USC-SIPI REPORT NO. 146

Artificial Neural Network Algorithms for Some Computer Vision Problems

By

Yitong Zhou

June 1989

Acknowledgements

I would like to thank Prof. Rama Chellappa, my advisor and the co-chairman of my dissertation committee, for his enthusiastic guidance, constructive criticism and moral support throughout the course of this work. I would also like to thank Prof. George A. Bekey, who served as the co-chairman of my dissertation committee, for his advice, support, and help during the course of my graduate studies and Prof. Louis Gordon for serving as a member of my dissertation committee.

I am indebted to Prof. Alexander Sawchuk, former Director of SIPI, for providing an enriching research environment; Prof. Keith Jenkins and Prof. Richard Leahy for helpful discussions and suggestions.

I would like to thank the graduate students at SIPI, especially Richard Hansen, Bob Frankot, Ted Broida, Shankar Chaterjee, V. Venkateswar, Min Shao, Qinfen Zheng, Gam-Sun Young, Tal Simchony, Anand Rangarajan, B. Manjunath, Aseem Vaid, Xiaofeng Zhao, Chen-Hsun Wang, Kung-Shiuh Huang, Xiaohong Yan for their friendship and wide-ranging and animated discussions.

I am very grateful to Dr. Allan Weber for his help in computer support and to Mr. Ray Schmidt for photographic services. Thanks also to everyone else in SIPI, especially Jerene Brooks, Linda Varilla and Gloria Bullock for their assistance.

The natural motion and stereo images used in this work were provided by Mr. Andres Huertas of the Institute of Robotics and Intelligent Systems at USC. His assistance is greatly appreciated.

Finally and especially, I thank my parents and my wife for their love, patience and support during my time at USC.

This work was funded in part by the AFOSR Contract No. F-49620-87-C-0007 and the AFOSR Grant No. 86-0196.

Contents

List of Figures

List of Tables

Abstract

The problem considered here involves the use of an artificial neural network for solving some computer vision problems such as static and motion stereo, computation of optical flow, and image restoration. The network used in this research contains massive mutually interconnected and self-connected binary neurons. Two decision rules, deterministic and stochastic, are used. The stochastic decision rule guarantees convergence to a global minimum but computationally is very expensive, while the deterministic decision rule greatly reduces computing time, but only gives a local minimum.

Two basic methods, static and motion stereo, for extracting 3-D information from more than one image are considered. The static stereo method is based on images taken by two cameras separated by a known baseline. The motion stereo method infers depth information from a sequence of monocular images. The derivatives of the intensity function are used for matching. A window operator which functions very similar to the human eye in detecting the intensity changes is proposed for estimating the derivatives. Under the epipolar, photometric and smoothness constraints the neural network is employed for the matching procedure. For motion stereo, two algorithms, batch and recursive, which allow the use of arbitrarily many image frames are presented. No surface interpolation step is involved in the algorithms because of the dense derivatives used.

An algorithm using rotation invariant primitives extracted from successive monocular images is presented for computing optical flow. Under local rigidity assumption and a smoothness constraint, the neural network is used to compute optical flow. To locate motion discontinuities, the information about occluding elements is utilized by embedding it into the bias inputs of the network. A batch solution is also developed for the case of pure translation.

An approach for the restoration of gray level images degraded by a known shift invariant blur function and additive noise is developed. The neural network is employed to represent a possibly nonstationary image whose gray level function is the simple sum of the neuron state variables. The nonlinear restoration method is carried out iteratively by using a dynamic algorithm to minimize the energy function of the network. Owing to the model's fault-tolerant nature and computational capability, a high quality image is obtained using this approach. A practical algorithm with reduced computational complexity is also presented. The choice of the boundary values to reduce the ringing effect is discussed and comparisons to other restoration methods such as the SVD pseudoinverse filter, minimum mean square error (MMSE) filter and modified MMSE filter using Gaussian Markov random field model are given. A schematic diagram of optical implementation of the restoration algorithm is described.

To demonstrate the efficacy of all these algorithms, experimental results using both synthetic and natural images are presented.

Chapter 1

Introduction

This dissertation is concerned with developing algorithms for some computer vision problems, especially at the low-level using an artificial neural network. The task of low-level vision is to recover physical properties of visible 3-D surfaces from 2-D images. One module of low-level vision, for instance, extracts depth information from two eyes–a pair of binocular images, or from one eye over a period of time–a sequence of monocular images. Low-level processes also provide motion information about brightness patterns over a sequence of 2-D images, the so called optical flow, for motion detection and representation. Certainly human brain is very good at performing low-level vision tasks but today's computers are not. This is because of massive amount of two dimensional array data that needs to be analyzed and the lack of learning or self organizing capabilities of most modern day computers. From a mathematical point of view, low-level vision problems are ill-posed in the sense of Hadamard [1, 2]. An efficient method for solving ill-posed problem using artificial networks is the Tikhonov regularization technique. The idea of regularization technique is to narrow the admissible solution region by introducing suitable *a priori* knowledge or stablize the solution by means of some auxiliary non-negative functional [2]. Cognitive scientists believe that brain uses cooperative computation to deal with the task of low-level vision [3]. Cooperative computation takes into account multiple and often mutually conflicting constraints or many pieces of information simultaneously to narrow the solution region. This is accomplished by employing massively parallel processing units, with information transfer taking place through the interconnections between these units, in the form of excitatory and inhibitory signals. This cooperative computation provides not only massive parallelism, but also a greater degree of robustness because of local connectivity among processing units and adaptability of interconnection strengths (weights). Attempting to provide a human-like performance, many neural network models have been developed based on stúdies in biological nervous systems for cooperative computation. The neural network models seem well suited for mimicking low-level vision processes and hence an attractive way for solving the low-level computer vision problems. The neural network models have greatest potential in low-level vision area where highly parallel computations are required and the current best computer systems are inferior to human performance.

In this research, a discrete artificial neural network is used to solve some computer vision problems such as static stereo, motion stereo, computation of optical flow and image restoration. This network containing massive mutually interconnected and self-connected binary neurons is very similar to Amari [4, 5] and Hopfield networks [6]. Since self-feedback may cause the energy function of the network to increase with a transition, two decision rules, deterministic and stochastic, are used to ensure convergence. The stochastic decision rule guarantees that the network will converge to a global minimum but is computational intensive. The deterministic decision rule greatly reduces computational time but gives only a local minimum, an approximate solution. Two types of activation functions, threshold function (step function) and maximum evolution function, are used in the updating scheme.

Usually, the measurement primitives used for stereo matching are the intensity values, edges and linear features. Conventional methods based on such primitives suffer from amplitude bias, edge sparsity and noise distortion, whereas human stereo process does not. Knowing that human visual system is very sensitive to the intensity changes, the derivatives of the intensity function which are more reliable, dense and robust are used for matching in this study. For estimating the derivatives, a window operator is suggested using a combination of smoothing and differentiation. However, as the natural stereo images are digitized, the resulting spatial quantization error affects the intensity function and the derivatives. The effects of noise and spatial quantization on the estimation of derivatives are discussed, leading to an appropriate choice of window size.

Static and motion stereo are the two basic methods for inferring depth information using more than one frame. The static stereo is based on binocular images with a known baseline. Under epipolar, photometric and smoothness assumptions the neural network is employed to implement the matching procedure based on the first order derivatives of intensity. Although many researchers have been using neural networks for stereo matching [7, 8, 9, 10, 11, 12], these algorithms rarely work on natural images. Experimental results using both synthetic and natural stereo images show that the approach presented in this dissertation successfully recovers the depth information. The robustness of this approach is tested using noisy image pairs. The motion stereo method infers depth information from a sequence of monocular image sequence. For motion stereo, both batch and recursive neural network formulations are presented. Existing recursive approaches usually use either a Kalman filter or recursive least square algorithm [12] to update the disparity values. Due to the unmeasurable estimation error, the estimated disparity values at each recursion are unreliable, yielding a noisy disparity field. Instead, our approach recursively updates the bias inputs of the network, the measurement primitives. The disparity field is then computed by using the neural network based static matching algorithm. Since the recursive algorithm implements the matching algorithm only once, and the bias input updating scheme can be accomplished in real time, a vision system employing such an algorithm is feasible. A detection algorithm for locating occluding pixels is also included. No surface interpolation and smooth procedures are required in all algorithms. One sequence of natural images is considered in the experiments.

Optical flow is the distribution of apparent velocities of motion brightness patterns in an image. Motion can be caused by moving objects and/or a moving camera. The problem considered here is how to calculate the optical flow from two or more image frames. Starting with conventional methods, an algorithm is proposed for computing optical flow from two image frames based on principle curvatures, a set of rotation invariant measurement primitives. Under local rigidity assumption and a smoothness constraint, the neural network is then used to compute the optical flow. A difficult problem in computing optical flow is to locate motion discontinuities. A line process is commonly used [13] for detecting discontinuities. However, the detected discontinuities may be shifted due to occluding region, because the optical flow at the occluding region is undetermined. In order to locate the discontinuities accurately, we first detect the occluding elements based on the initial motion measurements and then embed this information into the bias inputs of the network. Computer simulations on synthetic and natural images are presented. When motion is pure translation with constant velocity, optical flow can be estimated from a sequence of images. A batch solution is discussed for this problem and successfully tested on a sequence of natural images with a stationary background.

Restoration of a high quality image from a degraded recording is an important problem in early vision processing. An approach for restoration of gray level images degraded by a known shift invariant blur function and additive noise is developed. The neural network is employed to represent a possibly nonstationary image whose gray level function is the simple sum of the neuron state variables. The nonlinear restoration method is carried out iteratively by using a dynamic algorithm to minimize the energy function of the network. Owing to the model's fault-tolerant nature and computational capability, a high quality image is obtained using this approach. A practical algorithm with reduced computational complexity is also presented. Several computer simulation examples involving synthetic and real images are given to illustrate the usefulness of our method. The choice of the boundary values to reduce the ringing effect is discussed and comparisons with other restoration methods such as the SVD pseudoinverse filter, minimum mean square error (MMSE) filter and the modified MMSE filter using Gaussian Markov random field model are given. An optical implementation of this approach is also described.

1.1 Thesis Organization

Chapter 2 reviews the relevant artificial neural networks and gives a modified neural network, which is used in this research. To ensure convergence, two decision rules, deterministic and stochastical decision rules, are presented.

In Chapter 3, static stereo research is reviewed and a neural network based algorithm using epipolar, photometric and smoothness constraints is discussed. A window operator for estimation of the first derivatives of intensity functions is derived. An analysis of the effects of noise and spatial quantization on the estimation of the derivatives is given. Results from synthetical and natural images are presented.

Chapter 4 addresses the motion stereo problem where a monocular motion image sequence is used instead of binocular images. After briefly reviewing the existing literature, two approaches, known as the batch and recursive approaches, are proposed. Detection of occluding pixels based on the matching error is also discussed. One sequence of real motion images is used to illustrate the performances of these approaches.

Chapter 5 deals with optical flow. Existing methods for computing optical flow are discussed. A method for finding the principle curvatures based on the second order derivatives estimated with subpixel accuracy is suggested. A neural network algorithm for computing optical flow from two successive image frames is developed. Extension to the case of multiple sequential images is discussed. A method for detecting motion discontinuities is derived, and experimental results using synthetic and natural images are presented.

Chapter 6 presents an approach for image restoration. By using a simple sum number representation, a dynamic algorithm using the neural network is developed. A practical algorithm with reduced computational complexity is also presented. Discussions on the choice of boundary values and comparisons to other restoration methods are given. Several computer simulation examples are presented to show the usefulness of this method. An optical implementation of this approach is presented.

Conclusions and future research are presented in Chapter 7.

1.2 Contributions

The primary contributions of this dissertation are as follows:

1. A new approach to static stereo is developed [14, 15, 16]. The first derivatives of the intensity function are used as the measurement primitives. Guidelines for the choice of the window size used in estimating the derivatives are given by analyzing the effects of noise and spatial quantization on the estimation of the derivatives.

2. Two new approaches, batch and recursive approaches, for motion stereo are developed. A method for detecting occluding pixels based on matching error is also derived.

3. A neural network based approach for computing optical flow is introduced [17, 18]. Principle curvatures which are rotation invariant measurement primitives are used in the matching procedure. For estimating the principle

curvatures, a 2-D window operator is designed. For pure translational motion, a batch algorithm is also introduced. A method for detecting motion discontinuities is derived.

4. A new method for the restoration of gray level images degraded by a known shift invariant blur function and additive noise is presented [19, 20, 21]. A practical algorithm with reduced computational complexity is also presented. An optical implementation of the restoration algorithm is also given.

