CONNECTIONIST MODELS OF STEREO VISION

Charles Vernon Stewart

Computer Sciences Technical Report #776

June 1988

CONNECTIONIST MODELS OF STEREO VISION

by

CHARLES VERNON STEWART

A thesis submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN-MADISON

1988

Abstract

In this thesis we investigate the use of massively parallel computational models to solve the problem of establishing correspondence between stereo pairs of images. We present a number of steps toward improving matching (correspondence) results. First, we analyze and reformulate a number of important matching constraints, including uniqueness, coarse-to-fine multiresolution, fine-to-coarse multiresolution, detailed match, figural continuity and the disparity gradient. This reformulation yields *locally defined constraints*, i.e. constraints are functions of individual matches or pairs of matches. Second, we define an algorithm that integrates the influence of these constraints *cooperatively* and *in parallel* using the General Support Principle. This principle states that except for uniqueness, *only positive* constraint influences are employed. Third, we implement this General Support Algorithm using a connectionist network. Such a network allows the constraints to be integrated naturally in a parallel, relaxation computation. The resulting connectionist network has been simulated and tested (including a simulation on a shared-memory multiprocessor). It generally produces a high percentage (93-99%) of correct matching decisions. In addition it usually requires a small number of network iterations to resolve most of the ambiguities.

In examining the results of the General Support Algorithm we find that the errors in matching were generally caused by circumstances that can not be explicitly addressed using locally defined and integrated constraints. These include incorrect

matches near occlusions, ambiguous matches in occluded periodic regions of the images, and missing or erroneous matches due to significant structural variations between the images. We have developed two different approaches to overcoming these problems. The first incorporates additional information available from other image computations to identify problematic regions or to provide additional information that resolves these ambiguities. The second approach employs a third image in matching, and allows cooperative matching between two pairs of images. In addition to using the original General Support Algorithm, it employs new trinocular matching constraints. These are trinocular versions of uniqueness and the disparity gradient.

Acknowledgements

In terms of my intellectual development and in terms of the contents of this thesis I owe a great debt to Professor Chuck Dyer. His patience, guidance and encouragement were beyond value throughout my career as a graduate student. Thank you to the members of my committee, Professors Len Uhr, Gregg Oden, Jude Shavlik and Roland Chin for their thoughtful comments both on the dissertation and on the work leading up to it. I also wish to acknowledge the support of friends both within and outside the Computer Sciences Department, especially Harry, Marty, Mitch and Will.

My family, including both the Stewarts and the Elliots, has been of great assistance to me in my endeavors over the years. This is especially true of my parents, Bob and Ginny Stewart. Finally, and most importantly, I would like to thank my wonderful wife, Lou Ann. I could not have made it through this adventure without her love, support, and friendship. I am looking forward to many new adventures that we may share over the years.

Table of Contents

CHAPTER 1

Introduction

Moving about and manipulating objects in a three-dimensional environment requires three-dimensional understanding of the environment. Unfortunately, visual images, a natural source information about the world, are inherently two-dimensional. This difficulty may be overcome through the use of stereo vision. That is, two cameras are employed observing the same scene. Based on the differences between the appearance of scene features in the two images, the three-dimensional locations of those features may be inferred. An example of this is shown in Figure 1.1.

The main difficulty in using stereo vision effectively is deciding which features of the images correspond to each other. One approach to this correspondence problem is to recognize large objects in the two separate images and then match the features of these objects. However, automatically recognizing objects is a poorly understood and unsolved problem. Some algorithms attempting to solve it assume that the results of stereo vision are available as input. Thus, it is desirable to construct a stereo matching algorithm that works without prior recognition of objects. In addition, work by Julesz[35] has shown that humans can actually solve this matching problem before recognition. In his research, he created random-dot stereograms to test human stereo capabilities. In constructing such a stereogram, a three-dimensional scene is hypothesized and points in the scene are randomly sprayed either dark or light. The resulting scene is projected onto two separate images. Each resulting image appears

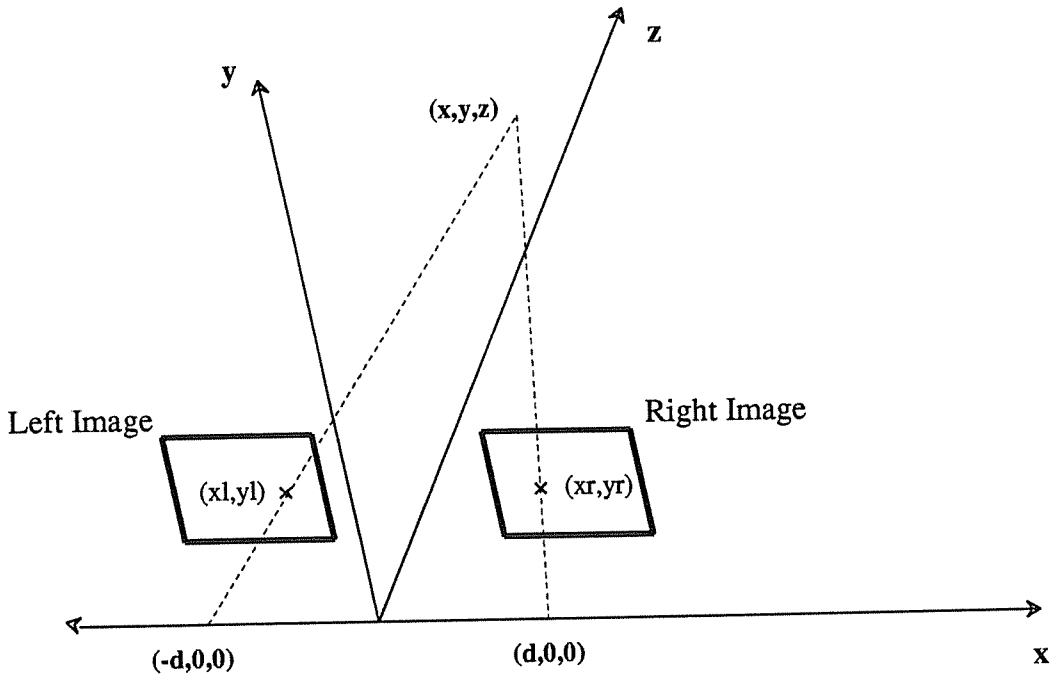


Figure 1.1. Camera positions for stereo vision. A point (x, y, z) in the world is projected onto locations in each of the images. Based on the difference between these locations, the position of that point in the world can be recovered.

to be a mass of arbitrary dots as shown in Figure 1.2. However, by matching the two images the three-dimensional surface can be recognized. Thus, although there is no structural information in the images, when they are combined through matching structure appears easily.

Because of the need for matching prior to recognition and because it has been shown that humans can perform this matching operation, there has been much interest in automatically solving the stereo matching problem. Random-dot stereograms provide a simple context for describing the general approach taken in many such algorithms, including the algorithms described in this thesis. In these algorithms,

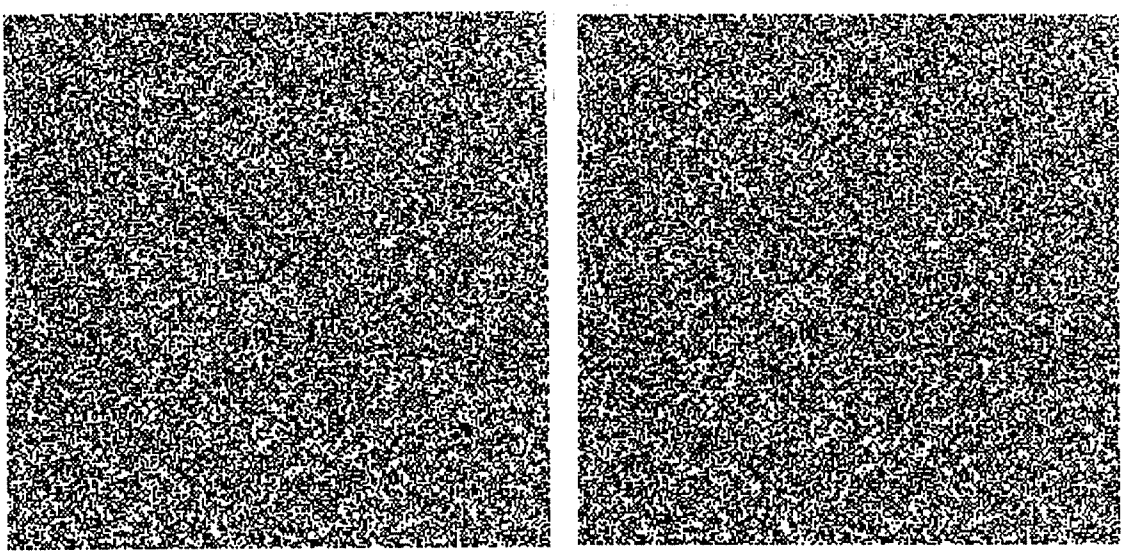


Figure 1.2. An example random-dot stereogram. See page 2 for discussion. (Some information in the images may be lost in copying this thesis.)

individual points are identified in one of the images, and potentially corresponding points are located in the other image. These points may be the individual dots that appear in the stereogram, or they may be other simple features of the images. Once the candidate matches for each point are identified, computational processes are used to select the correct matches, usually at most one for each point. The different computational processes employ assumptions about the spatial relations between the candidate matches in order to select the appropriate ones. For example, one assumption states that matches for nearby points should indicate similar depth values since they are likely to be from the same surface in the scene. Thus, when there is a pair of such matches for nearby points, the likelihood that each of the matches will be accepted as correct increases. This and a number of other similar *constraints* have

been used by many stereo matching algorithms.

In general, many solutions to the stereo matching problem have been proposed in the computational vision literature. These algorithms are fairly successful, but they also have a number of problems. First, they tend to produce spurious matches at certain important locations in images, primarily at significant occluding boundaries. These boundaries represent discontinuities in depth in the scene. Because the images represent different views of the scene, there will be points in one image near occluding boundaries that do not appear in the other. Existing stereo algorithms often find incorrect matches for these points.

A second problem with stereo algorithms is their sensitivity to noise in the images. This includes finding matches for extraneous points, and not finding matches for other points because the corresponding point in the other image is missing. More significant differences between the images cause even greater problems.

A third problem with prior algorithms is that they often have difficulty producing correct matches when image features include repetitive patterns. In these circumstances there is not enough variation in the images to distinguish between a number of different possible matches for an edge.

A final difficulty is that existing algorithms tend to be slow. In general, the quality of matching might be improved by using more information in the matching process. Unfortunately, with sequential computations, the use of this extra information makes the efficiency problem even worse.

In this thesis we address these problems in stereo matching. First, we present a careful analysis of the constraints used by stereo algorithms to select valid matches. These constraints are generally based on assumptions about surfaces in the world and about the imaging process. Our analysis exposes weaknesses in the assumptions motivating the constraints, in their definitions, and in their actual use. This results in a new formulation of some of the constraints, and leads to the conclusion that none of the constraints proposed thus far is completely sufficient for stereo matching. One possible way to overcome the difficulties with the individual constraints is to integrate their influences. In doing so, many of the weaknesses of the individual constraints may be overcome if the different constraints are not susceptible to the same problems.

The need for integrating multiple constraints leads to our development of the General Support Algorithm for stereo matching. The algorithm organizes the influence of constraints using the General Support Principle. The principle states that, with only one exception, all the constraints used in stereo matching provide *only positive influence* in selecting the valid matches. The algorithm is implemented using a *connectionist network*. This provides a natural model for constraint integration by allowing the constraints to interact cooperatively and in parallel to select the valid matches. The features of a connectionist network that allow this to be achieved include the use of a large number of simple computing elements and a massive interconnection structure between the elements. The interconnections between the nodes propagate information based on the constraint definitions.

In the General Support Algorithm for stereo matching, constraints are defined as pairwise interactions between potential matches (that is, the constraints are *locally defined*). The interaction of these constraints will usually force globally consistent matching results to emerge. However, there are some situations in which local matching is not sufficient and errors occur. These errors arise because the locally-defined constraints can not explicitly detect and overcome the problem situations. These correspond to some of the difficult circumstances described above. Most prominent among these are (1) occluding boundaries representing large changes in disparity, (2) large periodic regions that are partially occluded, and (3) significant structural differences between the images. Even in these circumstances the General Support Algorithm performs fairly well. However, it does not overcome all the matching difficulties associated with these problems.

To solve the remaining problems in stereo matching, we propose two different types of solutions. Both of them are added to the existing General Support Algorithm. The first employs specific mechanisms for overcoming each problem type. One example of this is to include average-disparity detectors to locate occluding boundaries. A second example is to incorporate information from other types of image computations to help resolve the ambiguity in periodic regions.

The second approach to overcoming the remaining problems in stereo matching involves the use of a third camera positioned above the left camera. The General Support Algorithm is used for matching the left and right images, and the left and top images. In addition, new constraints are proposed and implemented between these

two General Support Algorithm networks. These new constraints help the trinocular algorithm to overcome many of the remaining matching problems. This results in a matching algorithm that improves the results of the original algorithm. It also offers an improvement over existing three camera matching algorithms.

Finally, one of the problems in studying large scale connectionist networks is the difficulty of building practical simulations of them. To facilitate the study of the General Support Algorithm we present an implementation of its simulation on a shared-memory multiprocessor. Through careful design, the processing of each individual node in the network can be made nearly independent of the other nodes, and therefore near linear speedups (in the number of processors) can be obtained.

In the remainder of this chapter we define the stereo matching problem in detail, briefly describe the use of connectionist networks, and outline the chapters of the thesis.

1.1. Stereo Problem

In general there are several stages in the binocular stereo problem. The first is determining the geometry of the imaging system. This includes both the camera properties and the relative position and orientation of the cameras. Usually, the camera properties are assumed to be known. The other information may be assumed as well. The second step is to detect features in each image for matching. These may be anything from individual pixels to segmented regions. The third step involves matching these features. This is the *matching problem* (also called the

correspondence problem) and is the focus of our work. Finally, the results of matching are interpreted to obtain depth information.

The standard imaging geometry for the stereo problem is shown in Figure 1.1. The cameras are assumed to have point lenses and the images are formed through perspective projection. The lines of sight of the cameras are normal to the center of the image plane, they intersect the focal point, and are parallel to the z-axis in the $x-z$ plane. This model of imaging geometry, called the *parallel model*, is common to most of the stereo matching algorithms that appear in the literature.

In the discussion in the thesis we refer to **points** as general matching primitives. These may be pixels, edges, corners, contour segments, etc. Our only requirement is that they have a definite physical location. In the General Support Algorithm we will use edges detected at a number of different resolutions as the primary matching primitive. Reasons for this choice are discussed in Chapter 2.

One of the important steps in the stereo process is identifying the points that may match. Given a point (x_0,y_0) in one image, the **epipolar scanline** for (x_0,y_0) is the line in the other image containing candidate match points for (x_0,y_0). Assuming that the image plane is parallel to the $x-y$ plane as shown in Figure 1.1, the epipolar scanline for (x_0,y_0) is the line (x,y_0) in the other image. Thus the epipolar scanline is *parallel* to the x-axis for each point. This is one of the main advantages of the parallel model of imaging geometry.

There are some problems involving the use of epipolar scanlines when the matching primitives have an orientation component. Because oriented primitives tend

to occur along contours, those primitives oriented *parallel* to the epipolar scanline are problematic. There is likely to be a large number of possible matches for each primitive with little to distinguish between the matches. Also, slight errors in the position of these segments cause no matches to be found at all. Hence, many algorithms ignore the matching of such points (horizontal edges in the parallel imaging model). In the General Support Algorithm we only match a horizontal edge when it is relatively isolated from other horizontal edges. More complete matching of horizontal edges requires either non-local matching mechanisms or trinocular stereo.

Another feature of the imaging geometry has an important effect on the matching process. The **baseline** is the distance along the x-axis between the focal points of the two cameras. In Figure 1.1 this distance is $2d$. For small baselines, the appearance of the images is nearly the same and thus matching is easier. However, the depth information obtained is not very accurate. On the other hand, large baselines allow more accurate depth interpretations but increase the difficulty of matching. We assume no particular value for the baseline except that it is large enough to produce significant differences in the appearance of the images.

For a given match, the **disparity**, the camera parameters and the baseline width indicate the distance to the corresponding point in the scene. The disparity is defined to be the distance between the appearance of a scene point in the two images. In the parallel model, the disparity determines the value of the z-coordinate of the scene feature corresponding to the match as follows. Let (x_l, y) be the left image point and let (x_r, y) be the right image point; the distance to the scene point is given by

$$z = \frac{2\,d\,f}{(x_l - x_r)}$$

where f is the focal length of the lens, and $2\,d$ is the width of the baseline as above. Similar equations define the x and y coordinates of the scene point. Note that points closer to the images produce larger disparities.

In describing the matches in the General Support Algorithm we will make a distinction between **potential matches**, **candidate matches**, and **valid matches**. Potential matches are the possible matches determined by the limitations of the imaging geometry, including the epipolar scanline assumption and the range of allowed disparities. Thus, the potential matches are defined independent of any image data. In the computation of the General Support Algorithm there are as many computing elements in the connectionist network as there are potential matches. The candidate matches are the potential matches for a given pair of images that correspond to a pair of actual edges, one from each image. Thus, the candidate matches represent a subset of the potential matches defined by the image data. Finally, the valid matches are those selected by the algorithm as correct.

In the matching process, **constraints**, based on information derived from the images, are used to help select the valid matches from the candidates. A constraint may either be a function of a single candidate match or a relation between candidate matches. In the former case it is a statement about the compatibility of certain properties of the two points in a match. In the latter case it is a statement about compatibility within a group of two or more matches. Constraints are usually based on assumptions about the objects in the scene or about the results of the imaging

process. In developing the General Support Algorithm we carefully analyzed these constraints and the assumptions behind them. Using this analysis we developed our algorithm so that it would integrate these constraints and improve the matching results.

1.2. Connectionist Models

Connectionist models are motivated by the computing organization of the brain.[15,62] In a connectionist network there are simple computing elements (called nodes) and a massive interconnection network among these nodes. An example of a very simple connectionist network is shown in Figure 1.3. The nodes of the network each compute simple activation functions based on their summed input from other nodes. These activations may be a simple binary value, one of a larger set of discrete

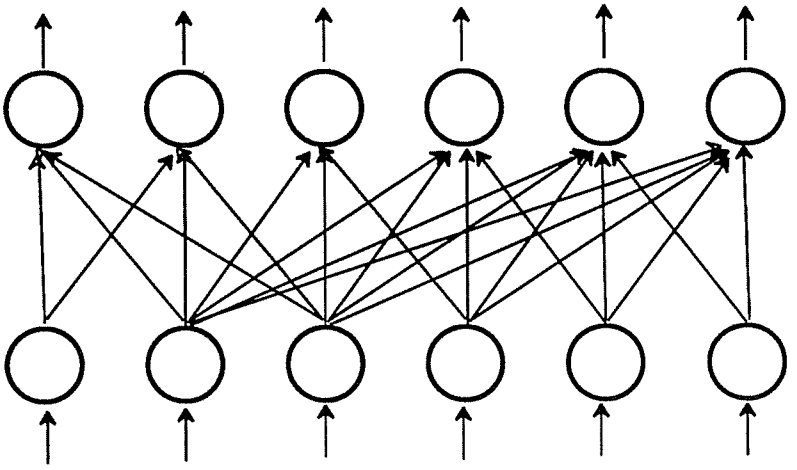


Figure 1.3. Simple example of a connectionist network. Nodes represent simple activation functions and arcs represent weighted outputs from one node to another.

values, or from a continuous range of values. This is done simultaneously by each node. There are many weighted connections between the nodes in a connectionist network. Each node outputs its activation to other nodes via its outgoing connections. A connection multiples the given activation by its weight and provides the result as input to the node at the end of the connection.

In Figure 1.3 the nodes of the network are structured hierarchically, with the outputs of one layer provided as input to the next. In the network defined for stereo matching, however, the computation involves more feedback between various nodes. In our network, nodes represent potential matches, and connections between nodes represent stereo matching constraints. The overall computation of the network is organized around iterative changes in the activations of the nodes. This continues until network settles on a solution. This style of computation is similar to relaxation algorithms.[12,32]

1.3. Organization of the Thesis

The major part of the thesis develops the General Support Algorithm. Specifically, Chapter 2 reviews prior work on stereo matching and related work on connectionist models. The discussion of stereo matching is continued in more detail in Chapter 3 where the constraints that we are most in interested are carefully analyzed. This analysis leads to a reformulation of a number of the constraints as well as the conclusion that no individual constraint is sufficient to completely solve the stereo matching problem. Chapter 4 uses the analysis of the constraints and the parallel relaxation computation of connectionist models to develop the General

Support Algorithm. The simulation and testing of this algorithm are described in Chapter 5. The experimental results demonstrate the success of the General Support Algorithm in integrating multiple constraints to improve matching, but they also highlight some remaining problems that the algorithm is unable to solve. The parallel simulation of the General Support Algorithm on a shared-memory multiprocessor is described in Chapter 6. Chapters 7 and 8 present two different approaches to overcoming the remaining problems in stereo matching. In Chapter 7 algorithms are developed for using non-local mechanisms and other intrinsic information sources to identify and overcome situations that the General Support Algorithm does not address explicitly. Chapter 8 describes the incorporation of a third camera, along with additional matching constraints based on the relation between the three cameras, into the General Support Algorithm. A summary of the main results and conclusions is presented in Chapter 9.

CHAPTER 2

Background

This chapter reviews the research literature related to our work. Section 2.1 discusses the stereo matching literature. Section 2.2 describes the use of connectionist models and, in particular, those used for vision.

2.1. Stereo

A significant amount of work has been done in stereo vision. This section reviews the stereo literature as it relates to our work. The review is intended to be fairly complete. However, some prior research pertinent to nonlocal mechanisms for matching is delayed until Chapter 7, and algorithms using more than two cameras in stereo matching are discussed in Chapter 8.

The next five subsections are organized as follows. Section 2.1.1 summarizes stereo approaches based on the types of primitives used for matching. This gives a broad overview of the algorithms. Section 2.1.2 outlines algorithms for edge-based matching that employ some of the constraints used in the General Support Algorithm. Detailed analysis of these constraints, independent of their use in specific algorithms, is deferred until Chapter 3. Section 2.1.3 reviews other algorithms employing multiple constraints. Section 2.1.4 discusses cooperative, connectionist style matching techniques. Finally, Section 2.1.5 highlights work on other problems that are closely related to binocular stereo vision. Examples of such work are combining

motion and stereo, matching using more than two images, and obtaining depth information from range data.

2.1.1. Types of Primitives

Stereo matching algorithms can generally be categorized by the types of primitives that they use as input to the matching process. This ranges from point-based matching to matching of sophisticated skeletal structures. At the low end are pixel or intensity based matching algorithms. These tend to match intensity profiles of the two images.[37,66] Some algorithms are designed specifically for use in matching random-dot stereograms.[69] Other methods, more common than point-based matching, use an operator to select points in areas of high interest and seek a corresponding point in similar regions of the other image.[7,23] Barnard and Fischler's survey of stereo algorithms highlights a number of these.[8] These algorithms tend to fail when the image data are regular and when there are intensity profile differences between the images, especially at occluding boundaries. In the case of occluding boundaries, the differing views of the occluded surface can produce significant differences between the two images.

The most common algorithms are those that match edges. Unless they arise from noise, edges may be attributed to physical features of the scene. The edges arise from specific underlying scene events. With pixels or other simple features it is difficult to establish correspondence between them and specific scene events, and therefore it is difficult to assign matches between pixels in a pair of images.[43] The major problem with matching edges is the abundance of candidate matches for each edge. This

implies the need for constraints to assist in selecting the valid matches from among the candidates.

To avoid the problem of disambiguating numerous candidate matches, some researchers have attempted to match higher level primitives. These include matching of skeletal structures,[10] and image segment matching.[18,24,49] These more sophisticated matching structures have more similarity criteria, and therefore, each primitive has fewer candidate matches. Unfortunately, there are two major problems with matching such structures. First, they are difficult to reliably extract from images. Second, slight appearance differences between the two images often cause significant differences in the structures that are found. This is most common near occluding boundaries. Because of these problems, most algorithms have used simpler primitives in matching.

2.1.2. Related Algorithms

Algorithms that rely on edges or simpler primitives require constraints to select the valid matches from among the candidates. In this section we review a number of edge-based matching algorithms. The discussion is organized by the type of constraints used, but our purpose is to outline the algorithms. In Chapter 3 we carefully analyze some of the constraints independent of their implementation.

Some of the earliest explicitly specified constraints were given by Marr and Poggio.[45] In their initial model for stereo vision, they proposed compatibility, continuity and uniqueness as the constraints used to disambiguate matches. Compatibility states that only edges with similar properties should match. Most

frequently the requirement is that the edges should have roughly similar orientations. The uniqueness constraint states that each point has at most one match in the other image. Every edge based matching algorithm uses some form of both uniqueness and compatibility. The continuity constraint states that nearby image points usually arise from the surface in the world, and so they should appear with nearly equal disparities. These three constraints were combined into a cooperative matching algorithm that worked fairly successfully on random-dot stereograms. The continuity constraint has been used directly by a number of algorithms,[1,7] including a recent Connection Machine algorithm requiring approximately one second to achieve matching results.[13] Also, as we will see shortly, a variety of other useful constraints have been derived from it. Section 2.1.4 discusses the use of cooperative algorithms in more detail.

A subsequent model by Marr and Poggio introduced the use of coarse-to-fine matching and vergent eye movements in a computational model.[43] Coarse-to-fine matching (also called multiresolution matching) starts with edges detected at coarse levels of resolution. It uses the results of matching at each successive level to realign the images so as to restrict the candidate matches at the next finer resolution. This results in at most two candidate matches for each edge. The correct match is then selected cooperatively. Matching errors at one level are detected when a large proportion of edges at the next finer level have no candidate matches.

Multiresolution matching has been used directly in a number of stereo algorithms.[19,30,60,67,79,81] These algorithms all use the results of coarser level matching to limit the range of possible disparities for finer level matches.

Unfortunately, as we will see in Chapter 3, there are a number of problems in depending on multiresolution to indicate the valid matches.

Mayhew and Frisby developed a competing theory to those of Marr and Poggio.[47] They based their ideas on the "Binocular Raw Primal Sketch" conjecture. This postulates that stereo matching should take place concurrently and in cooperation with the extraction of the raw primal sketch. (The raw primal sketch is an intermediate level of representation postulated by Marr.[45]) From this assumption they developed the "figural continuity" constraint. It states that edges in an image that appear to form a contour should match edges along a similar contour in the other image. When such matches are found they indicate that the contours in each image are valid. A number of related stereo matching algorithms have been developed that use "figural continuity" in a variety of ways. Some algorithms have used it as justification for using contour segments as matching primitives,[1,49,50] while others have used this to define algorithms where contours match if a given percentage of the edges along the contour match.[68] Finally, some approaches use it to eliminate edges from matching if they are not part of a minimum length contour.[19] Unfortunately, none of these, including the original implementation, implements the conjecture that edge matching work cooperatively with contour extraction. The approaches postulate feature extraction and matching as separate processes. Thus, the segment based matching derived from figural continuity is susceptible to problems arising from differences in contour segments due to noise and appearance variations between the two images.

Some recent approaches have extended the continuity constraint to develop local support algorithms for stereo matching. These methods have used either the disparity gradient,[58,73,74] or a Gaussian support model.[59] Both of these algorithms collect support for each match from its neighboring candidate matches based on the support model. This support computation is usually done only once. A candidate match having the greatest local support of the alternative matches for both of the edges of that match is selected. Uniqueness allows all other candidates for both edges to be eliminated. This is repeated until the valid matches have been selected. These algorithms tend to work fairly successfully. However, selecting matches in this manner is sensitive to noise and to significant occlusions. Errors arise when edges hidden in one image match noise edges in the other image. These matches often receive a substantial amount of support. Other problems with these algorithms, specific to the definition of the constraints, are discussed in the next chapter.

A different type of approach employs matching constraints in the context of dynamic programming algorithms.[4,5,41,53,61] These algorithms depend on having known epipolar scanlines. They work on each scanline separately, matching the edges from left to right within the scanline. Thus, they rely heavily on uniqueness and the lack of reversals. A reversal occurs when two edges in a row in one image match edges in the same row in the other image, but in reverse order. The constraint that no reversals are allowed is called the *ordering constraint.* It is violated only in situations such as transparent surfaces or narrow surfaces in front of wider surfaces. This limiting assumption allows the dynamic programming algorithm to be tractable.

The cost function used in dynamic programming is critical to its success. One of the earliest approaches, that of Baker and Binford,[5] used detailed edge compatibility, length of inter-edge segments, and coarse-to-fine techniques in their cost function. They handled occlusions and noise by allowing edges to go unmatched. After using dynamic programming to select the matches along each row, they use compatibility techniques between rows to eliminate errors. A significant extension to this was proposed by Ohta and Kanade.[53] Their algorithm ordered contour segments within each image and then used dynamic programming to match the segments. Thus, contour continuity was directly incorporated into the dynamic programming algorithm.

Dynamic programming techniques have tended to work fairly well, although they have somewhat restrictive assumptions. They depend on a strict form of uniqueness, on the ordering constraint and on having the images in register. The latter ensures that epipolar scanlines are horizontal. It is also not clear how well they adapt to repetitive image structures, noise and significant occlusions. Finally, since dynamic programming is essentially a sequential operation, the stereo algorithms relying on it tend to be slow. This latter criticism is somewhat mitigated by the recent development of specialized hardware that matches simple images in a few seconds.[55]

2.1.3. Algorithms Integrating Multiple Constraints

A number of researchers have employed multiple constraints in solving the matching problem. Baker and Binford's dynamic programming algorithm,[5] discussed above, employed several edge similarity constraints and multiresolution techniques in

their cost function, and used contour constraints in a final phase to eliminate erroneous matches. Eastman and Waxman[14] had some success integrating surface extraction with stereo matching. However, their algorithm tends to be slow and is susceptible to additional problems in using surface approximation techniques. Liu, Eastman and Davis[42] developed and analyzed a group of algorithms that used different constraints at different points in the matching process. Grimson[19] incorporated similar ideas in his improvement of the Marr-Poggio multiresolution matching algorithm.

There are a number of problems with the way most of these integration algorithms have employed multiple constraints. Most importantly, the constraints are used sequentially. Thus, each constraint is used to correct the matching errors of prior stages. Usually, once a match is rejected it can not be recovered. The General Support Algorithm presented here will offer an alternative. It integrates the constraints cooperatively, and in *parallel.* This allows the constraints to interact to determine the best matches.

2.1.4. Other Cooperative Matching Techniques

Starting with the work of Marr and Poggio,[45] a number of cooperative matching algorithms have been proposed.[7,40,69] Cooperative techniques simultaneously maintain many candidate matches and use relationships between matches for different edges to strengthen some of the matches and weaken others. This is repeated until the invalid matches are eliminated. Szeliski has studied several cooperative algorithms in matching random-dot stereograms.[70] One major problem with these techniques is that they incorporate only a few constraints into the matching process. They usually

employ a more or less direct implementation of the three constraints proposed by Marr and Poggio.[45] Thus, these algorithms do not incorporate other information that may be useful in matching, and are unable to detect situations in which the constraints fail to provide accurate information.

Marr criticizes approaches that rely too heavily on cooperative algorithms.[45] His argument is that they converge too slowly. This is a valid criticism for many cooperative algorithms, and is useful in judging any cooperative algorithm. Feldman and Ballard have suggested that recognition must take place in 100 or fewer processing iterations of a connectionist vision network.[15] A stereo algorithm, making up only a small part of such a network, should require many fewer iterations.

Regularization techniques have also been applied to develop cooperative matching algorithms.[46] The main idea has been to develop an energy equation that describes the image compatibility and disparity smoothness requirements. Disparity smoothness is represented by continuity in the spatial derivative of the disparity and low magnitudes of this derivative. For a given pair of images the possible disparities for the edges in one image are related in a network defined by the regularization equation. The energy of the network describes the amount of conflict between the disparities chosen for the edges. The energy minimum is therefore the set of matches representing the smoothest set of disparities, and the matching algorithm is a parallel search for this energy minimum. Unfortunately, the assumption of smooth disparity changes is violated at surface boundaries, so the original stereo energy equation is not effective in general. Recently, researchers have been attempting to incorporate

discontinuities in disparity directly into the regularization equations.[13]

2.1.5. Related Problems

There are a number of other vision problems related to binocular stereo. One important area of research studies the use of more than two cameras to obtain depth information.[2,33,51,54,57,75] Most commonly, these algorithms use three cameras. In general, the best approaches to trinocular stereo involve sophisticated matching between pairs of images. Thus, while these approaches can improve the results of stereo matching, they do not eliminate the matching problem. We discuss this further in Chapter 8 in presenting the extension of the General Support Algorithm to trinocular matching.

A second problem related to binocular stereo involves combining stereo matching with optical flow or motion detection.[34,76,77] Assuming that the camera motion parameters are known, the optical flow vectors for an object and its stereo disparity are directly related to its distance from the cameras at any time.[76] This requires that objects are stationary or have known trajectories. Unfortunately, this does not hold in an arbitrary environment. Also, in some circumstances the cameras may be stationary, or they may move in an uncontrolled manner. Hence, a general solution to the stereo problem must not rely on optical flow information. In Chapter 7 we discuss the potential use of optical flow as an additional constraint in the General Support Algorithm.

The final related problem concerns the use of active sensors for obtaining depth information. Such systems include ultrasound waves, radio waves, and laser pulses.[9]

Unfortunately, at present there are a variety of problems with such systems. They tend to be fairly expensive, slow, noisy and have low spatial resolution. Future developments, of course, may prove each of these statements false. At present, however, stereo matching offers the best hope for fast, high resolution acquisition of depth information.

2.2. Connectionist Models and Vision

We now present some background on connectionist models. The purpose is to give an outline of various models and show how these models have been used. Details of the computational model we use for stereo matching are deferred until Chapter 4. The discussion in this section is divided into two subsections. The first describes various issues in the design of connectionist networks. The second highlights some connectionist approaches to vision problems.

2.2.1. Computing Organization

This section discusses a number of important issues regarding the design of a connectionist network. Included are: the nature of the individual computing elements and connections, the knowledge organization in the network including a number of useful encoding schemes, the network structure, and the problem of assigning connection weights. Each of these is covered in turn below.

2.2.1.1. Computing Elements and Weights

A variety of different types of computing elements have been proposed for connectionist models. These elements are characterized by the way they combine their inputs, by their activation rules, and by their output rules. Often the latter two are not distinguished. Rumelhart, Hinton and McClelland give a survey of these.[62] We only give a brief summary.

The simplest units compute their total input as a weighted sum of their individual inputs. They determine both their activation and output by comparing this total to a threshold. The output is binary. These elements are called "linear threshold units".

More complicated units with binary outputs have been developed that differ in one or more of these features. Some models compute their activation as a probabilistic function of their input. One frequently used model, the Boltzmann machine,[26,28] uses a "temperature" factor to control this probabilistic function. Often, these computations employ simulated annealing, where the network starts with a high temperature which is gradually reduced until it is zero. When the temperature is high, a node is more likely to change state arbitrarily. As the temperature is lowered the network's behavior approaches that of a gradient descent algorithm. The computation of a Boltzmann machine is a search for the minimum energy in the network. The use of the temperature factor helps the algorithm to escape local energy minima.

Other more complicated models allow input functions with multiplicative interaction between inputs.[62] This allows an active input to be "turned off" by multiplying it by a connection with zero input. Other models allow a continuous

activation function[48] and either an expanded set of discrete outputs[15] or a continuous range of outputs.[48]

There is a fairly consistent definition of connection strengths throughout the literature. Weights are usually continuous and are often limited to the range [-1,1]. The biggest difference between various algorithms in their use of connection strengths is whether they use symmetric or asymmetric weights. Connections are symmetric if the weight from node i to node j must be the same as the weight from j to i. This is often a theoretical requirement used for analyzing simple networks.[28]

A final distinction in the types of computing nodes is between synchronous and asynchronous updating rules. A theoretical advantage of the latter is that it helps reduce oscillations in the network.[62] However, the use of a synchronous rule vastly simplifies the simulation of these networks.

2.2.1.2. Knowledge Organization Issues

The two basic approaches to knowledge organization in connectionist networks are local representations[15] and distributed representations.[27] In local models a single unit represents a single piece of information. In distributed models individual units participate in a number of different representations. Thus, the information represented in the network is distributed across many nodes and a given set of nodes may represent a number of different things. For our purposes, the distinction between distributed and local representations is not very important. However, a number of ideas that have arisen from studying distributed representations are useful. The most important of these is conjunctive coding. It is usually used for detecting and

representing relatively sparse spatial information. The idea is that the overlapping responses of a number of nodes can increase the accuracy in the position of the response. Conjunctive coding also introduces some redundancy in the network which is useful when the individual nodes are unreliable.

2.2.1.3. Network Structure

Connectionist networks are often organized hierarchically, with a cluster of nodes at each level. Different networks employ different interconnection structures within each level, and different types of connections between levels. For example, a network might only have positive connections between levels and negative connections within each level. These restrictions lend structure to the network and are often necessary for analyzing their behavior. They are also necessary for most learning algorithms.

2.2.1.4. Weight Assignment

The weight assignment problem is crucial since the knowledge of the network is stored in the connection strengths. Much of the research in connectionist models has centered on algorithms that automatically "learn" these weights.[26,62,63] These models start with a given network organization and random weights. Through examples the weights are modified to allow the network to solve a particular problem. Learning can be either supervised or unsupervised. In supervised learning, after the network has settled on its solution, the correct solution is provided to the nodes at the top level of the network hierarchy. Based on the differences between the expected and actual

values, the weights are altered. This is propagated down through the network to continue the weight modification.[63] In unsupervised learning, the network is essentially self-organizing. The nodes compete to determine which will respond to a given input.[21,64] Weights are modified to allow the responding node to win more easily on the same input. A single node or group of nodes will eventually respond to a pattern of input. Learning algorithms have worked reasonably well on networks with simple, feed forward structures.[63] However, they have not been applied successfully to problems where the network structure is as complicated as that required for stereo vision.

2.2.2. Connectionist Vision

Hinton and Sejnowski, among others, have described vision as a set of constraint satisfaction problems.[28] This makes them particularly amenable to connectionist solutions. For example, Hinton and Sejnowski have applied both Boltzmann machine computations and gradient descent algorithms to separating figure from ground in images.[39]

The possibility of connectionist solutions to vision problems has motivated Feldman and Ballard in their work.[15] Feldman claims that low and intermediate-level vision are well understood and he has tried to develop a model for high-level vision based on connectionist principles.[16] Clearly, the apparent sequential nature of high-level vision makes it the most difficult problem for connectionist models. However, lower level vision problems have not been adequately solved. The techniques generated from studying low and intermediate level problems may constrain the

solutions to higher level problems. Ballard has proposed a general scheme for solving vision problems making use of a structure called a parameter net.[6] Intermediate levels of the structure are called feature spaces. Each represents a different intrinsic image computation. The matches and depth map generated by stereo vision could be one of these feature spaces.

One of the most complete connectionist vision systems developed thus far was Sabbah's recognition network[65] for Kanade's Origami world.[36] Sabbah developed a hierarchy of subnetworks based on the Hough transform. Using this hierarchy and both bottom-up and top-down processing, the network managed to solve a number of reasonably difficult origami world problems. This included images with a number of corners significantly blurred.

Grossberg and Mingolla have also been developing a vision theory based on connectionist networks.[20,22] Their assumption in studying vision is that visual illusions can provide significant insight into the structure of the human vision system. While they have had reasonable success in constructing explanations for these phenomena, they seem to have underestimated the complexity of normal vision. As a result they have ignored the need for developing and incorporating real-world constraints into their vision networks.

2.2.3. Relaxation Algorithms

Connectionist networks employing cooperative computations are very similar to relaxation algorithms that have been applied successfully in vision.[12,32] There is one qualitative difference, however. Connectionist models implement a finer grained

computation than most relaxation algorithms. Each node in a relaxation algorithm represents a set of alternative decisions about a vision primitive. In a connectionist network, the alternatives are each represented by a separate processing unit (or group of units). Thus, the trade-off between alternative solutions is implicit and must be implemented through the connection structure and weights. In general, our view is that connectionist vision algorithms are a fine-grained implementation of relaxation computations.

Connectionist models, with their ability to naturally integrate influences from a number of different sources, provide a good model for studying vision problems. In our work on stereo vision we apply the computational principles of connectionist models to a real world vision problem. The research literature shows that there has been some success in solving the stereo matching problem, but none of algorithms have had complete success. Before presenting our algorithm we turn to a discussion of the constraints employed in stereo matching to examine why previous algorithms have been less than completely successful.

of individual candidate matches or of groups of two or more candidate matches. In our case a candidate match is formed by a pair of oriented edges, one from each image. A number of the constraints, most notably figural continuity, rely on the orientation of these matching primitives. If a different type of primitive were chosen, the use of the constraints would be different. Thus, to some extent, the analysis relies on the assumption that edges are the matching primitive.

3.1. Uniqueness

The uniqueness constraint has been used either explicitly or implicitly by nearly every stereo matching algorithm.[45] It is stated as follows: *Each matching primitive in an image should match at most one primitive from the other image.* Uniqueness is used to eliminate all but the strongest candidates for each matching point. It is derived from the assumption that each image point corresponds to one point in the world (thus there should be at most one point in the other image corresponding to that world point, and so there should be a unique match for each image point). Uniqueness is the *only* match selection constraint that will be used to *inhibit* candidate matches in the General Support Algorithm. This is allowed because it is a statement about limitations in matching. All the other constraints make statements about what constitutes a valid match.

The above definition of the uniqueness constraint is used by most stereo matching algorithms. However, there have been a number of other formulations of it. Marr and Poggio proposed *one-sided* uniqueness in which at least one of the edges in a valid match must have no other matches.[43] More recently, Drumheller and Poggio

CHAPTER 3

Constraints

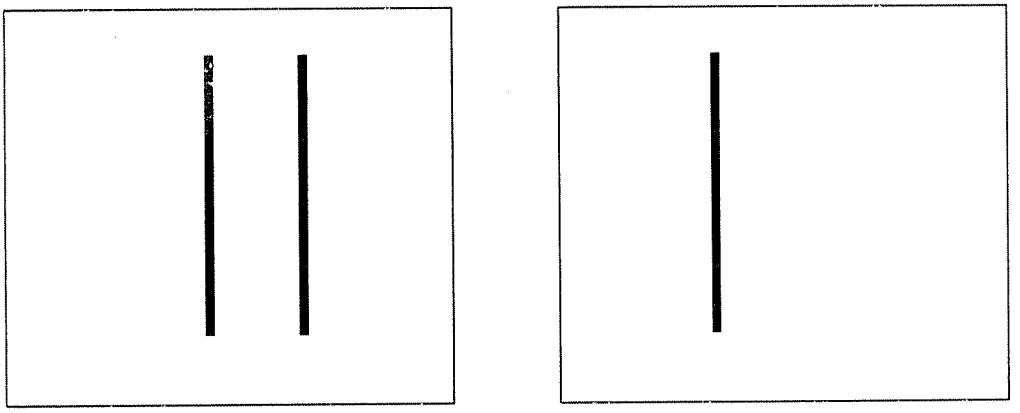
The previous chapter discussed a number of algorithms for solving the stereo matching problem. Now we turn to a detailed analysis of some of the important constraints that have been used by these algorithms. In order to make the analysis general, it is important that we avoid many of the specifics of the actual implementation of the constraints. Thus, the analysis emphasizes the assumptions behind the constraints and their general formulations. The main result is the conclusion that no constraint is sufficiently general to work in all matching circumstances. Also, based on our observations, new formulations of some of the constraints are described. All of the formulations are local in nature, and are amenable to parallel constraint integration in the General Support Algorithm defined in Chapter 4.

Before beginning our analysis, it is important to note the special role of the *uniqueness* constraint. All stereo matching algorithms make use of it either explicitly or implicitly. Thus when we say that no one constraint is sufficient for matching in all circumstances, we mean that no one constraint *in addition to* the uniqueness constraint is sufficient. The uniqueness constraint is discussed in detail in Section 3.1.

Finally, the type of primitive used in matching affects the definition and analysis of some of the constraints. Recall that the constraints are defined either as functions

proposed a stricter form of uniqueness in which there is a *forbidden zone* around each valid match that can not contain other valid matches.[13] This depends on the surfaces being opaque and the matching primitives being fairly sparsely distributed. Such assumptions are not general.

The assumption behind the original form of the uniqueness constraint is not always valid when the matching primitives are edges. Specifically, in some circumstances, the smoothing process involved in edge detection may cause more than one feature to be merged into a single edge in one image. Because of the different viewpoints of the two cameras, these merged features may appear as distinct edges in the other image. For a specific example where uniqueness is not enforced in human vision, consider the stereogram shown in Figure 3.1.[45] In the figure, the result of matching is the impression of one vertical bar floating above another in space. Thus, the vertical bar in the right image matches both bars in the left image.



Left Image Right Image

Figure 3.1. Non-unique matching bars.

The example shown in Figure 3.1 motivates our reformulation of the uniqueness constraint. This is based on Marr and Poggio's definition of *one-sided uniqueness* discussed above. Their definition stated that for a given match between image edges l and r, at least one of the two edges must have no other match. This liberalization of the original uniqueness statement solves both problems mentioned above. By using one-sided uniqueness in the context of their multiresolution matching algorithm, Marr and Poggio were able to abandon the original form of the uniqueness constraint. The formulation used in the General Support Algorithm is actually a combination of normal uniqueness and one-sided uniqueness. Specifically, we define uniqueness in terms of two separate influences on a candidate match (l,r), where l is an edge in the left image and r is an edge in the right image. The first influence is the maximum of the strengths of the competing matches for l. The second is the maximum of the strengths of the competing matches for r. Thus, when there are strong competing matches for both the left and right edges the uniqueness influence will be strong. When there are competing matches for only one of them, the input will be moderate. As we will see in Chapters 4 and 5, this will enforce normal uniqueness except in rare circumstances. One of these circumstances is the example in Figure 2.1. The matches for this figure all have only one uniqueness input. Because of the strong influence of other constraints the uniqueness inhibition is overcome and non-unique matches are accepted.

In addition to the above discussion motivating the idea of one-sided uniqueness, our form of uniqueness is justified by its computational effectiveness in resolving

matches in ambiguous regions. In the context of the relaxation matching algorithm, uniqueness will be able to gradually force the correct interpretation of these matches. Matches at the boundaries of the regions often have only one strong uniqueness input, allowing them to be accepted. The remaining matches are resolved through the propagation of the boundary results. Without defining uniqueness as two separate influences on a match the boundary matches could not have been resolved.

3.2. Detailed Match

Some matching algorithms use comparisons between edge properties to favor some matches for an edge over others.[4,5,41] These are used in addition to the compatibility requirements that define candidate matches. The edge comparison techniques are based on the assumption that the images are taken from similar viewpoints, so that edges are likely to have a similar appearance in both images. Unfortunately, differences in the cameras, inaccuracies introduced by digitization, and slightly differing viewpoints can cause significant variations in the appearance of an edge in the two images.

