The detection and representation of foreground vs. background objects

Da In Kim

Advisor: Ellen Hildreth

Submitted in Partial Fulfillment of the Prerequisite for Honors in Computer Science

May 2013

© 2013 Da In Kim
Acknowledgments

I would like to express my sincere gratitude to my advisor Professor Ellen Hildreth for her tremendous help, dedication, and encouragement. Without her support, this thesis would not have been possible. She guided me in every step of research and writing of this thesis. I could not have imagined having a better advisor for my thesis as well as my academic career at Wellesley.

Besides my advisor, I would like to thank the rest of my thesis committee: Professor Jeremy Wilmer and Professor Eniana Mustafaraj for their encouragement, insightful comments and questions. I especially thank Professor Wilmer for his collaboration with my research for the past two summers and for his generosity in lending us his laboratory; his great insights and knowledge of psychological research have been essential in setting up and analyzing the perceptual experiment parts of this thesis.

I would also like to thank the Barbara Peterson Ruhlman ‘54 Fund for Interdisciplinary Studies, the Wellesley College Office of the President, and the Jerome A. Schiff Fellowship for funding my research for the past two years.

Last but not least, I am grateful my family and friends for supporting me and bearing with me for the past year. I would not be here without their love and care.
Abstract

Recovering the 3-D layout of a visual scene from 2-D images is necessary both for the human visual system and computer vision technology to analyze the location and structure of objects in the scene. Previous perceptual experiments suggest that the depths of foreground objects may be processed more quickly and represented more accurately than background objects (O'Toole & Walker, 1997; Becker et al., 1999). The first part of this thesis explored this hypothesis more deeply through perceptual experiments that used a simple search task with 3-D displays that were viewed stereoscopically. We found that the human visual system is better and faster at distinguishing the depths of foreground objects than the depths of background objects. This finding may be important for the design of computer vision technology. It suggests that in order to achieve human level performance in stereo processing, it may not be necessary to process the whole visual scene at a high resolution. The present human experimental study suggests that we may only need to process foreground objects, and not the whole visual image, at a fine spatial scale. For the second part of this thesis, a multi-resolution approach, based on ideas proposed earlier by Marr, Poggio, and Grimson (Marr & Poggio, 1979; Grimson, 1981), was implemented and found to improve the analysis of surface depths in the vicinity of object borders. This approach may reduce the computational resources needed to process stereo images without losing the most essential information about the 3-D layout of the visual scene.
# Table of Contents

Acknowledgments  

Abstract  

1. Introduction  

2. Perceptual Experiments  
   2.1. Background  
   2.2. Methods  
      - Experiment 1. Varying presentation time  
      - Experiment 2. Varying disparity difference  
      - Experiment 3. Crossed versus uncrossed disparity  
   2.3. Subjects  
   2.4. Results  
      - Experiment 1. Varying presentation time  
      - Experiment 2. Varying disparity difference  
      - Experiment 3. Crossed versus uncrossed disparity  
   2.5. Discussion  

3. Computational Modeling  
   3.1. Pre-Processing stereo images  
   3.2 Coarse scale stereo processing  
   3.3 Fine scale processing of foreground surfaces  

4. General Discussion  

References  

Appendix: Code
1. Introduction

In order to navigate through the environment, recognize objects, and interact physically with object surfaces, we need to recover the 3-D layout of a visual scene from the 2-D images that are projected onto the eyes. A primary cue used by the human visual system to perceive the depths of surfaces in the scene is stereo disparity (Marr & Poggio, 1979; Howard & Rogers, 2002; Brown, Burschka, & Hager, 2003; Harris & Wilcox, 2009). Stereo disparity arises from the difference in perspective provided by the two eyes. As a result of this difference, objects can appear at slightly different positions in the left and right images. The human visual system is able to detect this disparity in position and use it to infer depth (Figure 1). For tasks such as the recognition and manipulation of objects in the scene, it is important to segment the image into regions that belong to distinct objects. A strong cue to the presence of an object boundary is a large change in depth between two adjacent image regions. Stereo processing enables the detection of these boundaries and computation of the relative depth between surfaces meeting at boundaries in the image.
Figure 1. Examples of stereo images.

(a) A diagram of a birds’ eye view of two eyes viewing a scene with three simple objects at different depths and a diagram of the left and right images for this simple scene are depicted. The middle object is the point of fixation. The front object has crossed disparities and the back object has uncrossed disparities.

(b) A real stereo image pair, obtained by photographing a scene from two camera positions that are displaced horizontally.
In the process of analyzing the 3-D layout of objects in a scene, the distinction between foreground and background surfaces is also important. We reach first for objects in the foreground, navigate around surfaces closest to us, and recognize objects more easily when they appear in full view. Perceptual studies suggest that the human visual system may process foreground and background objects differently. For example, people tend to fixate the foreground objects of a scene more often (Rayner et al., 2007; Rothkopf et al., 2007; Vincent et al., 2009) and the depths of foreground objects may be processed more quickly and represented more accurately (O’Toole & Walker, 1997; Becker et al., 1999). This thesis explores more deeply the hypothesis that foreground and background objects are processed differently by the human visual system. Using a simple search task, we examine both the minimum time and the minimum difference in depth needed to detect an object whose depth differs from other objects, when embedded in a visual context that places the objects in the foreground versus the background. Our experimental results support the hypothesis that the depths of objects placed in the foreground are processed more quickly and represented more accurately, relative to the case where the objects appear in the background.

In this thesis, we also consider how differences in the processing of foreground vs. background objects can be incorporated into a computational model of stereo processing. Many models have been proposed for stereo processing in the human visual system and in computer vision systems (Marr & Poggio, 1979; Grimson, 1981; Dhond, 1989; Brown, Burschka, & Hager, 2003; Lazaros, Sirakoulis, & Gasteratos, 2008). The recovery of the 3-D layout and structure of objects from stereo images has important applications that include computer vision for robots and autonomous vehicles, and the construction of digital maps from aerial photographs (Lazaros, Sirakoulis, & Gasteratos, 2008). In previous models, there is no distinction between the processing of foreground vs. background objects, in terms of the time taken to process
these different conditions or the resolution of the representation of disparity that is derived for foreground vs. background surfaces.

In most models, the recovery of the depths of surfaces in the scene from stereo images typically involves three steps: (1) identification of image regions or features in the two images whose stereo disparity is to be measured, (2) the determination of the correspondence between features in the left and right images that represent the same physical elements in the scene, and (3) the measurement of stereo disparity between corresponding image features and use of this disparity to recover depth. The most challenging step of the process is step (2), the stereo correspondence problem. To solve this problem, all models of stereo correspondence restrict the overall range of stereo disparity that can be processed. Some models use a multi-resolution strategy in which a large range of stereo disparity is first processed at a coarse resolution, and the representation is later refined by mechanisms that process a smaller range of stereo disparity at a finer resolution (for example, Grimson, 1985; Dhond, 1989). Regardless of whether a single or multi-resolution approach is used to solve the stereo correspondence problem, stereo disparity is ultimately represented at a uniform resolution. We explore a simple model that combines techniques for stereo processing used in previous models, and incorporates the idea of processing objects that are closer to the viewer more quickly and at a finer resolution in depth, relative to the processing of objects that are further away from the viewer.

Section 2 describes the perceptual work conducted in this thesis in more detail. We first summarize some of the previous research on human stereo vision that is particularly relevant to the hypotheses examined in this thesis. We then describe the methods used in the experiments and the results obtained, and discuss these results in relation to our hypotheses. Section 3 describes our computational work. We first summarize the ideas that we incorporated from previous work on models for the computation of stereo disparity in the human visual system and in computer vision systems. We then describe how the differential processing of foreground vs.
background surfaces can be incorporated into our model. Finally, we show the results of some computer simulations with our model. Section 4 summarizes the contributions of this thesis and presents ideas for future work in this area.
2. Perceptual Experiments

2.1. Background

One of the reasons humans can move in a complex, dynamic scene so easily is that we are able to quickly perceive distinct objects and their relative depths at object boundaries, using cues such as stereo. As noted in the introduction, stereo cues arise from the disparity between the images seen in the left and the right eyes (Howard & Rogers, 2002; Harris & Wilcox, 2009). We can fuse these two images together into a single percept of the scene, using stereo disparity to sense the relative depth between objects. This mechanism of perceiving relative depth from two slightly different images allows us to easily identify the foreground objects in a scene.

Identifying the foreground surfaces is essential for people to navigate around the environment. Especially for tasks involving physical interaction with objects, determining the depths of foreground objects accurately and quickly is important. Perceptual studies have revealed differences in the processing of foreground objects versus background surfaces. For example, studies have shown that people tend to fixate more on the foreground objects than the background (Rayner et al., 2007; Rothkopf et al., 2007; Vincent et al., 2009), and they pay more attention to foreground objects (Mazza et al., 2005). The distinction between foreground and background objects is different from the figure vs. ground distinction. A figure is traditionally defined as the “thinglike” region or a region that corresponds to an object; a ground is defined as the regions that extend behind the contour or the spaces between objects (Palmer, 1999, p.280-284). Thus the figure vs. ground distinction is the distinction between objects and the surrounding background space. The foreground vs. background distinction that we use in this thesis is between objects that are closer to the viewer and objects that are farther away from the viewer, and there is a background space that surrounds both types of objects. Thus both foreground and background objects are “figures” and we are distinguishing different types of
figure objects. The previous studies, however, did not make this difference clear and rather examined foreground objects vs. background which includes both background objects and the surrounding space. This thesis explores more in depth the difference in the processing of foreground objects vs. background objects.