Chapter 2

Computational Neural Networks

2.1 Introduction

Research on neural network modeling has a long history. Neurobiologists have discovered individual nerve cells existing in the brain and learned how neurons carry information, transmit information and receive stimuli. Based on the understanding of the nervous systems, many neural networks have been proposed by researchers. Over the past fifty years, hundreds of papers have been published in this area. As early as 1943, McCulloch and Pitts [22] developed a neural network by treating neurons as Boolean devices and showed that such a network could compute. Since then, learning has become the main focus in this area. In 1949, Hebb [23] proposed a learning rule which was first tested in the Edmonds and Minsky's learning machine-a simulated network, still used today in many learning paradigms. In the 50's, Rosenblatt [24, 25] invented a class of simple neuron learning networks in order to realize a dynamic, interactive and self-organizing system.

Meanwhile, Selfridge [26] developed a dynamic, interactive network for computational tasks in perception. Widrow and Hoff developed an adaptive network with a delta learning rule for pattern recognition [27]. Anderson [28] provided brain state in a box model which contains a feed-back loop for learning. Recently, Amari [29, 30], Arbib [31, 30] Malsburg [32], Fukushima [33] and Grossberg [34] have developed several competitive learning models. Also Kohonen [35], Feldman and Ballard [36], Rumelhart and McClelland [37], Hopfield[38, 39, 6], Sejnowski [40], and others have made important contributions to neural network modeling.

Since Hopfield and Tank [6] showed that a certain class of optimization problem can be programmed and solved on their neural network, the computational power of neural network has become more and more apparent. The neural network used in this research is based a great to extent on the Amari recurrent network and the Hopfield discrete network.

2.2 Amari and Hopfield Networks

In early 70s, Amari [4, 5] proposed two self-organizing random networks: a non-recurrent network for association and a recurrent network for concept formation. The recurrent network, shown in Figure 2.1(a), is a sequential network containing n bistable elements (neuron pools) $\{v_1(t), v_2(t), ..., v_n(t)\}$. Each element, shown in Figure 2.1(b), consists of mutually connected neurons. The outputs of the network are connected to its inputs. The bistable element can be in one of two states: $v_i(t) = 1$ (firing state) and $v_i(t) = 0$ (resting state). Each element $v_i(\cdot)$ receives weighted input signals from all the elements at time t. After summing the weighted inputs and comparing with a threshold h_i the state of the element

$v_i(\cdot)$ at time $t+1$ is determined by

$$v_i(t+1) = g(\sum_{j=1}^{n} T_{i,j}\, v_j(t) - h_i) \tag{2.1}$$

where $T_{i,j}$ is the weight (synaptic interconnection strength) from element j to element i and $g(x)$ is a step function defined by

$$g(x) = \begin{cases} 1 & if \;\; x \geq 0 \\ 0 & if \;\; x < 0. \end{cases} \tag{2.2}$$

The stable states are reached if

$$v_i(t+1) = v_i(t) \qquad for \quad i = 1, 2, ..., n.$$

The pattern specified by the stable states of the elements is the network output. Since the weight $T_{i,i}$ is nonzero, all the elements of the network have feedback. The recurrent network was actually derived from the McCulloch-Pitts formal neuron-the simplest form of neural network [22].

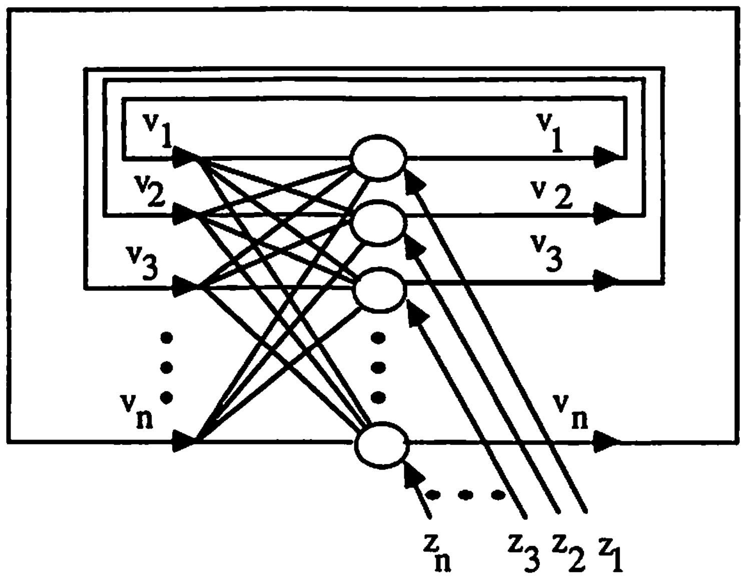
The Hopfield discrete network, shown in Figure 2.2, uses two state threshold neurons $\{v_1(t),\, v_2(t), ...,\, v_n(t)\}$. Each neuron receives external input I_i and weighted inputs from other neurons at random times. The total input of neuron i is

$$\sum_{j \neq i} T_{i,j}\, v_i(t) + I_i.$$

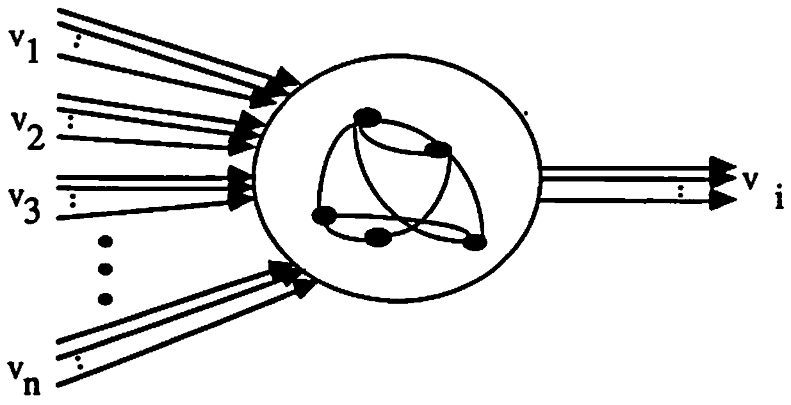
The state of the neuron i is asynchronously updated according to a threshold rule with threshold h_i

$$v_i = g(\sum_{j=1}^{n} T_{i,j}\, v_j(t) + I_i - h_i) \tag{2.3}$$

where $g(x)$ is in (2.2). The asynchronous property is introduced to represent a combination of propagation delay, jitter and noise in real systems. To ensure that



(a) Recurrent network.



(b) Bistable element (neuron pool).

Figure 2.1: Amari recurrent network.

the network converges to stable states, two conditions, symmetric interconnections ($T_{i,j} = T_{j,i}$) and no self-feedback ($T_{i,i} = 0$), have to be satisfied [39]. If we consider the bistable elements of the Amari network as the binary neurons, one can see that the Hopfield network is similar to the Amari network. The Hopfield network is originally designed for association. However, Hopfield and Tank [6] used an analog version of this network to solve a difficult but well defined optimization problem-the Traveling Salesman problem.

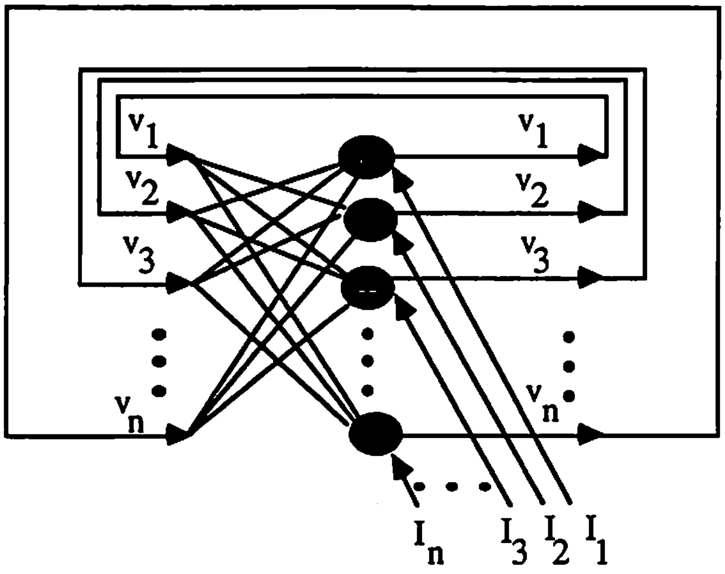


Figure 2.2: Hopfield discrete network.

2.3 A Discrete Neural Network for Vision

Since we are dealing with vision problems, even if each pixel in a 2-D image is represented by just one neuron, a massive 2-D neural network is required. For instance, for a 256 × 256 image, a total 65536 neurons are needed. If m neurons are used at each pixel, then 65536 × m neurons are needed. For simplicity of analysis, a 1-D version of such a discrete neural network is shown in Figure 2.3. More details about the 2-D network will be given in subsequent chapters.

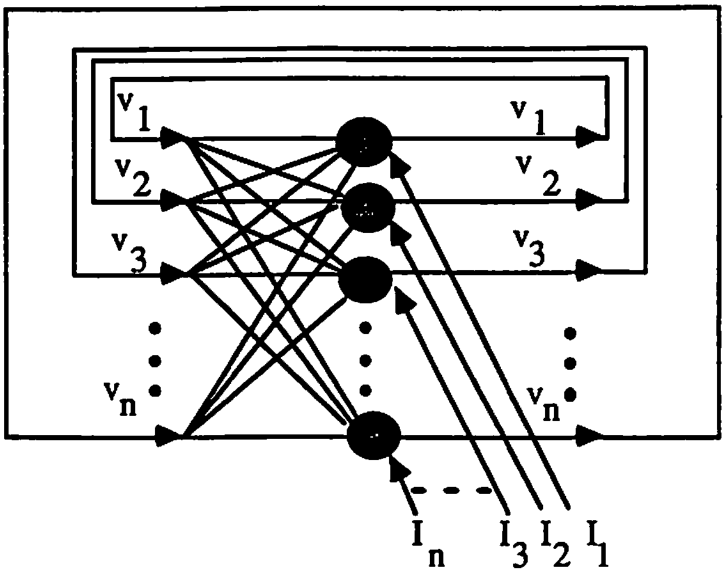


Figure 2.3: A discrete neural network.

2.3.1 A Discrete Network

The network consists of n mutually interconnected binary neurons $\{v_1, v_2, ..., v_n\}$. The neurons take on the values 0 for resting and 1 for firing. Let $T_{i,j}$ denote the strength (possibly negative) of the interconnection between neuron i and neuron j. The interconnections are symmetric

$$T_{i,j} = T_{j,i} \quad for \quad 1 \le i, j \le n$$

and the self-connection is nonzero

$$T_{i,i} \neq 0$$

which requires self-feedback for each neuron. In this network, each neuron (i, k) synchronously, or randomly and asynchronously receives inputs $\sum T_{i,j} v_j$ from all neurons and a bias input I_i

$$u_i = \sum_{j}^{n} T_{i,j} v_j + I_i \tag{2.4}$$

Each neuron u_i is fed back to corresponding neurons after either thresholding or maximum evolution

$$v_i = g(u_i). \tag{2.5}$$

where $g(x_i)$ is an activation function whose form is taken either as (2.2) for thresholding or

$$g(x_i) = \begin{cases} 1 & if \;\; x_i = max(x_k; \forall k \in \Omega_l). \\ 0 & otherwise. \end{cases} \tag{2.6}$$

for maximum evolution, where Ω_l's are disjoint subsets of index set $\Omega = \{1, 2, \ldots, n\}$ and $\bigcup \Omega_l = \Omega$, and $i \in \Omega_l$. The synchronous updating scheme uses the information about the old states of all neurons. The asynchronous updating scheme uses the latest information about other neurons, which means that any state change in a neuron will immediately affect the state of all the neurons.

2.3.2 Decision Rules

As mentioned above, this network has self-feedback, i.e. $T_{i,i} \neq 0$. As a result of having feedback, this network does not always converge to stable states. This can be explained as follows. According to [39], by setting thresholds $\{h_i\}$ to zero the energy function of the network can be defined as

$$E = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} T_{i,j}\, v_i\, v_j - \sum_{i=1}^{n} I_i\, v_i \tag{2.7}$$

Let changes in state and energy be denoted as

$$\Delta v_i = v_i^{new} - v_i^{old}$$

and

$$\Delta E = E^{new} - E^{old}.$$

Case 1: Step function.

When a step function is used as an activation function, the energy change ΔE due to a state change Δv_i of neuron i is given by

$$\Delta E = -(\sum_{j=1}^{n} T_{i,j}\ v_j + I_i\ )\ \Delta v_i - \frac{1}{2}\ T_{i,i}\ (\Delta v_i)^2 \tag{2.8}$$

Case 2: Maximum evolution function.

