Master of Science Thesis

Statistical Saliency Model
Incorporating motion saliency and an application to driving

by Alvin Raj

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of Master of Science at the

Massachusetts Institute of Technology
September 2008

Author. . . . . .

Department of Electrical Engineering and Computer Science
August 26, 2008

Certified by. . . . . .

Ruth Rosenholtz
Principal Research Scientist
Department of Brain and Cognitive Sciences
Thesis Supervisor

Certified by. . . . . .

Edward Adelson
John and Dorothy Wilson Professor of Vision Science
Department of Brain and Cognitive Sciences
Thesis Supervisor

Accepted by. . . . . .

Terry P. Orlando
Chairman, Department Committee on Graduate Students

This work is sponsored by Ford Motor Company.

Statistical Saliency Model Incorporating motion saliency and an application to driving

by

Alvin Raj

Submitted to the Department of Electrical Engineering and Computer Science on August 29th, 2008, in partial fulfillment of the requirements for the degree of Master of Science

Abstract

This thesis extends the Statistical Saliency Model to include motion as a feature, enabling it to compute the saliency of video sequences more effectively. The motion feature is represented as optical flow and incorporated into the model. The model is validated by testing its capability in predicting reaction time performance in a driving simulator. We find that the model does help predict reaction time and some eye-movements in some simulated driving tasks.

Thesis Supervisors:

Ruth Rosenholtz
Principal Research Scientist

Edward H. Adelson
John and Dorothy Wilson Professor of Vision Science

Contents

Acknowledgements

I had come from a background of applied mathematics and computer science, and armed with vague idea of how I thought vision would be a very intriguing area of research. Thus, I entered a field of computer vision and human vision, ending up feeling very uncertain about how things work and how to make sense of things.

Thankfully, through the help of my amazing (and very patient) advisors, Ruth Rosenholtz and Ted Adelson, I learned a tremendous amount about human and computer vision. I owe many thanks to Ce Liu for helping me understand motion, in addition to sharing code with me. Thanks to Lavanya Sharan, and Yuanzhen Li, for showing me the ropes and helping out in uncountable ways.

I would also like to thank my colleagues at Ford Motor Company for helping with the experimental design iterations, and for being very supportive of my work throughout the thesis duration. I especially want to thank Dev Kochhar for always taking the time to assist me, advise me, and making me feel at home while I was visiting Ford. I also extend many thanks to Louis Tijerna for being a source of many interesting ideas, being friendly, and helping me with many aspects of experimental setup while at Ford.

I thank my mother who has patiently supported me in many ways, and my brother who has been very amusing in slow times. And lastly, a deep gratitude to my friends who have been there for me.

Throughout my experience at MIT, the most important thing I learned is what my advisors taught me about science. While I thought I knew what science was through taking physics and chemistry classes in college, it barely scratches the surface. What I treasure most in my experience here was being able to learn how to think scientifically. This quote by Schwartz made in the Journal of Cell Science (2008) best summarizes my feelings on this topic, "One of the beautiful things about science is that it allows us to bumble along, getting it wrong time after time, and feel perfectly fine as long as we learn something each time. .... The more comfortable we become with being stupid, the deeper we will wade into the unknown and the more likely we are to make big discoveries."

Chapter 1

Background

This thesis is primarily concerned with extending the work of the Statistical Saliency Model [19] to incorporate motion, and to apply it to predict behavior in some driving tasks. We begin with understanding the psychophysical basis of the model, detailing the mathematical model, showing how it would be useful in a driving application, then illustrating some experiments conducted in this work.

1.1 Introduction

The human visual system is a complex and evolutionarily advantageous system in the human brain, comprising between a third to half of the brain's volume. The sensory inputs from the retina is piped to the visual cortex and parsed as some internal representation in the brain. This representation is, for the most part, stable and unaffected by top-down cognitive processes (i.e. one cannot by force of will change the color of this paper to blue). In addition, we are not conscious of every detail within a scene. For example, we are only able to track a limited number of objects at a time when presented with a stimulus with multiple moving objects [3]. These are some of the pieces of evidence that imply the processes of cognition and attention do not have instantaneous and complete knowledge of the visual information that the brain encodes. However, there cannot be complete independence between these processes, because we know that there exist some classes of stimuli that involuntarily attract our attention, i.e. the "pop-out" effect (Treisman et al [24]). This will be discussed in more detail below.

We consider the idea that the pop-out effect is on the extreme end of a spectrum of bottom-up mechanisms in attracting attention to a region of interest in the visual environment. Our notion of visual saliency is that the degree to which a given spatial location differs from its surroundings, is predictive of the

likelihood of attention applied to that location, regardless of the task at hand. We first look at why attention is needed in the first place, and then consider saliency as a task independent mechanism that helps select the important parts of the scene for attention to be applied.

1.2 Attention

The inquiry into the nature of consciousness and attention have grasped in turn the consciousness and attention of many philosophers and scientists over the ages [10]. For the purposes of this paper, we concern ourselves with bottom-up attention. The brain receives a tremendous amount of sensory input from millions of neurons. If the processing capacity of the brain is unable to handle such a massive amount of information, then it would be a positive survival trait to be able to discard the information which it does not need to survive and to attend to the information which it does need to survive.

As summarized in [16], the brain pays attention to things that are going to be damaging (i.e. that lion is going to attack me), nourishing (i.e. that fruit is edible), a potential mate (i.e. it wants to mate with me) [14], or some previously known pattern [5]. Brains also pay attention to novel stimuli in order to categorize it into one of the previously mentioned categories. For the purposes of this paper, it is sufficient to focus on a loose definition of bottom-up attention – the process of simple, low-level visual features computed in early vision ($< 150ms$) attracting attention to spatial regions, regardless of the task at hand, and we start with some of the experiments that shed light on how this might work.

1.3 Saliency and Visual Search

When subjects in a visual search experiment were given the task of finding a particular target in the middle of some distractors in a stimuli, Treisman found that there are some targets that 'pop-out'. These targets are so easily and quickly identified that the time it takes for the subjects to find the target remains constant regardless of the number of distractor items in the stimuli. Figure 1.1 shows an example of a pop-out stimuli. In common with all the stimuli that exhibited this pop-out phenomena, was that the target differed significantly from all its distractors in at least a single feature (i.e. a green bar among red, or a horizontal bar among diagonal bars, etc).

In contrast to the pop-out phenomena, there exist search tasks, for which the reaction time to find the target grows linearly with the number of distractors. One example of this is the task of searching for a target gray horizontal bar, among white horizontal bar distractors, and gray vertical bar distractors (see Figure 1.2). These stimuli have two populations of distractors, where each population

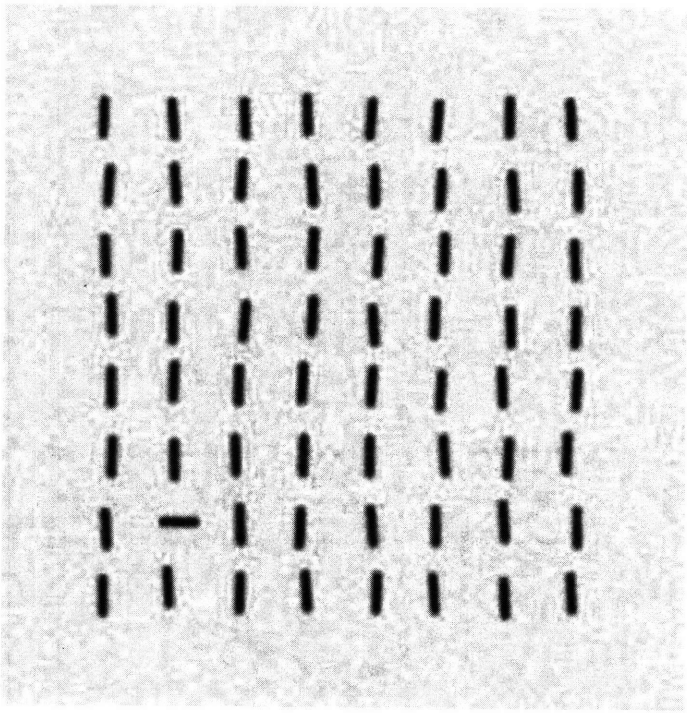

Figure 1.1: Visual Pop-out

shares at least one feature with the target. This concept is represented in Figure 1.3. If looking at only one feature at a time, we notice that there is no way to distinguish the target from both populations of distractors, and it is necessary to jointly consider both feature distributions in order to make the distinction.

Treisman proposed the Feature Integration Theory (FIT) [24] in order to explain these results. These experiments provide psychophysical evidence that the visual cortex represents visual stimuli with some number of separate feature maps (i.e. orientation, luminance, etc) extracted from visual input. FIT states that the reason why conjunctive search shows a reaction time proportional to the number of objects in the screen is that these parallel feature maps require an attentional spotlight to 'bind' the various feature maps together in order to 'know' that some object is the target.

While FIT predicted that conjunctive search would not have the 'pop-out' effect, and was validated by results of most conjunctive search experiments, there also existed some conjunctive searches (for example color and depth) which actually do exhibit pop-out. The main contribution of FIT can be thought of as asserting a representation of vision as parallel feature maps, but this alone is insufficient to explain the results from visual search experiments. Visual search remains a problem that has yet to be solved in general, but there exist many models which explain a great variety of phenomena [8, 27, 25]. What experiments have shown is that how much distractor objects differ from the target within

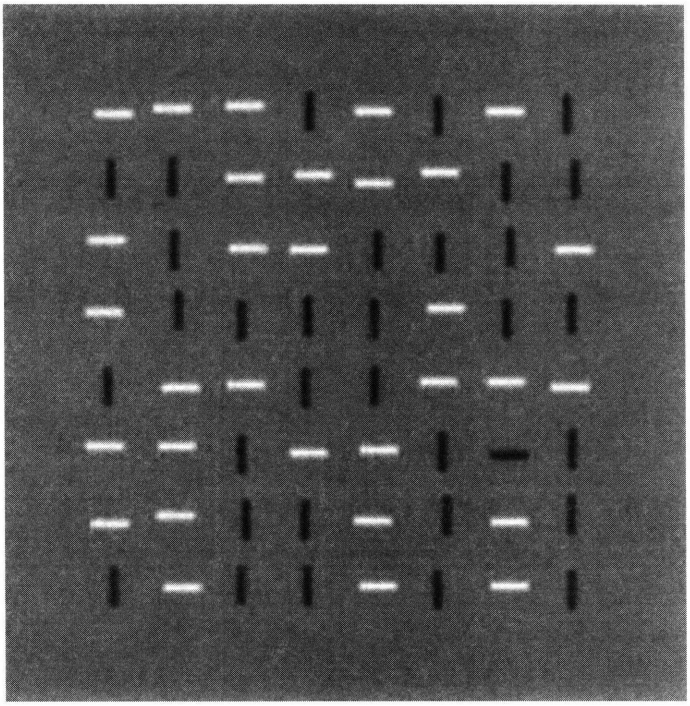

Figure 1.2: Conjunction Search: The target is a dark horizontal bar

the feature maps has been a strong indicator of how fast a subject will find the target, and has been the motivating idea behind the notion of bottom-up visual attention.

1.4 Related Work

Wolfe et al. proposed the Guided Search model which uses discriminability in simple features to guide attention [8]. In addition, it simulates attentional capacity by introducing a bottleneck for information processing ability to limit the number of objects that are 'considered' in order to more closely model the human performance in typical visual search tasks. In some ways, this is an implementation of the FIT. This model however, is restricted to visual stimuli for which we know the location and size of every object, and their distribution in feature spaces, and has not been extended to be able to process arbitrary real-world stimuli.

Koch and Ullman proposed a neural mechanism for bottom-up visual attention [11], which was the basis for a later implementation as a computational bottom-up saliency model by Koch and Itti [13]. Their model quantifies saliency as the difference in mean of a location compared to its surround, in multiple single dimensional feature maps. Saliency is computed by filtering the feature maps with a center surround filter to extract the difference in means, allowing

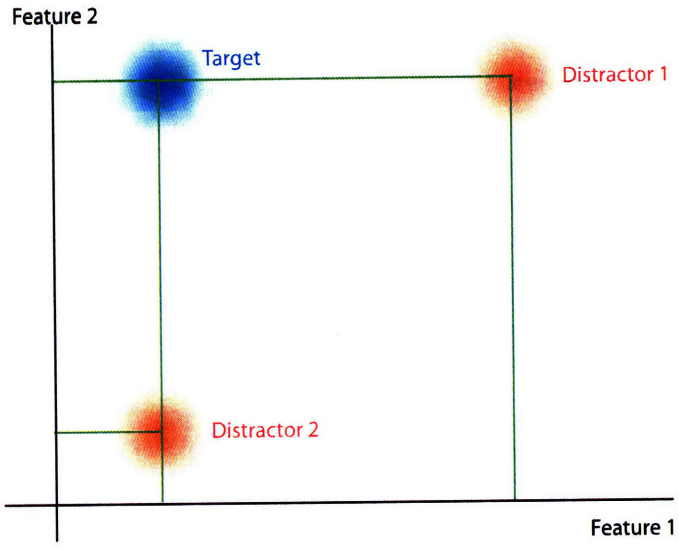

Figure 1.3: Distribution of features in a conjunctive search task

the model to be applied to actual images. However, this model does not take into account distractor variance – that when distractors have a lot of variance, the reaction time is slower than when the distractors are homogeneous.

Rosenholtz proposed a simple statistical outlier based approach, further refining the notion of discriminability or saliency, which does take distractor variances into account [22, 19]. This model will be discussed in detail in the following chapter.

