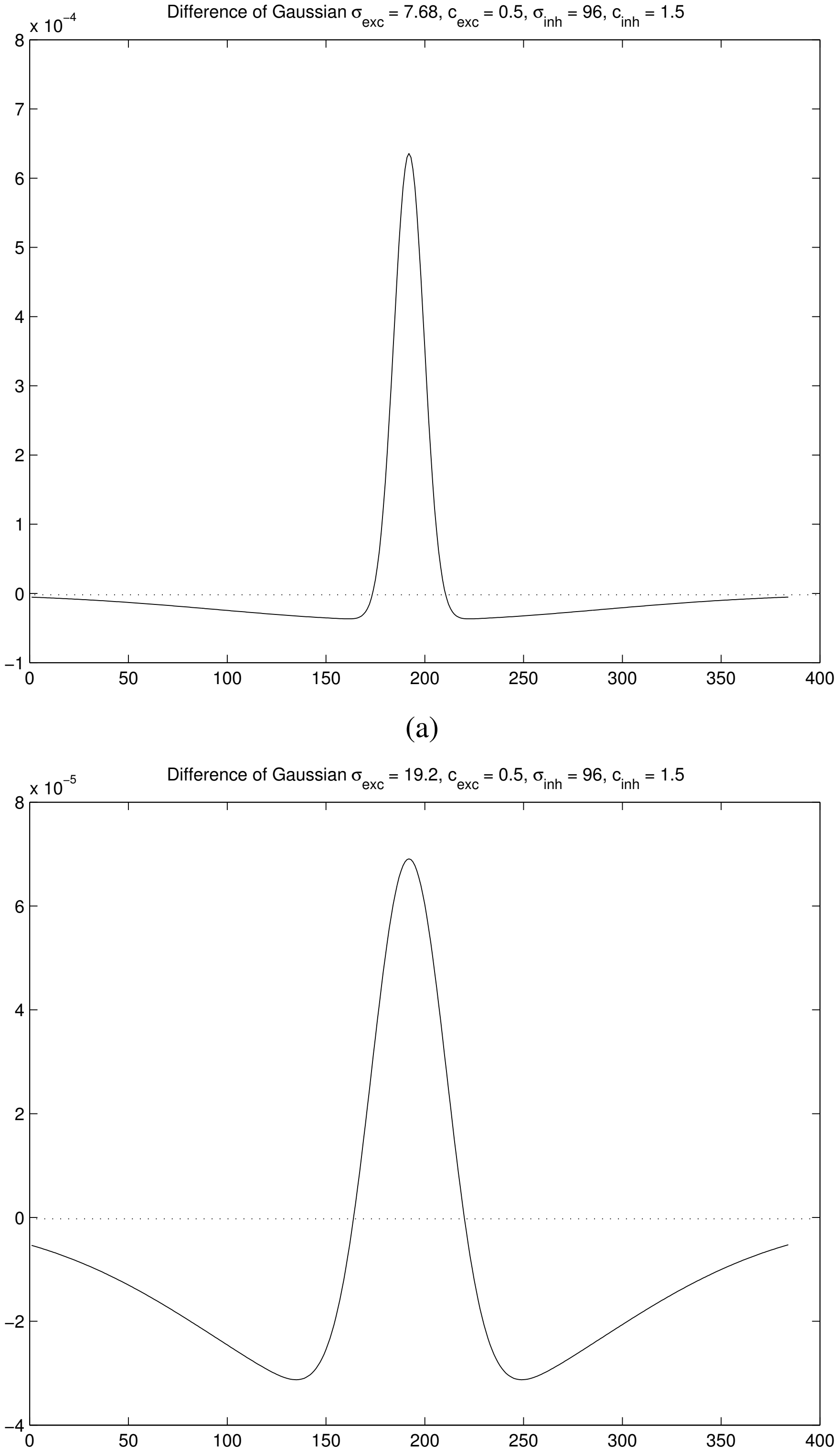
3.4.2 Convolution Algorithm

Convolving an image steps a weighting kernel across every pixel. An algebraic implementation in the image domain required in excess of two hours to complete ten iterations of a 768x768 pixel kernel on a 768x512 pixel image. For performance we instead selected to work in the signal domain using fast Fourier transforms. Convolution in the signal domain completes in a fractional second, with numerical results presented in chapter five. There are three steps to the FFT convolution algorithm (Press et al., 1992):

1. Transform both the image and kernel.
2. Multiply the two Fourier transforms component by component.
3. Compute the inverse Fourier transform to convolve the image.

3.5 Winner-Takes-All Network

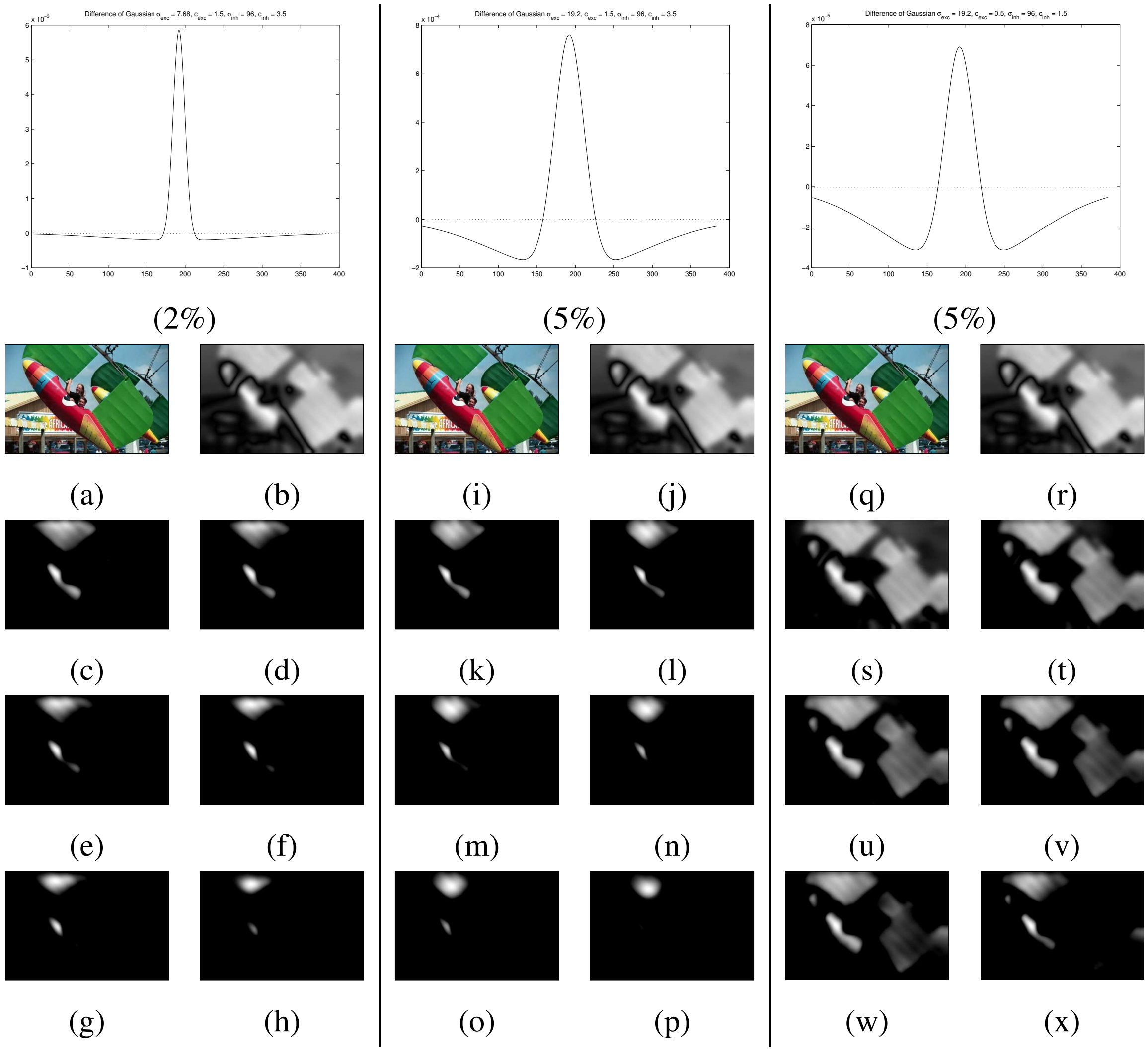
After convolution there are four red-green opponent maps, four blue-yellow maps, four intensity feature maps, and a single motion feature map. The opponent maps are first linearly summed across scale to return a single conspicuity map for each feature. Motion already exists as a single map so is used directly. There are three reasons for



(a)

(b)

Figure 3.8: Difference of Gaussian kernel (a) selected by Itti and Koch (b) used in this project.



(2%) (5%) (5%)

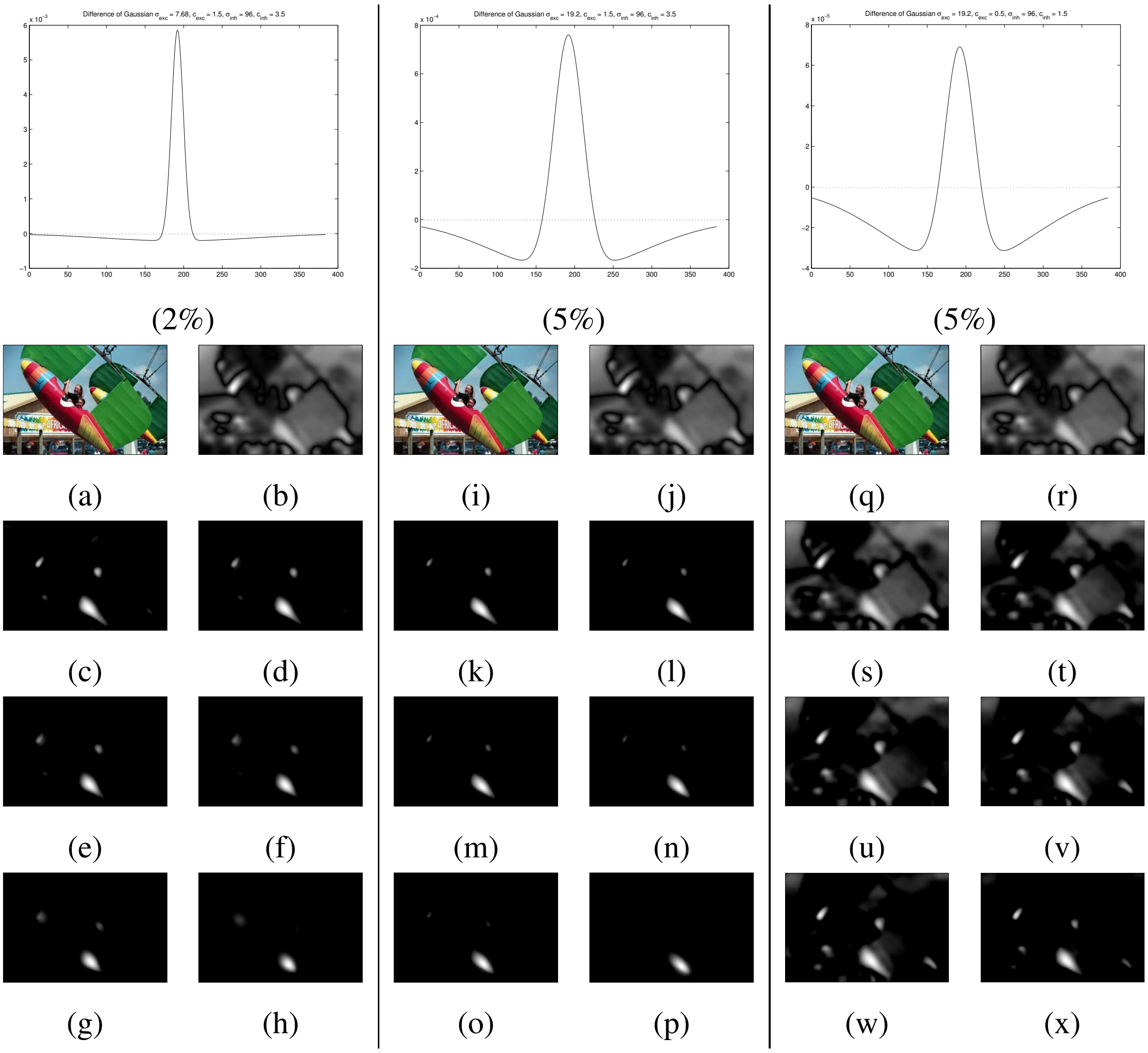
(a) (b) (i) (j) (q) (r)

(c) (d) (k) (l) (s) (t)

(e) (f) (m) (n) (u) (v)

(g) (h) (o) (p) (w) (x)

Figure 3.9: Difference of Gaussian convolution of red-green opponents using three different excitation sigma values. Images (a, i, and q) show the original image and (b, j, and r) shows the 3-7 centre-surround red-green opponent image with zero convolutions. Views (c–h) show the red-green opponent map with 1, 2, 3, 4, 5, and 10 convolutions respectively. The excitation function has a standard deviation of 2% image width. Images (k–p) show red-green opponents convolved with a DoG kernel using excitation 5% of image width. Images (s–x) show red-green opponents convolved with a weaker DoG kernel having excitation 5% of image width. Images originate from the visual attention software.



(2%) (5%) (5%)

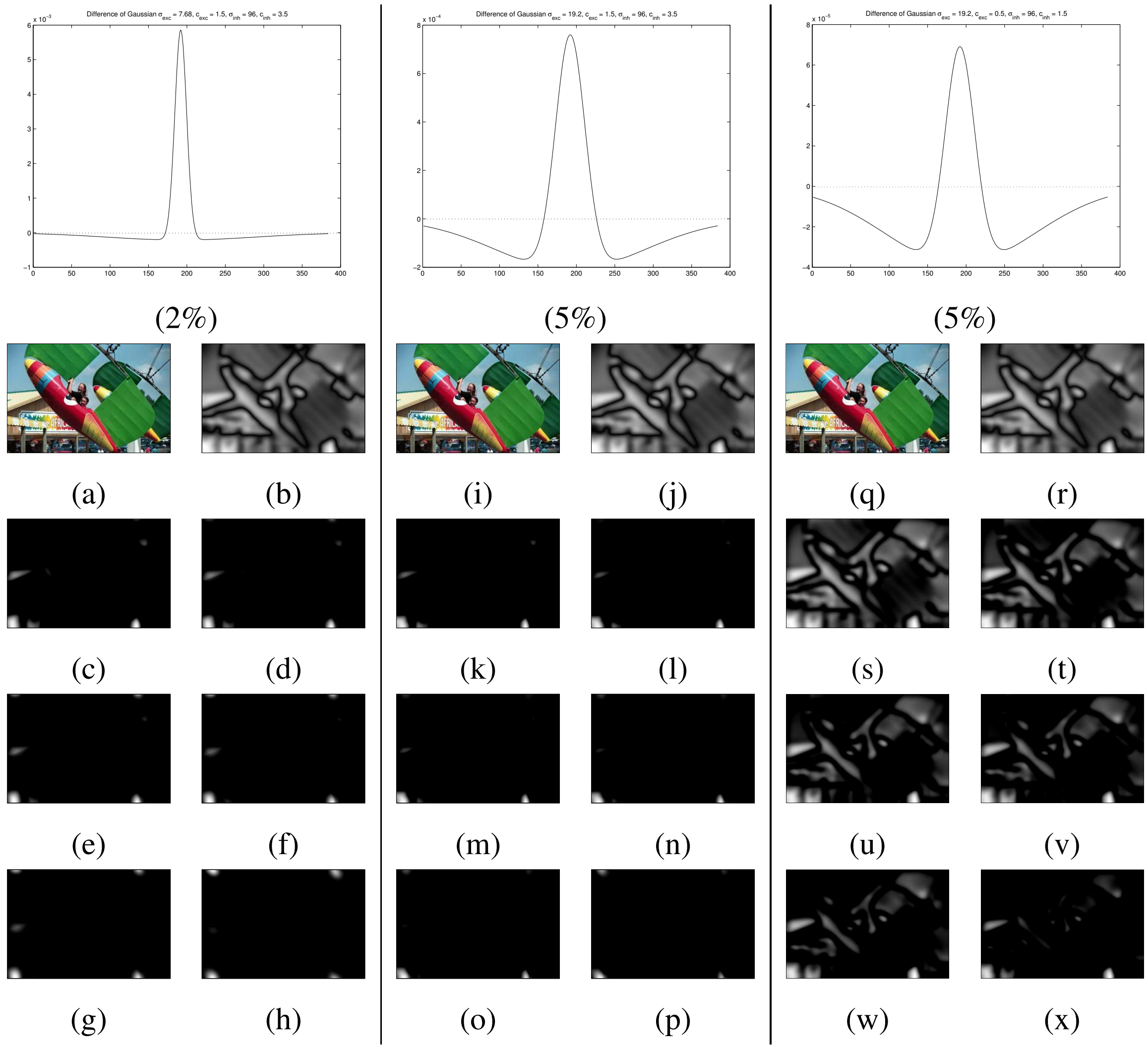
(a) (b) (i) (j) (q) (r)

(c) (d) (k) (l) (s) (t)

(e) (f) (m) (n) (u) (v)

(g) (h) (o) (p) (w) (x)

Figure 3.10: Difference of Gaussian convolution of blue-yellow opponents using three different excitation sigma values. Images (a, i, and q) show the original image and (b, j, and r) shows the 3-7 centre-surround blue-yellow opponent image with zero convolutions. Views (c–h) show the blue-yellow opponent map with 1, 2, 3, 4, 5, and 10 convolutions respectively. The excitation function has a standard deviation of 2% image width. Images (k–p) show blue-yellow opponents convolved with a DoG kernel using excitation 5% of image width. Images (s–x) show blue-yellow opponents convolved with a waker DoG kernel having excitation 5% of image width. Images originate from the visual attention software.