In the General Support Algorithm the detailed match constraint complements the compatibility requirements used in defining the candidate matches. As noted in Section 1.1, the two edges of a candidate match must have roughly equivalent orientations. Stricter criteria are not enforced because edge orientations are only roughly quantized, and because the true orientation of an edge in the two images can differ due to differing views of the scene. Detailed match compares the average intensities on each side of the edges involved in a candidate match. When these

intensities are similar, there is increased likelihood that this is a valid match. Similar ideas have been used in a number matching algorithms,[4,5,41] but for reasons enumerated above they can not be relied upon solely to discriminate between valid and invalid matches.

The specific definition of the detailed match constraint is as follows. Two intensity comparison measures are employed to support a candidate match. The first measure compares the intensity values on both sides of the edges. Under normal circumstances, each side of the edges in a valid match are likely to be similar. However, potential differences in the appearance of the edges in a match may be accommodated by the second measure. This provides support for a match when the intensities on only one side of the edges are similar. This often arises for occluding edges when there is a significant change in disparity. The use of these measures is mutually exclusive. As a third measure of similarity we could compare the contrasts of the edges in a match. This would accommodate differing illumination conditions. However, we assume that the images are taken simultaneously and from similar positions, so that illumination variations are not significant.

3.3. Multiresolution

Correspondence between similar matches at multiple resolutions provides another useful constraint. It is based on a number of assumptions about the relationship between edges detected at different resolutions. It was first proposed by Marr and Poggio [43] and has been used frequently since then.[19,29,67,79,81] This constraint can be formulated, as in the work of Marr and Poggio, so that the results of

matching at a coarse level are used to narrow the range of matches at the next finer level to no more than two candidate matches for each edge. There are a number of problems with using coarse-to-fine matching in this way. They are related to the assumptions behind multiresolution matching. First, as shown by Mayhew and Frisby,[47] matching by humans can occur outside of the range of the coarsest filter. While this has no definite significance for computational stereo, it suggests that perhaps there ought to be disparities larger than the range of the coarsest filter. In this case, simply choosing the nearest match at the coarsest level, as in the Marr-Poggio algorithm, is not sufficient. Second, the edges at the coarsest levels may be averages of several different features at finer levels. Matches between such edges will not always provide a meaningful measure of disparity for edges at finer resolutions. Most importantly, there might not be any relation between nearby edges detected at different resolution levels.[59] For example, edges arising from grass behind a picket fence do not have any relation to coarser resolutions where only the edges from the fence appear.

In spite of these problems, multiresolution can be a useful matching heuristic. In order to show how, we need to analyze the properties of multiple resolution edge detection and the relationship between edges at adjacent resolution levels. A number of papers on edge detection and scale-space images have discussed methods for relating edges detected at multiple resolutions.[11,25,44,80] These ideas include ways in which edge detection errors can arise, the accuracy of edge locations, and methods for detecting true edges. First, in general, edge errors can occur at any resolution.[25] At

coarser resolutions they may come from averaging a number of features. At finer levels errors arise due to noise in the image data. Second, edge locations tend to be less accurate at coarser resolutions.[80] However, as shown in scale-space images,[80] the position of a particular edge only changes locally between resolution levels, so that the same edge may be tracked across these levels. Finally, Marr and Hildreth showed that edges which persist across multiple resolutions tend to be true edges.[44] We can use these results to develop ways of using support between edges detected at multiple resolutions.

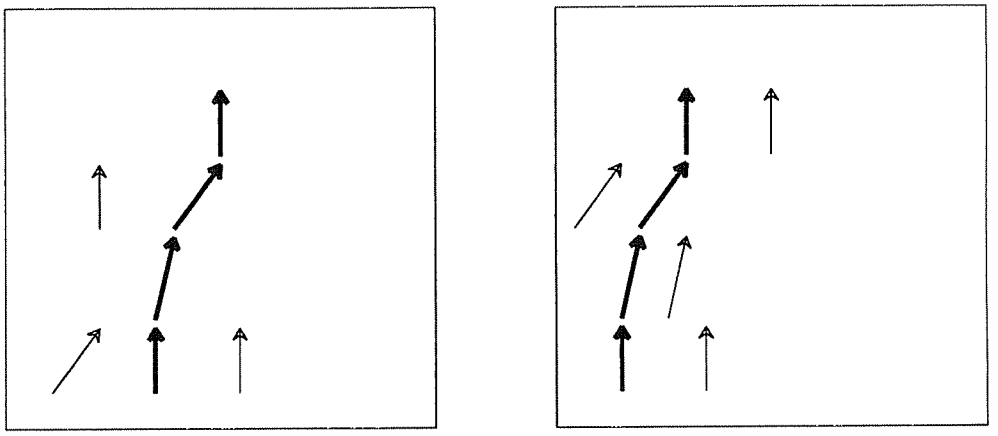
In the General Support Algorithm we use multiresolution matching in two ways. The first is similar to Marr and Poggio's approach although it is not as restrictive. That is, matches at a given resolution are used to provide support for matches at the next finer resolution. However, the disparities of the finer level matches must agree with the disparities obtained at the coarser levels. Also, the finer level edges must be in nearly the same spatial position as the coarse level edges. Coarse-to-fine matching is the main use of multiresolution support.

In the second use of multiresolution matching, coarse level matches receive support from similar matches with similar disparities at the next finer resolution level. When coarse level matches receive support the confidence in the validity of those matches will be enhanced. This fine-to-coarse matching is an important new use of multiresolution for stereo since coarse level matches often receive little support through other constraints.[71]

Note that our definitions of multiresolution support require pairwise interactions between matches at adjacent resolutions, whereas the Marr-Poggio definition used the coarse results to locally re-align the images for finer resolution matching. In our method, matches with no support from coarser levels may still be accepted as valid through the activity of other constraints. This accommodates many of the problems with multiresolution discussed above. Finally, note that our definition of multiresolution involves simultaneous computations at all levels. Previous algorithms have used it hierarchically starting at the coarsest level.

3.4. Figural Continuity

The figural continuity constraint was developed by Mayhew and Frisby as part of a competing theory to those proposed by Marr and Poggio.[47] The constraint states that edges along a contour should match edges along a similar contour in the other image. For example, in Figure 3.2 the edges on the contours (contour edges are in bold in the figure) in both of the images are more likely to match each other than other edges not on a contour. Figural continuity is based on two assumptions. The first is that contours in the scene appear similarly in both images. These "contours" arise in the images for a number of reasons. They can correspond to sharp changes in surface orientation, to borders between areas with differing reflectance properties, including shadow boundaries, and to occluding contours. The second assumption is the Binocular Raw-Primal Sketch (BRPS) conjecture which postulates that contour extraction should work cooperatively with edge matching.



Left Image Right Image

Figure 3.2. Figural continuity example.

The similar-contours assumption of figural continuity is not completely general for a number of reasons: (1) edges arising from fine texture are not always part of a smooth contour, (2) occasionally a contour in an image can arise from multiple contours in the scene at different depths (these will not be part of the same contour in the other image), and (3) because of other nearby features, a scene contour can have significantly different appearances in the two images. In addition to these problems with the assumptions, figural continuity does not provide any means for distinguishing between matches when there are nearby similar contours in an image. In this case, an edge on a given contour will match edges on a number of different contours in the other image with nothing to distinguish between the matches.

In spite of these problems we will make strong use of figural continuity. Our formulation is a purely local one where figural continuity is determined pairwise between matches. A pair of matches meets the requirements if the edges in each

image appear to be part of a contour, and if the hypothesized contours in each image have similar local appearance. Because this pairwise definition is only accurate over a limited range, we restrict figural continuity to nearby matches. This avoids many of the problems with the similar-contours assumption, but it is still sensitive to gaps in contours caused by noise. In addition, it can not overcome the problem of matching when there are multiple similar contours. The disparity gradient, which gathers matching support over a larger area of the image, can help to overcome this problem. Our definition of figural continuity is consistent with the local, cooperative nature of the BRPS conjecture.

Figural continuity is one example of the continuity constraints that have been used for stereo vision.[45] Each makes assumptions about object, depth or surface continuity that are violated in various circumstances. Thus, continuity constraints are only applicable between matches that obey the underlying assumptions. In the next section we discuss the disparity gradient which has similar assumptions about surface continuity, but it provides support between edges that are not locally known to be part of a contour. Together, figural continuity and the disparity gradient provide a strong, complementary formulation of object continuity constraints.

3.5. Disparity Gradient and Gaussian Support

A constraint that has recently received a lot of attention is the disparity gradient.[58,73,74] It is a measure of similar disparity values for two candidate matches. Simply put, matches that satisfy the disparity gradient limit support each other. Similar ideas were developed by Prazdny using a Gaussian support model based on

the assumption of object cohesiveness.[59] We will concentrate on the disparity gradient.

Both the disparity gradient and Gaussian support are based on the assumption that objects occupy a well-defined three-dimensional volume. The constraints thus assume that points from the surface of an object appear with relatively similar disparities in an image. See Figure 3.3 for example. The disparity gradient is formally defined as follows. Suppose l_1 and l_2 are points in the left image, and r_1 and r_2 are points in the right image. Further suppose that $p_1 = (l_1, r_1)$ is a potential match and $p_2 = (l_2, r_2)$ is a potential match. Finally, let $d(p_i)$ be the disparity of a match.

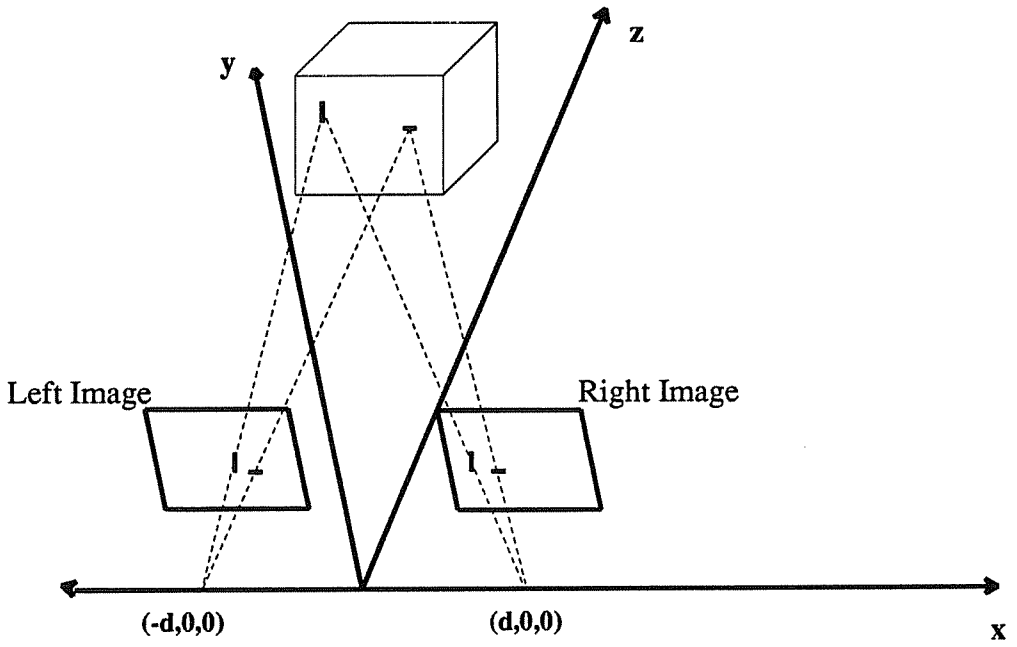


Figure 3.3. The points on the surface of the object are projected onto the image with similar disparities. Matches for these points will satisfy the disparity gradient limit.

Then the matches support each other if

$$\frac{|\, d\,(p_1) - d\,(p_2)\,|}{D\,(p_1, p_2)} \leq 1$$

where $D\,(p, q\,)$ is the distance between two matches. (The distance between two matches is the Euclidean distance between the mid-points of the edges in each match, as shown in Figure 3.4.) The value 1 is called the disparity gradient limit. In other words, they support each other if the difference in their disparities is less than their cyclopean separation.

Examining the above equation, it is easy to see that nearby matches must have similar disparities, while the disparity requirement for matches farther apart is more

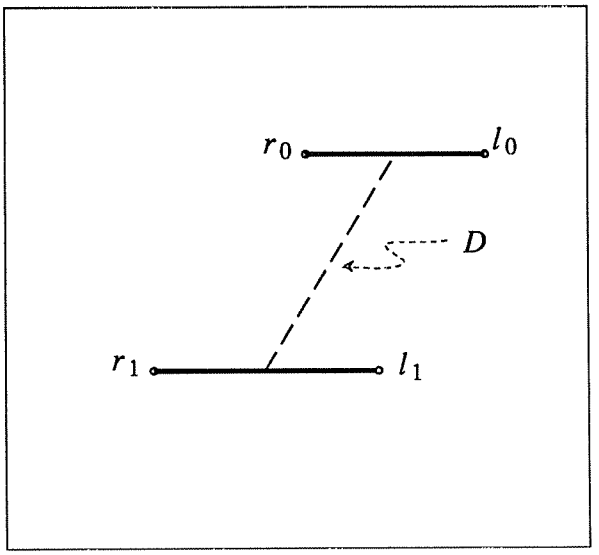


Figure 3.4. The distance between two matches in the disparity gradient is the distance between the mid-points of the points in each match.

flexible. Thus, nearby valid matches from the same surface tend to support each other. Invalid matches are less likely to receive support in this area.[58] This is true as long as the objects are on the same surface, and the orientation of the surface is not nearly perpendicular to the image. When two valid matches are from points on different objects at differing depths, the disparity gradient is often greater than the limit.

There are several problems with the disparity gradient. First, it often provides only weak discrimination between matches. In fact, given a maximum allowed disparity D, any matches farther apart than D will satisfy the disparity gradient limit. To make it more discriminating we will provide a slight skewing of the strength of the relations the disparity gradient defines in favor of matches for fronto-parallel surfaces. This follows the Gaussian definition of support used by Prazdny.[59]

A second problem with the disparity gradient is that, contrary to the surface continuity assumption, it allows support between matches from different surfaces. This occurs most often along occluding boundaries, and is especially a problem at boundaries between densely textured surfaces and relatively untextured surfaces in the images. Here, the dense matches for the textured surfaces can cause improper matches for the sparse surface. A complete solution to this problem requires non-local mechanisms to detect surface boundaries and cut-off support across those boundaries. These mechanisms are developed in Chapter 7. However, the combined use of figural continuity and the disparity gradient, along with the relaxation computation of the General Support Algorithm, can overcome these problems to some

extent.

The final difficulty with the disparity gradient is that it makes no distinction between support between matches along a contour, and support between matches that may be on different contours. Support in these two circumstances should be defined differently. Support between matches along a contour meets both disparity gradient and contour similarity requirements, and therefore, it gives a stronger indication of the validity of the matches. Other matches may only satisfy the disparity gradient limit. To address this, we only allow disparity gradient support between pairs of matches when they do not meet figural continuity requirements. Thus, figural continuity and the disparity gradient complement each other in their definition of support. Figural continuity provides support between matches that locally meet contour requirements; the disparity gradient provides support between matches for edges on different contours and between matches for edges that are too far apart for local contour similarity measures. It therefore gathers support over a much broader region.

3.6. Chapter Summary

The emphasis of this chapter has been the analysis and reformulation of existing constraints for stereo matching. These constraints include figural continuity, disparity gradient, coarse-to-fine and fine-to-coarse multiresolution, detailed match, and uniqueness. Our general conclusion is that none of the constraints (used along with uniqueness) is sufficient for matching in all circumstances. In addition, we have proposed new formulations of a number of constraints. The resulting constraints are defined locally as relations between pairs of candidate matches.

CHAPTER 4

Local Constraint Integration: The General Support Algorithm

Previous sections have highlighted the strengths and weaknesses of a number of stereo matching constraints and the algorithms employing them. This showed that none of the constraints provides correct matching information in all circumstances. To overcome this difficulty we propose the General Support Algorithm for integration of multiple constraints. The important features of this algorithm are: (1) the local definition of each constraint, (2) the *cooperative*, *parallel* integration of the constraints using only *positive* constraint influences, and (3) the implementation of the algorithm in a connectionist network. In this chapter we lay out this algorithm in detail. We begin with a discussion of the need for constraint integration.

4.1. Integrating Multiple Constraints

In order to improve stereo matching, the problems with the constraints must be solved. The two obvious approaches to this are: (1) propose a new constraint that provides accurate matching information in all circumstances, or (2) incorporate a number of the constraints in such a way that the weaknesses of one constraint are compensated for by the strengths of the other constraints. We examine each of these possibilities in turn.

The search for general constraints motivates much of the work in stereo vision. While continued research may produce improved constraints, it is unlikely that an

all-encompassing constraint will be produced. Such a constraint must be based on general assumptions about objects in the world. In addition, it must account for differences in the appearance of objects in both images, including noise and variations in the two images arising naturally from occlusions and differing camera positions. These two goals reflect the need for generality in defining the constraint. They conflict with the third requirement - that is, the constraint must be able to accurately discriminate between the valid and invalid candidate matches.

The alternative approach to overcoming the weaknesses of the individual constraints is to integrate a number of them. This is the approach taken in the General Support Algorithm. We use the following guidelines in constructing such an algorithm:

- The algorithm should work on a wide range of images. This includes random-dot stereograms and natural images. In addition, domain-specific assumptions should not be incorporated into the algorithm.
- The algorithm should overcome the weaknesses of the individual constraints. This is simply a restatement of the main reason for using multiple constraints. In situations where one constraint is weak or inapplicable, other constraints should provide matching information.
- The algorithm should produce a higher percentage of correct matching decisions (correct matches, or correctly finding no match for an edge) when more constraints are providing useful matching information. Thus, any conflict between the constraints should not show up as a reduction in the number of

correct matching decisions.

- The algorithm should work as well as possible in matching situations that have been problematic for other algorithms. This includes noise that shows up as spurious matches and missing primitives, more significant structural differences between the images, occlusions, and periodic image features.
- Ideally, the use of additional constraints should not slow down the algorithm, at least in a parallel implementation. This may be possible in a connectionist network, where the cost of using additional information is in the number of nodes and connections defined. Thus, if we assume that such an implementation is possible, we may coarsely measure the time for the network in terms of the number of iterations required.

4.2. The General Support Algorithm

The General Support Algorithm (GSA) is based on the above guidelines for the integration of stereo constraints. It employs constraints that are defined locally as pairwise relations between candidate matches, or functions of a single candidate match. The GSA is an iterative, parallel relaxation algorithm. Using the constraints, each candidate match propagates its strength to other matches, and in turn is influenced by those matches. This is repeated until the GSA determines what it will accept as a valid match.

In the remainder of this section we describe the algorithm in detail. This includes the General Support Principle which we use to organize the influences of the

constraints, the local definition of each constraint (and the justification for the local definition), and how the algorithm meets the requirements stated in the previous section. This description avoids the details of the connectionist implementation of the GSA. These details are given in Section 4.3.

4.2.1. The General Support Principle

The General Support Algorithm organizes the influence of the constraints using the *General Support Principle (GSP)*. This principle states that, except for uniqueness, *constraints are only used to support candidate matches*. There are a number of reasons for defining the GSP and using it to organize the interactions of the constraints. First, however, we consider some of the implications of the GSP as it affects what the GSA will and will not accept as a valid match.

Because of the GSP, the only negative constraint influence is that of uniqueness. In the absence of noise and occlusion, uniqueness would be sufficient for eliminating all invalid matches since every edge would have at least one valid match. However, because of noise and occlusions there are invalid matches that can not be removed through the uniqueness constraint. To solve this problem we use *decay* in the strength of a candidate match in the GSA. During each iteration of the GSA, candidate matches receive support from other matches. This changes the strengths of the matches. After each iteration we use decay to reduce these strengths. In this way, noise matches that receive little support will be suppressed by the decay factor. In general, because of the GSP, valid matches are those whose constraint support overcomes the inhibiting influences of uniqueness and decay.

In justifying the General Support Principle the main observation is that the constraints are derived from assumptions about objects and their appearance in images. These assumptions lead to assertions concerning relationships between valid matches, not about inhibition between matches that may or may not be valid. For example, the disparity gradient's assumption is that pairs of matches from the *same surface* will satisfy the disparity requirements. There is no claim about pairs of matches from different surfaces. Such matches may often violate the disparity gradient limit. (Prazdny[59] gives a similar argument.) Similarly, figural continuity is a statement about matches for edges along a contour, not about matches for texture edges or edges from different contours. Similar arguments can be made about the other constraints.

Another justification, following from the local definition of the constraints, can be given in defense of the General Support Principle. Specifically, since the constraints are defined as pairwise relations between candidate matches, there is no way to locally distinguish between the following possibilities: (1) the constraints are outside the assumptions of the constraints, or (2) one or both of the matches are invalid. For example, two matches from different surfaces are allowed to violate the disparity gradient requirements, whereas matches from the same surface are not. With purely local constraint interaction these cases can not be distinguished. We rely on the lack of support for a match (or weaker support than one of its competitors) to eliminate it. This is a "least commitment" strategy in defining the local interactions between the constraints.

4.2.2. Constraints Used in the GSA

The General Support Algorithm incorporates the following constraints: uniqueness, disparity gradient, figural continuity, coarse-to-fine and fine-to-coarse multiresolution support, and detailed match. Except for uniqueness, each of these constraints is defined locally as pairwise relations between candidate matches, or functions of the two edges involved in a candidate match. The local nature of each constraint is either built into the assumptions behind the constraint or in its actual formulation. Specifically, the disparity gradient equation that defines support is a function of a pair of candidate matches. Figural continuity is local because our definition of it follows the Binocular Raw Primal Sketch conjecture[47] in which pairs of matches that could be along the same contour support each other. The restricted version of multiresolution described in Section 3.3 allows support between nearby matches at adjacent resolutions. Support is both coarse-to-fine and fine-to-coarse. Finally, detailed match is by definition a function of a single candidate match. Thus, the local definition of each constraint is natural.

In the remainder of this section we discuss the influence of each constraint and how some constraints complement each other in their definition of support. In addition we examine a few other constraints, not implemented, that fit easily into the context of the GSA.

Uniqueness is necessary for any stereo algorithm matching primitives as simple as edges since there are usually many candidate matches for each edge. Also, the particular form of uniqueness used here is necessary to help disambiguate images with

repetitive structures.

The **disparity gradient** and **figural continuity** are complementary surface structure constraints. Both reflect surface continuity assumptions. However, figural continuity provides strong support between matches that locally appear to be along the same contour, while the disparity gradient gathers support over a broader region based on similar disparities. Thus, the disparity gradient provides support between textural features of a surface and between points on a contour that are too distant for figural continuity's local contour requirements.

The two surface structure constraints, figural continuity and the disparity gradient, also assist in overcoming each other's weaknesses. Figural continuity is sensitive to gaps in contours and it can not distinguish between locally similar contours. The disparity gradient may overcome these difficulties by gathering support in a larger region. On the other hand, the disparity gradient sometimes provides only weak discrimination between matches. Figural continuity is usually a much more certain indicator. In cases where the disparity gradient might distinguish only weakly, a number of matches may receive strong support through figural continuity. These matches can reinforce other matches with similar disparity through the disparity gradient. Thus, figural continuity and the disparity gradient assist each other in overcoming difficult matching situations.

Multiresolution is most useful for identifying scale-persistent features of the images and propagating support between appropriate matches for these features at various resolution levels. This accelerates the matching process, and provides strong

indication of the correct disparities for the most significant features. It also helps to resolve matches for ambiguous regions at fine resolutions. Note that multiresolution support is computed simultaneously at all resolution levels. This is contrary to a number of multiresolution matching algorithms which define a strict matching order starting at the coarsest resolution.[43]

Detailed Match serves two main purposes. First, since it provides an additional initial indication of which matches might be valid, it tends to accelerate the matching process. Second, it can help overcome ambiguities due to repetitive image structures. It does this by computing additional similarity measures that compare the areas near the edges in a match. This provides additional help in distinguishing the matches. When a coherent set of matches in a repetitive region receive detailed match support, the relaxation process of the General Support Algorithm will eventually select them as valid. Without this detailed match support the matches might remain ambiguous.

One commonly used constraint, the **ordering constraint** is not used in the GSA. (Recall from Chapter 2 that the ordering constraint states that edges in a row in one image should match edges in the same row in the other image in linear order.) There are two main reasons for this: (1) it relies on limiting assumptions about the types of surfaces, and (2) more importantly, it is not necessary. The reason for the latter is as follows. The disparity gradient will not allow support between matches that violate the ordering constraint (the disparity gradient result is greater than 1). None of the other constraints will provide support between pairs of matches that violate it either. Thus, in order for two nearby matches violating this ordering constraint to both be

accepted as valid, they must both receive strong support *independently* in the same neighborhood. This is exactly when the ordering constraint should not apply. In all other cases the activity of the General Support Algorithm will implicitly implement the ordering constraint.

Finally, some other constraints may easily be incorporated into the General Support Algorithm. The simple, regular structure of the algorithm makes this possible. The only requirement in using other constraints is that they integrate smoothly with the local support influences used by the GSA. Two examples of this, focal gradient[56] and optical flow, are discussed in Chapter 7 as ways of providing further assistance in resolving matches for repetitive image structures.

4.2.3. The Guidelines for Integrating Multiple Constraints

In this section we see how the General Support Algorithm follows the guidelines for integrating multiple constraints stated at the end of Section 4.1. Chapter 5 contains further discussion based on the results of testing the GSA on real images and random-dot stereograms. We discuss each of the guidelines in order.

- The GSA should work on many different types of images. There are two main reasons why it will do so: (1) the algorithm matches edges, which can be reliably extracted in a wide range of images, and (2) the GSA employs no domain-dependent assumptions.
- The GSA overcomes the weaknesses of the individual constraints. This can be seen in two ways: (1) the situations in which the constraints are prone to error do

not overlap. This is most easily seen in the complementary definitions of the surface continuity constraints: figural continuity and disparity gradient. (2) Because a number of constraints are used, the individual constraints can be focused on matching situations where they work best. The most prominent example of this is the use of multiresolution. Our definition emphasizes pairwise interaction between matches. Valid matches may not receive this multiresolution support for a number of reasons. However, these matches may still be accepted by receiving support through other constraints.

- Chapter 5 demonstrates the quality of the matching results of the GSA but, unfortunately, appropriate statistical data from other algorithms are not available. This problem is discussed further when we present the experimental results.

- The GSA should work well in a number of difficult types of matching circumstances, that are problematic for previous algorithms. Solutions to some of these problems are reflected in the formulation of the constraints and in their integration. For example, one common problem is that of fragmented contours and contours that have significant appearance differences between images. Figural continuity is sensitive to this problem, but the disparity gradient helps to overcome it by gathering support in a wider region. Multiresolution also helps to overcome this by searching for support orthogonally to surface structure constraints. In other words it searches for support in the resolution dimension instead of spatially. Other examples of problems that the GSA help to overcome include the following. The use of one-sided uniqueness and detailed match

sometimes helps to correctly handle scenes with periodic features. In addition, the combined use of a number of constraints helps to improve the overall performance of the algorithm in terms of the number of iterations required. The strongest constraints in this regard are figural continuity and multiresolution. These are less ambiguous than the disparity gradient which defines a wide range of support for a match. Finally, the problem of finding matches for occluded edges is implicitly addressed by the use of decay and uniqueness. An explicit solution to the occlusion problem requires non-local matching mechanisms (see Chapter 7).

- The use of more constraints does not increase the number of iterations required by the algorithm. Additional constraints can increase the iterations needed for matching when there are inconsistencies with other constraints. However, because of the General Support Principle, this conflict can only occur indirectly through support for competing matches. Also, since we rely on a number of constraints, we have implemented each constraint more conservatively. These combine to reduce the conflicts between the constraints, so that overall, the use of a number of constraints improves the time performance of the algorithm.

4.3. Connectionist Implementation of the GSA

The General Support Algorithm is implemented using a hierarchical connectionist network. The lower levels of this network represent the input images and implement edge detection. We do not consider the edge detection network in detail. All we require is that it provide initial activation to the appropriate candidate

matches.

The highest level of the network is the matching network. It is designed so that (1) each node represents a distinct potential match, and (2) most of the constraints are implemented directly in the connections between candidate match nodes. The matching works through a sequence of iterations. During each iteration each node computes its new activation. This new activation is based on the old activation, the decay rate, the supporting input and the blocking input through uniqueness.

In the remainder of this section we consider the activation and output functions of the nodes in the network, the implementation of the constraints, weight assignments for the constraints, and some additional issues concerning the design of the GSA. These are each covered in a separate subsection.

4.3.1. Node Activation and Output Functions

The activation of a node depends on the combined influences of the supporting constraints, uniqueness, decay and the node's prior activation. The supporting input to a match is a weighted, linear combination of the constraint input, given by:

$$I_{i,t+1} = \sum_{j=1}^{N_i} O_{j,t}\, w_{j\,i}$$

where the $O_{j,t}$ are the supporting inputs to match i, and $w_{j\,i}$ is the magnitude of the supporting connection. Uniqueness input is given by:

$$B_{i,t+1} = \beta\ \max\left(O_{j,t} \mid j \in Left_i\right) + \beta\ \max\left(O_{k,t} \mid k \in Right_i\right)$$

where $Left_i$ is the set of competing matches for the left edge of match i, and $Right_i$ is the set of competing matches for the right edge. This reflects the definition of the uniqueness constraint given in Chapter 3. The new activation is:

$$A_{i,t+1} = (1 - \delta)\, A_{i,t} + I_{i,t+1} - B_{i,t+1} + \theta_i$$

where δ is the decay rate and θ_i is the base activation value ($\theta_i = 0$ within the matching subnetwork). The output, O_i is simply a threshold function of the activation:

$$O_{i,t+1} = \begin{cases} A_{i,t+1}, & A_{i,t+1} \geq \phi \\ 0, & A_{i,t+1} < \phi \end{cases}$$

where ϕ is the threshold parameter. The activation is limited to [-1..1], and the output is limited to [0..1]. When the activation of a node reaches 1.0 it is said to be *saturated*, and it remains at 1.0 for the duration of the matching. Our model is slightly more complicated than many connectionist models (e.g. those of Feldman and Ballard,[15] and the PDP research group[62]).

4.3.2. Constraint Implementation

As noted previously, most of the constraints are implemented locally as pairwise relations between candidate matches or functions of individual matches. The initial input to the matching network comes from lower levels of the network. This includes the results of edge detection and slight additional input from the detailed match constraint when its requirements are met by a match. After the initial input, all the

other input comes from the constraints within the matching network. In this subsection we consider the details of the implementation of these constraints. This includes detailed match, disparity gradient, figural continuity, and multiresolution. The uniqueness constraint is built directly into the activation function for a match node (described in the previous section). Our description here will define the spatial range of support of a constraint and the weight of its connections. Each weight equation includes a parameter defining the overall strength of a constraint. The assignment of these parameters is considered in Section 4.3.3.

The implementation of the *detailed match* constraint requires that the smoothed intensities on either side of a given candidate match be compared separately. This produces three additional inputs to a candidate match: (1) input when the low intensity sides of the edges are similar, (2) input when the high intensity sides are similar, (3) input when both sides are similar. The three inputs are mutually exclusive. For (1) and (2) the additional input should be *BASE_ONE_SIDE*. For (3) the additional input should be *BASE_BOTH_SIDES*. A small subnetwork is required to determine the detailed match support. Details of it are given in Appendix 1. Those candidate matches meeting the requirements of detailed match will receive slightly higher initial activations.

The *disparity gradient* constraint is implemented as follows. For a given pair of matches $p_i = (l_i, r_i)$ and $p_j = (l_j, r_j)$, first it must be determined if they support each other. To check this, let the distance between p_i and p_j be $D(p_i, p_j)$ Further, let $d(p)$ be the disparity of a match. Then the matches support each other if

$$\frac{| d(p_i) - d(p_j) |}{D(p_i, p_j)} \leq 1$$

In addition, their distance apart $D(p_i, p_j)$ is limited by *MAX_DISTANCE*. *MAX_DISTANCE* is bounded by the maximum allowed disparity. Beyond this, any pair of matches support each other. Finally, a pair of matches that meets the figural continuity requirements do not also support each other through the disparity gradient.

To determine the connection weight for the disparity gradient, let $d_{i,j} = | d(p_i) - d(p_j) |$ then the weight is given by

$$w_{i,j} = \frac{BASE\_DG}{D(p_i, p_j)} \times \frac{c}{(d_{i,j} + c)}$$

where $c \geq 1$ is a constant, and *BASE_DG* is a parameter controlling the overall strength of the disparity gradient. The inverse distance factor is the same as the definition of the disparity gradient given by Pollard, Mayhew and Frisby.[58] It reflects the likelihood that the candidate matches are from the same surfaces. The $c / (d_{i,j} + c)$ term scales the support as the disparity difference between the matches increases. (Note that when $d_{i,j} = 0$ this term is 1.) This is necessary since the definition of support is extremely liberal. Without it, the disparity gradient only weakly discriminates between candidate matches for an edge. However, this scaling tends to favor surfaces that are parallel to the image plane. (Note that Prazdny's Gaussian support model does this also.[59]) Increasing c reduces the favoring of fronto-parallel surfaces, but decreases the discriminatory ability of the constraint. See Figure 4.1 to examine how the weight of the disparity gradient connections scale with distance and the difference in disparity.

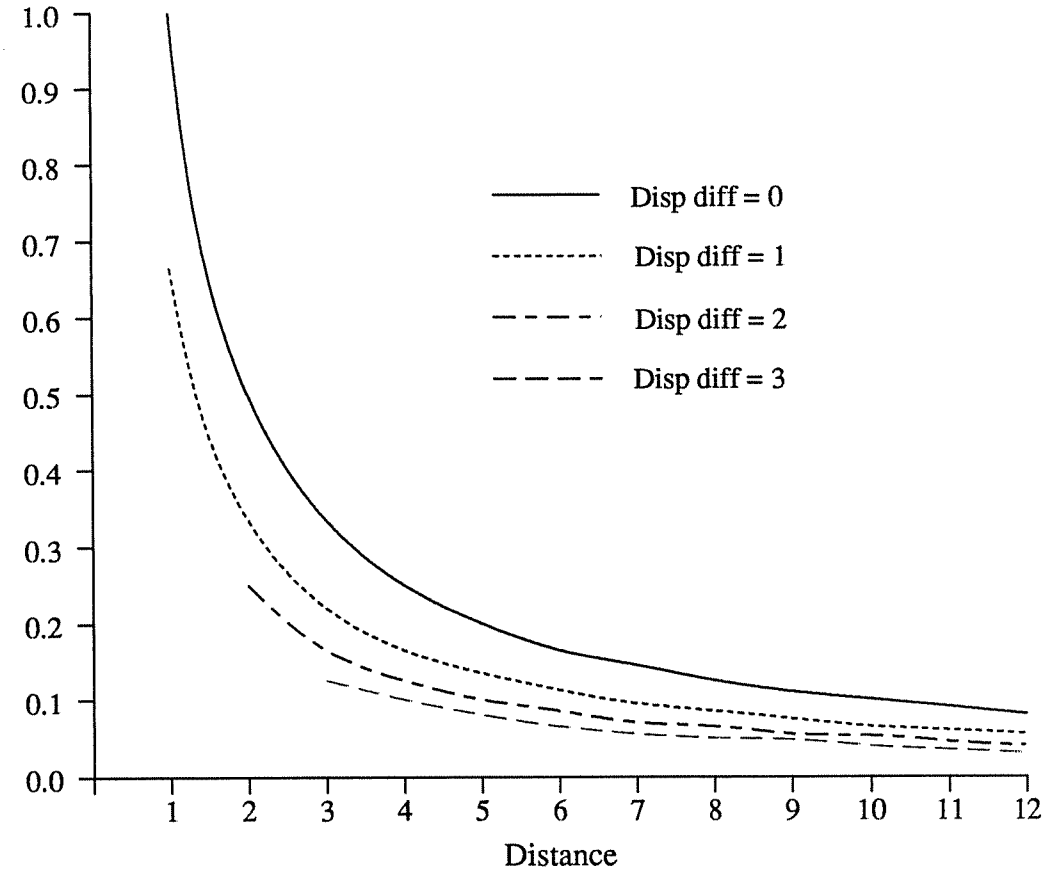
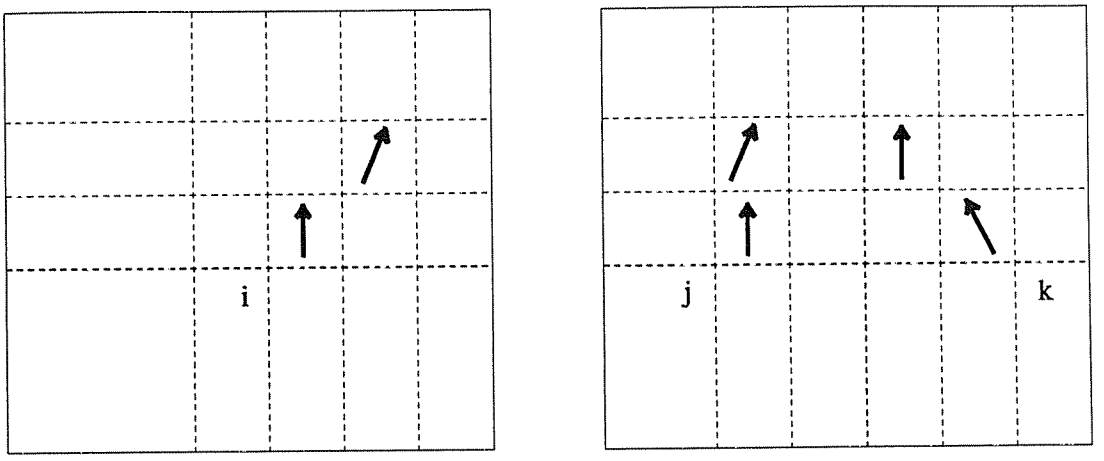


Figure 4.1. Weight of the disparity gradient connection as a function of the distance and the disparity difference between the matches. On the vertical axis 1.0 represents the maximum possible disparity gradient weight.

Figural continuity is implemented between pairs of candidate matches over a limited distance. A pair of matches will support each other if their left and right edges appear to be part of a contour in each image and if they meet the disparity gradient requirement. The disparity gradient is used to ensure that the contours in each image are similar. Figure 4.2 gives two examples of this. The actual support between these matches p_i and p_j is weighted inversely by their distance apart, $D\ (p_i, p_j)$. The weight is scaled using the parameter *BASE_FC* :



Left Image Right Image

Figure 4.2. The pair of matches between contour i in the left image and contour j in the right image support each other through figural continuity. However, the pair of matches between contour i and contour k do not support each other; they violate the disparity gradient limit.

$$w_{i,j} = \frac{BASE\_FC}{D(p_i, p_j)}$$

The implementation of multiresolution support is straightforward. Coarse support for a match is found by searching the next coarser resolution matches near a given match. The support requirements include similar position and orientation for the edges in each image, and nearly identical disparities. The weight is given by *BASE_COARSE_TO_FINE*. Fine-to-coarse support has the same requirements and has a weight of *BASE_FINE_TO_COARSE*. Note that these multiresolution weights involve no distance factor. The main reason for this is that the position of an edge at differing resolutions can vary. Refer back to Section 3.3 for a discussion of this.

The above definitions of the constraints imply a large number of connections for each match. However, for any particular pair of images very few of the connections

will be used. That is, only a few of the connections will communicate positive output between candidate matches. For example, a given candidate match will usually have at most one supporting match at the next coarser level. This observation is important both for setting the values of the parameters and in simulating the network.

4.3.3. Weight Assignment

The definitions of the constraints given above each involve a parameter that scales the strength of the constraint. Thus, the remaining problem in defining the network is to establish the values of the various parameters and thereby set the connection weights. These parameters are associated with the implementation of the constraints and the definition of the activation function, including the uniqueness parameter, β, and the decay parameter, δ. Our approach to parameter assignment involves three steps: (1) develop heuristics for the relative values of the parameters and their approximate values, (2) use simple synthetic images representing difficult matching problems to refine the values, and (3) fine-tune the values through experience with random-dot stereograms and real images. The first and second of these steps are developed below; the third step is discussed in Chapter 5.

4.3.3.1. Learning vs. Analytical Weight Assignment

Before presenting our heuristics for parameter assignment, we briefly examine the major alternative approach to weight assignment in connectionist models: learning. A large number of learning algorithms have been developed. Unfortunately, these algorithms are inadequate for this problem. Algorithms that employ supervised

learning[63] require *a priori* identification of all the valid and invalid matches. This is impractical for the large set of images necessary to accurately set the weights. Those using unsupervised learning[21,64] require internal feedback to adjust the weights. This implies the need for a more complete vision system than the General Support Algorithm. Without such a system, weight adjustment in unsupervised learning would be based solely on what is accepted as a valid match. This would tend to reinforce errors made in determining the valid matches.

In taking the analytical approach to weight assignment we claim that there is a fairly wide range of useful values for the parameters. Reasons for this are as follows. First, because of the General Support Principle (GSP) the supporting constraints only conflict indirectly through the uniqueness constraint. Second, because of the iterative, cooperative nature of the algorithm, under normal circumstances the valid matches in a region of one image form a coalition of support. (This behavior is consistent with the nature of relaxation algorithms.[12]) Within the coalition there is usually support through a number of constraints. Thus, precisely set values for the weights in this case is not crucial. These statements also imply that: (1) the most important parameter assignment involves the relative strengths of uniqueness and decay versus the supporting constraints, and (2) an important test of the constraint values comes when there is strong competition between conflicting groups of matches. Examples of this will be seen in the synthetic images used for tuning the parameter values.

4.3.3.2. Heuristics for Weight Assignment

The observation that the valid matches form a coalition of support provides us with our first weight assignment heuristic. Specifically, when there are a number of conflicting matching interpretations for an area of the image, the algorithm must have enough time to identify the valid matches. This implies that the constraint support parameters should be relatively low to allow the consensus to emerge. If the parameter values were too large, the algorithm might accept too many matches. This also implies that the value of the uniqueness parameter, β, needs to be relatively large. When there are competing matches for both edges in a candidate match, m, and the support for these matches is approximately equal to the support for m, relatively minor differences in the support for the matches will decide between competing matches. When $2\times\beta \approx 1.0$ this occurs naturally. Thus, we set $\beta = 0.5$. The decay rate δ is set much smaller since it is mainly used to eliminate candidate matches that receive little support.

There are a number of other heuristics that allow us to qualitatively relate the supporting constraint parameters, and to relate these to uniqueness and decay. For example, the relative strength of the parameters for figural continuity and the disparity gradient is defined through the following observations. First, both constraints are derived from assumptions about the appearance of an object surface in the images and they complement each other in their definition of support for a match. This implies that $BASE\_FC \approx BASE\_DG$. However, the disparity gradient defines support over a two-dimensional area (although the matches are relatively sparse), whereas figural

continuity is defined over a limited range in one dimension. In addition, the definition of support in figural continuity is more stringent than the disparity gradient. This implies that $BASE\_FC \gg BASE\_DG$. The relative weights for these parameters is a trade-off between these observations. In practice we have found that $BASE\_FC \approx 2 \times BASE\_DG$ works well. Also, in general, the disparity gradient should be the weakest parameter since it is the least discriminating in its definition of support.

Next, we need to define the multiresolution parameters. Coarse-to-fine is the strongest of the support parameters since it is the least ambiguous. However, it should not be large enough that it can not be over-ridden by a coalition of support from other constraints. Fine-to-coarse support is used to compensate for weak support that coarser matches receive through other constraints. Thus, we want $BASE\_FINE\_TO\_COARSE \approx \delta$.

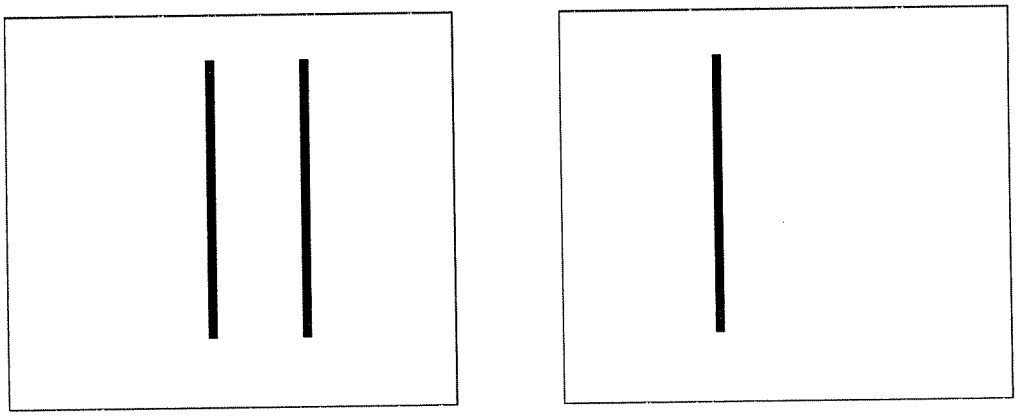
To summarize, in addition to some quantitative relations between the parameters we have the following ordering of the parameter values:

$$\beta > BASE\_COARSE\_TO\_FINE > BASE\_FC$$
$$> BASE\_FINE\_TO\_COARSE \approx \delta > BASE\_DG$$

Finally, the two detailed match weight parameters should be significantly smaller than the others. Clearly, $BASE\_BOTH\_SIDES > BASE\_ONE\_SIDE$. Also, $BASE\_ONE\_SIDE > BASE\_BOTH\_SIDES / 2$ since the appearance of one side of an occluding edge can vary significantly between two images.

4.3.3.3. Further Parameter Tuning Using Synthetic Images

We use three hand-drawn synthetic images for further refinement of the parameter values. From these images we derive five example matching situations. These represent significant tests for the parameter values and the algorithm. For each example there is assumed to be a correct set of matches. The parameter values should be adjusted so that the GSA works in all circumstances. The first, Example 4.1, is shown in Figure 4.3. In the figure, the edges for the bar in the right image should match edges for both bars in the left. Matches for edges in the middle of the bars receive strong support through figural continuity and multiresolution connections. They receive only weak disparity gradient support from distant matches on the matching bars. The negative input to each match from uniqueness involves input from only one image (a competing match in the left image). In this case the strength



Left Image Right Image

Figure 4.3. Synthetic image Example 4.1.

of decay and uniqueness should be overcome by the supporting inputs and all matches should be accepted.

Figure 4.4 shows a second pair of images. In this case, assume that the edges on a bar in one image could match edges from either bar in the other image. Hence, there are two uniqueness inputs for each match. In looking at the figure, it is obvious which bars should match. However, the only difference in support is that there is disparity gradient support between the matches along the two pairs of correctly corresponding bars (i.e. support between matches for edges on bars l_0 and r_0, and matches between edges on bars l_1 and r_1). Thus, normally, the correct matches should be chosen easily (Example 4.2). When the disparity gradient support is eliminated between the matches from l_0 and r_0 and matches from l_1 and r_1 no matches at all should be accepted (Example 4.3).