1.5 Motivation

Not only does visual saliency present an interesting intellectual problem, but also does it have many practical real-world applications. Having a model of what humans are likely to attend to can be used to bridge the gap between humans and computer interactions – in a sense, allowing the computer a very simplified 'theory of mind' of its user. Computer programmers, user interface designers, and advertisement makers have implicitly been using saliency for a long time intuitively to attract a person's attention – the blinking of text on webpages, flashing neon signs, etc.

In a driving context, allowing an intelligent vehicle to know what its driver is attending to, allows the intelligent vehicle to alert the driver to threats that the driver has not attended to. For example, suppose an intelligent vehicle had both cameras showing the visual field outside the windshield, and a laser range finder which allows the vehicle to identify objects in close proximity. If a pedestrian wearing black at night suddenly crosses the street, the cameras would only know

that there is not a lot of saliency in the scene since the pedestrian does not differ much from its surrounding. However, the laser range finder would have less problems identifying it as a threat since it functions by sending out a laser signal and records the reflections of the signal – and will detect the pedestrian without trouble. Then, knowing that the threat exists, and that the driver is unlikely to have noticed the threat, the vehicle can then alert the driver to the unnoticed threat. This thesis focuses on a driving application for saliency, and makes the case for why saliency would matter for this application.

Chapter 2

Bottom-up Saliency

As mentioned previously, saliency can be viewed as a component in a visual search model that tries to explain why some visual search tasks are easier from a bottom-up perspective. The Statistical Saliency Model makes the claim that a location is more likely to attract attention depending on how much of a statistical outlier it is in a multi-dimensional feature space compared to the distribution of its surrounding in the same feature space. We first describe the algorithm qualitatively, then delve into the details. This thesis is focused on extending the Statistical Saliency Model which was previously implemented and tested on static stimuli, to the domain of videos or dynamic stimuli, and testing the efficacy of its predictions on a practical application.

2.1 Statistical Saliency Model

The center-surround differencing as a measure of saliency as proposed by [13], does not translate easily to features that span more than one dimension. This is because if a feature has more say two dimensions, the difference between the center and surround will then be two-dimensional instead of a single scalar value quantifying how salient it is. Some examples of features that span multiple dimensions include color (which in [13] is dealt with by means of color opponency), and motion which typically requires two dimensions to describe. An alternative approach to the center-surround differencing is proposed by Rosenholtz [22, 19, 20, 21], which takes on a signal detection approach.

While the center-surround differencing uses only local mean features to compute saliency, there has been evidence that the representation of feature maps, in addition to the spatially local mean of the features, includes a measure of variance or uncertainty which may either be implicitly or explicitly encoded. For

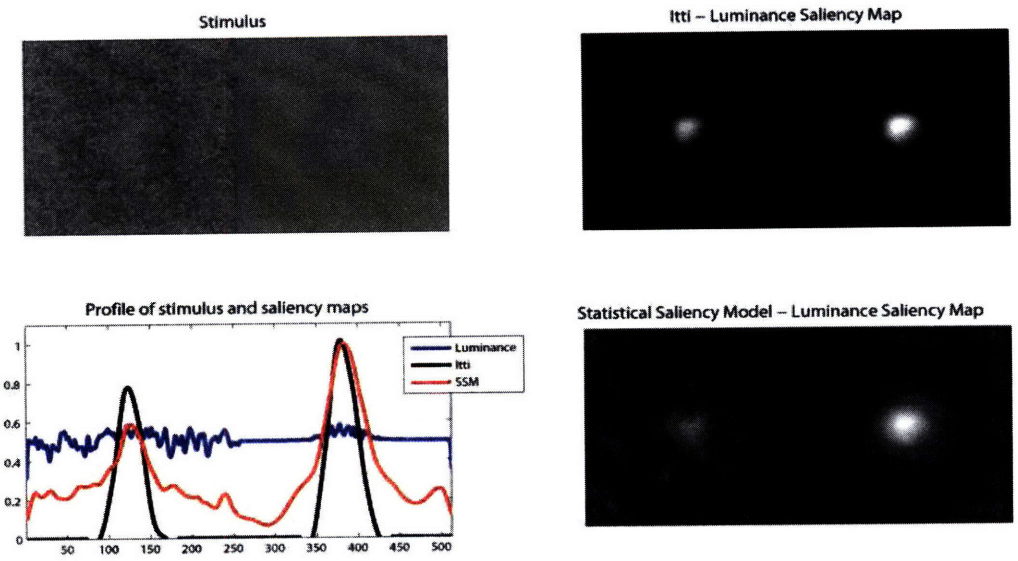

Figure 2.1: Top Left: The stimulus is composed two side-by-side squares of a bright center surrounded by a darker surround, such that the mean of the centers and surrounds of both sides are equal, but the variance of the left side is much higher than on the right side. Top Right: The luminance saliency map output from Itti's algorithm. Bottom Right: The luminance saliency map output from the Statistical Saliency Model. Bottom Left: A plot showing the profile takes across the stimulus of the profile and the two saliency outputs.

purposes of computational simplicity, we approximate what the neural system might encode, by estimating for each spatial location – a local mean x, the mean of the surrounding location μ, and the covariance matrix of the surrounding location Σ for a given feature space. These three parameters can be used to compute the Mahalanobis distance, $s = \sqrt{(x-\mu)^T \Sigma (x-\mu)}$, which nicely corresponds to a quantitative measure of the 'outlier'-ness of a spatial location for a multi-dimensional feature. Intuitively, it corresponds to how many 'standard deviations' away the target is from its surround in a given feature space.

In addition to being easily extendible to feature spaces that span more than one dimension, it has the benefit of taking the variance of the surrounding distribution into account when deciding how salient a spatial location is. Figure 2.1 shows a comparison of luminance saliency as center-surround differencing to compared to Mahalanobis distance. Notice that the side of the image with no variance in the surround is more discriminable (i.e. salient) according to the Mahalanobis distance and human perception.

2.2 Modeling

In order to tractably implement our saliency model, we make simplifications in modeling the physical and biological processes. The physical input we are concerned with is an infinite dimensional spectrum of electromagnetic waves, localized at a region in space-time. Since we concern ourselves with only what humans are able to detect, we only need consider the spectrum of visible light, which we can detect with sensors tuned to red, green, and blue. We then sample the continuous spatial region into discrete, equally-spaced spatial coordinates, and sample time at equally spaced intervals to obtain a snapshot of visible light at a regularly sampled time interval. This is then a spatio-temporal volume representing the visible light at a region of space over time (e.g. the CCD array of a video camera).

The goal is to obtain a value of saliency for each discrete space-time coordinate. We compute the saliency for each point on the image as though the eye were fixated on it, noting that if the eye weren't fixated on it, the visual attention applied would not be very significantly different anyway. Then we can obtain the static saliency computation by considering each frame independently, and computing the saliency contribution from the features we extract – Color, Contrast, and Orientation. By extracting the optical flow from the spatio-temporal volume (i.e. the video), we can then compute motion saliency as well.

In order to more closely match human perception which is able to perform the same computations at multiple scales, we extract the features at high resolution, and successively shrink the image and perform the computations again. This method for image processing on multiple scales is called the Gaussian pyramid [1]. We apply this for each frame in the video. The pyramid may not necessarily be dyadic which is just a size ratio difference between levels of $r = 0.5$, but may instead have an arbitrary ratio of image size between levels, $0 < r < 1.0$. This is done by successive smoothing and resampling with bilinear interpolation to resize the image. The features at every scale level of the pyramid are then extracted. Since it is not known what value of r is optimal, we have settled on an arbitrary choice of $r = 0.75$ to use in our model. Then, for each of these feature maps at every scale level in the pyramid, we compute the local saliency at every location.

2.3 Static Saliency

2.3.1 Color

While the direct physical representation of the three-dimensional color feature is useful for modeling the spectrum falling onto the retina, we are more interested in the perceptual space that the spectrum is effectively mapped to by the visual system. The CIELab color space was developed to map colors to a perceptually

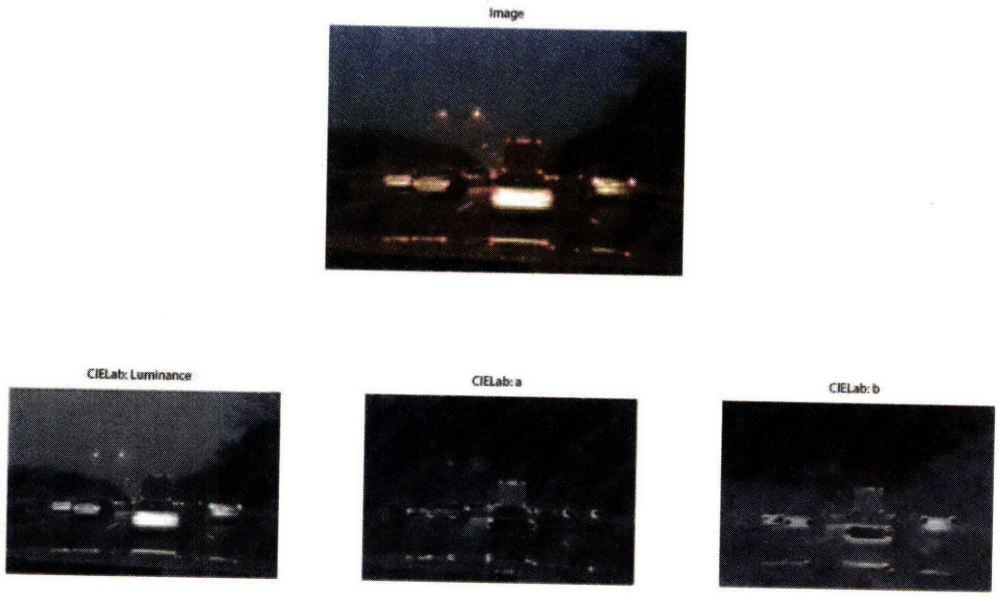

Figure 2.2: CIELab color space from the image

uniform space (i.e. the perceptual difference between two colors is proportional to the Euclidean distance in CIELab space), and is therefore the feature space of choice for this model.

We note that the saliency model put forth by Itti [13] makes explicit use of color-opponency (that the visual system records differences in the responses of the three different cones (color receptors) in the retina) in their computation of color saliency. Color-opponency theory makes the case for three opponent channels – (red-green), (blue-yellow), and (black-white). We implicitly use color-opponency by means of the CIELab color space. CIELab essentially tries to use the ideas and results from color-opponency in finding a mapping from RGB color space to a perceptually uniform space. In trying to make this a perceptually uniform space, CIELab also partially accounts for the effect of Weber's Law in luminance perception – that the smallest noticeable difference from a given starting amount of luminance is always some constant fraction of the given starting luminance.

CIELab does a relatively good job in modeling a perceptually uniform color space, but does not model it perfectly. However, given its simplicity in computation and its decent performance in modeling perception of color, we use it as the three-dimensional feature space in which to compute color saliency. Please refer to [4] for details of how to convert from RGB color space to CIELab.

We provide an example of extracting CIELab from an image 2.2. The a channel is a rough representation of 'redness' versus 'greenness' (for example, the red brake lights have large values in the a channel in the sample figure), while the b channel is a rough representation of 'yellowness' versus 'blueness' (for example, the blue sky has small values in the b channel in the sample figure).

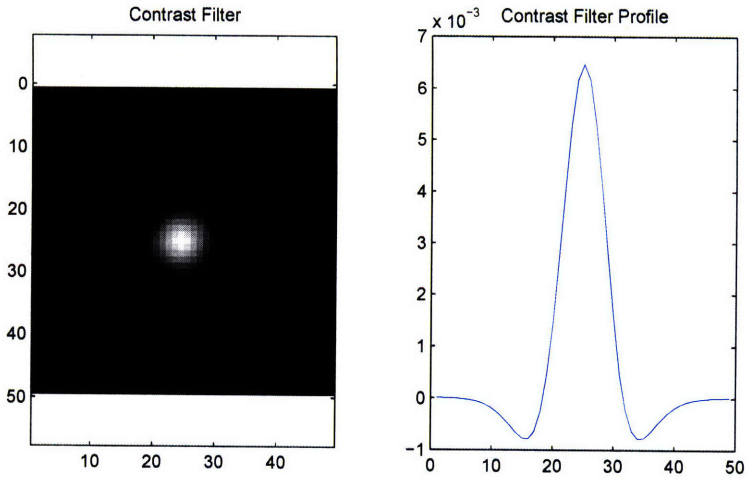

Figure 2.3: Contrast Filter

2.3.2 Contrast

While color has a direct and easily measurable physical cause, item size is harder to measure directly. Item size is one of the many features shown to exhibit pop-out. It is a very difficult problem to obtain a measure of size, because in an arbitrary image, we do not know where objects are and where their boundaries lie.

There has been much progress in the field of image segmentation to obtain these object boundaries more quickly and in a more perceptually accurate manner. However, since these methods are both computationally intensive and some do not have stable outputs, we instead use a simple alternative – luminance contrast.

The contrast feature map is obtained by filtering the luminance channel (from the CIELab colorspace) with a simple Difference of Gaussians filter (see Figure 2.3), then squaring the result to obtain contrast energy. If we make the simplifying assumption that an object consists of a closed shape of uniform luminance, this feature gives an estimate of how large an object is by essentially counting the number of nearby spatial locations that exhibits the change in luminance that occur at the boundary of two objects or surfaces. This contrast energy is also a stand-in for shape – a measure of 'textureness'. As shown in 2.4 we notice that the image has two populations of textures – and the contrast energy can distinguish between the two.

Figure 2.5 shows an image and the extracted contrast energy map. We provide the code for extracting contrast energy in the Appendix.