(2%) (5%) (5%)

(a) (b) (i) (j) (q) (r)

(c) (d) (k) (l) (s) (t)

(e) (f) (m) (n) (u) (v)

(g) (h) (o) (p) (w) (x)

Figure 3.11: Difference of Gaussian convolution of intensity opponents using three different excitation sigma values. Images (a, i, and q) show the original image and (b, j, and r) shows the 3-7 centre-surround intensity opponent image with zero convolutions. Views (c–h) show the intensity opponent map with 1, 2, 3, 4, 5, and 10 convolutions respectively. The excitation function has a standard deviation of 2% image width. Images (k–p) show intensity opponents convolved with a DoG kernel using excitation 5% of image width. Images (s–x) show intensity opponents convolved with a weaker DoG kernel having excitation 5% of image width. Images originate from the visual attention software.

merging the maps within feature. First, this further reinforces saliency for regions that exist across multiple spatial scales. Second, intra-feature maps have similar saliency across spatial scales so separately learning weights for each is redundant. Lastly, it enables dimensional reduction of the learning problem. Prior to combination the system has thirteen inputs leading to a single output; afterwards this is reduced to four inputs.

Conspicuity maps are convolved an additional five iterations and then multiplied by an inhibition map. The process of inhibition is discussed below. A single saliency map is then created as a weighted summation of features and iteratively convolved five times by the difference of Gaussian kernel to introduce dynamic inter-feature competition. A winner-takes-all network directs attention to the location of maximum saliency. To compute winner-takes-all we use a maximum operator to imitate a dynamical system.

3.5.1 Inhibition of Returns

When viewing a scene, attention dynamically shifts between points of interest rather than becoming fixated on a single location. In the computational model this is implemented through inhibition of returns. After the winner-takes-all network directs attention, the focal location is added to a global inhibition map as a circular region of value zero. Inhibiting saliency in the region surrounding focus causes a shift of attention to the next most salient stimuli. After each time step the prior inhibition values decay toward one. This computational model fully decays feature inhibition in five time steps.

3.6 Conclusion

Saliency maps provide a biologically plausible implementation of bottom-up attentional selection. In this model we include hue opponents, intensity opponents, and motion, as the preattentive features. A simplified computation of the derived yellow colour plane was introduced and the difference of Gaussian convolution kernel was studied extensively. The developed model is appropriate for either a static or dynamic environment. Other studies are often restricted to static scenes and instead of motion they use an edge orientation feature based on Gabor filters.

Chapter 4

Learning System

4.1 Introduction

Chapter three discussed an implementation of visual attention using saliency maps. This chapter extends the model by developing a system for learning the importance of each feature to attentional selection. Given a series of focal points for successive image frames, it is possible to learn an observer's internal model of relative feature weights. Inhibition of returns causes a shift of attention, but feature weights direct target selection by controlling the cumulative saliency of each stimuli. The learning problem determines weights that minimise saliency at unattended locations while maximising saliency at the intended focus of attention.

4.2 Tracking Attentional Focus

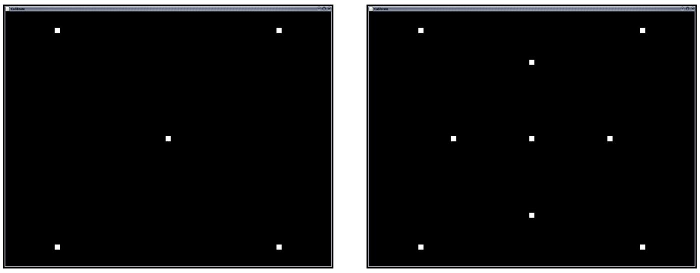
4.2.1 Eye Tracker

An ISCAN eye tracking system records gaze of a human subject viewing a projected scene. Head position remains fixed but the tracking hardware has a two degree of freedom servo control to compensate for small movements. Infrared light illuminates an eye and reflects off of the retina and cornea. A camera observes the reflections and sends a picture of the eye to a computer system where threshold algorithms determine the retinal and corneal centres. Because of the external infrared illumination, eye glasses reflect too much light and interfere with system operation. Only subjects with uncorrected vision or wearing contact lenses are suitable for experiments.

4.2.2 Calibration

Before tracking gaze, the system must first be calibrated with retinal and corneal reflections for known target locations. The ISCAN software system supports both a five point and nine point calibration. With a five point the subject directs focus to the centre and then corners of an image. The nine point calibration adds markers to the left, right, top, and bottom.

A computer desktop with black background and icons placed in the five point pattern was first used to calibrate the tracker. The subject focused on each icon in turn; however, the large icons caused confusion about where exactly to focus and resulted in poor calibration. Instead we developed an application to successively show white markers against a black background. After initially displaying all calibration points, they are turned on one at a time with the subject instructed to focus on the visible target. Figure 4.1 shows five and nine point configurations from the calibration application.



(a) (b)

Figure 4.1: Eye tracker calibration software with (a) a five point calibration and (b) a nine point configuration.

4.2.3 Determining Focus

Output from the eye tracking software is sent as a stream of coordinates over the serial port. The coordinates are defined within a reference frame of 512x512 pixels with its origin in the upper left corner. Learning feature weights depends only on fixation points and not on the temporal path between successive focal locations.

A first order dynamical system compares the change in position over time to detect shifts between movement and fixation states. Coordinates from the eye tracker have small fluctuations restricted to a local area when focusing on a single location. Moving gaze to a new area of the scene introduces large changes in velocity before settling again at the next point. It is not correct to simply rely on the boundary between large and small eye movements to signify a fixation shift. Inaccuracies in calibration and tracking can introduce occasional erroneous coordinate values that would otherwise signal a shift of attention. Instead it is better to analyse average velocity over time. A tracking failure returns a (0,0) coordinate so our system includes a failsafe mechanism to immediately discard any points at the origin without further processing.

The eight most recent coordinates returned by the eye tracker are maintained within a buffer. Two averages are computed on each addition of a new point. The first is the average euclidean distance between the buffered coordinates and the previous fixation. A large average distance tells that attention moved away from the previous focal location. For the case of a single erroneous value the average does not change a sufficient amount to register a focal shift. The second calculation is the average distance between successive points in the buffer. When gaze attends to a particular location, the distance between buffered points is small. If the first average is greater than a threshold value of 50 pixels and the second average is less than a threshold value of 15 pixels, gaze is presumed to be focusing on a new location. These thresholds were determined through tests with the experimental scenes. Attentional focus is assigned to the most recent coordinate and the system waits for the next shift of attention.

4.2.4 Adjusting for Calibration

It is difficult to calibrate the eye tracker system with an accuracy required to pinpoint gaze on a stimuli in the projected scene. Focus coordinates are often within close proximity of a known saliency but not on a target itself. To account for inconsistent calibration, a nearest neighbour algorithm determines on which target the fixation belongs. This is possible because experiments rely on contrived images with known target locations. Nearest neighbour assigns fixation to the closest target and saves its location for learning feature weights.

The eye tracker uses a 512x512 pixel reference frame while the saliency software assumes a 384x256 pixel image. Tracked coordinates are scaled to system coordinates (0.75 in the x-direction and 0.50 in the y-direction) prior to saving the location for learning.

4.3 Learning Feature Weights

From the saved attention coordinates, a least squares regression algorithm learns scalar weights for the preattentive features. Given one or more input variables, least squares computes a weight vector to minimise the sum squared error between actual and desired output (Vijayakumar, 2004). Since this problem has four conspicuity input maps, the regression learns weights w_0, w_1, w_2, w_3, and w_4, of equation 4.1 where the vector $\bar{x}$ represents the conspicuity maps. Without a priori information the bias w_0 is zero.

$$y = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 = \bar{w}^T \bar{x} \tag{4.1}$$

4.3.1 Deriving Least Squares

For a system output y and desired output $f(x)$, the least squares cost function is defined as follows.

$$\begin{aligned} E(w) &= \frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{f}(x_i))^2 && (4.2)\\ &= \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2 && (4.3)\\ &= \frac{1}{2}(y - Xw)^T(y - Xw) && (4.4) \end{aligned}$$

Minimising the cost function gives a general solution for the weight vector $\bar{w}$.

$$\frac{\partial E}{\partial w} = -(y - Xw)^T X = 0 \tag{4.5}$$

$$= -y^T X + (Xw)^T X \tag{4.6}$$

$$= -y^T X + w^T X^T X \tag{4.7}$$

$$y^T X = w^T X^T X \tag{4.8}$$

$$X^T y = X^T X w \tag{4.9}$$

$$\bar{w} = (X^T X)^{-1} X^T y \tag{4.10}$$

4.3.2 Input and Output Considerations

The variable X is a Nx4 matrix where each image pixel maps to a row of conspicuity map saliency values. Since the dynamic attentional system involves decaying inhibition of returns, it is necessary to account for this as part of the least squares problem. The conspicuity maps include saliency inhibition from previous time steps as discussed in the computational model.

Only fixation coordinates are considered when training the system; the actual saliency value and any other stimuli in the output map cannot be known from the eye tracker data. In addition, the temporal shift between foci of attention is disregarded. A Nx1 y matrix contains an ideal saliency map where a radius of twenty pixels surrounding the point of attention contains a normalised value of one and all other pixels have value of zero, corresponding to maximum and minimum saliency respectively. A circle of maximum saliency drawn around the fixation point accounts of attention falling anywhere within a stimuli.

4.3.3 Implementation

The solution for feature weights $\bar{w}$ includes the pseudo inverse $X^+ = (X^T X)^{-1} X^T$ to invert rectangular matrices (Vijayakumar, 2004). A single 384x256 pixel training image adds 98,304 rows of four columns to the X input matrix. Additional training images append their input and output pixel values to the existing X and y matrices. With only four training images the X matrix is of size 393,216x4 and the output matrix is of size 393,216x1. Calculating the input covariance matrix is computationally expensive and storage of the 32-bit floating point covariance matrix requires 4.72GB of memory. To solve the memory requirements we instead implemented a recursive least squares algorithm. For further savings, any rows having all zeros for input and output do not affect the feature weights and so are discarded.

4.3.4 Recursive Least Squares Algorithm

Recursive least squares is a method of computing feature weights without the pseudo inverse. With a recursive algorithm the weights are available at any time during the learning process and can later be updated with additional training data. The recursive solution is identical to that of batch least squares equation 4.10

Tabus (2004) provides a detailed explanation of the recursive least squares algorithm. λ is a forgetting factor. Here $\lambda = 1$ so there is no decay of past results.

$$P_{n+1} = \frac{1}{\lambda}(P_n - \frac{P_n x x^T P_n}{\lambda + x^T P_n x}) \tag{4.11}$$

$$w_{n+1} = w_n + P_{n+1} x (t - {w_n}^T x)^T \tag{4.12}$$

The algorithm can be broken into a series of computational steps. Given inputs x_1, x_2, x_3, ..., and output y_1, y_2, y_3, ...

1. Initialise $w_0 = 0$ and $P_0 = \delta I$ where $\delta << 0$

2. For each time instant n = 1 ... N

$$\begin{aligned}
\pi &= x_n^T P_{n-1} \\
\gamma &= \lambda + \pi x_n \\
k_n &= \frac{\pi^T}{\gamma} \\
\alpha_n &= y_n - w_{n-1}^T x_n \\
w_n &= w_{n-1} + k_n \alpha_n \\
P' &= k_n \pi \\
P_n &= \frac{1}{\lambda}(P_{n-1} - P')
\end{aligned}$$

4.4 Conclusion

Least squares regression uses a series of focal attention points and conspicuity maps to learn linear weights for each feature. The complete system involves tracking an observer's gaze in real time and subsequently compensating for any calibration inaccuracies with a nearest neighbour calculation. We selected the recursive least squares solution because of computational requirements but it additionally gives flexibility to update learnt weights if provided new training data.

Chapter 5

Implementation

5.1 Introduction

Chapters three and four discussed the computational model and learning system from a principled approach. This chapter focuses on the system hardware including the eye tracker and computer systems on which the software models runs. The software architecture is also discussed, including design decisions and performance metrics. Lastly the open source libraries used in the software are introduced.

5.2 Hardware

5.2.1 Eye Tracking System

To record gaze and visual attention, we purchased an ISCAN RK-464 eye tracking system. Included are a desk mounted tracking unit and a workstation computer running the ISCAN software. Figure 5.1 provides a schematic diagram of the system components. An external source, in this case either a Sony EVI D70 video camera or a second computer system playing MPEG files, supplies video data to the system. A video splitter directs the input to a capture card in the workstation and also to a LCD overhead projector displaying the experimental scenes. The desk mounted tracking unit, shown in figure 5.2, returns an eye image to the workstation where ISCAN software thresholds and tracks gaze based on a prior calibration phase. The full setup also includes two television monitors. As illustrated in figure 5.3, one shows an image of the eye for thresholding retinal and corneal reflection and the second monitor displays a copy of the projected scene with the tracked gaze overlaid. The tracking software outputs various data parameters for archival or processing. In this application we configure the tracking software to stream gaze coordinate points over the serial port at 4800 baud.

Using an overhead projector, images and video are displayed on a large screen. The projected image is 1.55 by 1.17 meters and the subject sits 2.1 meters from the screen. Available floor space limits size and distance to the subject.

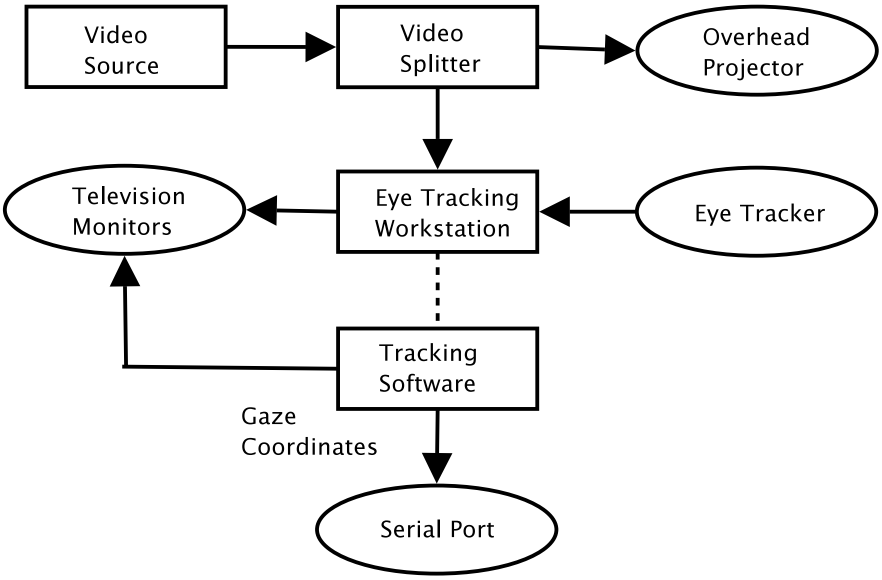


Figure 5.1: Schematic diagram of the eye tracking system.

5.2.2 Computer System

The visual attention software runs on a Dell Precision 360 workstation. On this we installed the Fedora Core 2 distribution of the Linux operating system. The computer has an Intel Pentium 4 3.2GHz Extreme Edition processor with hyper-threading enabled and 1GB of PC400 DDR RAM. A nVidia QuadroFX 500 graphics card with dual monitor support provides simultaneous output to both a conventional monitor and a video splitter for the eye tracking system.

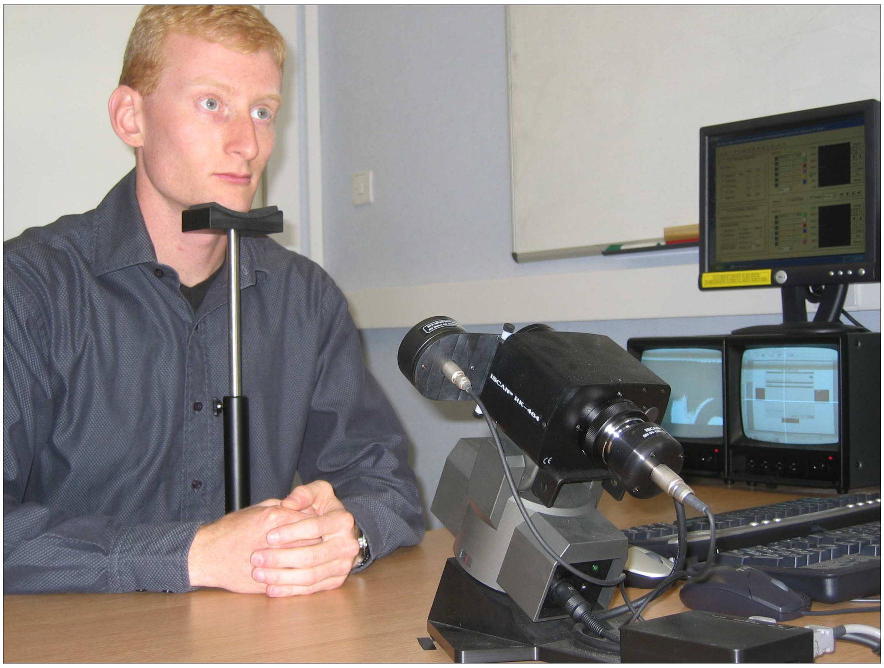
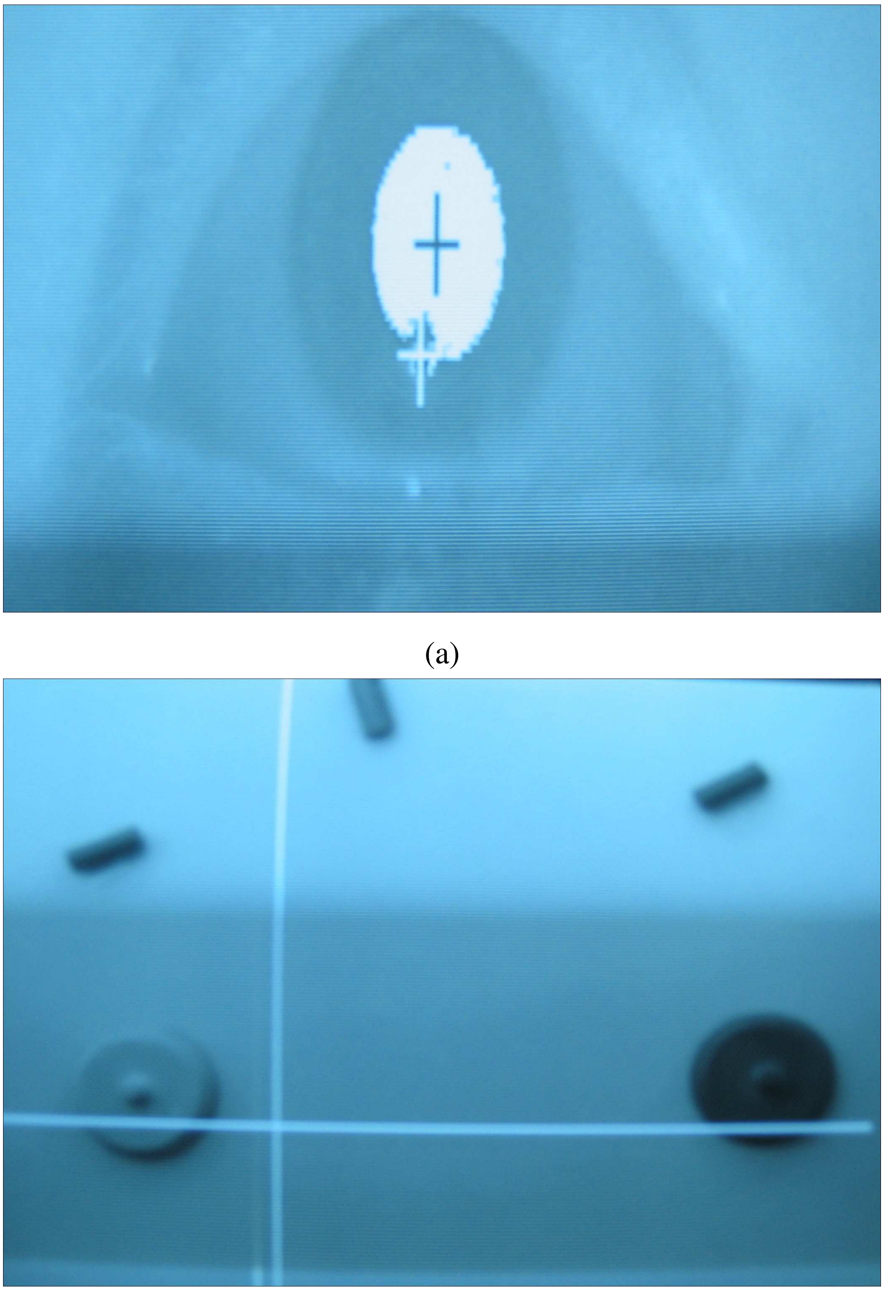


Figure 5.2: Eye tracker system.