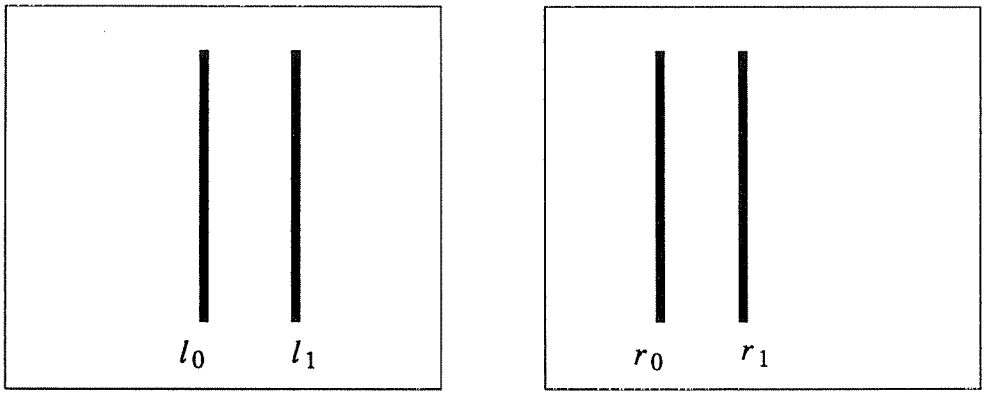


Figure 4.4. Synthetic image for Examples 4.2 and 4.3.

The final pair of images which is useful for studying the parameters is shown in Figure 4.5. In Example 4.4, assume that the candidate matches correspond to edges on the following pairs of bars, one from each image: (l_0,r_0), (l_0,r_1), (l_1,r_1), (l_1,r_2), (l_2,r_2), (l_2,r_3), (l_3,r_3). Thus, each edge has two candidate matches, except for edges along bars r_0 and l_3. Hence, matches for these edges, (l_0,r_0) and (l_3,r_3) have only one uniqueness input. Every other candidate match has two. Through the interactions of the network, one-sided uniqueness and slight disparity gradient differences should force the matches (l_i,r_i) to be selected as valid. For Example 4.5 suppose that there are more candidate matches. Specifically assume that the candidate matches correspond to edges on the following pairs of bars, one from each image: (l_0,r_0), (l_0,r_1), (l_1,r_0), (l_1,r_1), (l_1,r_2), (l_2,r_1), (l_2,r_2), (l_2,r_3), (l_3,r_2), (l_3,r_3). In this example, there is no difference in uniqueness support. The only difference is that the (l_i,r_i)

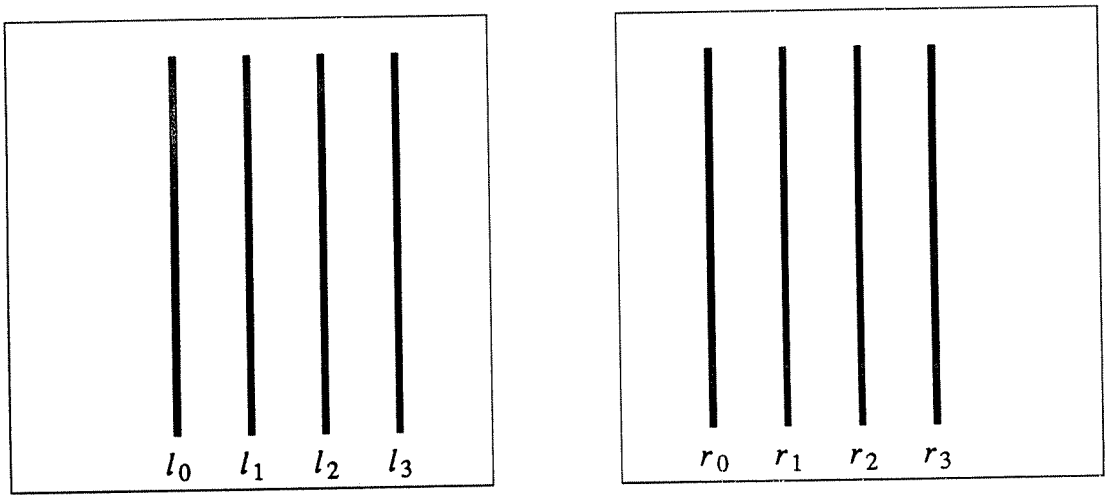


Figure 4.5. Synthetic image for Examples 4.4 and 4.5.

matches receive slightly more disparity gradient support. Normally, this would allow these matches to be accepted. Assume, however, that there are occlusions (there could be other structures not shown, for example), so that the valid matches are (l_i, r_{i+1}). Suppose also that the matches (l_i, r_{i+1}) receive additional detailed match support of *BASE_Both_Sides*. Then the algorithm should accept them as valid. This example demonstrates a case of repetitive image structures where detailed match should be strong enough to force the choice of matches.

The three example images yield five separate possible matching conditions. Proper choices for the parameters should to produce valid choices in each case. In Section 5.2 we will see that the algorithm does make the proper choices in each case. The final justification of the parameter assignments for the constraints comes in testing the algorithm on real images. Results of this are discussed in Section 5.3 and 5.4. The only additional modifications found necessary were slight adjustments to the *BASE_DG* parameter. This is discussed further in Chapter 5.

4.3.4. Discussion of the GSA

Before examining the performance of the algorithm, we briefly discuss some additional issues in the design of the General Support Algorithm. These include the size of the network in terms of the number of nodes, edge detection in the network, the use of conjunctive encoding, matching horizontal edges, and network termination conditions.

4.3.4.1. Network Size

One of the main problems with the General Support Algorithm is the size of the network. A straightforward implementation of the network described above would require

$$k\ p\ m\ N^2$$

match nodes for a given resolution, where k is the number of different compatible edge orientations for a given orientation, m is the number of distinct orientations that the edge detector recognizes, p is the number of distinct disparity values, and the image size is N *by* N. This assumes that there must be a distinct node for every *potential* match. For example, for $k = 3$ (each orientation matches with itself and one orientation on either side of it), $m = 12$ (edges are detected at 30 degrees increments), $disp = 25$, and $N = 256$, approximately 60 million nodes would be required. This is too large.

An alternative implementation, requiring slightly more complicated candidate match determination, significantly reduces the number of nodes required. Assuming that there can be at most one edge found at each pixel, then for any two pixel positions, one from each image, there can be *at most* one candidate match. If one node represented all the potential matches between edges located at the two positions, a significant savings would be realized. Specifically, the total number of match nodes would be reduced to

$$p\ N^2.$$

With $p = 25$ and $N = 256$, as above, only 1.6 million matches are now required at each resolution level. To realize this savings, two problems must be addressed. First, there must be some means of ensuring that only candidate matches with compatible edge orientations can activate the match node. Second, a match node no longer represents any orientation information. Multiresolution and figural continuity connections depend on this information, but the disparity gradient, which defines a large majority of the connections, does not. The solutions to these problems require the use of conjunctive connections.[15] That is, groups of connections are multiplied together before forming the weighted input to a node. Thus, if the output from the edge detector is binary and there is one conjunctive connection for each compatible set of orientations, only compatible edges will activate a candidate match. Similarly, conjunctive connections may be used to limit figural continuity and multiresolution to compatible orientations. An example of candidate match activation and figural continuity support using conjunctive connections is shown in Figure 4.6. With this additional complexity in defining some of the network connections, a significant saving in the number of match nodes is realized.

4.3.4.2. Edge Detection

Another issue we have not addressed completely is edge detection in the lower levels of the network. The General Support Algorithm assumes that the edges are detected and provided as input to the initial phase of matching. In practice we have not attempted the connectionist implementation of an edge detector. The only

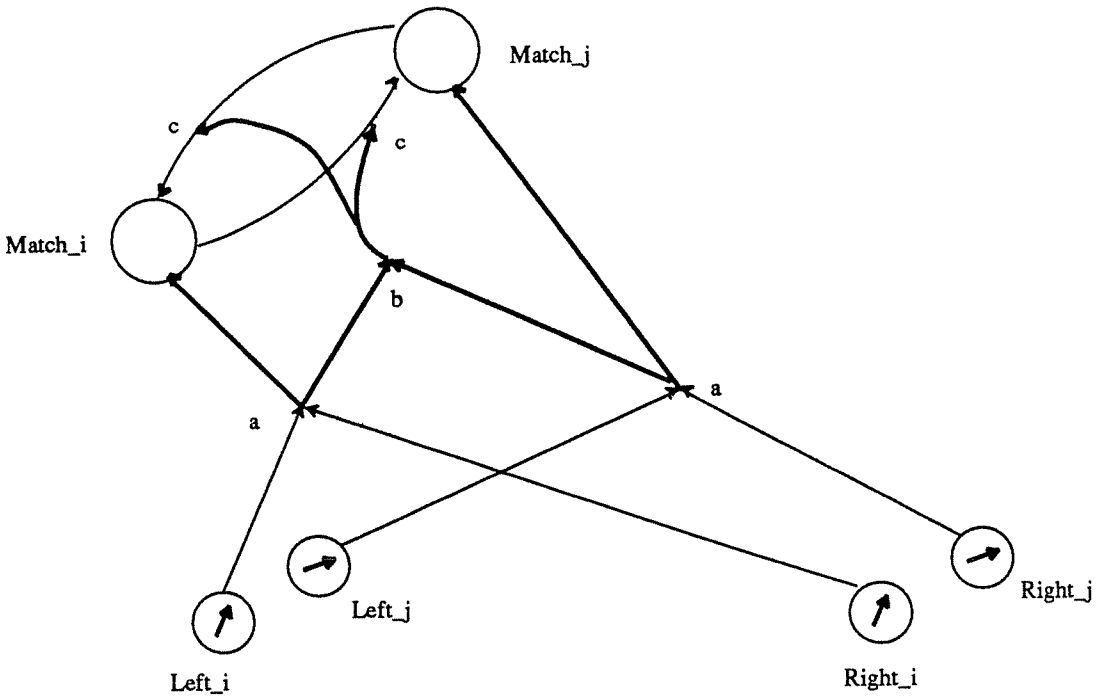


Figure 4.6. Conjunctive connections for candidate match node generation and for figural continuity. Each conjunction labeled *a* determines a candidate match. The conjunction labeled *b* determines that there will be figural continuity input between two nearby candidate matches. The connections labeled *c* allow the figural continuity inputs to pass between the nodes.

assumptions that the General Support Algorithm makes about edge detection is that (1) the edges are relatively sparse, and (2) when an edge occurs at multiple resolutions, it does so at similar positions and orientations.

Not implementing a connectionist edge detector has simplified timing problems in the matching network. Specifically, we assume that the input from the lower levels of the network is all available at the beginning of the matching networks' iterations.

We also assume that the input only effects the initial iteration. The main reason for these assumptions is simplicity. They allow us to study the constraint interactions in the matching network in isolation. A more complete connectionist implementation of a vision system would require that an edge detection network be built and the timing problem be addressed.

4.3.4.3. Conjunctive Encoding

An additional consideration in many connectionist networks is the use of *conjunctive encoding*.[27] This can significantly reduce the number of nodes needed for representing a relatively sparse, multidimensional space. In our case, where the edges are adjacent across rows and columns of the image this savings can not be realized. However, conjunctive encoding of edge nodes can: (1) add redundancy to the system so that the overall response to an edge degrades only gradually with the failure of individual nodes, and (2) allow more accurate position and orientation determination than single node responses. Again, because we are more concerned with the interaction of constraints in stereo matching we have not investigated the use of conjunctive encoding in detail.

4.3.4.4. Horizontal Contours

Thus far in our discussion of the General Support Algorithm we have ignored the problem of horizontal contours. Edges along these contours cause a number of problems in matching: (1) the matches are extremely ambiguous, and (2) slight position errors for the edges can cause a contour to shift rows, thereby causing no

matches to be found. Many algorithms completely avoid matching horizontally-oriented edges. The General Support Algorithm is slightly more flexible than this, matching horizontally oriented edges that are not part of horizontal contour.

In general, the horizontal contours matching problem can not be solved using local matching information. Ideally, the problem could be solved if the horizontal contours produced the same number of edges in a given row in each image. In this case, the interaction of the local constraints would eventually select the valid matches. In reality however, this simple situation rarely occurs. For example, a contour in one image may have one horizontal edge in a given row, while the matching contour in the other image has three. In this case, and in general, there is no clear choice for the correct match. In Chapter 7, we discuss possibilities for handling horizontal contour matching using non-local mechanisms. In Chapter 8, we discuss the extension of the General Support Algorithm to trinocular matching which allows horizontal edges to be matched in the third image.

4.3.4.5. Termination Conditions

A number of researchers have proven that their networks will reach steady-state conditions.[31] These networks usually formulate the search space as an energy surface and then search for the set of activations with minimum energy. This often causes such algorithms to halt at local minima. Simulated annealing is often used to escape from these local minima,[28] but the cost is an increased number of network iterations. We have no proof that the General Support Algorithm reaches a steady-state. There are a number reasons for this: (1) the structure of the network is irregular; networks

that are known to reach a steady-state are highly regular. (2) The algorithm is designed to handle realistic image input instead of random input. The connections and weights are designed to implement the matching constraints. These constraints are based on assumptions about the appearance of objects in images. Thus, given random input, the network's behavior would be unpredictable since the assumptions would no longer be valid.

In the General Support Algorithm we are more concerned with quickly reaching useful matching decisions for as many nodes as possible. Matches that are more ambiguous may take longer, and some matches may gradually reach a steady-state with an intermediate activation value. However, in a real-world vision system, having a high percentage of matches determined in a few iterations is more important than knowing that all matches will eventually be resolved.

4.4. Chapter Summary

In this chapter we have described in detail the General Support Algorithm for stereo matching. The algorithm integrates multiple, reformulated, stereo matching constraints cooperatively and in parallel using the General Support Principle. This principle states that, except for uniqueness, only positive constraint influences are used. In using multiple constraints the General Support Algorithm is able to overcome the weaknesses in the individual constraints and thereby improve the results of matching. The algorithm is implemented in a connectionist network, where the nodes in the network represent potential matches and the connections within the network implement the constraint relations.

CHAPTER 5

Experimental Results

This chapter presents experimental results for the GSA. This involves testing it on three different types of images: (1) simple, hand-drawn synthetic images, (2) random-dot stereograms, and (3) real images. The hand-drawn synthetic images are Examples 4.1 through 4.5 described in the previous chapter. They were used for tuning the parameter values and demonstrating the behavior of the algorithm. Random-dot stereograms are more complex than the hand-drawn synthetic images, but the correct matches are also known exactly. This allows us to easily gather quantitative data on the behavior of the algorithm. Experimenting with real images is the most important test of the algorithm's performance. Unfortunately, these images are noisy and variable, so the correct match for each edge is difficult to determine *a priori*. Thus, it is difficult to gather accurate statistics on the performance of the GSA using real images.

The remainder of this chapter is divided into six sections. Section 5.1 describes the simulation of the networks. Sections 5.2 through 5.4 present the matching results on the three different types of images. Section 5.5 analyzes the results in terms of the guidelines stated in Section 4.1 for an algorithm that integrates multiple constraints. Finally, Section 5.6 summarizes the algorithm and points out some difficulties with it. Most of these difficulties are general problems for any algorithm employing locally-defined matching constraints.

5.1. Simulation of the General Support Algorithm

The first step in testing the GSA is to design an efficient, realistic simulation of it. The main observation used in doing this is that candidate matches are sparsely distributed. Thus, for a given pair of images, we build only those nodes that represent a candidate match and we only define connections between the candidate matches. The result is a subnetwork that is *functionally isomorphic* to the entire network for this pair of images. Details of the construction and parallel execution of this simulation are given in the next chapter.

The simulation subnetwork is built using the results of edge detection. The edges are located using the Marr-Hildreth edge operator.[44] The operator is a two-dimensional Laplacian of a Gaussian ($\nabla^2 G$):

$$\frac{1}{\pi \sigma^2}\left(1 - \frac{r^2}{2\sigma^2}\right) e^{-\frac{r^2}{\sigma^2}}$$

In this operator $w = 2\sqrt{2}\,\sigma$ is the diameter of the central positive region. In the General Support Algorithm three different sizes of this operator, $w = 3, 6, 12$, are used to detect edges at multiple resolutions. The edge detection procedure involves convolving $\nabla^2 G$ with a given image and finding the locations of zero crossings. In practice, a small threshold is used on the magnitude of the zero-crossings, and some minor thinning of non-horizontal edges is necessary.

The results of edge detection are input to the simulation. Included in this input are the position and orientation of each edge and the smoothed intensities on both

sides of the edge. The simulation uses this input to determine the candidate matches and build the connections between them. (No matches are allowed for horizontal edges along contours of length two or more in a given row in an image.) The initial activation for each match is set at a base level, plus any additional support through detailed match. Then the simulation is allowed to run for either a given number of iterations (usually a max of approximately 16) or until only a small percentage (usually approximately 1%) of the matches have outputs in the range 0.25 to 0.75. Finally, the simulation was run using the following parameter values: $BASE\_Coarse\_to\_Fine = 0.225$, $BASE\_Fine\_to\_Coarse = 0.1$, $BASE\_FC = 0.15$, $BASE\_DG = 0.08$, $\delta = 0.12$, $\beta = 0.5$, $BASE\_Both\_Sides = 0.02$, and $BASE\_One\_Side = 0.12$. The only value that was ever changed was $BASE\_DG$. It varies with the density of the matches. This is discussed further below.

In addition to the simulation of the matching network, we developed a number graphics tools for use in displaying the images, observing the behavior of the network and collecting matching statistics. These are described where appropriate in the sections below.

5.2. Hand-Drawn Synthetic Images

The General Support Algorithm works on the synthetic examples in Section 4.3.3. It was successful in finding the correct matches for Examples 4.1, 4.2, 4.4, and 4.5, and it correctly rejected all candidate matches for Example 4.3. The times for all of the valid matches to reach saturation were: 8 iterations for Example 1, 5 iterations for Example 2, 6 iterations for Example 4, and 5 iterations for Example 5. For these

simulations the value, $BASE\_DG = 0.09$, was used.

As a demonstration of the behavior of the network, consider Example 4.4. Recall that each bar in the left image, l_i can match r_i and r_{i+1} in the right image. Thus, edges on both r_0 and l_3 have only one candidate match; all other edges have two matches each. It should be obvious that for each i, l_i and r_i form the valid matches. However, the only way the network can resolve this is through uniqueness. Specifically, matches between edges on bars l_0 and r_0 only receive uniqueness inhibition from competing matches for edges on l_0. Similarly, matches between edges on l_4 and r_4 only receive uniqueness inhibition from other matches for edges on r_4. All other matches have two-sided uniqueness inputs. Thus, the matches for l_0 and r_0, and for l_4 and r_4 pull the remainder of the matches into the proper correspondence.

The behavior of the matching network on this example is seen in detail in Figure 5.1. The rows of the figure represent time units 0 through 6. The four columns represent the matches for a sample edge for bars l_0 through l_3. In a given time unit, the candidate matches for an edge are shown by a bar graph ("mini-graph"). The horizontal axis of each mini-graph is the disparity of the match, while the vertical axis represents the output value from a match node. (These mini-graphs for tracing the output from each node are part of the graphic tools used for understanding the behavior of the networks.) Matches between l_i and r_i are represented by disparity 10. Matches between l_i and r_{i+1} are represented by disparity 0. After time unit 2 it was clear that the disparity 10 matches for l_0 and l_3 were going to win out. Later, after time unit 3, the disparity 10 matches for l_1 and l_2 were ahead of their competitors, but

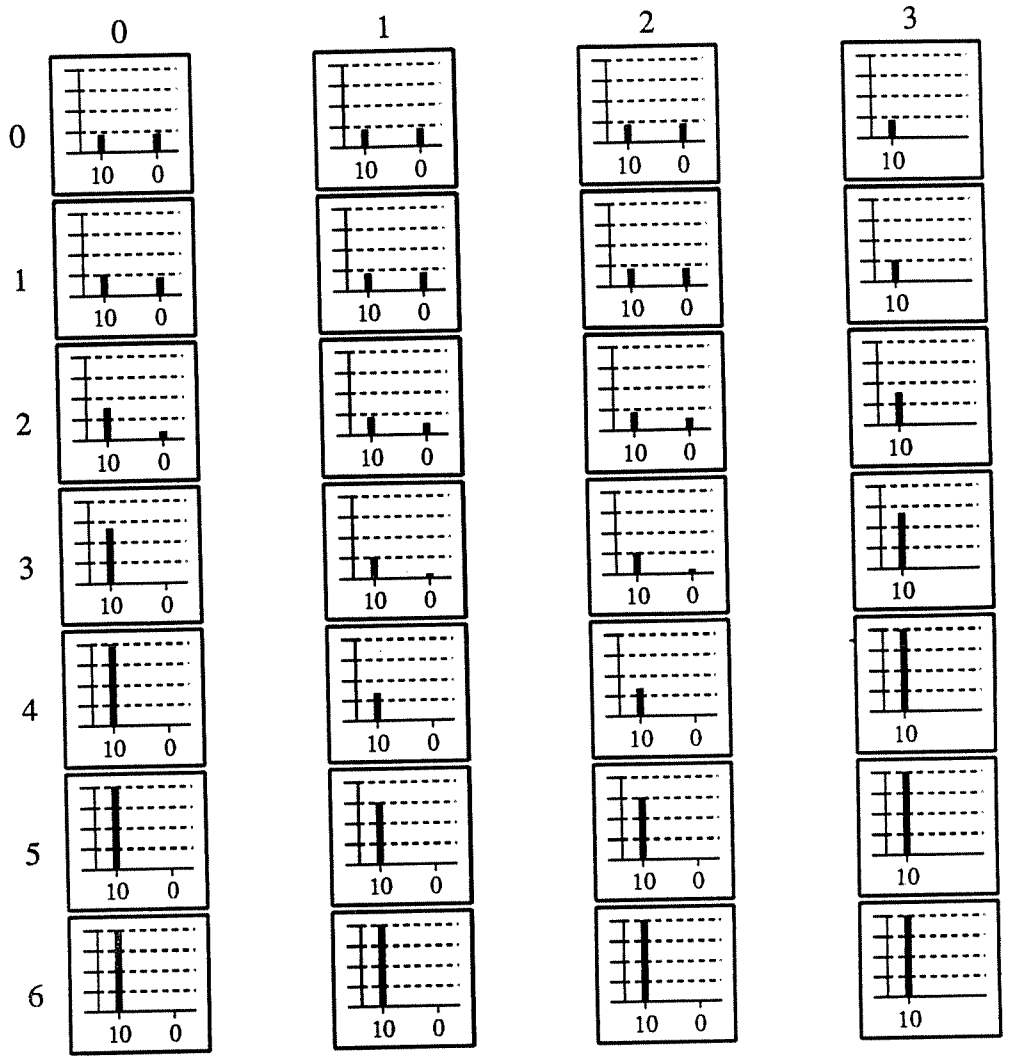


Figure 5.1. Iterations of the General Support Algorithm on Example 4.4. See the text for details.

notice how even though they were clearly ahead, the output values were still low. This is an example of the coalition effect discussed in Chapter 4. The weights were set so that the output values remain low until a coherent set of matches won out. Once this occurred, the matches in the coalition quickly drove each other up to 1.0. This occurred from time units 4 through 6 in the example.

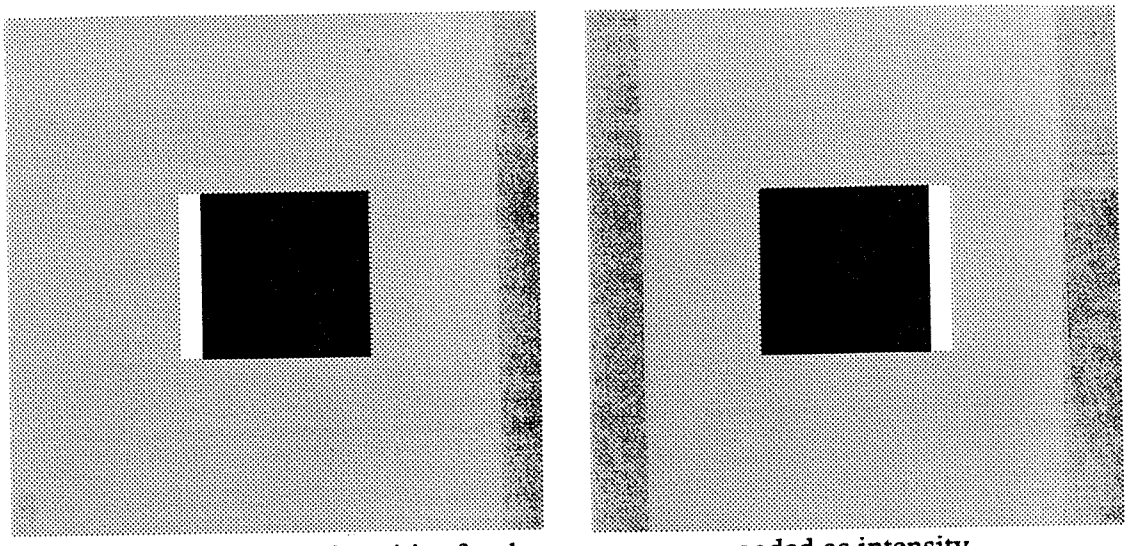
5.3. Random-Dot Stereograms

Random-dot stereograms have frequently been used for testing matching algorithms.[35] They are useful for studying matching for two main reasons: (1) there are no monocular depth cues that might assist in the stereo process, and (2) the correct disparity is known for every point. The latter gives an objective measure of matching accuracy that is not available for real images.

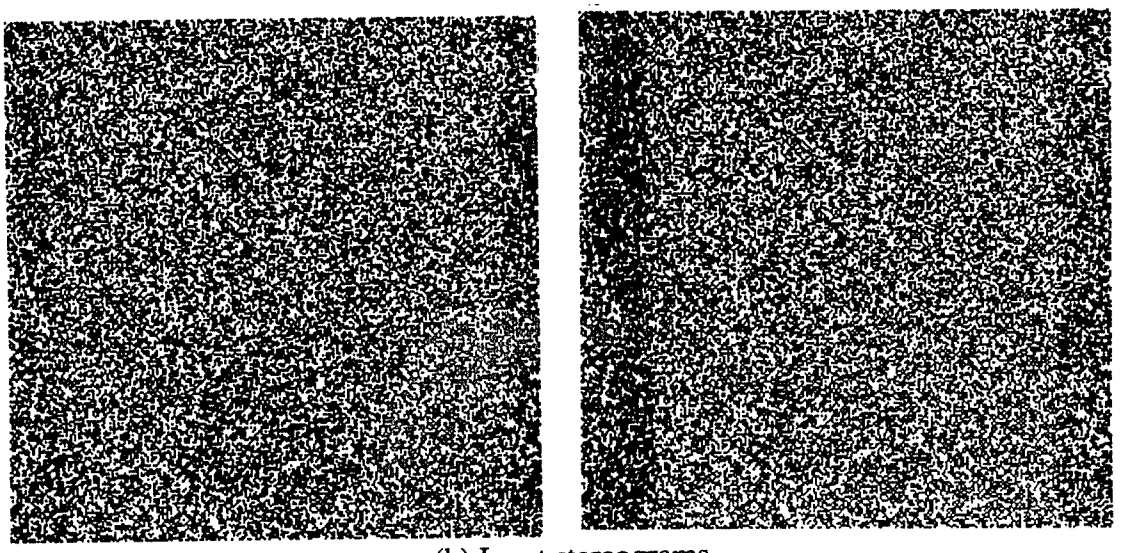
In generating a stereogram, surfaces of varying disparity are randomly colored dark or light. The resulting textures are projected onto the left and right images based on the disparity of the surface. In this way, each image appears as a collection of random dots, and therefore contains no structural information. It is only as a result of matching that any meaningful information is obtained.

Our method of testing stereograms was as follows. All stereograms used a series of planar surfaces. 256 by 256 images were first generated with 50 percent dot density. After this the stereograms were treated as normal inputs to edge detection. The network was run using $BASE\_DG = 0.04$. The results of matching were recorded in two ways. First we generated a "disparity image". That is, for each edge with an accepted match in a given image, the disparity of the match was recorded as an intensity. The second method of data collection was to automatically compare the disparities of the matches to the known correct disparities. A match was counted as correct if its disparity was within ± 1 of the surface disparity.

We present the results of matching for four stereograms. The first is shown in Figure 5.2. Part (a) of the figure gives the disparity of the surfaces encoded as

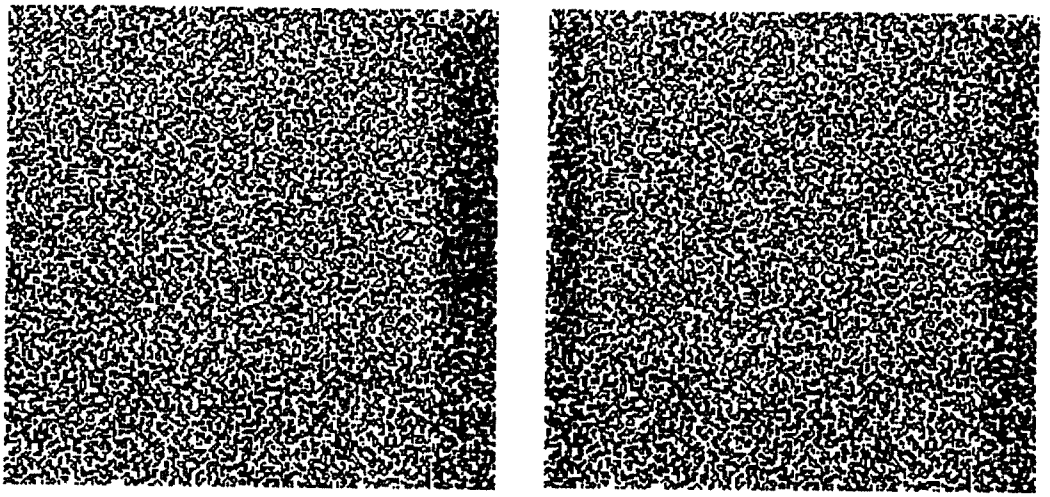


(a) Correct disparities for the stereogram, encoded as intensity.

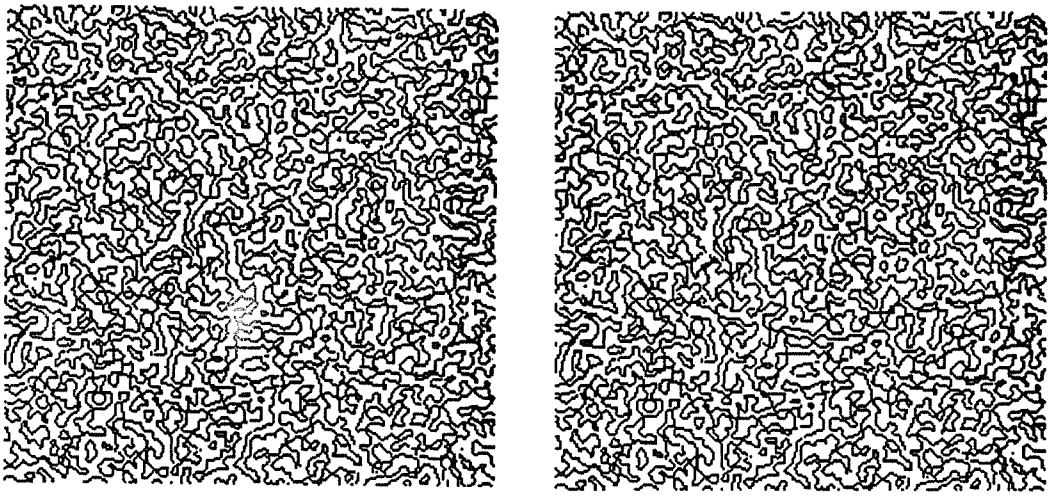


(b) Input stereograms.

Figure 5.2. First random-dot stereogram example.

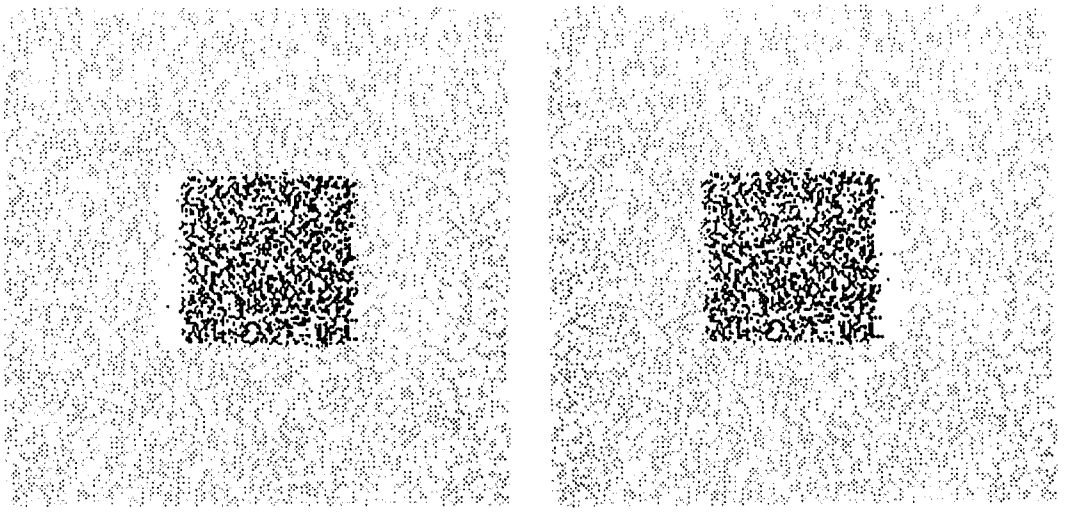


(c) Finest resolution edge detection results (level 0).

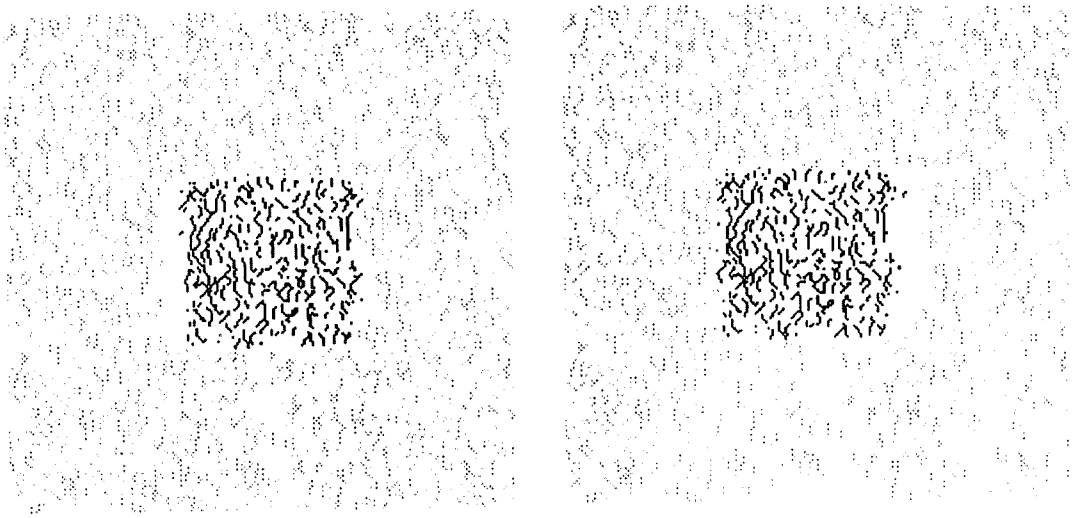


(d) Intermediate resolution edge detection results (level 1).

Figure 5.2 (continued).



(e) Finest resolution disparity image (level 0) resulting from matching.



(f) Intermediate resolution disparity image (level 1) resulting from matching.

Figure 5.2 (continued).

intensity. Darker points are nearer to the viewer. White patches indicate locations that appeared in the given image, but were occluded in the other image. No matches should be found for these points. The stereogram represents two surfaces, one at disparity 0 (the background), the other at disparity 10 (closer to the viewer). Part (b)

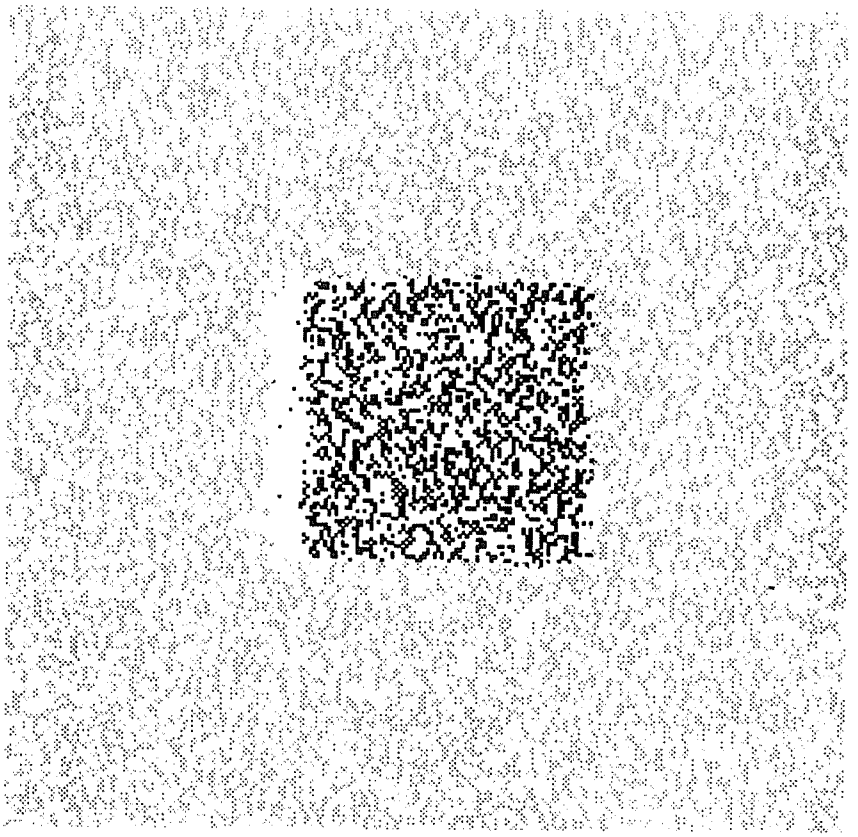


Figure 5.2 (g). Large scale image showing the left fine resolution disparity image. The remaining results will be given in this format. See page 86 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

of the figure shows the actual stereogram. Parts (c) and (d) show the results of edge detection for the two finer resolutions (fine resolution is level 0, intermediate resolution is level 1). Parts (e) and (f) give the disparity images that resulted from matching for these two resolutions. Finally, part (g) shows the left disparity image for the finest resolution in large format. The disparity results for the remaining images will be shown in this manner. Table 5.1 summarizes the statistical results of matching

	Correct No Match	Incorrect No Match	Correct Match	Incorrect Match	Percent Correct	
Left 0	309	13	17,617	53	99.6	
Left 1	210	53	8,661	29	99.1	
Right 0	310	11	17,613	56	99.6	
Right 1	198	46	8,658	28	99.2	

Table 5.1. Matching results for stereogram in Figure 5.2.

for levels 0 and 1. (The position of the edges varied too much at level 2 for the statistical method we used.) The second through fifth columns of this table represent, in order, correctly finding no match for an edge, incorrectly finding no match, (0.5 was used as the cut-off between valid and invalid matches) finding only the correct match, and finding an incorrect match. The sixth column represents the percentage of correct matching decisions. A correct decision is either finding the correct match for an edge, or correctly rejecting all matches for an edge. The final set of statistics is given in Figure 5.6. It shows the percentage of matches whose output was in the intermediate range 0.25 to 0.75 as a function of iteration number. (All four stereograms used are graphed in this figure.) Matching ended after six iterations since fewer than 1% of the nodes were in the intermediate activation range.

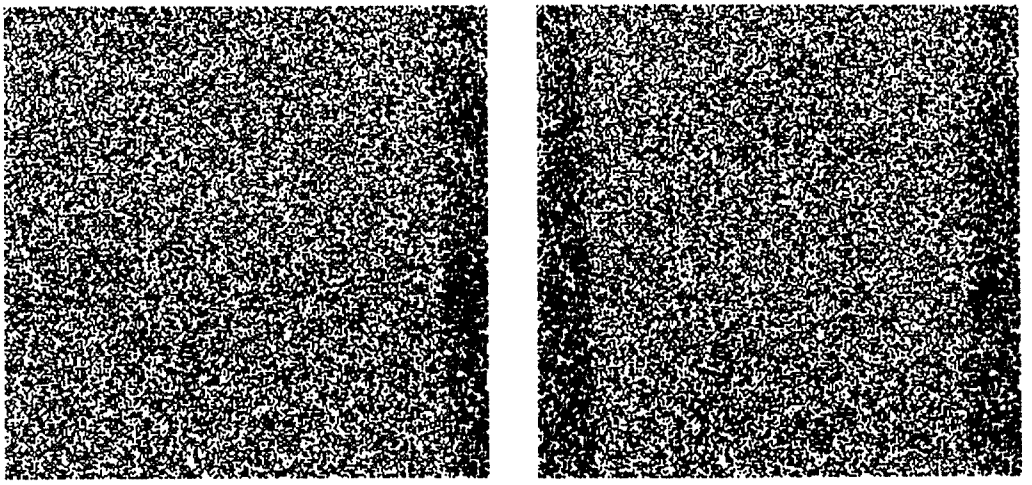
The algorithm performed extremely well on this example stereogram. The occlusion was significant (10 pixels), so there was little erroneous support across the occluded boundaries. In addition, there was little ambiguity in the images since there was no noise and the surfaces were smooth.

It is important to note that there were very few cases in which multiple matches were found for a given edge. We divided multiple matches into two categories. The

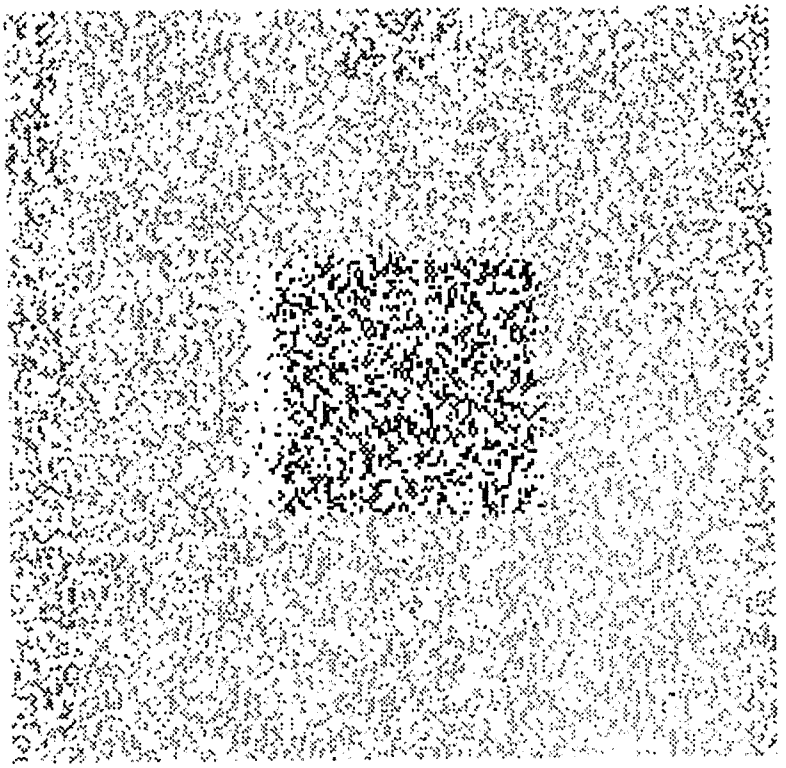
first included cases where there were two adjacent edges in the other image (this happened only rarely due to the edge thinning and horizontal match elimination procedures). In the right image at level 0 there were 16 edges with this type of multiple match. These were not included in the statistics. The second type of multiple match was considered an error. These involved edges in the other images that were not adjacent. There were only two edges with this type of multiple match in the left image at level 0. This represented a negligible percentage (%0.2).

Finally, a significant percentage of the edges at level 0 (23.3%) had no candidate matches. These edges were not included as part of the matching statistics. They were either part of a horizontal contour or near the boundaries of the images. Thus, horizontal contours were a significant problem in terms of the percentage of edges that were matched.

The second stereogram example used the same textured surfaces as the previous example, except that noise was introduced when the surfaces were projected onto the images. Specifically, each point in an image had a 2% chance of being changed (either from white to black or black to white). Note that these errors were introduced independently for the left and right images. The stereograms produced by this procedure are shown in Figure 5.3 (a). The results of level 0 matching are shown in Figure 5.3 (b). Table 5.2 summarizes the matching results. The matching completed after 14 iterations. The additional time was required to resolve ambiguous matches, or erroneous matches receiving a significant amount of noise support. Figure 5.6 graphs the percentage of matches with intermediate outputs as a function of the



(a) Stereogram.



(b) Level 0 left disparity image.

Figure 5.3. Stereogram for the square with 2% noise added to each image. See page 88 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

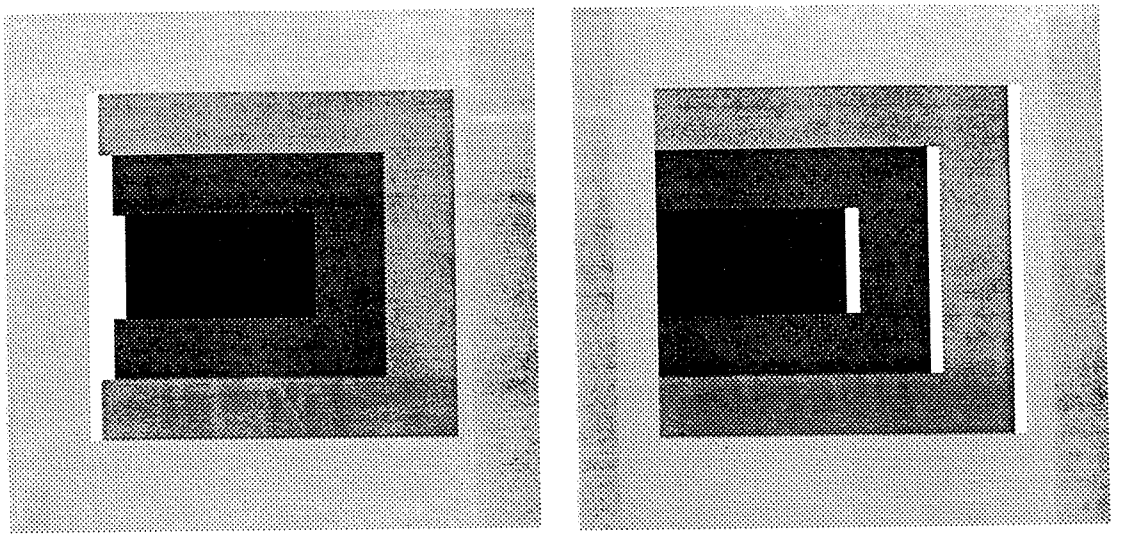
	Correct No Match	Incorrect No Match	Correct Match	Incorrect Match	Percent Correct
Left 0	2,590	75	14,204	642	95.9
Left 1	1,327	75	6,406	342	94.9
Right 0	2,587	101	14,215	648	95.7
Right 1	1,329	92	6,402	343	94.7

Table 5.2. Matching results for stereogram in Figure 5.3.