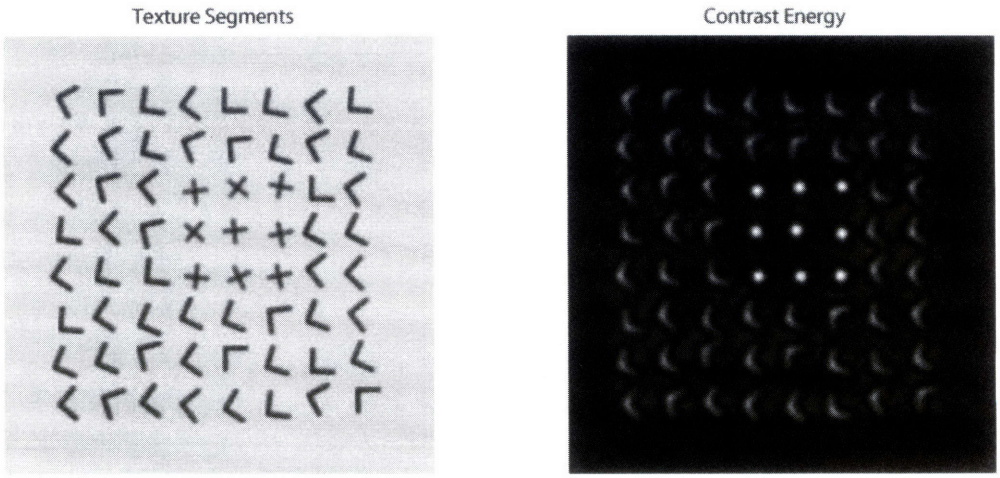

Figure 2.4: Contrast Energy as a measure of 'textureness'

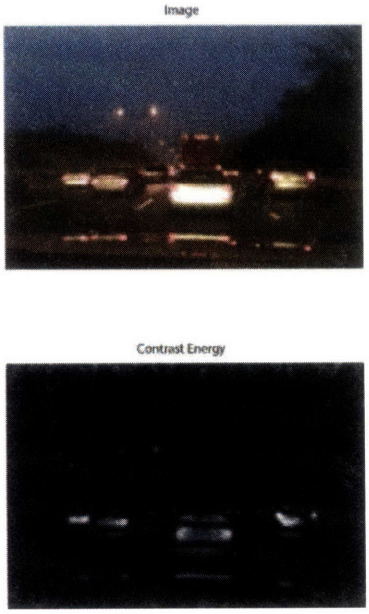

Figure 2.5: Extracting Contrast Energy from an image

2.3.3 Orientation

The orientation feature maps we use are based on those used in [12]. We use a two-dimensional feature representation for orientation, with the first dimension corresponding to horizontal (0 degrees) - vertical (90 degrees) opponency, and the other corresponding to opponency in the two diagonal (45 degrees and 135 degrees) orientations. This is essentially a version of steerable filters [6], extracting some measure of the best angle response to a steerable orientation filter.

We first extract the responses to a horizontal, vertical, left and right diagonal

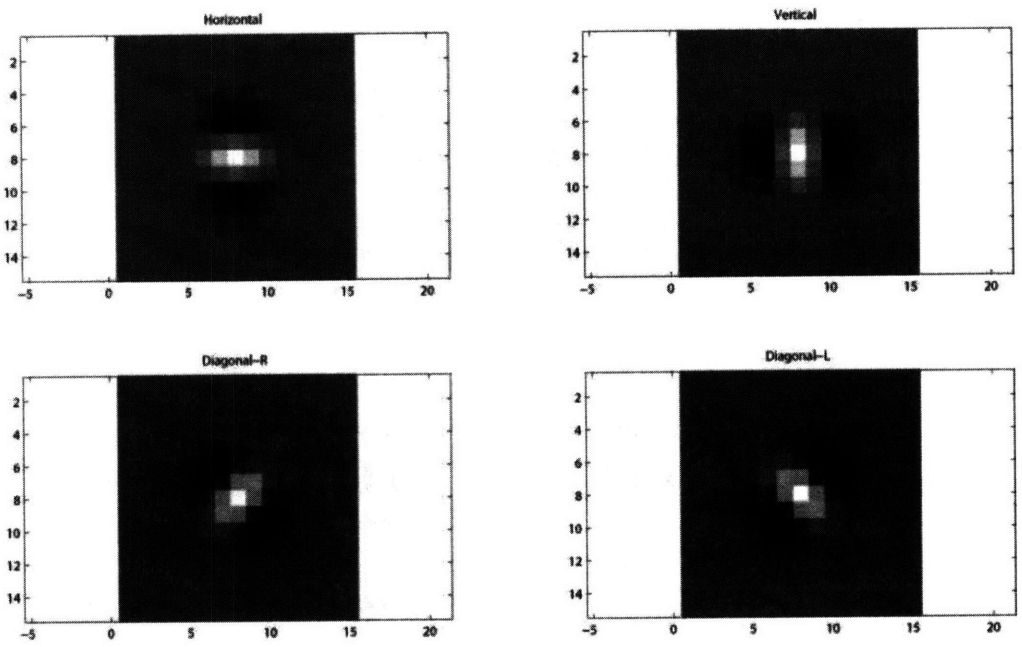

Figure 2.6: Orientation Filters

filters (see Figure 2.6) from the luminance map from the previously mentioned CIELab color space representation of the input image. These responses are then spatially pooled and squared to obtain the orientation 'energy'. Then, we obtain horizontal-vertical opponency by taking the difference between the horizontal and vertical energy and dividing it by the total energy (which is just the sum of the four orientation energy computations). In the same way, we use the two diagonal orientations to find the left diagonal - right diagonal opponency. As an example we show an image's orientation features extracted in Figure 2.7. We also provide the code for extracting this feature in the appendix.

2.4 Motion Saliency

The focus of the thesis is the extension of the computational form of the Statistical Saliency Model to the domain of dynamic stimuli. Ideally, we would like to use a feature space that closely models the representation of motion in the visual cortex. However, it is both not clear exactly how motion is encoded, nor how to compute what is encoded from the retinal image. Cortical analysis reveals that there are brain cells that are tuned to specific directions of motion, and is consistent with the local representation of spatial-frequency based representation of motion extracted through various local spatio-temporal filters of the stimuli [26, 2]. These local estimates are thought to be integrated to form a global motion field [17] in order to correct the mistakes that the local filters may make.

We consider optical flow as a feature representation of motion for it's simplic-

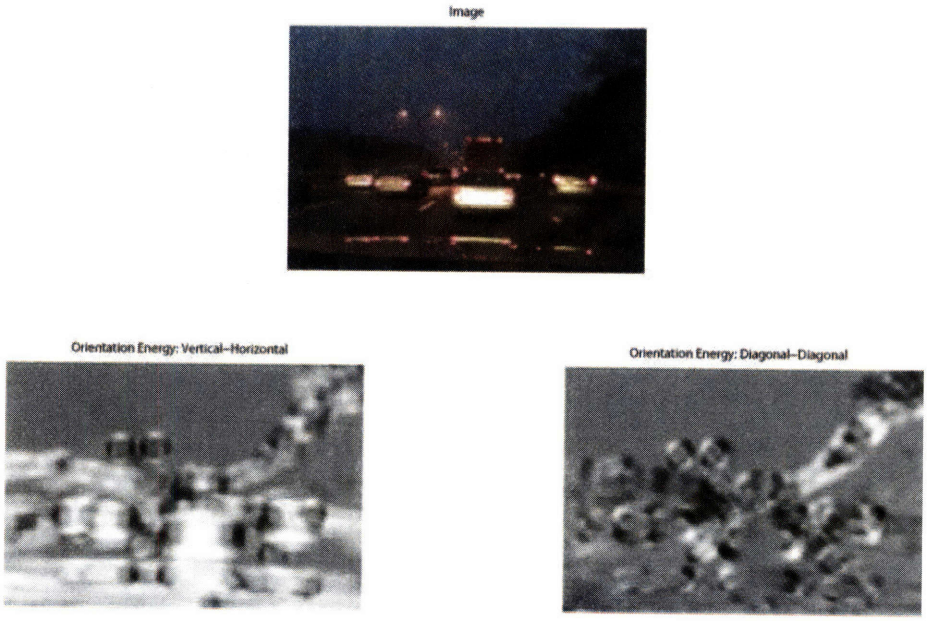

Figure 2.7: Extracting Orientation Energy from an image

ity in formulation, its representational power, and compatibility with local flow estimation and some aspects of global motion. Optical flow encapsulates the idea of finding a correspondence for every pixel between successive frames in a video sequence. We then have for every pixel, its motion to the next frame. Thus, for every spatial location, we have a horizontal and vertical displacement, forming a two-dimensional feature space.

We once again are faced with a representation choice – to encode optical flow as horizontal and vertical displacements, or to encode it as a magnitude and direction. As Rosenholtz points out in [20], representing optical flow as a two-component spatial displacement as a feature for motion would allow the Statistical Saliency Model to perform closer to psychophysical observations as compared to when represented as magnitude and direction.

2.4.1 Motion Estimation

As a proxy for the actual motion extraction used by the brain, we use the highly accurate algorithm developed by Brox et al [23] in estimating the optical flow for our stimuli. Ideally, we would like to know the actual 3D motion of every object, then project it to a 2D representation. However, we are only given the images presented in two (or maybe more) frames in a video, restricting us to the domain of pixels instead of objects.

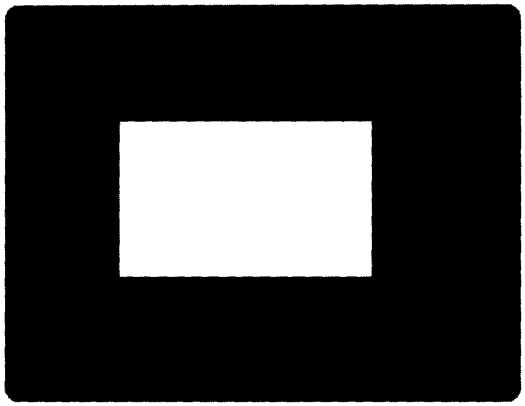

Figure 2.8: A white box moving to the right in a black background offers local motion estimates along the left and right borders but nowhere else

Color and Gradient Constancy Assumption

In order to track pixels from frame to frame, we start with the assumption that pixels generally do not change their color from frame to frame. This leads to a natural goal of finding the displacement of that pixel into the next frame, so as to minimize the difference in color.

However, there are some cases when a change in illumination (i.e. due to the sun suddenly being hidden by clouds, or vice versa) makes this assumption false. The gradient of the pixel however, does not change under an illumination change because they are a measure of boundaries of objects or regions. Thus, in addition to the color constancy assumption, we assume that the gradient of the pixel is constant across frames.

These two constraints together form the evidence of motion given the data, providing a local estimate of motion. However, that is insufficient to solve the problem of finding optical flow that is consistent with perception, in regions of low contrast or noisy regions. For example, consider a large white box on a black background as shown in Figure 2.8. If this white box were to move to the right by one pixel, the local estimates of motion would indicate that there is strong evidence that there is motion to the right at the left and right edges of the box, and strong evidence that there is no motion along the top and bottom edges of the box. However, there is no discriminating what type of motion if any, lies at the center of the box using the local evidence. But we know that the top, bottom, and center of the box has moved to the right along with the left and right sides. Therefore, there must be a mechanism for propagating the information about motion to ambiguous regions in the image.

Piecewise Smoothness Assumption

The piecewise-smoothness assumption, makes the claim that the optical flow field smoothly varies within a region of space. This allows to infer the motion of a pixel in locations whose motions are locally ambiguous, but when taken into context of the larger picture, are determined. It allows for discontinuities to account for the fact that objects can move independently of each other allowing the optical flow to be discontinuous at the boundary of objects.

This assumption additionally helps minimize the impact of noisy data, by weighing the evidence from the surrounding pixels. This results in more robust estimates of optical flow under noisy inputs.

Multi-scale estimation

One problem with implementing the optical flow from these assumptions, is that most methods make the simplifying approximation of the spatio-temporal stimuli as a first order Taylor series approximation, and is thus only able to capture linear relationships which tend to hold for only small motions (less than a pixel). Unfortunately, optical flow is often non-linear and involve motion that is far more than a single pixel. To overcome this difficulty, we find the optical flow at a very low resolution to obtain the large motions, and interpolate this motion to a higher resolution of the image and warp the current video frame toward the next frame and compute the optical flow after correcting for the larger motion. This is repeated at successively higher resolutions until the original resolution case is solved to get our final optical flow output.

Implementing such assumptions to obtain optical flow involves finding the horizontal and vertical displacements that maximize the objective function (i.e. what motion best satisfies the previously mentioned assumptions). We refer the interested reader to [23] for the details on computing the optical flow. For our purposes, we note that we are interested in the theoretical limit, assuming we can get the best optical flow estimate possible, and then proceeding to compute motion saliency. We consider this a module which we can replace with a better method for estimating optical flow that is more consistent with human perception and cortical representation when available.

Figure 2.9 shows the optical flow estimate from two frames of video.

The effect of 'flicker' can also be classified as a manifestation of motion saliency in an optical flow representation. Essentially, the displacements of the pixels are infinite, causing high saliency, and attract a lot of attention.