When a maximum evolution function is used, a batch of m neurons $\{v_k;\ k \in \Omega_l\}$ is simultaneously updated at each step. At most two of the m neurons change their state at each step. The energy change ΔE due to the state changes of neuron i and neuron i' is given by

$$\begin{aligned} \Delta E \ = \ & -(\sum_{j=1}^{n} T_{i,j}\ v_j + I_i\ )\ \Delta v_i - \frac{1}{2}\ T_{i,i}\ (\Delta v_i)^2 \\ & -(\sum_{j=1}^{n} T_{i',j}\ v_j + I_{i'}\ )\ \Delta v_{i'} - \frac{1}{2}\ T_{i',i'}\ (\Delta v_{i'})^2 \\ & -T_{i,i'}(\Delta v_i\ v_{i'}^{new} + \Delta v_{i'}\ v_i^{new}) \end{aligned} \tag{2.9}$$

In both cases, the energy changes are not always negative, i.e. $\Delta E > 0$, which means that the energy function does not decrease monotonically with a transition. Therefore, E is not a Lyapunov function and the network is unstable. Consequently, the convergence of the network is not guaranteed [41].

To ensure convergence of the network to a minimum, one can design some decision rules for updating the states of neurons. Depending on whether convergence to a local minimum or a global minimum is desired, a deterministic or stochastic decision rule can be used, respectively. In some cases, for example when the energy function is convex, the deterministic decision rule will ensure the network to converge to a global minimum.

Deterministic Decision Rule:

The deterministic rule is to take a new state v_i^{new} of neuron i if the energy change ΔE due to state change Δv_i is less than zero. If ΔE due to state change is > 0, no state change is affected.

Stochastic Decision Rule:

A stochastic rule is similar to the one used in simulated annealing techniques [42, 43, 44]. Define a Boltzmann distribution by

$$\frac{p_{new}}{p_{old}} = e^{\frac{-\Delta E}{T}}$$

where p_{new} and p_{old} are the probabilities of the new and old global state respectively, ΔE is the energy change and T is the parameter which acts like temperature. A new state v_i^{new} is taken if

$$\frac{p_{new}}{p_{old}} > 1, \qquad or\ if\ \frac{p_{new}}{p_{old}} \leq 1\ \ but\ \frac{p_{new}}{p_{old}} > \xi$$

where ξ is a random number uniformly distributed in the interval [0,1].

2.4 Discussion

In this chapter, we have presented a discrete artificial neural network. Owing to the self-feedback, two decision rules have been suggested to ensure convergence of the network. Comparing this network with the Amari network and Hopfield network, some differences listed in Table2.1 can be noted. The major difference

Network	Neuron	Self-feedback	Decision rule	Activation function
Amari	Binary, pool	Yes	No	Step
Hopfield	Binary	No	No	Step
Ours	Binary	Yes	Deterministic, Stochastic	Step, Maximum evolution

Table 2.1: Comparisons to the Amari network and Hopfield network.

among them is that the network used in this dissertation has self-feedback and hence needs a decision rule to ensure convergence. Self-feedback arises naturally in the problems considered in this dissertation. For the Traveling Salesman problem [6] based on simulations we have found that our network needs less than 100 iterations, while the Hopfield network needs more than 1000 iterations. It appears that using the deterministic decision rule in a network with self-feedback results in fewer iterations than is required by a network without self-feedback.

Chapter 3

Static Stereo

3.1 Introduction

Static stereo is a primary means for recovering 3–D depth from two images taken from different viewpoints. The two central problems in stereo matching are (1) extract and match corresponding feature points or lines, and (2) obtain a depth map or disparity values between these points. In this chapter, we present a method for computing the disparities between the corresponding points in the two images recorded simultaneously from a pair of laterally displaced cameras based on the first order intensity derivatives.

Basically, there exist two types of stereo matching methods: region based and feature based methods according to the nature of the measured primitives. The region based methods use the intensity values as the measurement primitives. A correlation technique or some simple modification is applied to certain local region around the pixel to evaluate the quality of matching. The region based methods usually suffer from the problems due to lack of local structures in homogeneous regions, amplitude bias between the images and noise distortion. Recently, Barnard [45] has applied a stochastic optimization approach for the stereo matching problem to overcome the difficulties due to homogeneous regions and noise distortion. Although this approach is different from the conventional region based methods, it still uses intensity values as the primitives with the aid of a smoothness constraint. Barnard's approach has several advantages: simple, suitable for parallel implementation and a dense disparity map output. However, too many iterations, a common problem with the simulated annealing algorithm, makes it unattractive. It also suffers from the problems of amplitude bias between the two images and oversmoothing.

The feature based methods use intensity edges or linear features (for example, see Grimson [46] and Medioni [47]) or intensity peaks which correspond to discontinuities in the first order derivatives of intensity [48]. The intensity edges are obtained using edge detectors such as the Marr–Hildreth edge detector [49] or the Nevatia–Babu line finder [50]. Since amplitude bias and small amounts of noise do not affect edge detection, feature based methods can handle natural images more efficiently. Owing to fewer measurement primitives to deal with, feature based methods are usually faster than the region based methods. However, a surface interpolation step has to be included. In order to obtain a smooth surface, several types of smoothness constraint techniques have been introduced [46]. The common problem in feature based methods is that if features are sparse, then surface interpolation step is difficult. Among the feature based methods, the Marr-Poggio-Grimson algorithm gives impressive results. But it is difficult to ensure continuity of disparity over an area of the image. To overcome this problem, Grimson [51] proposed a new algorithm including the figural continuity constraint [48] and other modifications. The figural continuity constraint is superior to the region continuity constraint. However, an occluding boundary or a sloping surface may cause problems. Another interesting approach is the integrated approach, which combines matching, contour detection and surface interpolation steps [52]. The integrated approach uses only piecewise smoothness assumption. A number of stereo images were tried in [52] to illustrate the performance of this approach. Some problems of this approach reported by the authors are misplacement and missing of contours, and disparity errors due to inaccuracies in edge detection.

Julesz's example of random dot stereograms shows that stereo matching occurs very early in the visual process and is relatively independent of other forms of visual processing [53]. Early stereo process implies that more dense measurement primitives are used in matching. It seems that the region based methods are closer to the human stereo process than the edge based methods, because the intensity values are dense measurement primitives. However, region based methods suffer from the problems of amplitude bias and noise distortion, whereas human stereo process does not. The question then is what kind of measurement primitives human stereo process does use. Arguing that the amplitude bias can be eliminated by the differential operation, the intensity derivatives are dense, and human visual system is sensitive to the intensity changes, the first order intensity derivatives (simplest derivatives) may be considered as appropriate measurement primitives for the stereo matching problem. Noise distortion, which the first order derivatives are very sensitive to, can be reduced by some smoothing techniques such as a polynomial fitting technique. The first order intensity derivatives can be obtained by directly taking the derivative about the resulting continuous intensity function. Actually, the choice of window size is closely related to the theory of human visual system. There exits at least four independent channels containing different sized spatial filters in the early visual system [54, 55]. Combination of smoothing and differentiation results in a window operator which functions very similar to the human eye in detecting intensity changes. To give some insights into the resulting window operator, a theoretical analysis of the variances of the estimated derivatives is given. Since the natural stereo images are usually digitized for the implementation on a digital computer, we consider the effect of spatial quantization on the estimation of the derivatives for the natural images.

Recently, many researchers have been using neural networks based on either intensity values or edges [8, 9, 10, 11] for stereo matching. Early work on extracting depth information from the random dot stereogram using neural network may be found in [56, 7]. In [7], a cooperative algorithm is employed to compute correspondence between the two descriptions, subject to uniqueness and continuity. Unlike standard correlation techniques, this algorithm is not restricted to minimum or maximum correlation areas to which the analysis is subsequently restricted. Although the algorithm is based on primitive descriptions such as edges for matching, no preprocessing procedure was involved in their experiments because they considered each white dot in the binary random dot stereogram as a primitive. Extension of this algorithm to natural images was reported in [57]. The natural images are first converted into binary maps by taking the sign of their convolution with the Laplacian of a Gaussian. Then the resulting binary maps serve as inputs for the cooperative algorithm. Grimson [46] further extended this algorithm using zero crossings. In this chapter, we use a neural network with maximum evolution function to solve the stereo matching problem based on the first order intensity derivatives under the epipolar, photometric and smoothness constraints. We illustrate the usefulness of this approach by using the random dot stereograms and natural image pairs.

3.2 Estimation of Intensity Derivatives

Natural digital images usually are corrupted by certain amount of noise due to electronic imaging sensor, film granularity and quantization error. The derivatives obtained using a difference operator applied to digital images are not reliable. Since a digital image comes about by sampling an analog image on an equally spaced lattice, a proper way to recover a smooth and continuous image surface is by a polynomial fitting technique. We first assume that, a point at the right image corresponding to a specified point in the left image lies somewhere on the corresponding epipolar line which is parallel to the row coordinate, i.e. in a horizontal direction, and second, in each neighborhood of image the underlying intensity function can be approximated by a fourth order polynomial. The first assumption is also known as the epipolar constraint. With the help of this constraint, the first order intensity derivatives we need for matching are computed only for the horizontal direction. Under the second assumption, the intensity function in a window, centered at the point (i, j), of size $2\omega + 1$ is represented by a polynomial of the form

$$g(i, j + y) = a_1 + a_2 y + a_3 y^2 + a_4 y^3 + a_5 y^4 \tag{3.1}$$

where y is lies in the range $-\omega$ to $+\omega$ and $\{a_i\}$ are the coefficients. If the window size is 3, then a second order polynomial is sufficient to represent the intensity function. The first order intensity derivative at point (i, j) can be easily obtained by taking the derivative about $g(i, j+y)$ with respect to y and then setting $y = 0$

$$g'(i,j) \triangleq \frac{\partial g(i,j)}{\partial j} = \frac{dg(i,j+y)}{dy}\Big|_{y=0} = a_2 \tag{3.2}$$

Thus, the estimation of first order intensity derivatives is equivalent to determination of a_2.

3.2.1 Fitting Data Using Chebyshev Polynomials

In order to estimate each coefficient independently, an orthogonal polynomial basis set is used. Several existing orthogonal polynomial basis sets can be found in [58, 59]. We use the discrete Chebyshev polynomial basis set, used by Haralick for edge detection and topographic classification [60, 61]. The important property of using polynomials is that low order fits over a large window can reduce the effects of noise and give a smooth function.

Let a set of discrete Chebyshev polynomials be defined over an index set $\Omega = \{-\omega, -\omega+1, ..., \omega-1, \omega\}$, i.e. over a window of size $2\omega + 1$, as

$$\begin{aligned}
Ch_0(y) &= 1 \\
Ch_1(y) &= y \\
Ch_2(y) &= y^2 - q_2/q_0 \\
Ch_3(y) &= y^3 - (q_4/q_2)\, y \\
Ch_4(y) &= y^4 + [(q_2q_4 - q_0q_6)/(q_0q_4 - q_2^2)]\, y^2 + (q_2q_6 - q_4^2)/(q_0q_4 - q_2^2)
\end{aligned} \tag{3.3}$$

where

$$q_n = \sum_{k \in \Omega} k^n.$$

With the window centered at point (i, j), the intensity function $g(i, j+y)$ for each $y \in \Omega$ can be obtained as

$$\hat{g}(i, j+y) = \sum_{m=0}^{4} d_m\, Ch_m(y) \tag{3.4}$$

where $\hat{g}(i, j + y)$ denotes the approximated continuous intensity function. For $\omega = 1$, only the first three Chebyshev polynomials are needed. By minimizing the least square error in estimation and taking advantage of the orthogonality of the polynomial set, the coefficients $\{d_m\}$ are obtained as

$$d_m = \frac{\sum_{y \in \Omega} Ch_m(y)\, g(i, j + y)}{\sum_{u \in \Omega} Ch_m^2(u)} \tag{3.5}$$

where $\{g(i, j + y)\}$ are the observed intensity values.

Expanding (3.4) and comparing with (3.1), the first order intensity derivative coefficient a_2, is given by

$$\begin{aligned} a_2 &= d_1 - \frac{q_4}{q_2} d_3 \\ &= \sum_{y \in \Omega} M(y)\, g(i, j + y) \end{aligned} \tag{3.6}$$

where $M(y)$ is determined by

$$M(y) = \frac{Ch_1(y)}{\sum_{u \in \Omega} Ch_1^2(u)} - \frac{q_4\, Ch_3(y)}{q_2 \sum_{u \in \Omega} Ch_3^2(u)} \tag{3.7}$$

For $\omega = 1$, the second term in (3.7) is zero. From (3.6) one can see that $M(y)$ is a filter for detecting intensity changes.

3.2.2 Analysis of Filter $M(y)$

Basically, the filter $M(y)$ used for detecting intensity changes has to satisfy the following requirements. First, it should eliminate the amplitude bias completely. Second, it should remove noise very efficiently.