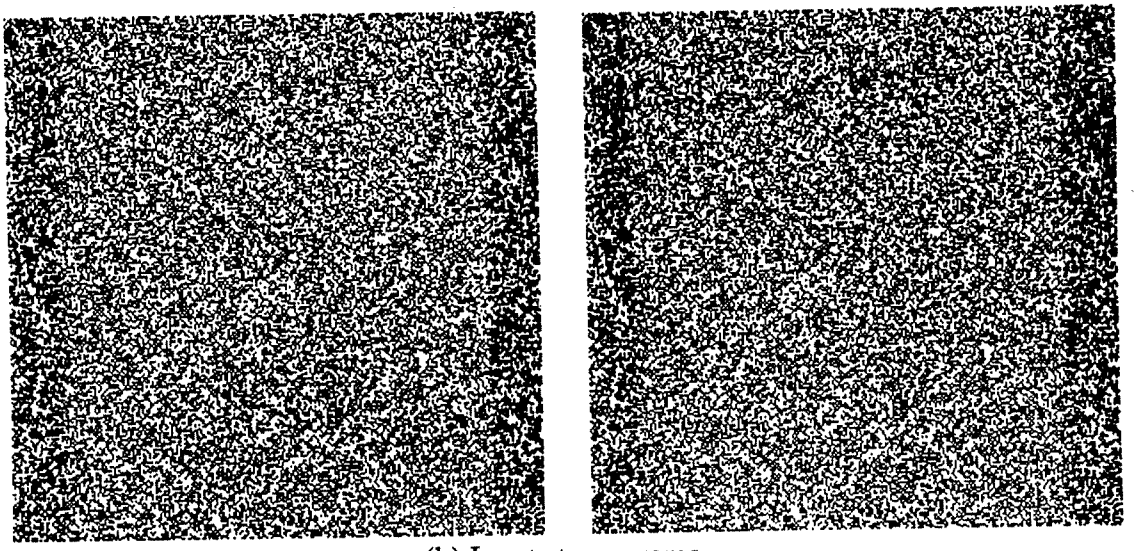
iteration number.

Even though 2% of the points in each image involved an error, nearly 96% of the edges at level 0 had the correct matching decision. It is interesting to note that of the 642 edges with an incorrect match in the left image at level 0, the correct match was not available for 540 of them. Because of the noise in the image there was likely to be more random support. Those noise matches that received a large coalition of support could be accepted. However, 2,590 edges with no correct matches had none accepted by the algorithm. Finally, it is apparent from the disparity image that the locations with noise matches were sporadic. This included additional erroneous matches in the occluded regions.

Our third stereogram is presented in Figure 5.4. Part (a) of the figure gives the disparity of the surfaces encoded as intensity. The background surface had disparity 0; the other surfaces had disparities 6, 12 and 18 pixels. Note that in the right image the three non-zero disparity surfaces aligned. This gave rise to a significant occlusion in the left image. Part (b) of the figure shows the actual stereograms. Part (c) gives the level 0 disparity image resulting from matching. The matching ended after 8 iterations. Table 5.3 summarizes the statistical results of matching for levels 0 and 1.

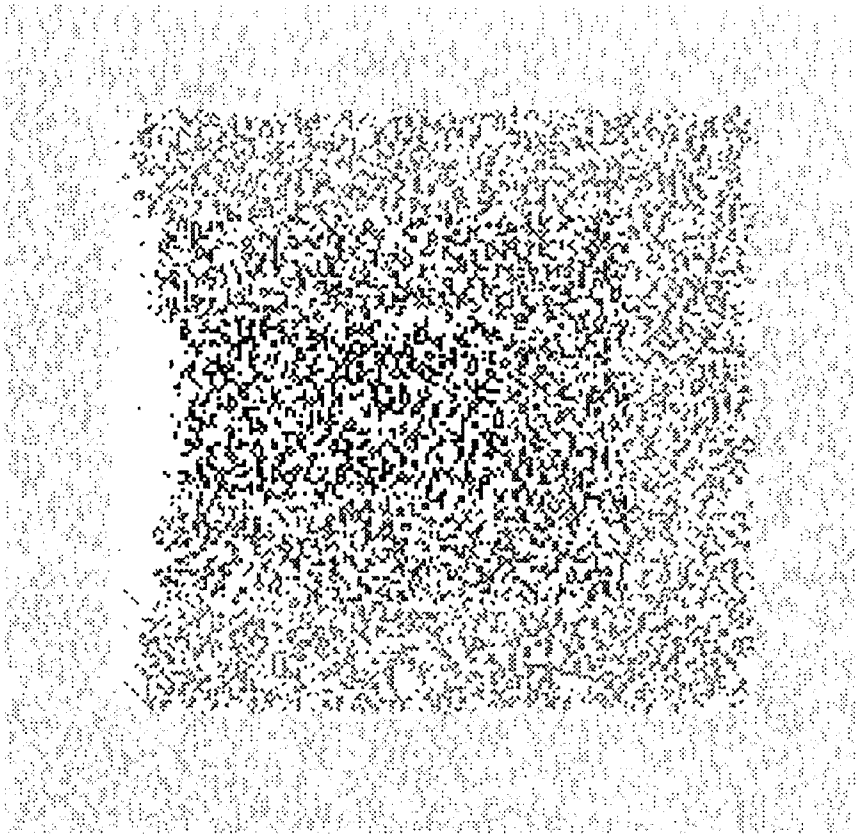


(a) Disparities of the surfaces, encoded as intensities.



(b) Input stereograms.

Figure 5.4. Stereogram with multiple layers.



(c) Level 0 disparity results, encoded as intensity.

Figure 5.4 (continued). See page 90 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

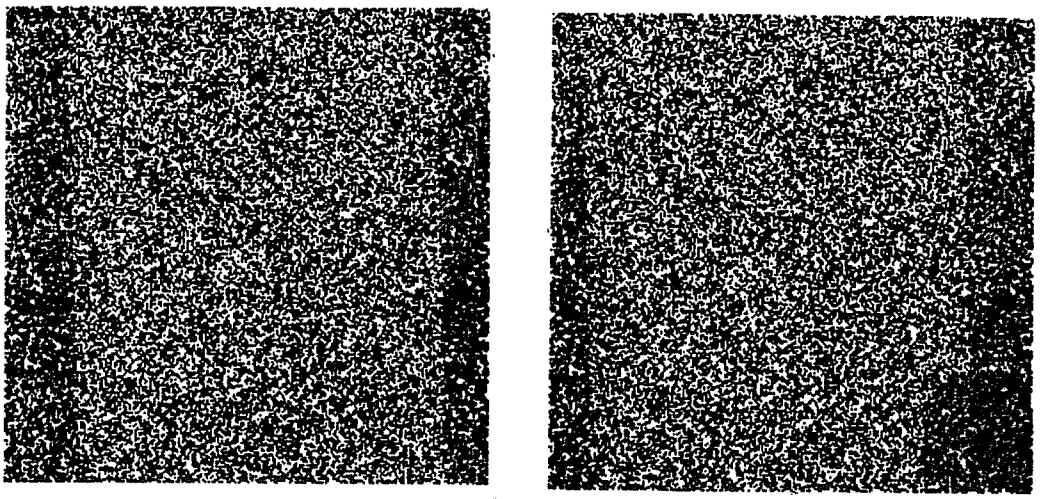
	Correct No Match	Incorrect No Match	Correct Match	Incorrect Match	Percent Correct
Left 0	872	28	16,857	281	98.3
Left 1	552	48	7,924	153	97.7
Right 0	809	35	16,858	288	98.2
Right 1	507	49	7,923	154	97.7

Table 5.3. Matching results for stereogram in Figure 5.4.

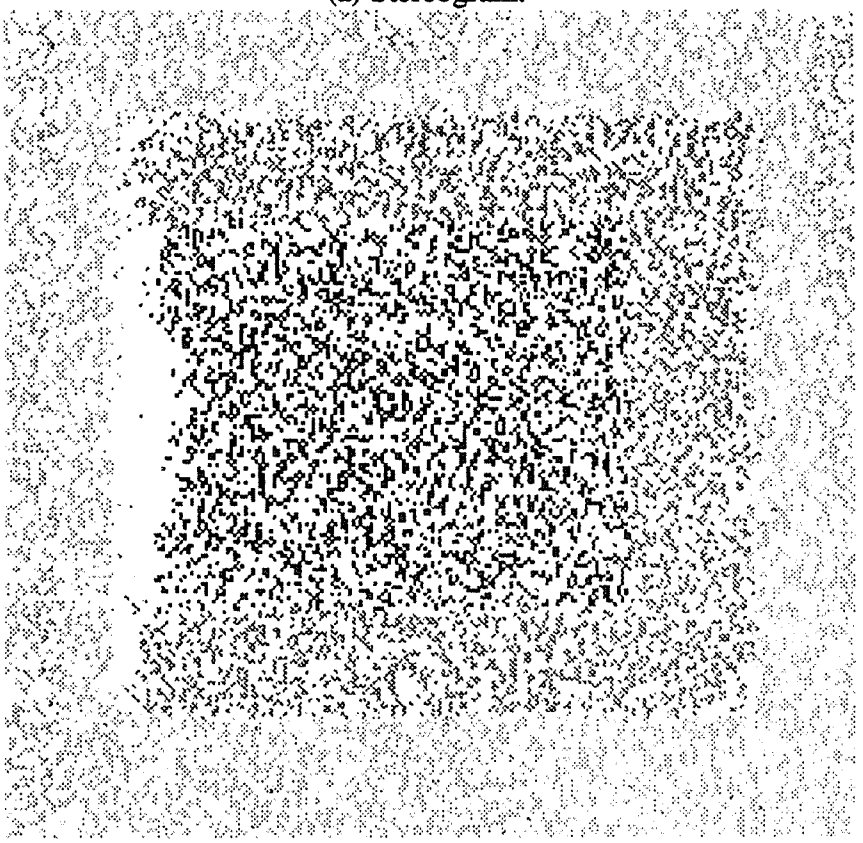
Figure 5.6 shows the percentage of matches whose output was in the intermediate range 0.25 to 0.75 as a function of iteration number. Matching stopped after 8 iterations.

Once again, there was a large percentage of correct matching decisions. Many of the errors involved occluded regions. For example, 64 of the 281 incorrect matches at level 0 in the left image were for occluded edges (for 538 occluded edges it correctly found no matches). The numbers were higher in the right image, 115 of 288. The main reason for the differential between the left and right images in the number of occluded edges with matches is that there was a larger occluded region in the left image. Support between matches in an occluded region is essentially random. More coherent support comes from outside the occluded region. Due to the large occlusion in the left image in Figure 5.5 this support was generally further away from a given match and was therefore weaker. In addition to the erroneous matches for the occluded edges, there were also errors near step changes in disparity that did not directly involve occlusions.

The final stereogram used the same surface as the previous example. Errors were introduced into the images with 1% probability. The stereograms produced by this



(a) Stereogram.



(b) Level 0 disparity image resulting from matching.

Figure 5.5. Stereogram with multiple layers. 1% noise was added. See page 93 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

procedure are shown in Figure 5.5 (a). The results of level 0 matching are shown in Figure 5.5 (b). The statistical results are summarized in Table 5.4. Figure 5.6 graphs the percentage of matches with intermediate outputs as a function of the iteration number. The matching required 12 iterations. In spite of the combination of noise and a number of occlusions, the algorithm worked well for this stereogram. The appearance of the resulting disparity image did not vary significantly from the image where no noise was introduced.

In general the algorithm worked very well on random-dot stereograms even though it was not specifically designed to do so. These results are comparable to other algorithms that make more explicit use of the dense, relatively uniform distribution of the matching points in the images. Those errors that we found involved differences between the images due to either noise or occlusions.

5.4. Real Images

Matching real images represents the most important test for the General Support Algorithm. Unfortunately, evaluating the results is also more difficult than either

	Correct No Match	Incorrect No Match	Correct Match	Incorrect Match	Percent Correct
Left 0	2,208	73	14,897	580	96.3
Left 1	1,133	100	6,613	343	94.7
Right 0	2,017	83	14,909	585	96.2
Right 1	1,104	85	6,613	337	94.8

Table 5.4. Matching results for stereogram in Figure 5.5.

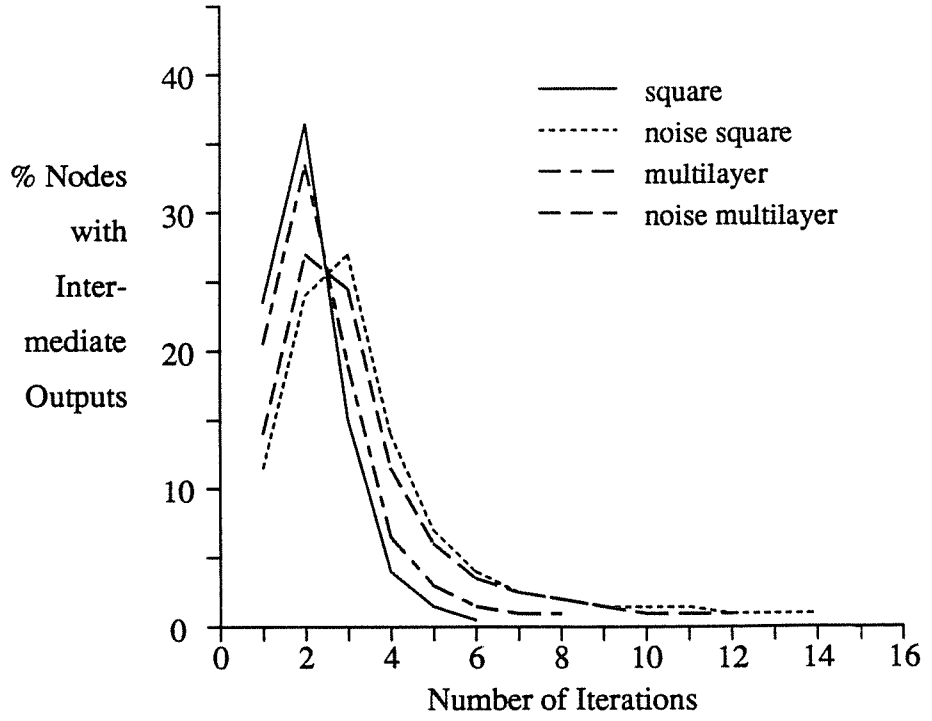
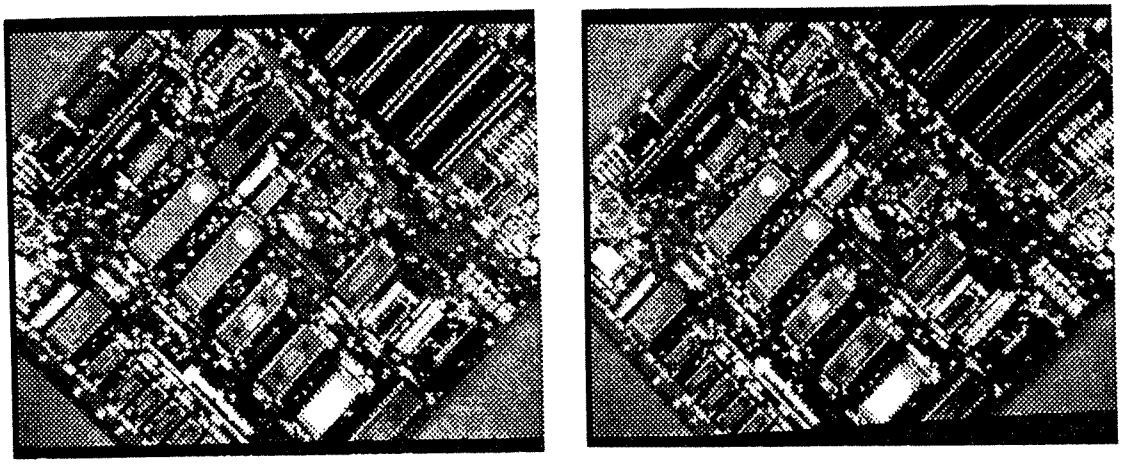
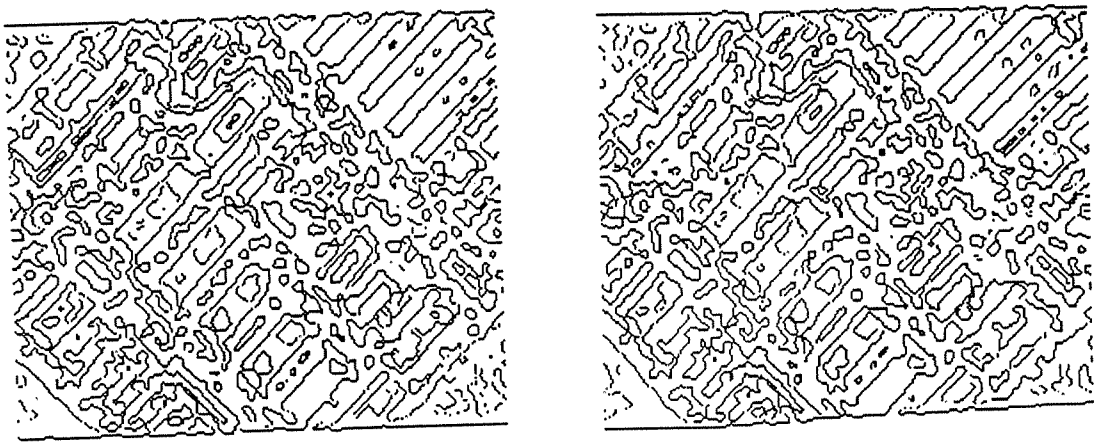


Figure 5.6. Percent of matches with intermediate output values as a function of the number of iterations. Graphed for the stereograms.



(a) Input images.



(b) Intermediate resolution edge detection results (level 1).

Figure 5.7. First real image example: Apple processor board.

(c) Intermediate resolution disparity image (left image) resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

Figure 5.7 (continued). See page 102 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

random-dot stereograms or synthetic images. This is mainly because the correct disparities for points in the images are not known.

The simulations for real images were run in the same manner as for the stereograms. The matching for each stereo pair was allowed to run for up to 16 iterations or until fewer than 1% of the matches had outputs in the intermediate range. The parameter *BASE_DG* was varied between 0.045 and 0.085 depending on the

density of the edges in the images. In collecting data for the results, we attempted to follow similar procedures as for random-dot stereograms. This involved generating disparity images and collecting matching statistics. However, since the ground truth disparity was unknown, we used subjective judgment to determine whether the algorithm made correct matching decisions. The images were graphically displayed and, for each edge, the candidate and accepted matches were highlighted. Statistics on correct decisions were then gathered using the following judgment criteria: First, the disparity of a match was compared to an approximately known disparity for the region of the image. Second, the two images were carefully examined to discern, subjectively, if matching edges appeared to be the same physical edge. This was mostly based on the structure of the images. A limited percentage of the edges were actually checked, and only resolution level 1 data were analyzed (level 0 edges were too dense). This procedure gave a rough quantitative measure of how well the

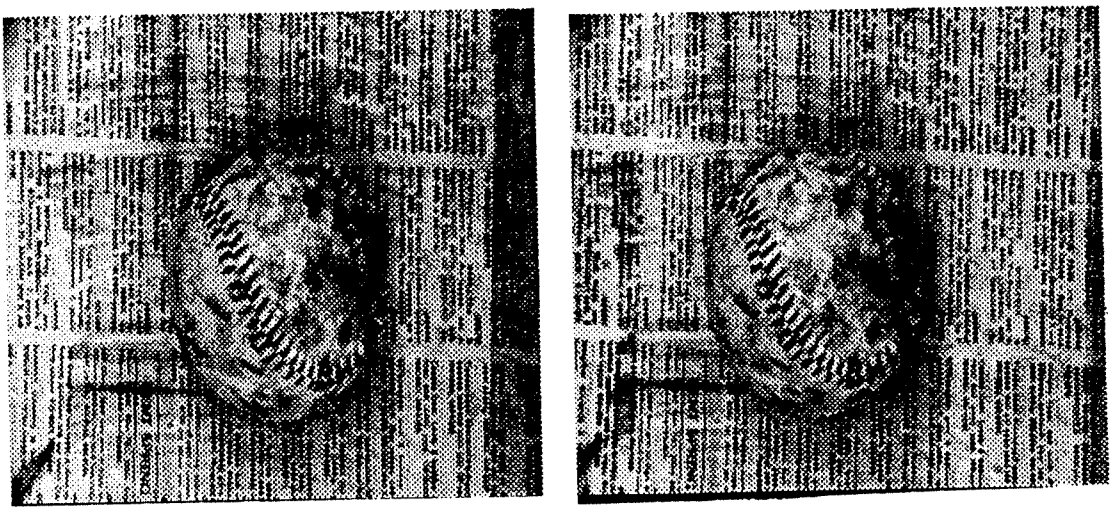
	Level 1			% Mid Range After Time			
	Number Examined	Percent Correct	BASE_DG	6	9	12	Total Iterations
Apple	514	99	0.07	3	1		10
Ball	513	98	0.045	4	2	1	13
Books	719	95	0.08	6	2	1	16
Fruit	526	93	0.085	10	4	2	16
Pentagon	565	97	0.05	6	3	2	16
Renault	449	98	0.08	6	3	2	16
Rocks	602	98	0.065	8	3	2	16
Ruts	623	99	0.085	10	4	2	16
Sandwich	493	98	0.085	13	6	4	16

Table 5.5. Matching Statistics for Real Image Pairs.

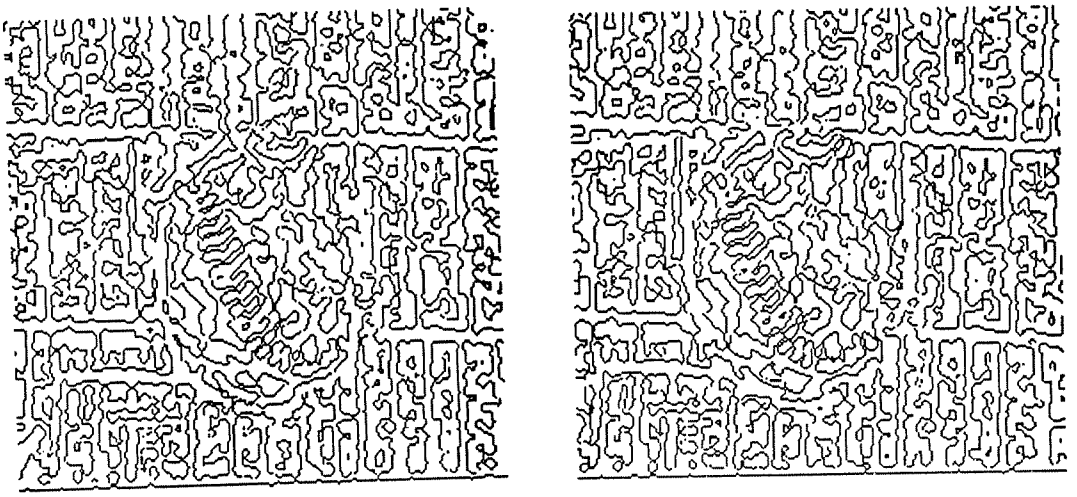
algorithm performed, and allowed us to examine the matching results in detail to observe where the algorithm succeeded and failed.

We tested the algorithm on the images in Figures 5.7-5.15. Each of the figures show the images, intermediate results of edge detection (level 1) and level 1 disparity images for the other images. Level 1 results are shown because they were used for results analysis. The statistics for all of the real image examples are summarized in Table 5.5. The second and third column of this table give the number of edges examined and the percent of correct matching decisions respectively. The fourth column shows the value of the parameter BASE_DG that was used. The fifth through seventh columns of the table give the percent of matches in the intermediate range after the sixth, ninth and twelfth iterations. The final column shows the total number of iterations required. Finally, the graphs in Figures 5.16 and 5.17 show the number of intermediate nodes as a function of the number of iterations for the images. The results of matching for each of the images are discussed below.

Overall, the results show a high rate of correct decisions. In addition, there are several other general comments to note about the results. First, the disparity images are sparser than the edge images. This is because unmatched edges do not appear in the disparity images. These are usually horizontal edges, but they may also be occlusion edges. Second, the graphs shown in Figures 5.16 and 5.17 indicate widely varying percentage values and peaks. The height of the peaks varied with the percentage of matches that were accepted by the algorithm for the images. For example, the apple image had 46% acceptance, the rocks image had 19% acceptance,



(a) Input images.



(b) Level 1 edge detection results.

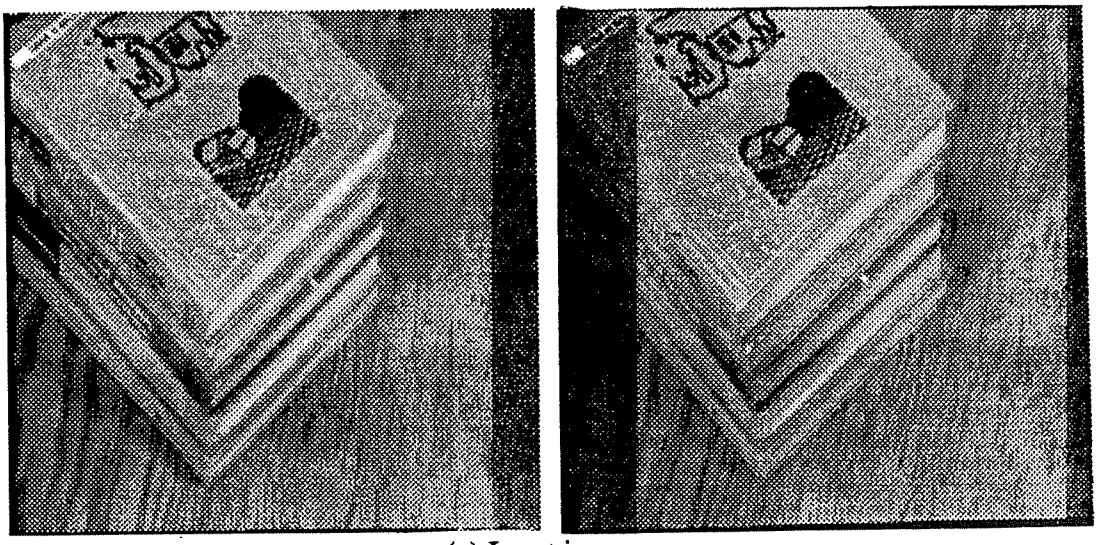
Figure 5.8. Stereo pair with a baseball on top on newspaper.

(c) Level 1 disparity images resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

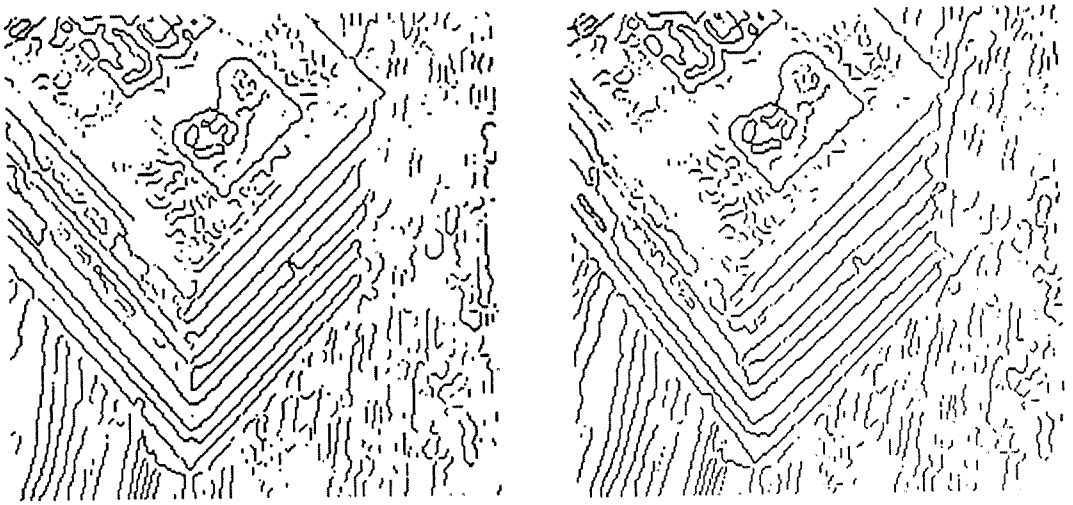
Figure 5.8 (continued). See page 104 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

and the sandwich image had 43% acceptance. The main reason for the lower peak in the sandwich image is that matches were sparser than in the apple image.

The matching for the apple image in Figure 5.7 was extremely smooth and fast. The appearance differences between the images were slight. The errors that appeared were sporadic and on the outer edges of the images.

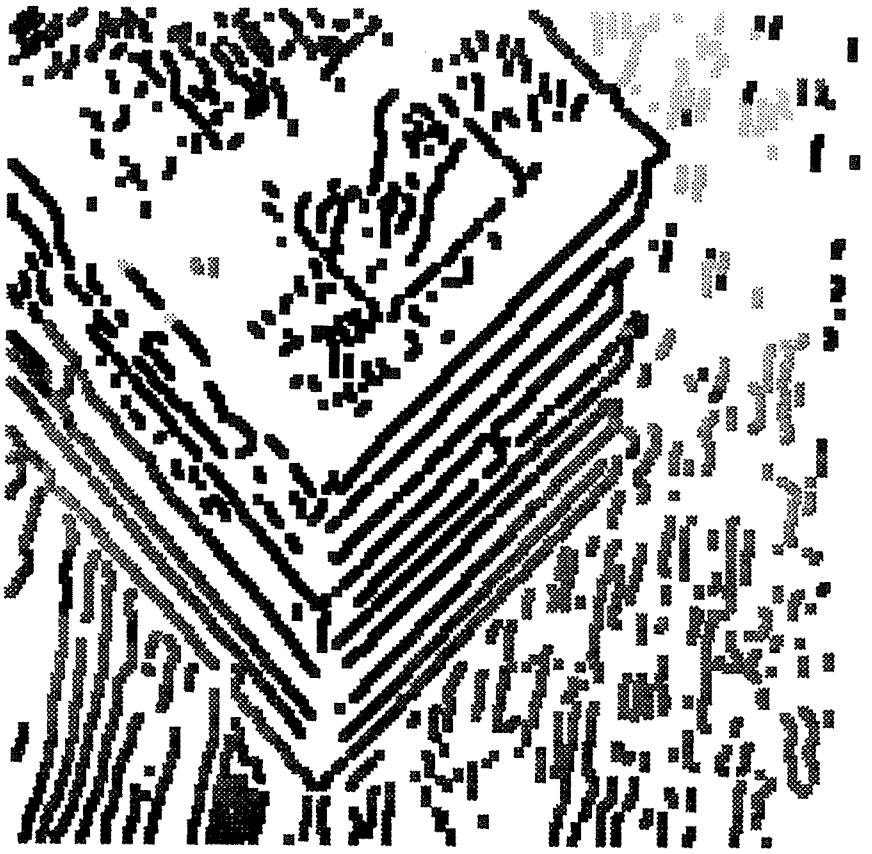


(a) Input images.



(b) Level 1 edge detection results.

Figure 5.9. Stereo pair of a stack of books.



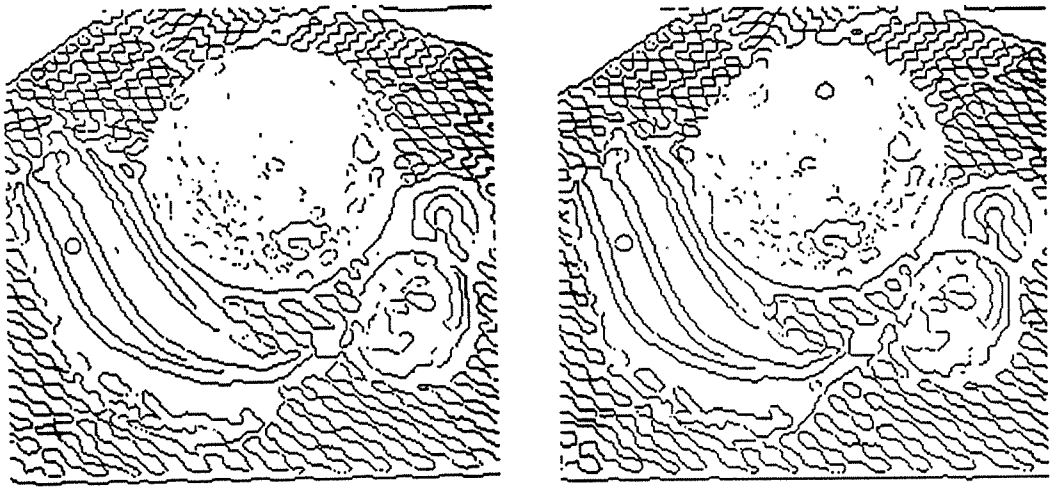
(c) Level 1 disparity image resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

Figure 5.9 (continued). See page 106 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

In Figure 5.8 the disparity difference between the baseball and the newspaper behind it was approximately 12 pixels. Ideally there would be no matches accepted for the occluded regions. These appeared to the left of the ball in the left image and to the right of it in the right image. Even though the matches were sparser in these regions, there were still some spurious matches. In addition, there were some other minor errors above and below the ball where there were no occlusions, but where

there were discontinuities in disparity. The other matching errors for this pair were sporadic, and many of them appeared near the image boundaries.

(a) Input images.



(b) Level 1 edge detection results.

Figure 5.10. Stereo pair of fruit on a tablecloth.

(c) Level 1 disparity image resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

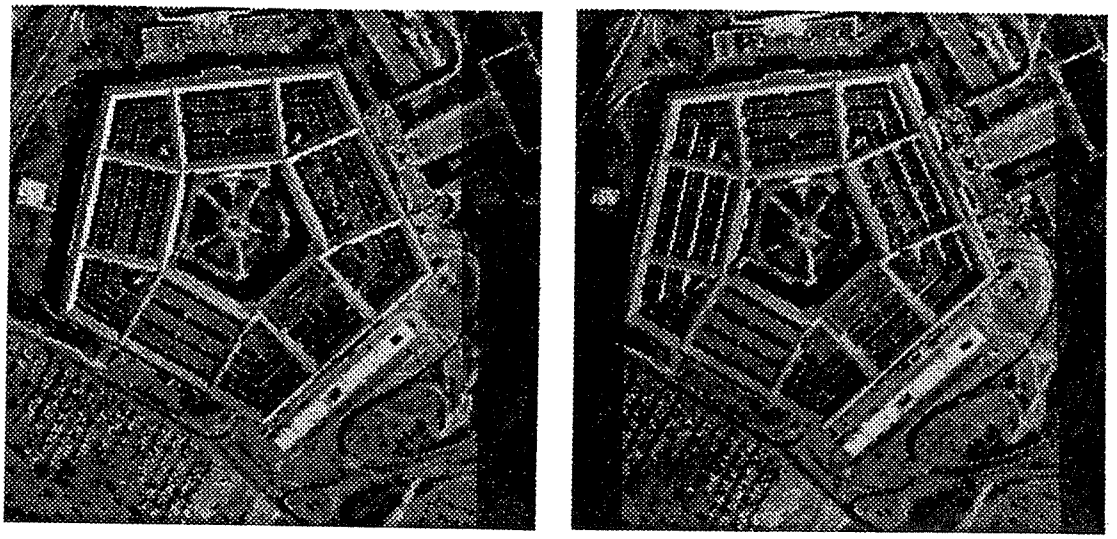
Figure 5.10 (continued). See page 107 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

In Figure 5.9 there were significant differences between the contours appearing in the upper-right of the two images. There were similar problems in the upper half of the image on the far left. These differences caused severe problems for the constraints since contours occurred in one image but not in the other. This caused arbitrary matches between the edges along contours. In the lower part of the images, where the

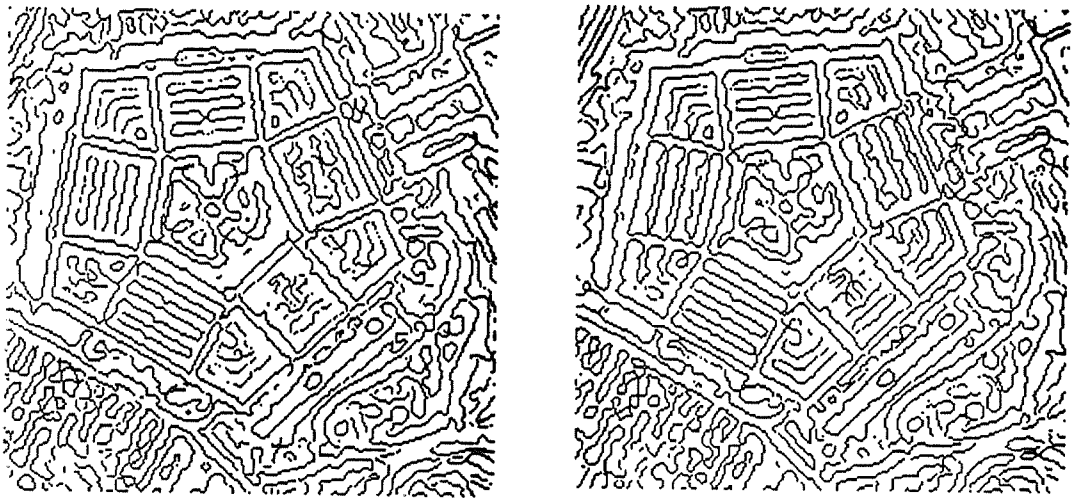
data were more consistent, almost all correct matching decisions were made.

The performance on the fruit image in Figure 5.10 was the worst of the real images. The systemic errors occurred in three places: on either side of the boundary of the grapefruit, and along the right border of the images. These errors were primarily caused by the combination of a periodic image structure and an occluding surface (the boundary of the image is effectively an occluding surface). Because of the periodic structure, the edges near the boundaries in each image matched each other with strong support. However, these were incorrect matches: one of the edges was occluded and should have had no match, while the other should have had a different valid match. Unfortunately, there was no local information available to force the correct decision. In addition, the incorrect matches received only one-sided uniqueness inhibition, and therefore they tended to be accepted. In the areas away from the boundaries, the correct matches were gradually accepted. In other areas of the image, the matches for the periodic structure were correctly resolved through the combined interaction of the constraints. This example is discussed further in Section 7.2 where a potential solution to the general problem of matching in periodic image regions is offered.

The errors for matching the pentagon image in Figure 5.11 were relatively sporadic. The most common cause of the errors was the structural differences between the images. This is apparent in examining the internal patterns of the building itself. In addition, the blank areas in the images were where the edges were mostly horizontal.

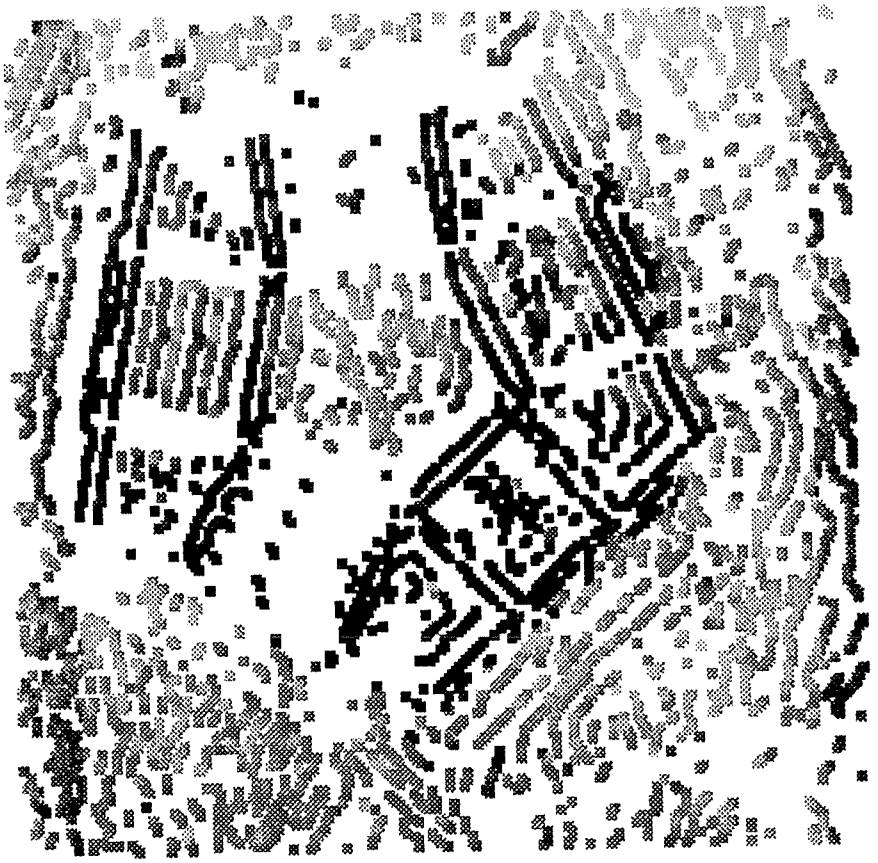


(a) Input images.



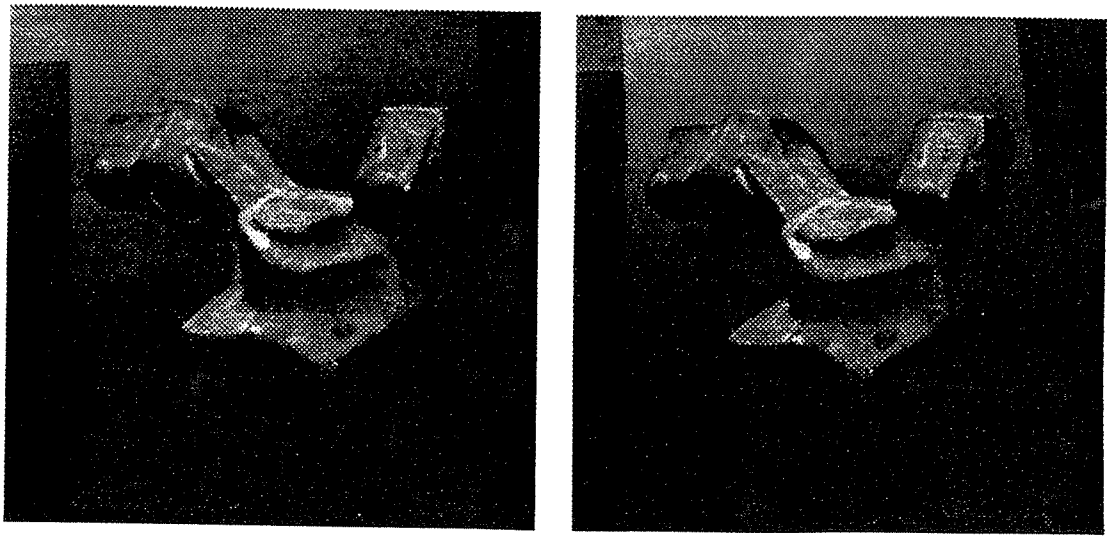
(b) Level 1 edge detection results.

Figure 5.11. Stereo pair of the Pentagon.



(c) Level 1 disparity image resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

Figure 5.11 (continued). See page 107 for discussion of the results. (Some information in the images may be lost in copying this thesis.)



(a) Input images.

(b) Level 1 edge detection results.

Figure 5.12. Stereo pair of a Renault auto part.

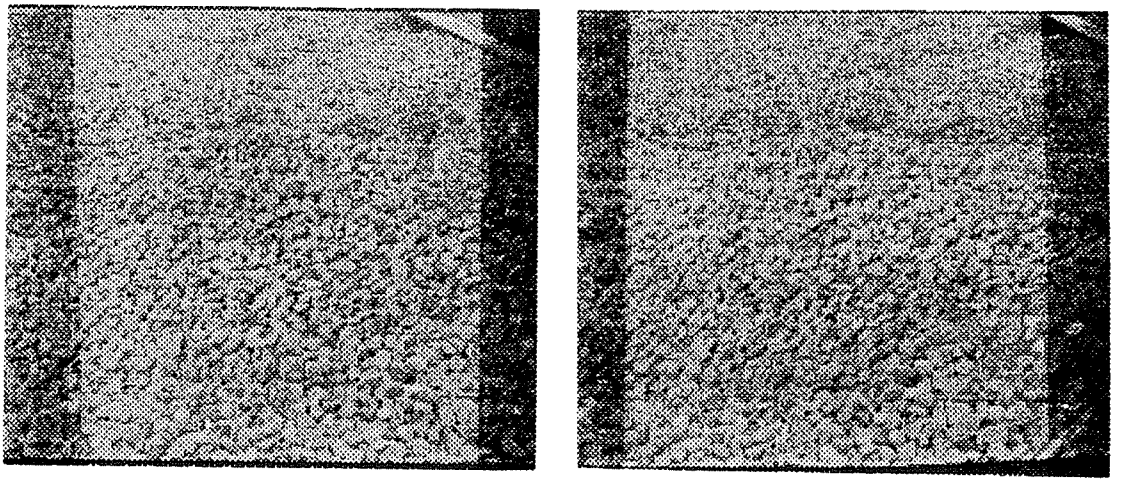
(c) Level 1 disparity image resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

Figure 5.12 (continued). See page 111 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

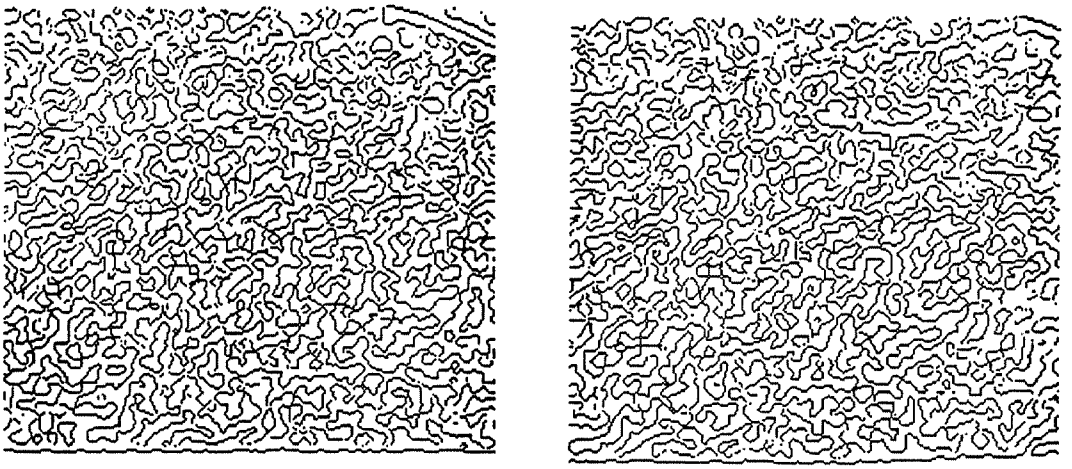
The matching was quite successful for the image of a Renault auto part in Figure 5.12. There were some minor errors outside of the part. The only important problem occurred in the middle, near the narrow area of the part. On the left side of the boundary in this area, three contours appeared in the left image, but only one in the right. For the upper edges of the single contour, the algorithm chose matches consistent with the front surface of the part. For the lower edges, it picked the

matches consistent with the boundary of the part.