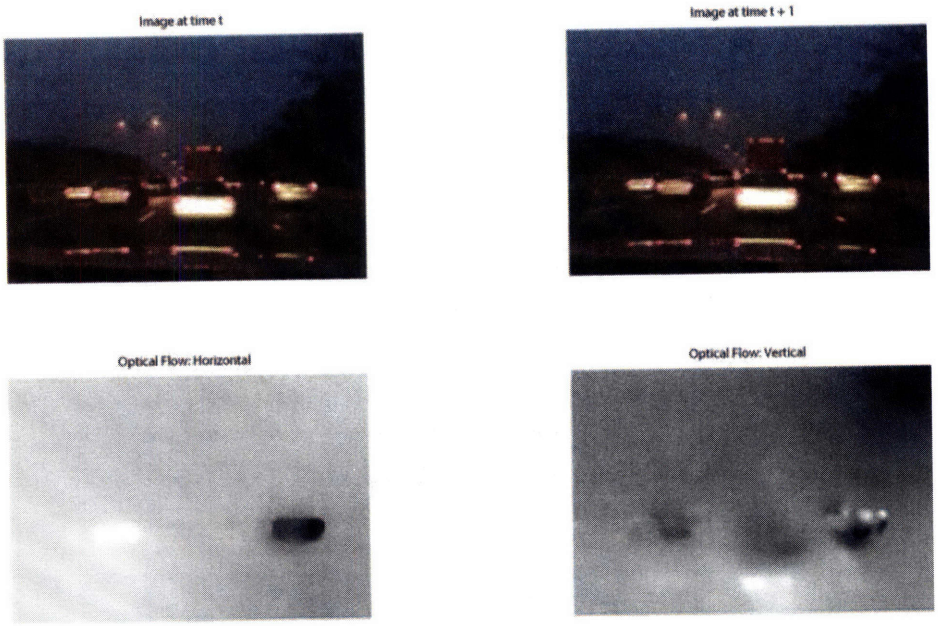

Figure 2.9: Extracting Optical Flow from two video frames

2.5 Computing Saliency

The entire process of computing saliency can then be summarized as per Figure 2.10. The input stimuli is a discrete spatio-temporal volume with each spatio-temporal coordinate specifying a color. For each slice in that volume at time t, we extract color, orientation, and contrast features, and in addition, when we take two successive slices, t and $t+1$ we extract the optical flow specifying the motion between the two time-slices.

We then proceed to compute the saliency for each feature map, by calculating the Mahalanobis distance between the mean feature value of a spatial location, compared to the neighboring distribution of feature values. We estimate the local mean feature, x, with a spatially weighted average according to a two dimensional normal distribution. Similarly, for the two other estimated parameters which involve the surround, we weigh each neighboring location's contribution according to a 'donut' like shape. This shape is produced by taking a two dimensional normal distribution with a larger standard deviation (i.e. larger pooling area) than that of the local mean computation, and subtracting the weights used to compute the local mean and then renormalized to be a valid discrete probability distribution. See Figure 2.11 for an image showing what these spatial weights look like.

The left image in figure 2.12 shows how the motion saliency of the point in the center of the red circle is computed according to the Statistical Saliency Model. Each pixel within the red circle around the target location is marked by

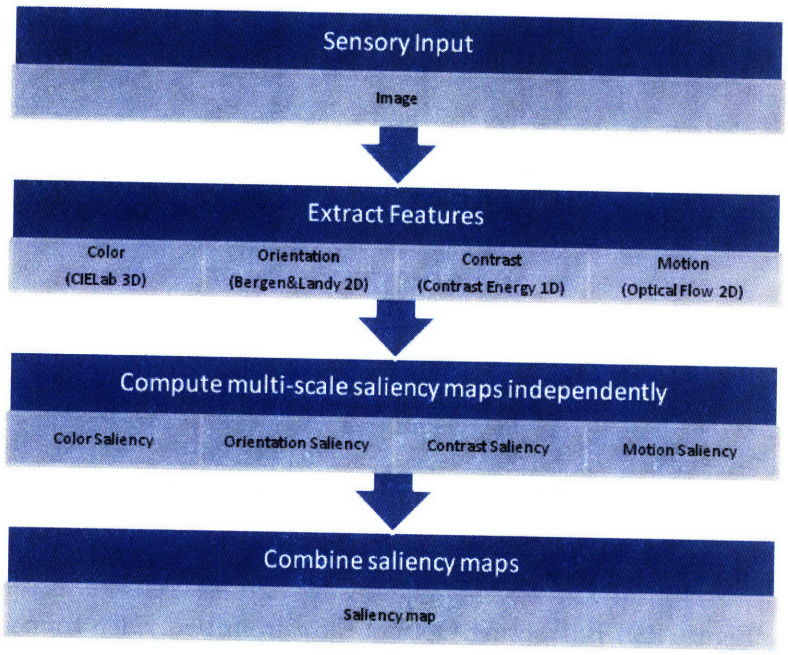

Figure 2.10: Saliency Process

a red dot in optical flow space in the center image. Similarly, each pixels in the blue neighboring area are marked by a blue dot in optical flow space in the center image. We then compute the mean of the target distribution (the red dots), x, the mean of the distractor distribution (the blue dots), μ, and the covariance matrix computed as though the distractor distribution were a two dimensional Gaussian distribution, Σ, as shown in the right image. The green and black ellipses around the mean show the Mahalanobis distance, $s = \sqrt{(x-\mu)^T \Sigma (x-\mu)}$, at the $2D$ equivalent of 1 standard deviation and the target's Mahalanobis distance from the surrounding pixels' distribution. The optical flow feature dimensions for the entire two-frame image sequence are shown in Figure 2.9.

This computation is done for every spatial location in the feature map at multiple scales in the Gaussian pyramid. Then we let the saliency contribution for each spatial location be the sum over all the scales (i.e. by resizing all the saliency maps computed back up to the original scale via linear interpolation then adding them together). In order to combine the saliency information across the multiple saliency maps for every feature, we assign equal weight to each saliency map, s_i, to obtain our master saliency map, $s_m = \sqrt{\sum s_i^2}$. We caution that we have not tested this combination strategy in detail. We based this decision on some simple test cases – there weren't a consistent set of small variations from

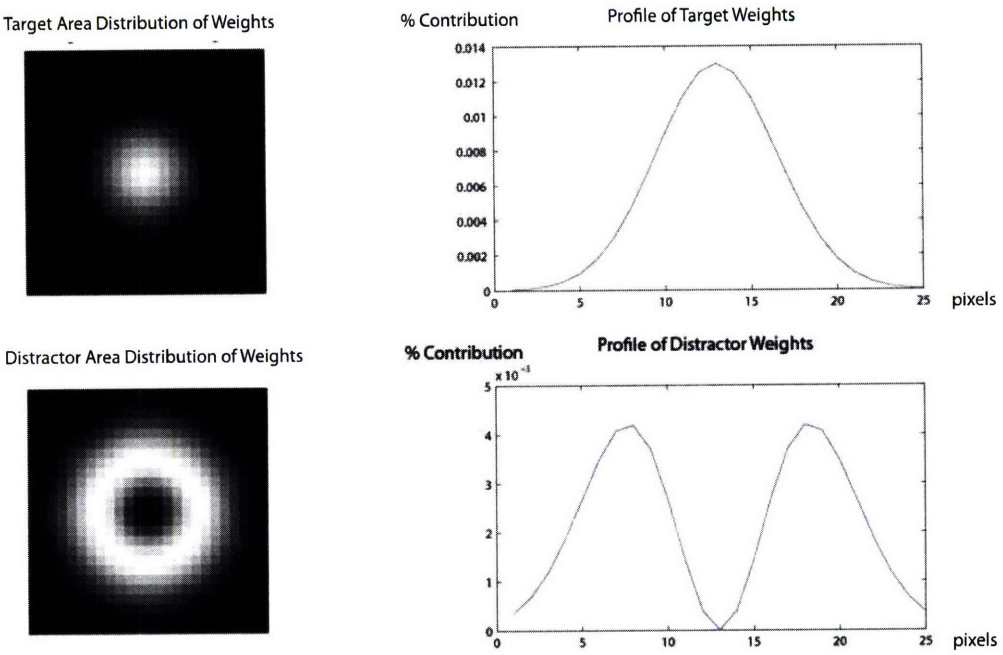

Figure 2.11: Illustration of weights of contribution from each spatial location neighboring the spatial location for which saliency is being computed

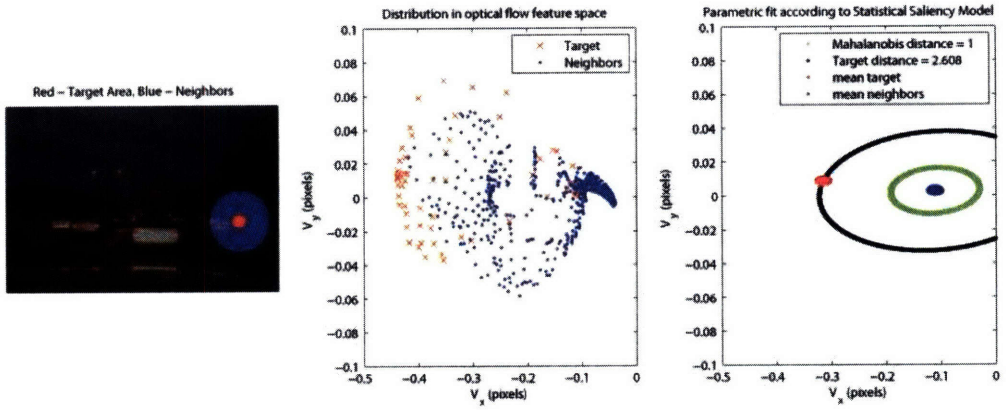

Figure 2.12: Computing the saliency of a given location

equal weights on each feature map that performed significantly better than the equal weight strategy.

This chapter discussed in detail the Statistical Saliency Model and its implementation. While the model is based on some psychophysical evidence, we feel strongly that these theories need to be tested extensively in more realistic environments, such as in a driving context as will be discussed in the next chapter.

Chapter 3

Saliency in a Driving Environment

3.1 Driving

Because we are interested in understanding saliency in relation to everyday life, we take the approach of using more realistic stimuli and contexts. Previously, most stimuli used in trying to test the efficacy of saliency models tended to be very well controlled, and very simple visual stimuli in a lab environment, like Figures 1.1 and 1.2. While the models put forth in explaining a variety of visual search tasks have been successful, they have been limited to simple stimuli which are rarely encountered in real life. There is little work showing whether what we learned in the laboratory can carry over to what we see in everyday life.

Both as an application and as an opportunity to better understand saliency and visual attention in a real-world environment, we look at the visual environment within the task of driving. In considering the context of driving, the visual field we are interested in takes on some characteristics that are not present in an arbitrary visual environment.

Viewed as a tool to aid the task of driving, visual saliency could potentially be used as part of an intelligent vehicle in modeling the perception of the driver to help the intelligent vehicle assist the driver in making decisions quickly and safely. By identifying the salient regions in the visual environment, we are effectively marking the most likely locations a driver is looking at. In combination with a threat detection system, we can then identify threats that drivers are unlikely to notice.

3.2 Thought Experiment: Information Processing Ability in Drivers

Driving is a very complicated behavior. The typical driving scenario involves driving on a road with a few lanes, oncoming cars on the other side of the road, cars ahead on the same lane, and cars going in the same direction in a parallel lane. A driver might have to stop at intersections, merge into traffic, interpret signs and lights at the sides of the road, or painted onto roads, etc.

A naïve characterization of the task of driving might require, while keeping in mind directions for where to go and controlling the pedals, steering wheel, and gear shift, an almost instantaneous interpretation of the light field to extract where objects are, what these things are, whether, where and how fast these objects are moving, self motion, whether any collisions are going to occur, and more. Further complicating the task, these objects may be traffic signs informing the driver of road conditions to expect or regulations to obey. We have clearly not managed to carry out this task perfectly, as evidenced by the number of accidents, but on the other hand one typically expects not to get into an accident commuting to or from work with great likelihood. Accidents are the exception instead of the norm. This begs the question, how do drivers manage to perform such a complicated task with relatively few critical mistakes that cause accidents?

As a thought experiment, one might imagine the ideal driver, who has infinite information processing capacity and speed, whom we shall call Driver Infinity. This driver makes the optimal decision with respect to controlling the steering, acceleration, brakes, and gears of the car in order to carry out the driving task, minimizing the chance of an accident, and following the rules of the road, given the information about the visual world, provided to this driver. Driver Infinity would then perform better as more information about the driving environment is added. We first simplify this situation by considering only the visual information that can be gathered by looking out the front windshield of a car. Then the maximum amount of information that can be obtained is the highest resolution possible image of the visual environment in front of the car sampled at the highest possible framerate.

Driver Infinity might then know every single detail about the environment – such as the location of rocks and potholes on the ground, and subtly maneuver the vehicle in order to minimize the likelihood of a tire puncture or other mechanical failures, in addition to tracking all the other vehicles on the scene, and where the drivers of these vehicles are looking at in order to model what they may be aware of, and estimate the likely behaviors of these vehicles.

We then might be curious, how much of the visual input is really necessary? If we remove parts of the visual stimuli, can Driver Infinity still perform well as a driver? How much can we remove before this driver becomes prone to accidents,

or is unable to drive? We might imagine an inverse relationship between driving performance and information provided. Let us suppose that driving performance can be somehow quantified into a number, and the instantaneous driving performance at time t is D_t which varies from $D_t = 0$ being the worst performance, up to $D_t = 100$ which is the best possible driving performance.

Not all parts of the visual input are equal – parts of the visual scene are more important than others, for example the visual input indicating a mountain range many miles away in the distance would be less important to the driving task than the visual input indicating the car ahead is making an emergency stop. To quantify this notion, we now define a one dimensional visual information variable, $V_t \in [0, 100]$ where $V_t = 0$ indicates zero visual information which also corresponds to the worst possible input for Driver Infinity and $V_t = 100$ indicates the maximum amount of visual information or the best possible input for Driver Infinity, at time t. We then calibrate V_t such that if Driver Infinity were given as input V_t, Driver infinity would have a driving performance $D_t = V_t$.

In this paradigm we have two variables we can vary – the information processing capacity of the driver, and the information from the visual environment fed to the driver. Let us consider some driver models: Driver Ten, and Driver Twenty, who are respectively able to process up to ten, and twenty units of visual information per second.