A Pinnacle PCTV Rave framegrabber card captures real-time video from a Sony EVI D70 series colour video camera. Communication with the framegrabber requires the Video4Linux (V4L) API to interface with the bttv Linux kernel driver. We developed framegrabber interface software using the original V4L rather than the second release 4VL2 because the Fedora distribution does not yet support development libraries for the latest version.

5.3 Software

Software components are written in the C programming language and compiled with gcc 3.3.3. Libraries from several open source projects provide underlying functionality and were selected for completeness of implementation and easy of integration into the code base. Specifically, the Open Computer Vision library (OpenCV) simplifies image processing, FFTW is used for fast Fourier transforms, and FFmpeg provides the framework for MPEG video encoding and decoding.



(a)

(b)

Figure 5.3: (a) Thresholded image of the eye seen by the tracker with crosses on the retinal and corneal reflection. (b) Monitor display of the experimental scene with real time gaze tracked by the white cross.

Project source code is available in the /home/slmc/attention directory of the computer vertigo.dcs.ed.ac.uk.

5.3.1 Multithreaded Architecture

To provide optimal performance with the current computer system and include mechanisms for upgrading to a multiprocessor machine in the future, the saliency map software implements a partially threaded design while ensuring the internal structure is conducive to upgrading with a fully multithreaded architecture. We use the pthreads API because it is POSIX compatible and supported by many Unix and Unix-like operating systems. Pthreads provides an easier and lighter weight alternative to Unix processes and the 'fork' command; however, it is the operating system's role to schedule thread execution and distribute load across the microprocessors. Nichols et al. (1996) fully discusses pthreads and several of the traditional thread development models.

The visual attention application contains two threads. The first is a looping background process extracting frames from the framegrabber or MPEG file. Video4Linux imposes minimal system overhead by enabling the kernel driver to directly access framegrabber memory and registers. This thread captures frames at a consistent rate of 25 frames per second.

A second thread computes saliency maps using the model outlined in chapter three. To simplify implementation and optimise the project for the uniprocessor computer, saliency computation is performed in serial without dividing the work among additional threads. Computationally intensive programs run on single processor can experience a decrease in performance when dividing a task into multiple processes. The reason is that switching execution between threads imposes additional overhead due to stack buffering and interprocess communication. There are also design complexities to avoid long synchronisation waits, deadlocks, and race conditions. For multiprocessor computers, threads are required to run the application on several processors simultaneously. In preparation for future updates, the software application is divided into logical components that can later be wrapped in threads.

5.3.2 Performance Optimisation

5.3.2.1 Parameter and Size Values

At first the software used input images of 768x512 pixels with nine Gaussian pyramid levels and ten convolution iterations per feature map. The centre-surround feature maps used centre scale $c \in \{2,3,4\}$ and surround $s = c + \delta$ where $\delta \in \{3,4\}$.

For these parameters, the total number of feature maps for red-green and blue-yellow opponents is twelve, where ten convolutions of each means a total of 120 hue convolutions per time step. Table 5.1 outlines processing statistics for the feature maps. With execution time of 11.007 seconds for convolving the hue feature maps, fast Fourier convolution requires approximately 0.092 seconds for each iteration.

Operation	Hue (seconds)	Intensity (seconds)
Gaussian Pyramids	0.346	0.125
Convolution	11.007	5.553
Total Time	11.675	5.811

Table 5.1: Computation time for hue and intensity feature maps of 768x512 pixel images.

Convolution accounts for 94.3% and 95.6% of run time for hue and intensity features respectively. We tested a reduced image size to determine the benefit to overall computation time. Halving the width and height reduces image pixel count by a factor of four. The smaller size of 384x256 pixels supports up to eight Gaussian pyramid levels instead of nine, so we changed the centre scale to $c \in \{2,3\}$.

Table 5.2 lists updated computation times for an image size of 384x256 pixels, eight Gaussian pyramid levels, and ten convolution iterations per feature map. Hue and intensity computations completed 84.2% and 80.8% faster respectively. Additionally, each convolution iteration completed in 0.0207 seconds, a 77.5% improvement.

As mentioned previously, the selected difference of Gaussian kernel enables the the convolution iterations to be reduced from ten to five. This greatly improved system performance since convolving feature maps accounts for the largest percentage

Operation	Hue (seconds)	Intensity (seconds)
Gaussian Pyramids	0.083	0.069
Convolution	1.653	0.845
Total Time	1.840	1.118

Table 5.2: Computation time for hue and intensity feature maps of 384x256 pixel images.

of computation on each frame. The final system processes incoming video data and selects the next point of focal attention at a rate of approximately once per second.

5.3.2.2 Software Optimisation

To achieve fast computation of saliency maps we made several optimisations to improve overall system performance. All memory is preallocated at the start of the application and is only deleted prior to exit. This includes working and temporary images, data structures, and buffers. Because the application allocates over three hundred megabytes of memory during execution, moving this step from an inline operation to a one-time event increased the processing rate by 57%, from 0.35 frames per second (fps) to 0.55 fps.

The software is naturally divided into two halves by the FFT convolution of feature maps. Prior to convolution all working images are stored in OpenCV *IplImage* data structures and are manipulated with functions from the OpenCV API. The result of convolution is data of type *double* *. Instead of converting the data back to OpenCV format we use direct pointer arithmetic and a custom matrix library for all post-convolution processing. The matrix library is faster than the equivalent OpenCV functions and resulted in a further performance increase from 0.55 fps to 0.59 fps (7.2%).

A rewrite of the framegrabber interface code to use memory registers instead of accessing data through file handles improved image capture rate from 8 fps to 25 fps. An additional benefit is a reduction of computational overhead. Between the rewrite of the framegrabber access code and replacing additional OpenCV operations with direct pointer arithmetic, the final frame rate for the application is approximately one frame per second.

5.3.3 OpenCV

Basic image processing uses the Open Computer Vision (OpenCV) library created by Intel and now maintained by an open source community. OpenCV is a cross platform toolkit supporting the Linux, Macintosh, and Windows operating systems. The latest packaged release is beta3.1 dated 5 March 2003. The reason for selecting the older release rather than a recent CVS snapshot is that beta3.1 is considered stable with support readily available through a user forum.

OpenCV processes image data with matrix operations and also includes features for working with colour planes, Gaussian pyramids, motion analysis, and displaying images independent of platform. Other useful features such as capturing images from a camera and working with AVI files are only supported when programming for Microsoft Windows.

Intel distributes a separate library entitled Intel Performance Primitives (IPP) to improve performance of certain computationally expensive functions and add support for advanced operations such as fast Fourier transforms. While OpenCV is a free open source project, IPP is closed source and requires a license. We do not use the IPP library and it is unknown what performance improvement could be obtained.

http://www.intel.com/research/mrl/research/opencv/

http://www.intel.com/support/performancetools/libraries/ipp/ia/opencv.htm

http://sourceforge.net/projects/opencvlibrary/

5.3.4 FFTW

FFTW is a discrete Fourier transform library whose name is an acronym for *Fastest Fourier Transform in the West*. The library is written entirely in C and runs on Unix based operating systems. As a free software project it was developed at MIT over a 35 year period and is now maintained by Matteo Frigo and Steven Johnson. In terms of implementation, FFTW computes discrete Fourier transforms of both real and complex data in one or multiple dimensions.

We selected the library because of impressive benchmark results compared to other popular open and closed source FFT libraries. FFTW is optimised for current processors and and includes support for SSE, SSE2, 3dNow!, and Altivec. In published benchmark tests FFTW closely trails the Intel IPP library in terms of raw speed. The API is not overly complex and documentation includes helpful examples.

http://www.fftw.org/

5.3.5 FFmpeg

The open source FFmpeg library encodes and decodes video streams. It supports a variety of codecs and is used by many successful video and image processing applications. Though originally a project for Linux, it has since been ported to many other platforms.

http://ffmpeg.sourceforge.net/

5.4 Conclusion

The ISCAN eye tracker is a capable system but in practise we found it very difficult to calibrate. When viewing the upper half of the projected image a large eye angle often resulted in the loss of corneal reflection. With stationary eye trackers, researchers often display stimuli on a computer monitor to avoid the problems associated with large eye angles. OpenCV simplifies development of image processing applications but for optimal performance we preallocated memory and replaced post-convolution OpenCV functions with pointer arithmetic and a custom matrix library.

Chapter 6

Experiments

6.1 Introduction

Thus far discussion of the dynamic visual attention system focused on theory and developing a model of bottom-up saliency maps. This chapter presents experiments involving static and dynamic scenes to verify correctness of the implemented attention model and investigate the least squares method of learning feature weights from fixation points. Previous studies of saliency maps often focused on detecting targets in natural scenes (Itti and Koch, 2000; Niebur et al., 2001; Funk, 2004) or cluttered environments (Itti et al., 2001). These experiments instead investigate how feature weights define fixation patterns in simple scenes.

Throughout this chapter we refer to ‘unbiased saliency maps’. This term implies a saliency map composed from features of weight one. In an unbiased saliency map all features contribute equally to winner-takes-all attentional selection.

6.2 Design of Experimental Scenes

6.2.1 Static Scene

Preattentive features in the model are hue, intensity, and motion. Hue is further divided into red-green and blue-yellow colour opponent maps. The first three experiments involve a static image without motion. Visible in figure 6.1, there are four distinct and

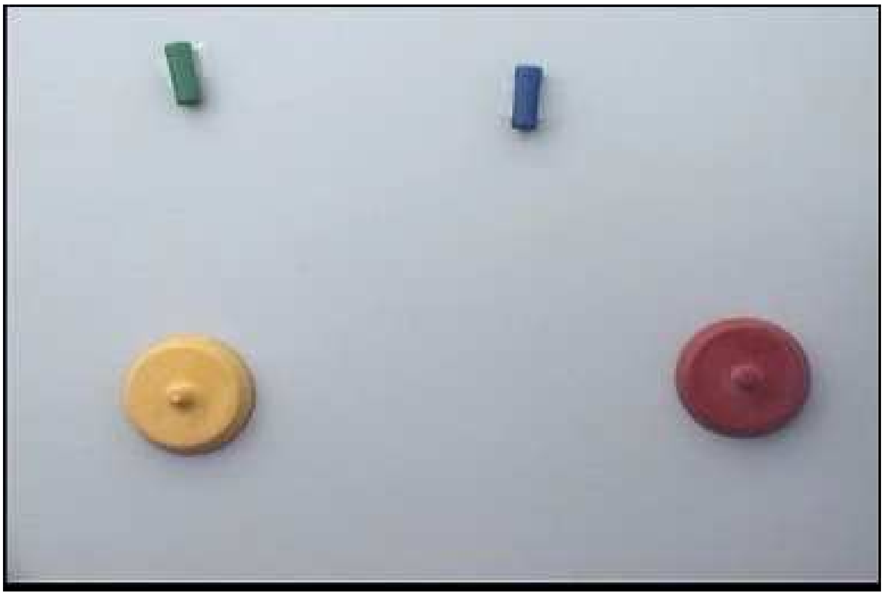


Figure 6.1: Static scene used in the visual attention experiments.

well separated targets on a lightly coloured background: a rectangular blue object in the upper right, a red circular object in the lower right corner, a yellow circular target in the lower left, and a green rectangular shape in the upper left corner. Targets reflect the four colour channels while their separation ensures accurate compensation of tracking data with the nearest neighbour technique as outlined in chapter four.

6.2.2 Dynamic Scene

Experiment four employs a dynamic scene with simple motion. Visible in figure 6.2, the dynamic environment is similar to the static image so that it retains many of the same properties. Widely separated elements are again selected to represent the four hues. As before, the upper right corner contains the blue target, a red object is in the lower right, yellow resides in the lower left, and green in the upper left corner. These four objects remains static. In the top middle of the scene is a moving red rectangle swinging from side to side throughout the experiment.

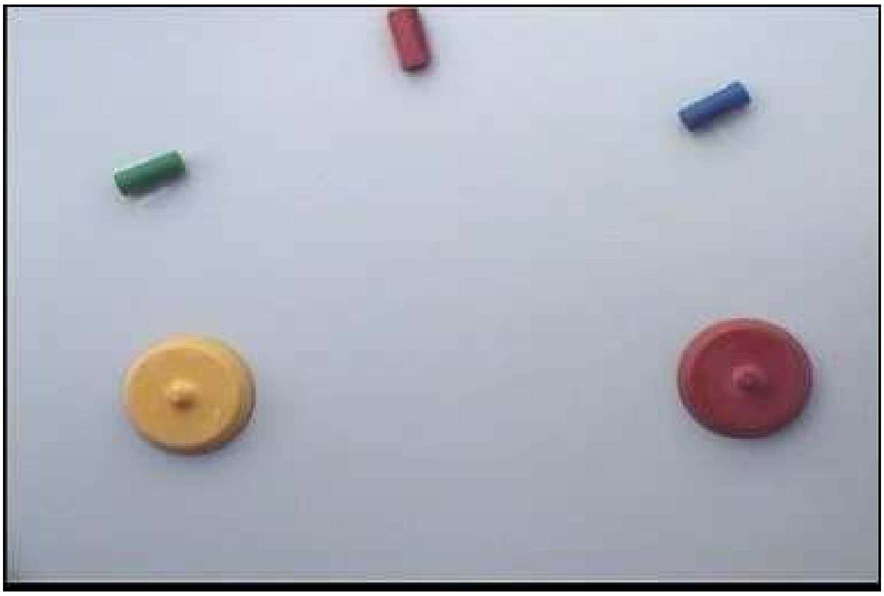


Figure 6.2: Dynamic scene used in the visual attention experiments.

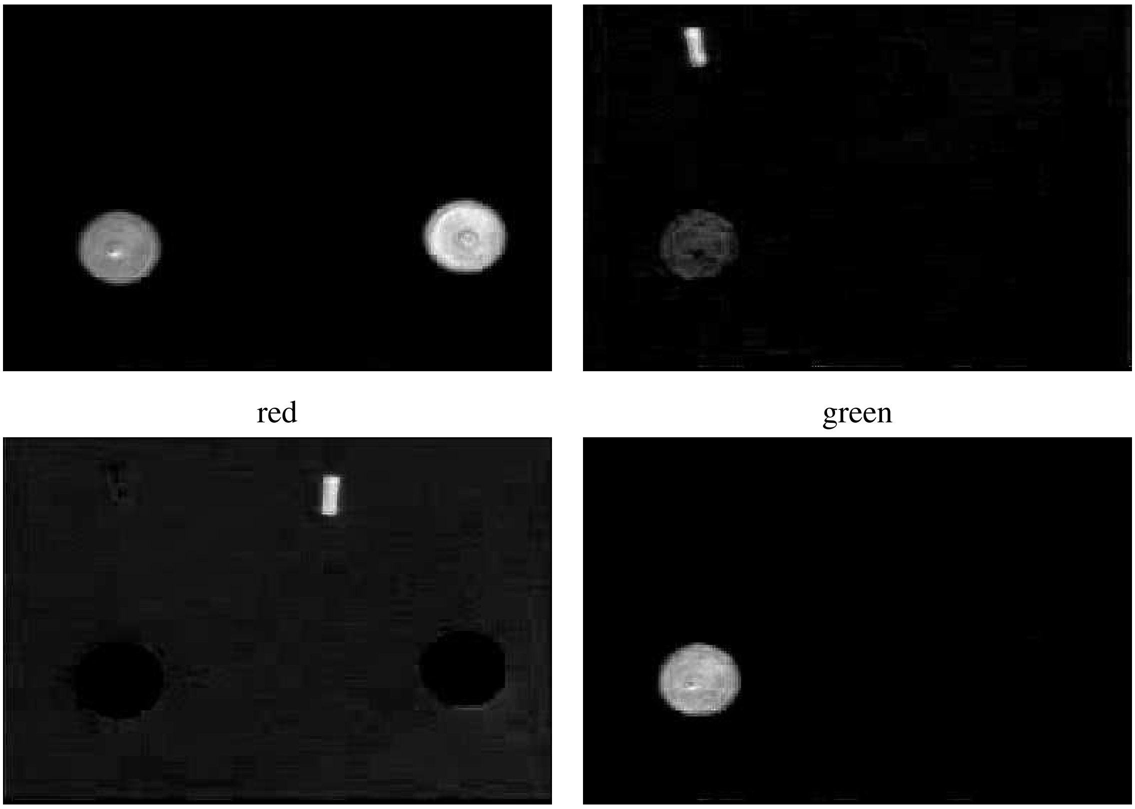
6.3 Experiment 1: Learning Unbiased Weights in a Static Environment

6.3.1 Defining the Experiment

The intent of the first experiment is to demonstrate functionality of the computational saliency map model in a static environment and then show that underlying feature weights can be recovered from a limited set of training data. Intermediate computations are displayed to confirm the software performs as expected at each step. Where discussed in terms of saliency, greyscale images represent a gradient extending from zero saliency (black) to maximum saliency (white).