Some perceptual studies indicate that the human vision system may process crossed and uncrossed disparities differently. Crossed disparity occurs when objects are in front of the point of focus (Figure 1a). Uncrossed disparity occurs when objects are behind the point of focus. Foreground objects produce crossed disparity and background objects produce uncrossed disparity if the point of focus is between the foreground and background objects. Studies have shown that crossed disparity is processed more accurately than uncrossed disparity (Grabowska, 1983; Schor & Wood, 1983; Mustillo, 1985; Patterson et al., 1992), and detected more rapidly (Mustillo, 1985; Manning et al., 1987; Finlay et al., 1989; Patterson et al., 1995; Tam and Stelmach, 1998) although this bias toward crossed disparities is not seen in all studies (Howard & Rogers, 2002). As a result, people are more sensitive to differences in crossed disparities relative to uncrossed disparities (Schor & Wood, 1983). This sensitivity to crossed disparity may lead to people being more sensitive to the foreground objects than background objects.

These previous studies of the processing of crossed vs. uncrossed disparity did not make a distinction between the processing of foreground vs. background objects. Crossed vs. uncrossed disparity, and foreground vs. background, are not interchangeable, as foreground objects include all the objects that are in front of other objects regardless of where the point of focus is located. Likewise, background objects include all the objects that are behind other objects even though the point of focus may be behind the background objects. The human visual system may be more sensitive to the distinction between foreground and background objects than the distinction between crossed and uncrossed disparities.
O’Toole and Walker (1997) and Becker et al. (1999) examined the relation between the foreground/background and crossed/uncrossed distinctions and found that the distinction between foreground and background objects may be more important in depth perception than the distinction between crossed and uncrossed disparities. Using a traditional search paradigm, O’Toole and Walker (1997) displayed several small random-dot squares, with one at a different depth from the rest. All the squares were surrounded by an extended plane of random dots. The subjects were asked to detect the square that is at a different depth. The results indicated that subjects found the square at a different depth faster when all the squares were in front of the plane than when all the squares were behind the plane. O’Toole and Walker postulated that this difference in speed arises from two factors. One factor is that when the squares are placed in back, they are occluded by the surrounding plane, lacking clearly defined borders. When the borders are not clearly defined, it is hard to detect a precise depth. The other factor is that when squares are placed in back, there is a tendency to group these squares together into an extended plane and see all of these objects as one background surface at a single depth.

Becker et al. (1999) further examined the distinction between foreground and background objects and the distinction between crossed and uncrossed disparity. Subjects were shown a display with two small squares surrounded by a plane of random dots placed at a different depth and were asked to judge the depth between the two squares. There were three experiments, which varied the location of fixation. For one condition, the point of fixation was centered in depth in the display, placing crossed stimuli in front of and uncrossed stimuli behind the background. The other two conditions had the point of fixation either in front of the display, placing both crossed and uncrossed stimuli in front of the background, or behind the display, placing both crossed and uncrossed stimuli behind the background. Becker et al. (1999) found that performance was better when the stimuli were placed in front of the background than when they were placed behind the background, regardless of whether the stimuli produced crossed or
uncrossed disparities. They suggested that performance was worse in the latter case because of occlusion, which prevents an accurate measure of depth for objects in the background (Becker et al., 1999).

This thesis explores the difference between the processing of foreground and background objects in a more detail, examining whether the processing of foreground objects is more accurate or more rapid or both. It also explores whether the distinction between crossed and uncrossed disparities plays a significant role in the processing of relative depth between objects. All the experiments used a simple search task in which subjects were shown a display of four circular patches of random dots of known position and size. These circles were placed around the fixation point (above, below, left, and right) with the same distance from the fixation point in the 2-D image. The circular patches were surrounded by an extended plane of random dots that were at a different depth from the circles. One of the four circles was at a different depth from the rest and the subjects' task was to find this circle. This search task is more complex than the simple two-alternative forced choice tasks employed in Becker et al. (1999) and other studies of disparity discrimination. The added complexity was introduced to enhance the differences between the processing of foreground and background objects. It may also reveal individual differences across subjects more clearly. Previous studies have shown large individual differences in the processing of stereo information (Hildreth and Royden, 2011; Wilmer, 2008).

2.2. Methods

The visual displays for the search task consisted of a random dot surface with four circular patches arranged in a concentric pattern around a central fixation symbol (Figure 2). The depths of the four circular patches differed from the surrounding plane, and one circle was placed at a different depth from the other three. The task was to indicate the circle that is at a
different depth. Color anaglyphs were created from the left and right images and viewed with red-green glasses. Each trial began with a small disk appearing at the center of the display, with a fixation cross inside. The subject moved the mouse onto the disk, fixated the cross, and clicked the mouse to initiate the stimulus display. The random-dot pattern then appeared for a brief time. After the random-dot pattern disappeared, an image of four circular outlines was displayed and the subject specified the location of the odd-man-out circle by moving the mouse to the appropriate region and clicking.

![Figure 2. A sample visual display for the stereo-search task.](image)

(a) When viewed with red-green glasses with the red filter over the left eye, the top circular region appears in front of the surrounding plane of dots and the other three circular areas appear behind the plane. The displays were presented on a monitor viewed from a distance of 52 cm in a dark room. The overall size of the display is 17.5° x 17.5°, each circular region has a diameter of 4.4°, and the inner edge of each circle is located at a distance of 2.2° from the center of the display, as shown in the diagram in (b). Displays contained 4000 dots of size 0.1°, yielding an average dot density of 13 dots per square degree. For all displays, the fixation symbol was displayed in the plane of the computer screen, at zero disparity.

To measure the difference between the processing of foreground versus background objects, three separate experimental conditions were created in which the disparity of the surrounding plane of dots was varied, as shown in the three birds’ eye views in Figure 3. In each case, three circles were placed in front of the fixation symbol and one circle was placed in back,
or vice versa. In the diagrams, the short segments above and below the fixation point indicate the two depths of the circles. The surrounding plane of dots was placed at zero disparity (Figure 3a), behind the circles (Figure 3b), or in front of the circles (Figure 3c). The disparities of the circles were kept constant across all three conditions, and only the disparity of the surrounding plane changed. The first condition contains a mix of foreground and background objects (Figure 3a). The second one contains only foreground objects (Figure 3b) and the third contains only background objects (Figure 3c).

![Figure 3](image)

**Figure 3. Variation in the disparity of the surrounding plane.**

The angular disparities of the circles relative to fixation were +2' or -2' of visual arc, and the disparity of the surrounding plane was 0, +4' or -4', for all three conditions. The scenario (a) contains a mix of foreground and background objects. (b) All of the circles appear as foreground objects. (c) All of the circles appear as background objects.

**Experiment 1. Varying presentation time**

The purpose of this experiment was to measure the exact threshold for the time required to perceive depth using the stereo cue. To measure the threshold, the presentation time was varied using a staircase procedure. A fixed disparity of 4' visual arc between the two depths of the circles was used, as described in Figure 3. Each staircase began with a long duration of 800 ms. The display duration was initially decreased by 50 ms following each correct response, and after the first reversal (a reversal occurs when the subject answers incorrectly after a correct
response, or answers correctly after an incorrect response), this decrease was changed to 30 ms. The presentation time was increased by 100 ms following each incorrect response. Catch trials with a duration of 800 ms were also included to prevent the subjects from becoming frustrated with the increasingly difficult trials. Each staircase consisted of 8 reversals (4 peaks and 4 troughs) and the durations at which these reversals occurred were averaged to obtain the threshold presentation time needed to perform the task.

There were three experimental runs for this task. The first run had the surrounding plane in between the circles (Figure 3a). The second run had the plane in back (Figure 3b) and the third run had the plane in front (Figure 3c). Each experimental run contained separate interleaved staircases for the two disparity configurations in which the odd-man-out circle was placed in front of, or behind, the other three circles. Two staircases were completed for each disparity configuration and these staircases were not interleaved, but occurred successively. Thus, at any one time, there were only two staircases, with different disparity configurations (odd-man-out in front or odd-man-out in back), interleaved. The display for each trial was randomized so the subjects could not predict which disparity configuration will appear. The catch trials appeared between the staircase trials at random with 10% chance.

**Experiment 2. Varying disparity difference**

The purpose of this experiment was to measure the exact threshold for the difference in disparity required to perceive depth using the stereo cue. To measure the threshold, the difference in disparity between the circles was varied using a staircase procedure similar to Experiment 1. In this experiment, the presentation time for the random-dot pattern was held constant at 700 ms, and the relative angular disparity between the circles was varied. Similar to Experiment 1, this experiment also consisted of three separate runs using the three depth conditions shown in Figure 3.
The circles were again placed at positive and negative disparities that were arranged symmetrically around zero disparity, but the difference in disparity between the front and back circles was varied from trial to trial, using a staircase procedure. Each staircase began with a large angular disparity difference of 5.6' between the circles. This difference was initially decreased by 0.8' following each correct response, and after the first reversal, this decrease was changed to 0.4'. The relative angular disparity was increased by 1.2' following each incorrect response. Catch trials with a disparity difference of 6.4' were also included. Similar to Experiment 1, staircases for configurations in which the odd-man-out circle was in front or behind the other three circles were interleaved, and two staircases were completed for each configuration. Each staircase consisted of 8 reversals and the disparity differences at which these reversals occurred were averaged to obtain the threshold angular disparity difference needed to perform the task.

**Experiment 3. Crossed versus uncrossed disparity**

The purpose of this experiment was to determine whether the distinction between crossed and uncrossed disparities influences the resolution at which disparity is represented for foreground and background objects. The experiment consisted of four separate experimental runs that measured the threshold angular disparity difference needed to perform the search task for four depth conditions shown in Figure 4. For each condition, the fixation symbol appeared at zero disparity, in the plane of the monitor. For the conditions shown in Figures 4a and 4b, the surrounding plane of dots also had zero disparity, while the four circles all appeared as (a) foreground or (b) background objects. For the conditions shown in Figures 4c and 4d, the entire configuration of circles and surrounding plane was placed (c) behind fixation (uncrossed disparities) or (d) in front of fixation (crossed disparities).