The rocks image in Figure 5.13 is an example where there were significant structural differences between images. The most important feature in the images was

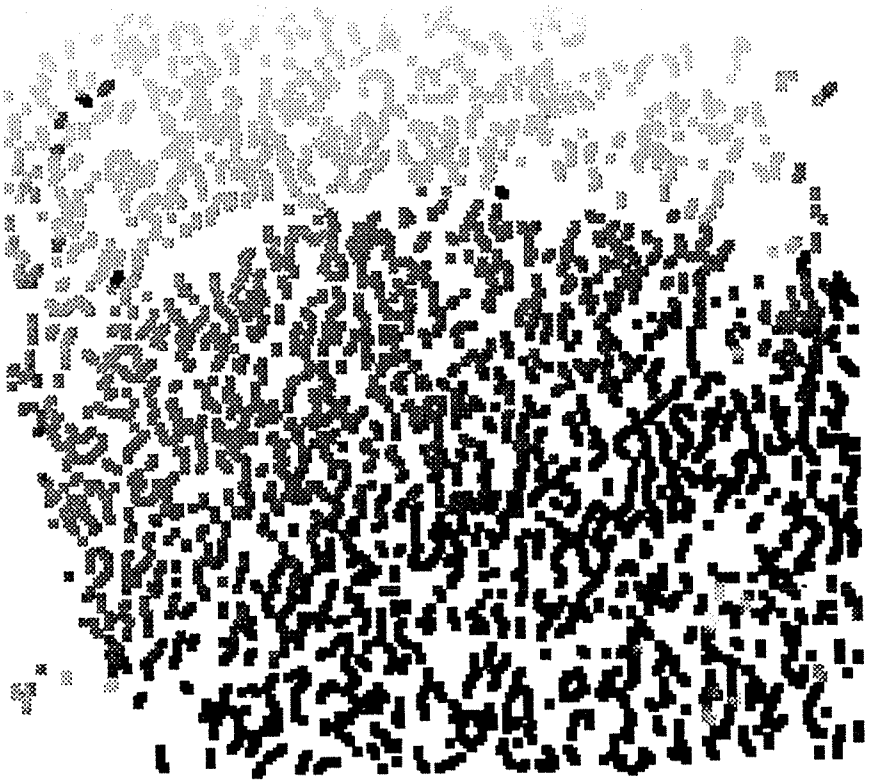


(a) Input images.



(b) Level 1 edge detection results.

Figure 5.13. Stereo pair of a pile of rocks.



(c) Level 1 disparity image resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

Figure 5.13 (continued). See page 112 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

the large horizontal occluding boundary in the upper half of the images. In general, the small percentage of matching errors was caused by three factors. First, there were minor errors near the occluding boundary. Second, as usual, there were some image boundary errors. Third, there were errors caused by appearance differences between the images. Support gathered over a wide range was often able to overcome these

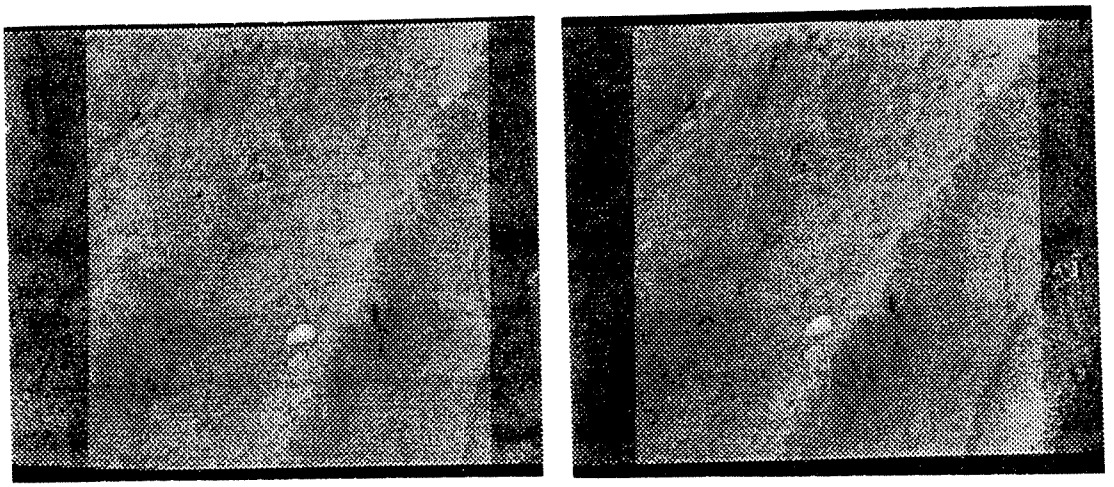
differences, but occasionally it could not. This occurred most often when the corresponding edge was completely missing. In this case the disparity gradient sometimes supported an incorrect match for the edge. Occasionally, there were enough structural differences so that the wrong match was found even though the correct one was available. Overall, however, matching was highly successful, with approximately 98% correct matching decisions.

One of the most successful sets of matching results occurred for the ruts image in Figure 5.14. The edges were relatively sparse, but there was not a lot of variation in the appearance of the two images. The differences between the images tended to be missing edges instead of more significant structural variations. The sporadic errors that were found occurred when a correct match could not gather enough support nearby to overcome the inhibiting influences of uniqueness and decay.

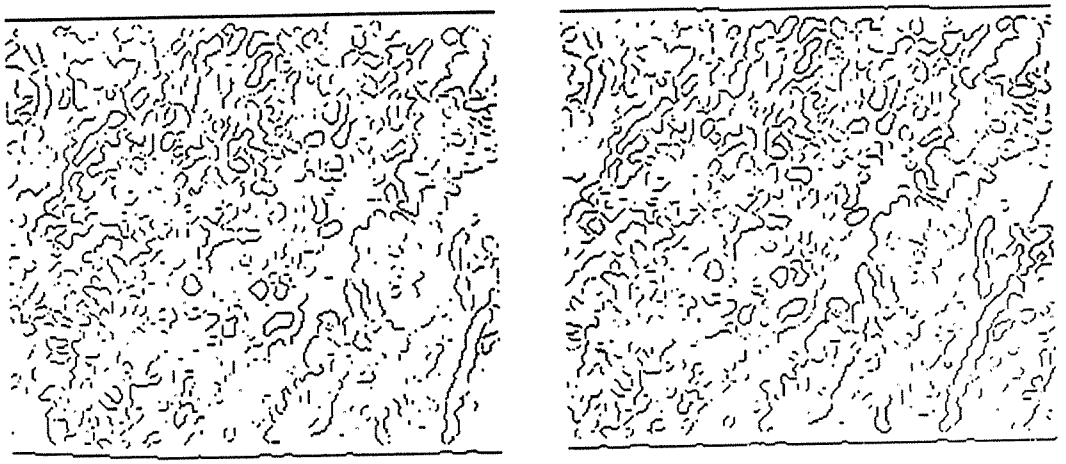
Finally, in the sandwich image in Figure 5.15 the matching was slow, but successful. The errors were sporadic and occurred when the appearance of the contours in the two images differed significantly.

5.5. Discussion

The General Support Algorithm has been experimentally shown to be successful in producing a large percent of correct matching decisions in the examples tested. Note that we emphasize the percentage of *correct matching decisions*. A correct decision occurs when either the correct match is found for an edge or when the algorithm correctly finds no match for an edge. An incorrect decision occurs when an incorrect match is found for an edge or when the algorithm misses the correct match

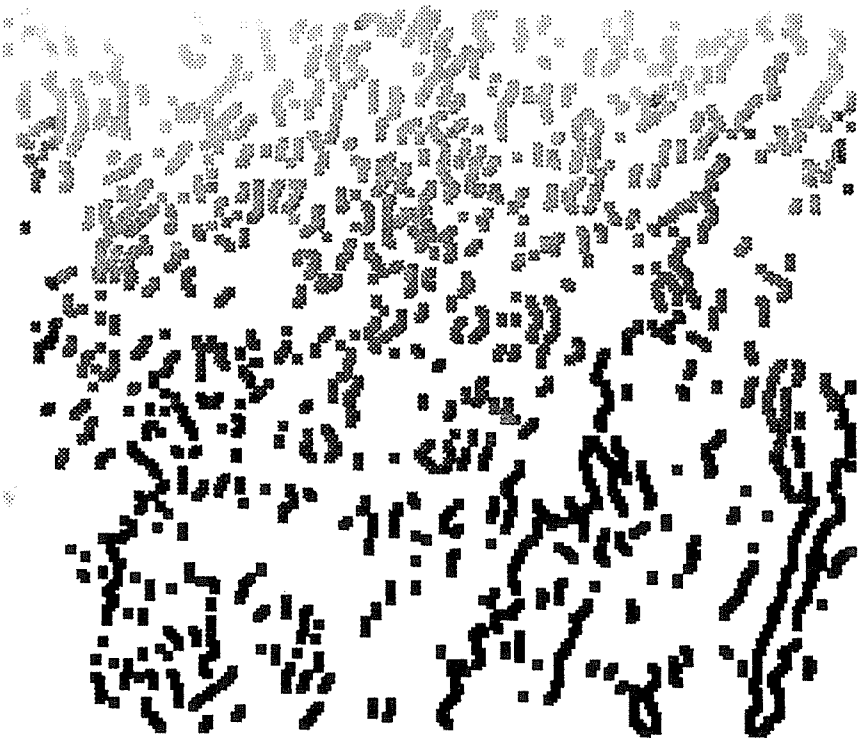


(a) Input images.



(b) Level 1 edge detection results.

Figure 5.14. Stereo pair of ruts in the ground.



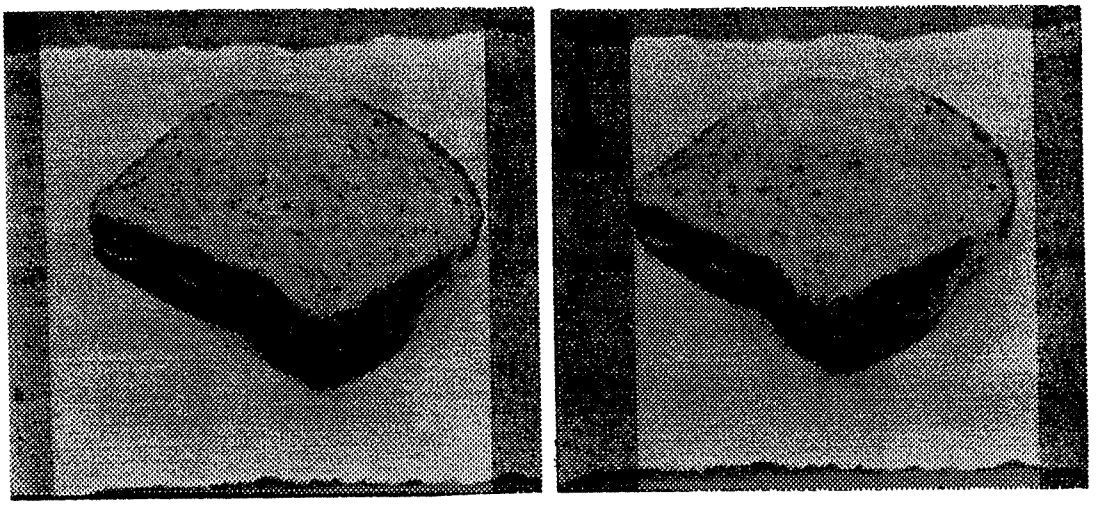
(c) Level 1 disparity image resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

Figure 5.14 (continued). See page 114 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

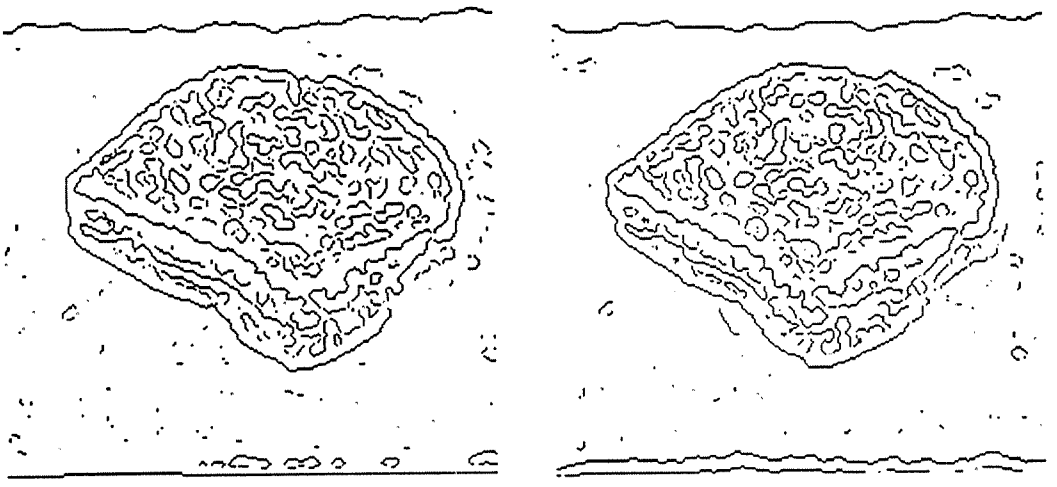
for an edge. Other algorithms have compared the number of valid matches to the number of invalid matches. Optimizing this figure is misleading since it can lead to an overly sparse set of matches.

In Chapter 4 we discussed how the GSA was designed following our guidelines for constraint integration. Here, we briefly evaluate its performance in terms of these guidelines. This evaluation is based on qualitative observations made in examining

the matching results. (Unfortunately, it was difficult to separate the influence of the constraints because the constraint definitions used in the algorithm rely on the existence of the other constraints. Generally, this makes it inappropriate to compare

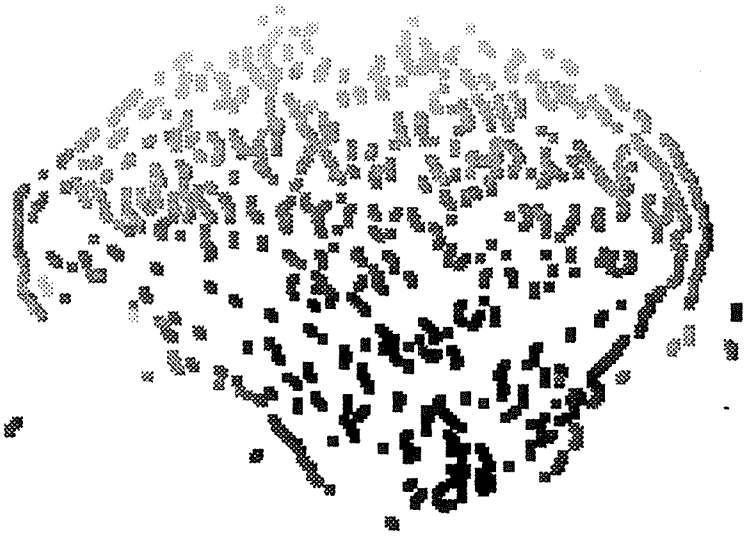


(a) Input images.



(b) Level 1 edge detection results.

Figure 5.15. Stereo pair of a sandwich.



(c) Level 1 disparity images resulting from matching. Non-white pixels are expanded into a 3 by 3 neighborhood.

Figure 5.15 (continued). See page 114 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

the performance of the algorithm with and without certain constraints. Also, in examining the results of matching, it is difficult to assign credit or blame to particular parts of the algorithm.)

- The algorithm matches a wide range of images from random-dot stereograms to real images. This is obvious from the quality of results obtained on the various

test images.

- Constraint integration improves upon the weaknesses of the individual constraints. In the results we have seen circumstances where the interaction of the constraints has improved the results. Most importantly, in dense regions the matches receiving strong multiresolution and figural continuity support tended to pull the rest of the matches into correspondence through the disparity gradient. With multiresolution or figural continuity alone, not all of the matches could be resolved, while with the disparity gradient alone there was often little to discriminate between the matches. Another example of this was where weak figural continuity support was overcome through broader based support given by the disparity gradient. This was seen in the Rocks image where the dense, fine resolution matches did not have strong figural continuity support between them.

- The GSA produces a high percentage of correct matching decisions. At present we have no way to compare our results with other algorithms. Some algorithms do not present any numerical data. Other emphasize finding only a sparse set of matches.

- The GSA performs fairly well in circumstances that have caused difficulty for previous algorithms. For example, it often is able to resolve matches for periodic structures. The most difficult of the images in this respect was the fruit image (Figure 5.10). There was a significant number of incorrect matches due to the co-occurrence of the periodic structure of the table cloth and the occluding boundary of the orange. At other places the matching of the table cloth worked

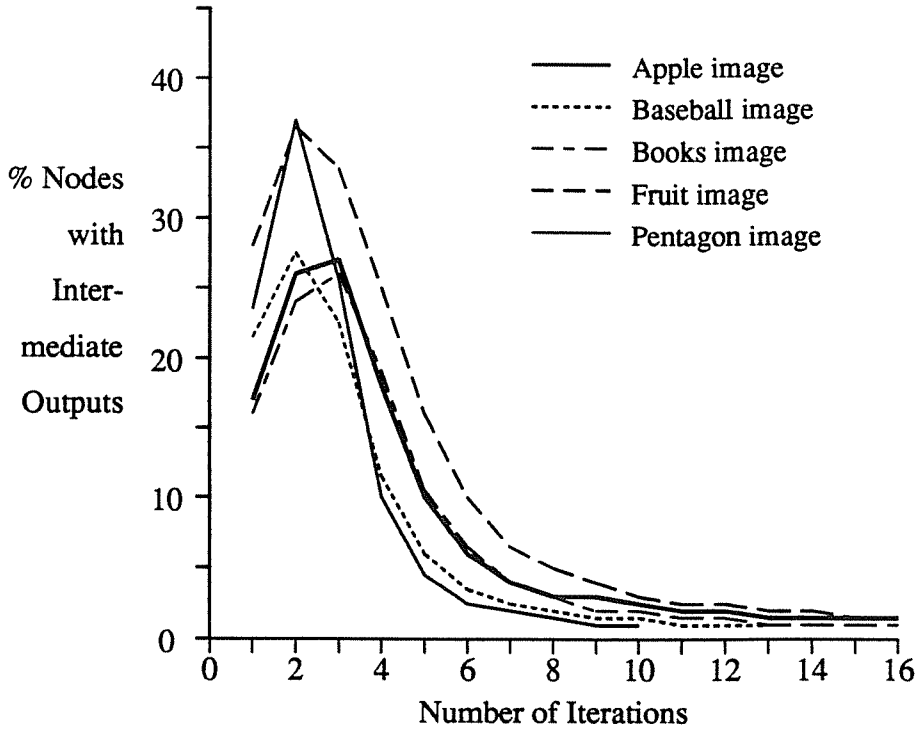


Figure 5.16. Percent of matches with intermediate output values as a function of the number of iterations. Graphed for the images in Figures 5.7 through 5.11.

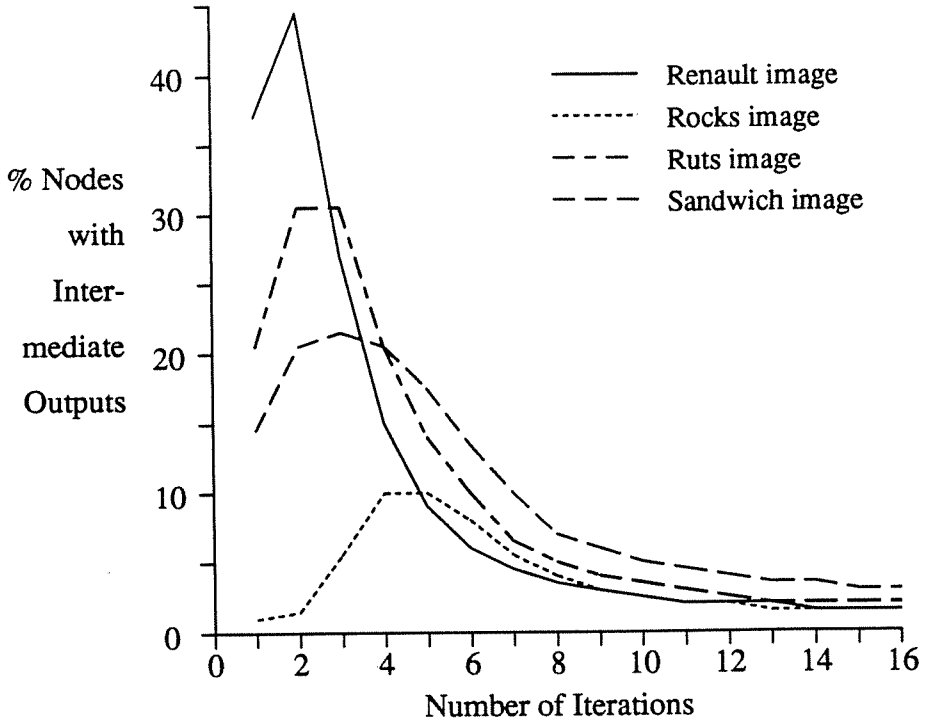


Figure 5.17. Percent of matches with intermediate output values as a function of the number of iterations. Graphed for the images in Figures 5.12 through 5.15.

well. We discuss this problem further below and in Chapters 7 and 8. Another problem with previous algorithms is that as the structural differences between the two images increases, the quality of the matching decreases. In our case, the combined use of multiresolution, the disparity gradient and figural continuity helped the algorithm to perform fairly well. Finally, the GSA handles occlusions fairly well. As was shown by random-dot stereograms and the baseball image (Figure 5.7), most edges that appeared in one image but not in the other did not have matches accepted for them.

- Finally, even with the use of a number of constraints, the matching did not require very many iterations. This was seen in the percentage of intermediate

matches in the middle range as a function of the number of iterations. In fact, with fewer constraints, the algorithm could require more iterations. For example, in experiments where figural continuity was not used, time performance of the algorithm suffered considerably. Raising the disparity gradient weight enough to compensate for this time performance increased the number of matching errors, especially near occluding boundaries.

5.6. Concluding Remarks on Local Constraint Integration

The General Support Algorithm uses integration of *locally defined constraints* to produce high quality stereo matching. These constraints include surface structure constraints such as figural continuity and the disparity gradient, hierarchical image constraints such as fine-to-coarse and coarse-to-fine multiresolution, and edge property constraints such as detailed match. However, in spite of careful integration of these constraints, there are matching situations that they cannot explicitly address. These situations occur in three general types of circumstances which are problematic for any stereo algorithm based solely on locally defined constraints:

- The first involves significant structural differences between the two images. This arises from noise and from the differing viewpoints of the images. Simple cases of these differences can generally be handled implicitly through the interaction of a number of the constraints. However, the problem is often more severe, as seen in a number of the real images we tested.
- The second circumstance arises in resolving matches near occlusions. Occluded edges often match edges in the other image that have no valid matches. These

occlusion matches often receive support from matches across occlusion boundaries. This support is in violation of the assumptions of the surface structure constraints, but not their support definitions. In addition, erroneous matches may receive strong support across discontinuities in disparities that do not involve occluded edges. Again, this support is in violation of the assumptions of the constraints.

- The third problem involves periodic patterns that are partially-occluded. There is often little to distinguish between matches for these edges. The local constraints can give nearly equal support to a number of competing matches for edges arising from periodic structures. Normally, in these situations the results of matching at the boundaries of the regions are propagated to the interior matches to resolve the ambiguities. Our formulation of uniqueness facilitates this. However, when the periodic region is partially-occluded, incorrect matches may align near the occluding boundary, causing erroneous matches to be accepted. A dramatic example of this appeared in the fruit image in Figure 5.10.

In addition to these three general problems with local constraint interaction, our General Support Algorithm is sensitive to the density of edges in regions of the images. This only effects the disparity gradient weight, however. To accommodate this problem in the tests presented here we changed the parameter *BASE_DG* according to the density of the edges.

Solutions to the general problems with locally defined constraints and the specific additional problem in the General Support Algorithm of needing to vary

BASE_DG are discussed in detail in Chapter 7. Another approach to solving to the general problems using trinocular stereo is discussed in Chapter 8. First, however, we present some experiments using a parallel simulation of the General Support Algorithm on a shared-memory multiprocessor.

5.7. Chapter Summary

In this chapter we have examined the results of testing the General Support Algorithm. The connectionist implementation of the algorithm was simulated and run on a wide range of images, including hand-drawn synthetic images, random-dot stereograms, and real image pairs. Overall the algorithm made approximately 97% correct matching decisions. In addition, in examining the results, we observed several situations that have caused problems for previous algorithms, but that the combination of multiple constraints in the General Support Algorithm handled correctly. The few matching errors corresponded to circumstances that can not be explicitly recognized using locally-defined constraints. These include partially-occluded periodic regions, occlusions in general, and significant structural differences between the images.

CHAPTER 6

Parallel Simulation of the General Support Algorithm

A major problem in studying realistic connectionist networks is the time required to simulate them. This is a particular problem with the General Support Algorithm because it defines a large number of nodes and connections. On a serial computer such a simulation is cumbersome since an update to the activations of one node requires gathering outputs from neighboring nodes, scaling each one by the appropriate connection weight, and summing them. This process is repeated for each node in the network at each iteration. Because of the simple, regular computational requirements involved, the simulation is an obvious candidate for parallel implementation.

In this chapter we describe the parallel simulation of the connectionist network implementing the General Support Algorithm on a Sequent Balance 21000. The Sequent Balance® is a true shared-memory multiprocessor. It contains a number of identical 32-bit microprocessors, a high speed data bus, and a single common memory. Contention for the bus is reduced by 16 kilobyte caches that are local to each processor. The caches are write-through, i.e. when one processor writes to a shared variable stored in a cache, that write is echoed to main memory. This

---

Balance and DYNIX are registered trademarks of of Sequent Computer Systems, Inc. UNIX is a registered trademark of AT&T.

invalidates all other copies of the variable that might be stored in other caches. Actually, cache locations are allocated in small blocks so that writes to any part of a block invalidates all copies of the block in other caches.

The software environment of the Balance system is controlled by the DYNIX® operating system, a distributed version of UNIX®. There are two forms of parallelism available on the system, function partitioning and data partitioning. Function partitioning involves the simultaneous activation of a set of distinct procedures; data partitioning involves multiple activations of the same procedure. The most common instance of data partitioning occurs when a group of identical procedures divides up the iterations of a large loop. In doing so there are two methods for controlling the division of the work. First, in static scheduling the iterations of the loop are partitioned *a priori*. Second, in dynamic scheduling each process works on an iteration (or group of iterations), and upon completion, allocates more iterations from a central list. Data partitioning requires more overhead, but it may be preferable for a two main reasons: it accommodates variations in the processing times of the iterations, and it automatically adjusts the workload when there is contention for the processors from other programs. As a final feature of the software environment, synchronization and mutual exclusion in the DYNIX system are realized using system primitives built on top of semaphores.

In the remainder of this chapter we describe the parallel simulation of the General Support Algorithm (GSA) in detail. Section 6.1 describes a simulation based on the GSA network. Section 6.2 analyzes the parallel implementation of the

simulation. Finally, Section 6.3 discusses the results of testing the parallel simulation on a Sequent Balance system with 10 processors.

6.1. Simulation of the General Support Algorithm

The connectionist network for the GSA above is extremely large and therefore quite expensive to simulate directly. As discussed in Chapter 4, using a careful encoding scheme it requires $L\ p\ N^2$ nodes, where L is the number of resolution levels, p is the number of discrete disparity values, and the images are of size N by N. In addition, each node has a large number of connections. Fortunately, for a given pair of images, the set of nodes that actually represent candidate matches is extremely sparse. We use this observation to structure the simulation as follows. For each pair of images we construct only the active part of the network. This contains only those nodes representing candidate matches and only those connections between pairs of candidate matches. The result is subnetwork that is *functionally isomorphic* to the entire network for the given pair of images.

The simulation involves two phases of processing: the first phase builds the three main data structures used by the simulation, and the second phase executes the iterations of the network. The data structures consist of the edge images (note that we do not consider edge detection as part of the simulation), the list of candidate matches, and a list of connections between these matches. During the iteration phase of the simulation, the network alternately updates the nodes' activations and their outputs. This process repeats either for a fixed number of iterations, or until a small percentage of the nodes have intermediate output values (e.g. $0.25 \leq O_i \leq 0.75$).

The three main data structures are shown in Figure 6.1. The edge images are represented as arrays. Those positions having no edges are represented as null entries. Each entry contains a pointer to a list of candidate matches for that edge (the "Match ptrs" in Figure 6.1). The match nodes list actually consists of three separate lists: the node activations ("A" in Figure 6.1), the node outputs ("O" in Figure 6.1), and the description of the nodes. This description includes the image coordinates for the edges and a pointer the list of input connections to the match. Candidate matches are

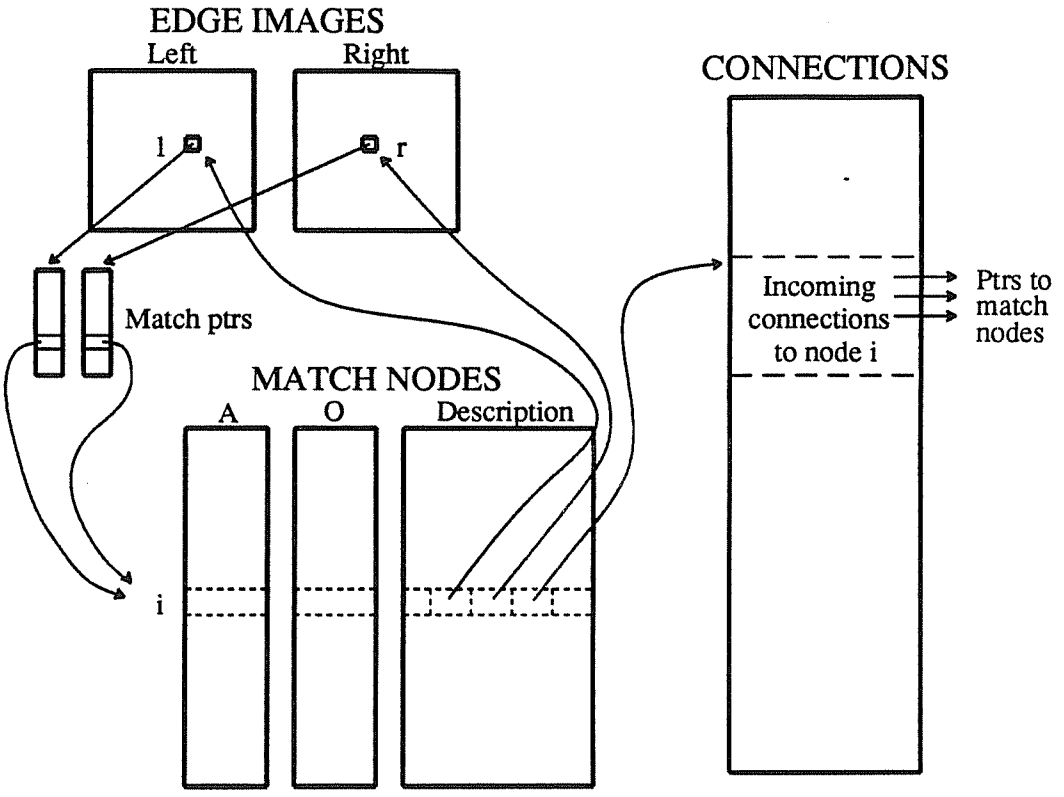


Figure 6.1. Diagram of the three main data structures used in the simulation. See the text for discussion.

determined by searching the edge images. When a candidate node is found a location in the match node list is allocated, and the initial activation and output of the match are computed. In addition, the match is added to the linked list of matches for both the left and right edges. This list facilitates the uniqueness computation.

The third data structure, the connections list, requires the most space. Each entry contains only the weight of the connection, and the index of the output match node for the input connection. Empirically, we have found that for random-dot stereograms there are approximately 150 active connections per match. This varies of course with the density of the edges in the images. Finding the active incoming connections to a given match requires examining both the edge images and match list for other matches that meet the requirements of a given constraint. Fortunately, many of the time consuming computations, such as the relative distance functions and weight values in the disparity gradient, may be precomputed and stored in a look-up table. When the list of input connections to a match is completed, space for them is allocated in a contiguous block in the connections list.

After the three data structures are built, the simulation of the network iterations begins. In each iteration the new activations of all the nodes are computed. Once this is complete these activations are used to compute the new outputs of the nodes. During the activation computation for a given match, the previous activation for the match, the list of its connections in the connections array, and the list of competing matches for both the left and right edges are needed. Thus, via the list of connections, various locations in the output list are referenced. During the output computation of

an iteration, the activation of a node is read and then its new output value is stored. Other details of the match description are not needed during the output computation.

6.2. Parallel Implementation

The simulation described in the previous section is designed so that it can easily be implemented in parallel. In this section we describe the details of this implementation including synchronization and mutual exclusion among the parallel loop iterations, static vs. dynamic allocation of the iterations, the number of consecutive loop indices allocated by each processor at once, and some additional hardware contention issues. In making the choice between static and dynamic allocation for a given loop of the simulation, and in choosing the number of iterations allocated at one time, we used simulation results for a test random-dot stereogram. Once the best choices were identified we measured the speed-up of the parallel implementation as compared to the sequential version of the simulation. In the remainder of this section we consider each of the above parallel issues beginning with the parallel structuring of the individual parts of the simulation.

During the input of the edge images, a separate process is created for each resolution level for each of the left and right images. The edge data for each of these images resides in a different file. Thus, there are $2L$ different processes created, where L is the number of resolution levels. The duration of these processes will vary according to the number of edges in a given input file, but since input is sequential, no further division of work is possible. When there are fewer than $2L$ processors available, each processor must handle multiple input images sequentially.

The construction of the match list is not as straightforward as reading the image input. As noted above, matches are allocated from a shared array. In a parallel implementation, multiple processes need to allocate matches simultaneously. In addition, the linked list of matches for a given edge must be updated for each match node. Both of these issues can potentially cause significant problems because each requires mutual exclusion. For the most part, however, these problems can be avoided by making use of the epipolar scanline assumption. That is, all matches for the edges in a row must be in the same row in the other image. This leads to the idea of having one processor handle all of the matches within a given row. A processor identifies all the candidate matches and stores them in its local data area. When all of the matches have been found within a row, the processor allocates a contiguous area in the shared match list and then copies the local data into the shared list. Note that while copying, no mutual exclusion is needed. In this way, there is reduced contention in allocating space in the match list since large blocks are allocated at once. The problem of mutual exclusion in updating the linked list of matches for each edge is completely eliminated since all of the matches for an edge are found and stored sequentially by a single process.

The parallel construction of the connection list works in much the same way as the match list. One parallel process finds all of the connections for a single match node. The process determines these connections and stores them in its local data area. When all of the connections for the node are found, space is allocated in the global connections list. This has the additional advantage that the connections for a node are

always stored consecutively.

Once the data structures are initialized in shared-memory, the simulation of network iterations begins. During the iterations there is no need for mutual exclusion among the processors. In updating the activation for a node, a processor must reference the outputs of a variety of nodes. However, during the activation computation, these values are *read-only*, so no mutual exclusion is needed in referencing them. During the output computation for a node, only the activation and output values of the given node are referenced, so there is no need for mutual exclusion here either.

Synchronization does not cause a major problem during any part of the simulation. It must occur after each step in building the data structures, between each iteration, and between the activation and output computations within each iteration. The synchronization between activation and output computations is realized using a DYNIX program primitive called a "barrier" that all processes must reach before any of them proceeds to the outputs computation. In all other cases synchronization is handled implicitly. This occurs because processes initiated in parallel from a DYNIX procedure must all end before the procedure may continue.

Next, consider the issues of dynamic vs. static allocation of loop iterations, and the number of iterations allocated at one time. Generally, when different options were tested, there were only minor differences in processing time. (The numbers below represent the average performance using nine processors when there were no other users active on the system.) Clearly, there is no need for dynamic allocation of the

output update computation during the network iterations. In each of the other parts of the computation the time required for each iteration varies, but this does not greatly affect the performance of the simulation. For the matches computation, dynamic allocation in small blocks (2 rows at a time) was slightly faster than static allocation (1.58 seconds vs 1.53 seconds). For the construction of the connections list, dynamic allocation of single iterations was slightly faster than static allocation (36.83 seconds vs. 37.35 seconds). The performance using dynamic allocation was improved slightly by increasing the number of loop iterations allocated at once by each processor to 8 (36.40 seconds). Finally, the activations computation was not improved much through dynamic allocation of larger blocks, i.e. the times were 25.02 seconds for static allocation of a single iteration vs. 24.85 seconds for blocks of 4 iterations. Thus, according to these experiments, although dynamic allocation improved the performance of the algorithm slightly, the differences were not very significant.

As a final consideration in the parallel simulation, the major potential hardware problem involves contention for the main memory bus. (When there are multiple users, main memory allocation and paging may be issues, but we will not consider them here.) The caches local to each processor are designed to alleviate this somewhat. However, the simulation of the General Support Algorithm is significantly larger than the combined size of the caches (the example used here required approximately 16M bytes). This implies that there is a potential for bus contention when a large number of processors is in use. Fortunately, as will be shown in the next section, this problem did not occur for the tests we ran. This is demonstrated by the

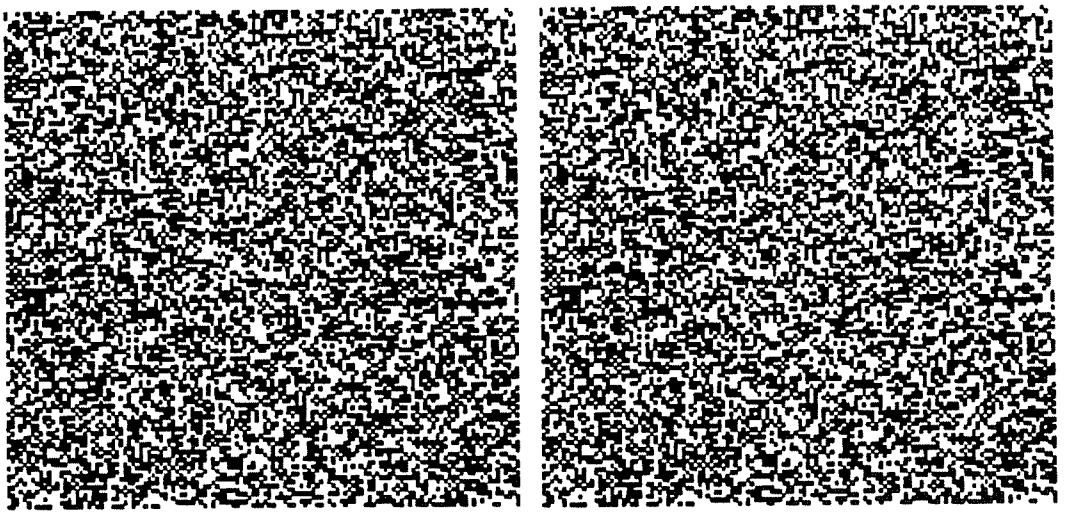
near optimal speed-ups obtained.

6.3. Parallel Results

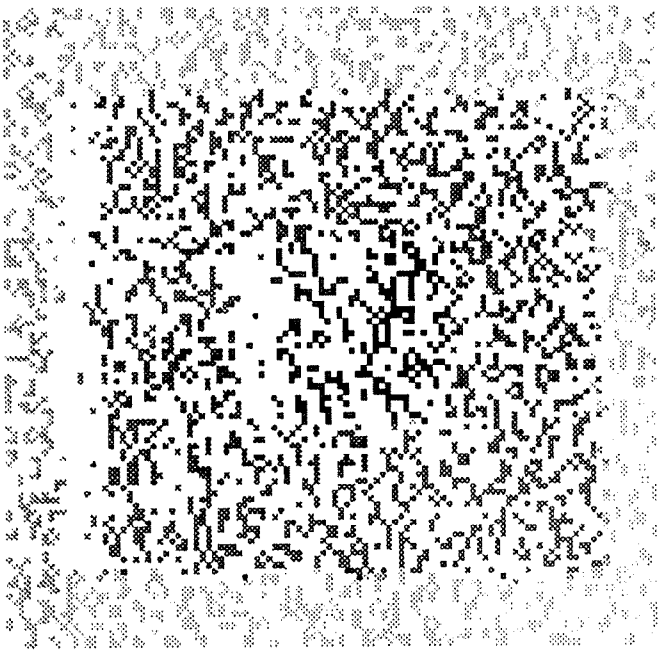
This section presents the results of the parallel simulation on the random-dot stereogram shown in Figure 6.2. The stereogram was 128 by 128 and represented three different surfaces at disparities 0, 6 and 12. The algorithm constructed 11,090 active nodes and 1,637,570 active connections for this stereogram. The algorithm produced 97% correct matching decisions in this example.

The Sequent Balance configuration uses 10 Intell 80386 processors linked via a high-speed bus and sharing common memory. All but one of the processors may be allocated for parallel execution.

Our results demonstrate the speed-ups obtained using N processors, $N \leq 9$, compared to the time required for a sequential implementation. For each value of N, the simulation was run three times with no other active users on the system, and the results were averaged. The times for the simulation are separated into times for input, match list building, connections list building, and the time to run 10 iterations of the network. These are shown in Table 6.1. The results are graphed for building the match list, building the connections list, and simulating the network iterations in Figures 6.3, 6.4 and 6.5. Figure 6.6 presents the overall speed-up obtained. In each figure, the dark curve shows the actual speed-ups obtained and the broken line shows the optimal speed-ups. Speed-up is measured in terms of the effective number of processors. That is, for N processors, the effective number of processors is,



(a) Input stereogram



(b) Disparity image results for the finest resolution (disparity is encoded as intensity).

Figure 6.2. Random-dot stereogram with three layers. The darker points in the disparity images (b) represent points that are closer to the viewer. See page 134 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

Simulation Times for N Processors					
Num	Phases of the Simulation				Total
Processors	Input	Match	Connections	Iterations	Time
1	17.43	4.54	297.3	213.9	533.1
2	11.87	2.67	150.8	105.6	270.9
3	9.23	1.98	101.6	71.2	184.0
4	7.67	1.68	76.8	54.2	140.5
5	6.56	1.54	62.1	44.6	114.8
6	5.13	1.48	52.3	36.3	95.2
7	5.14	1.47	45.4	31.5	83.5
8	5.21	1.49	40.6	27.9	75.2
9	5.15	1.53	36.4	25.0	67.0

Table 6.1. Timing results for the parallel simulation.

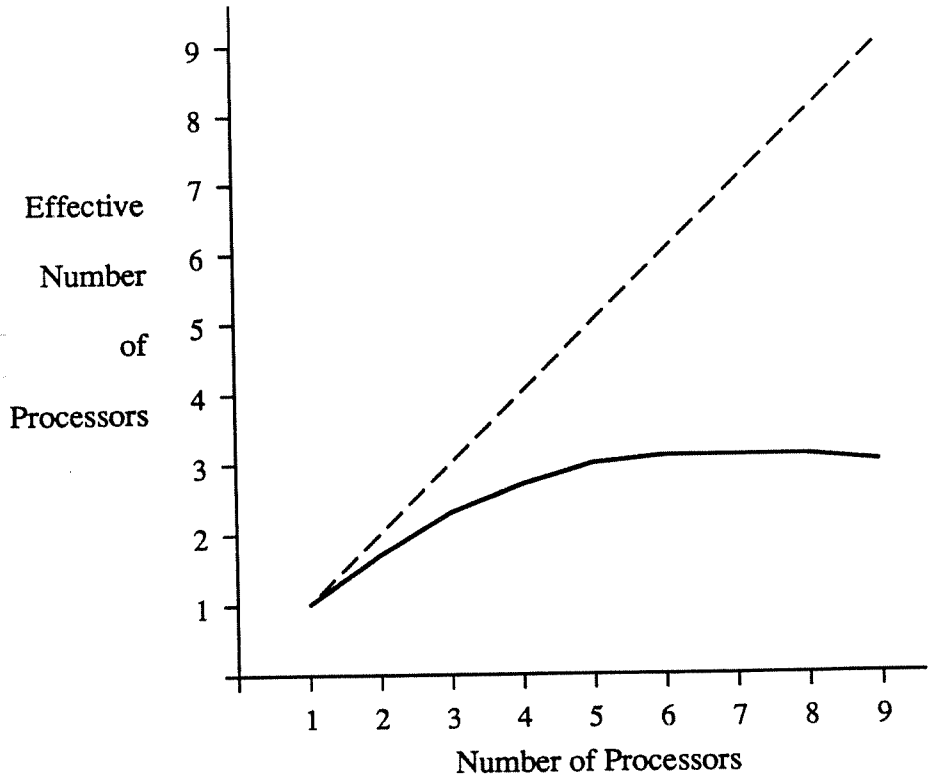


Figure 6.3. The speed-up obtained for building the list of candidate matches. Note that the best processing time is obtained for 7 processors. See the text for the discussion of this.

$T(1)/T(N)$, where $T(i)$ is the execution time in running i processors.

As shown in Figure 6.3, the match list building does not approach the optimum speed-up, and does not improve beyond the use of 7 processors. Since the loops of match list building are relatively independent, the only likely explanation is that the speed-up is limited by the non-trivial time required to initiate processes. With images requiring more active match nodes the allocation of match nodes will require more time and so speed-ups may continue to be obtained beyond 7 processors. Due to main memory size problems we have been unable to test larger images.

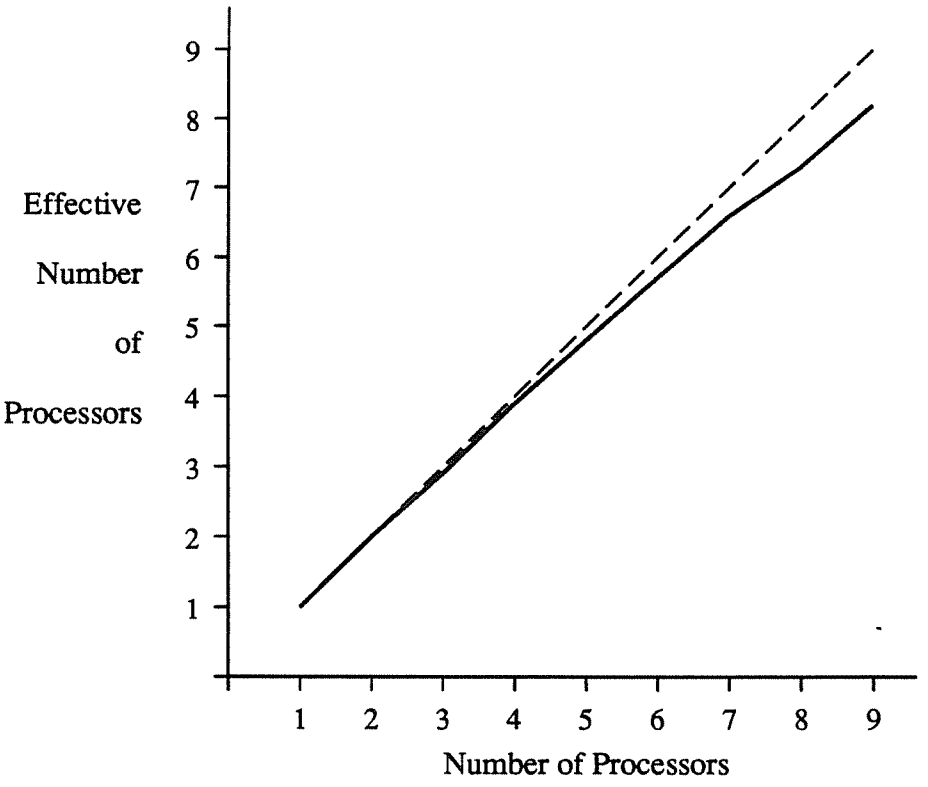


Figure 6.4. The speed-up obtained for building the connections list.