For simplicity of notation, we rewrite (3.6) as

$$a_2 = M(j) * g(i, j) \tag{3.8}$$

where "*" denotes the convolution operator. Suppose that the image is corrupted by amplitude bias b and additive white noise $\{n_{i,j}\}$ with zero mean and variance σ_n^2. The observed image is

$$\tilde{g}(i,j) = g(i,j) + b + n(i,j). \tag{3.9}$$

where $\tilde{g}(i,j)$ and $g(i,j)$ are the corrupted and original intensity functions, respectively. Noting that the filter $M(j)$ is an anti-symmetric function of j, the amplitude bias b is completely eliminated after the convolution operation. Therefore,

$$M(j) * \tilde{g}(i,j) = M(j) * (g(i,j) + n(i,j)). \tag{3.10}$$

The expected value of the filter output can be written as

$$\mathbf{E}\{M(j) * \tilde{g}(i,j)\} = M(j) * g(i,j). \tag{3.11}$$

Accordingly, the variance can be expressed as

$$\begin{aligned}&\mathbf{E}\{(M(j) * \tilde{g}(i,j) - \mathbf{E}\{M(j) * \tilde{g}(i,j)\})^2\}\\ &= \mathbf{E}\{(M(j) * n(i,j))^2\}\\ &= \sigma^2 \sum_{j\in\Omega} M^2(j)\end{aligned} \tag{3.12}$$

By using (3.7), it is straightforward to prove that

$$\sum_{j\in\Omega} M^2(j) = \frac{q_6}{q_6 q_2 - q_4^2} \tag{3.13}$$

where

$$q_i = \sum_{y\in\Omega} y^i.$$

Hence, the variance of the filter output is

$$\mathbf{E}\{(M(j) * \tilde{g}(i,j) - \mathbf{E}\{M(j) * \tilde{g}(i,j)\})^2\} = \frac{\sigma_n^2 q_6}{q_6 q_2 - q_4^2} \tag{3.14}$$

For large window size, $q_6 \gg q_2$.The variance can be approximated as

$$\mathbf{E}\{(M(j) * \tilde{g}(i,j) - \mathbf{E}\{M(j) * \tilde{g}(i,j)\})^2\} = \frac{\sigma_n^2}{q_2} \tag{3.15}$$

From (3.15), one can see that the variance becomes smaller and smaller as the window size increases. For instance, if the window size is 5, then the variance is $0.9\sigma^2$. If the window size is 11, then the variance is significantly reduced to $0.009\sigma^2$. However, large window causes some loss of local information due to smoothing which smears or erases local features. If one desires to retain local features, then a small window may be used, but more noise remains and the estimated intensity function is rough. Also in order to reduce the effect of spatial quantization error for the natural images, a window as small as size of 3 may be used, as discussed in the next section. The variance of the estimated derivatives using a 3×3 window is the same as that in (3.15). It appears that the choice of the window size is closely related to the theory of human visual system. It is known [54, 55] that at least four different size channels exist in a human visual system. Marr suggested [62] that in order to efficiently detect intensity changes, the filter used should be first a differential operator, taking either a first or second order spatial derivative of the image and second be capable of being tuned to act at any appropriate scale.

The following examples show that by choosing a proper window size, the effects of noise can be very efficiently eliminated. A 256×256 real image is used in these examples.

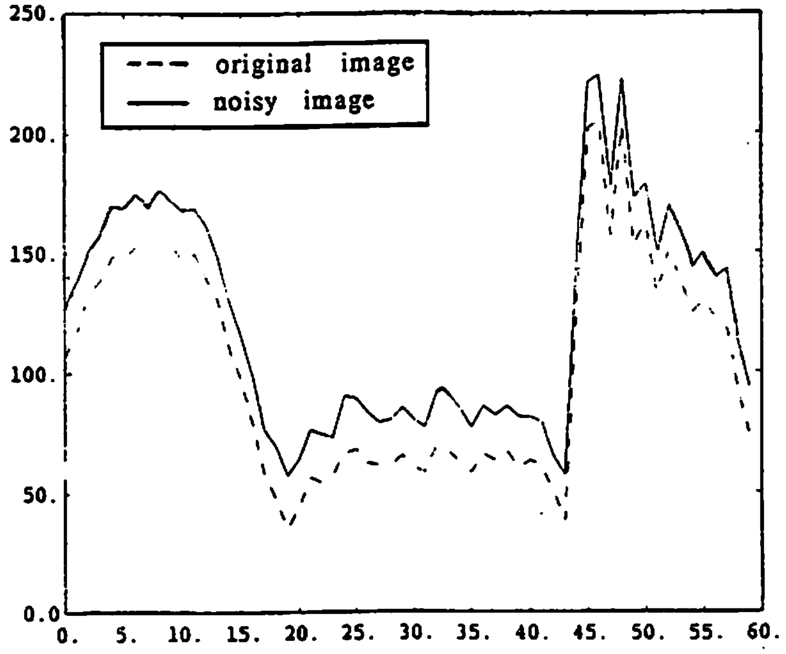
Example 1: An amplitude bias of strength 20 and white Gaussian noise (30 dB SNR) were added to the image. A section of the image is shown in Figure 3.1. The dashed and solid lines in Figure 3.1(a) represent the original and noisy images, respectively. Obviously, there is no way to match these two image based on the noisy biased intensity values only. Figure 3.1(b) shows the estimated first order intensity derivatives from these two images using the polynomial method. The window size is 5, i.e. the index set is $\{-2,-1,0,1,2\}$.

Example 2: An amplitude bias of size 20 and white Gaussian noise corresponding to 20 dB SNR were added to the original image. Figure 3.2 shows a section of the image taken from the same location as in Example 1. Figure 3.1(a) gives the original and noisy biased images. Figure 3.1(b) shows the estimated first order intensity derivatives of these two images. Since noise in this case is large, a large window of size 11 was used to reduce its effect. One can see that the derivatives of two images are matched very well.

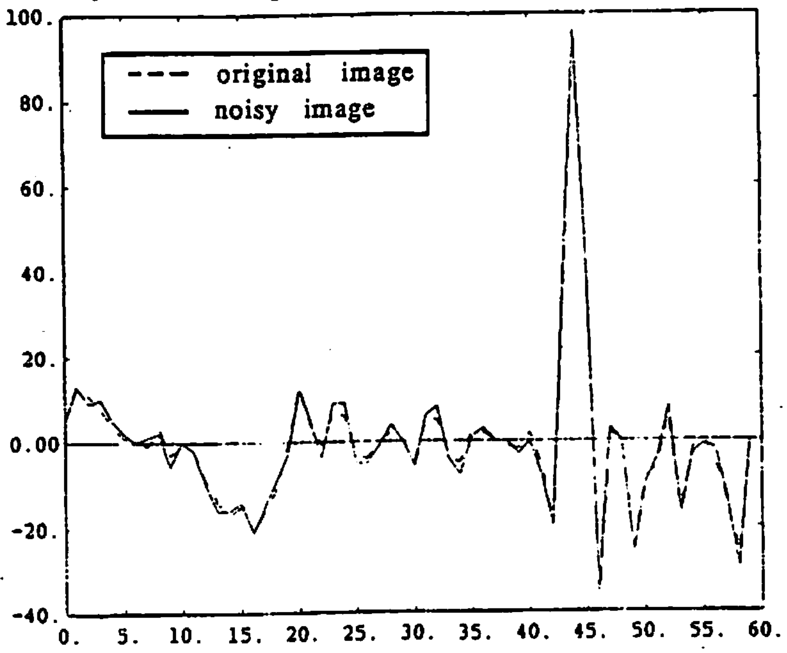
3.2.3 Computational Consideration for Natural Images

For the implementation on a digital computer, the natural stereo images must be digitized both spatially and in the amplitude. Under the perspective projection, the natural stereo pair images, that is, the left and right images, can not be matched very well at sample points because of the spatial quantization error. The spatial quantization error affects the intensity function as well as the derivatives. In this section, we consider the effect of spatial quantization error on the estimation of intensity derivatives. Similar results also hold for edge detection. A recent discussion about the problems of quantization error in stereo matching can be found in [63].

For analysis purposes, a typical camera configuration system similar to that used by Horn [64] is given in Figure 3.3. Assume that the two cameras are rigidly attached to each other so that their optical axes are parallel and separated

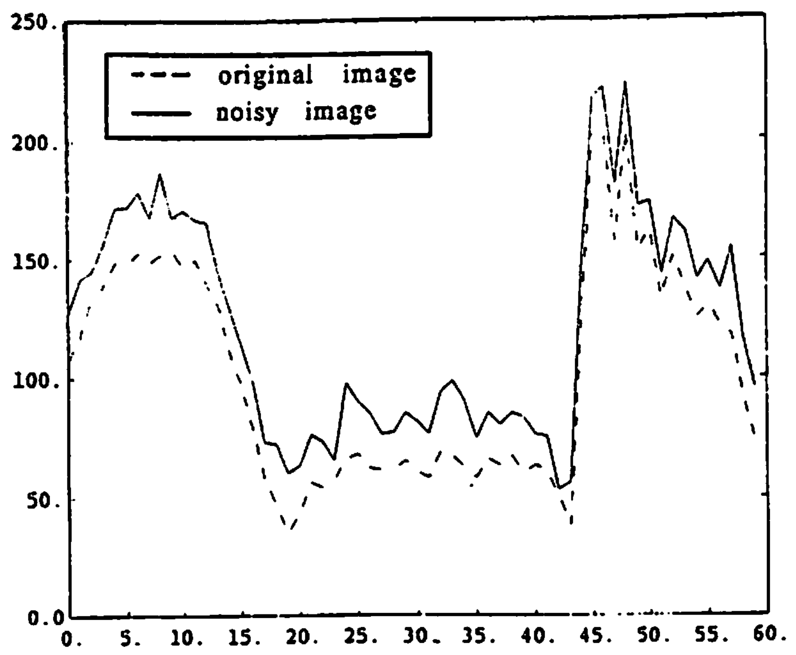


(a) Intensity values of original and noisy images.

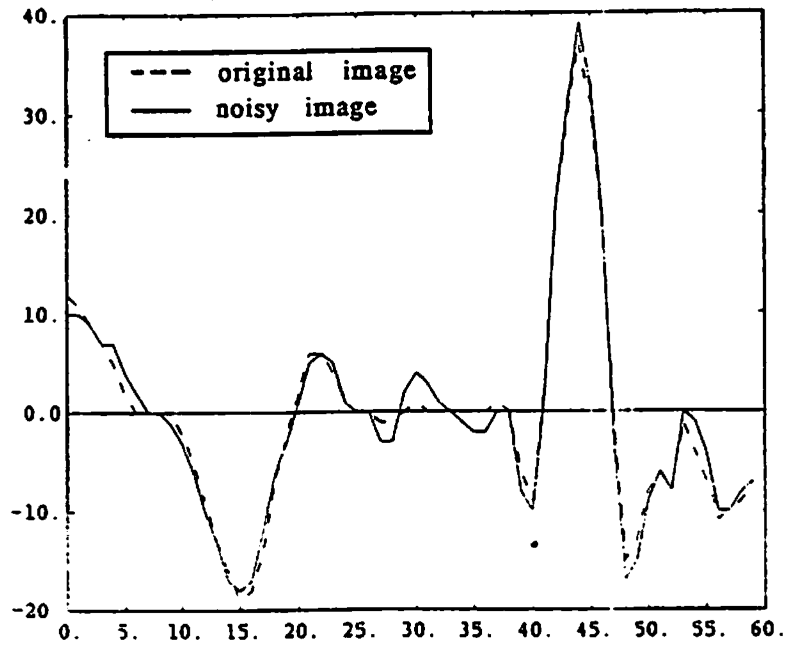


(b) First order derivatives of intensity values of original and noisy images.

Figure 3.1: A section of a real image with amplitude bias 20 and 30 dB noise.



(a) Intensity values of original and noisy images.



(b) First order derivatives of intensity values of original and noisy images.

Figure 3.2: A section of a real image with amplitude bias 20 and 20 dB noise.