If the distinction between crossed versus uncrossed disparity is a factor in determining
the threshold disparity difference needed to perform the search task, one might expect, for example, that condition (a), in which the circles are foreground objects and have crossed disparities, might yield especially low thresholds, while condition (b), in which the circles are background objects and have uncrossed disparities, might yield especially high thresholds. The conditions shown in Figures 4c and 4d trade off the advantages of being foreground objects or having crossed disparity with the disadvantages of being background objects or having uncrossed disparity. In conditions (c) and (d), the presence of all surfaces at crossed or uncrossed disparities may yield a strong driving signal for eye movements. Such eye movements are less likely for conditions (a) and (b), in which the fixation symbol is embedded in the surrounding plane. The staircases for this experiment were carried out in the same way as in Experiment 2, yielding thresholds for the angular disparity difference between the circles that are needed to perform the search task.
Figure 4. A sample of crossed versus uncrossed display.

For conditions in (a) and (b), the surrounding plane and the fixation symbol had zero disparity and the circles were placed symmetrically in depth around +4’ or -4’ of angular disparity, respectively. In condition (c), the surrounding plane had a disparity of +8’ and the circles were placed symmetrically in depth around +4’ of angular disparity, while in condition (d), the plane had a disparity of -8’ and the circles were placed symmetrically around -4’ of angular disparity.

2.3. Subjects

A total of 22 subjects participated in the stereo search task. Of the 22 subjects, two were male and the rest were female, and all were aged 18 to 26. All subjects had normal or corrected-to-normal vision and adequate stereo vision. Some subjects had participated in previous stereo and motion experiments, and those who did not had practice sessions before
starting the experiments. All 22 subjects completed Experiment 1, but a subset of 5 of these 22 subjects performed the condition in Figure 3a with a larger separation in depth between the circles and the background plane (circles were placed at angular disparities of +3’ and -3’ of visual arc relative to fixation rather than +2’ and -2’). Of the 22 subjects, 21 completed Experiment 2 and 15 completed Experiment 3.

Each subject typically conducted three one-hour sessions. The sessions did not occur on the same day, to allow the subjects to rest (one subject, however, had two sessions on the same day, separated by several hours). Due to time constraints, the time periods between sessions were not fixed. The total time it took for the subjects to complete the full three sessions ranged from one to three weeks. The sessions were conducted in a room painted completely black with no light, except that emitted by the computer monitor. The subjects viewed the display on the computer monitor with their heads fixed so that all the subjects viewed the display from the same distance. Subjects were asked to maintain central fixation through the display of the dot pattern, but eye movements were not recorded, so it is possible that subjects made some eye movements. They were allowed to rest between experiments, and during the experiments if needed. They were given a few test trials before each experiment to adjust to the task and the display. These test trials were also used to confirm that the subjects had adequate stereo vision to perform the search task.

2.4. Results

Experiment 1. Varying presentation time

To compare the time needed to process foreground objects and background objects, the individual time thresholds for the condition in which all circles appeared as background objects is plotted against the time thresholds for the condition in which all circles appeared in front (Figure 5a). Each data point represents an individual subject. The solid line indicates where the data
would lie if these thresholds were the same. A point above this line indicates a subject who performs the task more quickly when the circles are in front than when they are in back, and vice versa for points below the line. Time thresholds for both depth configurations spanned a large range, about 120-1100 ms with a mean value of 338 ms for the case where all circles appeared in front of the surrounding plane, and about 200-1000 ms with a mean of 520 ms for the case where all circles appeared in back. For 18 of the 22 subjects, less time was needed to detect the odd-man-out circle when all of the circles appeared in front. The remaining four subjects had higher time thresholds for the opposite case. Three of the four subjects within this latter group had time thresholds for the “all circles in back” condition that were within the range observed for all subjects, but lacked the advantage in the processing of foreground objects seen in other subjects, whose time thresholds were in the 100-300 ms range.

The dotted and dashed lines in Figure 5a highlight the asymmetry between the time thresholds for the two cases of background vs. foreground objects. The dotted line has a slope of 3 and the dashed line has a slope 0.5. A paired t-test indicates a significant difference in the time thresholds for the two conditions (p=0.03). The mean difference between the two time thresholds across all 22 subjects was 177 ms, but for individual subjects, the difference between the time thresholds for the “all circles in back” condition and the time threshold for the “all circles in front” condition ranged from -370 ms to 580 ms. We also examined the ratio between the time thresholds for the “all circles in back” and “all circles in front” conditions. This ratio ranged from 0.5-5.1 across 22 subjects, with a mean ratio of 2.3. This ratio was significantly different from 1.0, as indicated by a t-test (p=0.0002).

To compare the condition in which both foreground and background objects appear, the average time threshold for the “all front” and “all back” conditions for each subject is plotted against their time threshold for the condition in which the plane was at the center and there was a mix of foreground and background circles (shown in Figure 3a), shown in Figure 5b. The time
thresholds for the central-plane condition ranged from about 50-950 ms with a mean value of 237 ms. All but one subject required the same or less time to process this condition. The central-plane condition may have an advantage, because only the sign of disparity of the circles relative to the plane, and not the magnitude of disparity, is required to determine the odd-man-out. Furthermore, there is evidence of simultaneous contrast effects that may enhance the apparent difference in depth of the circles in this case (Brookes & Stevens, 1989; Lunn & Morgan, 1995), although these effects may be weak in the stereo disparity domain (Howard & Rogers, 2002).

![Figure 5. Results of Experiment 1, comparing time thresholds.](a) Time thresholds for the case in which all circles appeared in back of the surrounding plane are plotted against time thresholds for the case where all circles appeared in front. (b) The average time thresholds for the two cases where all circles were in front or all in back is plotted against the time thresholds for the case where the surrounding plane was placed between the circles, at zero disparity. The open circles are shown for the five subjects who performed the central-plane condition with a larger separation in depth between circles.

Traditional search studies reveal an asymmetry between the time needed to find an object that appears in front of a field of distractor items and the time needed to detect an object that is behind the distractors (e.g. O’Toole & Walker, 1997). We also examined the data from Experiment 1 to determine whether there is a difference between the time needed to process
depth configurations in which the odd-man-out circle is in front of the other three circles, and the time needed for configurations in which the odd-man-out circle is behind the other three. The mean time thresholds for these two cases (combining the conditions with all of the circles in back or in front) were 382 ms (odd circle in front) and 357 ms (odd circle in back). A paired t-test indicates no significant difference between the two (p=0.4).

Finally, we found that there is no significant difference in the time taken to find the odd-man-out circle for the first and second staircase within the individual experimental runs, combining the conditions with all circles in front or in back. The average time across subjects was 390 ms for the first staircase and 350 ms for the second staircase. A paired t-test indicates no significant difference between the two (p=0.1). This suggests that there was not a significant effect of learning on performance of the task over the short time frame of the individual experimental runs.

**Experiment 2. Varying disparity difference**

To compare the disparity needed to process foreground and background objects, the individual angular disparity thresholds for the condition in which all circles appeared as background objects is plotted against the disparity thresholds for the condition in which all circles appeared in front (Figure 6a). Each point represents a single subject and a total of 21 subjects completed this experiment. The solid line indicates where the data would lie if these thresholds were the same. Disparity thresholds for the “all circles in front” case lie in the range from 1'-4' of angular disparity with a mean value of 2.17', while thresholds for the “all circles in back” case lie in the range from 2'-5' of angular disparity with a mean value of 3.25'. In Experiment 1, the displays used a fixed disparity difference between the circles of 4', which is larger than the disparity thresholds obtained for most subjects in Experiment 2. Also in Experiment 2, the displays used a fixed presentation of 700 ms, which is larger than the disparity thresholds...
obtained for most subjects in Experiment 1.

The difference between the disparity thresholds for the “all circles in back” and “all circles in front” conditions ranged from -1’ to 3’ with a mean difference 1.08’. A paired t-test indicates a significant difference in the disparity thresholds for the two conditions (p < 0.0001). The dashed line in Figure 6a has a slope of 2, highlighting the large asymmetry between the front and back conditions for some subjects. We also examined the ratio between the disparity thresholds for the “all circles in back” and “all circles in front” conditions. This ratio ranged from 0.8 to 2.5 across the 21 subjects, with a mean ratio of 1.6. This ratio was significantly different from 1.0, as indicated by a t-test (p < 0.0001).

To compare the condition in which both foreground and background objects appear, the average disparity threshold for the “all front” and “all back” conditions for each subject is plotted against the disparity threshold for the condition in which the plane is at the center and there is a mix of foreground and background objects (Figure 6b). The disparity thresholds for the central-plane condition ranged from 1’ to 5’ of angular disparity with a mean value of 2.1’. All but five subjects required the same or less disparity difference for this condition. Again, the central-plane condition may have an advantage, because only the sign of disparity of the circles relative to the plane, and not the magnitude of disparity, is required to determine the odd-man-out. There was a slight difference between the conditions in which the odd-man-out circle was in front (mean disparity difference threshold of 2.62’) vs. behind the other three circles (mean disparity difference threshold of 2.40’). This difference was not significant as indicated by a paired t-test (p=0.07). There was no significant difference in performance for the first and second staircases within the individual experimental runs (p=0.4).
Figure 6. Results of Experiment 2, comparing angular disparity thresholds.

(a) Angular disparity thresholds for the case in which all circles appeared behind the surrounding plane are plotted against disparity thresholds for the case where all circles appeared in front. (b) The average disparity thresholds for the two cases where all circles were in front or all in back is plotted against the angular disparity thresholds for the case where the surrounding plane was placed between the circles, at zero disparity.

To determine how the time thresholds and angular disparity thresholds are related, the differences in angular disparity thresholds between the “all circles in back” and “all circles in front” conditions is plotted against the difference in time thresholds between these two conditions in Figure 7. The dotted lines indicate where these differences are zero and the solid line is a regression line, indicating a significant positive trend (\(\rho=0.62, p=0.029\)). In general, subjects who needed more time to perform the task when all of the circles appear as background objects also needed a greater difference in disparity to perform the task successfully in this case.
Figure 7. Disparity vs. time thresholds.

The difference between the angular disparity thresholds for the “all circles in back” and “all circles in front” conditions is plotted against the time thresholds for these two conditions.