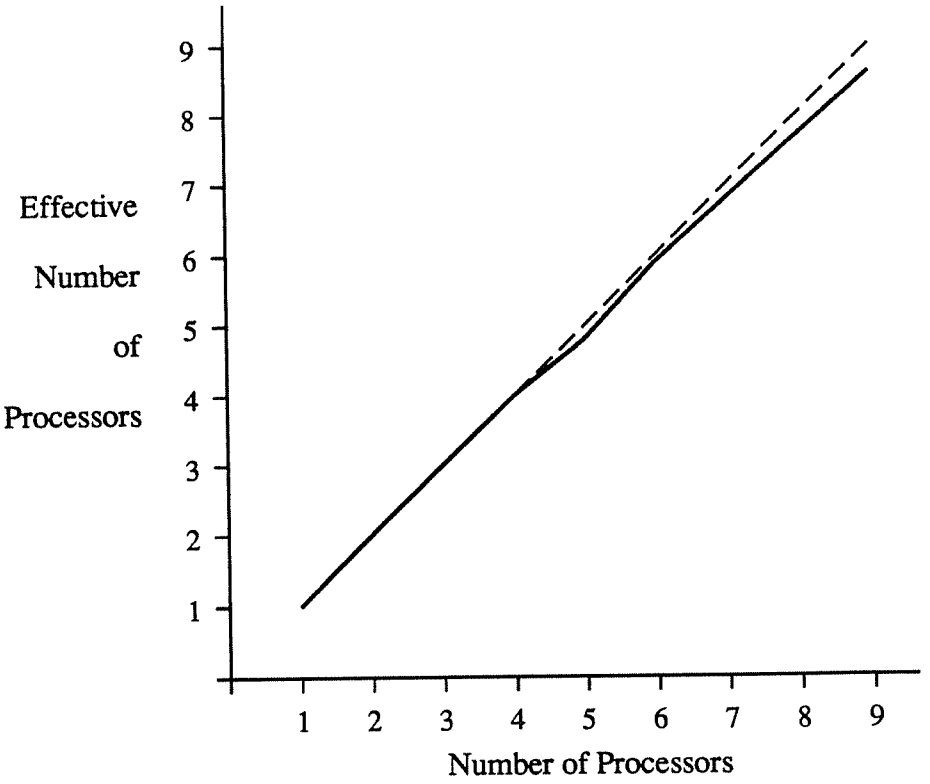


Figure 6.5. The speed-up obtained for the iterations of the network.

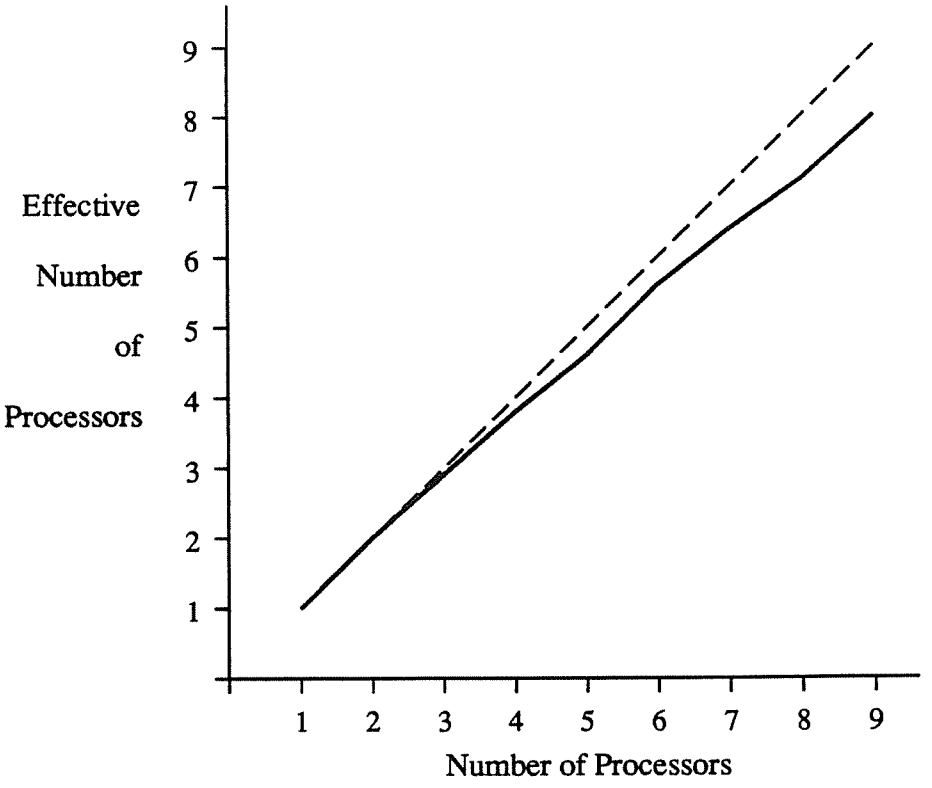


Figure 6.6. The speed-up obtained for the entire simulation. Note that this is near optimal, even for nine processors.

The speed-ups for building the connection list and for the iterations of the network were both almost optimal. This is encouraging since these steps required by far the greatest percentage of the total time. Connection list building required 56% of the sequential processing time, and 54% when 9 processors were used. Ten iterations of the network required 40% and 37% of the total time respectively.

Because the connection list building and the iterations dominated the processing time, the parallel speed-up for 9 processors was nearly optimal. Increasing the number of processors is likely to continue improving the performance. Eventually,

the decrease in time due to increases in the number of processes may flatten for large N because of either: (1) contention for the main memory bus, or (2) mutual exclusion in allocating space in the shared memory. Based on our data it is not possible to predict when this will occur.

6.4. Chapter Summary

The results presented in this chapter have shown the effectiveness of the parallel implementation of the simulation of the General Support Algorithm on the Sequent Balance 2100 shared-memory multiprocessor. This was realized by careful development of the simulation so that there was little contention among the parallel processes in terms of mutual exclusion and synchronization. This performance shows that the study of connectionist network vision algorithms can be greatly facilitated through parallel simulation on shared-memory multiprocessors.

CHAPTER 7

Extensions to the General Support Algorithm

Chapter 5 demonstrated the effectiveness of the General Support Algorithm in producing high quality stereo matching results. However, it also identified a number of problems with the algorithm. These included the sensitivity of the disparity gradient's influence to the density of edges, the difficulty in resolving matches in periodic regions that are partially occluded, the general problem of occlusions, and errors that are due to significant structural differences between the images. The first problem is specific to the General Support Algorithm. The latter three problems are general to stereo algorithms employing locally defined constraints.

In this chapter we investigate solutions to these problems. The solutions involve extensions to the GSA, including the use of intrinsic image information, non-local matching mechanisms, and interactions between edge detection, contour extraction and matching. Some of the solutions have been implemented and tested. Others include experimental demonstrations of their potential effectiveness. The remainder are limited to theoretical investigation. An entirely different approach to solving these problems that employs three cameras for stereo matching is described in Chapter 8.

7.1. Compensating for Edge Density

In the matching of real image pairs presented in Chapter 5, it was necessary to tune the base disparity gradient parameter to improve the algorithm's performance.

Essentially, we varied the parameter inversely with the density of the edges in a given image. The density of the edges influences the number of active disparity gradient connections in a given region, and may significantly alter the overall influence of the constraint. The other constraints are not as sensitive to the density, since, for a given match node there are at most two active multiresolution connections and four non-zero figural continuity connections. Given a fixed, intermediate value for the disparity gradient parameter, variations in the density of edges effect the performance of the algorithm in a number ways. Theoretically, they alter the overall influence of the disparity gradient relative to the other constraints. Practically, a denser set of edges tends to increase the number of multiple matches and the number of erroneous matches for occluded edges. When edges are sparse, the disparity gradient can not help to overcome weaknesses in the other constraints and matching requires more iterations.

7.1.1. Solution to the Edge Density Problem

The solution to this problem involves detecting the density of edges in a given region of the images and using this to locally modify the disparity gradient influence. From the definitions given in Chapter 4, the disparity gradient input to a given node i is:

$$\sum_{j=1}^{M} \frac{BASE\ DG}{D_{i,j}} \frac{c}{c+d_{i,j}} O_j$$

where $D_{i,j}$ is the distance between the two nodes, $d_{i,j}$ is the disparity difference

between the nodes, M is the number of disparity gradient connections, and O_j is the output from node j in the previous time unit. Factoring out the disparity gradient parameter, we have:

$$BASE\_DG \sum_{j=1}^{M} \frac{1}{D_{i,j}} \frac{c}{c+d_{i,j}} O_j$$

In this equation, the sum is independent of the parameter $BASE\_DG$. Therefore, to adjust the influence of the disparity gradient we make $BASE\_DG$ a function of the density of edges γ a given region:

$$BASE\_DG = W(\gamma)$$

Implementing these ideas in a connectionist network requires mechanisms to: (1) determine the density of edges in regions of the images, (2) collect the disparity gradient input for each match node separately from the other constraint inputs, and (3) modify the disparity gradient input to the node according to $W(\gamma)$. First, we compute the edge density γ for each image point (not each match node). For a given point, γ is determined over a region with diameter R by a node m. There is one incoming connection to m from each edge detection node in the region. The weight of each such connection is simply $1 / A(R)$, where $A(R)$ is the area over which the density is computed. Node m sums the weights to determine γ and adds a constant to the sum (defined below) to determine its activation and output. Finally, note that this adds N^2 additional nodes to the network, where the image size is N *by* N. This computation does not distinguish between resolution levels.

The second new mechanism collects the disparity gradient input separately at each match node i. This requires an additional node, i_0 as shown in Figure 7.1. The weight of the incoming connection from node j to i_0 is:

$$\frac{1}{D_{i,j}} \frac{c}{c+d_{i,j}}$$

Node i_0 sums the disparity gradient inputs and outputs them to node i.

The final new mechanism uses the density, γ, to modify the output from node i_0 to node i. Specifically, the disparity gradient input that is collected by i_0 is scaled by $W(\gamma)$ using a conjunctive connection. Assume that $W(\gamma)$ is linear, i.e.

$$W(\gamma) = a\,\gamma + b$$

The weight of the connection from m is a (see Figure 7.1), and node m adds b / a to its summed density input. Thus, the value output along the connection is $W(\gamma)$. At the conjunction, the two weighted output values are multiplied to determine the scaled

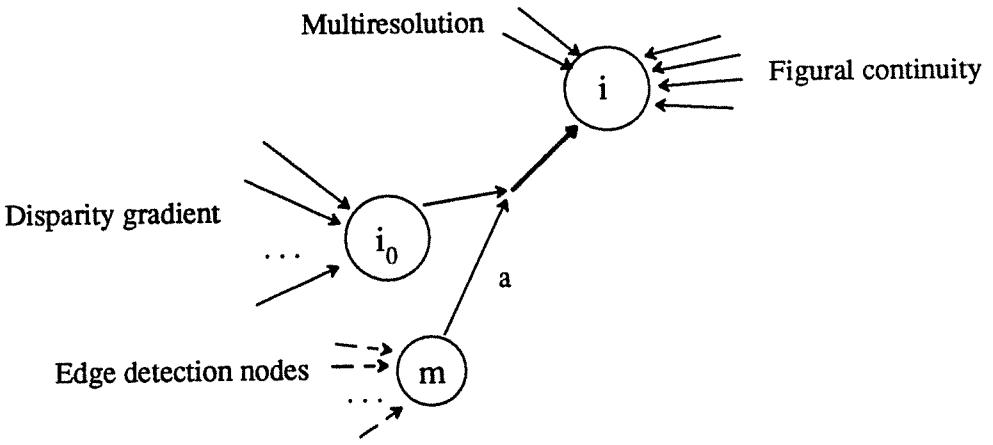


Figure 7.1. The disparity gradient input to a node modified by the weight-density function.

disparity gradient input. The actual values of a and b are chosen empirically as described in the next subsection.

As noted above, this weight modification algorithm does not distinguish between different resolution levels. The density computed is the average over all levels, and $W(\gamma)$ is the same for each resolution. The main reason for the simplification is that the General Support Algorithm works well with no variations in the *BASE_DG* value between resolutions. If we attempted to use different weight density values at different resolutions we would need to take into account the different overall influences on the matches at each level. In addition, the data available from Chapter 5 that we will use to determine $W(\gamma)$ are based on a constant value of *BASE_DG* over all resolutions.

7.1.2. Determining the Weight Density Function

$W(\gamma)$ is determined by (1) computing the average edge density in the matching regions of a number of image pairs, (2) plotting the *BASE_DG* value as a function of these results, and (3) fitting a line to the data. We use six image pairs from Chapter 5 to determine this line. These are the four-layer stereogram in Figure 5.4 ($\gamma = 0.265$), the baseball image in Figure 5.7 ($\gamma = 0.222$), the Apple processor board image in Figure 5.8 ($\gamma = 0.169$), the fruit image in Figure 5.10 ($\gamma = 0.116$), the Renault auto part image in Figure 5.12 ($\gamma = 0.137$), and the rocks image in Figure 5.13 ($\gamma = 0.137$). Using a least-squares fit, the resulting line is

$$W(\gamma) = -0.327\,\gamma + 0.124$$

The x-intercept of this line is $\gamma = 0.38$, which is an extremely large edge density. The

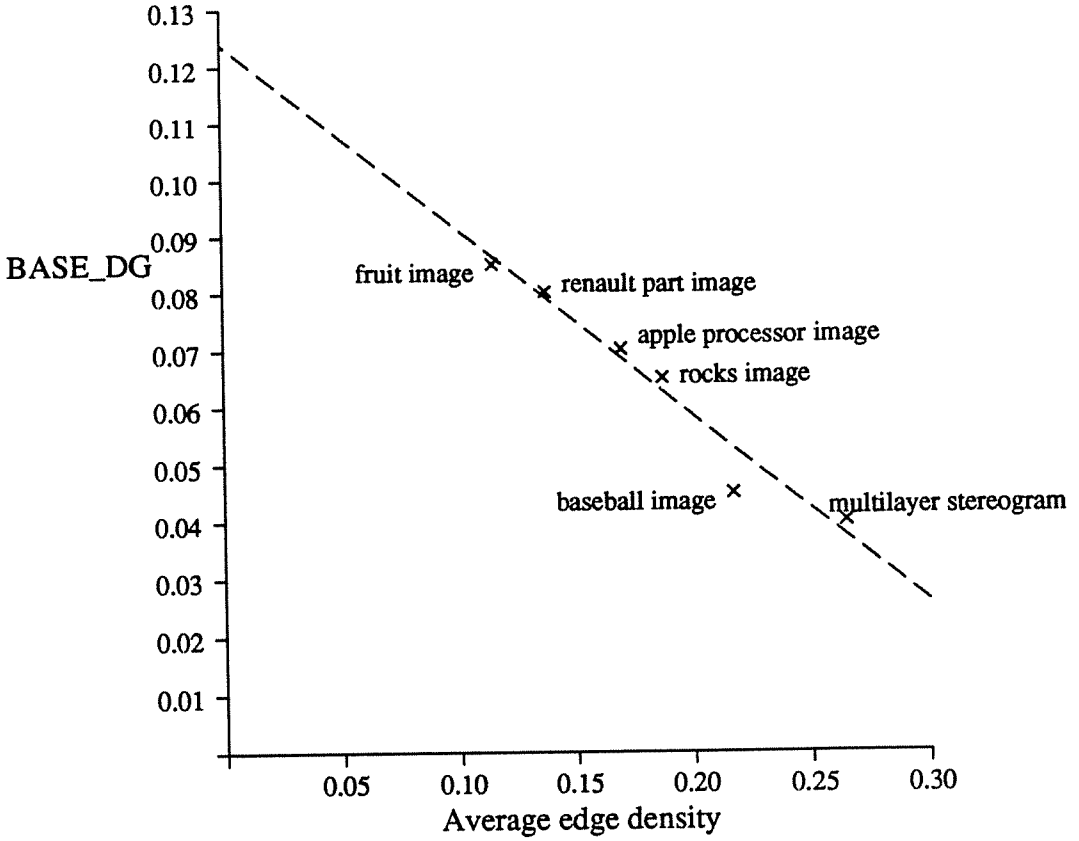


Figure 7.2. *BASE_DG* as a function of the average density. The line fit to these points determines $\bar{W}(\gamma)$.

points used to fit the line and the resulting line $W(\gamma)$ are shown in Figure 7.2. The plot of the points is fairly linear which further justifies our assumption that $W(\gamma)$ is a linear function.

7.1.3. Test Results

To test the usefulness of the disparity gradient weight modification algorithm using $W(\gamma)$ determined above, the algorithm was run on three image pairs. These were the fruit image (Figure 5.10), the pentagon image (Figure 5.11) and the sandwich image (Figure 5.15). One of the important features of the new algorithm is that the

weight varies locally with changes in γ. Previously, *BASE_DG* was set for an entire image. Because of this difference, the fruit image is tested to ensure that local variations in density did not adversely affect the algorithm. The other two examples tested the new algorithm's performance on image pairs that were not used in determining $W(\gamma)$.

Figure 7.3. Left level 1 disparity results for the fruit image run using the disparity gradient weight modification algorithm. See page 149 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

The results of testing were uniformly positive. In each case there was no significant variation in the percentage of correct matching decisions as compared to the original results. The new left disparity images are displayed in Figures 7.3 through 7.5. There was little variation in appearance between the old and the new results for any of them. For the fruit image (Figure 7.3), there were some additional erroneous matches near the boundary of the image and the edge of the grapefruit.

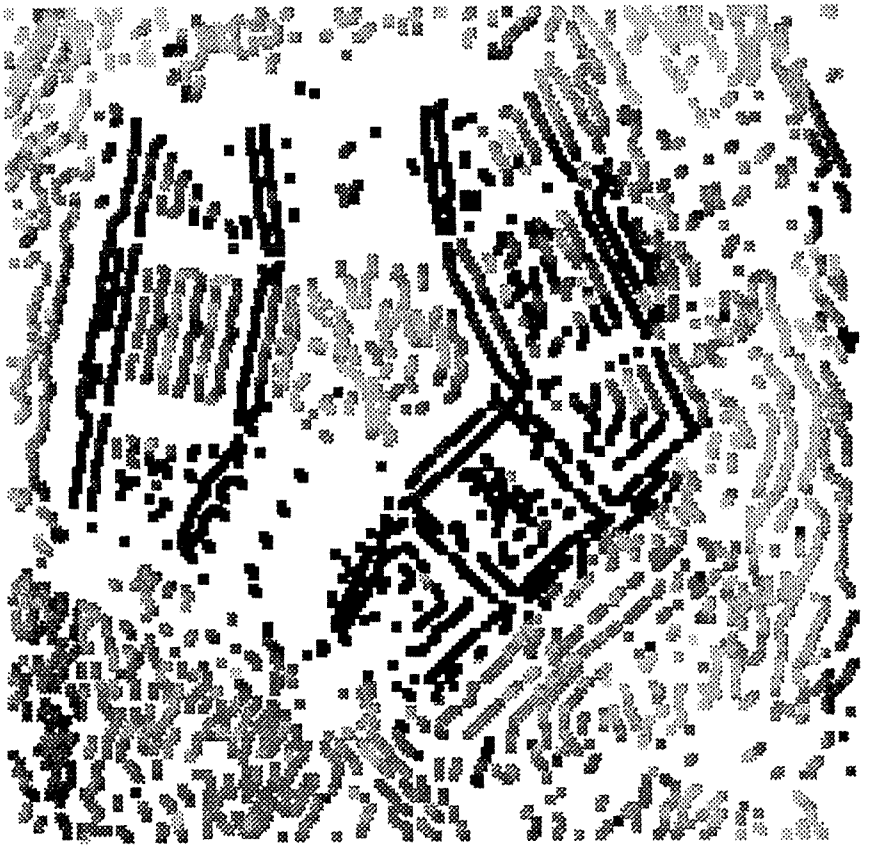


Figure 7.4. Left level 1 disparity results for the pentagon image run using the disparity gradient weight modification algorithm. See page 150 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

This was counterbalanced by additional correct texture matches on the surface of the grapefruit. There were a few more correct matches for horizontal regions in the pentagon image (Figure 7.4). In the sandwich image there was an extra pair of matches outside of the boundary of the sandwich (Figure 7.5). As a final measure, the timing results (in the number of iterations) were quite similar to the originals in each case.

These results demonstrate that the matching algorithm can be made independent of the density of the edges in the images. In addition, the algorithm can now adjust itself to the local density of edges in different regions of the images. Further improvements might be obtained by introducing separate weight density functions and density computations at each resolution level.

7.2. Periodic Image Structures

The remaining problems discussed in this chapter are general to the use of locally defined constraints in stereo matching. The problem considered in this subsection occurs in matching periodic image structures, especially near occluding boundaries. An extreme example of this is the fruit image in Figure 5.10, where the tablecloth is partially occluded by the grapefruit. The General Support Algorithm accepted erroneous matches for the edges from the tablecloth near the occluding boundary, and near the right edge of the image.

The difficulty in selecting the correct matches for periodic structures is inherent in the way the constraints are used to distinguish between candidate matches. Obviously, the constraints are defined in terms of support input to a given match.

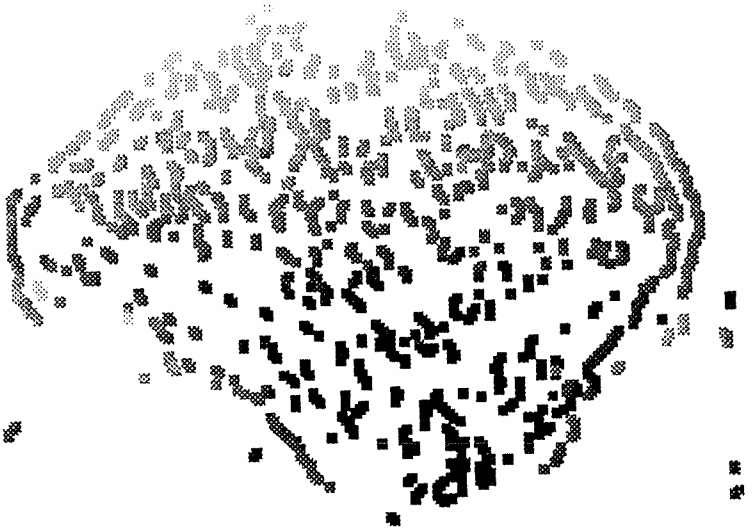


Figure 7.5. Left level 1 disparity results for the sandwich image run using the disparity gradient weight modification algorithm. See page 150 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

However, implicit in their definition and use is the assumption that there is enough variation in the images to produce variations in the support for competing matches, and thereby allow one of the these matches to be selected over the others. This variation must either be structural for the surface structure constraints such as the disparity gradient and figural continuity, or hierarchical for multiresolution. In the interior of a periodic region this variation does not occur (although in many cases

there are slight differences that allow the matches to be distinguished). If the period of the pattern is k, and the correct disparity is d, there will be groups of matches supporting each other at disparities $d-k$, d, and $d+k$. Nothing in the definitions of the support constraints allows these situations to be distinguished.

To resolve the ambiguity inherent in matching edges from periodic regions, stereo algorithms usually employ some mechanism that allows matching decisions at the boundaries of these regions, where there is less ambiguity, to help resolve the other matches. Thus, decisions made at the boundary propagate to the interior of the region using the support constraints. In the General Support Algorithm, our form of uniqueness emphasizes this. (See Example 4.4 and the demonstration of the GSA's behavior in matching it in Section 5.2)

In many cases, the above described behavior of matching algorithms on periodic image structures is correct. However, when the periodic region is partially occluded, significant errors can occur. Specifically, if the magnitude of the discontinuity in disparity across the occluding boundary is l, then l pixels will appear in one image that do not appear in the other. If $l \geq k$, where k is the horizontal frequency of the texture, then the incorrect matches will align near the occluding boundary. Because there is little to distinguish the matches on the interior of this region, the incorrect matches at the boundaries will tend to be accepted. As described above, this will propagate to nearby matches, allowing more incorrect matches to be accepted. (This is exactly what occurs for the fruit image in Figure 5.10.) If the opposite boundary of the region is occluded as well, then it is possible that *every* match accepted in this

region will be incorrect. (For an example of this see Section 8.5.)

7.2.1. Improving Matching For Periodic Regions Through Other Depth Estimation Techniques

The most complete solution to the problem of matching periodic image features involves obtaining an estimation of disparity through orthogonal information sources. This estimation can be used to favor matches whose disparity nearly agrees with the given estimation. The disparity estimation must be obtained independent of any surface structure constraints or image hierarchy constraints which, as we have seen, are unable to distinguish between matches in such regions. There are two other important visual sources of depth information, aside from stereo, that may be useful. These are motion information[78] and focus information.[3,56] They are discussed in turn below.

In Section 2.1.5 we discussed recent work using optical flow and motion in combination with stereo matching. Motion information may provide an independent measure of depth that can be inverted to obtain disparity estimations. Unfortunately, obtaining optical flow and motion measures in periodic image regions may be susceptible to the same problems as stereo matching. This happens if the measurement techniques rely on establishing correspondence between the images. To overcome this problem the magnitude of the image motion must be significantly less than the spatial frequency of the pattern.

Focus information may be used as a depth measurement technique in two ways. First, Krotkov and Bajcsy[3] developed a method that involves (1) determining the focal

length that brings a given object into the sharpest focus, and (2) inverting this focal length to obtain the depth. This method requires active use of the cameras and may also require a number of trials to smooth noisy data.

Pentland[56] developed a different algorithm for obtaining depth estimations. It follows from the observation that, depending on the camera parameters, there is one scene depth where corresponding image points are in precise focus. Other edges are blurred by amounts depending on their distance from the point in focus. Pentland's algorithm determines the standard deviation of the smoothness of an edge and removes the known contribution of the edge detector's smoothing filter. This is a blur measure that can be used to gain a rough estimate of the distance to the point in the scene corresponding to that edge. This estimation may be inverted to obtain the approximate disparity for the edge. The main problem with the use of a blur measure is that it is based on the assumption that there are only step edges in the scene. This does not occur for edges arising from averages of image features or for shadow edges, however.

7.2.2. Incorporating Other Depth Estimation Sources into the General Support Algorithm

In this subsection we investigate how depth estimation techniques might be incorporated into the General Support Algorithm and how effective they may be in overcoming matching errors due to periodic patterns and occlusions. Note that these techniques only provide a coarse estimation of depth. Fortunately, this will be all that is necessary.

Disparity estimations from these sources are used in the General Support Algorithm in the same way as the support through the detailed match constraint. Specifically, the information provides an additional, initial input to those candidate matches whose disparity roughly agrees with their estimation. Since the information is only a function of the edges of an individual candidate match, it is not necessary that it be input to the network over multiple iterations. It only effects the relaxation computation indirectly when the matches that it strengthens propagate their influence to other matches. As a final note, in a parallel vision system this additional disparity information might not be available before the first iteration. In this case it would be incorporated later, when it becomes available.

The potential effects on matching of using additional sources of disparity estimations are demonstrated using the fruit image (Figure 5.10). To show the effects, the correct matches in the upper regions of the images containing the tablecloth were provided with additional initial activations. Specifically, matches for edges in those regions whose disparity ranged from -14 to -8 were given an additional 0.02 initial activation (the base initial activation was 0.2). This was done for both the left and right images, so the maximum additional activation any match could receive was 0.04 (20% of the base activation). Figure 7.6 shows the level 1 left disparity images for both the original simulation and the new one. Note the dramatic reduction in the number of erroneous matches and missing matches near the occluding surface and near the image boundary. The new version produced 97% correct matching decisions whereas the previous version produced only 93% correct decisions. Finally, Figure

Figure 7.6. Results for the original run of the fruit image (top), and the new version with certain matches favored initially (bottom). See page 155 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

7.7 graphs the percentage of intermediate outputs as a function of the iteration number for the original and new matching results.

As is readily apparent, the potential effects of using other, independent sources of depth information such as the focal gradient, are significant. They improve the matching results for periodic image structures by providing an initial, coarse indication of disparity. Even though the range of disparities indicated by these information sources may be broad, they influence the matching enough that the correct matches can be selected in ambiguous matching situations.

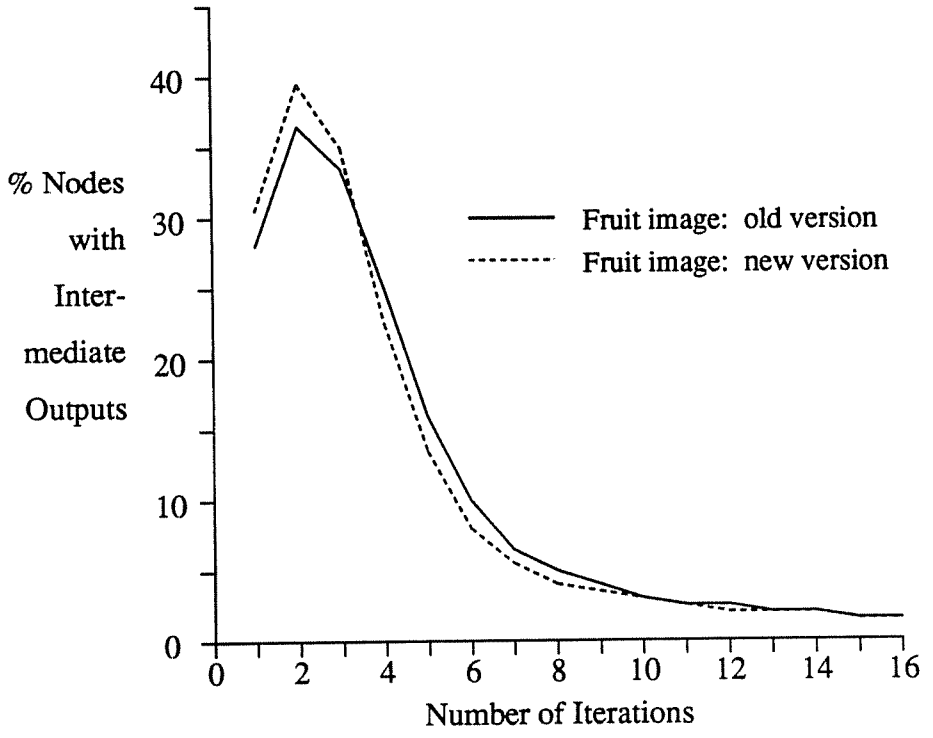


Figure 7.7. The number of nodes with intermediate activations as a function of the iteration number for the old and new runs of the fruit image.

7.3. Occlusion Detection

Occlusions are another general problem for stereo matching algorithms. Near occluding boundaries there tend to be two types of problems. First, erroneous matches for an edge can receive significant support across the boundary. This support may be great enough that these matches are accepted, especially when the correct match for an edge is not available. When there is a vertical component to the occluding boundary a second type of problem arises as well. In this case there will be regions that appear in only one image. They will be in the left image when the occluded surface is on the left side of the boundary, and they will be in the right image otherwise. No matches should be accepted for the edges in these regions. However, because of erroneous support across the occluding boundary, matches are sometimes found for the occluded edges. For example, in the multilayer stereogram in Figure 5.4, matches were found for 64 of the 602 matchable edges in the occluded region.

The matching problems occurring near occluding boundaries can not be solved directly using locally defined constraints. Recall that although the assumptions of the constraints state relationships between how the points on object surfaces appear in the images, their definitions do not explicitly discriminate between support for pairs of matches on a surface and pairs of matches from different surfaces. Discrimination occurs only implicitly by limiting the range of support and the disparity differences between pairs of matches. This is particularly a problem with the disparity gradient since it is fairly liberal in its definition of support and it is defined over longer distances than the other constraints. Thus, it tends to support erroneous matches

across occluding boundaries. In practice, the General Support Algorithm can eliminate many occlusion errors. It does this implicitly through the interaction of the support constraints with decay and uniqueness. For example, a small percentage of the occluded edges in the multilayer stereogram actually had matches accepted.

Improving stereo matching behavior near occluding boundaries requires the addition of two related, non-local matching mechanisms. First, there needs to be some means of detecting step changes in disparity and using this to inhibit disparity gradient support across these boundaries. This must occur during matching and is useful for eliminating erroneous support. The second mechanism can be used after matching has completed. It requires detecting the exact position of occluding boundaries as well as the disparity change across the occlusion. This can then be used to locate occluded regions and remove any matches found for edges in those regions. Below we present ideas for building these mechanisms.

A number of researchers have attempted to incorporate occlusion detection into stereo matching. These include both low-level and high-level processes. Some algorithms have fit surfaces to the disparity data using approximation techniques and examined the results looking for occlusions.[14,30] Unfortunately, these techniques introduce some additional problems. Mainly, they are sensitive to noise in the disparity data and may fit surfaces where there should not be any. In addition, the complexity of complete surface approximation may not be necessary. A second type of approach involves building the search for discontinuities directly into regularization equations for stereo matching.[13,72] Unfortunately, these approaches assume that the

disparity data are relatively dense. They also require a large number of iterations.

In the remainder of this section we sketch an approach for solving the occlusion problem. What we propose is similar to both types of previous strategies. It is based on the assumption that the local variation in disparity is relatively small except near occluding boundaries. Thus, it uses average-disparity detectors that respond when there is a group of matches in a given area with similar disparities. These nodes will be used to recognize step changes in disparity. In this way, occlusion detection is physically separated from the matching network, but it does not require sophisticated surface reconstruction techniques.

Occlusions are located using nodes that respond to average disparities in a region. These nodes overlap each other in both spatial dimensions and disparity dimensions. This is an example of coarse coding that has been used frequently in connectionist models.[27] The nodes are activated as follows. Each node receives input from match nodes that are within its spatial and disparity boundaries. The idea is that when a large enough group of match nodes have medium to large outputs, an average-disparity node's output will rise quickly. Its equation is of the form:

$$f(A) = \frac{1}{1 + e^{(-A + K) / t}}$$

where A is input from the match nodes, t controls the sharpness of the curve, and K shifts the position of the knee of the curve. See Figure 7.8 for a graph of the function. K is an additional input to the average disparity node that is based on the density of the matchable edges in the region. K increases with increases in edge density. This

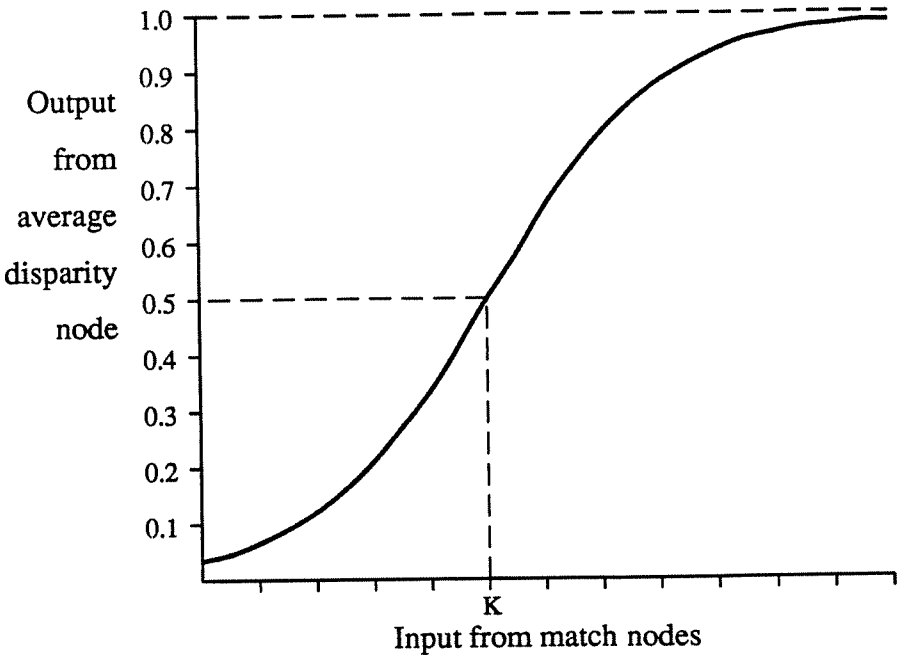


Figure 7.8. The output response of the average disparity nodes as a function of the input from the match nodes.

allows the node to differentiate between many weakly responding matches in a region with a dense set of edges, and a smaller group of strong responses in a more sparsely populated region. Thus, the disparity node's response is moderated by the density of matchable edges in the region. Finally, there will be additional inhibiting input to this node, not shown in the above equation, from nodes in the same region whose disparity range is outside this one's range. This prevents the response of any average-disparity detection nodes in ambiguous regions.

The outputs of the average disparity nodes are used in matching in two ways. These correspond to the desired mechanisms outlined above. First, let i be a node in region A_i that responds to disparity d_i. Let A_j be a region adjacent to A_i. Finally,

suppose that node i is activated by match nodes in its region, but none of the nodes in region A_j with similar disparities are activated. This indicates a step change in disparity. Hence, all supporting connections from match nodes in region A_i to match nodes in region A_j are made inactive through conjunctive inputs. This eliminates erroneous disparity gradient support across surface boundaries. Note that if there is an occluded region between A_i and A_j in the other image this will implicitly help eliminate matches for these edge by eliminating an important source of support for those matches. This mechanism assumes that the matches for the occluded edges accumulate support more slowly than the correct matches in region A_i. Otherwise, these erroneous matches might be accepted before the support from region A_i is removed.

The second use of the average-disparity nodes is to completely eliminate matches for edges that are found to be occluded. Let A_0 and A_1 be spatially adjacent regions in the right image. Let d_0 be the average disparity in A_0, and d_1 be the average disparity in A_1. Assume that A_0 is to the right of A_1 and that $d_0 > d_1$. This is shown in Figure 7.9. In this situation there will be an occluded region of width $d_0 - d_1$ in the left image. All matches for edges in this region should be eliminated. This may be implemented by introducing an additional node for each image location that recognizes when an edge is occluded and suppresses all matches for that edge. The node will be activated by conjunctive connections from disparity detectors in the opposite image, such as in the above example.

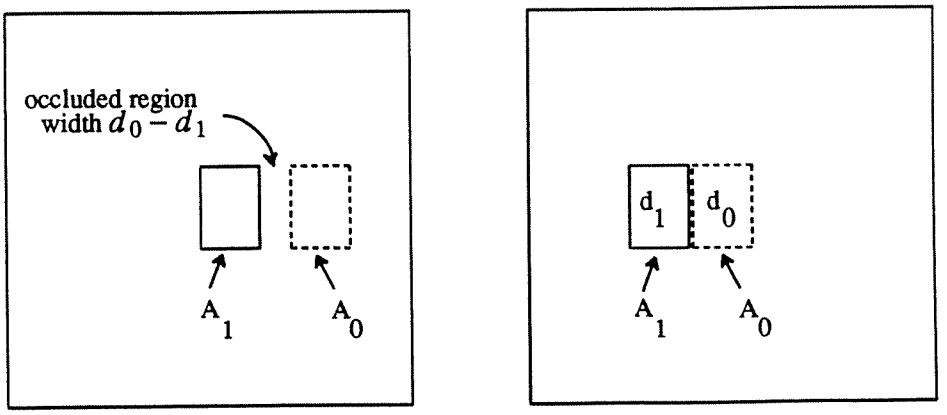


Figure 7.9. The adjacent regions with different disparities in the right image indicate the presence of an occluded region in the left image.

The preceding discussion has described an approach for solving the occlusion problem. It provides only an outline of the ideas, with much work remaining to be done. While the proposed mechanisms can potentially help reduce occlusion errors in stereo matching, there are a number of potential difficulties with them as well. (1) They will be most effective in recognizing step changes in disparity when these changes are large. Unfortunately, this situation is also the easiest for the local matching in the General Support Algorithm to handle implicitly (see the results of matching the stereogram in Figure 5.2). Smaller scale occlusions are more difficult (although they indicate smaller occluded regions). (2) The occlusion mechanisms are most naturally suited to random-dot stereograms that depict planar surfaces. Extending them to real images where the surface orientations vary more significantly and the data are more sporadic, may be difficult. (3) These mechanisms *are not* useful for solving the periodic images problem. Simply eliminating the support across an

occluding surface will not correctly resolve the matches in a periodic region. The magnitude of the occlusion must be determined. This involves recognizing the average disparity of both the front and back surfaces. Unfortunately, when the occluded surface is periodic there will be conflicting interpretations for it and the average disparity of the matches in this region will not be known. Thus, the magnitude of the occlusion can not be determined.

In spite of these potential difficulties, the proposed non-local matching mechanisms hold promise for helping to solve the occlusion problem. They have a natural implementation in a connectionist network and can therefore be incorporated easily into the General Support Algorithm. Developing these ideas further requires implementing and testing them on random-dot stereograms and real images. This is an interesting area of future research.

7.4. Significant Appearance Differences Between the Images

The final problem that locally defined matching constraints have difficulty overcoming occurs when there are significant appearance differences between corresponding regions of the images. A number of examples of this problem were noted in Chapter 5 in examining the results of matching real images. One particular example is shown in Figure 7.10 where parts of the intermediate level edge detection images from the Pentagon image are shown in larger scale. Note the variation between the images, especially in comparing the left side of each image.

Differences in the appearance of the images cause two major problems for stereo matching algorithms. The first is when no match is found for an edge. This can arise

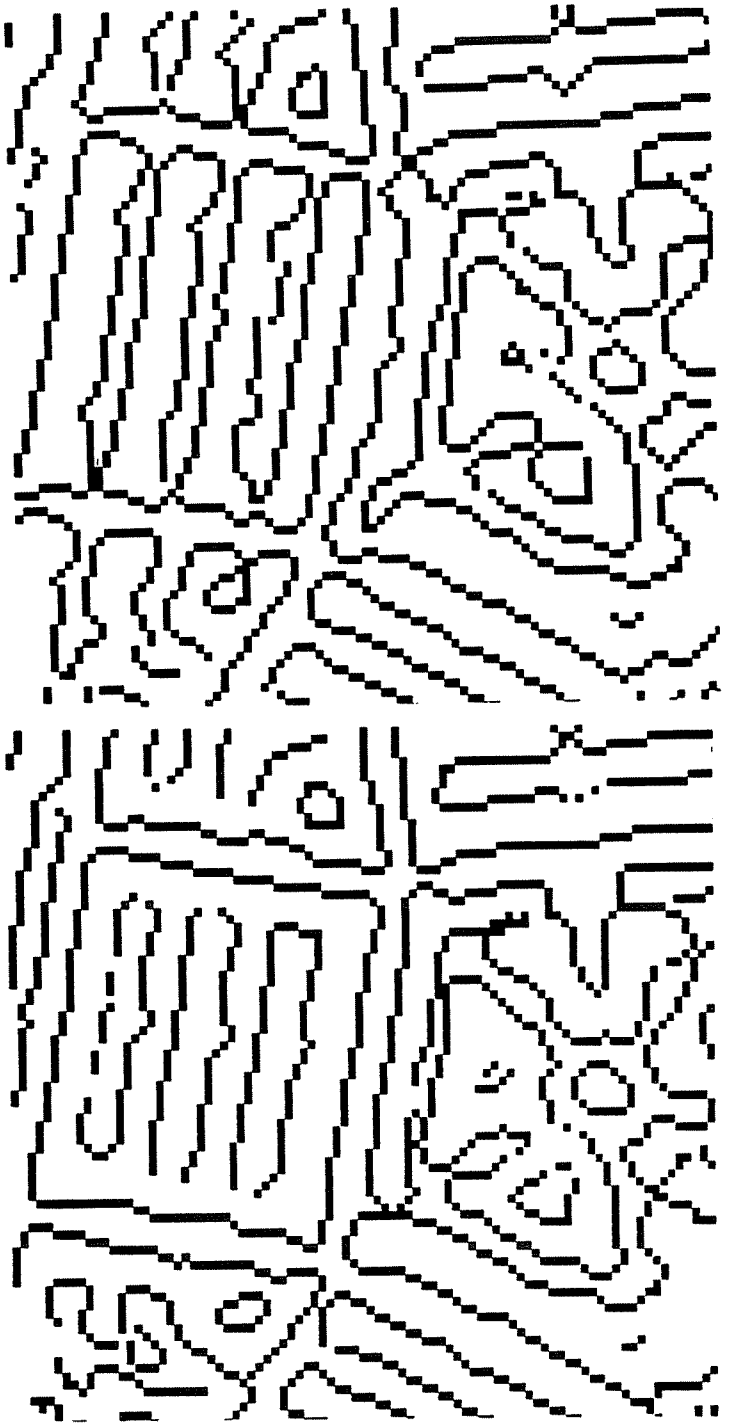


Figure 7.10. Intermediate level edge detection results for corresponding parts of the Pentagon images. The left image is shown above the right image.

either when the appropriate candidate match is missing, or when this candidate match is present but does not receive enough support because of the variations between the images. The second major type of error is more prevalent. This occurs when an erroneous match is accepted for an edge because the correct match was missing. Both of these errors contribute greatly to the number of incorrect matching decisions made by the General Support Algorithm.

The locally defined matching constraints can not directly handle either type of problem arising from appearance differences between images. Thus, they can not distinguish between support for an erroneous match when the correct match is missing and weak support for a correct match. The disparity gradient often helps to overcome missing nearby structural support for a match by gathering support over larger regions of the images. Unfortunately, this flexible support gathering can cause erroneous matches to receive enough support to be accepted when the correct match is missing.

The above problems may be solved by removing their source. That is, by reducing the arbitrary variation between edge detection results the matching can be improved. The Marr-Hildreth edge operator used in this algorithm has some well-known problems. The operator's optimal response is to isolated step edges.[52] For other edges the edge position found tends to be shifted from the true position. The operator tends to introduce errors in edge positions through its smoothing filter. When there is noise in the image and when the smoothing filter crosses multiple features, contours can change in seemingly arbitrary ways. To address these types of problems, Kass[38] has studied edge errors in the context of stereo matching and has developed

methods of assigning matching confidences to edges. Edges with low confidences simply are not matched. This leads to a lower number of matched edges, but fewer matching errors.

A second approach to reducing the arbitrary appearance differences between images is to allow cooperative interaction between edge detection and matching. As before, the initial results of edge detection would initiate matching. Later, the matching results after a few iterations could align corresponding regions and allow cooperative interactions to reduce the arbitrary differences between the images and lead to a denser, more accurate set of matches. This approach has not been developed in any detail. It will be the subject of future research with the goal of reducing the differences that lead to matching errors.