The saliency map is constructed from red-green opponent, blue-yellow opponent, and intensity opponent conspicuity maps, each having a feature weight of one. This is referred to as an unbiased saliency map because all features contribute equally to attentional selection. Motion is not a factor for static scenes.

Using the first eight unbiased attention points as training data, the least squares learning method recovers a set of equivalent feature weights. Attentional selection



red

green

blue

yellow

Figure 6.3: Four broadly tuned colour channels from the static image.

for the unbiased and learnt saliency maps is compared to determine if the weights are representative of the original system.

6.3.2 Colour Channels

Colour channels and intensity are computed from the source image in figure 6.1. Shown in figure 6.3, the red, green, and blue targets are each strongly visible in their respective colour channels. The yellow target is represented among the yellow, red, and green colours because yellow is derived from the RGB colour model. This is further explained in chapter three and by equation 3.12.

6.3.3 Conspicuity Maps

Red-green opponent, blue-yellow opponent, and intensity opponent feature maps are created across multiple spatial scales and iteratively convolved. Figure 6.4 shows the resulting conspicuity maps. In the red-green opponent map, the red object is highly salient while the green is only slightly salient after dynamic competition through iterative difference of Gaussian convolution. Though the yellow object is visible in both red and green channels, computation of red-green centre-surround eliminates yellow areas. This is due to the colour opponent equation 3.14.

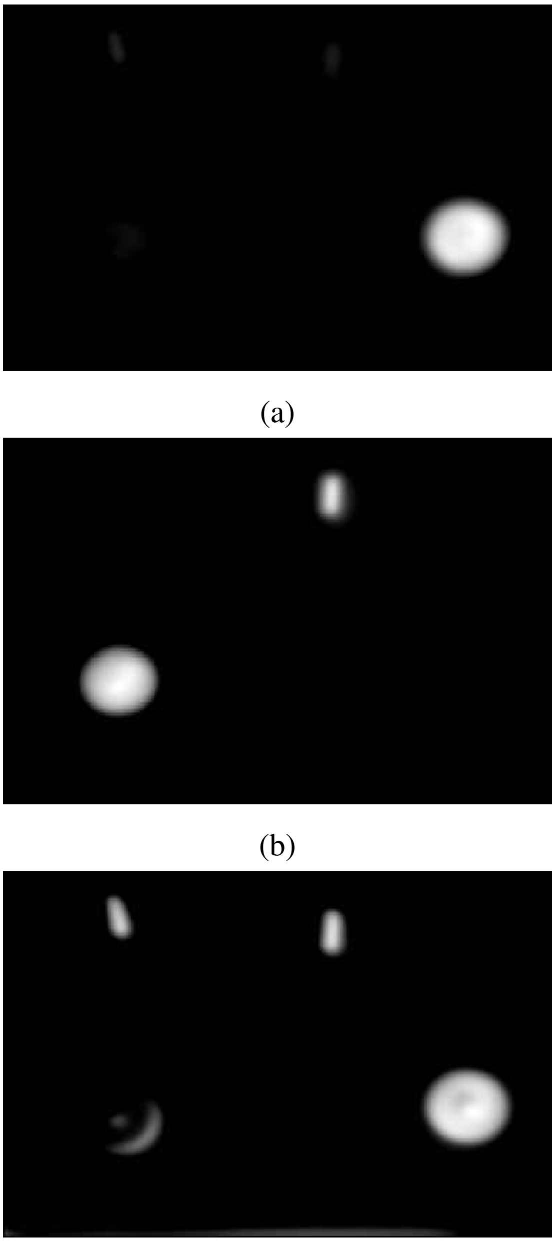
While intra-feature competition suppresses the green object, the blue-yellow conspicuity map contains strongly salient blue and yellow features. This raises a question of why the small blue rectangle remains salient in the colour opponents yet the green target is suppressed by dynamic competition. Examining the colour maps of figure 6.3, because the red target is much larger than green and equally salient, green does not survive saliency competition. In the blue and yellow colour channels, the large yellow feature is less salient than the smaller blue region so the blue-yellow conspicuity map retains both targets.

Contrast between the background and the red, green, and blue objects causes those regions to be strongly salient in the intensity feature map. Figure 6.4 shows a slight response at the location of yellow but this is attributed to shadows on the original image and has essentially no effect on the final saliency map. Competition suppresses yellow because it does not pop out against the lightly coloured background.

6.3.4 Attentional Selection

For unbiased saliency, the three conspicuity maps are linearly summed without individual scaling. Figure 6.5 shows the resulting saliency map. The first ten points of attention and corresponding inhibitions are computed and shown in figure 6.6. Each saliency map includes the effects of prior inhibition and indicates the focus of attention with a white circle.

Initially the winner-takes-all network directs attentional focus to the large red object in the lower right corner of the scene. Next most salient is the blue target since



(a)

(b)

(c)

Figure 6.4: Conspicuity maps for the (a) red-green colour opponents, (b) blue-yellow colour opponents, and (c) intensity opponents.

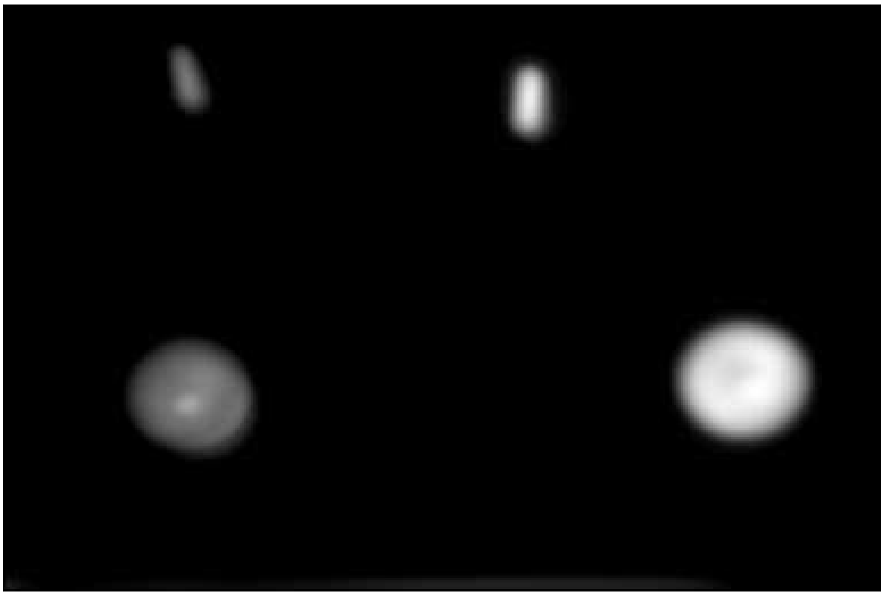


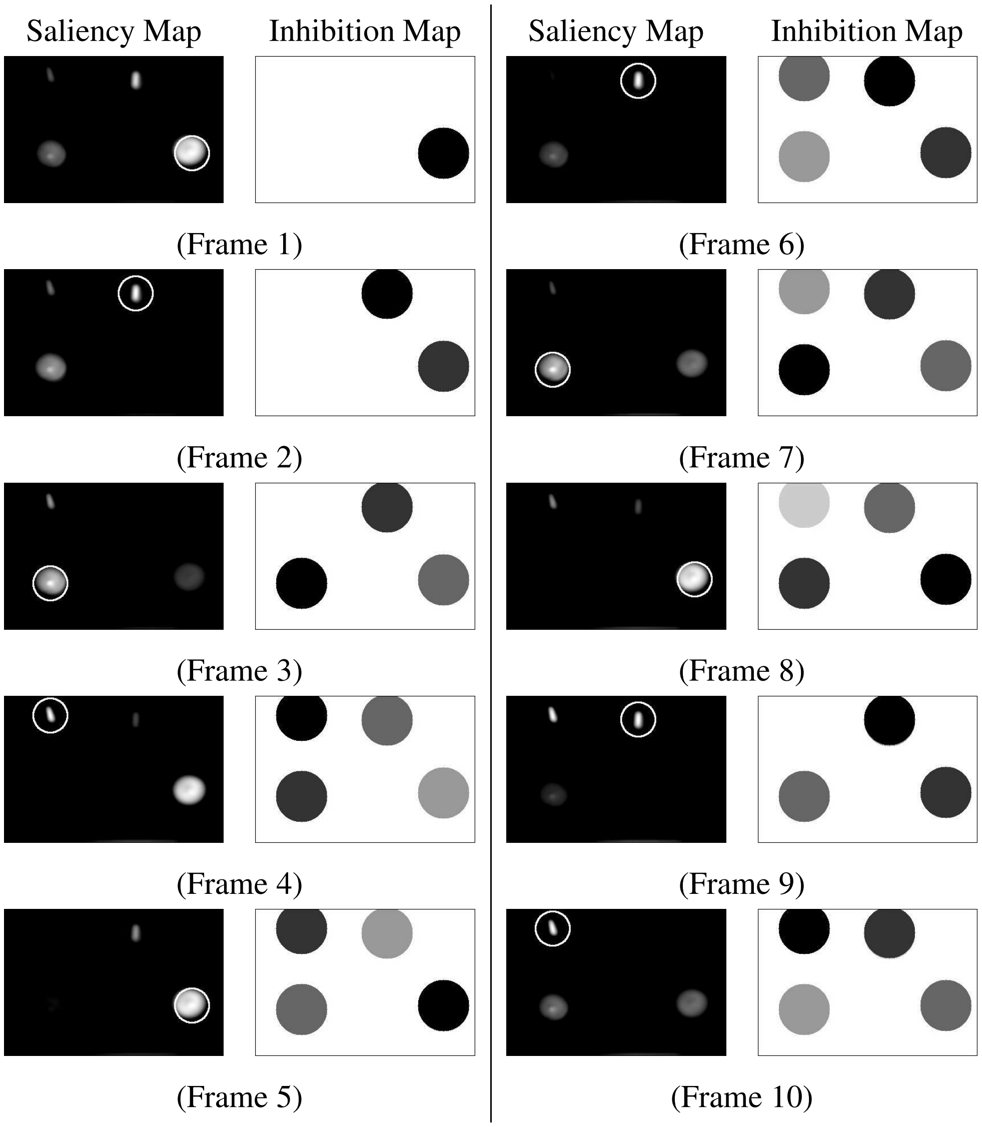
Figure 6.5: The unbiased saliency map when all feature weights are linearly summed without scaling.

it is present among both the blue-yellow and intensity opponents. In the third and fourth time steps focus scans to the yellow and green objects respectively. The system maintains this focal pattern until frame seven.

In frames eight through ten, attention moves from yellow back to red and blue prior to focusing on the green target. In the original pattern of frames three and four, focus shifts to green immediately following attention on yellow. Feature weights create an internal hierarchy of priorities to direct focus, but the visual system is not entirely predictive because inter-feature competition within the saliency map and decaying inhibition of returns contribute to system dynamics. Internal feature weights exist as relative principles and not to define an exact order of element selection.

6.3.5 Results From Learning

Table 6.1 lists equivalent feature weights learnt from the first eight unbiased focal points. For an unbiased map it is expected that the equivalent weights also have similar values. The weights have a mean of $\mu = 0.3549$ and variance $\sigma^2 = 0.000453$.



(Frame 1)

(Frame 2)

(Frame 3)

(Frame 4)

(Frame 5)

(Frame 6)

(Frame 7)

(Frame 8)

(Frame 9)

(Frame 10)

Figure 6.6: Experiment 1: Dynamic attentional selection and inhibition in the unbiased saliency map. A saliency map and the resulting global inhibition is shown for the first ten time steps. A white circle defines focus of attention in the saliency maps.

Feature	Unbiased Weight	Learnt Weight
Red-green colour opponents	1.0	0.335054
Blue-yellow colour opponents	1.0	0.352247
Intensity colour opponents	1.0	0.377396

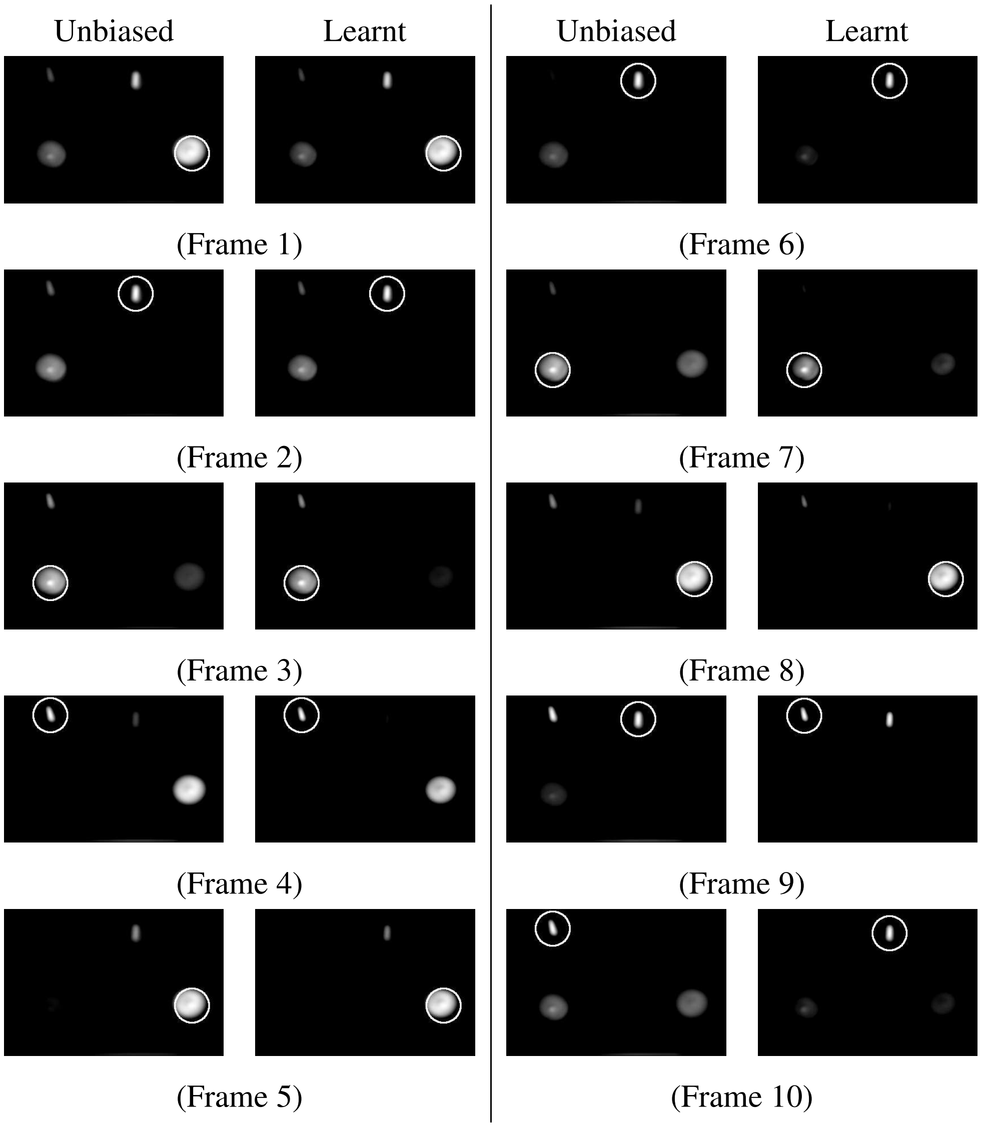
Table 6.1: Experiment 1: Comparison of unbiased and learnt feature weights. The equivalent weights have mean $\mu = 0.3549$ and variance $\sigma^2 = 0.000453$.

For a perfectly recovered weighting model, attentional selection at every time step will be identical to that of the unbiased response. Saliency maps built with the equivalent weights may exhibit small differences from the unbiased maps but in the ideal case this does not affect winner-takes-all selection. Learning systems perform best when given a large amount of training data but we wish to examine how the system performs with a limited set of coordinates.

An increasingly larger feature weight for blue-yellow and intensity opponents shows a correlation between weight and the number of salient regions present in a conspicuity map. For an unbiased map, the features having more objects exhibit a greater influence directing attention. This is because there is often overlap between salient regions of conspicuity maps, so features representing fewer targets are best suited for finer saliency adjustments.

To further verify operation of the learning model, figures 6.7 and 6.8 compare focus in the first twenty saliency maps for unbiased attention and learnt feature weights. Learnt focus exactly matches the unbiased case for sixteen of the twenty frames. At time step nine and ten the two systems select the green and blue objects in a different order. The systems again select identical attentional focus until frames nineteen and twenty. Here the selections of yellow and green differ in priority.

Results demonstrate how with limited training data the system learnt weights that closely mimic attentional selection of the unbiased case. The few differences in selective attention confirm that the weight model gives an approximate but not exact solution to the dynamical attention system.