Experiment 3. Crossed versus uncrossed disparity

This experiment used the same simple search task used in Experiments 1 and 2, using the conditions shown in Figure 4. Similar to Experiment 2, the presentation time for the random-dot pattern was held constant at 700 ms and the disparity difference between the circles was varied using a staircase procedure. A total of 15 subjects completed this experiment. The average disparity threshold for the two “all circles in back” conditions, combining the crossed and uncrossed disparity conditions (Figures 4b and 4d), was plotted against the average disparity threshold for the “all circles in front” conditions, again combining the crossed and uncrossed disparity conditions (Figures 4a and 4c) in Figure 8. The mean difference between the disparity thresholds for the “all circles in back” and “all circles in front” conditions was 1.12’ and the mean ratio between the disparity thresholds for these conditions was 1.7’ similar to that obtained in Experiment 2. By both measures, the difference in processing of foreground vs. background objects was significant as indicated by t-tests ($p < 0.0001$ for both measures). Thus, similar to the results of Experiment 2, a smaller disparity difference was needed to detect the
odd-man-out circle, when all of the circles appeared as foreground objects relative to the case where all of the circles appeared as background objects, regardless of the presentation of crossed or uncrossed disparities.

Figure 8. Results of Experiment 3, comparing “all back” and “all front” conditions. The average disparity thresholds for the configurations in which all circles appeared in back (Figures 4b and 4d) are plotted against the average disparity thresholds for the configurations in which all circles appeared in front (Figures 4a and 4c).

To determine whether circles appearing with crossed vs. uncrossed disparity affects the result, the disparity difference thresholds for the crossed disparity conditions is plotted against the disparity difference thresholds for the uncrossed disparity conditions (Figure 9). One plot shows the disparity difference thresholds for the depth configurations in which all circles appear in the foreground (Figure 9a) and the other for the depth configurations in which all circles appear in the background (Figure 9b). One might expect an advantage of the circles being in front of the background plane and having crossed disparity, but this does not appear to be the case. There is no consistent difference in performance for crossed vs. uncrossed disparity conditions. Paired t-tests also support this conclusion that there is no significant difference between crossed and uncrossed disparities (p=0.3 for circles appearing in the foreground, p=0.7 for circles appearing in the background).
Figure 9. Results for Experiment 3, comparing crossed and uncrossed disparities.
(a) The disparity difference thresholds for the depth configuration shown in Figure 4a is plotted against thresholds for the configuration shown in Figure 4c. (b) The disparity difference thresholds for the depth configuration shown in Figure 4d is plotted against thresholds for the configuration shown in Figure 4b.

2.5. Discussion

The results of these experiments build upon our understanding of the temporal and spatial differences in the processing of foreground vs. background objects. It has been hypothesized that the differences arise due to the differences in the processing of crossed vs. uncrossed disparities because foreground objects often produce crossed disparities and background objects often produce uncrossed disparities. Becker et al. (1999), however, were among the first to differentiate the issue of foreground vs. background objects from the issue of crossed vs. uncrossed disparities. Their results suggest that the distinction between foreground and background objects may play a bigger role than the distinction between crossed and uncrossed disparities. The results of these experiments support the findings of Becker et al. (1999). However, as we did not measure subjects’ eye movements, the subjects could have moved their fixation from the specified fixation point to the stronger depth cue of the surrounding surface in Figure 4c and 4d. If this is true, it would have made our displays of crossed and
uncrossed disparities more similar to each other. This could explain why we did not find a large difference between crossed and uncrossed disparities (Figure 9).

The Becker et al. study used a small set of combinations of disparity differences and presentation times in the experiments that compared depth order judgements for foreground vs. background objects. The experiments summarized here examined in more detail, the minimum presentation time and disparity difference needed to perform a simple search task, comparing thresholds for foreground vs. background objects. For most subjects, more time and larger disparity differences were needed to discriminate background objects at different depths, in comparison to foreground objects. For a few subjects, the time thresholds or disparity thresholds were 3-5 times larger for background vs. foreground objects.

The study by O'Toole and Walker (1997) demonstrated that under some circumstances, serial search is needed to detect an object that is at a different depth from a field of distractor objects, and that more time is needed to search for a target among background objects, compared to searching for a target among foreground objects. It is likely that the additional time needed to shift attention between background objects contributed to the differences in temporal thresholds that we observed in Experiment 1. For the stimuli used in Experiments 2 and 3, however, the presentation time was fixed at 700 ms, which should provide sufficient time for subjects to shift their attention over the four circular regions of the displays. Even with this relatively long viewing time, there is a large difference in the spatial resolution of the representation of disparity, or depth, of foreground vs. background objects.

We observed large individual differences in the threshold presentation time and disparity difference needed to perform our search task. There was a large spread in these two thresholds across subjects, and a few subjects had especially large asymmetries in thresholds for foreground vs. background objects. Most of this group of subjects had longer time and larger disparity thresholds for background objects, but a few subjects had longer time thresholds for
the case where the circles all appeared in front of the surrounding dot plane. Large individual
differences are also found in previous studies (Hildreth and Royden, 2011; Wilmer, 2008). Tam
and Stelmach (1998) also observed a large range of 20-1000 ms in the temporal thresholds
needed for the simpler task of discriminating the depth between two isolated patches of random
dots in dynamic random-dot stereograms.

These results, however, do not indicate at what point in the processing of stereo disparity
and construction of a 3-D representation the distinction between the processing of foreground
vs. background objects arises. There are at least three possibilities. First, the initial detection of
stereo disparity by disparity selective mechanisms may be influenced by the interpretation of an
image region as belonging to a foreground object or background surface. Second, the solution to
the stereo correspondence problem, during which ambiguous disparity signals are resolved,
may be influenced in some way by the surface context (e.g. the stereo correspondence
computation is resolved more quickly, or with greater precision, for foreground object surfaces).
Third, the creation of an explicit representation of disparity or depth for each object surface,
possibly requiring focused attention, may be influenced in some way by the surface context.
Further empirical studies are needed to determine the processing stages at which the distinction
between foreground vs. background plays a role.

The finding that the human visual system is better and faster at detecting and
distinguishing foreground objects than background objects may be important for the design of
computer vision systems. This finding suggests that there may be two stages of processing
required to analyze stereo images. These two stages can be performed using a multi-resolution
approach, where the whole visual image is processed at a coarse scale first and then the
surfaces in the foreground are processed at a finer scale. A multi-resolution approach for solving
the stereo correspondence problem was first proposed by Marr and Poggio (1979) and later
implemented by Grimson (1985). This approach, however, processes the whole visual scene at
a finer scale (Grimson, 1985). The present human experimental study suggests that this is not necessary; we only need to process foreground objects, and not the whole visual image, at a finer scale. A modified multi-resolution approach, where the visual image is processed at a coarse scale and then only the foreground objects are processed at a finer scale, may be sufficient to achieve human level processing. This modified approach may reduce the computational resources needed to process stereo images without losing the most essential information about the 3-D layout of surfaces in the scene.
3. Computational Modeling

The perceptual experiments described in the previous section support the hypothesis that in human stereo processing, the disparity (or depth) of foreground objects is analyzed more quickly and accurately than that of background objects. As noted in the introduction, many models have been proposed for human stereo processing and for the analysis of stereo images in computer vision systems (Marr & Poggio, 1979; Grimson, 1981; Dhond, 1989; Brown, Burschka, & Hager, 2003; Lazaros, Sirakoulis, & Gasteratos, 2008). These models generally do not make any distinction between the processing of foreground vs. background surfaces during the solution of the stereo correspondence problem and construction of a representation of stereo disparity. In this thesis, I implemented a simple stereo model that integrates some basic techniques for stereo processing drawn from previous work, and used this model to explore ideas for processing the disparity of foreground vs. background surfaces differently.

In the simple displays used in the perceptual experiments, the identification of a circular patch of dots as being in the foreground vs. background is well defined. In a natural scene containing many surfaces at different depths with complex 3D structure, the concept of foreground vs. background is more difficult to define, as many surfaces have some parts that are occluded by other objects that are closer to the viewer, while also occluding other, more distant surfaces. In the model that I implemented, stereo correspondence is first computed at a coarse spatial scale that considers a large range of stereo disparity between corresponding regions of the left and right images. This processing yields a rough estimate of the depths of surfaces in the scene. The results of this coarse scale processing are then used to guide the computation of stereo disparity at a finer spatial scale, but at this scale, the system focuses on foreground objects defined as those objects that are closer to the viewer (or camera), and computes a more accurate estimate of disparity for these object surfaces. In general, there may
be objects at many different depths. The algorithm arbitrarily chooses the closest half of the observed range of disparities to be the foreground, and the further half to be the background. As a result, the depth of surfaces closer to the observer are computed more quickly and accurately, which can be useful for tasks such as navigation and object manipulation. Furthermore, the overall computational resources needed to process stereo are reduced, because the stereo disparity of more distant object surfaces is computed only at a coarse spatial scale.

Section 3.1 describes the first stage of stereo processing in which the images are filtered and regions are selected for stereo matching. Section 3.2 describes the methods used for the coarse scale computation of stereo disparity and presents sample results of this processing. Section 3.3 describes the strategy for refining the processing of disparity for surfaces closer to the viewer and presents further results.

**3.1 Pre-Processing stereo images and selecting regions for stereo correspondence**

As described in the introduction, the first stage in stereo processing is to identify the image regions or features in the two images whose stereo disparity is to be measured. Stereo systems differ greatly in this first stage of processing. Some stereo models attempt to match small regions of the left and right images (Dhond, 1989; Brown, Burschka, & Hager, 2003; Bradski & Kaehler, 2008) while others match localized features such as intensity edges (Marr & Poggio, 1979; Grimson, 1981; Dhond, 1989; Brown, Burschka, & Hager, 2003). Most models first filter the images in a way that embodies smoothing and differentiation of the image intensities (Marr & Poggio, 1979; Grimson 1981; Dhond, 1989). The smoothing operation reduces the sensitivity of the matching process to noise in the images and controls the spatial scale at which the images are processed.