If a constant stream of 10 units of visual information per second were sent to Driver Ten and Driver Twenty, they would both perform at the best possible driver performance, $D_t = 10$, given the information. Driver Twenty would have no problems reaching the best possible driver performance, $D_t = 20$ when given an input of 20 units of visual information per second, but Driver Ten would take two seconds to process every second of input and thus be unable to keep up, and even if Driver Ten were able to instantaneously detect the best 10 units of information to process every second, Driver Ten would only be able to perform at $D_t = 10$. However, there is a cost in having to consider more information than one is able to process, making $D_t = 10$ the theoretical maximum performance of Driver Ten when given 20 units of information per second, but whose performance is less than the maximum due to having too much information to handle.

There have been many attentional studies showing that the serial processing is indeed limited, especially concerning tasks that involve effortful thinking, while the essentially parallel tasks of attention and intuition are unfettered [9, 18]. The complex task of driving possibly involves both the fast and parallel (i.e. interpreting motion, recognizing objects, etc) as well as the slow and serial computations (i.e. reading signs, searching for non-salient threats, etc).

As mentioned in Chapter 1, the human brain can selectively attend to visual input. This is consistent with a view of an information processing brain that has a limited information processing capacity. By selecting the information that is

most useful the brain reduces the amount of workload so it may attempt to keep a decent performance under the flow of constant information, much like Driver Ten. Because it is not possible to know ahead of time what parts of the visual environment is most useful, the visual system must first accept every part of the visual input, but later decide what parts are most useful for the task at hand. We might then speculate that the bottom-up attention process which selects subsets of the visual scene corresponds makes use of the fast and parallel nature of low-level visual processing to pick out a small set of useful information to pass down the chain of processes to the slower and serial reasoning systems in the brain.

And indeed, this is what we have found in human vision. The Feature Integration Theory as mentioned in Chapter 1 posits a serial attentional binding step to integrate the various information extracted via parallel processing.

There has been much research on how attentional systems in humans work that shed light on why it is not necessary for humans to be conscious about many of the things that are going on in a driving environment, but instead need only pay attention to a subset of important aspects of the driving visual environment. However, what is most salient in the visual scene may not correspond to the most important part of the visual scene according to our measure of visual information. For example, a well camouflaged person standing in the middle of a highway is not salient, but is very important towards the task of driving. While this spectrum of saliency and importance spans a scope beyond this work, this view of a driver as an entity that is fed visual information, will recognize the separation of the two concepts, and acknowledge that there is an overlap between the two. This is partially explored in the experiments we conducted.

3.3 Visual Environment while Driving

Compared to an arbitrary visual stimuli, the driving task has a distinct and characteristic set of features. Roads, signs, cars, ego-motion, traffic lights, seen at a variety of speeds, are some of the features of this visual stimuli.

A driving environment includes the constant motion of the car mostly moving forwards, occasionally turning, and rarely reversing. See Figure 3.1 for an example of the motions involved in a typical driving environment. The camera is moving forward causing a looming motion, and in addition, there are independent moving cars in front and on parallel sides of the road. In this scene, we note that the car ahead is not moving relative to the motion of the camera, while the two cars flanking the car ahead are moving at a significant speed relative to the camera.

We note that in this scene, if we had simply taken the difference between successive frames, the resulting information only tells us there was some slight change in the flanking cars. The magnitude of motion gives us information about

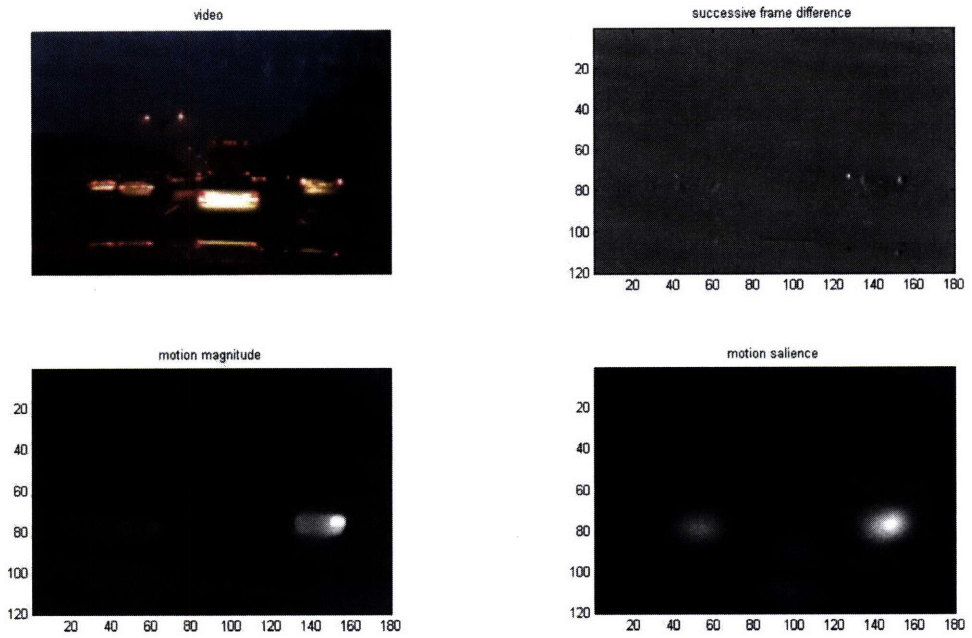

Figure 3.1: Motion in a driving environment

the motion of the scene, but if we were to use the magnitude of motion solely as a determination of what attracts attention, we would not predict that the car ahead which is moving differently from the cars around it should be paid attention to. When we look at the saliency map computed from the optical flow field, we note that the three closest cars are salient, assigning more saliency to the car that corresponded to the greatest motion magnitude.

This chapter has discussed the motivation for why a vision scientist might want to apply the models and theories developed in a lab into a driving context, as well as how saliency can be used as a strong tool in modeling driver behaviors. In the next chapter, we test our model in some behavioral experiments.

Chapter 4

Experiments

We explore the idea that saliency affects driving performance by varying the saliency in a simulated driving task, and measuring driver performance. If performance improves with visual saliency, while the visual information is somewhat constant, then we show that a process akin to visual saliency is at least part of a mechanism that guides a human driver's attention and has an impact on performance.

4.1 Drive Simulator Experiment

We put our theory to the test in seeing whether a bottom-up saliency model is predictive of observed behaviors within a driving task. We make the hypothesis that driver performance, as measured by how quickly a driver is able to detect a pedestrian, can be predicted by how salient that pedestrian is.

4.1.1 STISIM Drive Simulator

We had subjects steer a simulated car using the STISIM Drive Simulator [7] whose speed was set to a constant $40mph$ on a relatively straight road with no intersections or stops. We added some minor road curvature in order to increase the cognitive load. The road was divided into $1000ft$ segments and within each segment contained some combination of background clutter variables (See table 4.1) in order to change the saliency of the target pedestrian. The number of distracting objects in visual search tasks is known to have a significant influence on search time. Subjects were given the task of detecting a pedestrian about to cross the road, by pressing a button on the steering wheel. The road had two lanes, separated by a double yellow line.

The STISIM drive simulator consists of a car shell with pedals and steering wheel connected to the STISIM drive simulator. The simulator was run on a Dell XPS 600 computer with a 3GHz Pentium D processor, and 1GB Ram. The screen is $152cm$ away from the subject, and is $94cm$ by $58cm$, which translates to about a visual angle of about 34 deg by 20 deg.

Pedestrians	Cars	Trees	Buildings
{0,5,10}	{0,3}	{None, Few, Many}	{None, Few, Many}

Table 4.1: Scene Clutter Variables

Target Pedestrian
{Left, Right, Absent}

Table 4.2: Target Variable

We discuss the target variable, and the nominal clutter variables – pedestrians, cars, buildings, and trees in detail below. In each stretch of $1000ft$, a subject drove through some combination of number of distracting pedestrians, oncoming cars, buildings and background trees, and tried to detect a target pedestrian trying to cross the road (see Figure 4.2). It took between 15 to 20 seconds to complete each trial.

We had a total of 18 male and 2 female subjects whose ages were between 25 and 70 years old. Each subject drove for two sessions of 15 minutes with a small break between the two sessions. There were approximately 120 trials of $1000ft$ comprising some selection of variables, per subject. The simulation that each subject participates in was randomly generated so that the 120 trials spanned a selection of the variables in a random order. Each trial was also randomized to vary the visual appearance of the buildings, trees, pedestrians, cars and road curvature.

Target Pedestrian

The target pedestrian was either absent, or trying to cross the road from the left or right side of the road. If the target were present, the target pedestrian would have started walking from the sidewalk towards the road and stopped at the edge of the road. This event was triggered when the subject's car was $450ft$ from the target. The target pedestrian then walked at a speed such that if the pedestrian were to carry on walking onto the road, the pedestrian would collide with the subject's car.

Subjects were asked to mark the pedestrian target by pressing a "left" or "right" button located on the steering wheel in the simulator, in order to indicate whether the pedestrian were crossing from the left or right side of the road

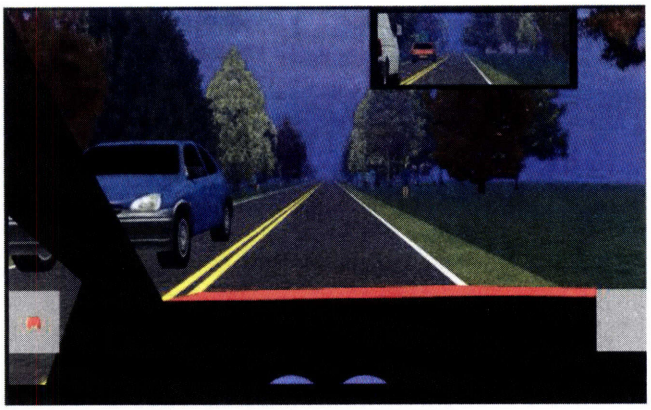

Figure 4.1: STISIM Experiment Screenshot 1

Figure 4.2: STISIM Experiment Screenshot 2

respectively. The drivers were asked to press the correct button as soon as they noticed a pedestrian moving towards the road or if they noticed a pedestrian standing at the side of the road as though about to cross. We recorded the response time and incorrect responses.

The target pedestrian had the same appearance as the distractor pedestrians as discussed in more detail below.

Pedestrians

Within each trial, there would be 0, 5, or 10 distracting pedestrians who simply walked along the sidewalks either to or from the horizon on either side of the road. They all walked at similar speeds to each other, approximately $5ft$ from the road. There were some number of model pedestrians which the STISIM drive simulator can render. Some examples are shown in Figure 4.3.

Figure 4.3: Some pedestrians in STISIM

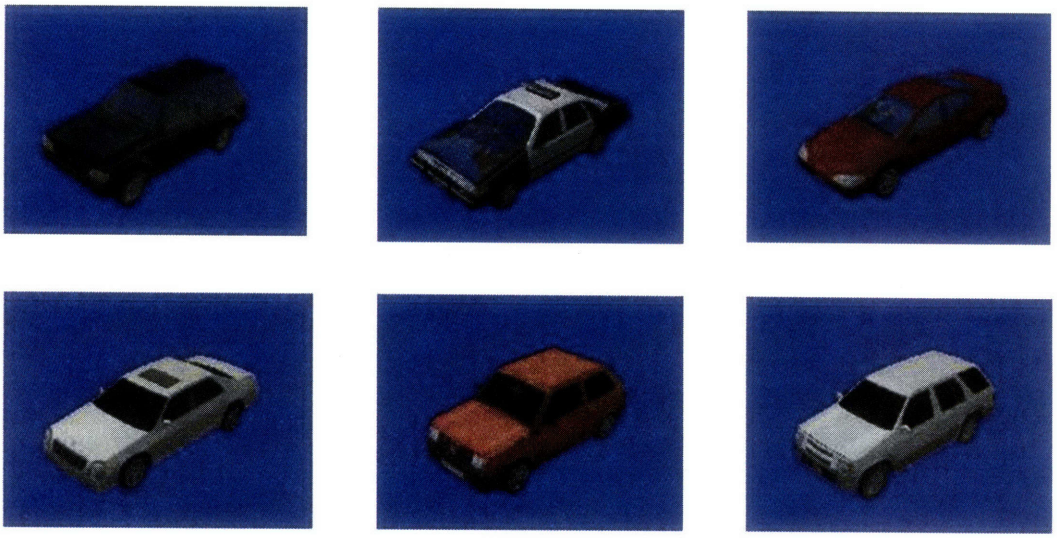

Figure 4.4: Some cars in STISIM

Cars

On the oncoming side of the street, during a trial, there may be 0, or 3 cars traveling at around $40mph$.

Buildings

Buildings on either side of the street constituted a static clutter variable. In each $1000ft$ stretch in a trial. There would be one of three conditions encountered within each trial: no buildings, some sparsely located buildings, or densely packed buildings on both sides of the road. In Figure 4.5 we show some sample buildings that the STISIM Drive simulator had rendered.

Trees

Similarly, within each trial, there were trees (see Figure 4.6) on either side of the road. There were also 3 possible choices within each trial – no trees, a few trees,

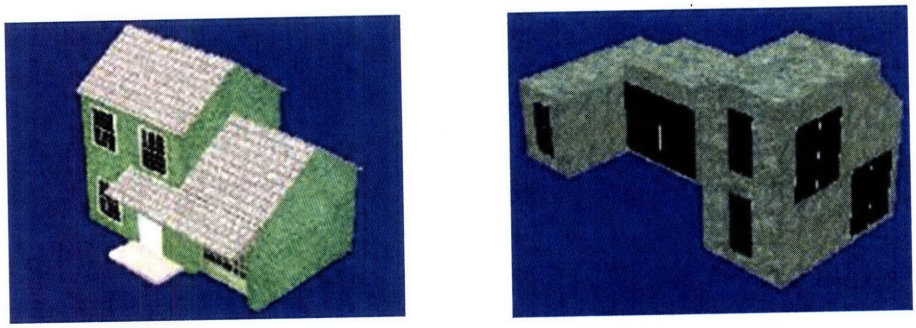

Figure 4.5: Buildings in STISIM

Figure 4.6: Some types of trees in STISIM

or densely packed trees on both sides of the road.