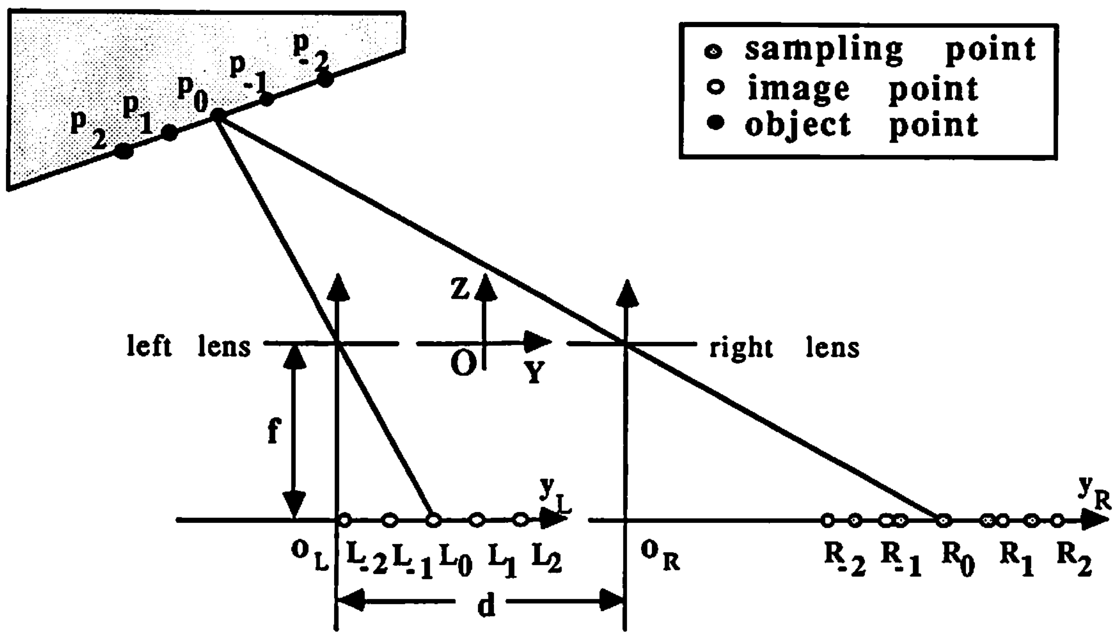


Figure 3.3: Camera geometry for stereo photography.

by a distance d. The focal length of the lens is denoted by f which takes a negative value in the world coordinate system $OXYZ$. The origin of a right handed coordinate system of the world is located midway between the camera lens centers. The positive Z-axis is directed along the camera optical axes. The baseline connecting the lens centers is assumed to be perpendicular to the optical axes and oriented along the Y-axis. Let the coordinate systems of the left and right image plane be $o_L x_L y_L$ and $o_R x_R y_R$, respectively. Then a point in the world, (X, Y, Z), projects into the left and right image planes at

$$(x_L, y_L) = (\frac{f\,X}{Z}, \frac{f\,(Y + \frac{d}{2})}{Z}) \tag{3.16}$$

and

$$(x_R, y_R) = (\frac{f\,X}{Z}, \frac{f\,(Y - \frac{d}{2})}{Z}) \tag{3.17}$$

respectively. Disparity $D_{X,Y}$ can then be defined as

$$D_{X,Y} \triangleq y_R - y_L = -f\,d\,\frac{1}{Z} \tag{3.18}$$

Suppose we sample the left image uniformly at line $X_L = X_R = X_0$, a set of equally-spaced points $\{..., L_{-2}, L_{-1}, L_0, L_1, L_2, ...\}$ are obtained at

$$\{..., (x_0, y_{L_{-2}}), (x_0, y_{L_{-1}}), (x_0, y_{L_0}), (x_0, y_{L_1}), (x_0, y_{L_2}), ..., \}$$

The corresponding object points $\{..., P_{-2}, P_{-1}, P_0, P_1, P_2, ...\}$ are located at

$$\{..., (X_0, Y_{-2}, Z_{-2}), (X_0, Y_{-1}, Z_{-1}), (X_0, Y_0, Z_0), (X_0, Y_1, Z_1), (X_0, Y_2, Z_2), ...\}$$

on the surface. These object points also project into the right image plane at

$$\{..., (x_0, y_{R_{-2}}), (x_0, y_{R_{-1}}), (x_{R_0}, y_{R_0}), (x_0, y_{R_1}), (x_0, y_{R_2}), ..., \}$$

When the object surface is not parallel to the image plane, the corresponding points on the surface are unequally-spaced. Consequently, the image points in the right image plane are also unequally-spaced which means that the image points do not match the sample points everywhere if the right image is uniformly sampled. This phenomenon is shown in Figure 3.3.

We assume that in the left image plane, the sample points match the image points exactly, and in the right image only the image point R_0 matches the sample point as illustrated in Figure 3.3 and the other image points do not match the sample points. Thus

$$y_{L_i} - y_{L_0} = y_{L_i^s} - y_{L_0^s} \tag{3.19}$$

and

$$y_{L_i} - y_{L_0} = y_{R_i^s} - y_{R_0} \tag{3.20}$$

where the "s" denotes the sample point. By (3.19) and (3.20), the spatial quantization error, i.e. the distance between the sample point and the corresponding image point can be calculated as

$$\eta_i = \begin{cases} (y_{R_i} - y_{R_0}) - (y_{R_i^s} - y_{R_0}) = f\, d\,(\frac{1}{Z_0} - \frac{1}{Z_i}), & i > 0 \\ (y_{R_0} - y_{R_i}) - (y_{R_0} - y_{R_i^s}) = f\, d\,(\frac{1}{Z_i} - \frac{1}{Z_0}), & i < 0 \end{cases} \qquad (3.21)$$

Obviously,

$$\eta_i > \eta_{i-1}, \;\; i > 0,$$

and

$$\eta_i < \eta_{i-1}, \;\; i < 0.$$

This shows that the spatial quantization error depends on Z, f and the distance d between the cameras. If the object surface is parallel to the image plane, then the sample points will match the corresponding image points perfectly because

$$Z_0 = Z_i, \quad \forall \; i.$$

An interesting aspect of (3.21) is that by definition of disparity in (3.18) the spatial quantization error is exactly equal to the difference of the disparities between points P_0 and P_i. Therefore, stereo matching algorithms using intensity values as the measurement primitives can not detect such a difference if the sample interval is twice as large as the spatial quantization error.

We further assume that the incident illumination and absorption characteristics of the object surface are roughly constant, and the surface orientation and the distance to two cameras are almost same. Therefore, the left and right image planes receive the same amounts of light which means that the intensity functions of conjugate image points are almost same. It is assumed that the intensity function is differentiable with respect to y. Expand the intensity function $g(x_0, y_{R_i^s})$

as a Taylor series about $(x_0, y_{R_i^s}) = (x_0, y_{R_i})$

$$g(x_0, y_{R_i^s}) = \begin{cases} g(x_0, y_{R_i} - \eta_i) = g(x_0, y_{R_i}) - \eta_i\, g'(x_0, y_R)|_{y_R = y_{R_i}} + O(\eta_i^2), & i > 0 \\ g(x_0, y_{R_i} + \eta_i) = g(x_0, y_{R_i}) + \eta_i\, g'(x_0, y_R)|_{y_R = y_{R_i}} + O(\eta_i^2), & i < 0 \end{cases} \tag{3.22}$$

where $g'(x_0, y_R)|_{y_R = y_{R_i}}$ is the derivative of the intensity function at (x_0, y_{R_i}). By using the sampled intensity function to estimate the first order derivative of the intensity function $g(x_0, y_{R_0})$, (3.6) becomes

$$\tilde{g}'(x_0, y_{R_0}) = \sum_{y \in \Omega} M(y)\, g(x_0, (y_{R_0} + y)^s) \tag{3.23}$$

where the "$\sim$" denotes the estimate of intensity derivative using the sampled intensity functions.

Replacing the sampled intensity functions in (3.23) by (3.22), we have

$$\tilde{g}'(x_0, y_R)|_{y_R = y_{R_0}} \simeq \sum_{i \in \Omega} M(i)\, g(x_{R_0}, y_{R_i}) + \sum_{i \in \Omega} u(i)\, \eta_i\, M(i)\, g'(x_0, y_R)|_{y_R = y_{R_i}} \tag{3.24}$$

where $u(i)$ is a step function

$$u(i) = \begin{cases} 1, & i > 0 \\ -1, & i < 0. \end{cases}$$

Clearly, the first term in the right side of (3.24) is equal to (3.6) which means it is a correct estimate, and the second term is an estimation error caused by the spatial quantization error. Since the spatial quantization error is proportional to f and d, and is inversely proportional to Z, the estimation error will be small when the camera is close enough and/or the object is far enough. When the surface is not parallel to the image plane and the object is close to the camera, using a large window to estimate the derivatives will give a large error due to the accumulated quantization error. Hence, a small window is preferred if the object is close to the camera. As proposed in [65], the smallest channel in the human visual system contains a filter with a central diameter of $1.5'$, roughly corresponding to 4 pixels. Therefore, considering the effects of noise distortion and the spatial quantization error, a filter $m(y)$ with size of $3-7$ pixels is the proper one for the natural stereo images.

In fact, (3.24) can be considered as a general form for both derivative estimation and edge detection as most of the edge detection algorithms can be considered as a window operation followed by appropriate thresholding. Owing to the output error of the window operation, the edge detector may miss an edge, give a false edge or shift the edge. In other words, the edge output also suffers from the spatial quantization error.

Noting that the filter $m(y)$ is an anti-symmetric function of y and assuming that the derivatives at sample points are the same, (3.24) can be simplified as

$$\tilde{g}'(x_0, y_r)|_{y_r=y_{r_0}} \simeq g'(x_0, y_r)|_{y_r=y_{r_0}}[1 - \sum_{i=1}^{\omega} m(i)\ (\eta_i + \eta_{-i})]. \tag{3.25}$$

Substituting (3.21) and then (3.18), we finally have

$$\tilde{g}'(x_0, y_r)|_{y_r=y_{r_0}} \simeq g'(x_0, y_r)|_{y_r=y_{r_0}}[1 - \sum_{i=1}^{\omega} m(i)\ (D_i - D_{-i})] \tag{3.26}$$

The estimate of the derivatives may be either larger or smaller than the true value which depends on the orientation of the object surface.

3.3 Matching Using a Network

Binary neurons are used to represent the disparity values between two images. The network consists of $N_r \times N_c \times (D+1)$ mutually interconnected neurons, where D is the maximum disparity, N_r and N_c are the image row and column sizes, respectively. Let $V = \{v_{i,j,k}, 1 \leq i \leq N_r, 1 \leq j \leq N_c, 0 \leq k \leq D\}$ be a binary state set of the neural network with $v_{i,j,k}$ (1 for firing and 0 for resting) denoting the state of the (i,j,k)th neuron. When $v_{i,j,k}$ is 1, this means that the disparity value is k at the point (i,j). Every point is represented by $D+1$ mutually exclusive neurons, i.e. only one neuron is firing and others are resting, as a result of the uniqueness constraint of the matching problem.

The network parameters, the interconnection strengths $T_{i,j,k;l,m,n}$ and the bias inputs $I_{i,j,k}$, can be determined in terms of the energy function of the network. As defined by (2.7), the energy function of the neural network can be written as

$$E = -\frac{1}{2}\sum_{i=1}^{N_r}\sum_{l=1}^{N_r}\sum_{j=1}^{N_c}\sum_{m=1}^{N_c}\sum_{k=0}^{D}\sum_{n=0}^{D} T_{i,j,k;l,m,n}\, v_{i,j,k}\, v_{l,m,n} - \sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=0}^{D} I_{i,j,k}\, v_{i,j,k} \tag{3.27}$$

In order to use the spontaneous energy–minimization process of the neural network, we reformulate the stereo matching problem under the epipolar assumption as one of minimizing an error function with constraints defined as

$$E = \sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=0}^{D} (g'_L(i,j) - g'_R(i,j\oplus k))^2\, v_{i,j,k} + \frac{\lambda}{2}\sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=0}^{D}\sum_{s\in S} (v_{i,j,k} - v_{(i,j)\oplus s,k})^2 \tag{3.28}$$

where $\{g'_L(\cdot)\}$ and $\{g'_R(\cdot)\}$ are the first order intensity derivatives of the left and right images, respectively, S is an index set excluding $(0,0)$ for all the neighbors in a $\Gamma \times \Gamma$ window centered at point (i,j), λ is a constant and the symbol $\oplus$ denotes that

$$f_{a\oplus b} = \begin{cases} f_{a+b} & if \;\; 0 \leq a+b \leq N_c, N_r \\ 0 & otherwise \end{cases} \tag{3.29}$$

The first term called the photometric constraint in (3.28) is to seek disparity values such that all regions of two images are matched in a least squares sense.

Meanwhile, the second term is the smoothness constraint on the solution. The constant λ determines the tradeoff between the two terms to achieve the best results.

By taking $\Gamma = 5$ and comparing the terms in the expansion of (3.28) with the corresponding terms in (3.27), we can determine the interconnection strengths and bias inputs as

$$T_{i,j,k;l,m,n} = -48\lambda\delta_{i,l}\delta_{j,m}\delta_{k,n} + 2\lambda\sum_{s\in S}\delta_{(i,j),(l,m)\oplus s}\delta_{k,n} \tag{3.30}$$

and

$$I_{i,j,k} = -(g'_L(i,j) - g'_R(i,j\oplus k))^2 \tag{3.31}$$

where $\delta_{a,b}$ is the Dirac delta function. The size of the smoothing window used in (3.30) is 5. However, one can choose either a larger or smaller window. From (3.30) one can see that the interconnections are symmetric and the self-connection $T_{i,j,k;i,j,k}$ is not zero which requires self-feedback for neurons.