(Frame 1)

(Frame 6)

(Frame 2)

(Frame 7)

(Frame 3)

(Frame 8)

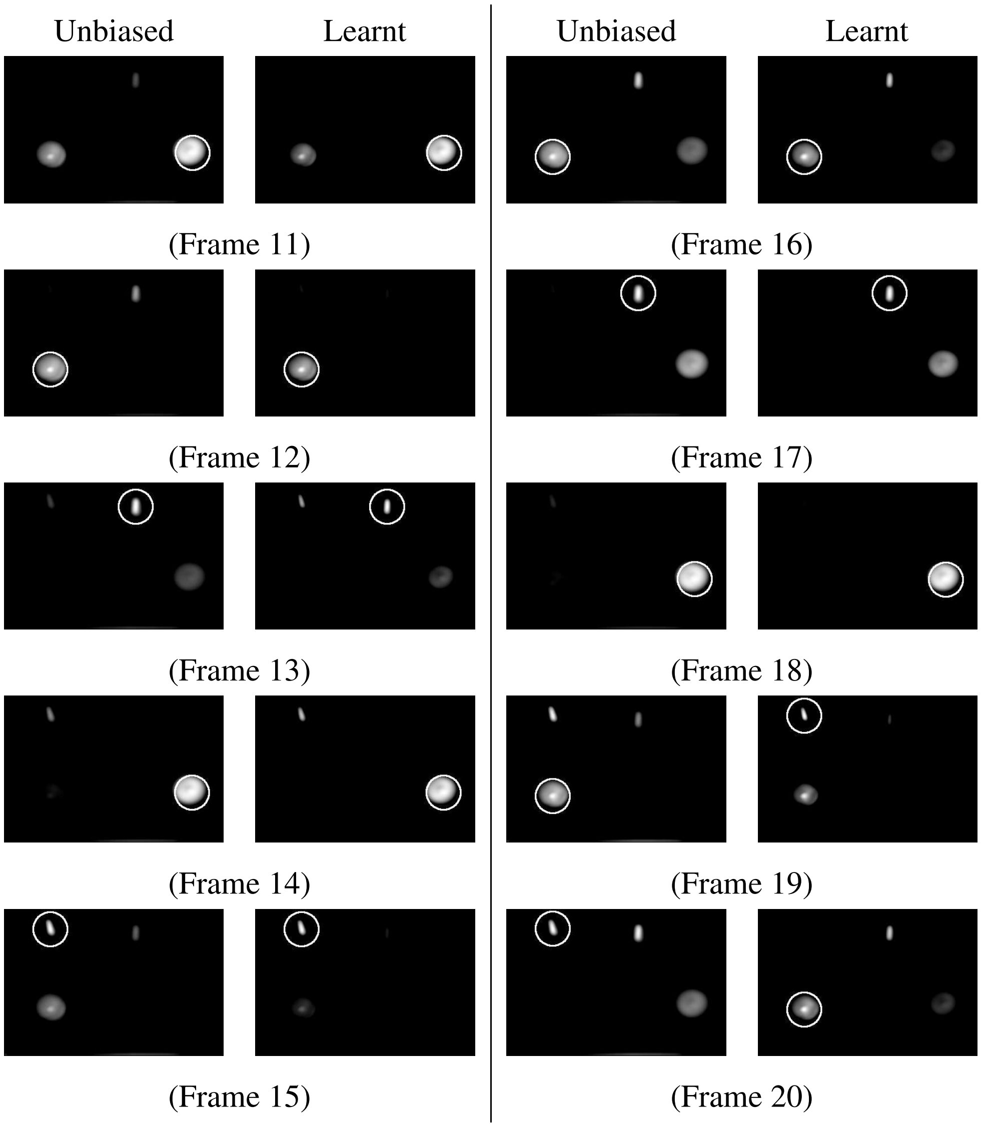
(Frame 4)

(Frame 9)

(Frame 5)

(Frame 10)

Figure 6.7: Experiment 1: Comparison of unbiased focal points and learnt equivalent attentional selection for frames 1–10.



Unbiased Learnt Unbiased Learnt

(Frame 11)

(Frame 16)

(Frame 12)

(Frame 17)

(Frame 13)

(Frame 18)

(Frame 14)

(Frame 19)

(Frame 15)

(Frame 20)

Figure 6.8: Experiment 1: Comparison of unbiased focal points and learnt equivalent attentional selection for frames 11–20.

6.4 Experiment 2: Recovering Known Feature Weights From Focal Attention

6.4.1 Defining the Experiment

The second experiment uses saliency maps generated from known feature weights to recover the original weight values. For a perfectly learnt system the attentional selection with recovered weights exactly matches the training focal point at each time step. Because the training data is based on defined feature properties, it is guaranteed that at least one solution exists to solve the problem.

We first artificially define feature weights for the hue and intensity opponents with the intent of giving a different attentional response than the unbiased case. The selected training weights are shown in table 6.2. Saliency maps and attention points are computed for the first twenty-one time steps. The coordinates are then used to relearn the original feature weights. From the learnt weights, saliency maps and attentional focus are computed for comparison to the original focus points.

Feature	Training Weight
Red-green colour opponents	0.15
Blue-yellow colour opponents	0.30
Intensity colour opponents	0.40
Motion	0.0

Table 6.2: Experiment 2: Feature weights used to train the feature weights.

6.4.2 Results of Learning

The first twenty-one attention points are extracted from saliency maps to learn feature weights. Unlike the first experiment which used a limited set of training data, all twenty-one points contribute to learning. Table 6.3 shows the learnt weights and their original values.

Feature	Training Weight	Learnt Weight	Percent Change
Red-green colour opponents	0.15	0.144596	-3.6%
Blue-yellow colour opponents	0.30	0.559240	+86.4%
Intensity colour opponents	0.40	0.473655	+18.4%
Motion	0.00	0.00	–

Table 6.3: Experiment 2: Comparison of training and learnt feature weights.

The recovered weight of red-green colour opponents is 3.6% smaller than the original, while an increased significance is attributed to both blue-yellow and intensity opponents. The weight of blue-yellow opponents is 86.4% higher and intensity has a 18.4% gain over its specified weight.

Saliency maps and attention points are generated to determine how representative the learnt weights are of the original training values. Figures 6.9 and 6.10 compare attention in the first fourteen frames for the training and learnt weights. Unbiased saliency maps are included for comparison. A white circle on the saliency map designates the location of visual attention.

By comparing the attention points for the training and learnt saliency maps we determine how accurately the weights reflect training data. In all fourteen points the selection of blue and green targets is synchronised between the two systems; however, the focus on attention is different for yellow and red regions in the scene. After every selection of the blue target, attention directs to red in the training points but yellow in the learnt system. This is visible in frames two, six, ten, and thirteen. Next, attention shifts to the yellow and red respectively, followed by mutual attention on the green and blue targets to restart the pattern.

Figure 6.11 shows larger pictures of the training and learnt saliency maps for the second frame. In both maps the red object (bottom right) appears more salient; however, the learnt weights result in the winner-takes-all mechanism selecting the yellow target (bottom left) for attention. Though the yellow target has a lower average saliency, a small bump in the centre has a higher saliency than the rest of the region. If referring back to the original scene in figure 6.1, this bump corresponds to a protruding feature on the yellow object. Learning a strong weight for the blue-yellow opponents raises the average yellow saliency toward the ideal response, but a result is that the centre patch of pixels becomes more salient than the red object and incorrectly attracts the focus of attention. While the problem has been identified there was insufficient time to fully investigate solutions to the issue and we instead leave it to future research.

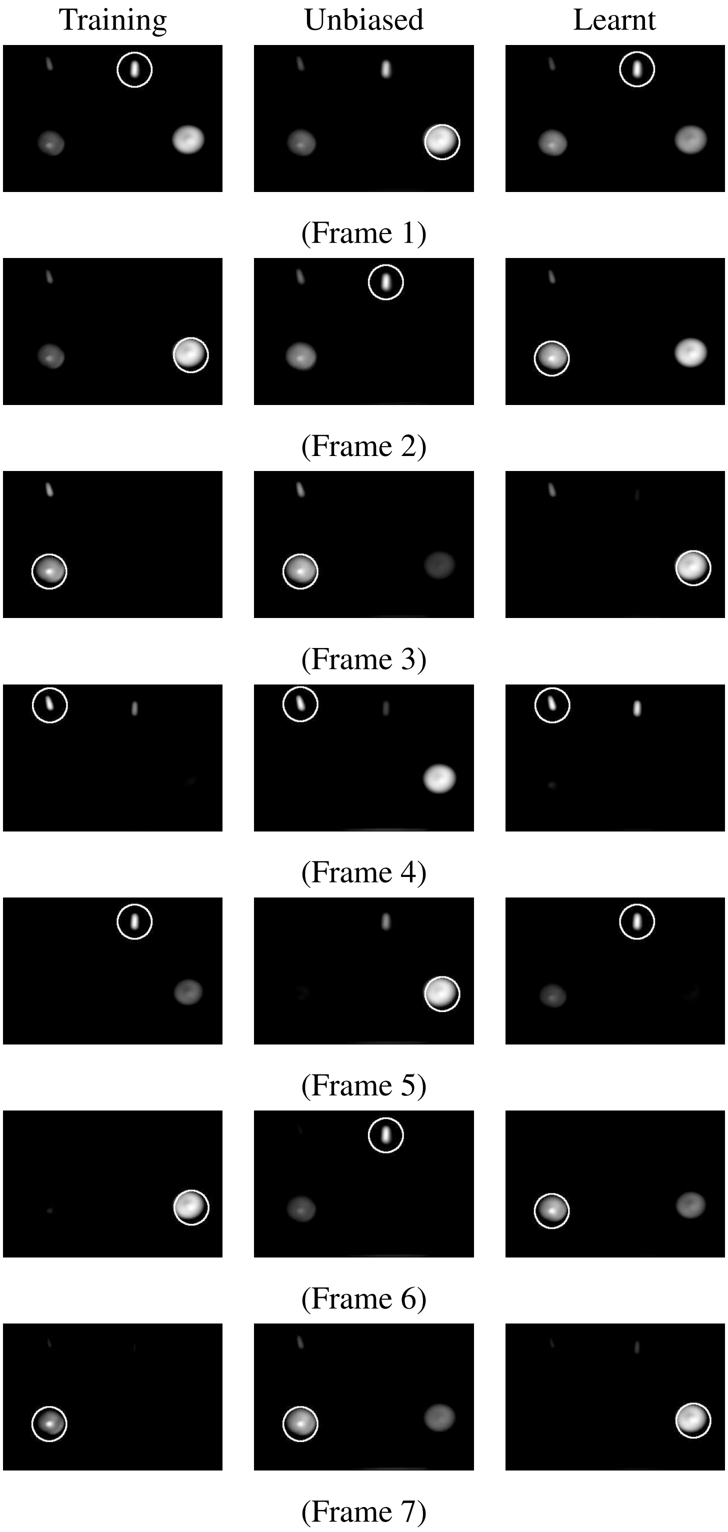
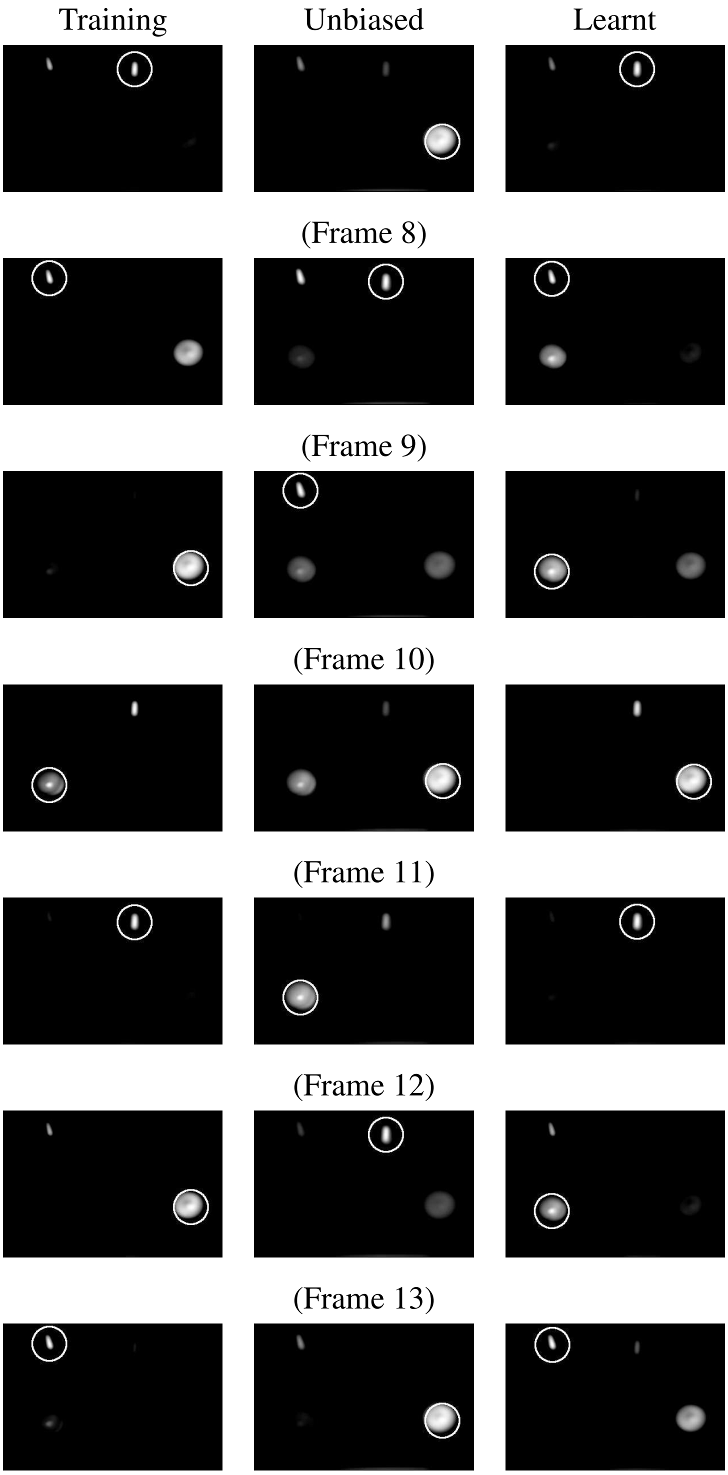


Figure 6.9: Experiment 2: Comparison of training, unbiased, and learnt attentional selection for frames 1–7.



Training Unbiased Learnt

(Frame 8)

(Frame 9)

(Frame 10)

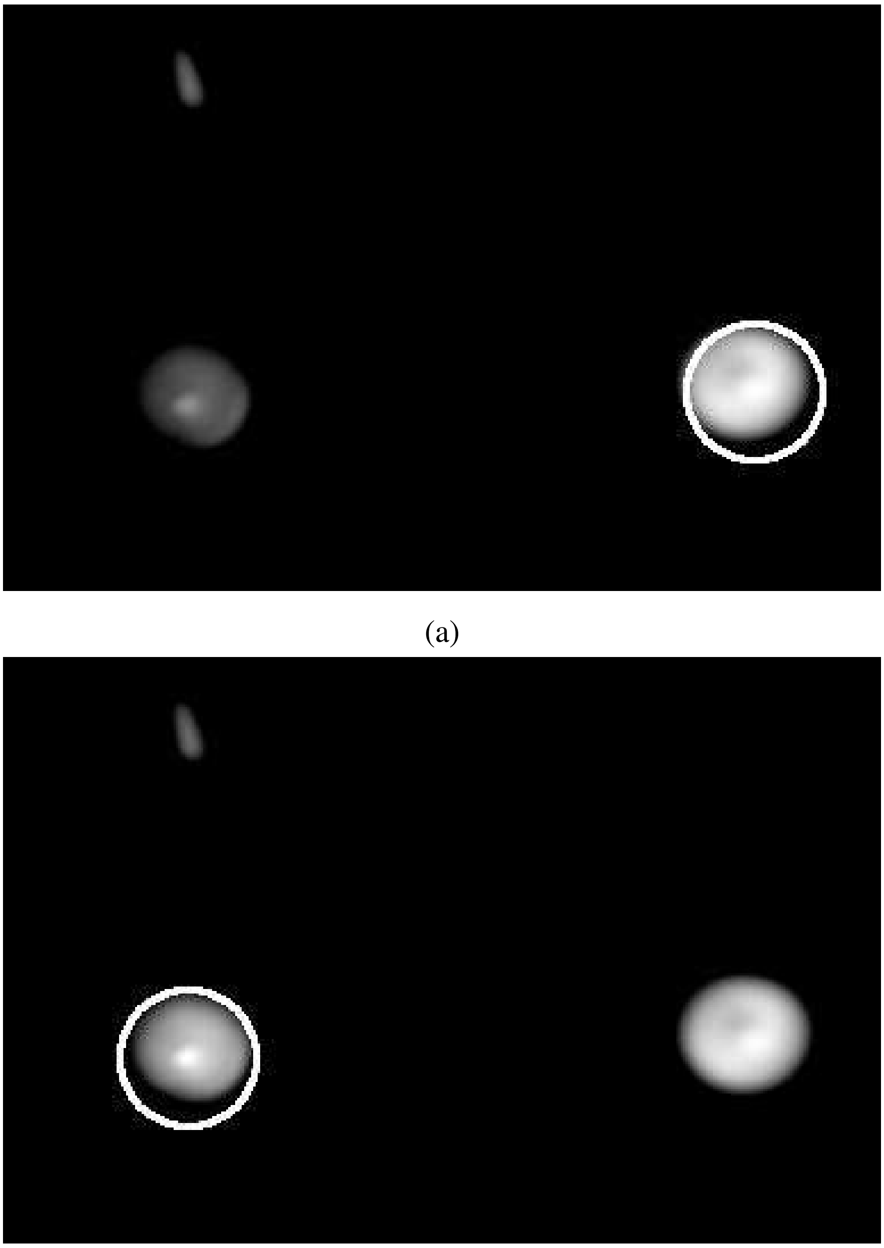
(Frame 11)

(Frame 12)

(Frame 13)

(Frame 14)

Figure 6.10: Experiment 2: Comparison of training, unbiased, and learnt attentional selection for frames 8–14.



(a)

(b)

Figure 6.11: Saliency maps with attentional selection for the second time step. Image (a) is constructed from the training feature weights and (b) is based on the learnt weights.

6.5 Experiment 3: Learning Feature Weights for a Complex Focal Pattern

6.5.1 Defining the Experiment

The third experiment builds a contrived scenario to examine how the learning system handles a situation not entirely solvable by linear weighting of the conspicuity maps. Complex feature interdependencies for the training pattern highlight limitations of conspicuity within this implementation of saliency maps. The discussion relies on the static scene conspicuity maps displayed in figure 6.4.

A fixation pattern directs attention from green, to blue, to yellow, and lastly to the red object. This pattern is repeated for two cycles encompassing a combined total of eight training points. Representative feature weights are learnt using least squares regression. Following we describe the attention pattern to predict the underlying weights and demonstrate how the problem can not be fully solved with the current model.

Winner-takes-all attention is first directed at the green object. This area is only salient in the intensity map so the intensity feature must be highly weighted. Because saliency of the red region is also contributed by the intensity feature, for the red object to be attended last the red-green opponency must have a very small weighting.

The second object of focus is the blue target followed thirdly by the yellow circle. Saliency for yellow area is only affected by the blue-yellow map. For winner-takes-all attention to focus on yellow prior to red, the blue-yellow conspicuity map must be higher weighted than intensity opponents. The blue object would then have very high cumulative saliency defined by both blue-yellow and intensity opponents. The saliency of blue ensures that the green target will not be attended first as in the training pattern.