4.1.2 Results

We first look at the correlation between reaction time to detect the pedestrian versus the nominal clutter variables, then correlate the reaction time to visual saliency. We separate the cases of when the pedestrian is crossing from the left and when the pedestrian is crossing from the right side of the road because the car is driving on the right lane, which makes the visual appearance of both cases asymmetric. Because some subjects are simply faster than others, we also try to normalize the reaction time for each subject by subtracting the subject's mean reaction time and dividing by the standard deviation for that subject to obtain the z-score for each reaction time. In the following parts we refer to this as the normalized reaction time.

Nominal correlations

As mentioned in Chapter 1 in conjunction visual search tasks (and many other complex visual search tasks) the number of items in a visual stimuli have a very strong influence on the reaction time. We find some significant but small correlations between the nominal measure of set size and normalized reaction time.

Table 4.3 shows the Pearson Correlation coefficients between nominal clutter and normalized reaction time.

Nominal clutter variable	Right: r	Left: r
Number of Buildings	.06395	.03137
Number of Cars	.05210	.14268
Number of Pedestrians	.07868	.00496
Number of Trees	.14009	.04906

Table 4.3: Correlation coefficient between nominal clutter and normalized reaction time

These correlations are small because the nominal clutter can be thought of as a pseudo-measure of set size, but it does not consider the visual scene, and the 'effective' set size or the area in which a subject is searching in to find the pedestrian is not every single object on the scene. As can be seen in the sample snapshots, we notice that the number of objects is not very useful because of their high variability in the scene.

Correlation of reaction time with visual saliency

We computed static and motion saliency, then obtained the combined saliency map. In addition to the saliency maps, we also have for each frame in a video, an approximate bounding box of the location of the pedestrian. This bounding box was estimated via a semi-supervised labeling of the pedestrian location. To speed up computations, we ran our algorithm on a lower resolution of the video.

We picked some sample images and show the corresponding saliency maps. Figures 4.7, 4.8, 4.9, and 4.10 respectively show a snapshot taken at random from the video corpus, in a trial with many trees, a low clutter trial, and a high clutter trial. We notice that the road markings (i.e. yellow double line, etc) are always salient, and that motion saliency tends to pick out the pedestrian even when most of the static features are having trouble. Color saliency is most admissive, and allows the boundary of the vehicle and the road be very salient. Some of the more mundane and less informative areas which are salient according to this bottom up model, we expect are inhibited by top-down processes after learning the task of driving. In the highly cluttered scene, the saliency of everything else seem to overwhelm that of the target. This is an example where the saliency model predicts a slow reaction time in subjects.

For this application, in order to account for the clutter in the background, we define the saliency signal as the ratio of the mean saliency of the target to the mean saliency of the entire scene because presumably eye movements occur not just as a function of local saliency, but as a function of local saliency compared to everything else. Thus, we try to predict the time at which the subject detects the target by finding the first time when the saliency signal exceeds some threshold

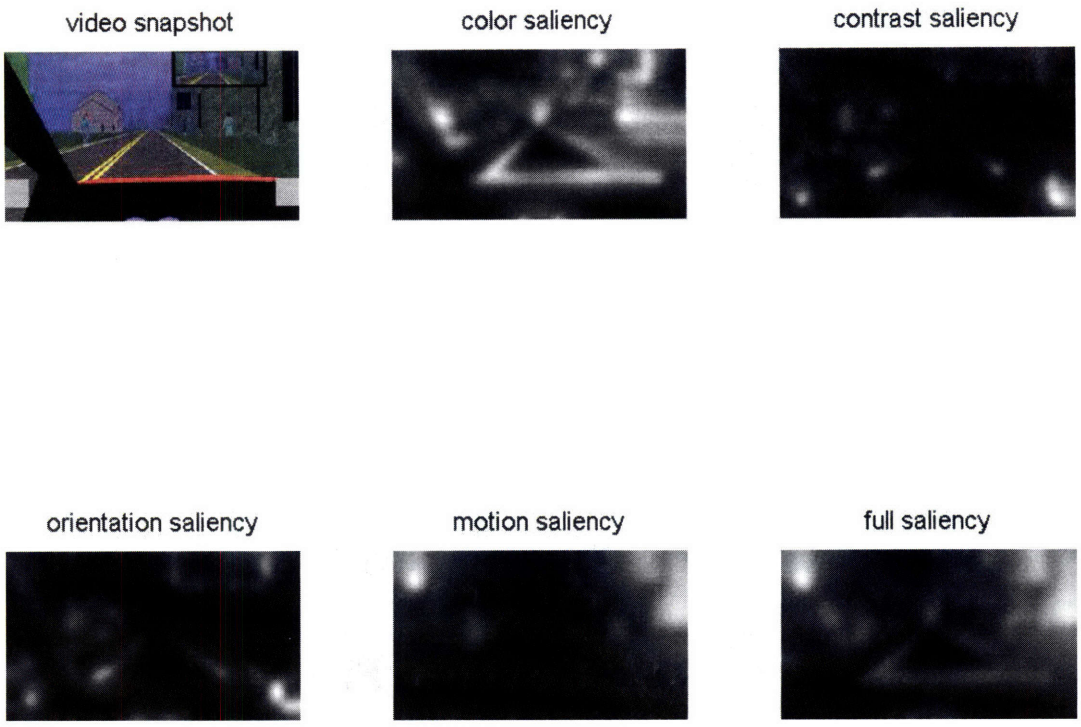

Figure 4.7: A randomly selected snapshot from the video with its corresponding saliency maps

ratio. We hypothesize that when something is very salient it would attract the attention of the subject as well as enable the subject to discriminate whether or not it is the target. We then look at how well this predicts the observed reaction time by looking at the Pearson Correlation Coefficient between the predicted reaction time and the normalized reaction time. Figures 4.11 and 4.12 show the plots for correlation between reaction time and predicted reaction time according to the different saliency maps. The reaction times were quantized to fixed sized bins, and each point corresponds to the mid value of the reaction time bin, and the mean value of the predicted time of values in the bin. The error bars show the standard deviation within the bin. We note that for the case of pedestrians crossing from the left, that there seems to be a nonlinearity involved, and so the Pearson Correlation Coefficient computed are underestimates of the correlation relationship.

As seen in the plots, we find that our predicted reaction time based on saliency does indeed correlate with normalized reaction time.

4.2 Eye Movements Analysis

In addition to recording the reaction time, for a small subset of the subjects, we also recorded eye-movements. We used a faceLAB [15] eye-tracking setup with a

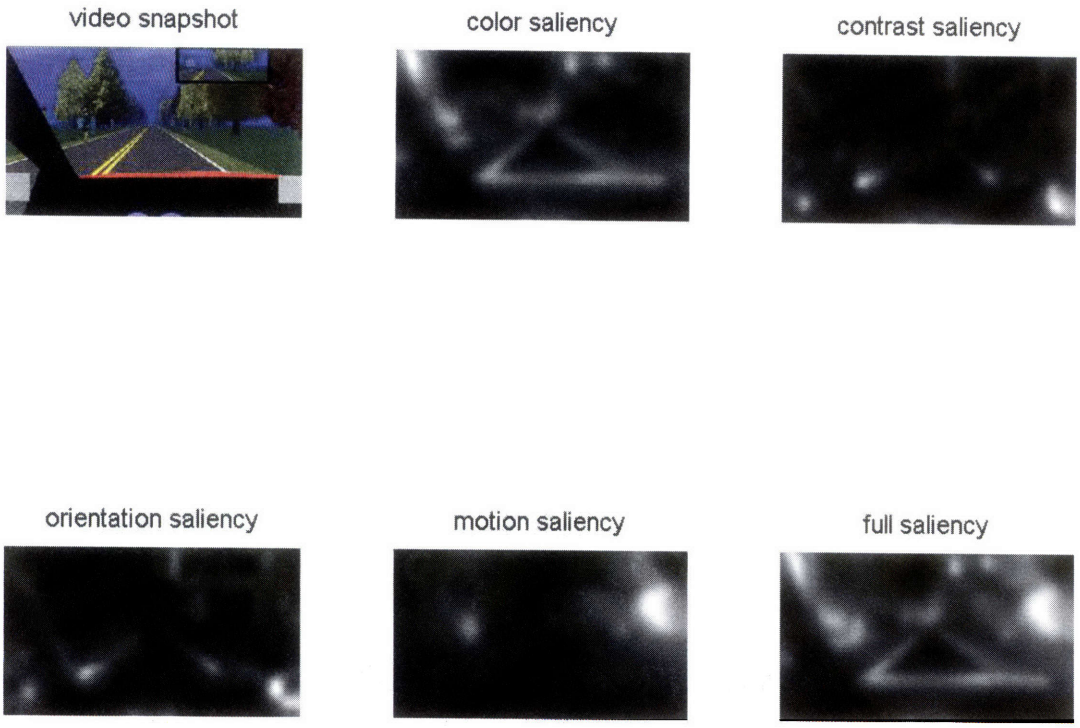

Figure 4.8: A snapshot from a trial with many trees

pair of cameras. To synchronize the simulator with the eye-tracking machine, we used a crossover patch ethernet cable with the Network Time Protocol client on the faceLAB computer, and the STISIM Drive simulator running the server.

Instead of trying to predict reaction time, we can also look at how closely saliency can predict eye-movements. However we do need to keep in mind the caveat that attention applied to a spatial location does not necessarily correspond to someone fixating on that spatial location.

Eye movements were recorded for 6 subjects. We try to observe how much of the eye-movements we can explain by essentially considering how much our saliency maps intersect with the eye-movements. We can essentially try to evaluate this by letting the saliency map be a probability distribution of estimated eye gaze intersection, and computing the probability that the observed eye-gazes were generated by the probability density map we had estimated from the saliency information. To evaluate this, we also compute the probability that it was generated from a uniform distribution over the video (i.e. every spatial location is equally likely).

Let the observed eye-tracking data, which comprises a x and y coordinate for every frame in the video, t, be denoted by $(x, y)_t$. Then the probability of observing $(x, y)_t$ is denoted by $P_{s_t}[(x, y)_t]$ where s is one of the candidate probability distributions computed at time t.

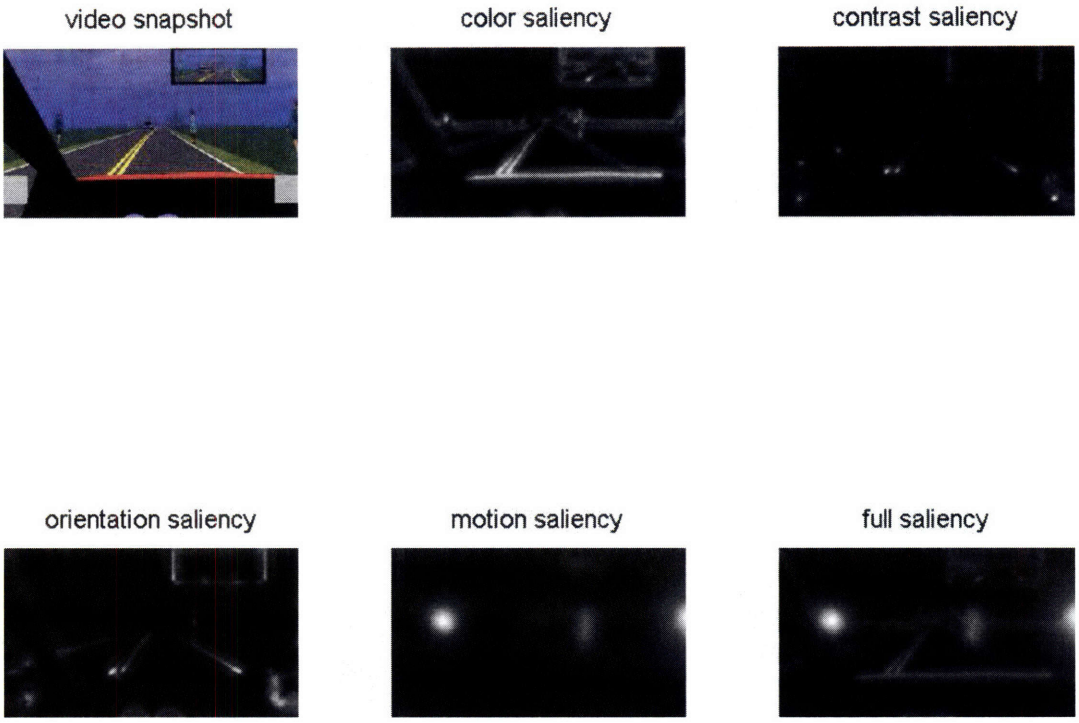

Figure 4.9: A snapshot from a trial with little clutter

We define the uniform distribution as

$$P_u[(x,y)_t] = \frac{1}{wh}$$

where w and h are respectively the width and height of the video.

We assume that the saliency map is proportional to the probability distribution of that saliency map. And in order to account for eye-tracking estimation noise, we compute the probability of observing the eye tracking location by integrating over a uniformly weighted disc centered at $(x,y)_t$ with a radius of 1^o (i.e. place a disc of 1^o centered at $(x,y)_t$). Then the probability of observing $(x,y)_t$ is just the saliency at that location divided by the sum over the entire saliency map at that time frame. We also look at how this might vary depending on an exponentiation, γ applied to the saliency map before the normalization to a probability distribution.