7.5. Chapter Summary

In this chapter we have examined a number of potential solutions for the problems in using only locally-defined constraints for stereo matching. These solutions include using focus information to provide a coarse estimation of disparity and resolve the ambiguity in a periodic region, using average disparity detectors to locate occlusions and eliminate matches for occluded edges, and developing interactions between edge detection and matching to eliminate structural variations between images. In addition we developed a method for adjusting the influence of the disparity gradient that eliminated the General Support Algorithm's sensitivity to edge densities in images. The solutions were added to the existing General Support Algorithm. In the next chapter we investigate a different approach to overcoming the

matching problems by using a third image and new constraints to integrate the matching between multiple image pairs.

CHAPTER 8

Trinocular Stereo

In the previous chapter we investigated ways that the remaining problems in stereo matching can be overcome through the use of non-local constraints and information obtained from other intrinsic images. Those solutions used additional mechanisms to handle specific problems. In this chapter we investigate a different approach to overcoming these problems. This involves incorporating a third camera into stereo matching as shown in Figure 8.1. The General Support Algorithm can be extended to this trinocular matching approach in the following way. Matches are located between the base and right image as before, and between the base and top images as well. The two pairs of matching images use the original General Support Algorithm. In addition, the Trinocular General Support Algorithm uses two additional constraints, trinocular uniqueness and the trinocular disparity gradient, to define relations between the vertical and horizontal matching networks. The resulting *cooperative trinocular matching* helps to overcome some of the problems highlighted in Chapter 5, including partially-occluded periodic regions, occlusions, and structural differences between the images.

As a brief aside, consider the possibility of matching between the top and right images in addition to vertical and horizontal matching. First, note that in interpreting the results of matching we find disparity values with respect to one image. In the trinocular case this is the base image. Matches between the top and right images

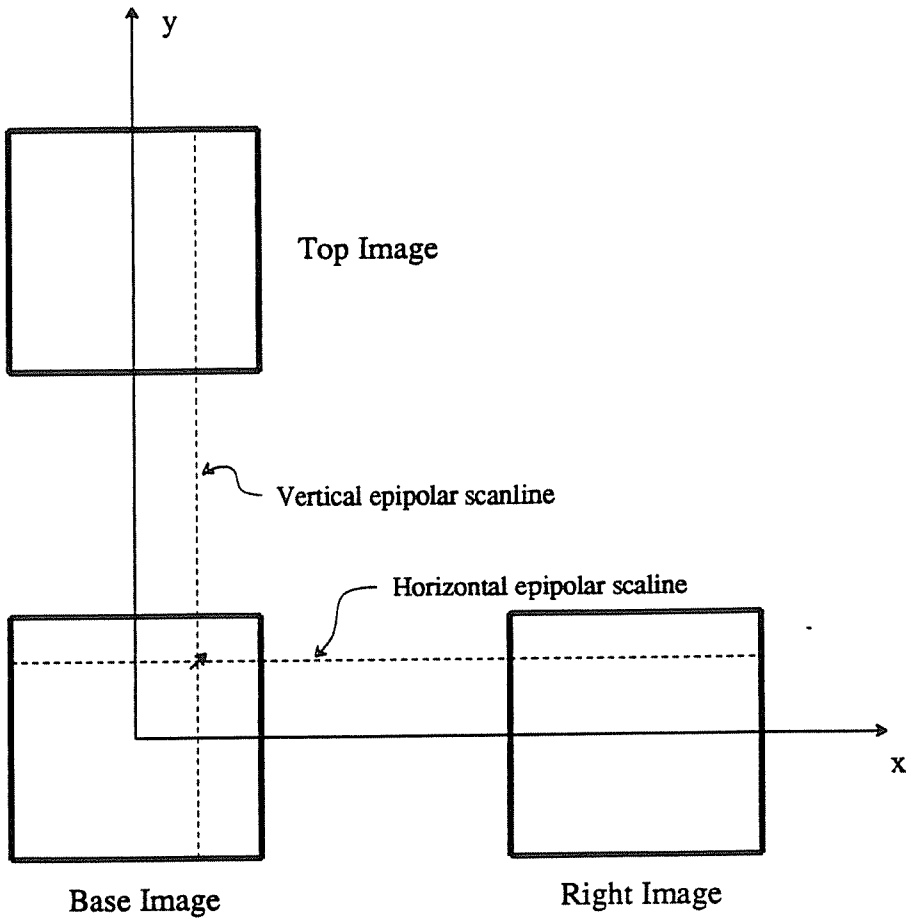


Figure 8.1. Trinocular stereo imaging geometry. The base image matches both the top and the right images.

could potentially influence the disparity results in the base image. However, the trinocular constraints defined between vertical and horizontal matches for edges in the base image capture this information directly. Therefore, the matches between the top and right images are unnecessary.

The remainder of this chapter describes the Trinocular General Support Algorithm (TGSA) in detail. It is divided into the following sections. Section 8.1

defines the imaging and matching geometry. Section 8.2 outlines the previous work on multiple camera matching, emphasizing algorithms that employ a similar imaging geometry. Section 8.3 describes the TGSA carefully, including the definition of the new trinocular constraints. Section 8.4 discusses why this trinocular algorithm overcomes many of the problems of binocular matching. Finally, Section 8.5 describes the experimental results using synthetic images representing occluded periodic structures and noisy random-dot stereograms.

8.1. Trinocular Geometry

The imaging geometry employed in the TGSA is shown in Figure 8.1. All three cameras are assumed to be identical and are positioned on the vertices of an isosceles right triangle. The lines of sight of the cameras are all perpendicular to the $x-y$ plane. The isosceles triangle assumption ensures that the same vertical and horizontal disparity indicates the same depth. The General Support Algorithm is employed for both vertical and horizontal matching. In addition, as mentioned above, there are two new constraints used in the trinocular algorithm that specify relations between vertical and horizontal matches. The first, **trinocular uniqueness** states that both vertical and horizontal matches for an edge should have the same disparity. The second, the **trinocular disparity gradient** defines relations between nearby pairs of vertical and horizontal matches based on their relative disparities. These are discussed in detail in Sections 8.3.1 and 8.3.2. (Note that when we discuss relations between vertical and horizontal matches we will refer to them as *orthogonal matches*).

8.2. Prior Work on Multi-Camera Matching

Recently there has been a significant amount of interest in the use of more than two cameras for stereo vision. The approaches described in the literature vary in the relative positions of the cameras and in the methods for using the additional information in matching. In this section we review these approaches, highlight some of their weaknesses, and show how the TGSA differs from them.

An interesting early approach involves Tsai's[75] use of a large number of cameras at fixed, known positions to reduce errors in matching. Based on the disparity of a candidate match between points in two of the images, the positions of the hypothesized scene point in all other images are known. For each candidate match these additional images provide a statistical measure that helps to increase the confidence of a correct match. Tsai demonstrated that this method works well with noisy images. Unfortunately, aside from the obvious expense of the additional cameras, the method is sensitive to occlusions that cause a given point to appear in only a few of the images. With fewer cameras and more sophisticated constraints we can overcome the matching errors that this system is intended to solve.

The most popular approach to multi-camera stereo uses three cameras positioned at the vertices of a triangle, usually either an equilateral triangle, or an isosceles right triangle. These algorithms differ in their methods of employing the third image. Some use it simply for eliminating the horizontal matches problem.[57] That is, when an edge or matching segment is vertically oriented, its corresponding match is sought in the horizontal image; when the edge is horizontal the match is sought vertically.

While this may produce a denser set of matches, the results are limited by the quality of matching between a pair of images. Therefore, it is susceptible to the same problems as binocular stereo matching.

Another approach uses the third camera as an additional source for measuring the accuracy of a match.[33,82] Given a match between two images, the disparity of that match is used to predict the location of the corresponding point in the third image. Unfortunately, matching points may not appear in all three images because of occlusions and errors in the positions of edges (especially those oriented along horizontal or vertical epipolar scanlines). To account for this problem some trinocular algorithms employ more structured matching primitives.[2] These algorithms are not susceptible to the problem of matching along epipolar scanlines because they are two dimensional. However, they rely on domain-dependent assumptions (usually indoor or urban scenes) in order to allow the matching primitives to be reliably extracted. Another general problem with approaches that rely on confirmation of the match position is that when three images are involved a larger number of points are occluded in one of the images.

The final type of approach is to match both vertically and horizontally using an existing stereo matching algorithm, and "smooth" the results together.[17,54] This assumes that the errors arising from the matching between each pair of images do not overlap. In addition, it assumes that the smoothing process will favor correct matches over errors in a given region. This is not always the case in matching situations such as periodic regions where the errors tend to be systematic. Finally, this approach

ignores the possibility that weaknesses in the support between two images can often be overcome by using the third image during matching.

8.3. The Trinocular General Support Algorithm

The Trinocular General Support Algorithm (TGSA) offers both an alternative to existing trinocular stereo algorithms and a method for overcoming the errors observed in the binocular General Support Algorithm. It does this by (1) employing the existing GSA both vertically and horizontally, and (2) defining new constraints between orthogonal matches. In this algorithm, strongly supported binocular matches require no trinocular support. Thus, when a match is occluded only vertically or horizontally, but not both, it can be accepted solely through binocular support. However, in circumstances where binocular support is either weak or noisy, the addition of trinocular constraints can help to resolve ambiguous matches.

The TGSA involves both vertical and horizontal implementations of the original GSA. Thus, when a contour is oriented along either the vertical or horizontal epipolar scanline, the edges along that contour will only have matches in the horizontal or vertical image respectively. Other edges may have matches in both images. The vertical and horizontal matches are related through the trinocular uniqueness constraint and the trinocular disparity gradient. These constraints specify weighted connections between matches just as in the binocular GSA. These two new constraints are discussed in Sections 8.3.1 and 8.3.2. Section 8.3.3 discusses the assignment of connection weights in the TGSA. Finally, Section 8.3.4 analyzes the size of the new network.

8.3.1. Trinocular Uniqueness

The trinocular uniqueness constraint is based on the same assumption as binocular uniqueness. In the binocular case we assumed that each image edge point corresponds to one scene feature, so there should be at most one valid match per edge. Extending this to the trinocular case, we assume that the valid matches found both vertically and horizontally for a given edge should correspond to a single scene feature. Thus, because of the geometry of the camera positions, these matches should have the same disparity.

Using these assumptions we define the trinocular uniqueness input to a horizontal match $m = (b, r)$, where b is the point in the base image and r is the point in the right image, as

$$\beta_t \times \max(O_j \mid j \in Top_b \text{ and } \mid d(i) - d(j) \mid > k)$$

where Top_b is the set of vertical matches for base image point b, $d(m)$ is the disparity of a match, and k is a small constant. In other words, the trinocular uniqueness inhibiting input to a horizontal match is the maximum of the vertical matches for the base image edge whose disparities differ from the horizontal match disparity by more than a small constant. The trinocular uniqueness for a vertical match is defined similarly.

There are several important points to note about trinocular uniqueness. First, ideally, the constant k should be 0 to reflect the fact that the vertical and horizontal disparities should be identical. However, because of potential errors in the position of

an edge, there is a slight tolerance for minor differences in disparity. Second, there is *only one* trinocular uniqueness input to a given match, whereas there are *two* binocular inputs to a match. Finally, and most importantly, a large percentage of the matches have no trinocular uniqueness input. To see this, recall that because of the difficulty in matching edges on horizontal segments in the original General Support Algorithm a significant percentage of the edges were not matched. Empirically we found this to be approximately 30% in random-dot stereograms. In the trinocular algorithm, 60% of the edges will have only vertical or only horizontal matches; approximately 40% edges will have matches both vertically and horizontally. These are the only matches that will have trinocular uniqueness input.

8.3.2. Trinocular Disparity Gradient

The trinocular disparity gradient (TDG) is motivated by the same assumptions as the original disparity gradient. As in the latter, the main assumption is that points from a surface appear at similar positions and with similar disparities in the images. Thus, the trinocular disparity gradient defines relations between nearby, orthogonal matches (one match is vertical and the other is horizontal) based on the relative distance between them and on the difference in their disparities. We also use the original binocular disparity gradient to define relations between pairs of vertical matches or between pairs of horizontal matches.

In defining the trinocular disparity gradient equation, a straightforward generalization of the original disparity gradient equation is insufficient. To see this, recall that in the original equation, matches $m_0 = (l_0, r_0)$ and $m_1 = (l_1, r_1)$, where the

l_i are left image points and the r_i are right image points, will support each other if

$$\frac{|\, d(m_0) - d(m_1) \,|}{D(m_0, m_1)} \leq 1.0$$

where $d(m)$ is the disparity of a match, and D is the distance between matches. This distance was computed by taking the midpoint between the left and right image points for each match, and measuring the distance between these midpoints. This is shown in Figure 8.2. Generalizing this to the trinocular case would yield the following. Let $v_0 = (b_0, t_0)$ be a vertical match, where t_0 is a point in the top image and b_0 is a point in the base image. Similarly, let $h_1 = (b_1, r_1)$ be a horizontal match. Using the same disparity gradient equation as above we have:

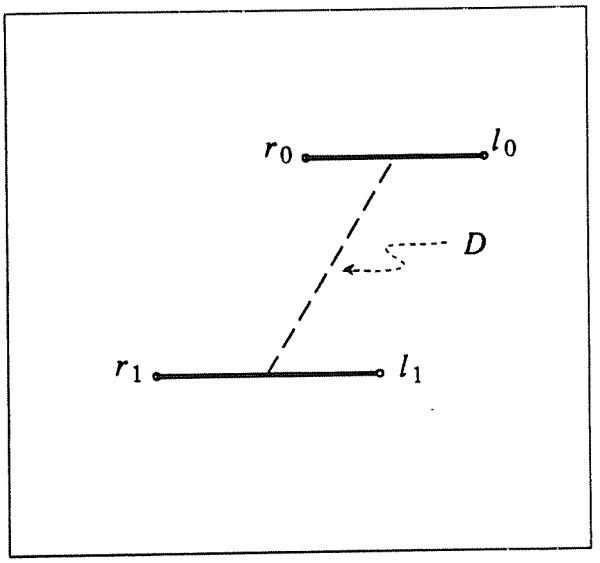


Figure 8.2. The distance D between two matches in the disparity gradient is the distance between the midpoints of the matches.

$$\frac{| d(v_0) - d(h_1) |}{D(v_0, h_1)} \leq 1.0$$

The generalization of the distance relation is shown in Figure 8.3. The problem with this definition is that the line segment between the points forming the vertical match and the line segment between the points forming the horizontal match are orthogonal. When these segments cross, the distance between midpoints is no longer an accurate measure of the desired property. An example of this is seen in Figure 8.4. The disparity difference between these two matches is quite small (2 units), but their distance as defined above is 0. Thus, according to the above definition, they do not support each other.

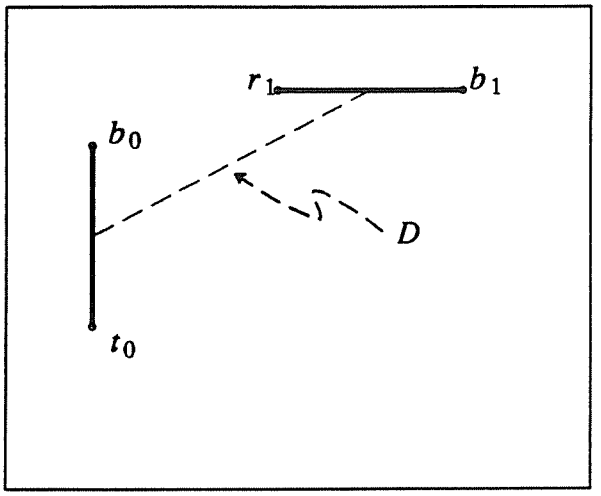


Figure 8.3. Measuring the distance between matches in the trinocular disparity gradient using a similar measure as in the original disparity gradient.

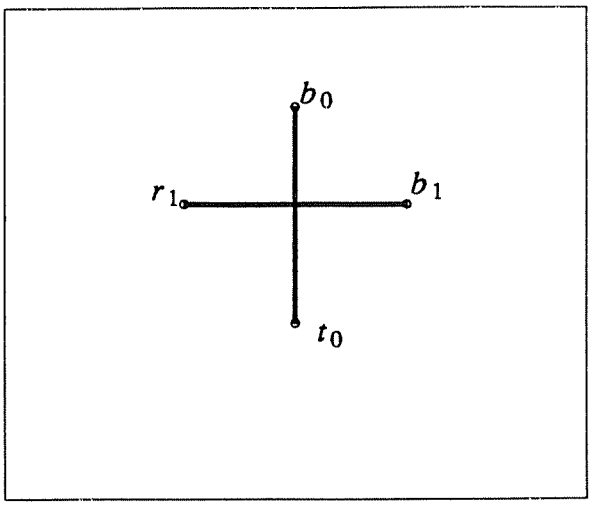


Figure 8.4. An example where the straightforward extension of the disparity gradient distance definition is problematic. The disparity difference between these matches is quite small, but the measured distance is 0.

Because of the problems in using the original distance measure, we need a new measure for the trinocular disparity gradient. The simplest approach is to measure only the distance between points in the base image. Using this, the trinocular disparity gradient is defined as follows. Let $v_0 = (b_0, t_0)$ be a vertical match and let $h_1 = (b_1, r_1)$ be a horizontal match. They support each other if

$$\frac{| d(v_0) - d(h_1) |}{D(b_0, b_1)} \leq 1.0$$

where $D(b_0, b_1)$ is now simply the Euclidean distance between b_0 and b_1. This is demonstrated in Figure 8.5.

This definition introduces some minor inconsistencies between the existing definition of the disparity gradient and the new definition of the trinocular disparity

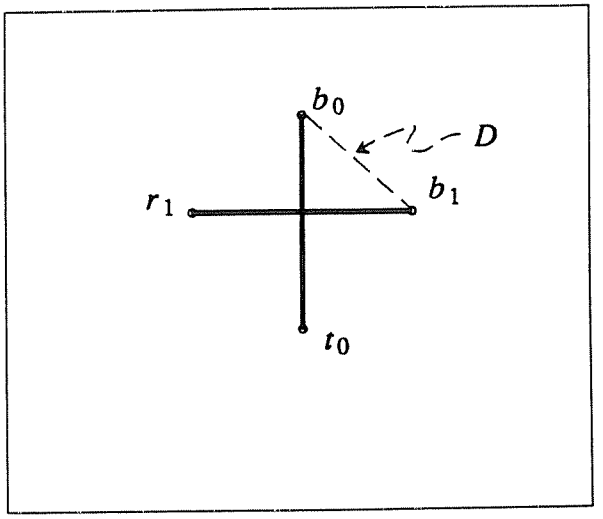


Figure 8.5. The trinocular disparity gradient distance between a vertical and a horizontal match.

gradient. However, it actually offers a compromise between the definitions of distance used in vertical binocular matching and horizontal binocular matching. To see the difference, let $b_0 = (x_0, y_0)$ and $b_1 = (x_1, y_1)$ be points in the base image and let d_0 and d_1 be the disparities for those points. When these points and disparities define horizontal matches, the distance between the matches is (letting $\Delta d = d_0 - d_1$, $\Delta x = x_0 - x_1$, and $\Delta y = y_0 - y_1$):

$$D_h = \sqrt{\Delta y^2 + (\Delta x - \Delta d / 2)^2}.$$

If the points and disparities describe vertical matches, then the distances between the matches is:

$$D_v = \sqrt{(\Delta y - \Delta d / 2)^2 + \Delta x^2}.$$

There is no logical inconsistency between these two distance measures. This is because the image pairs, including the epipolar scanlines, are 90^0 rotated from each other. Now let's examine the distance measured used in the trinocular disparity gradient. If the points and disparities above describe one vertical and one horizontal match, then the distance between the matches is:

$$D_t = \sqrt{\Delta y^2 + \Delta x^2}.$$

Because one vertical match and one horizontal match are used, the difference in disparity does not enter into either the x distance or the y distance. Thus, the trinocular disparity gradient distance definition represents a combination of the vertical and horizontal definitions used in the binocular disparity gradient.

The final consideration in the definition of the trinocular disparity gradient concerns the relationship between vertical and horizontal matches for a given point in the base image. Specifically, because the distance between the matches is 0, the trinocular disparity gradient equation is undefined. This is not a problem in the binocular disparity gradient because different matches for a given edge *always* inhibit each other. The situation is handled by defining the minimum distance in trinocular disparity gradient between matches to be 1. In addition to handling the problem, it introduces a minor tolerance in the difference between the vertical and horizontal match disparities for an edge.

8.3.3. Weight Equations and Parameter Assignment

In the Trinocular General Support Algorithm (TGSA) the new trinocular constraints are added to the existing constraints defined by the original General Support Algorithm (GSA). To complete the description of the trinocular algorithm the weight parameters for these new constraints must be assigned. In addition, a minor adjustment is necessary in the binocular disparity gradient weight parameter. Reasons for this are discussed below.

The parameters defining the strengths of the new constraints are β_t for trinocular uniqueness, and *BASE_TDG* for the trinocular disparity gradient. β_t was introduced above in defining trinocular uniqueness. *BASE_TDG* is built into a trinocular disparity gradient weight equation that is similar to the equation for the original disparity gradient. Specifically, let v_0 and h_1 be candidate matches that support each other through trinocular disparity gradient. The weight of the connection between them is defined as follows. Let $disp\_diff = |d(v_0) - d(h_1)|$. Then the weight is given by:

$$w_{i,j} = \frac{BASE\_TDG}{D(v_0, h_1)} * \frac{c}{(disp\_diff + c)}$$

As in the disparity gradient, c is a constant that controls the skewing of the weight in favor of fronto-parallel surfaces. In doing so it increases the discriminatory ability of the constraint.

The actual weight values are set by specifying values for parameters β_t and *BASE_TDG*. We consider each in turn. First, the trinocular uniqueness parameter β_t

should be set below the value of the binocular uniqueness constraint β. The main reason for this follows from our observations about the percentage of edges that have both vertical and horizontal candidate matches. As noted above, approximately 60% of the edges will have matches in only one direction. In addition, on average, the only variation in support as a result of having matches in both directions will be the potentially strong trinocular disparity gradient input from an orthogonal match with the same disparity. Thus, the trinocular uniqueness parameter β_t is set relatively low.

The parameter *BASE_TDG* is assigned based on the following observations. In the TGSA no distinction is made between disparity gradient support and trinocular disparity gradient in the strength of the relations they define. The constraints both gather support by searching over a small neighborhood around a match for other matches with similar disparities. The disparity gradient searches for support between the same pair of matching images, while the trinocular disparity gradient searches orthogonally. The only minor variation in the two constraints' definitions of support is that in the binocular case some nearby matches define support through figural continuity. In the trinocular case there is no similar constraint. However, this should not affect the strength of the parameters. Thus, the parameter *BASE_TDG* will be set to correspond exactly to that of *BASE_DG*.

In practice, the combined strength of the disparity gradient and the trinocular disparity gradient in the TGSA is only slightly higher than that of the disparity gradient alone in the GSA. The main reason for this follows from considering what the algorithm will accept as a valid match. Specifically, in the GSA a match is

accepted when its support input during the iterations of the network outweighs the inhibiting strength of uniqueness and decay. Thus, a balance is established between the positive and negative influences on a match. In the trinocular algorithm only a minor inhibitory influence is added. Thus, the combined trinocular and binocular disparity gradient constraints should be only slightly stronger than the initial disparity gradient. An alternative to this would be to assign the value of *BASE_TDG* to the value used for *BASE_DG* in the binocular case, and then normalize the overall strengths of all the positive constraints. However, the trinocular disparity gradient has the same weakness as the binocular disparity gradient, i.e. it is indiscriminate in defining support. The other positive constraints are generally less ambiguous. Reducing their strengths with respect to both disparity gradient constraints would make the algorithm more susceptible to noise matches that gather support through the disparity gradient constraints.

8.3.4. Size of the Trinocular General Support Algorithm Network

As a final consideration in the definition of the TGSA, we examine the size of the network defined by it. This is important both for the simulation of the algorithm and for any potential parallel implementation of it. Let the number of nodes defined by the original binocular GSA be N_b and let the number of connections be C_b. In the trinocular algorithm the number of nodes is:

$$N_t = 2 * N_b$$

This is because the number of nodes in the TGSA network is equal to the number of

nodes in two separate GSA networks. However, the number of connections is:

$$C_t = 4 * C_b$$

This result can most easily be seen as follows. First, in the binocular case the number of disparity gradient connections dominates the total number of connections. Also, because the trinocular disparity gradient involves a similar definition of support and a similar search area, it defines on average the same number of connections as the binocular disparity gradient. Thus, there are twice the number of connections per node, and so, because there are twice the number of nodes, there are four times the number of connections defined throughout the network. Note that these size results hold regardless of whether total numbers of nodes and connections are considered, or whether only the number of active nodes and connections for a given pair of images is required.

8.4. Solutions to Problems in Binocular Matching

The Trinocular General Support Algorithm is designed to solve a number of the problems in matching binocular image pairs. These include problems with partially occluded periodic regions, other occlusions, and structural differences between the images. In addition, horizontal matches are no longer a problem since matches are sought both vertically and horizontally. When an edge is part of a horizontal contour, it will have matches only vertically. Because the original GSA is implemented vertically as well as horizontally, having only vertical matches is sufficient for finding the correct match for horizontally oriented edges.

The most dramatic improvement that the TGSA offers is in handling the periodic structures problem. As the fruit image example showed in Chapters 5 and 7, this problem can result in clusters of erroneous matches near an occluding boundary. In a periodic region there are no variations in support to allow the valid matches to be distinguished. Therefore the disparities for matches found at the boundaries of these regions must be propagated into the interior of the region to resolve ambiguities. When the boundary is occluded, boundary matches may be incorrect and errors may occur.

The TGSA can overcome problems in matching periodic regions through the trinocular disparity gradient. In a periodic region where there is either is no vertical periodicity or where the vertical and horizontal frequencies are not equal, the trinocular disparity gradient provides stronger input to the valid matches than it does to the invalid matches. This differential implies that the ambiguity in structural support no longer exists, and the valid matches can be identified without relying on the propagation of the disparities found at the boundaries.

The performance of the TGSA will be demonstrated on periodic regions in the next section. However, analytically, it is easy to see how the trinocular disparity gradient establishes the support differential necessary to resolve the ambiguities. Assume that the correct disparity in a periodic region is d and the vertical and horizontal frequencies in the region are v and h, respectively. The competing horizontal matches will have disparities $d-h$, d and $d+h$. The competing vertical matches will have disparities $d-v$, d and $d+v$. Through the binocular disparity

gradient each of the competing disparities for an edge on the interior of the region will receive approximately the same support. This is the source of the ambiguity. However, the trinocular disparity gradient will support the matches at disparity d much more strongly than the others because there are both vertical and horizontal matches with this disparity. Assuming that $h \neq v$, the incorrect matches receive much less support through the trinocular disparity gradient. This observation is strengthened by the performance of the TGSA and the GSA on appropriate examples below.

The TGSA solves some of the other problems in the GSA. The obvious benefit of using the third camera is that there are fewer regions in the base image that are totally occluded. That is, because of the third image there are fewer regions in the base image that do not appear in either of the other two images. However, there are also more regions in the base image that are occluded in one of the other images (we call these regions partially occluded). Because of this, the algorithm must be able to resolve matches using the information available only from matches in either the right or top images. This is accomplished by using the original GSA both vertically and horizontally, so that matches can be accepted without relying on trinocular support. One problem with finding matches in a partially occluded region is that many incorrect matches can receive strong support. For example, suppose a region in the base images appears in the top image but not the right. In this case all horizontal matches are incorrect. However, those matches whose disparities nearly agree with the vertical disparity receive strong support through the trinocular disparity gradient,

so some of them may be accepted. Fortunately, these will not affect the estimation of disparity in this region. Other horizontal matches, i.e. those with incorrect disparities, will tend to be suppressed.

The final problem with binocular matching involves significant appearance differences between the images. When these differences arise from random variations, the use of the additional image can improve the results of matching. The improvement obtained in overcoming stochastic noise through the use of many images in matching has been demonstrated by Tsai.[75] Specifically, in the TGSA the noise variations are overcome through the use of trinocular constraints. Trinocular uniqueness helps to suppress noise matches that occur when the correct match is missing for an edge since the support for such matches is usually relatively weak. Because the trinocular disparity gradient gathers support from orthogonal matches, it can strengthen matches whose support is eroded by noise variations between binocular image pairs. This orthogonal support may also be helpful in overcoming errors that are more systematic as well. However, it is difficult to make accurate claims about the effect of the algorithm on these systematic, structural variations. Testing on real image data is necessary to study this more carefully.

8.5. Experimental Results

The previous section argued that the Trinocular General Support Algorithm overcomes many of the problems in the binocular algorithm. In this section we present experiments demonstrating its success on synthetic images. The first group of images, discussed in Section 8.5.1, presents results on matching occluded, periodic

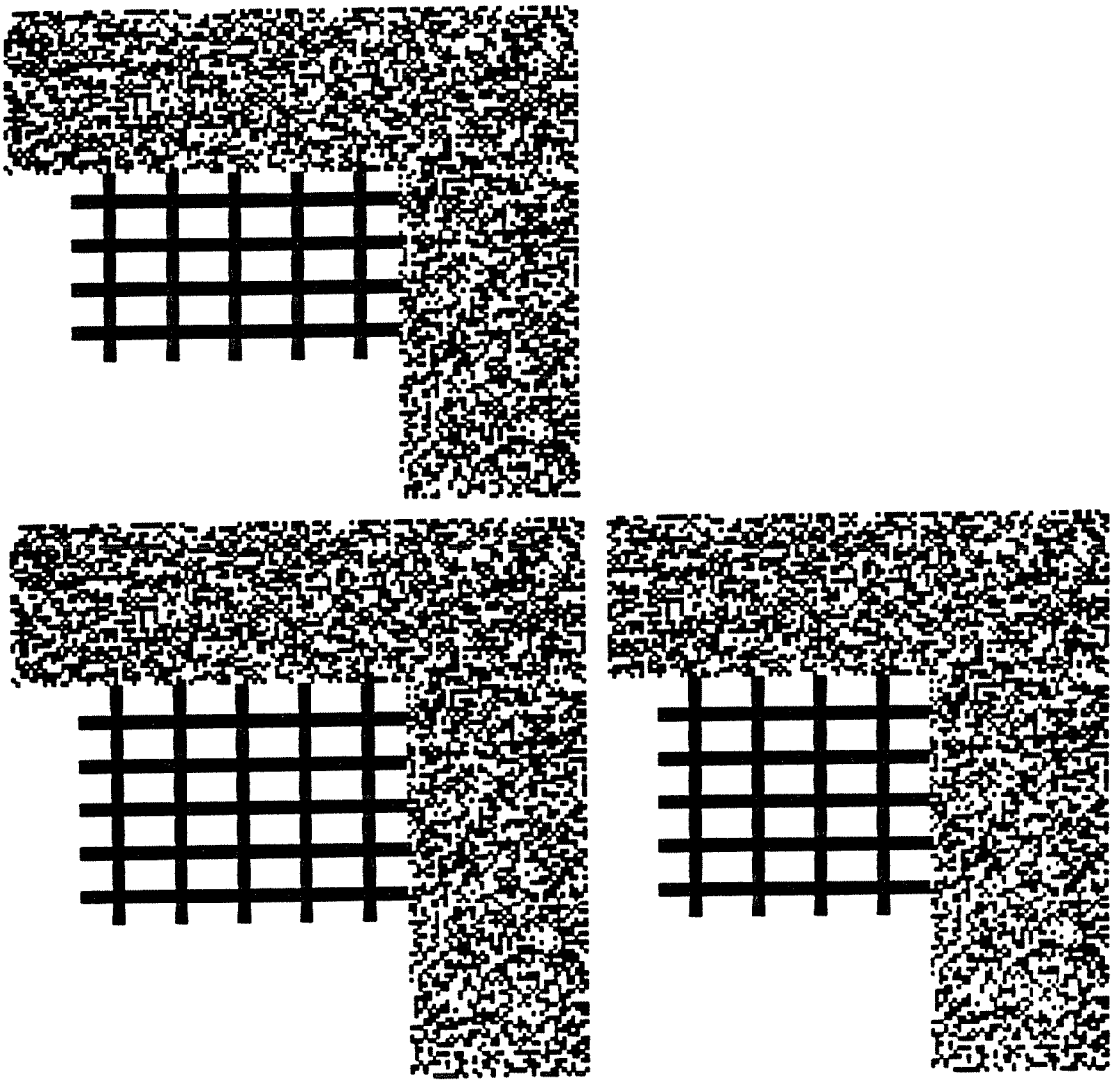


Figure 8.6. Top, base and right images for the first trinocular example. The random-dot texture is at disparity 16 and the bars are at disparity 4. The top bar is occluded in the top image. The right bar is occluded in the right image. See page 191 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

regions. The second group of images, discussed in Section 8.5.2, describes the algorithm's performance on random-dot stereograms involving occlusions and noisy

data.

8.5.1. Occluded Periodic Regions

Results on two different stereograms containing occluded, periodic regions are presented in this section. The periodic region in the images contains multiple vertical and horizontal bars. In the first example, the bars in the base image are occluded by a surface that is above and to the right of the region. In the second example, the occluding surface completely surrounds the matching bars. The occluding regions are textured with a random-dot pattern. Because the algorithm makes nearly 100% correct decisions in the occluding regions, the matching results are not presented numerically, only pictorially.

The base, top and right images for the first example are shown in Figure 8.6. Figure 8.7 gives the ground truth disparity for the base image encoded as intensity. (The white holes at the boundaries occur because those points do not appear in either the right image or the top image. This is because they are either shifted off the image by their large disparities, they are occluded, or both.) The occluding region, represented by the random-dot pattern, is at disparity 16, while the bars have disparity 4. Because of the occluding surface, the top horizontal bar does not appear in the top image, and the right vertical bar does not appear in the right image. The result is that five vertical and five horizontal bars appear in the base image, but only four horizontal bars appear in the top image, and only four vertical bars appear in the right image. The horizontal frequency of the bars is 14 pixels; the vertical frequency is 10.

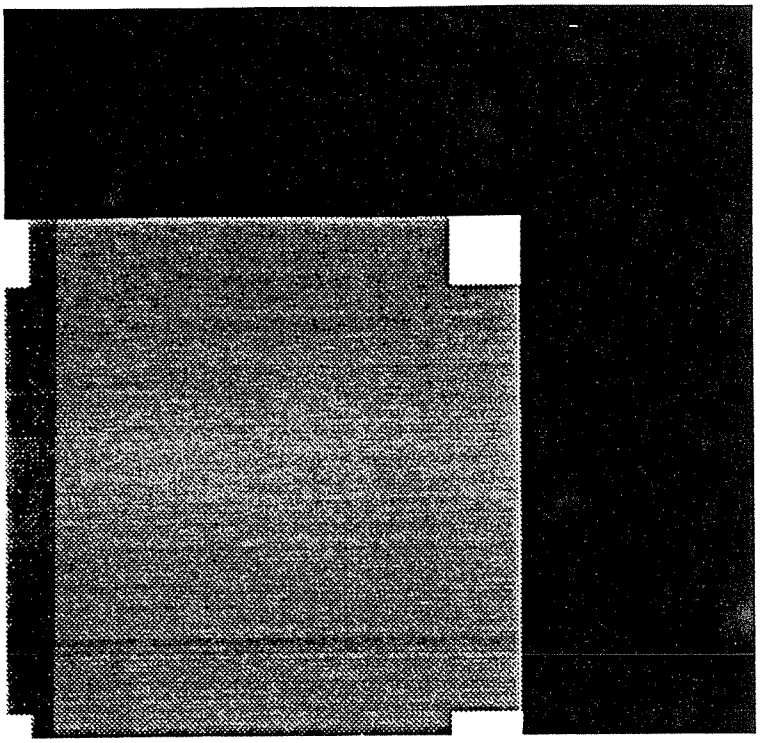


Figure 8.7. Ground truth disparity for the base image. The blank areas represent regions that appear in neither the right nor the top image. See page 191 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

The results of trinocular matching are shown in Figure 8.8 for the base image; the binocular matching results are shown in Figure 8.9. In binocular matching, the interior of the region is ambiguous, so the only way the algorithm could disambiguate the matches was to propagate the results from the horizontal boundaries of the images. The correct matches were found on the left boundary; the incorrect matches were found on the right. In addition, disparity gradient support from the occluding regions (disparity 16) favored the incorrect matches (disparity 18) near the top of the periodic region. Finally, note that in the binocular case only the vertical bars were matched and the random-dot texture was not matched on the left side of the base image (it did

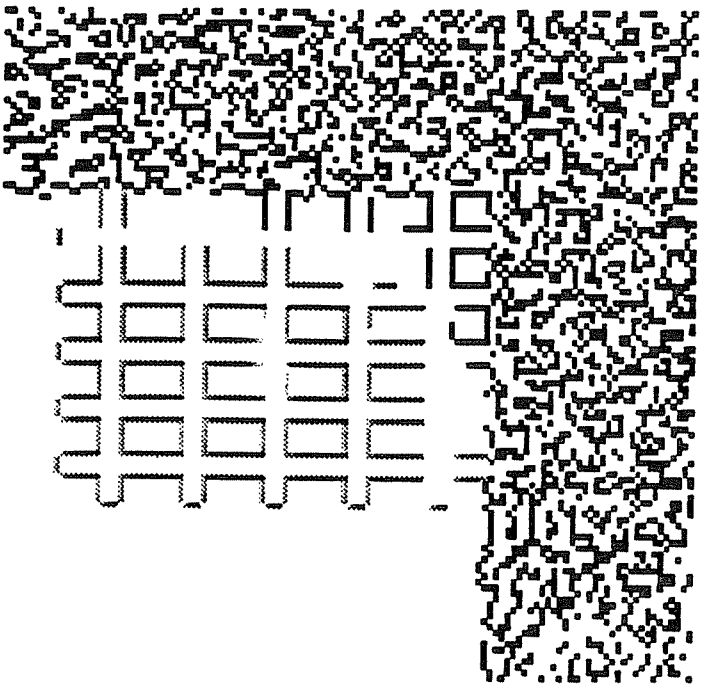


Figure 8.8. The base disparity image results for trinocular matching. See page 192 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

not appear in the right image), so the matches for the bars extended beyond this region.

The only errors in trinocular matching for the first example occurred near the upper-right corner of the periodic region. In this area the incorrect matches (at disparity 18 vertically, and 14 horizontally) received strong disparity gradient support from the textured surface (disparity 16). In addition, the incorrect matches at the right and top borders received less uniqueness inhibition. The combination of these two factors caused the errors. In the remaining part of the periodic region, the strength of the trinocular disparity gradient support for the correct matches resolved the

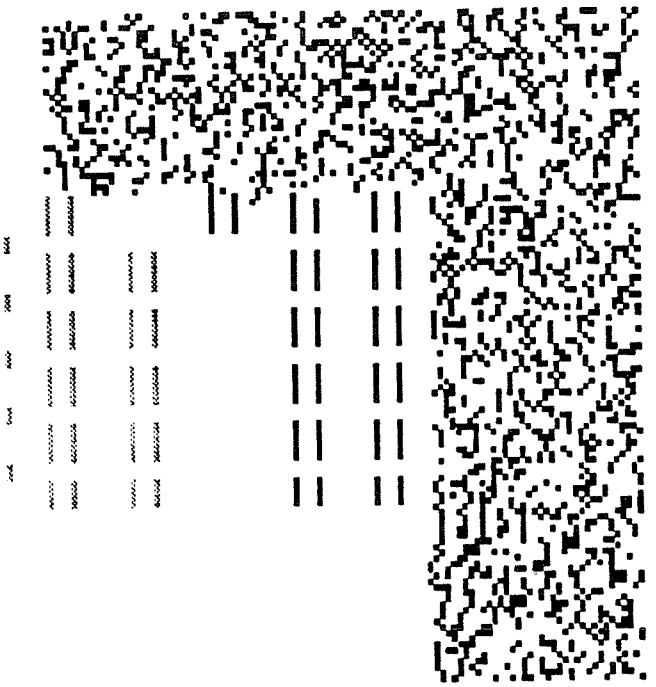


Figure 8.9. The base disparity image results for binocular matching. Note that the results on the left side of the random-dot texture did not extend as far as in the trinocular case. See page 191 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

ambiguity and allowed the correct matches to be accepted.

The images for the second periodic-image example are shown in Figure 8.10. The ground truth disparity for the base image is shown in Figure 8.11. The periodic region is at disparity 4; the occluding region is at disparity 16. Note that five vertical bars and five horizontal bars appear in each image. However, all of these bars should not be matched. The top bar and the right bar in the base image should not be matched, while the bottom bar in the top image, and the left bar in the right image should not be matched. The remaining bars should match to yield the correct

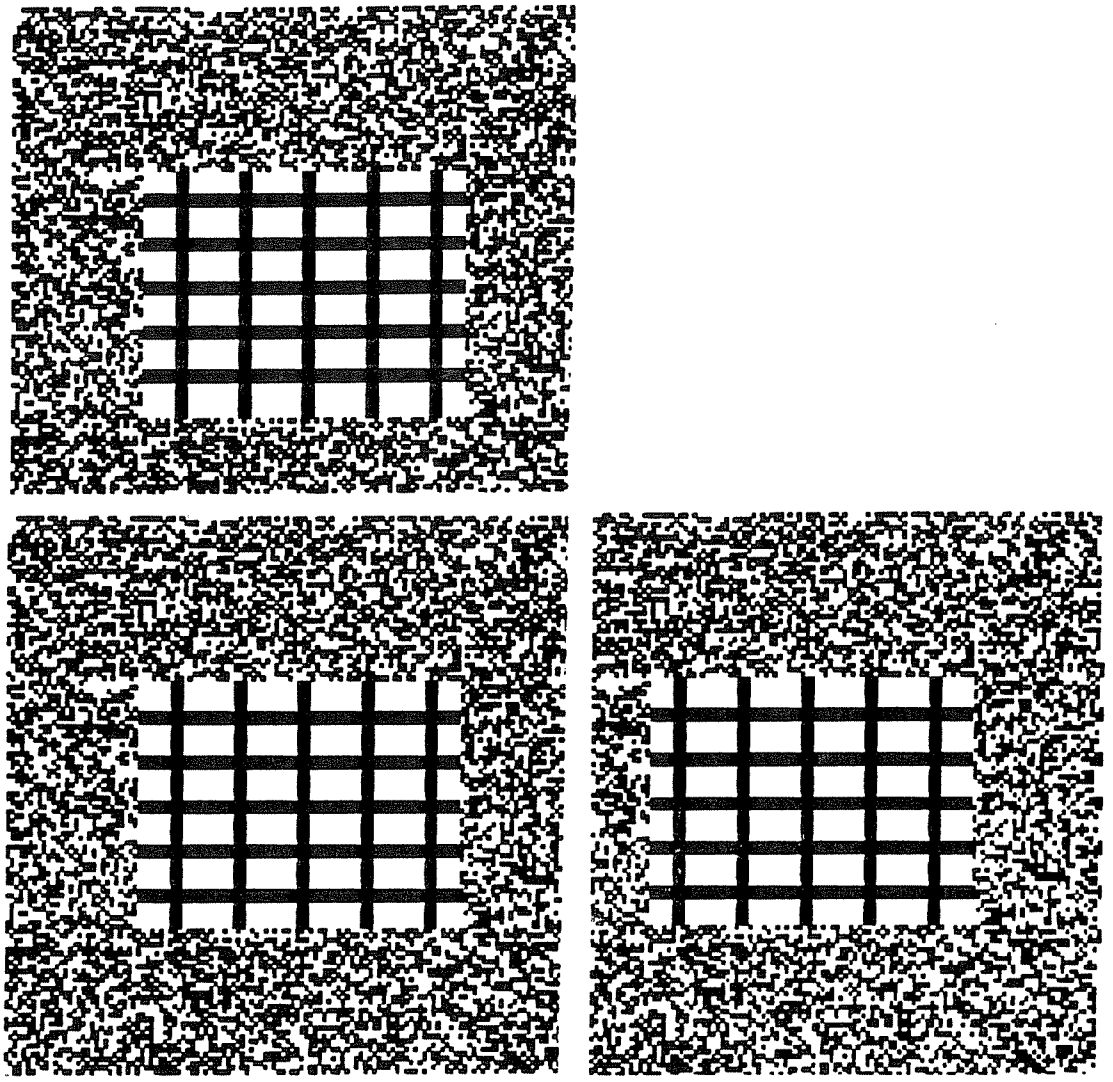


Figure 8.10. Top, base and right images for the second trinocular example. The random-dot texture is at disparity 16 and the bars are at disparity 4. Of the five vertical bars in the base image, the right one does not appear in the right image. Similarly, of the five horizontal bars in the base image, the top one does not appear in the top image. See page 195 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

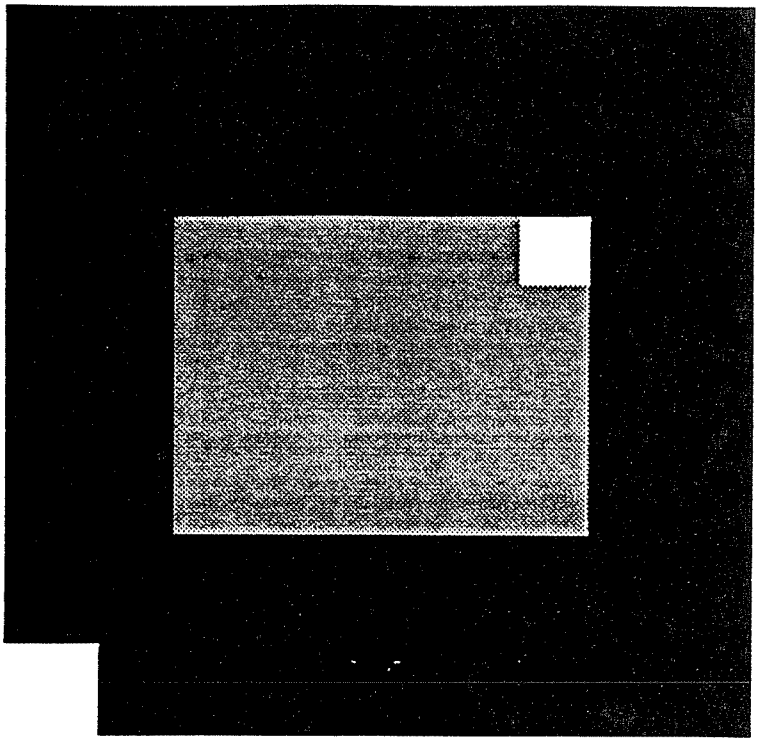


Figure 8.11. The correct disparities in the base image for the second trinocular example. The blank areas represent points that appear in neither the right nor the top images. See page 195 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

disparity of 4.