We first filter the stereo images by convolving them with a Laplacian-of-Gaussian operator (Marr & Hildreth, 1980; Grimson, 1981; Dhond, 1989) whose shape is similar to the
receptive fields of neurons in the human retinae (Grimson, 1981; Palmer, 1999, p.147-148). The Laplacian-of-Gaussian operator is expressed mathematically as:

\[ \nabla^2 G = \frac{1}{\sigma^2} \left( \frac{r^2}{\sigma^2} - 2 \right) e^{-\frac{r^2}{2\sigma^2}} \quad r^2 = x^2 + y^2 \]

The spatial scale of this operator is determined by the space constant (\( \sigma \)) of the Gaussian, which yields an operator whose central positive region has a diameter \( \omega \) given by:

\[ \omega = 2\sqrt{2\sigma} \]

An example of a Laplacian-of-Gaussian operator is shown in Figure 10a and the result of convolving an image with this operator is shown in Figure 10b. For this example, \( \omega \) is 8 pixels. The size of each image of the stereo pair is 400 x 420 pixels.

Figure 10. The effect of Laplacian-of-Gaussian filter.
(a) Example of a Laplacian-of-Gaussian operator. (b) Results of convolving the left image of the real stereo image pair from Figure 1c with the operator.
The Laplacian-of-Gaussian filter enhances the changes of intensity, or edges in the image, which are important for determining stereo correspondence. The correspondence between regions in the left and right images that have only small fluctuations in intensity, or low-contrast edges, cannot be identified reliably, so the algorithm that I implemented only computes stereo disparity for regions in which the overall contrast is above a particular threshold. We measured the total contrast within each small region of the left filtered image by calculating the sum of all absolute values in the region and dividing the sum by the number pixels of the region. The threshold was set to be 5% of the maximum contrast. The stereo disparities of regions with contrast smaller than this threshold were not computed (Figure 11).

![Figure 11. The regions with contrast larger than the threshold.](image)

(a) The natural image and (b) the regions with contrast larger than the threshold, shown in white.

### 3.2 Coarse scale stereo processing

The model uses an area based matching scheme for calculating disparity. Square patches of the left filtered image are compared to square patches of the right filtered image at the same vertical position in the two images, but for different horizontal positions (stereo disparity), to determine which patch in the right image represents the best match (Figure 12).
The size of the neighborhood and the range of horizontal disparities considered are free parameters that can be set in the implementation. Initial testing considered two different measures for comparing patches in the left and right images, correlation and the sum-of-absolute-differences (for example, Lazaros, Sirakoulis, & Gasteratos, 2008). The sum-of-absolute-differences yielded better results; it is also the method used by the Open Source Computer Vision Library (OpenCV) to find matching regions between left and right images (Bradski & Kaehler, 2008, p.439). This measure was used for the results reported here.

For each patch in the left image, the sum-of-absolute-differences is computed between this left image patch and a set of patches in the right image at different horizontal positions, and the disparity of the patch that yields the minimum difference is initially identified as the best disparity for the center location of the left image patch.

![Figure 12. Square patches of left and right images to be matched.](image)

For each point \((x, y)\) in the left image, a square patch around that point is selected. Another square patch of the same size and centered around the location \((x+d, y)\) is selected from the right image, where \(d\) is stereo disparity. These patches are compared to each other at the same vertical position in the two images, but for different horizontal positions.
Figure 13 shows an example of a stereo disparity map that contains the best disparity computed for each location of the left image obtained from this first stage of the matching process. To obtain this disparity map, a Laplacian-of-Gaussian operator with diameter $\omega$ of 8 pixels was used. The disparities were computed within the range of -24 to 24 pixels and the neighborhood size used was 25 pixels for both horizontal and vertical dimensions. Stereo disparity is displayed in different shades of gray, with the disparities for objects closer to the camera shown with darker intensity. The zero-crossings of the convolution result, which correspond roughly to the locations of the intensity edges in the image, were computed to see how the computed disparities compare to the object surfaces in the real image. The map containing the zero-crossings has been overlayed on top of the computed disparities in Figure 13c.
Figure 13. Stereo disparity map.
(a) The left image of stereo images. (b) Stereo disparities have been computed by matching regions of the left filtered image to regions of the right filtered image. (c) Zero-crossings of the filtered image, which correspond roughly to intensity edges, have been overlayed on top of the stereo disparities.

This basic stereo matching algorithm does a decent job of computing disparities for the images, but the disparities of high-contrast regions tend to spread into low-contrast regions. For example, you can see in 13c that the depth of the upper left region of the Arizona can has spread into the nearby regions of the clock. This occurs in part because the high-contrast regions contribute more to the differences between patches in the left and right filtered images. When determining the best match by calculating the sum-of-absolute differences between the patches in the left and right images, it is hard to find the best match between low-contrast regions.
because points that are a few pixels apart are relatively similar to each other in value. In high-contrast regions, however, points that are a few pixels apart can be very different in value and can skew the resulting disparities. This problem is called over-reaching in the literature. Okutomi et al. (2002) proposed a solution that reduces over-reaching in stereo processing. When calculating the disparity for each point, their solution takes into account the disparities of the neighboring points. Thus, after an initial disparity map has been created, the modified algorithm goes through each image location again. For each location \((x, y)\), it compares the sum-of-absolute differences obtained for the best disparities at neighboring locations. The disparity that yielded the minimum difference over the neighborhood is then assigned to the location \((x, y)\) (Okutomi et al., 2002). The authors found that incorporating this procedure into their stereo matching algorithm produced a cleaner result in tests with several natural image examples (Okutomi et al., 2002). The authors used five different images from a 2-D arrangement of five cameras rather than two stereo views, but their solution may still improve the results of the stereo matching algorithm with two images.

I implemented this algorithm and applied it to the results obtained earlier (Figure 14). At every location \((x, y)\), the algorithm records both the best disparity, \(d(x, y)\), and the value of the sum-of-absolute differences \(v(x, y)\) for this particular disparity. Within a neighborhood of each location, the algorithm compares values of the sum-of-absolute differences and selects disparity \(d_{\text{best}}\) that yielded the smallest difference. Then it finds all disparities within a small range around \(d_{\text{best}}\) and calculates an average of these disparities. This average disparity is then assigned to the location \((x, y)\). This strategy has the effect of reducing the distortion of disparities calculated in low-contrast regions that are adjacent to high-contrast object borders. Applying the over-reaching strategy improves the result obtained, especially in the vicinity of the object borders.
Another problem that both the human visual system and computer vision systems face is the problem of half-occlusions. Half-occlusions occur when a region is visible in one eye but not the other eye because it is occluded by an object (Anderson & Nakayama, 1994; Brown, Burschka, & Hager, 2003; Harris & Wilcox, 2009) (Figure 15a). These half-occlusion regions can produce errors in stereo matching because there can be no match for these regions. Egnal and Wildes (2002) analyzed several strategies that have been proposed to help with matching these regions. One method they studied was left-right checking. This method performs the same stereo matching algorithm twice, one matching the left image to the right and the other matching the right image to the left. When an area is visible in both images, it will produce opposite disparities from these two matching directions. For half-occlusion regions, however, the two disparities produced would not be matched. The left-right checking detects these regions with differing disparities and replaces the disparities for these regions with those from neighborhood regions. The left-right checking increases the computational resources needed for the algorithm because it performs stereo matching twice, but the improvement in results obtained over algorithms with only one direction of stereo matching compensates for this increase.
Figure 15. The results of left-right checking.

(a) This illustration shows the cause of half-occlusions. The small object closer to the viewer occludes regions of the surface in the back. If the scene was only visible from the right eye, the region A would not be visible. Likewise if the scene was only visible from the left eye, the region B would not be visible. (b) Regions of the right filtered image have been matched with regions of the left filtered image and the areas with differing stereo disparities obtained from the two matching directions have been selected (shown in black). (c) For the selected regions, the stereo disparities of the neighborhood points have been applied. (d) Zero-crossings have been overlayed on top of the result.
As predicted, the strategy implemented by Okutomi et al. (2002) for reducing the over-reaching problem refined the stereo matching. Implementing the left-right checking strategy also refined the stereo matching by cleaning up the areas around the borders of the objects. For example, the neck of the wine bottle after left-right checking is more accurate than before. As a result, the most prominent objects in the original image (Figure 11a), the can and wine bottle, also stand out with clear borders in the stereo matching result. The areas between objects, however, still need improvement. For example, in the original image (Figure 11a), there is a space between the can and the wine bottle, but this space cannot be seen in the result (Figure 15). To further improve the computation of depth for these regions, we will apply the same stereo matching algorithm again at a finer scale.

3.3 Fine scale processing of foreground surfaces

The stereo processing described in section 3.2 is first applied to stereo images that have been filtered with large Laplacian-of-Gaussian operators that perform a coarse spatial filtering of the images. At a coarse spatial scale, matching can be performed over a large range of stereo disparity without yielding many “false matches” (Marr & Poggio, 1979; Grimson, 1981, 1985; Dhond, 1989), but the disparities are not very precise due to the large amount of smoothing of the image intensities. The results from the coarse scale processing can be used to guide stereo matching at a finer scale that considers a smaller range of stereo disparity (Marr & Poggio, 1979; Grimson, 1981, 1985; Dhond, 1989).

The same stereo images (Figure 1c) were filtered again with a smaller Laplacian-of-Gaussian operator with diameter $\omega$ of 4 pixels; it was smaller than the previous one by a factor of two. Image patches in the left filtered image were again compared to image patches in the right filtered image at the same vertical position and for different horizontal positions, but the stereo disparity obtained at the coarse scale was used to determine the region
of the right image that is searched for a matching right patch (Figure 16). The sum-of-absolute-differences measure was computed for a smaller range of disparities around a horizontal location in the right image indicated by the disparity calculated at the coarse scale. The disparities were computed within the range of -6 to 6 pixels and the neighborhood size used was 13 pixels for both horizontal and vertical dimensions.