Matching is carried out by neuron evaluation. Once the parameters $T_{i,j,k;l,m,n}$ and $I_{i,j,k}$ are obtained using (3.30) and (3.31), each neuron can randomly and asynchronously (or synchronously) evaluate its state and readjust according to

$$u_{i,j,k} = \sum_{l=1}^{N_r}\sum_{m=1}^{N_c}\sum_{n=0}^{D} T_{i,j,k;l,m,n}v_{l,m,n} + I_{i,j,k} \tag{3.32}$$

and

$$v_{i,j,k} = g(u_{i,j,k}) \tag{3.33}$$

where $g(x_{i,j,k})$ is a maximum evolution function

$$g(x_{i,j,k}) = \begin{cases} 1 & if\ \ x_{i,j,k} = max(x_{i,j,l}; l = 0, 1, ..., D). \\ 0 & otherwise. \end{cases} \tag{3.34}$$

The synchronous updating scheme can be implemented in parallel, while the asynchronous updating scheme can be sequentially implemented. Another updating scheme called the hybrid updating scheme results when some neurons are synchronously updated and the others are asynchronously updated. For natural stereo images, we will use the hybrid neural network. The uniqueness of matching problem is ensured by a batch updating scheme–$D+1$ neurons $\{v_{i,j,0}, ...v_{i,j,D}\}$ at location (i,j) are updated at each step simultaneously.

The initial state of the neurons were set as

$$v_{i,j,k} = \begin{cases} 1 & if \;\; I_{i,j,k} = max(I_{i,j,l}; l = 0,1,...,D). \\ 0 & otherwise \end{cases} \tag{3.35}$$

where $I_{i,j,k}$ is the bias input.

As mentioned in Chapter 2, the self–feedback may cause energy function to increase with a transition. The batch updating scheme simultaneously updates $(D+1)$ neurons $\{v_{i,j,k}; k = 0,...,D\}$ corresponding to the image point (i,j) at each step. However, at most two of the $(D+1)$ neurons change their state at each step. By defining the state changes $\Delta v_{i,j,k}$ and $\Delta v_{i,j,k'}$ of neurons (i,j,k) and (i,j,k') and energy change ΔE as

$$\Delta v_{i,j,k} = v_{i,j,k}^{new} - v_{i,j,k}^{old}$$

$$\Delta v_{i,j,k'} = v_{i,j,k'}^{new} - v_{i,j,k'}^{old}$$

and

$$\Delta E = E^{new} - E^{old}$$

the change ΔE due to a changes $\Delta v_{i,j,k}$ and $\Delta v_{i,j,k'}$ can be obtained as

$$\Delta E \;\; = \;\; -(\sum_{l=1}^{N_r} \sum_{m=1}^{N_c} \sum_{n=0}^{D} T_{i,j,k;l,m,n} \, v_{l,m,n} + I_{i,j,k} \,) \, \Delta v_{i,j,k} - \frac{1}{2} \, T_{i,j,k;i,j,k} \, (\Delta v_{i,j,k})^2$$

$$-(\sum_{l=1}^{N_r}\sum_{m=1}^{N_c}\sum_{n=0}^{D} T_{i,j,k';l,m,n}\, v_{l,m,n} + I_{i,j,k'}\ )\,\Delta v_{i,j,k'} - \frac{1}{2}\, T_{i,j,k';i,j,k'}\, (\Delta v_{i,j,k'})^2$$
$$-T_{i,j,k;i,j,k'}(\Delta v_{i,j,k} v^{new}_{i,j,k'} + \Delta v_{i,j,k'} v^{new}_{i,j,k}). \tag{3.36}$$

When

$$v^{old}_{i,j,k} = 0, \qquad v^{old}_{i,j,k'} = 1,$$

$$u_{i,j,k} > u_{i,j,k'}$$

and the maximum evolution function is as in (3.34), we have

$$v^{new}_{i,j,k} = 1, \qquad v^{new}_{i,j,k'} = 0$$

and

$$\Delta v_{i,j,k} = 1, \qquad \Delta v_{i,j,k'} = -1.$$

Noting that

$$T_{i,j,k;i,j,k'} = 0 \quad if \quad k \neq k',$$

(3.36) can be simplified as

$$\Delta E = (u_{i,j,k'} - u_{i,j,k}) - \frac{1}{2}\,(T_{i,j,k;i,j,k} + T_{i,j,k';i,j,k'}) \tag{3.37}$$

Thus, the first term in (3.37) is negative. But

$$T_{i,j,k;i,j,k} + T_{i,j,k';i,j,k'} = -96\ \lambda < 0$$

leading to

$$-\frac{1}{2}\,(T_{i,j,k;i,j,k} + T_{i,j,k';i,j,k'}) > 0$$

When the first term is less than the second term in (3.36), then $\Delta E > 0$.

A deterministic decision rule is used to ensure convergence of the network, probably to a local minimum. The stereo matching algorithm can then be summarized as

1. Set the initial state of the neurons.

2. Update the state of all neurons randomly and asynchronously (or synchronously) according to the deterministic decision rule.

3. Check the energy function; if energy does not change anymore, stop; otherwise, go back to step 2.

3.4 Experimental Results

A variety of images including random dot stereograms and natural stereo image pairs were tested using our algorithm. A 5×5 (i.e. $\Gamma = 5$) smoothing window was used for all images.

3.4.1 Random Dot Stereograms

The random dot stereograms were created by the pseudo random number generating method described in [66]. Each dot consists of only one element. All the following random dot stereograms are of size 128×128 and in the form of a three level "wedding cake". The background plane has zero disparity and each successive layer plane has additional two elements of disparity. In order to implement this algorithm more efficiently on a conventional computer, we make the following simplifications. Since only one of $D+1$ neurons is firing at each point, we used one neuron lying in the range 0 to D to represent the disparity value instead of $D+1$ neurons. From (3.30) one can see that the interconnections between the neurons are local (a $\Gamma \times \Gamma$ neighborhood) and have the same structure for all neurons.

Therefore, for $\Gamma = 5$ we used a 5×5 window for computing $U_{i,j,k}$ and energy function E instead of a $N_r N_c(D+1) \times N_r N_c(D+1)$ interconnection strength matrix. The simplified algorithm greatly reduces the space complexity by increasing the program complexity a little. Therefore, it is very fast and efficient.

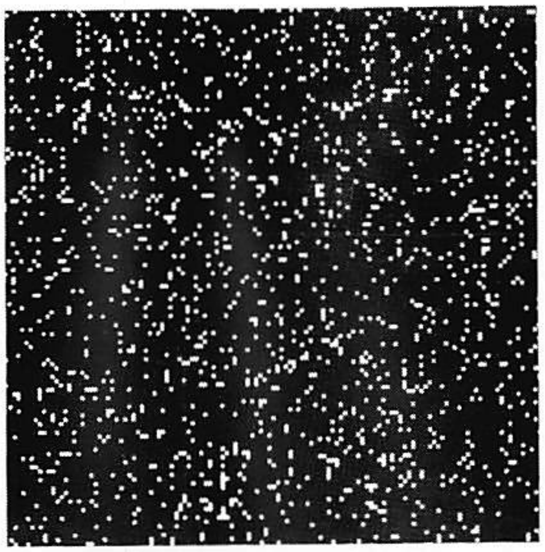
Figure 3.4 shows a 10% random dot stereogram. Intensity values of the white and black elements are 255 and 0, respectively. Figure 3.4(c) is the resulting disparity map after 10 iterations using the asynchronous updating scheme. When the synchronous updating scheme is used, 23 iterations are needed. The disparity values are encoded as intensity values with the brightest value denoting the maximum disparity value. We used $\lambda = 20$, $D = 6$ and $\omega = 2$ (i.e. window size was 5). Note that the disparity map is dense.

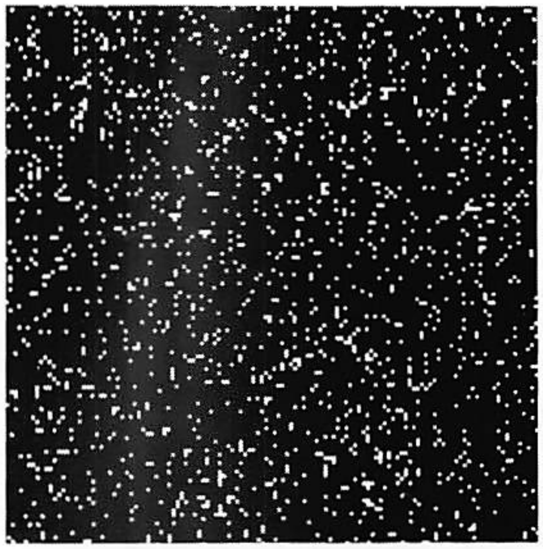
A similar test was run on the decorrelated stereogram [53]. The original stereogram is 50% density random dots. In the left image, 20% of the dots were decorrelated at random. By setting $\lambda = 2800$, $D = 6$ and $\omega = 2$, a dense disparity map in Figure 3.5(c) was obtained after 12 asynchronous iterations. The same result can be obtained after 19 synchronous iterations. .

Another type of perturbation is Gaussian white noise [66]. Figures 3.6(a) and 3.6(b) show a pair of multi gray level random dot images with intensity value in the range $(0 - 255)$. Gaussian white noise corresponding to 5 dB SNR was added to the left image. The SNR is defined as

$$SNR = 10 \log_{10} \frac{\sigma_o^2}{\sigma_n^2}$$

where σ_o^2 and σ_n^2 are the variances of original left image and noise. The parameters $\lambda = 450$, $D = 6$ and $\omega = 2$ were used. Only 6 asynchronous iterations were needed


(a) Left image.


(b) Right image.

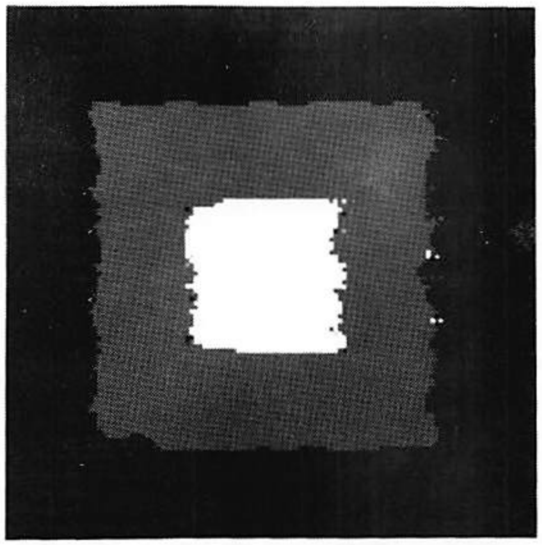
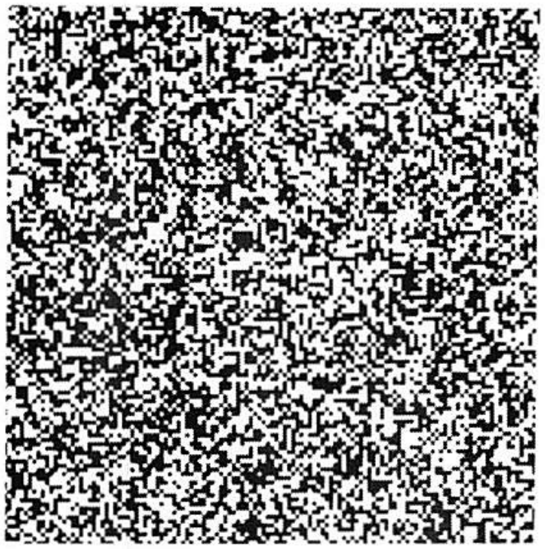
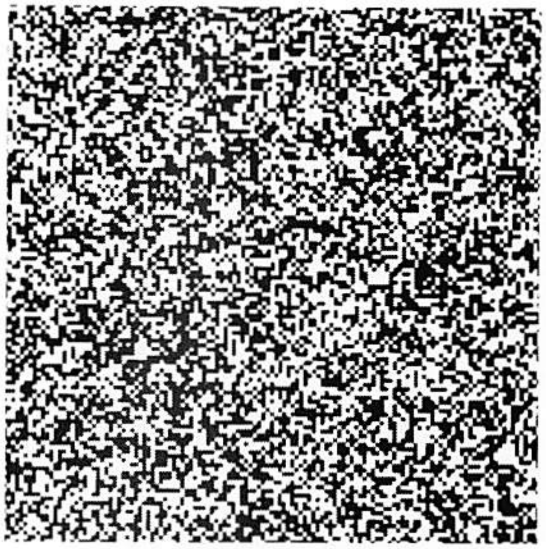

(c) Disparity map represented by an intensity image.