6.5.2 Results from Learning

This scenario creates a paradox where saliency for the green rectangle can not be highest in the saliency map due to the other feature constraints. While an exact solution may not be possible, the least squares algorithm best fits weights to represent the training data as a whole. Table 6.4 shows weights learnt from the first eight training points.

As predicted, red-green colour opponents receive a very small weight approaching zero. A negative value indicates that to match the training pattern, the circular red object repulses rather than attracts attentional selection. This is understandable given the strong positive bias contributed by the intensity opponent feature. While defining the experiment we examined how interrelated feature dependencies affects the ability to model the training data. Strongly weighted blue-yellow opponents confirm that the green target will not be attended first as in the training pattern.

Feature	Relative Weight
Red-green colour opponents	-0.003428
Blue-yellow colour opponents	0.287295
Intensity colour opponents	0.275939

Table 6.4: Experiment 3: Learnt feature weights.

Attentional selection for the training data and the resulting inhibition maps are displayed in figure 6.12. A white circle designates focal attention for the first ten time steps. Since the training pattern only specifies coordinates for the first eight frames, in all subsequent training maps the attentional focus is determined from weights learnt in real time.

Figures 6.13 and 6.14 show focal attention in the first fourteen saliency maps for training data and learnt feature weights. In a perfectly learnt system the region of maximum saliency is identical for all time steps. Unbiased saliency maps are included for comparison.

Learnt focal attention does not select the green object in the first time step but instead finds the blue rectangle most salient. After this initial failure at modelling the training system, the subsequent response is fascinating. Attention in frame one of the learnt system is the same as focus in frame two of the training data, learnt attention for frame two is identical to attention to frame three of the training data, and so forth. A one frame delay for attentional selection in the learnt and training data points continues through all fourteen time steps without a single error.

Figures 6.15 and 6.16 more clearly show this pattern by delaying the learnt features by one time step. The inability to learn the first frame is expected, but that the learning system accurately computes weights for the underlying training pattern is unexpected. The intent of this experiment was to highlight a situation outside the system abilities. Results show however that even in seemingly complicated situations the least squares learning algorithm can often find a solution to solve large portions of the problem.

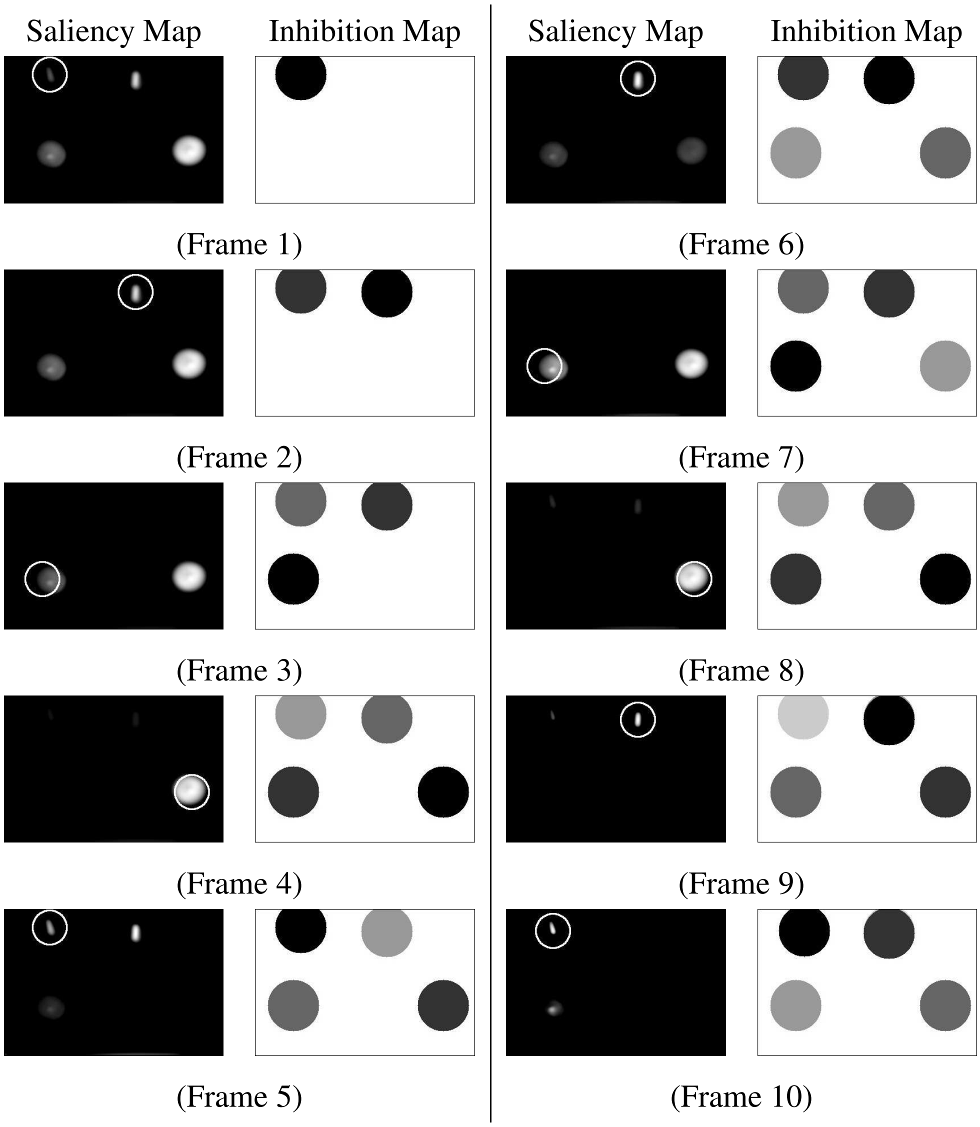
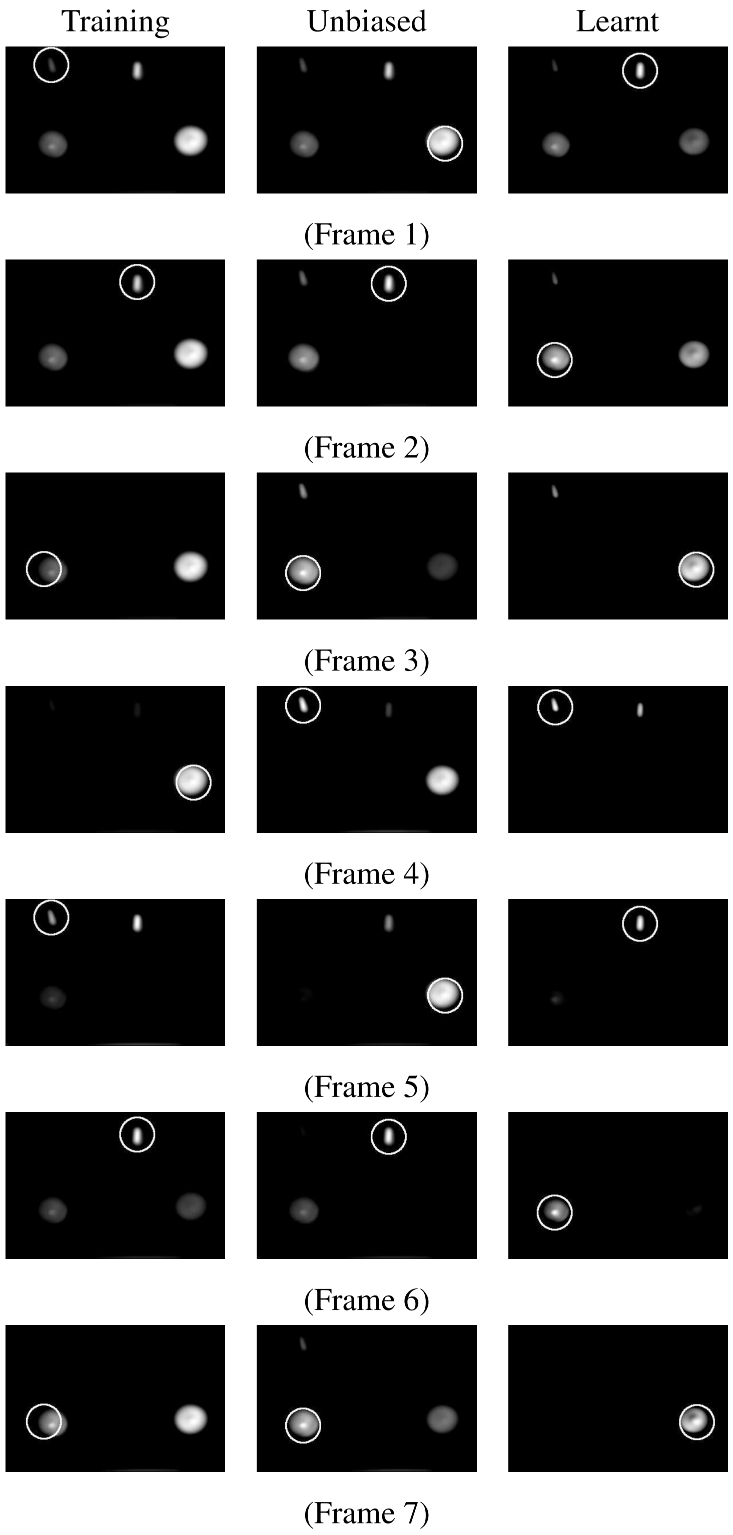


Figure 6.12: Experiment 3: Saliency and inhibition maps for the training pattern of attention.



(Frame 1)

(Frame 2)

(Frame 3)

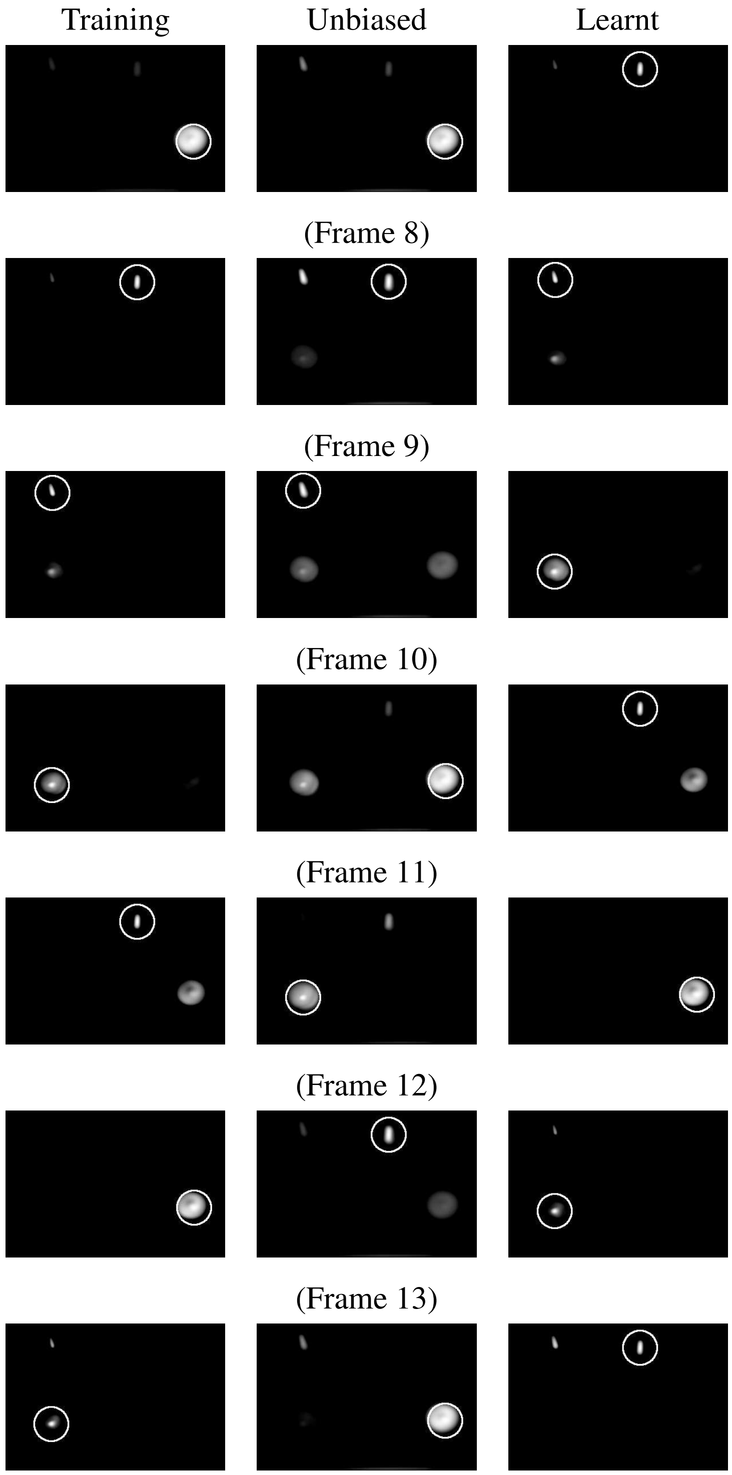
(Frame 4)

(Frame 5)

(Frame 6)

(Frame 7)

Figure 6.13: Experiment 3: Comparison of training, unbiased, and learnt attentional selection for frames 1–7.



(Frame 8)

(Frame 9)

(Frame 10)

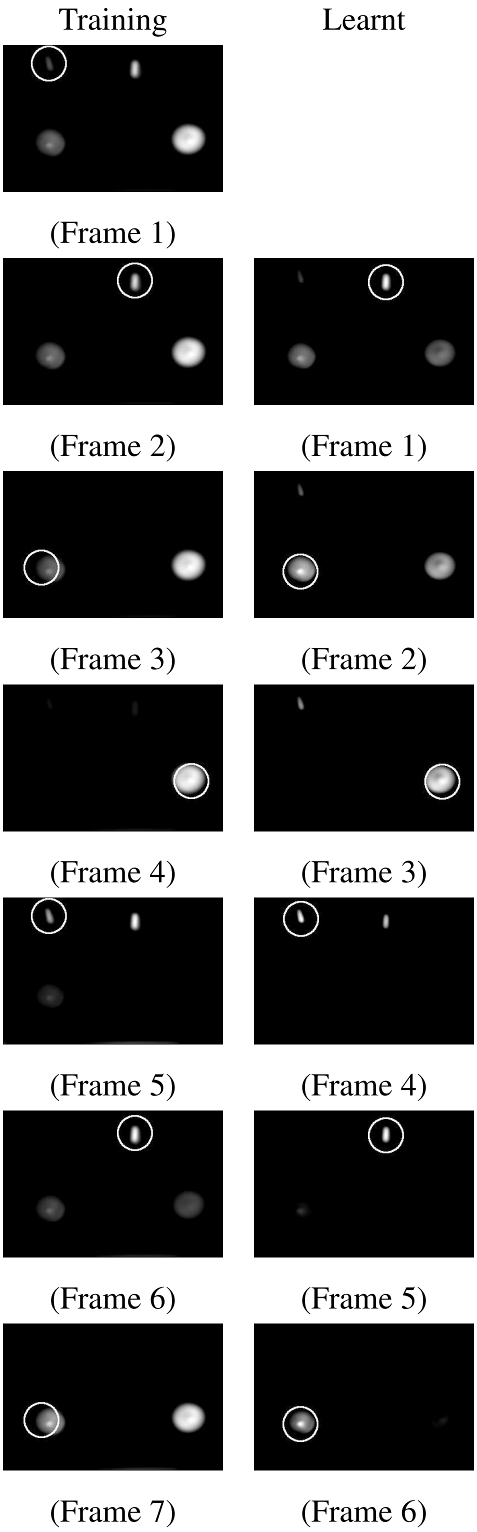
(Frame 11)

(Frame 12)

(Frame 13)

(Frame 14)

Figure 6.14: Experiment 3: Comparison of training, unbiased, and learnt attentional selection for frames 8–14.



Training

Learnt

(Frame 1)

(Frame 2)

(Frame 1)

(Frame 3)

(Frame 2)

(Frame 4)

(Frame 3)

(Frame 5)

(Frame 4)

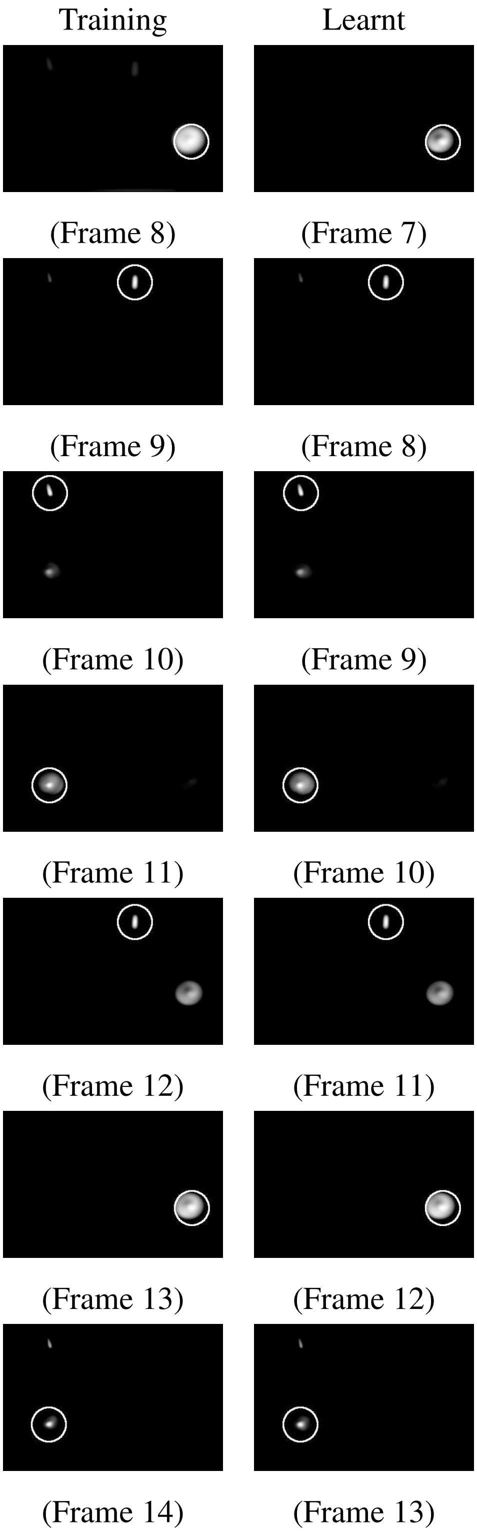
(Frame 6)

(Frame 5)

(Frame 7)

(Frame 6)

Figure 6.15: Experiment 3: Comparison of training and delayed learnt attentional selection for frames 1–7. Learnt attention exactly matches the training data when delayed by one time step.