Motivated by the results of the perceptual experiments, this fine-scale processing of stereo disparity is focused on regions of the image where the coarse disparities indicate that the object surfaces are closer to the observer. The regions are selected by taking the first 60% of the range of disparities detected at the coarse scale, and choosing regions with any disparities in that range. For each point in the selected regions, the offset disparity is the disparity obtained at that point using the coarse scale. The square patch around this point is compared to patches of the right image at different horizontal positions (Figure 16). Furthermore, to improve the disparity
computation in the vicinity of the object borders, the algorithm needs to know more precisely, where the transition occurs between the disparity of the background surface and that of the foreground surface. Thus, some of the background surface near the foreground object was included to refine the stereo computation on both sides of the border. To consider all the possible disparities at the borders, I calculated the minimum and maximum disparity in the square patch around each point. If the minimum was smaller than the minimum range the algorithm was considering, then the algorithm considered all disparities starting at this minimum. For example, suppose at a point \((x, y)\), the coarse scale disparity obtained is 5, so that the minimum disparity to consider is 5-6 = -1. The algorithm then found the minimum disparity in the neighborhood to be -5. Then the algorithm would consider disparities starting from -5 for this point \((x, y)\). The same consideration was applied for the maximum disparity.

After the calculating the disparities for the foreground objects, the over-reaching and left-right checking strategies were applied to the disparities, similar to the coarse scale processing. The final result shows an improvement over the previous coarse scale result (Figure 17). In particular, the disparities in the areas around the book, the borders of the can, and the neck of the wine bottle are improved (highlighted with ovals in Figure 17d). The result shows that applying the stereo matching processing to a fine scale results in a marked improvement that compensates for the increase in computational resources. In addition, some of the computational resources were reduced by only applying the results to the foreground objects and not the background.

The stereo algorithm described here performs a fixed amount of computation at each image location. The amount of computation depends on parameters such as the size of the image patches used in the stereo matching, the size of the neighborhood used to implement the over-reach strategy, and the range of disparities considered for each location. The use of left-right checking doubles the amount of computation, because stereo disparity is computed
from the left to the right image, and again from the right to the left image. The use of this strategy, however, yields better results in the vicinity of object boundaries where half-occlusions occur. At the fine scale, the amount of computation is reduced by restricting the analysis to a subset of image locations on surfaces that are closest to the observer. The disparity of more distant surfaces does not need to be so precise for tasks such as navigation and object manipulation, in which the observer typically interacts primarily with surfaces in the environment that are closest to the observer.
Figure 17. The result of applying the stereo matching algorithm at a fine scale.

(a) The original left image of stereo images is shown. (b) The result from the previous coarse scale matching is shown for comparison, same as Figure 15c. (c) The regions whose disparities were calculated at the fine scale are shown in white. (d) The result of the fine scale matching is shown with circles highlighting the regions where the most improvement occurred, compared to Figure 17a. (e) The zero-crossings are overlayed on top of the result of the fine scale matching.
4. General Discussion

The human visual system and computer vision technology need to analyze 2-D images to construct the 3-D visual scene. In constructing these 3-D visual scenes, there may be differences in the processing of foreground vs. background objects as the foreground objects are more relevant when navigating around the scene and reaching for objects. Through perceptual experiments, this thesis found that indeed, conditions with foreground objects have lower thresholds in both presentation time and difference in the disparities between objects needed to detect depth changes. The results suggest that the foreground objects are processed more quickly and represented more accurately than background objects. Furthermore, the results for time and disparity thresholds have shown to be highly correlated with each other; a subject who needed more time to perform the task when all the circles appear as background objects also needed a greater difference in disparity to perform the task successfully in this case. This result suggests that these two processes, the time and difference in disparity needed to distinguish objects at different depths, are related. This correlation may arise from a multi-resolution mechanism that selectively processes foreground objects at a high resolution and this may happen very early on in visual processing.

We applied this result to the design of a computer stereo system and implemented a multi-resolution approach. This approach analyzes the whole visual scene at a coarse scale. Then it selects and analyzes only the foreground objects at a fine scale using the disparities obtained at the coarse scale as a guide. Before the algorithm could analyze stereo disparity at a fine scale, we needed to implement an algorithm to analyze this information at a coarse scale. We implemented a basic stereo matching algorithm, using an area based matching strategy that compared left and right filtered image patches using a sum-of-absolute-differences measure, which produced a reasonable map of stereo disparities, but it still needed to be refined. Thus, we
implemented a strategy to reduce the over-reaching problem where the high-contrast regions contribute more to the differences between patches in the left and right filtered images than the low-contrast regions (Okutomi et al., 2002). We also implemented the left-right checking strategy to identify half-occlusions and refine the borders around objects (Egnal & Wildes, 2002). Both these strategies improved the results of stereo matching (Figure 15). We further implemented the stereo matching algorithm at a fine scale, using the disparities obtained at the coarse scale as a guide (Marr & Poggio, 1979; Grimson, 1981). We also applied the previous two strategies that reduce the over-reaching problem and identify half-occlusions to the fine scale matching. The result obtained after fine scale matching shows improvement over the previous coarse scale result (Figure 17).

The proposed multi-resolution approach has the potential to improve the stereo matching algorithm without using a lot of computational resources. For future research, however, there are several aspects of the current algorithm that can be improved. First of all, with the current algorithm, the disparity for every point is calculated at a coarse scale. However, since the disparities of the foreground objects are calculated again at a fine disparity, not all the disparities need to be calculated. The disparity can be calculated for every other point and the points whose disparities are not calculated can take on the value of the disparity in the neighborhood with the best correlation (this method is also used to reduce the over-reaching problem). This strategy can reduce the computational resources considerably.

Furthermore, for a better result, the strategy to find the half-occlusions could be improved. With the current strategy, some areas between the objects are not yet detectable. For example, in Figure 1b, you can see an area with half-occlusions between the can and the wine bottle. If you examine Figure 15b, however, the algorithm that detects the half-occlusions is not finding this area. If these areas were detected and disparities for these areas were measured correctly, it would significantly improve the result. Moreover, more research needs to be done on
ways to accurately detect matches for low contrast regions (Figure 11b). The algorithm currently does not measure disparities for these regions. This is not a huge problem for the current images because there are a lot of areas with high contrast in the images, but for images with many low-contrast regions, this method could be problematic. In addition, the time frame of this thesis did not allow for extensive testing. Thorough testing on both artificial images, where we know the exact answer, and natural images needs to be done to confirm that the promising results of this approach are not restricted to the current stereo images.

The finding that applying the stereo matching algorithm a second time to foreground objects at a fine scale improves the result has many applications. In computer vision technology, applying this approach could improve the result of stereo matches without much increase in computational resources. Furthermore, this approach was driven by the result of human perceptual experiments. Further study on the human visual system could lead to more discoveries about ways to improve computer vision technology.
References


Toronto.


Canadian J. Experimental Psychology, 52(1), 56-61.


Appendix: Code

First, the `stereoMatchingCoarse.m` script is executed and then the `stereoMatchingFine.m` script is executed. In `stereoMatchingCoarse.m`, the `compStereo` function is used to calculate stereo disparities at each point. In `stereoMatchingFine.m`, the `compStereoFront` function is used to calculate stereo disparities at each point using the disparities calculated at the coarse scale matching as a guide. The `compStereoNeighbour` function is used to reduce the over-reaching problem and the `leftRightChecking` function is used to identify the half-occlusions and remove those areas.

```matlab
%% stereoMatchingCoarse.m
% specialized script to process a pair of stereo images at a coarse spatial scale
% reduce image sizes to 400x420
left = imread('lt2.jpg'); left = left(60:459,80:499); right = imread('rt2.jpg'); right = right(60:459,80:499);
% convolve images with laplacian-of-gaussian operator, w = 8
lap8 = laplacian(8.0); cleft8 = conv2D(left,lap8); cright8 = conv2D(right,lap8);
% compute zero-crossings - zcleft and zcright store the slopes (magnitude of the gradient at the locations of % the zero-crossings), zmapleft and zmapright store the contours (with sufficient slope)
zcleft = zeros2D(cleft8); zcright = zeros2D(cright8);
zmapleft = zcMap(zcleft,0.05); zmapright = zcMap(zcright,0.05);

% compute stereo disparities for left -> right images
border = fix(size(lap8,1)/2);
[dmap bestd bestc useDisp contrast] = compStereo(cleft8, cright8, -24, 24, 12, 12, 0.05, border);
% figure 1 displays the results; figure 2 displays the results with zero crossings overlayed on top
figure('Position', [50 50 1000 2000]), figure('Position', [50 50 1000 2000])
figure(1), subplot(2,4,1), minv = min(min(bestd)); maxv = max(max(bestd)); imshow(bestd, [minv maxv])
figure(2), subplot(2,4,1), newim1 = overlayZC(bestd-minv,zmapleft,5.0); imshow(newim1)
% compute best disparities for left -> right images within neighbourhood;
[bestmap corrLeft] = compStereoNeighbour(bestd, bestc, useDisp, -24, 24, 12, 12, border);
figure(1), subplot(2,4,5), minv = min(min(bestmap)); maxv = max(max(bestmap)); imshow(bestmap, [minv maxv])
figure(2), subplot(2,4,5), newim1 = overlayZC(bestmap-minv,zmapleft,5.0); imshow(newim1)
% compute stereo disparities for right -> left images
[dmapRight bestdRight bestcRight useDispRight contrast nodisp] = compStereo(cright8, cleft8, -24, 24, 12, 12, 0.05, border);
figure(1), subplot(2,4,2), minv = min(min(bestdRight)); maxv = max(max(bestdRight)); imshow(bestdRight, [minv maxv])
figure(2), subplot(2,4,2), newim1 = overlayZC(bestdRight-minv,zmapright,5.0); imshow(newim1)
% compute best disparities within neighbourhood from right to left
[bestmapRight corrRight] = compStereoNeighbour(bestdRight, bestcRight, useDispRight, -24, 24, 12, 12, border);
figure(1), subplot(2,4,6), minv = min(min(bestmapRight)); maxv = max(max(bestmapRight)); imshow(bestmapRight, [minv maxv])
figure(2), subplot(2,4,6), newim1 = overlayZC(bestmapRight-minv,zmapright,5.0); imshow(newim1)
```
% left-right checking
[bestBefLeft bestLeft bestBefRight bestRight] = leftRightChecking(bestmap, bestmapRight, 1);
figure(1), subplot(2,4,3), minv = min(min(bestBefLeft));, maxv = max(max(bestBefLeft));, imshow(bestBefLeft, [minv maxv])
figure(2), subplot(2,4,3), newim1 = overlayZC(bestBefLeft-minv,zmapleft,5.0);, imshow(newim1)
figure(1), subplot(2,4,7), minv = min(min(bestLeft));, maxv = max(max(bestLeft));, imshow(bestLeft, [minv maxv])
figure(2), subplot(2,4,7), newim1 = overlayZC(bestLeft-minv,zmapleft,5.0);, imshow(newim1)
figure(1), subplot(2,4,4), minv = min(min(bestBefRight));, maxv = max(max(bestBefRight));, imshow(bestBefRight, [minv maxv])
figure(2), subplot(2,4,4), newim1 = overlayZC(bestBefRight-minv,zmapleft,5.0);, imshow(newim1)
figure(1), subplot(2,4,8), minv = min(min(bestRight));, maxv = max(max(bestRight));, imshow(bestRight, [minv maxv])
figure(2), subplot(2,4,8), newim1 = overlayZC(bestRight-minv,zmapleft,5.0);, imshow(newim1)
% save all the results for fine scale processing
save('stereoMatchingCoarse.mat')