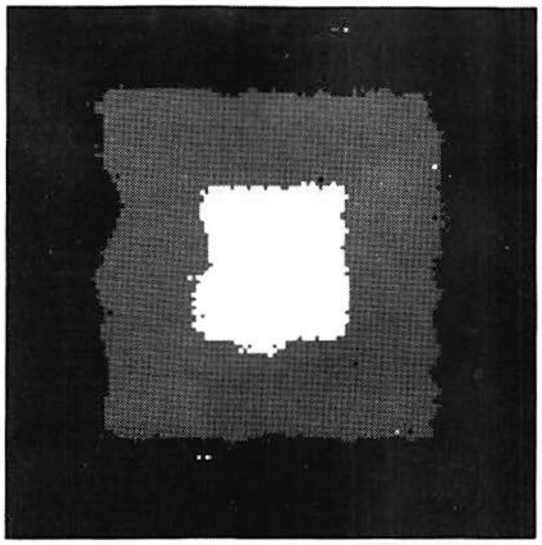
Figure 3.4: A 10% density random dot stereogram.



(a) Left image.



(b) Right image.



(c) Disparity map represented by an intensity image.

Figure 3.5: A 50% density random dot stereogram. In the left image, 20% of the dots were decorrelated at random.

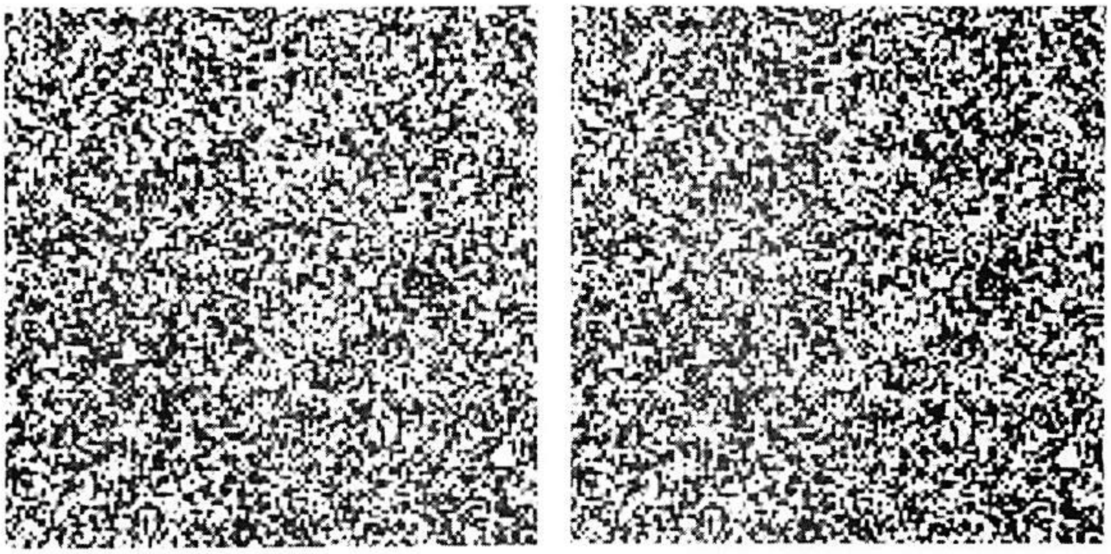
to get the final result in Figure 3.6(c). When the synchronous updating scheme was used, 9 iterations were needed to get the same result.

As expected, both the synchronous and asynchronous updating schemes work very well, although the latter takes more iterations. The synchronous updating scheme is suitable for parallel processing.

3.4.2 Natural Stereo Images

Two stereo pairs of natural images, the Renault part and the Pentagon images, were used to test our algorithm. All images are of size 256×256. Since natural stereo images may not satisfy the epipolar constraint, small alignment corrections in the vertical direction are needed. A hybrid updating scheme was used for both the Renault and the Pentagon image pairs. The image is segmented into homogeneous and nonhomogeneous regions by using a local variance criterion. A homogeneous region is defined as a smooth region with the small local variances. The corresponding neurons corresponding to homogeneous image regions are updated sequentially, while the neurons corresponding to nonhomogeneous regions are updated in parallel. Since the first derivatives of the intensity function in homogeneous regions are small, the inputs are small and the neurons tend to take the same state as their neighbors because of the smoothness constraint. No doubt, the neurons near the boundary will be first affected by the neighbors corresponding to inhomogeneous regions. As the neurons corresponding to homogeneous region are sequentially updated, they will all be affected by the boundary conditions which means surfaces in homogeneous regions can be interpolated.

For the Renault images, the parameters were set as $\lambda = 12$, $D = 13$ and $\omega = 1$. The threshold for the local variances was set at 1.0. The local variance was


(a) Left image.

(b) Right image.

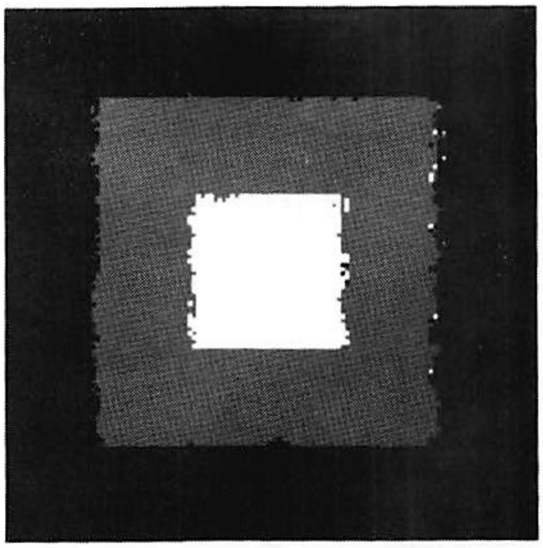

(c) Disparity map represented by an intensity image.

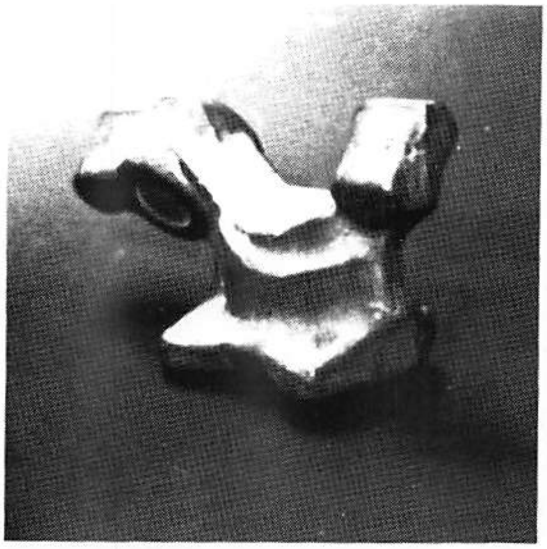
Figure 3.6: A multilevel random dot stereogram. In the left image, Gaussian white noise corresponding to 5 dB SNR has been added.

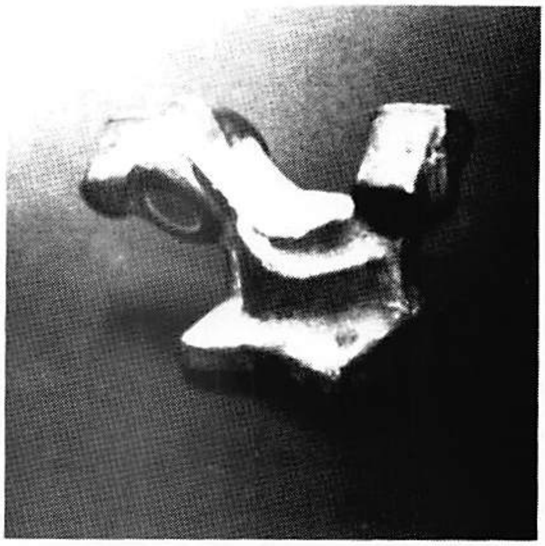
computed in a 5×5 window. About 72 iterations were required. Since a discrete network was used, the disparities only take integer values. A simple smoothing technique was applied to the nonzero portions of the disparity surface. Figures 3.7(a) and 3.7(b) show the left and right Renault part images. The final result is given in Figure 3.7(c), while (d) shows the smoothed version of (c) by using a 9×9 mean filter. Figures 3.8 and 3.9 give the plots of the unsmoothed and smoothed disparity surfaces corresponding to Figures 3.7(c) and 3.7(d), respectively.

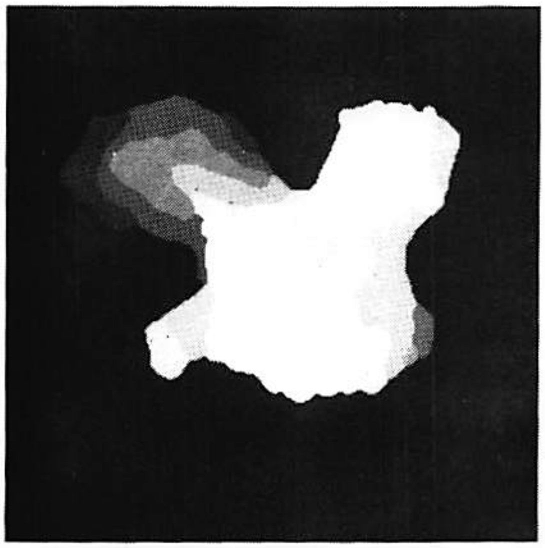
Figures 3.10(a) and 3.10(b) show the left and right Pentagon images. By choosing parameters $\lambda = 10$, $D = 4$, $\omega = 1$ and the local variance threshold 0.01, a disparity map was generated after 51 iterations. A 5×5 window was used for computing the local variance. Figures 3.10(c) and 3.10(d) give the unsmoothed and smoothed disparity maps, respectively. A 13×13 filter was used for smoothing. The plots of Figures 3.7(c) and 3.7(d) are shown in Figures 3.11 and 3.12, respectively.

3.5 Discussion

This chapter has presented an approach for extracting depth from a pair of static stereo images. A discrete neural network is employed to iteratively minimize the error function and an estimate of the disparity values is obtained when the neural network converges to a minimum. The experimental results offer support for the hypothesis that the first order derivatives of the intensity function may be considered as appropriate measurement primitives for the stereo matching problem. No surface interpolation step is involved in this approach because of the dense first order derivatives of the intensity function used as measurement primitives.


(a) Left image.


(b) Right image.


(c) Disparity map.

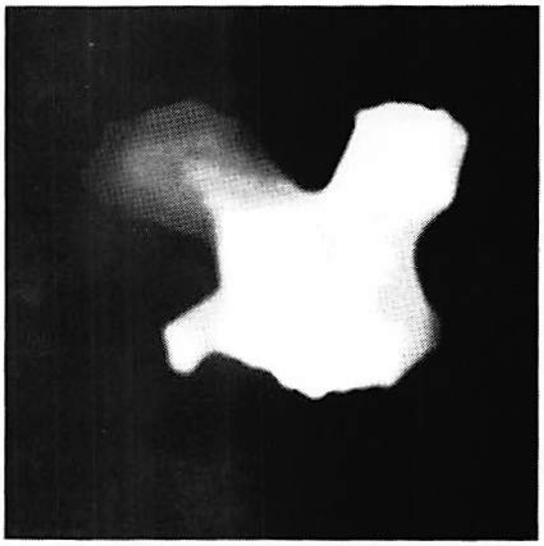

(d) Smoothed disparity map.

Figure 3.7: The Renault part images. The disparity maps are represented by intensity images.