In order to incorporate a simple measure of top-down influence, we modulate the saliency map with a preference to look at the center of the image. We combine contributions from a Gaussian distribution and the saliency map, with a weighting parameter $0 \leq \alpha \leq 1$. In our experiments, we find that $\alpha = \frac{1}{2}$ is a reasonable parameter choice.

$$P_{sal_t}[(x,y)_t] = \frac{1}{Z}\sum_{(a,b)} U(a,b;(x,y);1^o)\times(\alpha\times sal_t[(a,b)_t]^\gamma+(1-\alpha)\times N(a,b;(c_x,c_y);\sigma_c))$$

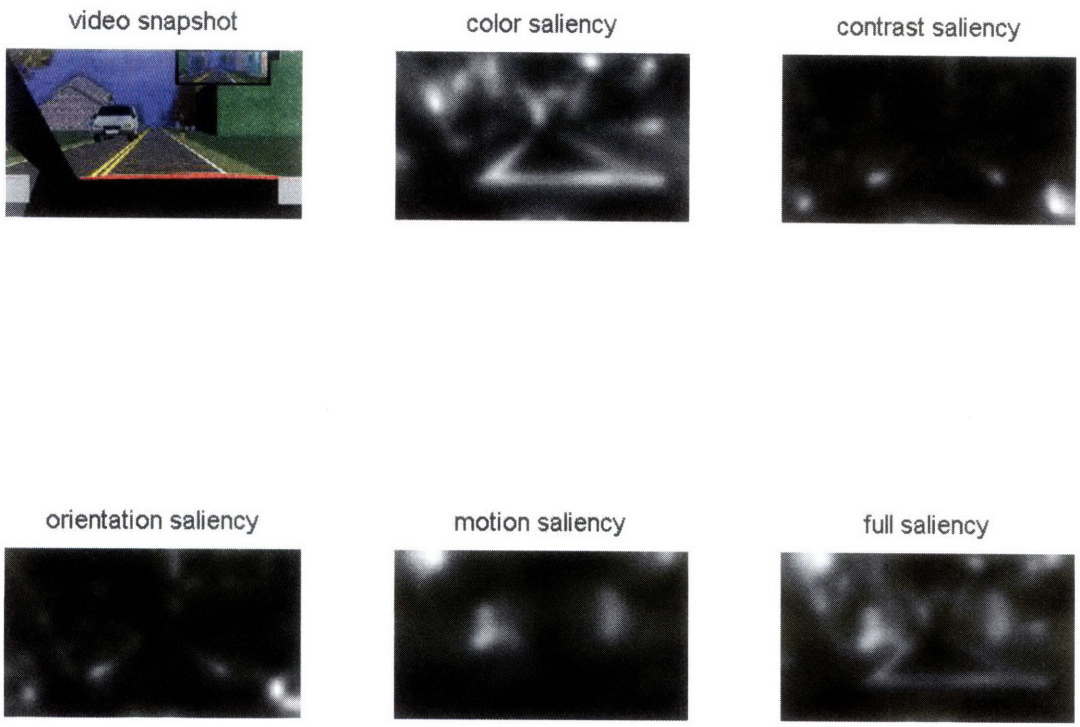

Figure 4.10: A snapshot from a trial with a lot of clutter

Where $Z = \sum_{(u,v)_t} sal_t[(u,v)_t]^\gamma$ is the normalization constant, $U(x,y;\mu;r)$ is a uniformly weighted disc centered at (x,y) with radius r, and $N(x,y;\mu;\sigma)$ is a bivariate normal distribution centered at μ with both dimensions independent and equal standard deviation on both spatial dimensions, σ, and (c_x, c_y) is the center of the image.

For evaluation, we compute the probability that the eye-gaze observations were generated by a given distribution P_s. For simplicity in computations, we assume that there are no correlations across time, so that the probability of observing $(x_1,y_1)_t$ and $(x_2,y_2)_{t+1}$, can be simplified as

$$P_s[(x_1,y_1)_t \wedge (x_2,y_2)_{t+1}] = P_s[(x_1,y_1)_t]P_s[(x_2,y_2)_{t+1}]$$

.

Let $D = \{(x_1,y_1)_1, (x_2,y_2)_2, ..., (x_W,y_H)_T\}$ denote entire sequence of observations. We are interested in finding the probability that this sequence of observations were generated by a given distribution, which we call the likelihood of the observations given the probability distribution.

$$\begin{aligned} P_s[D] &= P_s[(x_1,y_1)_1 \wedge (x_2,y_2)_2 \wedge ... \wedge (x_T,y_T)_T] \\ &= P_s[(x_1,y_1)_1]P_s[(x_2,y_2)_2]...P_s[(x_T,y_T)_T] \\ &= \prod_{i=1..T} P_s[(x_i,y_i)_i] \end{aligned}$$

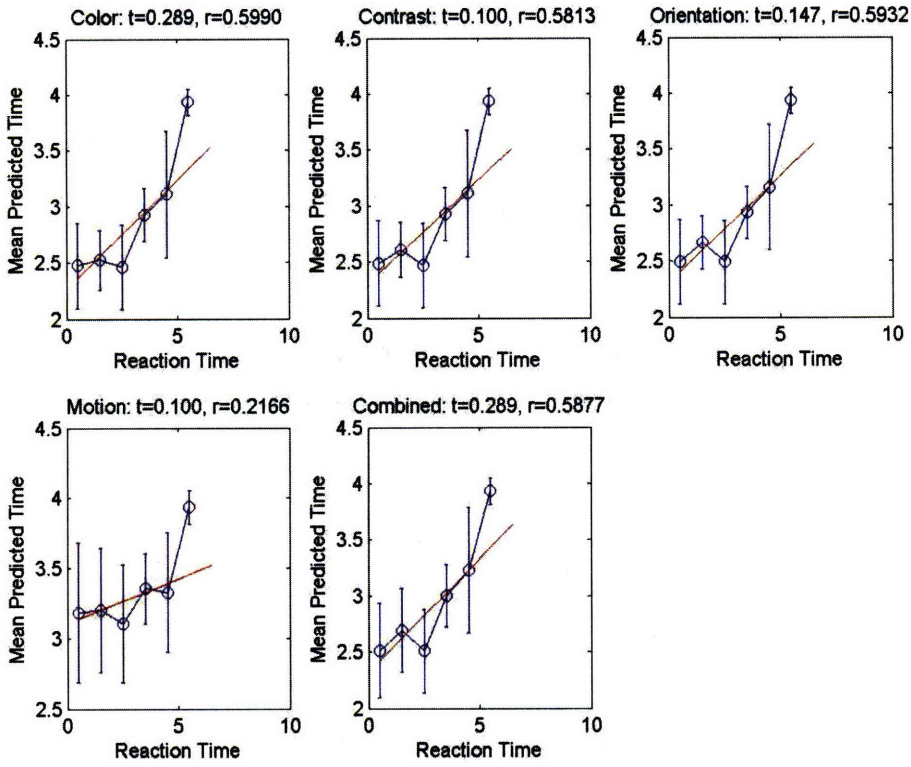

Figure 4.11: Reaction Time correlation with Predicted Time for a pedestrian crossing from the left. t is the threshold used, and r is the Pearson Correlation Coefficient. Error bars are standard deviation for that bin.

Then, in order to allow numerical stability in computations as these numbers will become very close to zero rapidly as we are multiplying many numbers between 0 and 1, we compute the log likelihood, $\log P_s[D] = \log \prod_{i=1..T} P_s[(x_i, y_i)_i]$ simplified as,

$$\log P_s[D] = \sum_{i=1..T} \log P_s[(x_i, y_i)_i]$$

Table 4.4 shows the results on the videos. We found that 2^o provided a local maxima with respect to maximizing the likelihood on our data. We also found that the choice of γ between a range of values between .125 and 2 made little difference (within 5 percent of the values) in the likelihood of the data.

These results indicate that our measure of saliency has a predictive power a few orders of magnitudes better than random chance, about an order of magnitude better than a model simply predicting we look at the road ahead of us all the time.

We find that saliency does in fact predict better than chance and some simple

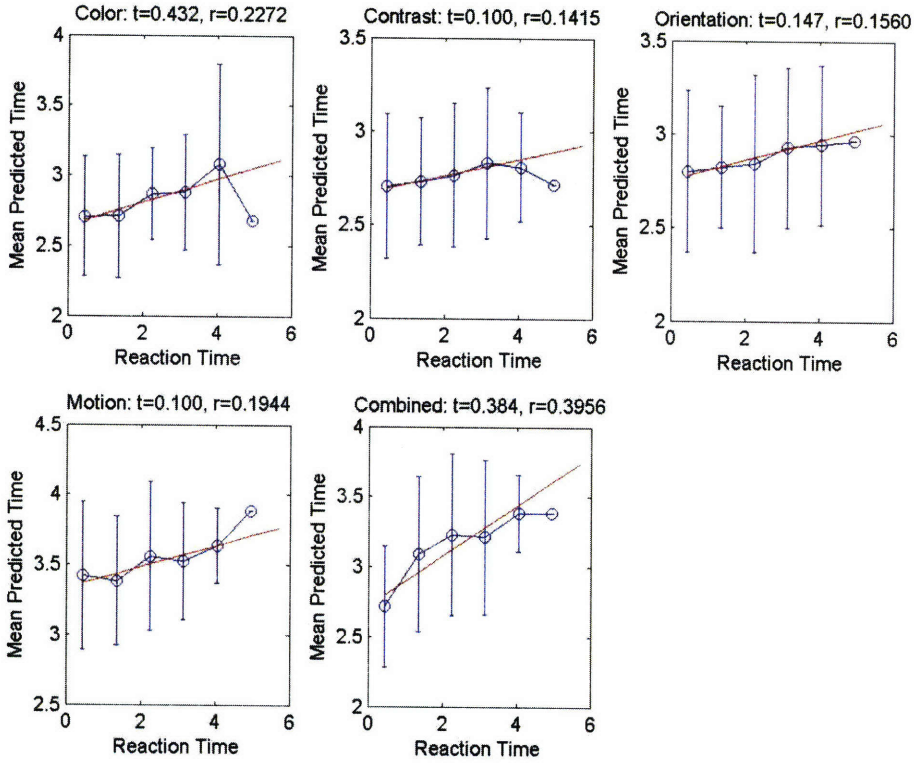

Figure 4.12: Reaction Time correlation with Predicted Time for a pedestrian crossing from the right. t is the threshold used, and r is the Pearson Correlation Coefficient. Error bars are standard deviation for that bin.

Probability Distribution	log likelihood	ratio to uniform dist
Uniform	-1.6199×10^6	1
Combined Saliency $\gamma = 1$ $\alpha = 0$	-1.6102×10^6	9.7×10^3
Normal distribution $\sigma = 2^o$	-1.5776×10^6	4.23×10^4
Combined Saliency $\gamma = 1$ $\alpha = .5$	-1.4445×10^6	1.754×10^5

Table 4.4: Probability of eye movements given saliency

measures. These results indicate that we can indeed use saliency to predict some proportion of driver behaviors. However, we still cannot predict a lot of the driver's behavior. We have also shown in these stimuli that the saliency model predicts that many of the objects that drivers are told to keep in mind while driving such as lane diving lines, and cars which are relatively close, are naturally salient.

In this chapter, we have shown that the saliency model we have developed can predict a significant proportion of what a driver either looks at or pays attention to. We discuss how we can take this further in the next chapter.

Chapter 5

Discussion

5.1 Summary

In summary, we have shown in this thesis, the implementation of the Statistical Saliency Model for video sequences, along with the extension of the model to add an optical flow feature in order to capture motion saliency. By essentially performing a statistical test to find outliers, we can identify spatial regions that are likely to attract attention.

Additionally, we have applied the saliency algorithm to stimuli that are more complex and realistic. The results we found in the experiments we conducted show that the algorithm does well in predicting certain behaviors in a driving simulation environment.

5.2 Future Work

5.2.1 Further Investigations and Extensions of the Model

From a modeling point of view, we would like to examine in detail how the different feature maps interact with each other in influencing the tendency to attract attention. For instance, the perception of motion of an object often requires the object to have features that we can track across time in a video, thereby making the object salient in both color and motion. This model as is, does not take time beyond that of estimating optical flow into account, and so ignores many physiological effects such as adaptation.

Additionally, this model incorporates information in an isotropic manner. This may not be the correct thing to do as it incorporates information across boundaries of surfaces and objects thus blurring the distinctions.

The human eye and brain both compensate for a lot of the jittery and noisy

motion to give us the percept of a smooth continuous dynamic stimuli. In order to model this better, we think that stabilizing a video would yield better performance. The analysis of how ego-motion affects a person's attraction to objects are also not well understood. Some unanswered questions on this include whether or not the brain compensates for ego-motion since it expects one to do this motion anyway, and so it is not surprising or salient.

5.2.2 Future Directions

Some of the future work we want to focus on include trying to answer the question of how much top-down influence is exerted in driving scenarios, and can we develop a model for this influence on driving behaviors.

Another promising and interesting direction of this research is the incorporation of machine learning into predicting saliency or driver behaviors. One can imagine having a machine learn by example what is considered salient and what is not, to build a computer representation and model of how a human would perceive.