%% stereoMatchingFine.m
% specialized script to process a pair of stereo images of a collection of objects at different depths, using the results of coarse scale stereo matching as a guide
% load the results of coarse scale stereo matching; images are stored in variables 'left' and 'right'
load ('stereoMatchingCoarse.mat')
% convolve images with laplacian-of-gaussian operator, w = 4
lap4 = laplacian(4.0);, cleft4 = conv2D(left,lap4);, cright4 = conv2D(right,lap4);
% compute zero-crossings - zcleft and zcright store the slopes (magnitude of the gradient at the locations of the zero-crossings), zmapleft and zmapright store the contours (with sufficient slope)
zcleft = zeros2D(cleft4);, zcright = zeros2D(cright4);, zmapleft = zcMap(zcleft,0.05);, zmapright = zcMap(zcright,0.05);

% compute stereo disparities for left -> right images
[dmap bestd bestc useFront] = compStereoFront(cleft4, cright4, -6, 6, 6, 6, nodisp, bestLeft, border, 0.6, 0);
% figure 1 displays the results; figure 2 displays the results with zero crossings overlayed on top
figure('Position', [50 50 1000 2000]), figure('Position', [50 50 1000 2000])
figure(1), subplot(2,4,1), minv = min(min(bestd));, maxv = max(max(bestd));, imshow(bestd, [minv maxv])
figure(2), subplot(2,4,1), newim1 = overlayZC(bestd-minv,zmapright,5.0);, imshow(newim1)
% compute best disparities for left -> right images within neighbourhood
[bestmap corrLeft] = compStereoNeighbour(bestd, bestc, useFront, -6, 6, 6, 6, border);
figure(1), subplot(2,4,5), minv = min(min(bestmap));, maxv = max(max(bestmap));, imshow(bestmap, [minv maxv])
figure(2), subplot(2,4,5), newim1 = overlayZC(bestmap-minv,zmapleft,5.0);, imshow(newim1)

% compute stereo disparities for right -> left images
[dmapRight bestdRight bestcRight useFrontRight] = compStereoFront(cright4, cleft4, -6, 6, 6, 6, nodisp, bestRight, border, 0.6, 1);
% figure 1 displays the results; figure 2 displays the results with zero crossings overlayed on top
figure('Position', [50 50 1000 2000]), figure('Position', [50 50 1000 2000])
figure(1), subplot(2,4,2), minv = min(min(bestdRight));, maxv = max(max(bestdRight));, imshow(bestdRight, [minv maxv])
figure(2), subplot(2,4,2), newim1 = overlayZC(bestdRight-minv,zmapright,5.0);, imshow(newim1)
% compute best disparities for right -> left images within neighbourhood
[bestmapRight corrRight] = compStereoNeighbour(bestdRight, bestcRight, useFrontRight, -6, 6, 6, border);
figure(1), subplot(2,4,6), minv = min(min(bestmapRight));, maxv = max(max(bestmapRight));,
imshow(bestmapRight, [minv maxv])
figure(2), subplot(2,4,6), newim1 = overlayZC(bestmapRight-minv,zmapright,5.0);, imshow(newim1)

% left-right checking
[bestBefLeftFine bestLeftFine bestBefRightFine bestRightFine] = leftRightChecking(bestmap, bestmapRight, 1);
figure(1), subplot(2,4,3), minv = min(min(bestBefLeftFine));, maxv = max(max(bestBefLeftFine));,
imshow(bestBefLeftFine, [minv maxv])
figure(2), subplot(2,4,3), newim1 = overlayZC(bestBefLeftFine-minv,zmapleft,5.0);, imshow(newim1)
figure(1), subplot(2,4,7), minv = min(min(bestLeftFine));, maxv = max(max(bestLeftFine));,
imshow(bestLeftFine, [minv maxv])
figure(2), subplot(2,4,7), newim1 = overlayZC(bestLeftFine-minv,zmapleft,5.0);, imshow(newim1)
figure(1), subplot(2,4,4), minv = min(min(bestBefRightFine));,maxv = max(max(bestBefRightFine));,
imshow(bestBefRightFine, [minv maxv])
figure(2), subplot(2,4,4), newim1 = overlayZC(bestBefRightFine-minv,zmapleft,5.0);, imshow(newim1)
figure(1), subplot(2,4,8), minv = min(min(bestRightFine));,maxv = max(max(bestRightFine));,
imshow(bestRightFine, [minv maxv])
figure(2), subplot(2,4,8), newim1 = overlayZC(bestRightFine-minv,zmapleft,5.0);, imshow(newim1)

% save the results
save ('stereoMatchingFine.mat')

function [dmap bestmap bestcorr useDisp contrast nodisp] = compStereo(lt, rt, dmin, dmax, hsize, vsize, cthresh, border)
% dmin and dmax are minimum and maximum disparities
% measure = 0 (sum of absolute differences) or 1 (correlation) or 2 (sum of squared differences)
% patch for stereo matching covers neighborhood with 2*vsize+1 rows and 2*hsize+1 columns, centered on
% each location
% average absolute value of lt within above neighborhood is used as a measure of contrast - cthresh
% (between 0 and 1) is the fraction of maximum contrast needed to preserve local disparity calculation
% no disparities computed within border around image; dmap stores strength of all disparities, bestmap
% stores best disparity; bestcorr stores measure associated with best disparity

[rows cols] = size(lt);
numd = dmax-dmin+1; % total number of disparities considered
dmap = zeros(rows, cols, numd); % value indicating no disparity calculated
bestmap = nodisp*ones(rows,cols); % initialize to "no disparity" value
bestcorr = zeros(rows,cols); % number of locations in neighborhood
contrast = zeros(rows,cols); % stores contrast within neighborhoods
r1 = border+vsize+1; % range of rows where disparity is computed
r2 = rows-border-vsize;
if (dmin < 0) % range of cols where disparity is computed
  c1 = border+hsize+1-dmin;
else
  c1 = border+hsize+1;
end
if (dmax > 0)
  c2 = cols-border-hsize-dmax;
else
  c2 = cols-border-hsize;
end
% measure contrast within neighborhoods
for r = r1:r2
    for c = c1:c2
        contrast(r,c) = sum(sum(abs(lt(r-vsize:r+vsize,c-hsize:c+hsize))))/nsamp;
    end
end
thresh = cthresh*max(max(contrast));
useDisp = (contrast > thresh);  % indicates where disparity is computed

% use sum of absolute differences
for r = r1:r2
    if (mod(r,50) == 0)  % indicate progress, every 50 rows
        disp(['row ' num2str(r)])
    end
    for c = c1:c2
        if (useDisp(r,c))  % calculate disparity at this location
            for d = dmin:dmax
                dmap(r,c,d-dmin+1) = ...
                    (sum(sum(abs(lt(r-vsize:r+vsize,c-hsize:c+hsize)-...
                        rt(r-vsize:r+vsize,c-hsize+d:c+hsize+d)))))/nsamp;
            end
            % minimize sum of absolute differences
            [minv mini] = min(squeeze(dmap(r,c,:)));
            bestmap(r,c) = mini+dmin-1;
            bestcorr(r,c) = minv;
        end
    end
end

function [dmap bestmap bestcorr useFront] = compStereoFront(lt, rt, dmin, dmax, hsize, vsize, nodisp, best, border, percentFront, right)
% dmin and dmax are minimum and maximum disparities; patch for stereo matching covers neighborhood with 2*vsize+1 rows and 2*hsize+1 columns, centered on each location
% nodisp is the disparity marking regions whose disparities are not computed; best is a map of the best disparities computed at coarse scale; border is an area around the image whose regions are not computed
% percentFront is a percentage of disparities that should be included in the foreground
% right is a boolean, which is true if this stereo matching is done from right to left image
% dmap stores strength of all disparities, bestmap stores best disparity, bestcorr stores measure associated with best disparity, useFront stores the regions whose disparities are computed