Training

Learnt

(Frame 8) (Frame 7)

(Frame 9) (Frame 8)

(Frame 10) (Frame 9)

(Frame 11) (Frame 10)

(Frame 12) (Frame 11)

(Frame 13) (Frame 12)

(Frame 14) (Frame 13)

Figure 6.16: Experiment 3: Comparison of training and delayed learnt attentional selection for frames 8–14. Learnt attention exactly matches the training data when delayed by one time step.

6.6 Experiment 4: Learning Unbiased Weights in a Dynamic Environment

6.6.1 Defining the Experiment

The fourth experiment demonstrates functionality of a computational saliency map model using the dynamic environment of figure 6.2. Subsequently the underlying unbiased feature weights are determined from limited training data and are used to compare learnt attention unbiased focal attention. Results from intermediate steps are displayed as greyscale images representing normalised saliency values ranging from zero (black) to maximum (white).

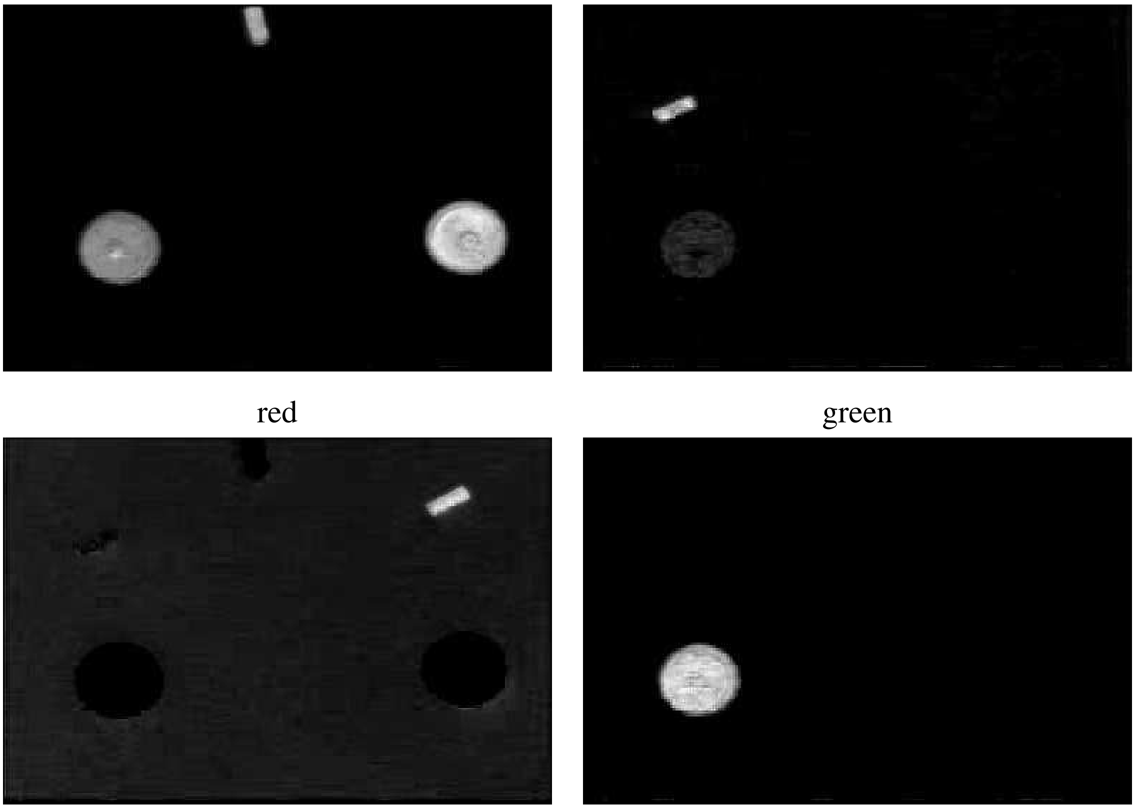
From eight unbiased attention coordinates, the system learns features weights for the red-green opponent, blue-yellow opponent, intensity opponent, and motion conspicuity maps. The intent is to determine feature weights that mimic the attentional response when all features are equally weighted with a value of one. Comparison to unbiased attention for a dynamic scene demonstrates how the additional complexity of a dynamic environment affects learning.

6.6.2 Colour Channels

Colour channels are computed from the dynamic scene and shown in figure 6.17. Results of hue calculations are identical to the static environment with the addition of a moving red object. The red, green, and blue targets are each identified within their respective colour channels and the yellow object is represented among the red, green, and yellow colours. This is explained by the derivation of the yellow channel from red and green components.

6.6.3 Conspicuity Maps

Figure 6.18 displays conspicuity maps for red-green colour opponents, blue-yellow colour opponents, intensity opponents, and motion. The red-green conspicuity map strongly returns the static red circular object while the smaller red and green targets



red

green

blue

yellow

Figure 6.17: Four broadly tuned colour channels from the dynamic image.

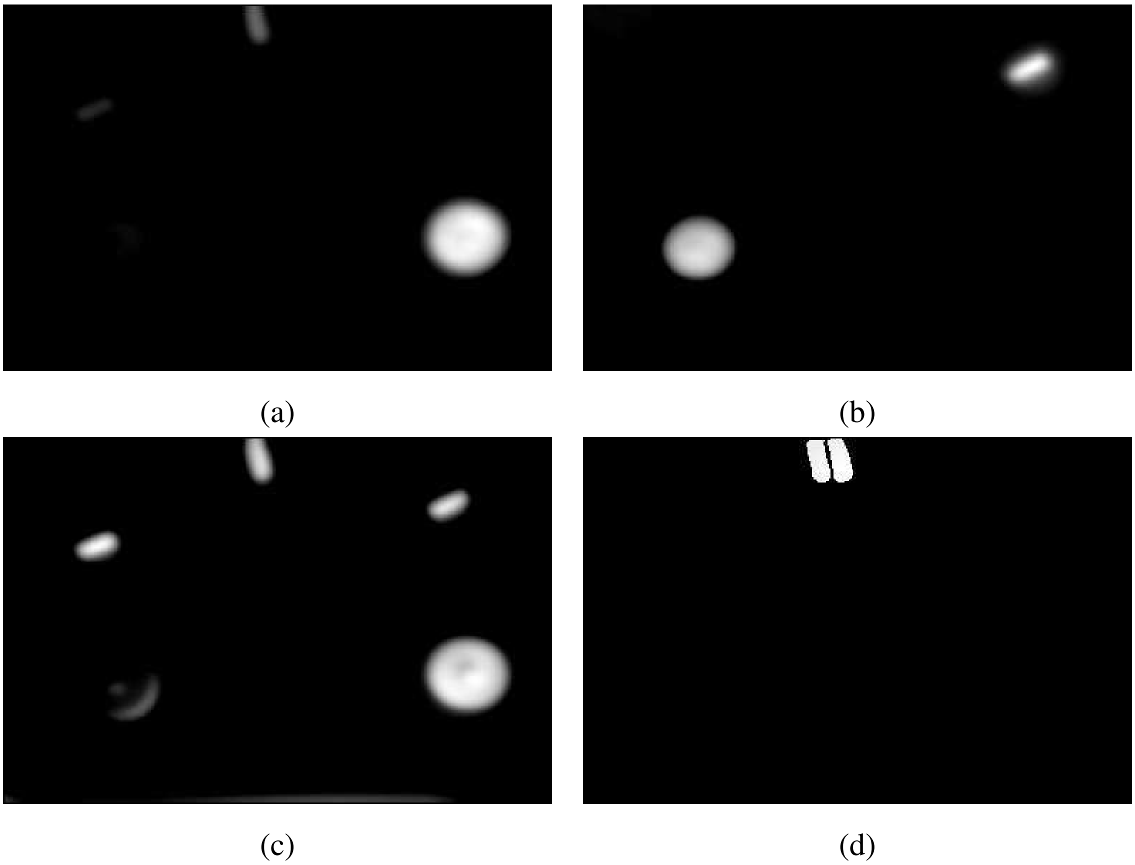


Figure 6.18: Conspicuity maps for the (a) red-green colour opponents, (b) blue-yellow colour opponents, (c) intensity opponents, and (d) motion.

have slight saliency values. Because the large red target has stronger saliency, the other two objects are suppressed by iterative convolutions.

The blue-yellow and intensity conspicuity maps return the same results as for the static scene and so will not be discussed again. Refer to section 6.3.3 for analysis.

Dynamic scenes contain a motion feature not present in the learning problem for a static environment. A motion history buffer determines movement and returns those pixels as fully salient. The motion map changes at every time step with an example visible in figure 6.18. Here the two salient regions signify a change in location between successive image frames. For slower moving object the salient regions can overlap.

6.6.4 Attentional Selection

The four conspicuity maps are linearly summed into an unbiased saliency map shown in figure 6.19. Because the moving red object appears strongly among the intensity opponents and with a maximum binary value in the motion map, it will be attended first in the unbiased saliency map. Representation in both the colour and intensity opponents makes the static red and blue objects the next most salient. Finally the yellow and green objects are the last to receive focal attention because they are strongly present in only a single conspicuity map. These predictions assume equally weighted features.

Figure 6.20 displays saliency and inhibition maps for the first ten time steps. A white circle denotes winner-takes-all attention in each saliency map. There is a strong correlation between actual and predicted results until frames five and six when decaying inhibition of returns makes the red targets more salient than the yet unattended yellow. It is not until the red objects receive a second focus that the yellow target is first attended in frame seven. Frames nine and ten reiterate a focal bias toward red with the two targets capturing attention for the third instance.

6.6.5 Results From Learning

Table 6.5 shows feature weights learnt from the first eight coordinates of unbiased attention. To analyse results of the learning mechanism we assume the conspicuity maps are correct and review the feature weights based on the given saliency implementation.

Motion saliency combines with the intensity map to make the rectangular red object the strongest region in the saliency map. Learnt weights for motion and intensity reflect their importance to capturing attentional focus in the dynamic scene. Because the colour opponents reflect other highly salient regions that are not attended until after the moving object, the learning system determines these hue features are not as important to focal attention and weights them lower than intensity and motion.

Figure 6.21 compares focal selection in the unbiased saliency map to the attention pattern with learnt weight equivalents. The unbiased and learnt systems maintain an identical focal pattern for the first six frames, but beginning in frame seven the atten-

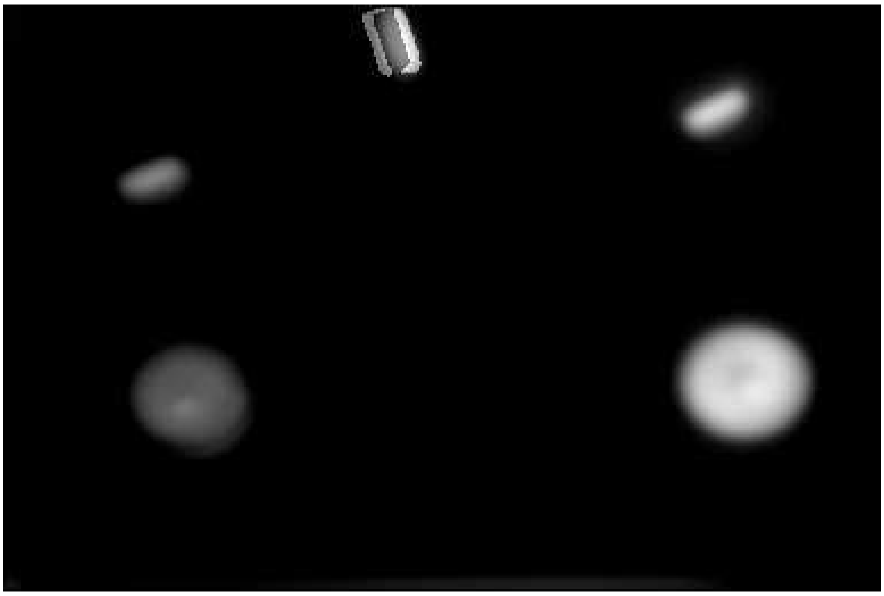


Figure 6.19: The unbiased saliency map when all feature weights are linearly summed without scaling.

tional selection is not consistent between unbiased and learnt maps. Feature weights capture system dynamics within the provided training data, but introduction of a moving target makes the learning system less accurate beyond the explicit training points.

While eight points of training data was sufficient to solve weights in the static environment, this proved insufficient for the dynamic scene. We expect additional training data to improve learning of the dynamic scene but were unable to verify this due to time constraints. An interesting point for future research is to examine how many training points are required to accurately learn feature weights for dynamic environments.

Feature	Relative Weight
Red-green colour opponents	0.142274
Blue-yellow colour opponents	0.191442
Intensity colour opponents	0.319503
Motion	0.265014

Table 6.5: Experiment 4: Feature weights learnt from unbiased attention.

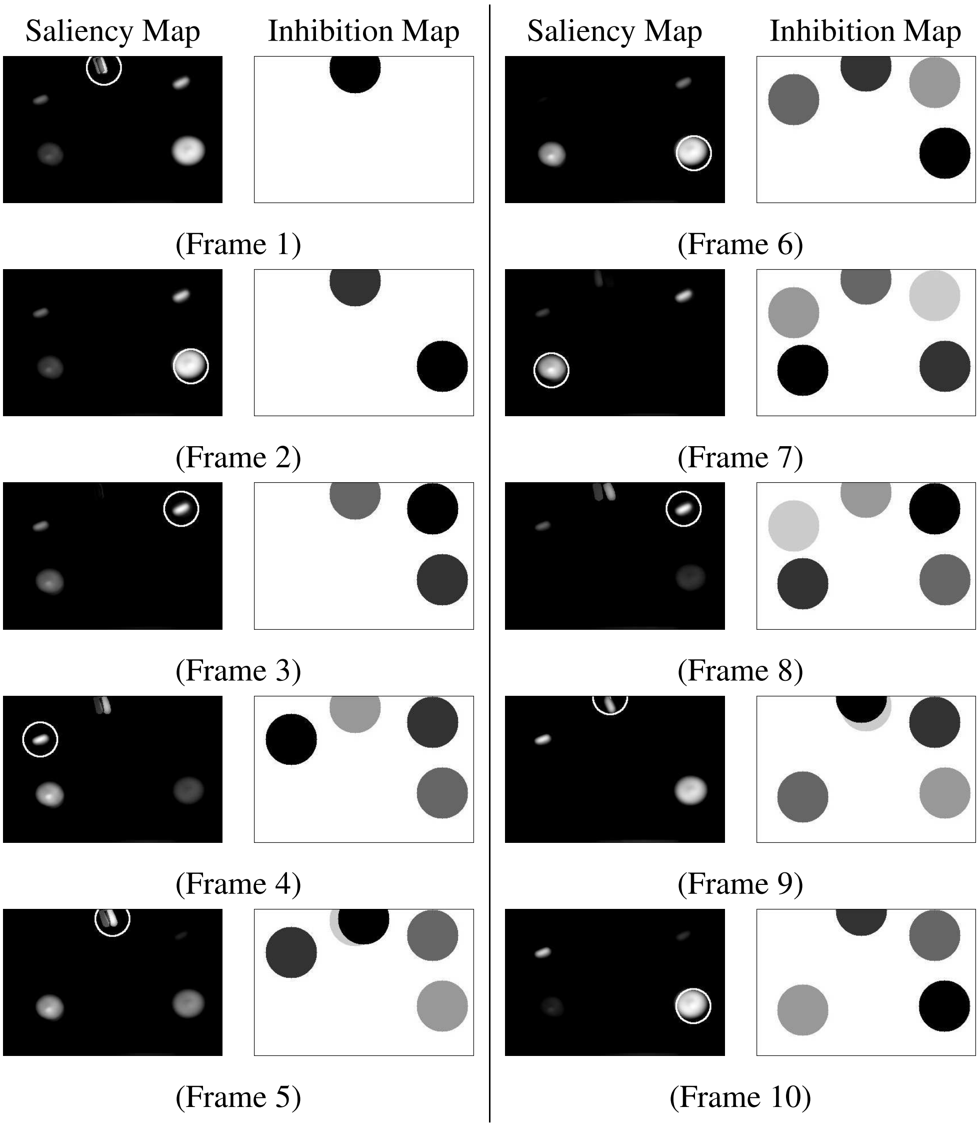
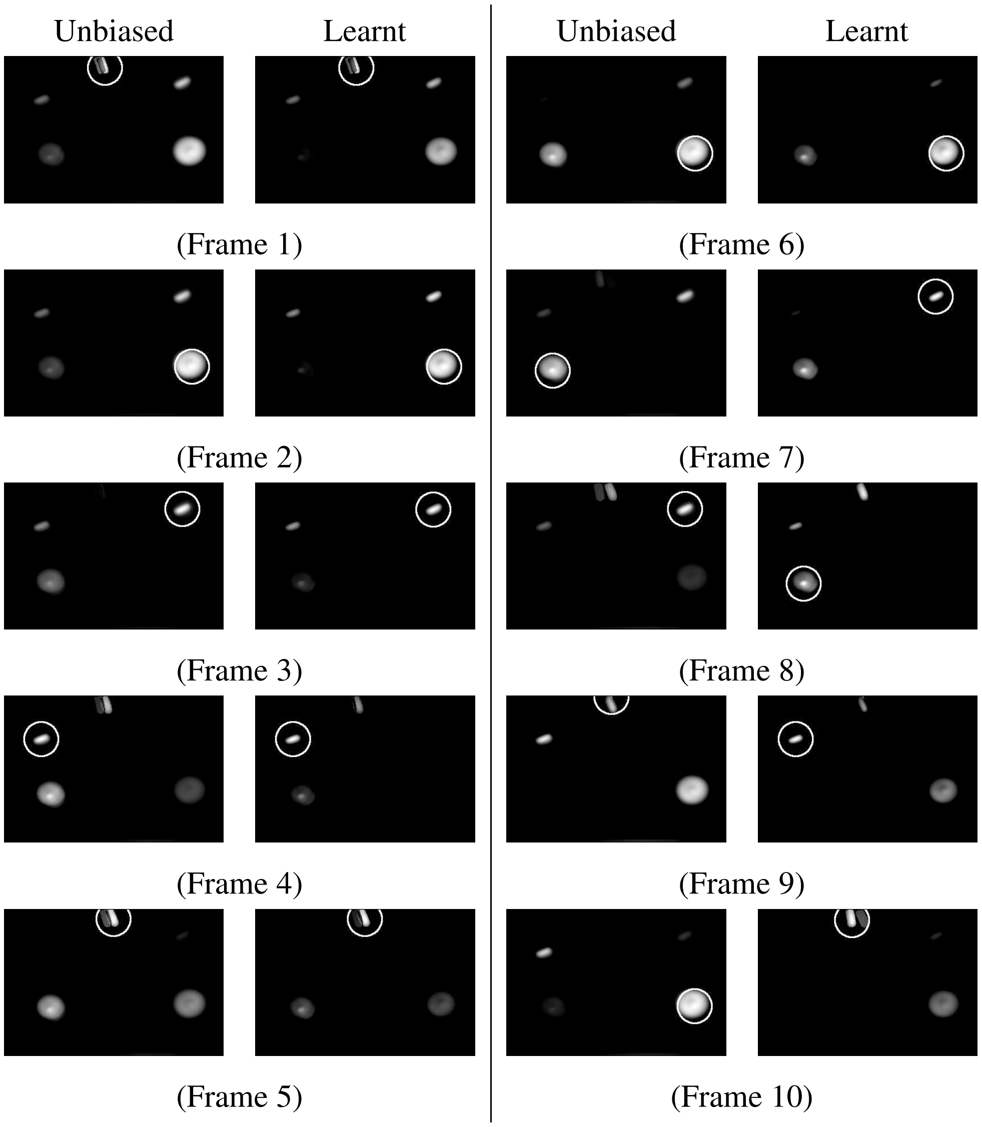


Figure 6.20: Experiment 4: Dynamic selection of the unbiased saliency map.