Bibliography

[1] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid methods in image processing. *RCA Engineer*, 29(6), 1984. [cited at p. 11]

[2] E. H. Adelson and J. R. Bergen. Spatiotemporal energy models for the perception of motion. *Optical Society of America, Journal*, 1985. [cited at p. 15]

[3] George A. Alvarez and Steven L. Franconeri. How many objects can you track?: Evidence for a resource-limited attentive tracking mechanism. *J. Vis.*, 7(13):1–10, 10 2007. [cited at p. 3]

[4] C. Connolly and T. Fleiss. A study of efficiency and accuracy in the transformation from rgb to cielab color space. *Image Processing, IEEE Transactions on*, 6(7):1046–1048, Jul 1997. [cited at p. 12]

[5] Turk-Browne N et al. Linking implicit and explicit memory: common encoding factors and shared representations. *Neuron*, 2006. [cited at p. 4]

[6] W.T. Freeman and E.H. Adelson. The design and use of steerable filters. *Pattern Analysis and Machine Intelligence*, 1991. [cited at p. 13]

[7] Systems Technology Inc. Stisim drive: http://www.stisimdrive.com/. [cited at p. 29]

[8] Kyle R. Cave Jeremy M. Wolfe and Susan L. Franzel. Guided search: An alternative to the feature integration model for visual search. *Journal of Experimental Psychology: Human Perception and Performance.*, 15(3), 1989. [cited at p. 5, 6]

[9] Daniel Kahneman. Maps of bounded rationality: Psychology for behavioral economics. *American Economic Review*, 93(5):1449–1475, 2003. [cited at p. 25]

[10] C. Koch and N. Tsuchiya. Attention and consciousness: two distinct brain processes. *Trends in Cognitive Sciences*, 2007. [cited at p. 4]

[11] C. Koch and S. Ullman. Shifts in selective visual attention: Towards the underlying neural circuitry. *Human Neurobiology*, 4:219227, 1985. [cited at p. 6]

[12] M. S. Landy and J. R. Bergen. Texture segregation and orientation gradient. *Vision Research*, 1991. [cited at p. 13]

[13] Christof Koch Laurent Itti and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. [cited at p. 6, 9, 12]

[14] J. LeDoux. *Synaptic Self: How Our Brains Become Who We Are.* NY: Viking Press, 2002. [cited at p. 4]

[15] Seeing Machines. facelab: http://www.seeingmachines.com/. [cited at p. 35]

[16] John Medina. *Brain Rules.* Pear Press, 2008. [cited at p. 4]

[17] Adelson E. H. Gizzi M. S. Movshon, J. A. and W. T. Newsome. The analysis of moving visual patterns. *Experimental Brain Research*, 1985. [cited at p. 15]

[18] H. Pashler and J. Johnston. *Attention: Attentional Limitations in Dual-task Performance*, chapter 4. Psychology Press, 1998. [cited at p. 25]

[19] R. Rosenholtz and Z. Jin. A computational form of the statistical saliency model for visual search. *Journal of Vision*, 2005. [cited at p. 3, 7, 9]

[20] Ruth Rosenholtz. A simple saliency model predicts a number of motion popout phenomena. *Vision Research*, 1999. [cited at p. 9, 15]

[21] Ruth Rosenholtz. Search asymmetries? what search asymmetries? *Perception and Psychophysics*, 2001. [cited at p. 9]

[22] Ruth Rosenholtz. Visual search for orientation among heterogeneous distractors: Experimental results and implications for signal-detection theory models of search. *Journal of Experimental Psychology: Human Perception and Performance*, 2001. [cited at p. 7, 9]

[23] Nils Papenberg Thomas Brox, Andrs Bruhn and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. *European Conference on Computer Vision*, 2004. [cited at p. 15, 18]

[24] A.M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 1980. [cited at p. 3, 5]

[25] P. Verghese. Visual search and attention. a signal detection theory approach. *Neuron*, 31, 2001. [cited at p. 5]

[26] Andrew B. Watson and A. J. Ahumada Jr. A look at motion in the frequency domain. *Motion: Perception and representation*, 1983. [cited at p. 15]

[27] J. M. Wolfe. *Integrated Models of Cognitive Systems. Guided Search 4.0: Current Progress with a model of visual search.* 2007. [cited at p. 5]

Appendices

Appendix A

Code

A.1 Statistical Saliency Model

The following matlab code computes the local saliency according to the Statistical Saliency Model at a single scale given the feature in the $2-D$ matrix, *img*.

```
function std = sally(img, prm)
% std = sally(img, prm)
% Input arguments
%     img - height x width x d input image, where d is the number of dims
%     prm - optional parameters argument
%     prm.noise - noise in variance, if scalar, noise is applied
%                 equally to each dimension,
%                 if vector noise(k) is applied to variance
%                 in the k-th dimension
%     prm.filterScale - target pooling scale
%     prm.distractorRelativeSize - relative size of distr. pooling
%                                  scale
%     prm.cosmeticBlur - size of cosmetic gaussian blur
% Output
%     std - height x width saliency map


%%%%%%%%%%%%%%%%%%%%%%
% PARAMETER HANDLING %
%%%%%%%%%%%%%%%%%%%%%%
% grab some size info
[nr nc nd] = size(img);
```

```
l          = nr * nc;
% grab given parameters or use default if no parameters were provided.
if (~exist('prm','var'))
    prm = [];
end
[noise tSigma dSigma blur] = processParams(prm,nd);

%%%%%%%%%%%%%%%%%%%%%
% INITIALIZATION     %
%%%%%%%%%%%%%%%%%%%%%
img        = single(img); % for speedier convolution
% Useful Kernels and constants %
% the filter for getting the target region
sFilt  = fspecial('gaussian',ceil(2*tSigma)*2+1,tSigma);
% the filter for getting the distractor region (kind of)
bFilt  = fspecial('gaussian',ceil(2*dSigma)*2+1,dSigma);
% normalizing constant calculations
maxS   = max(sFilt(:)); maxB   = max(bFilt(:)); k      = maxB / maxS;
smallK = k/(1-k); bigK   = 1/(1-k);

%%%%%%%%%%%%%%%%%%%%%
% Here we go!!       %
%%%%%%%%%%%%%%%%%%%%%
% compute the mean of the target and distractor distributions
meanT  = nconv2(sFilt, img);
meanD  = bigK * nconv2(bFilt, img) - smallK * meanT;

% compute the covariance matrices for the distractor distributions
covarD = zeros(nd,nd,nc*nr);
for ii = 1 : nd
    for jj = ii : nd
        cross = img(:,:,ii).*img(:,:,jj);
        exIJT = nconv2(sFilt, cross);
        exIJD = bigK*nconv2(bFilt, cross);
        cvd = (exIJD - smallK*exIJT) - (meanD(:,:,ii).*meanD(:,:,jj));
        % add noise to the variance
        if (ii == jj)
            cvd = cvd + noise(ii);
        end
        covarD(ii,jj,:) = cvd(:); covarD(jj,ii,:) = cvd(:);
    end
```

```
end
% compute the inverse and determinants of the covariance matrices
inversD = invMats(covarD);

% Compute the mahalanobis distance from the mean of the target distribution
% to the mean of the distractor distribution.
meanD =  reshape(meanD,[nr*nc,nd])'; meanT =  reshape(meanT,[nr*nc,nd])';
mTD   =  mahalanobis(meanT, meanD, inversD, nd, 1);
std = sqrt(mTD); std = reshape(std, [nr nc]);

% final blur for cosmetic  purposes
if (blur)
    g = fspecial('gaussian',ceil(tSigma*2)*2+1, tSigma*2);
    std = imfilter(std, g,'replicate');
end

return

%%%%%%%%%%%%%%%%%%%%
% HELPER FUNCTIONS  %
%%%%%%%%%%%%%%%%%%%%
% computes [x - mu] * invcov * [x - mu], nd = # dimensions, l = # pixels
function md = mahalanobis(x,mu,invcov,nd,l)
    xMinusMu = x - mu;
    xMinusMuRep = repmat(reshape(xMinusMu, [nd,1,l]),[1,nd,1]);
    xMinusMuTimesInvers = sum(xMinusMuRep .* invcov, 2);
    xMinusMuTimesInversTimesXMinusMu =  ...
        sum(reshape(xMinusMu,[nd,1,l]).*xMinusMuTimesInvers,1);
    md = reshape(xMinusMuTimesInversTimesXMinusMu, [1,l]);
return

% (covars)^(-1) for each k in covars(:,:,k)
% handles special cases for 1,2, and 3 dimensions, otherwise slowly
% iterates for each pixel
function [invs dets] = invMats(covars)
    [nd nd l] = size(covars);
    if (nd == 1)
        dets = reshape(covars, [l 1]);
        invs = 1./covars;
    elseif (nd == 2)
        a = covars(1,1,:); b = covars(1,2,:); c = covars(2,2,:);
```

```
        dets = a.*c - b.^2;
        invs(1,1,:) = c;        invs(1,2,:) = -b;
        invs(2,1,:) = -b;       invs(2,2,:) = a;
        invs = invs ./ repmat(dets,[2 2 1]);
        dets = dets(:);
    elseif (nd == 3)
        a = covars(1,1,:); b = covars(1,2,:); c = covars(1,3,:);
        d = covars(2,2,:); e = covars(2,3,:); f = covars(3,3,:);
        dets        = a.*d.*f-a.*e.^2-b.^2.*f+2*b.*c.*e-c.^2.*d;
        invs(1,1,:) = d.*f-e.^2;   invs(1,2,:) = -b.*f+c.*e;
        invs(1,3,:) = b.*e-c.*d;
        invs(2,1,:) = invs(1,2,:); invs(2,2,:) = a.*f-c.^2;
        invs(2,3,:) = -a.*e+b.*c;
        invs(3,1,:) = invs(1,3,:); invs(3,2,:) = invs(2,3,:);
        invs(3,3,:) = a.*d-b.^2;
        invs        = invs ./ repmat(dets,[3 3 1]);
        dets        = dets(:);
    else
        invs = zeros(nd,nd,l);
        dets = zeros(l);
        for ii = 1 : l
            invs(:,:,ii) = inv(covars(:,:,ii));
            dets(ii) = det(covars(:,:,ii));
        end
    end
    dets = dets';
return

function [noise tSigma dSigma blur] = processParams(prm,nd)
    if (~isfield(prm,'noise'))
        noise = .001;
    else
        noise = prm.noise;
    end
    if (numel(noise) == 1)
        noise = noise * ones(nd,1);
    end
    if (~isfield(prm,'filterScale'))
        tSigma = 7/2;
    else
        tSigma = prm.filterScale;
```

```
    end
    if (~isfield(prm,'distractorRelativeSize'))
        dSigma = 4 * tSigma;
    else
        dSigma = prm.distractorRelativeSize * tSigma;
    end
    if (~isfield(prm,'cosmeticBlur'))
        blur = 2.3 * tSigma;
    else
        blur = prm.cosmeticBlur;
    end
return

function s = nconv2(kernel, img, border)
    if (~exist('border','var'))
        border = 'replicate';
    end
    [h w d] = size(img);
    img(:,:,d+1) = ones(h,w);
    u = imfilter(img,kernel,border);
    s = sum(kernel(:)) * u ./ repmat(u(:,:,end),[1 1 d+1]);
    s = s(:,:,1:d);
return
```

A.2 Extracting orientation

The following extracts orientation features given the luminance image.

```
function ori = ftOrientation(lum)
% ori = ftOrientation(lum)
% Input arguments
%     lum - height x width input CIELab Luminance of image
% Output
%     ori - height x width x 2 orientation feature map

noise = 1;
sigma = 16/14 * 1.75;
poolSc = 1.75 ;
g = fspecial('gaussian', 2*ceil(poolSc)+1, poolSc);
[h,v,r,l] = orientFilts(sigma);

% orientation energy
```

```
hi = imfilter(lum, h, 'replicate'); hi = hi.^2;
vi = imfilter(lum, v, 'replicate'); vi = vi.^2;
ri = imfilter(lum, r, 'replicate'); ri = ri.^2;
li = imfilter(lum, l, 'replicate'); li = li.^2;

% pool with scale
hi = imfilter(hi, g, 'replicate');
vi = imfilter(vi, g, 'replicate');
ri = imfilter(ri, g, 'replicate');
li = imfilter(li, g, 'replicate');

tot = hi + vi + li + ri + noise;
hv = (hi - vi) ./ tot;
dd = (ri - li) ./ tot;
ori = zeros(size(hv,1), size(hv,2), 2);
ori(:,:,1) = hv;
ori(:,:,2) = dd;

return

function [H,V,R,L] = orientFilts(sigma)
    h = ceil(3*sigma);
    x = repmat(-h:h,[2*h+1,1]);
    y = x';
    Gb = exp(-.5*((x/sigma).^2+(y/sigma).^2));
    Gb = Gb ./ sum(Gb(:));
    Ga = exp(-.5*((x/sigma).^2+((y-sigma)/sigma).^2));
    Ga = Ga ./ sum(Ga(:));
    Gc = exp(-.5*((x/sigma).^2+((y+sigma)/sigma).^2));
    Gc = Gc ./ sum(Gc(:));

    H = -Ga+2*Gb-Gc;
    V = H';
    GGa = imrotate(Ga, 45, 'bicubic', 'crop');
    GGa = GGa/sum(GGa(:));
    GGb = imrotate(Gb, 45, 'bicubic', 'crop');
    GGb = GGb/sum(GGb(:));
    GGc = imrotate(Gc, 45, 'bicubic', 'crop');
    GGc = GGc/sum(GGc(:));
    R = -GGa+2*GGb-GGc;
    GGa = imrotate(Ga, -45, 'bicubic', 'crop');
```

```
    GGa = GGa/sum(GGa(:));
    GGb = imrotate(Gb, -45, 'bicubic', 'crop');
    GGb = GGb/sum(GGb(:));
    GGc = imrotate(Gc, -45, 'bicubic', 'crop');
    GGc = GGc/sum(GGc(:));
    L = -GGa+2*GGb-GGc;
return
```

A.3 Extracting Contrast

The following matlab code extracts the contrast energy feature from the luminance image.

```
function c = ftContrast(lum)
% c = ftContrast(lum)
% Input arguments
%     lum - height x width input CIELab Luminance of image
% Output
%     c - height x width contrast feature map
a = 8; % 1/2 tap size
sigma = 5.33/2;
sigi = 0.71*sigma; sigo = 1.14*sigma;
t=-a:a;
gi = exp(-t.^2/(2*sigi^2)); gi = gi/sum(gi);
go = exp(-t.^2/(2*sigo^2)); go = go/sum(go);
innerFilt = gi' * gi;
outerFilt = go' * go;
c = imfilter(l, innerFilt - outerFilt,'replicate');
c = c.^2;
```