[row col] = size(best);
useFront = zeros(row, col);
dispRange = zeros(row, col, 2);
minDisp = min(best(best > nodisp));
maxDisp = max(best(best > nodisp));
% takes first percentFront of the total disparities
if (right)
    limit = maxDisp-(maxDisp-minDisp)*percentFront
else
    limit = minDisp+(maxDisp-minDisp)*percentFront
end
% calculate objects within the limit and tag the neighboring points as well
if (right)
    for r = vsize+1:(row-vsize-1)
        for c = hsize+1:(col-hsize-1)
            if ((best(r,c) >= limit) & (best(r,c) > nodisp))
                useFront(r-vsize:r+vsize, c-hsize:c+hsize) = 1;
            end
        end
    end
else
    for r = vsize+1:(row-vsize-1)
        for c = hsize+1:(col-hsize-1)
            if ((best(r,c) <= limit) & (best(r,c) > nodisp))
                useFront(r-vsize:r+vsize, c-hsize:c+hsize) = 1;
            end
        end
    end
end
% mark the regions whose disparities are not computed
for r = vsize+1:(row-vsize-1)
    for c = hsize+1:(col-hsize-1)
        if (useFront(r,c) & best(r,c) <= nodisp)
            useFront(r,c) = 0;
        end
    end
end
dispRange(:,:,1) = best+dmin; % stores the min disparity to check
dispRange(:,:,2) = best+dmax; % stores the max disparity to check
for r = vsize+1:(row-vsize-1)
    for c = hsize+1:(col-hsize-1)
        neighborhood = best(r-vsize:r+vsize, c-hsize:c+hsize);
        % calculate the disparity only if the point is in the foreground
        if (useFront(r,c))
            % if minimum disparity of the neighborhood is smaller than disparity range, change the minimum
            minDisp = min(neighborhood(neighborhood > nodisp));
            if minDisp < dispRange(r,c,1)
                dispRange(r,c,1) = round(minDisp);
            end
            % do the same with maximum disparity
            maxDisp = max(max(neighborhood));
            if maxDisp > dispRange(r,c,2)
                dispRange(r,c,2) = round(maxDisp);
            end
        end
    end
end
bestmap = best;
bestcorr = zeros(row,col);
nsamp = (2*vsize+1)*(2*hsize+1); % number of locations in neighborhood
for r = vsize+1:(row-vsize-1) % use sum of absolute differences
    if (mod(r,50) == 0) % indicate progress, every 50 rows
        disp(['row ' num2str(r)])
for c = hsize+1:(col-hsize-1)
    if (useFront(r,c))  % calculate disparity at this location
        frontmin = round(dispRange(r,c,1));
        frontmax = round(dispRange(r,c,2));
        dmap = zeros(1,frontmax-frontmin);
        i = 1;
        for d = frontmin:frontmax;
            % compute sum of absolute differences
            dmap(i) = ...
                (sum(sum(abs(lt(r-vsize:r+vsize,c-hsize:c+hsize)-...
                    rt(r-vsize:r+vsize,c-hsize+d:c+hsize+d)))))/nsamp;
            i = i+1;
        end
        % find the minimum of the sum of absolute differences
        [minv mini] = min(dmap);
        bestmap(r,c) = mini+frontmin-1;
        bestcorr(r,c) = minv;
    end
end
end

function [bestmap corr] = compStereoNeighbour(bestdisp, bestcorr, useDisp, dmin, dmax, hsize, vsize, border)
% bestdisp stores best disparity and bestcorr stores measure associated with best disparity at each point.
% at each point, find the disparity with the best correlation in the neighbourhood with size of 2*hsize+1 columns and 2*vsize+1 rows. After the best diparity is found, find all the disparities within range cthresh in the neighbourhood region and find the average of these disparities.
% This average becomes the disparity at the point. returns map of best disparities at each point.
cthresh = 2;
bestmap = bestdisp;
[rows cols] = size(bestdisp);, corr = zeros(rows,cols);
r1 = border+vsize+1;, r2 = rows-border-vsize;  % range of rows where disparity is computed
if (dmin < 0) % range of cols where disparity is computed
    c1 = border+hsize+1-dmin;
else
    c1 = border+hsize+1;
end
if (dmax > 0)
    c2 = cols-border-hsize-dmax;
else
    c2 = cols-border-hsize;
end
for r = r1:r2
    if (mod(r,50) == 0); % indicate progress, every 50 rows
        disp(['row  ' num2str(r)])
    end
    if (useDisp(r,c))
        % find the disparity with the best correlation within a region
        region = bestcorr(r-vsize:r+vsize,c-hsize:c+hsize);
        use = useDisp(r-vsize:r+vsize,c-hsize:c+hsize);
        disparity = bestdisp(r-vsize:r+vsize,c-hsize:c+hsize);
        bestcorr(r,c) = disparity;
    end
end
% mark regions whose disparities we are not calculating
region(~use) = 1000;
% find a point with best correlation (best correlation = minimum)
[minv mini] = min(region);
[minv2 mini2] = min(min(region));
% store the disparity of that point
d = disparity(mini(mini2), mini2);
% find a range of disparities centered around the disparity with best correlation
sumTotal = 0;
numTotal = 0;
for t = d-cthresh:d+cthresh
    temp = disparity(disparity==t);
    sumTotal = sumTotal + sum(temp);
    [num col] = size(temp);
    numTotal = numTotal + num;
end
% the average of the disparities within the range becomes the disparity at this point
bestmap(r,c) = sumTotal/numTotal;
corr(r,c) = minv2;
end
end
end

function [bestBefLeft bestLeft bestBefRight bestRight] = leftRightChecking(bestmap, bestmapRight, range)
% bestmap = result from stereo matching algorithm matching left image to right
% bestmapRight = result from stereo matching algorithm matching right image to left
% range = threshold range: if the sum of disparities of bestmap and bestmapRight at a single point is
greater than this range, the point is considered too different (=half-occluded point)
% bestBefLeft = finds all the half-occlusion regions, matching left to right
% bestLeft = replaces the disparities of the half-occlusion regions with left neighboring points
% i.e if point (x,y) is half-occluded, its disparity is replaced by that of the point (x-1,y)
% bestBefRight = finds all the half-occlusion regions, matching right to left
% bestRight = replaces the disparities of the half-occlusion regions with right neighboring points
% i.e if point (x,y) is half-occluded, its disparity is replaced by that of the point (x+1,y)
[rows cols] = size(bestmap);
minv = min(min(bestmap));
bestBefLeft = zeros(rows,cols);
for r = 1:rows
    for c = 1:cols
        % at each point in the left image, get the disparity
dispLeft = bestmap(r,c);
        if dispLeft > minv
            % then take the disparity of the point in the right image at the corresponding location
            dispRight = bestmapRight(r, c+round(dispLeft));
            dispRight = -dispRight;
            % check if the left and right disparities are within a range of each other
            if (dispLeft >= (dispRight-range)) && (dispLeft <= (dispRight+range))
                bestBefLeft(r,c) = dispLeft;
            else
                bestBefLeft(r,c) = -100;
            end
        end
    end
end

else
    bestBefLeft(r,c) = dispLeft;
end
end

bestLeft = bestBefLeft;
for r = 1:rows
    for c = 2:cols
        if (bestLeft(r,c) == -100)
            bestLeft(r,c) = bestLeft(r,c-1);
        end
    end
end

% left-right checking for the right side
[rows cols] = size(bestmapRight);
minv = min(min(bestmapRight));
bestBefRight = zeros(rows,cols);
for r = 1:rows
    for c = 1:cols
        % at each point in the left image, get the disparity
        dispRight = bestmapRight(r,c);
        if dispRight > minv
            % then take the disparity of the point in the left image at the corresponding location
            dispLeft = bestmap(r, c+round(dispRight));
            dispLeft = -dispLeft;
            % check if the left and right disparities are within a range of each other
            if (dispRight >= (dispLeft-1)) && (dispRight <= (dispLeft+1))
                bestBefRight(r,c) = dispRight;
            else
                bestBefRight(r,c) = -100;
            end
        else
            bestBefRight(r,c) = dispRight;
        end
    end
end

bestRight = bestBefRight;
for r = 1:rows
    for c = cols:-1:1
        if bestRight(r,c) == -100
            bestRight(r,c) = bestRight(r,c+1);
        end
    end
end
function lap = laplacian(w)
% lap = laplacian(w); returns a 2D matrix of samples of a Laplacian-of-Gaussian function with
% diameter w for the central positive region (w = 2*sqrt(2)*sigma). The returned value is a 2D square matrix
% of size 4*w+1 and the x and y values range from -2*w to 2*w
sigma = w/(2*sqrt(2));
range = fix(2*w);
lap = zeros(2*range+1, 2*range+1);
s2 = 1.0/(sigma^2);
for x = -range:range
  for y = -range:range
    dist2 = x^2 + y^2;
    lap(x+range+1, y+range+1) = (2-s2*dist2)*exp(-0.5*dist2*s2);
  end
end

function result = logTransform(convIm)
% transforms values of input convIm using natural log, preserving sign
% used to transform the convolutions of images with Laplacian operator
result = zeros(size(convIm));
% note: very small values, where abs(convIm) < 1, are set to 0
result(convIm >= 1) = log(convIm(convIm >= 1));
result(convIm <= -1) = -1*log(abs(convIm(convIm <= -1)));

function newImage = overlayZC(image, zcMap, scale)
% newImage = overlayZC(image, zcMap)
% given an input 2D 8-bit image and a 2D 8-bit matrix of the locations of zero-crossings, creates and returns
% a 2D 8-bit image of the same size, with white zero-crossing contours obtained from the input zcMap
% superimposed on a low-contrast version of the original image
if (nargin < 3)
  scale = 0.75;
end
newImage = uint8(scale*double(image));
newImage(zcMap == 255) = 255;

function zc = zeros2D(conv)
% zc = zeros2D(conv)
% returns a 2D matrix of the slopes of the zero-crossings of the input 2D matrix of convolution results. A
% value of 0 is placed in locations that do not correspond to zero-crossings
[xdim ydim] = size(conv);
zc = zeros(xdim,ydim);
for x = 2:xdim
  for y = 2:ydim
    % check if adjacent convolution values have opposite sign
    if ((conv(x,y)*conv(x-1,y)) < 0) || ((conv(x,y)*conv(x,y-1)) < 0)
      dx = conv(x,y)-conv(x-1,y);    % calculate slope of
      dy = conv(x,y)-conv(x,y-1);    % zero-crossing
      zc(x,y) = sqrt(dx^2 + dy^2);
    end
  end
end