(Frame 1)

(Frame 6)

(Frame 2)

(Frame 7)

(Frame 3)

(Frame 8)

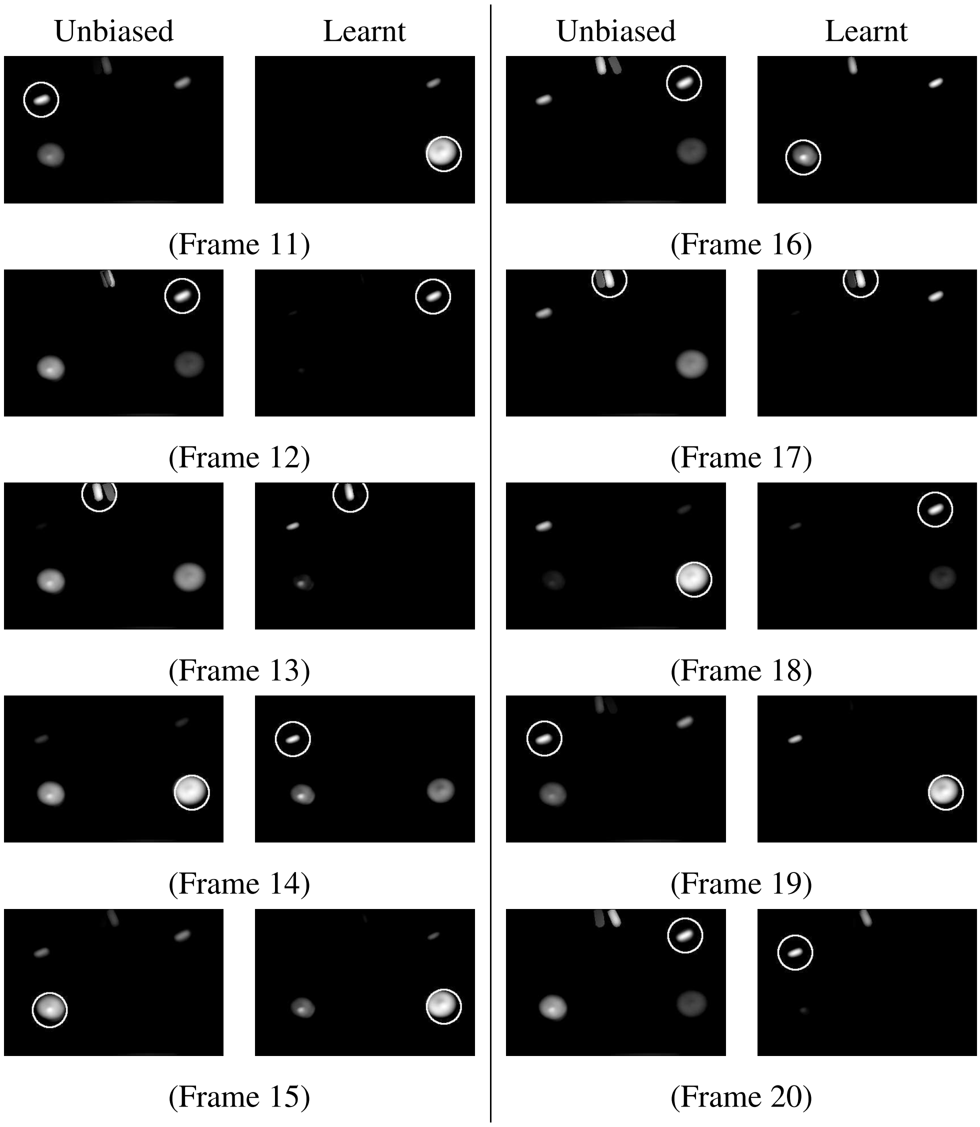
(Frame 4)

(Frame 9)

(Frame 5)

(Frame 10)

Figure 6.21: Experiment 4: Comparison of unbiased focal points and learnt attentional selection for frames 1-10.



(Frame 11)

(Frame 12)

(Frame 13)

(Frame 14)

(Frame 15)

(Frame 16)

(Frame 17)

(Frame 18)

(Frame 19)

(Frame 20)

Figure 6.22: Experiment 4: Comparison of unbiased focal points and learnt attentional selection for frames 11-20.

6.7 Conclusion

Results show that the least square algorithm is able to recover feature weights from a series of focal attention training points. To imitate attentional selection of the training data over many time steps, both the focal pattern and the underlying system dynamics must be captured in the feature weights. With a static scene containing four targets, the least squares method learnt weights that predicted future focal attention with few mistakes. Learning weights for a dynamic environment resulted in a loss of accuracy. We believe a larger number of training points would greatly improve learning.

Chapter 7

Discussion

7.1 Introduction

Chapters three and four detail a computational model of visual attention and method to learn feature weights from eye tracking data. The sixth chapter uses these systems in visual attention experiments and analyses the results for static and dynamic scenes. This chapter relates experimental findings to the computational model and learning system. Opportunities for future research are also discussed.

7.2 Feature Competition

Selecting a convolution kernel for the environment and expected targets also requires consideration of the system computational and timing requirements. While kernels have been proposed in literature for visual attention in natural scenes, one kernel is not optimal for all conditions. That humans readily adapt visual attention to the environment implies that dynamic saliency competition is not purely bottom-up as in the saliency map model but that it is additionally controlled by top-down effects.

In the computational implementation, top-down control takes the form of selecting Gaussian parameters during development. As the solution space for excitation and inhibition functions is exceedingly large, we made a brief comparison of several convolution kernels. However, the red-green conspicuity maps for the static and dynamic experimental scenes demonstrate the difficulty predetermining an appropriate difference of Gaussian function.

An overly strong global inhibition across the scene causes important target elements to disappear from the colour opponents. For the selected parameters $c_{exc} = 0.5$, $\sigma_{exc} = 5\%$, $c_{inh} = 1.5$, and $\sigma_{exc} = 25\%$, the red-green opponent conspicuity map shows reinforced saliency of the circular red area while suppressing saliency of the smaller green region. A necessary reliance on intensity for saliency of the green object coupled with questions regarding the fundamental relationship of intensity to visual attention (Einhäuser and König, 2003), leads us to believe that the selected kernel is too aggressive. Further studies are needed to find a general method of determining optimal convolution kernels for saliency competition based on scene and computational requirements.

7.3 Feature Weights

7.3.1 Learning from Training Data

Chapter six compares focal attention for the training data and learnt weights over multiple time steps. The learnt feature weights are summarised in table 7.1. There is a large difference in the feature influence depending on the scene and attentional pattern, but intensity remains strongly weighted across all experiments because it provides basic saliency for the red, green, and blue elements. Since the yellow object is not captured within the intensity map, the blue-yellow colour opponents are consistently influential to account for attention directed at the yellow and blue features. The red-green colour opponent exhibits the largest variation because in both the static and dynamic scene it primarily controls saliency for the circular red object. The hue feature augments intensity an needed to scale the red target for attention prior to other regions.

Comparison of attentional focus from training data and learnt feature weights defines a measure of learning performance. In experiments involving static scenes, the learnt weights general reconstructed attentional selection of the training data. Mistakes were often based on exchanging the selection priority of two targets and repeated throughout the experiment.

Feature	Experiment 1	Experiment 3	Experiment 4
Red-green colour opponents	0.335055	-0.003428	0.142274
Blue-yellow colour opponents	0.352247	0.287295	0.191442
Intensity colour opponents	0.377396	0.275939	0.319503
Motion	0.0	0.0	0.265014

Table 7.1: Comparison of the feature weights across experiments.

The third experiment intended to demonstrate failure learning weights for an complex problem; however, least squares regression found a partial solution previously unknown to the authors. The learning algorithm was able to disregard the initial point of conflict and determine weights that perfectly reflected subsequent focal points in the attention pattern.

We introduced a dynamic scene to see how the learning system responds to a more difficult situation. The dynamic scene introduces a fourth input parameter and fifth salient region while continuing to learn with eight training points. Unlike in the static scene, the learnt feature weights fail to track attention beyond the explicit training locations. Future study with a larger training set is needed to validate the learning system in dynamic scenes.

7.3.2 Improving the Learning System

During development of the learning system in chapter four, we noted that the least squares algorithm equally weights errors for all pixels. The regression simultaneously tries to maximise saliency at the intended fixation point and minimise saliency elsewhere. While technically correct this is an overzealous approach. A more optimal solution may be found by recognising that several salient locations can coexist so long as the strongest is located at the correct fixation point. Future research into learning visual feature weights can improve the least squares regression method by providing a stronger bias toward minimising saliency errors at the point of attention.

7.4 Conclusion

The saliency map and learning implementation demonstrates varying degrees of success when extracting feature weights from points of focal attention. We suggest two areas for future research: parameterization for difference of Gaussian functions and tailoring the least squares algorithm to bias errors at the focus of attention. With improvements in both areas, future systems could optimise focal attention to changing environmental conditions in real-time.

Chapter 8

Conclusion

8.1 Research Summary

This thesis developed a biologically plausible computational model of visual attention based on saliency maps. To the study of visual attention, the research contributes a novel approach of learning preattentive feature weights from eye tracking data. This work also provides insight into modelling attention within dynamic scenes and highlights the need for continued effort to define a robust means of parameterising intra-feature saliency competition.

Three experiments with static scenes demonstrated that feature weights can accurately be determined from focal coordinates. The system learnt conspicuity weight values that account both for underlying preferences of hue and intensity and also the attentional dynamics of feature competition and decaying inhibition of return. A fourth experiment of learning feature weights from a dynamic scene was less accurate and illustrated that additional training data is needed to account for the extra dynamical effects. We proposed an updated algorithm for learning a more optimal weight solution.

8.2 Future Research

There are many opportunities for continued research into the biological foundation and methods of attention. Going forward it is important to compare learnt feature weights to human performance in similar novel scenes. It is thought that weight models can be extracted and categorised based on individual and scene properties. This requires careful consideration to isolate bottom-up control from the influence of top-down effects.

Another aspect for future research is a focus on how top-down control manipulates the underlying feature weights for a particular scene or task. The existing model of saliency maps is entirely bottom-up though ongoing research attempts to integrate top-down direction into the model (Vaingankar, 2004). By understanding the effect of environment and top-down control on feature weights, an appropriate weight model could be selected for a particular situation. Here we suggest learning feature weights for various individuals and environments. Predictive models could be used to select an appropriate feature weight model based on environmental properties.

As implemented, the system learns only from visual attention coordinates in a scene and discards the temporal path between fixation points. Gaze contains additional information about the importance of unattended feature components. By incorporating scan paths into the learning algorithm a more robust understanding of system dynamics can be achieved. This is especially useful as scene complexity increases to the point where training patterns do not include focus on every saliency object.

8.3 Applications

There are several applications for a physiological understanding of the visual attention system. Transmitting variable resolution video signals reduces bandwidth requirements by only enhancing detail in salient regions (Parkhurst, 2002). With the integration of autonomous mobile robots in hospitals and semiconductor fabrication (fab) clean rooms, there is an increasing need to provide advanced visual sensing to cope with the dynamic environment. An equally if not more important application is to gain understanding of our own human biology.

Bibliography

Amari, S. (1977). Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:77–87.

Braun, J. and Julesz, B. (1998). Withdrawing attention at little or no cost: Detection and discrimination tasks. *Perception and Psychophysics*, 60(1):1–23.

Broadbent, D. E. (1958). *Perception and Communication*. Pergamon, London.

Burt, P. J. and Adelson, E. H. (1983). The laplacian pyramid as a compact image code. *IEEE Trans. Communications*, 31(4):532–540.

Einhäuser, W. and König, P. (2003). Does luminance-contrast contribute to a saliency map for overt visual attention? *European Journal of Neuroscience*, 17:1089–1097.

Engel, S., X.Zhang, and Wandell, B. (1997). Colour tuning in human visual cortex measured with functional magnetic resonance imaging. *Nature*, 388:68–71.

Eriksen, C. W. W. and Hoffman, J. E. (1972). Temporal and spatial characteristics of selective encoding from visual displays. *Perception and Psychophysics*, 21:201–204.

Folk, C. L., Remington, R. W., and Johnston, J. C. (1992). Involuntary covert orienting is contingent on attentional control settings. *Journal of Experimental Psychology: Human Perception & Performance*, 18:1030–1044.

Francolini, C. M. and Egeth, H. E. (1979). Perceptual selectivity is task dependent: The pop-out effect poops out. *Perception & Psychophysics*, 25:99–110.

Freud, S. (1915). *The Interpretation of Dreams*. The Macmillan Company, London and New York.

Funk, N. (2004). Cmput 616 – implementation of a visual attention model. Technical report, University of Alberta.

Hikosaka, O., Miyauchi, S., and Shimojo, S. (1996). Orienting of spatial attention – its reflexive, compensatory, and voluntary mechanisms. *Cognitive Brain Research*, 5:1–9.

Itti, L., Gold, C., and Koch, C. (2001). Visual attention and target detection in cluttered natural scenes. *Optical Engineering*, 40(9):1784–1793.

Itti, L. and Koch, C. (1999a). A comparison of feature combination strategies for saliency-based visual attention systems. In *Proc. SPIE Human Vision and Electronic Imaging IV (HVEI'99), San Jose, CA*, volume 3644, pages 473–82.

Itti, L. and Koch, C. (1999b). Target detection using saliency-based attention. In *Proc. RTO/SCI-12 Workshop on Search and Target Acquisition (NATO Unclassified), Utrecht, The Netherlands, RTO-MP-45 AC/323(SCI)TP/19*, pages 3.1–3.10.

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

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

Koch, C. and Ullman, S. (1984). Selecting one among the many: A simple network implementing shifts in selective visual attention. Technical report, Massachusetts Institute of Technology.

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

Levitt, J. B. and Lund, J. S. (1997). Contrast dependence of contextual effects in primate visual cortex. *Nature*, 387:73–76.

Miau, F. and Itti, L. (2001). A neural model combining attentional orienting to object recognition: Preliminary explorations on the interplay between where and what. In *Proc. IEEE Engineering in Medicine and Biology Society (EMBS)*.

Minsky, M. (1961). Steps toward artificial intelligence. In *Proceedings of the Institute of Radio Engineers*, volume 49, pages 8–30.

Neisser, U. (1966). *Cognitive Psychology*. Appleton-Century Crofts, New York.

Nichols, B., Buttlar, D., and Farrell, J. P. (1996). *Pthreads programming: a POSIX standard for better multiprocessing*. O'Reilly & Associates, Inc.

Niebur, E., Itti, L., and Koch, C. (2001). Controlling the focus of visual selective attention. In Hemmen, L. V., Domany, E., and Cowan, J., editors, *Models of Neural Networks IV*. Springer Verlag.

Parkhurst, D. J. (2002). *Selective attention in natural vision: using computational models to quantify stimulus-driven attentional allocation*. PhD thesis, Johns Hopkins University, Baltimore, Maryland.

Posner, M. I. (1980). Orienting of attention. *Quantatiative Journal of Experimental Psychology*, 32:2–25.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press.

Schachtel, E. G. (1959). *Metamorphosis: On the Development of Affect, Perception, Attention, and Memory*. Basic Books, New York.

Schluppeck, D. and Engel, S. A. (2002). Color opponent neurons in v1: a review and model reconciling results from imaging and single-unit recording. *Journal of Vision*, 2:480–492.

Solley, C. M. and Murphy, G. (1960). *Development of the Perceptual World*. Basic Books, New York.

Tabus, I. (2004). Lecture 10 – recursive least squares estimation. Tampere University of Technology, Tampere, Finland. Class handout for course Adaptive Signal Processing.

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

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

Vaingankar, V. (2004). Goal directed visual search based on color cues: co-operative effects on top-down and bottom-up visual attention. Master's thesis, Rochester Institute of Technology, Rochester, New York.

VanRullen, R. (2003). Visual saliency and spike timing in the ventral visual pathway. *Journal of Physiology (Paris)*, 97:365–377.

Vijayakumar, S. (2004). Lecture 6 – regression: Linear methods for regression. University of Edinburgh, Edinburgh, Scotland. Class handout for course MLSC.

Vijayakumar, S., Conradt, J., Shibata, T., and Schaal, S. (2001). Overt visual attention for a humanoid robot. In *Proc IEEE/RSJ Int Conf Intell Robots and Systems*.

Walther, D., Itti, L., Riesenhuber, M., Poggio, T., and Koch, C. (2002). Attentional selection for object recognition – a gentle way. In *Proceedings of the Second International Workshop on Biologically Motivated Computer Vision*, pages 472–479. Springer-Verlag.

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