The results of matching using the trinocular algorithm are shown in Figure 8.12; the results using the binocular algorithm are shown in Figure 8.13. As expected, the matches for bars in the binocular case are *all incorrect*. There was no information in the center of the region to disambiguate the matches, so the matching results at the boundaries of the region were propagated inward. The disparities found at the boundaries of the region were incorrect. Thus, the disparity 18 matches were accepted for each edge in this region. In trinocular matching, the trinocular disparity gradient provided enough support for the correct matches in the center of the regions to

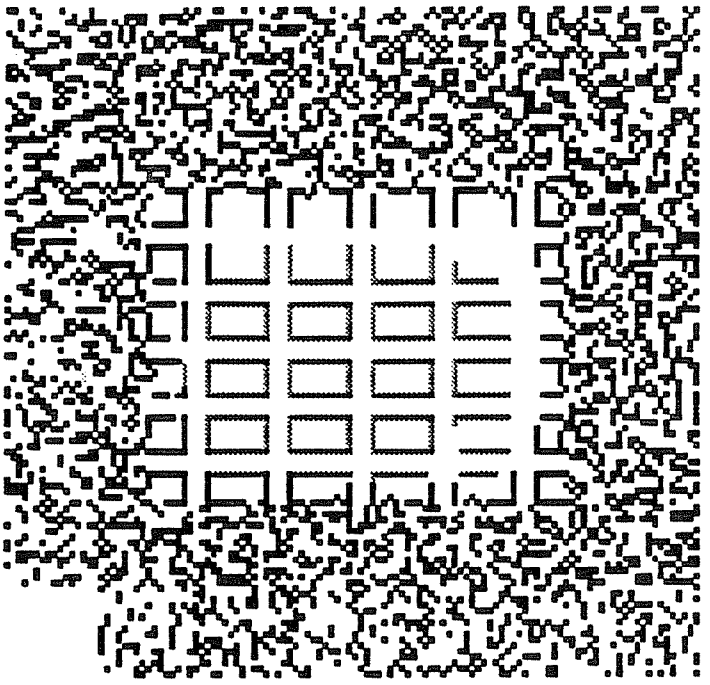


Figure 8.12. The disparity results of matching using the trinocular algorithm for the base image. See page 196 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

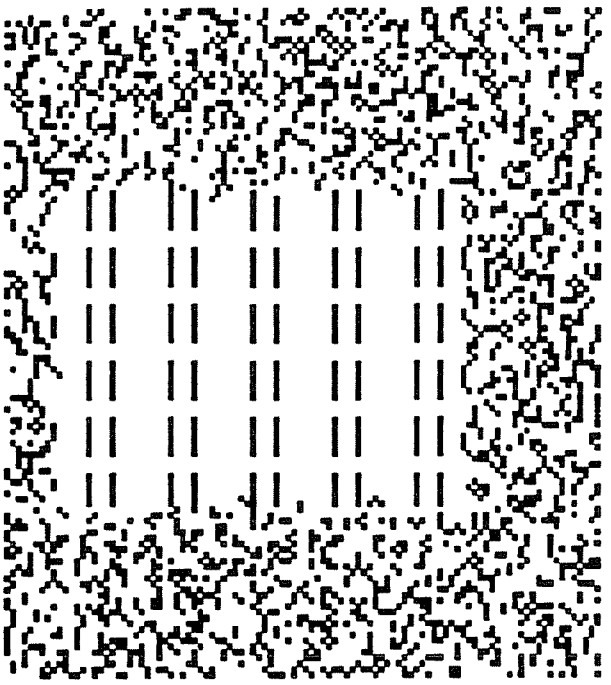


Figure 8.13. The disparity results of matching using the binocular algorithm for the base image. None of the disparities for the bars in the center of the image were correct. See page 195 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

overcome the ambiguity and resist the propagation of erroneous disparities from the boundaries. At the boundaries of the periodic region, matches with incorrect disparities were found because of the strength of uniqueness and the support through the disparity gradient from the occluding surface.

These results demonstrate the success of the TGSA in handling occluded, periodic regions. The algorithm did this by combining vertical and horizontal matching networks using additional constraints. Trinocular algorithms that only combine results as a post-matching process can not overcome these difficulties.[17,54] In matching a triple of images such as in Figure 8.10, these algorithms find incorrect

matches both vertically and horizontally. When the results are combined, the disparities disagree and none of them are correct. Algorithms relying on the third image for confirmation of a match will not work on this example either. The edges are all either vertically or horizontally oriented in the periodic region. Any horizontal match (between vertically-oriented edges) will be confirmed in the vertical image because it is on a line segment that is parallel to the vertical epipolar scanline. The reverse is true for vertical matches between horizontally-oriented edges. Thus, confirmation is found for each of the competing matches, so they can not be distinguished in this way.

8.5.2. Random-Dot Stereograms

In this section we present results for experiments in trinocular matching of random-dot stereograms. These examples are intended to demonstrate the trinocular algorithm's usefulness in handling problems due to noise and occlusions. The test cases include stereograms with 0%, 2% and 5% noise added to each image. Each test case is based on the disparity image shown in Figure 8.14. In this image there are three surfaces. The disparities of the surfaces are 0, 6, and 12 units.

The disparity images resulting from matching these images for both the trinocular and binocular algorithms are shown in Figures 8.16 - 8.21. The numerical results are summarized in Table 8.1. Figure 8.15 shows a graph of the percentage of nodes with intermediate activations as a function of the iteration number for the three trinocular algorithms and the binocular case of 2% noise.

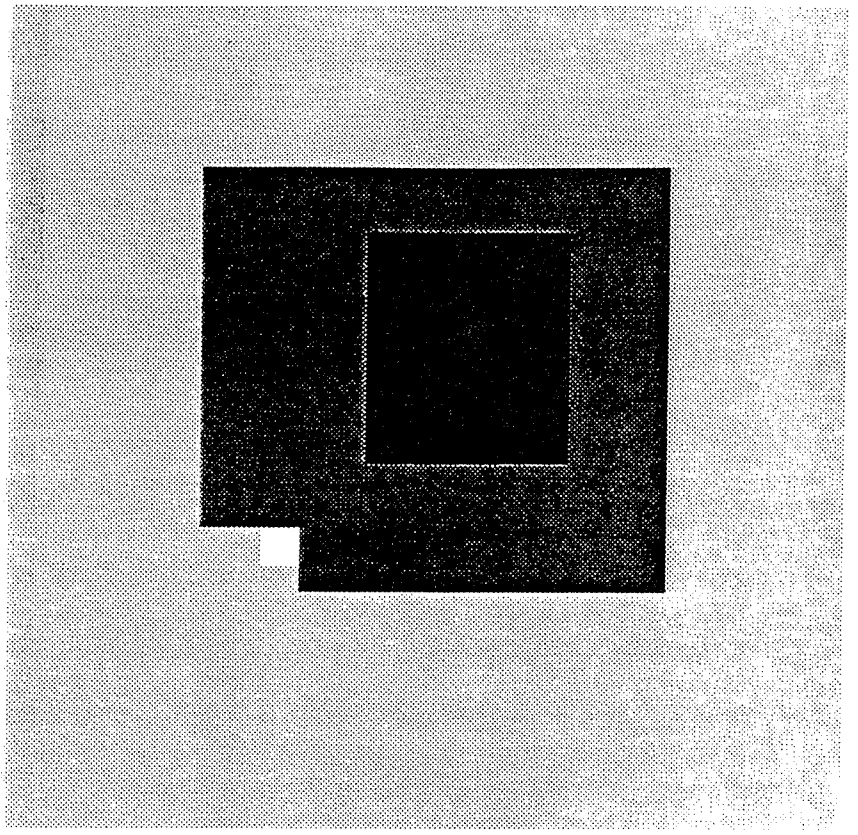


Figure 8.14. The base disparity image for the three random-dot stereogram examples. The disparities of the surface are 0, 6 and 12 units. See page 198 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

	Noise %	% Correct Decisions: Total	% Correct Decisions: No Occlusion	% Correct Decisions: Right Image	Number of Unmatched Edges
Trinoc	0	97.6	98.5	97.9	16
Binoc	0	97.9	98.3	97.4	1341
Trinoc	2	93.1	94.6	94.6	60
Binoc	2	93.2	93.4	93.2	1482
Trinoc	5	87.9	89.3	90.1	94
Binoc	5	85.0	84.6	85.7	1528

Table 8.1. Matching results for stereograms in Figures 8.16 - 8.21.

In Table 8.1 there are four results presented for each run. The first column shows the overall percentage of correct matching decisions. In trinocular matching, when both vertical and horizontal matches are involved, a correct matching decision for an edge is defined as either (1) both a vertical and horizontal correct match are accepted, (2) a correct match is accepted either vertically or horizontally and no incorrect matches are accepted, or (3) no matches are accepted when there are no correct matches available. All other decisions are counted as incorrect. The second column in the table shows the percentage of correct decisions when no occlusions are involved. For trinocular matching this implies that the region is not occluded in either the right or the top image. The third column shows the percentage of correct decisions from the perspective of the right image. The fourth column shows the number of edges which have no candidate matches. This demonstrates that a much denser set of matches is found when using the trinocular matching algorithm.

The second and third columns of Table 8.1 show that the trinocular algorithm produces a larger percentage of correct decisions in unoccluded regions, especially in the presence of noise. Unfortunately, the statistical measure for correct matches in the

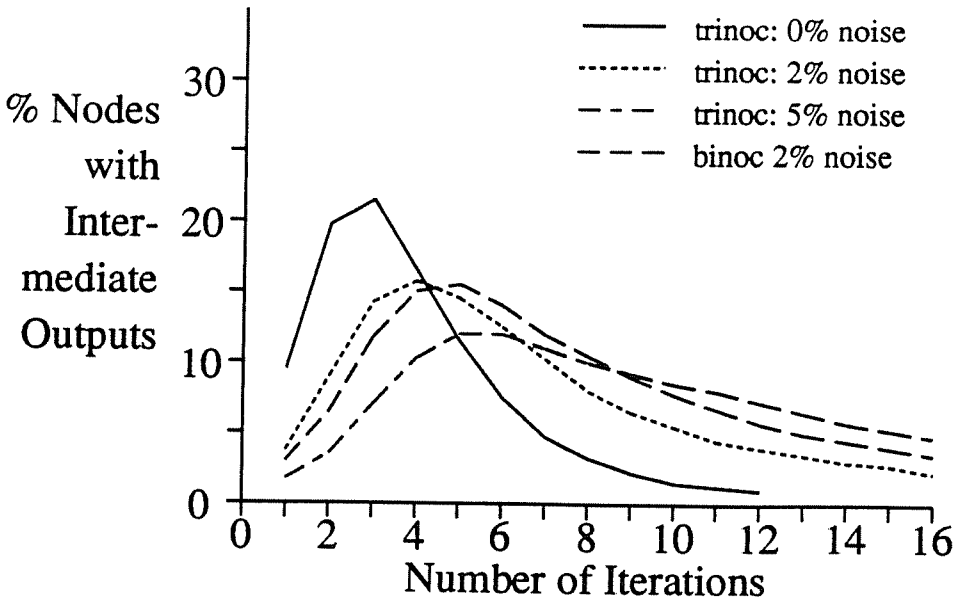


Figure 8.15. Graph showing the percentage of nodes with intermediate activations as a function of the iteration number. All three trinocular examples are shown, but only the 2% noise binocular example is given.

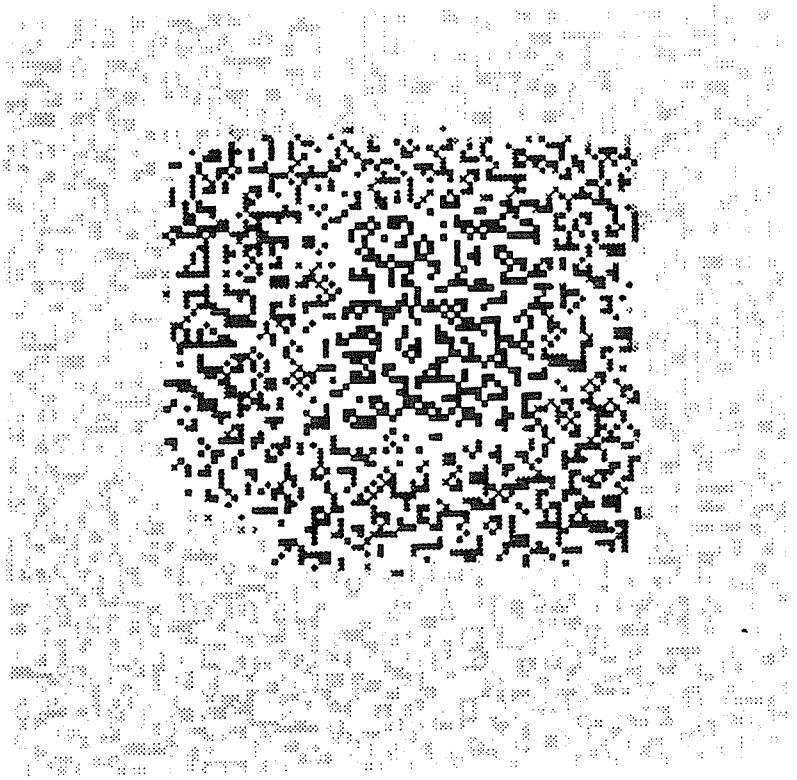


Figure 8.16. Results of trinocular matching for the base image with 0% noise in each image. See page 200 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

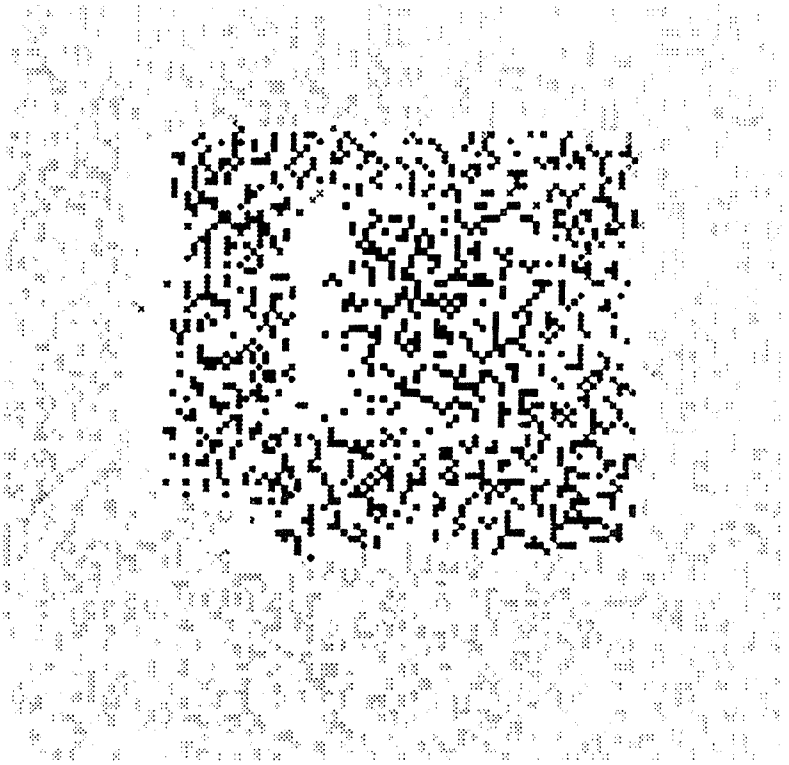


Figure 8.17. Results of binocular matching for the base image with 0% noise in each image. See page 200 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

entire image (column 1) is slightly lower for the trinocular case until the percentage of noise is large. The main reason for this is that the trinocular algorithm deals with regions that are only vertically or horizontally occluded. The algorithm finds a large number of correct matches in these regions. However, it also finds a number of incorrect matches. This is because these matches receive strong support through the trinocular disparity gradient that they do not receive in the binocular case. Note, however, that the erroneous matches receiving this support tend to have disparities that agree with the correct disparity for the region.

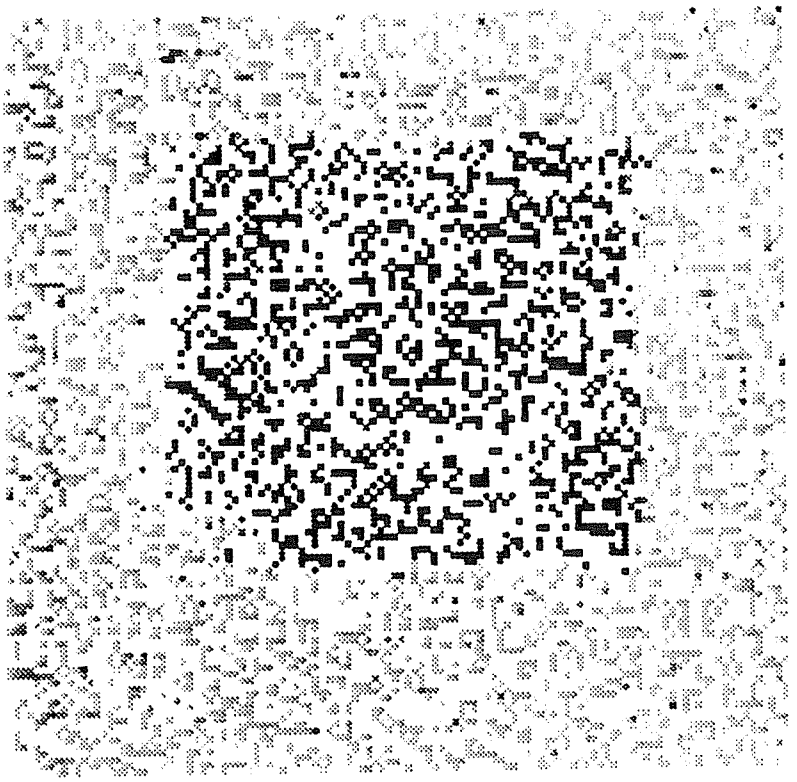


Figure 8.18. Results of trinocular matching for the base image with 2% noise in each image. See page 200 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

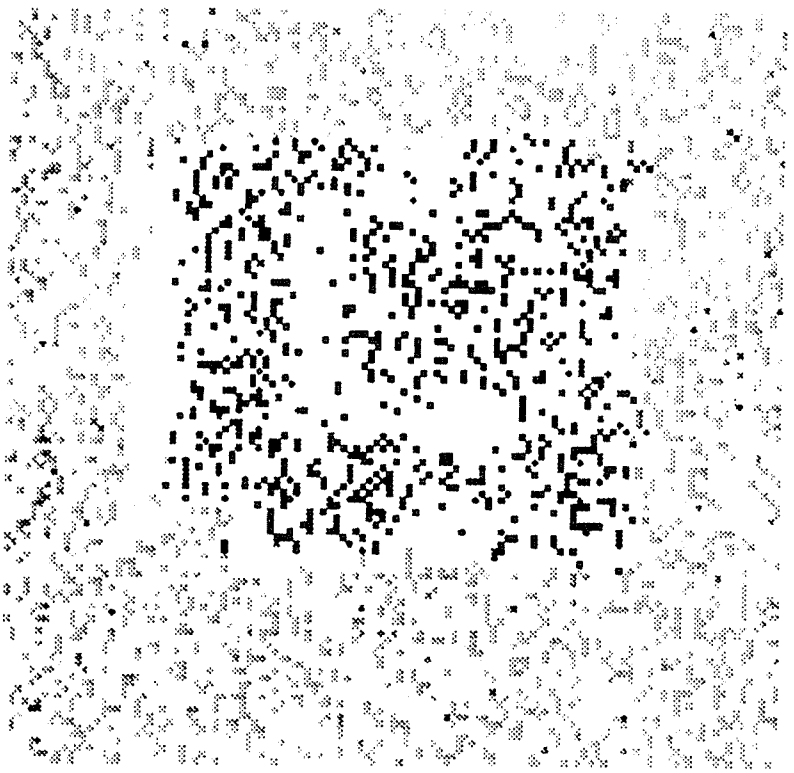


Figure 8.19. Results of binocular matching for the base image with 2% noise in each image. See page 200 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

The following numbers quantify the algorithm's performance in partially occluded regions (regions that appear in the base image and *one* of the other two images). The data are taken from the results of matching for the stereogram with 5% noise added to each image. First, consider matches between edges in the region and the image where they *did not* appear. Of the edges with candidate matches, the trinocular algorithm accepted no matches for 81% of them. Fortunately, of the 53 incorrect matches found, 36 were at the surface disparity. Thus, these incorrect matches do not give incorrect disparity measures. Next, consider matches between

edges in the partially occluded region and edges in the image where the region does appear. For edges with candidate matches, the trinocular made 89% correct matching decisions. Of the incorrect decisions, 11 were incorrect matches, and 24 were due to missing the correct match. Thus, although statistically the trinocular algorithm's performance in partially occluded regions was not great, the errors did not produce incorrect estimates of disparity.

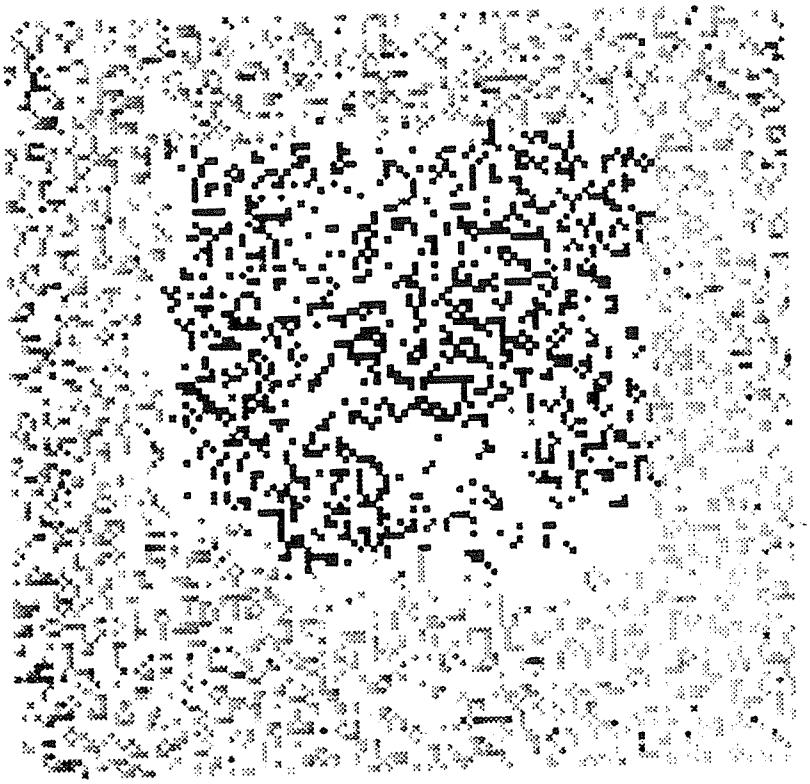


Figure 8.20. Results of trinocular matching for the base image with 5% noise in each image. See page 200 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

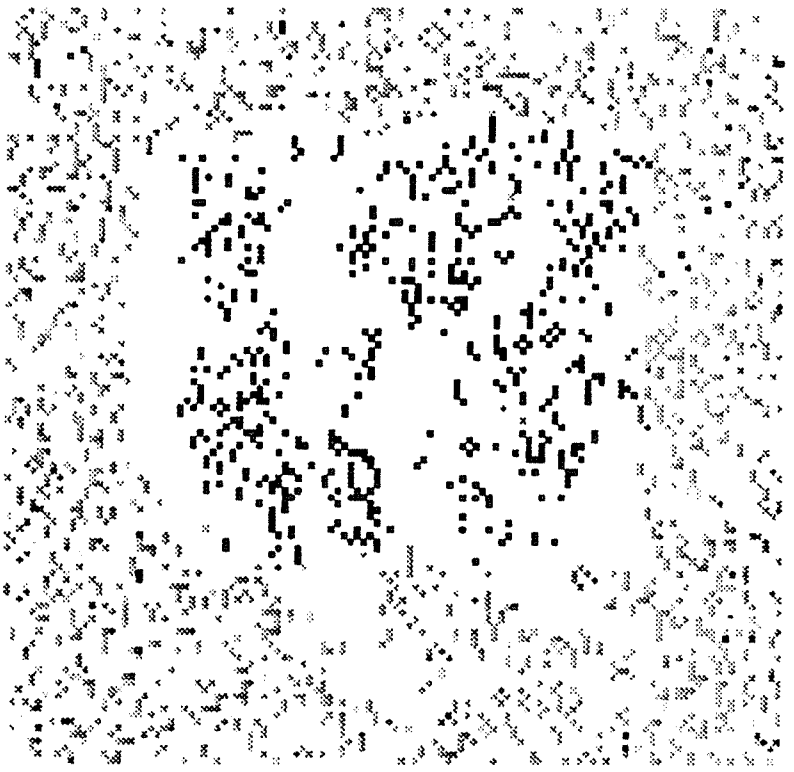


Figure 8.21. Results of binocular matching for the base image with 5% noise in each image. See page 200 for discussion of the results. (Some information in the images may be lost in copying this thesis.)

The results on the random-dot stereogram demonstrate that the trinocular algorithm accommodates noise in the images better than the binocular algorithm does. In addition, because points can match both vertically and horizontally, a denser set of matches found. Finally, the algorithm produces correct matches in occluded regions, although a number of erroneous matches are found as well.

8.6. Chapter Summary and Discussion

The Trinocular General Support Algorithm was successful in overcoming many of the remaining problems in stereo matching through the use of (1) the General Support Algorithm matching both vertically and horizontally, and (2) new constraints that defined relations between vertical and horizontal matches. The new constraints, particularly the trinocular disparity gradient, provided additional information that could resolve ambiguity in periodic regions. In addition, because the algorithm gathered support for a match both vertically and horizontally, the trinocular algorithm performed better in the presence of noise. Finally, the use of vertical and horizontal binocular matching implies that the algorithm did not rely on an edge being present in all three images in order for it to be matched.

Remaining problems include finding ways to improve the algorithm's performance in regions that are occluded either vertically or horizontally, but not both. It is also not clear how well the algorithm performs when noise variations between the images are not stochastic. The first problem may be handled through a post-matching smoothing process such as Ohta's trinocular algorithm.[54] It may also be possible to identify occlusions after matching has been completed and use only unoccluded matches. We have not addressed the second problem because only stochastic noise was used in our test images. Fully understanding this issue requires testing on real image data. This will determine how important a concern it is and how well our algorithm performs in overcoming it.

CHAPTER 9

Conclusions and Future Work

This thesis has studied the stereo matching problem using massively parallel computational models such as connectionist networks. The main part of the thesis was the development of the General Support Algorithm which included the following contributions:

- The analysis and reformulation of a number of important matching constraints including figural continuity, the disparity gradient, coarse-to-fine multiresolution, fine-to-coarse multiresolution, detailed match and uniqueness. The reformulated constraints were locally defined, that is, they were defined as interactions between pairs of candidate matches. The analysis of the constraints concluded that none of the constraints, used in addition to uniqueness, was sufficient to select the appropriate matches in all circumstances.
- The construction of the General Support Algorithm that integrated these constraints in parallel using a relaxation computation. In this computation the influence of the constraints was organized by the General Support Principle which stated that, except for uniqueness, the constraints only define positive relations between matches.
- The implementation of the General Support Algorithm in a connectionist network. This represents one of the few implementations of a connectionist

network for solving a real-world vision problem.

- The high percentage of correct matching decisions made by the resulting network. This included quality results on simple synthetic images, random-dot stereograms, and real images. For simple synthetic images, 100% correct matching decisions were made. For random-dot stereograms, 98-99% correct decisions were made on noise-free images, and the performance degraded gradually with added noise. Finally, for real images the algorithm ranged from 93-99% correct decisions, and averaged 97%.

In observing the matching results we were able to identify the underlying structural circumstances from which the errors arose. These included ambiguities in partially-occluded, periodic regions, erroneous matches for occluded edges, and matching errors due to significant structural differences between the images. Our analysis led to two different types of approaches for solving these problems. Both of these were added to the existing structure of the General Support Algorithm. These approaches were:

- The use of non-local mechanisms and results from other intrinsic image computations to explicitly overcome errors in matching. These include using disparity estimates obtained from focus to resolve ambiguity in periodic regions, and using average disparity detectors to locate occluding boundaries and occluded regions.
- The use of a third camera and additional trinocular constraints in matching. The General Support Algorithm was applied to two pairs of images and the new

constraints, trinocular uniqueness and the trinocular disparity gradient, related the matches between these pairs. This helped to resolve the ambiguity in periodic regions, eliminate noise errors, and provide a denser set of matches.

Finally, the General Support Algorithm was simulated in parallel on a Sequent Balance shared-memory multiprocessor. The simulation was executed in two phases. First, data structures were built representing the active match nodes and connections for a given pair of images. Second, these lists were used to execute the iterations of the network. By reducing the need for synchronization and mutual exclusion, near-linear speed-ups were obtained for the simulation running up to nine processors.

There are several areas for future work following from this thesis. Briefly they include the following;

- Further development of the mechanisms for eliminating more matching errors. These include implementing an occlusion detection mechanism, developing an algorithm for overcoming errors due to structural variations between images, and testing the trinocular algorithm on real images.
- Incorporation of the General Support Algorithm into an active vision system that observes its environment over time. Such a system would select a region of interest in the scene and focus both cameras on it. The disparities for image features from this area are relatively small and therefore precise disparity estimates can be found rapidly. Coarser disparity information may be obtained in the surrounding region. The result is a smaller and faster matching network.

However, such an approach requires additional algorithms for selecting areas of interest and maintaining depth results over a large area of the scene over time.

- Extension of ideas for intrinsic image combination. In overcoming ambiguity in periodic regions we used focus information as a coarse estimation of disparity. This favored the correct matches and allowed them to be accepted. In general, our approach to intrinsic image combination is to understand the weaknesses of a given computation and overcome these problems using appropriate information from other image sources.

APPENDIX 1

Detailed Match Implementation

This appendix describes the implementation of the subnetwork for determining detailed match support. Recall from Section 4.3 that there are two different types of support: (1) if both sides of the edges in a candidate match are similar, and (2) if one side of the edges if similar. Figure A.1 shows a network of seven additional nodes that implements this. The nodes of this network are simpler than the those of the matching network. Specifically, there is no uniqueness input and the outputs are binary. Thus, for a given node i, the activation is given by:

$$A_i = \theta_i + \sum_{j=1}^{N_i} O_j \, w_{ji}$$

The output is given by:

$$O_i = \begin{cases} 1, & A_i \geq \phi_i \\ 0, & A_i < \phi_i \end{cases}$$

The behavior of the nodes are dependent on the weighted input, the base activation, θ_i and the output threshold, ϕ_i.

In Figure A.1, node 1 outputs a value of 1 if $L_{low} \geq R_{low} + \phi_i$, and node 2 outputs 1 if $R_{low} \geq L_{low} + \phi_i$. Thus, $\theta_i = 0$, and ϕ_i is the threshold for intensity similarity of the low intensity sides of the edges. One of the two nodes will respond when there is a significant difference between the intensities. Nodes 3 and 4 are similar for the high side intensities. Node 5 has a base activation of $\theta_5 = 1$, and an output threshold of

$\phi_5 = 0.5$. Thus, it will output a 1 unless it receives blocking input from one of the lower level nodes. Its output signals the presence of intensity similarity on the low intensity side. Node 6 behaves similarly for the high intensity side of the edges. Finally, node 7 must respond with compensating input when both nodes 5 and 6 are active. Its base activation is $\theta_7 = 0$, but its output threshold is $\phi_i = 0.75$. The top level node in the network represents the match node.

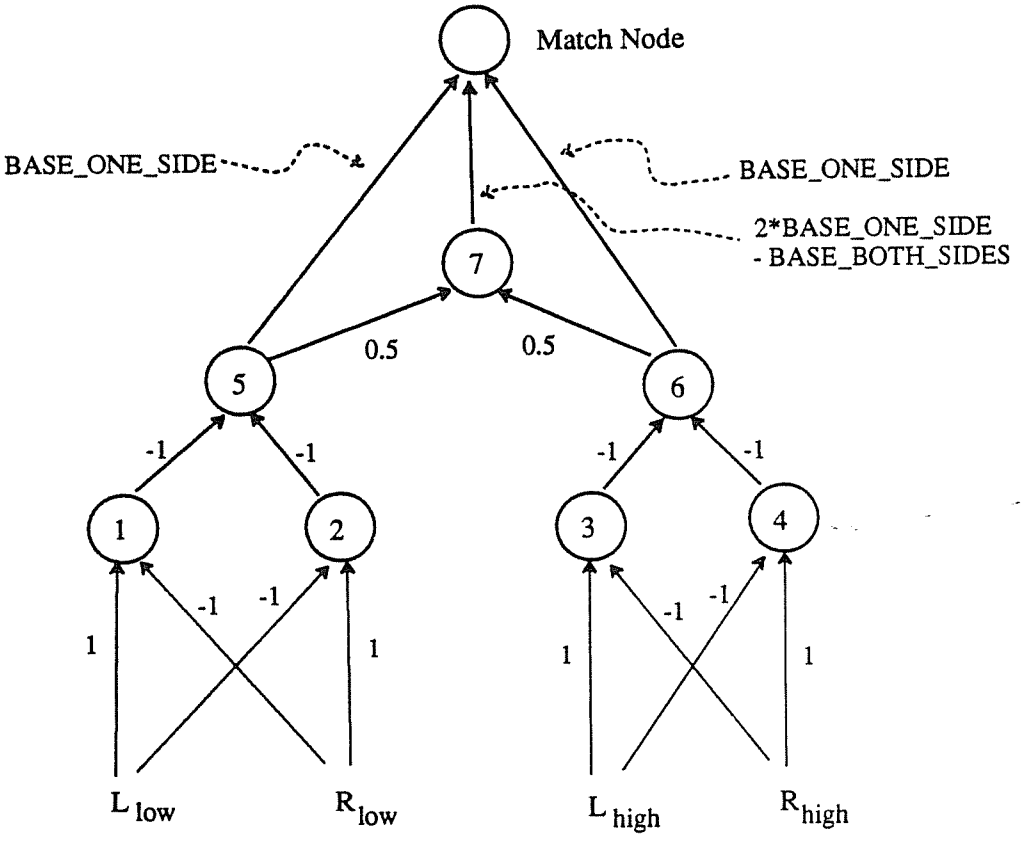


Figure A.1. Nodes that determine the detailed match constraint support.

This network design is fairly involved. There are seven additional nodes for each matching node. This is only practical if when the actual implementation of the network is in the reduced form described in Section 4.3.4.1. Similar mechanisms to the ones described in that section must be used to ensure that the values for the appropriate edges (those in the candidate match) are input to the network described here.

References

1. N. Ayache and B. Faverjon, "A fast stereovision matcher based on prediction and verification of hypotheses," *Workshop on Computer Vision: Representation and Control*, pp. 27-37 (1985).
2. N. Ayache and F. Lustman, "Fast and reliable passive trinocular stereovision," *Proc. First Int. Conf. on Computer Vision*, pp. 422-427 (1987).
3. R. Bajcsy, E. Krotkov, and M. Mintz, "Models of erros and mistakes in machine perception," *Proc. DARPA Image Understanding Workshop*, pp. 194-203 (1987).
4. H. Harlan Baker, T. O. Binford, J. Malik, and J-F. Meller, "Progress in stereo matching," *Proc. DARPA Image Understanding Workshop*, pp. 327-335 (1983).
5. H. H. Baker and T. O. Binford, "Depth from edge and intensity based stereo," *Proc. Seventh Int. Joint Conf. on Artificial Intelligence*, pp. 631-636 (1981).
6. D. H. Ballard, "Parameter nets," *Artificial Intelligence* **22** pp. 235-267 (1984).
7. S. T. Barnard and W. B. Thompson, "Disparity analysis of images," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **2** pp. 333-340 (1980).
8. S. T. Barnard and M. A. Fischler, "Computational Stereo," *Computing Surveys* **14** pp. 553-572 (1982).
9. P. Besl and R. Jain, "Range image understanding," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 430-449 (1985).
10. K.L. Boyer and A.C. Kak, "Symbolic stereo from structural descriptions," *Proc. Conf. on Artificial Intelligence Applications*, pp. 82-87 (1985).
11. J. Canny, "Computational approach to edge detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **8** pp. 679-698 (1986).
12. L. S. Davis and A. Rosenfeld, "Cooperating processes for low-level vision: A survey," *Artificial Intelligence* **17** pp. 245-263 (1981).
13. M. Drumheller and T. Poggio, "On parallel stereo," *Proc. IEEE Int. Conf. on Robotics and Automation* **3** pp. 1439-1448 (1986).
14. R. D. Eastman and A. M. Waxman, "Using disparity functionals for stereo correspondence and surface reconstruction," *Computer Vision, Graphics, and Image Processing* **39** pp. 73-101 (1987).
15. J. A. Feldman and D. H. Ballard, "Connectionist models and their properties," *Cognitive Science* **6** pp. 205-254 (1982).
16. J. A. Feldman, "Connectionist models and parallelism in high level vision," *Computer Vision, Graphics, and Image Processing* **31** pp. 178-200 (1985).
17. A. Gerhard, H. Platzer, J. Steurer, and R. Lenz, "Depth extraction by stereo triples and a fast correspondence estimation algorithm," *Proc. Eighth Int. Conf. on Pattern Recognition*, pp. 512-515 (1986).

18. A. Goshtasby, "A refined technique for stereo depth perception," *Workshop on Computer Vision: Representation and Control*, pp. 125-129 (1984).

19. W.E.L. Grimson, "Computational experiments with a feature based stereo algorithm," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **7** pp. 17-34 (January, 1985).

20. S. Grossberg, "The quantized geometry of visual space: the coherent computation of depth, form, and lightness," *The Behavioral and Brain Sciences* **6** pp. 625-692 (1983).

21. S. Grossberg, "Competitive learning: From interactive activation to adaptive resonance," *Cognitive Science* **11** pp. 23-64 (1987).

22. S. Grossberg and E. Mingolla, "Neural dynamics of surface perception: boundary webs, illuminants, and shape-from-shading," *Computer Vision, Graphics, and Image-Processing* **37** pp. 116-165 (1987).

23. M. J. Hannah, "SRI's baseline stereo system," *Proc. DARPA Image Understanding Workshop*, pp. 149-155 (December, 1985).

24. M. Herman, T. Kanade, and S. Kuroe, "Incremental acquisition of a three-dimensional scene model from images," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **6** pp. 331-340 (1984).

25. E. C. Hildreth, "Edge detection," MIT AI Laboratory A.I. Memo 858 (September, 1985).

26. G. E. Hinton, T. J. Sejnowski, and D. H. Ackley, "Boltzmann machines: Constraint satisfaction networks that learn," Technical Report CMU-CS-84-119 (1984).

27. G. E. Hinton, "Distributed representations," Technical Report CMU-CS-84-157 (1984).

28. G. E. Hinton and T. J. Sejnowski, "Learning and relearning in Boltzmann Machines," pp. 282-317 in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, ed. D. E. Rumelhart and J. L. McClelland,The MIT Press, Cambridge, MA (1986).

29. W. Hoff and N. Ahuja, "Surfaces from stereo," *Proc. DARPA Image Understanding Workshop*, pp. 98-106 (December, 1985).

30. W. Hoff and N. Ahuja, "Extracting surfaces from stereo images: an integrated approach," *Proc. First Int. Conf. Computer Vision*, pp. 284-294 (1987).

31. J. J. Hopfield and D. W. Tank, "Computing with neural circuits: A model," *Science* **233** pp. 625-633 (August 8, 1986).

32. R. A. Hummel and S. W. Zucker, "On the foundations of relaxation labeling processes," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **5**(3) pp. 267-287 (1983).

33. M. Ito and A. Ishii, "Range and shape measurement using three-view stereo analysis," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp.

9-14 (1986).

34. M. Jenkin and J. K. Tsotsos, "Applying temporal constraints to the dynamic stereo problem," *Computer Vision, Graphics, and Image Processing* **33** pp. 16-32 (1986).
35. B. Julesz, *Foundations of Cyclopean Perception*, University of Chicago Press, (1971).
36. T. Kanade, "Recovery of the three-dimensional shape of an object from a single view," *Artificial Intelligence* **17** pp. 409-460 (1981).
37. M. Kass, "Computing visual correspondence," *Proc. DARPA Image Understanding Workshop*, pp. 54-60 (1983).
38. M. Kass, "Linear image features in stereopsis," *Proc. Fifth National Conf. on Artificial Intelligence*, pp. 707-713 (1986).
39. P. K. Kienker, T. J. Sejnowski, G. E. Hinton, and L. E. Schumacher, "Separating figure from ground with a parallel network," *Perception* **15** pp. 197-216 (1986).
40. Y. C. Kim and J.K. Aggarwal, "Finding range from stereo images," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 289-294 (1985).
41. H. S. Lim and T. O. Binford, "Stereo correspondence: features and constraints," *Proc. DARPA Image Understanding Workshop*, pp. 373-380 (December, 1985).
42. F.J. L. Liu, R. Eastman, and L. S. Davis, "Experiments in stereo matching using multiresolution local support," Center for Automation Research, University of Maryland CAR-TR-183 (January, 1986).
43. D. Marr and T. Poggio, "A computational theory of human stereo vision," *Proc. Royal Society of London, B.* **204** pp. 301-328 (1979).
44. D. Marr and E. Hildreth, "Theory of edge detection," *Proc. Royal Society of London B.* **207** pp. 187-217 (1980).
45. D. Marr, *Vision*, W.H. Freeman and Company, New York (1982).
46. J. Marroquin, S. Mitter, and T. Poggio, "Probabilistic solutions of ill-posed problems in computational vision," *Proc. DARPA Image Understanding Workshop*, pp. 293-309 (1985).
47. J. E.W. Mayhew and J. P. Frisby, "Psychophysical and computation studies towards a theory of human stereopsis," *Artificial Intelligence* **17** pp. 349-385 (1981).
48. J. L. McClelland and D. E. Rumelhart, "Distributed memory and the representation of general and specific information," *Journal of Experimental Psychology: General* **114**(2) pp. 159-188 (1985).
49. G. Medioni and R. Nevatia, "Segment-based stereo matching," *Computer Vision, Graphics and Image Processing* **31** pp. 2-18 (1985).

50. G. G. Medioni and R. Nevatia, "Segment-based stereo matching," *Proc. DARPA Image Understanding Workshop*, pp. 128-136 (1983).

51. V. J. Milenkovic and T. Kanade, "Trinocular vision using photometric and edge orientation constraints," *Proc. DARPA Image Understanding Workshop*, pp. 163-175 (December, 1985).

52. V. S. Nalwa and T. O. Binford, "On detecting edges," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **8** pp. 699-714 (1986).

53. Y. Ohta and T. Kanade, "Stereo by intra- and inter-scanline search using dynamic programming," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **7** pp. 139-154 (1985).

54. Y. Ohta, M. Watanabe, and K. Ikeda, "Improving depth map by right-angled trinocular stereo," *Proc. Eighth Int. Conf. on Pattern Recognition*, pp. 519-522 (1986).

55. Y. Ohta, K. Takano, and K. Ikeda, "A highspeed stereo matching system based on dynamic programming," *Proc. First Int. Conf. on Computer Vision*, pp. 335-342 (1987).

56. A. P. Pentland, "A new sense for depth of field," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **9** pp. 523-531 (1987).

57. M. Pietikainen and D. Harwood, "Depth from three camera stereo," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 2-8 (1986).

58. S. B. Pollard, J. E.W. Mayhew, and J. P. Frisby, "PMF: a stereo correspondence algorithm using a disparity gradient limit," *Perception* **14** pp. 449-470 (1985).

59. K. Prazdny, "Detection of binocular disparities," *Biological Cybernetics* **52** pp. 93-99 (1985).

60. L. H. Quam, "Hierarchical warp stereo," *Proc. DARPA Image Understanding Workshop*, pp. 149-155 (1984).

61. G.V.S. Raju, T.O Binford, and S. Shekhar, "Stereo matching using viterbi algorithms," *Proc. DARPA Image Understanding Workshop*, pp. 766-776 (1987).

62. D. E. Rumelhart, G. E. Hinton, and J. L. McClelland, "A general framework for parallel distributed processing," pp. 45-76 in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, ed. D. E. Rumelhart and J. L. McClelland,The MIT Press, Cambridge, MA (1986).

63. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," pp. 318-362 in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, ed. D. E. Rumelhart and J. L. McClelland,The MIT Press, Cambridge, MA (1986).

64. D. E. Rumelhart and D. Zipser, "Feature discovery by competitive learning," pp. 151-193 in *Parallel Distributed Processing: Explorations in the*

Microstructure of Cognition, Volume 1: Foundations, ed. D. E. Rumelhart and J. L. McClelland,The MIT Press, Cambridge, MA (1986).

65. D. Sabbah, "Computing with connections in visual recognition of origami objects," *Cognitive Science* **9** pp. 25-50 (1985).
66. G. B. Smith, "Stereo reconstruction of scene depth," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 271-276 (1985).
67. D. L. Smitley and R. Bajcsy, "Stereo processing of aerial, urban images," *Proc. Seventh IEEE Int. Conf. on Pattern Recognition* **1** pp. 433-435 (1984).
68. R. Srinivasan, K.R. Ramakrishnan, and P.S. Sastryy, "A contour based stereo algorithm," *Proc. First Int. Conf. on Computer Vision*, pp. 677-681 (1987).
69. R. Szeliski and G. Hinton, "Solving random-dot stereograms using the heat equation," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 284-288 (1985).
70. R. Szeliski, "Cooperative algorithms for solving random-dot stereograms," CMU-CS-86-133 (1986).
71. D. Terzopoulos, "Image analysis using multigrid relaxation," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **8** pp. 129-139 (1986).
72. D. Terzopoulos, "Regularization of inverse visual problems involving discontinuities," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **8** pp. 413-424 (1986).
73. H. P. Trivedi, "A computational theory of stereo vision," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 277-282 (1985).
74. H. P. Trivedi and S. A. Lloyd, "The role of disparity gradient in stereo vision," *Perception* **14** pp. 685-690 (1985).
75. R. Y. Tsai, "Multifram image point matching and 3-D surface reconstruction," *IEEE Trans. on Pattern Analysis and Machine Intelligence* 5 pp. 159-173 (1983).
76. A. Waxman and J. H. Duncan, "Binocular image flows," *Proc. IEEE Workshop on Motion: Representation and Analysis*, pp. 31-38 (1986).
77. A. M. Waxman and S. S. Sinha, "Dynamic stereo: passive ranging to moving objects from relative image flows," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **8** pp. 406-412 (1986).
78. A. M. Waxman and J. H. Duncan, "Binocular image flows: steps toward stereo-motion fusion," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **8** pp. 715-729 (1986).
79. L. R. Williams, "Spectral continuity and eye vergence movements," *Proc. Ninth Int. Joint Conf. on Artificial Intelligence*, pp. 985-987 (1985).
80. A. P. Witkin, "Scale-space filtering," *Proc. Eighth Int. Joint Conf. on Artificial Intelligence*, pp. 1019-1022 (1983).

81. G. Xu, S. Tsuji, and M. Asada, "Coarse-to-fine control strategy for matching motion stereo pairs," *Proc. Ninth Int. Joint Conf. on Artificial Intelligence*, pp. 892-894 (1985).

82. M. Yachida, Y. Kitamura, and M. Kimachi, "Trinocular vision: a new approach for correspondence problem," *Proc. Eighth Int. Conf. on Pattern Recognition*, pp. 1041-1044 (1986).