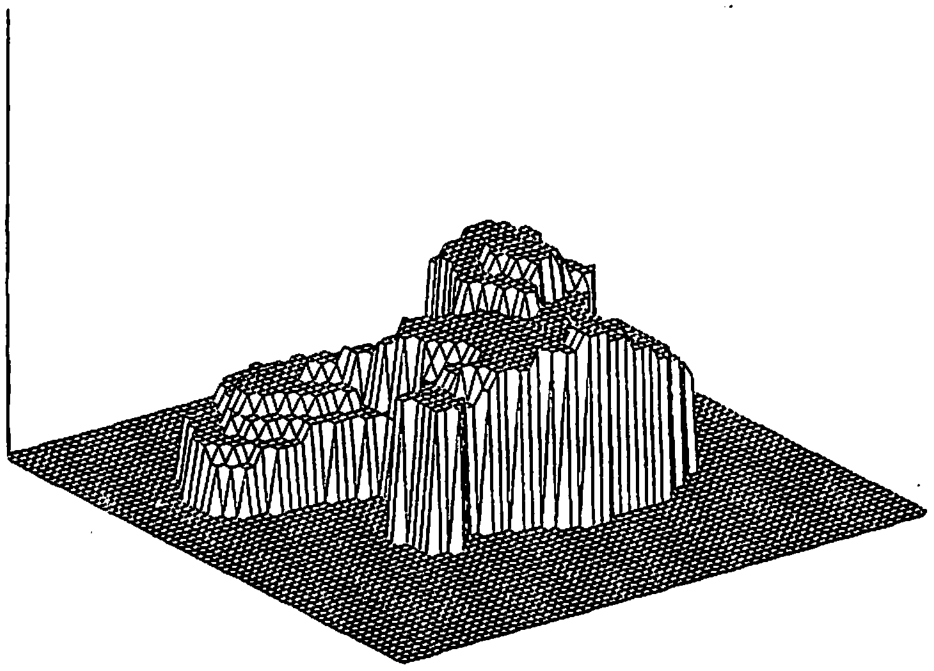
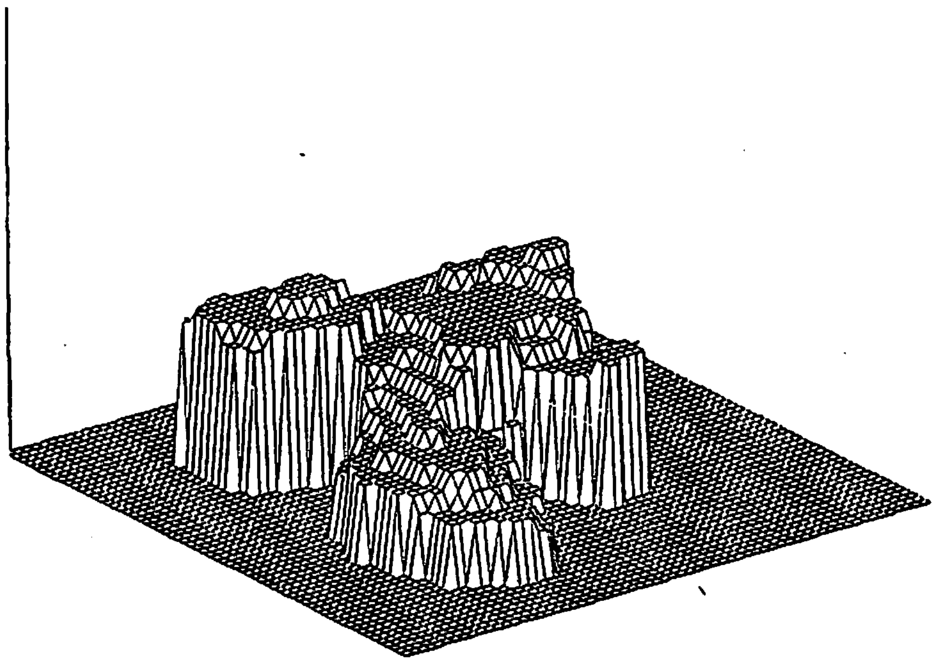



Figure 3.8: 3-D plots of the disparity map for the Renault part images.

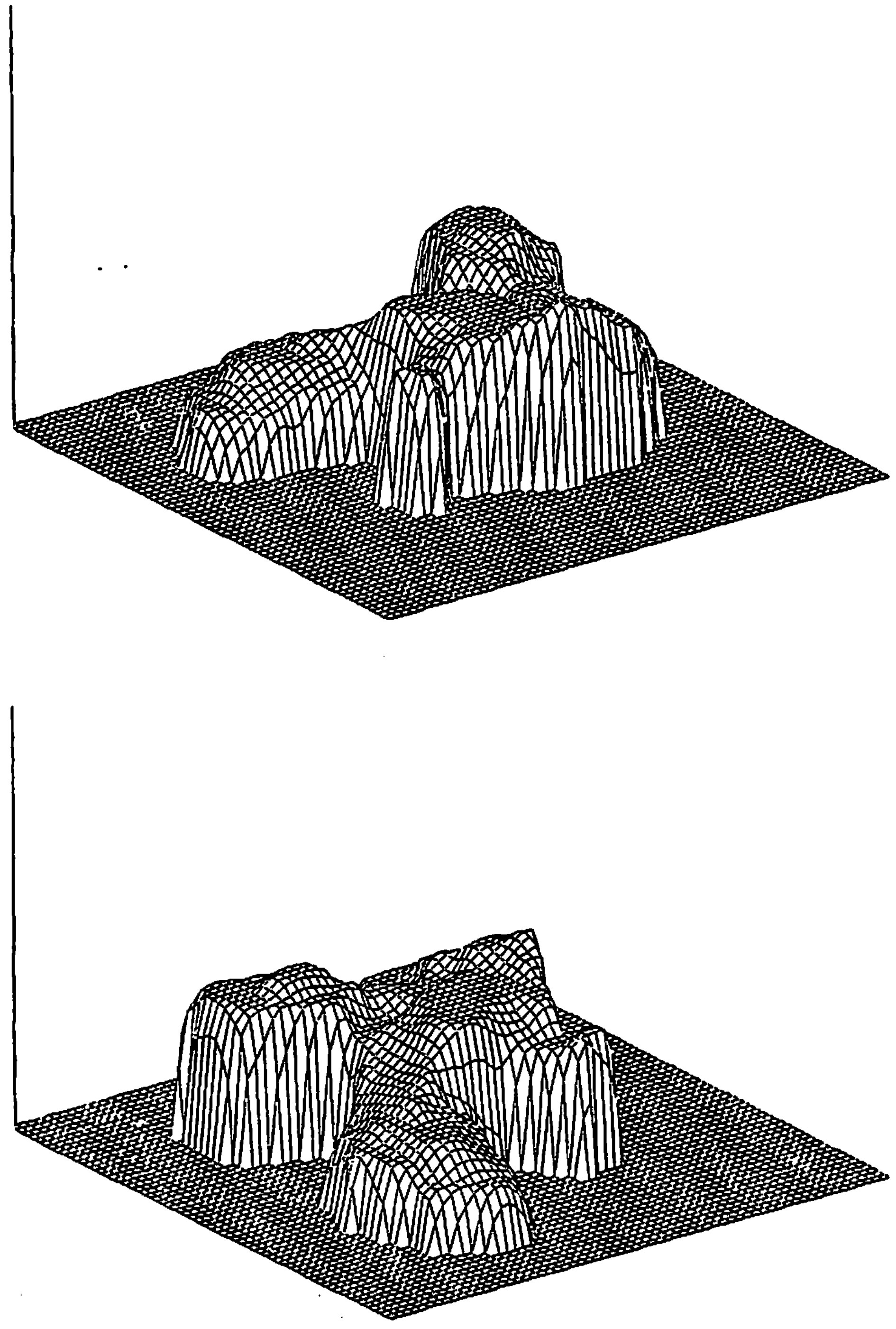
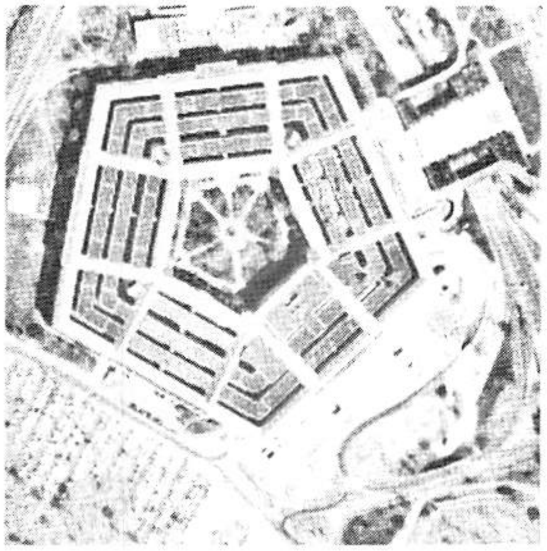
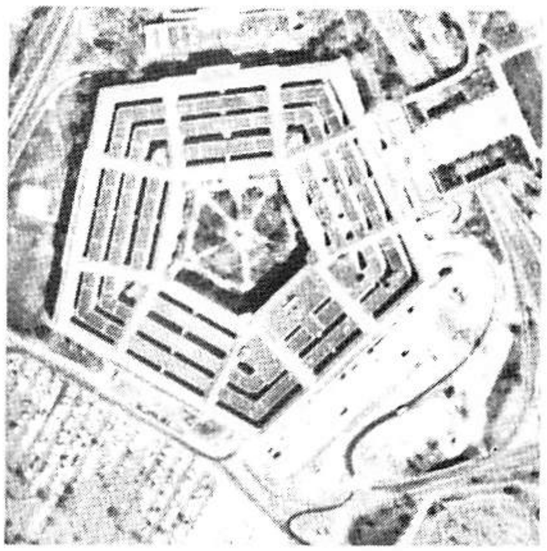


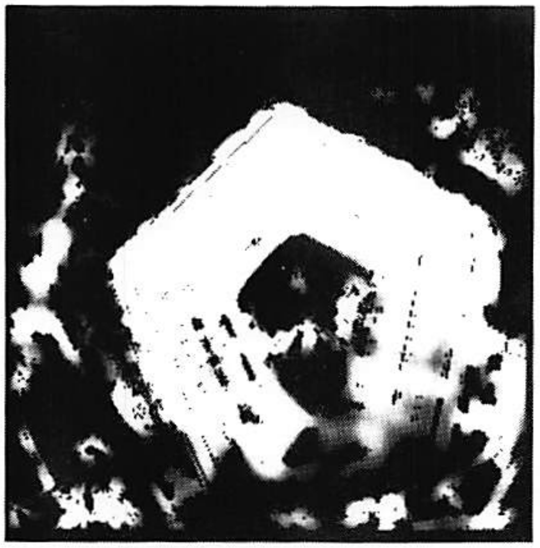
Figure 3.9: 3-D plots of the smoothed disparity map for the Renault part images.



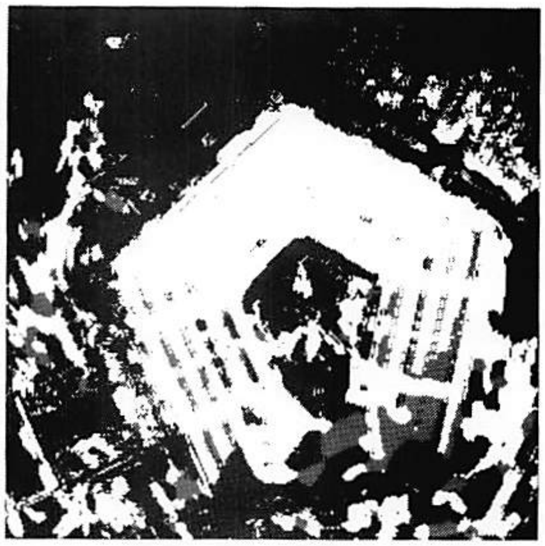
(a) Left image.



(b) Right image.

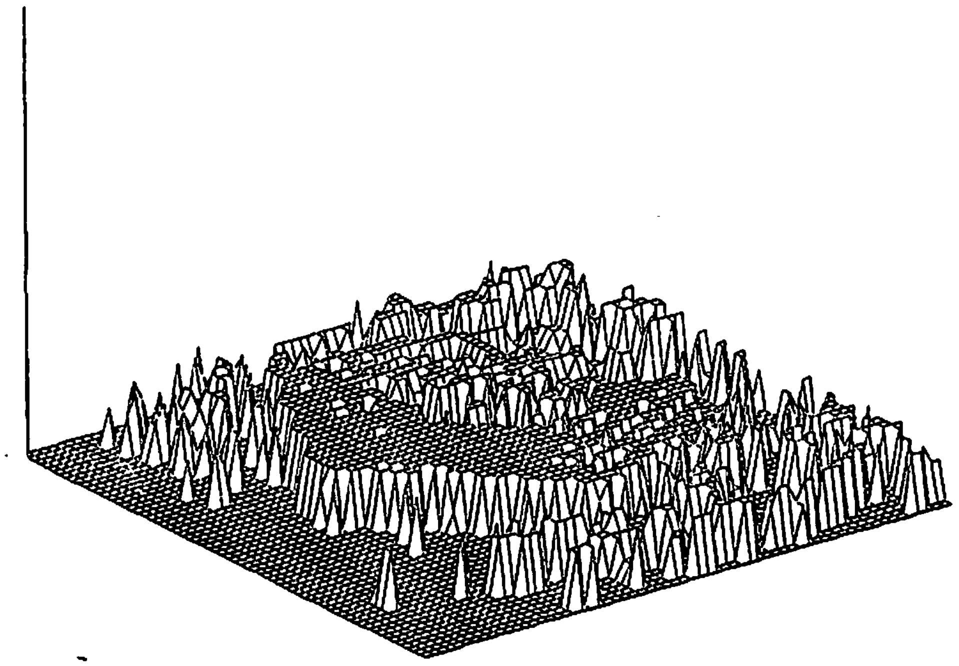


(c) Disparity map.



(d) Smoothed disparity map.

Figure 3.10: The Pentagon part images. The disparity maps are represented by intensity images.



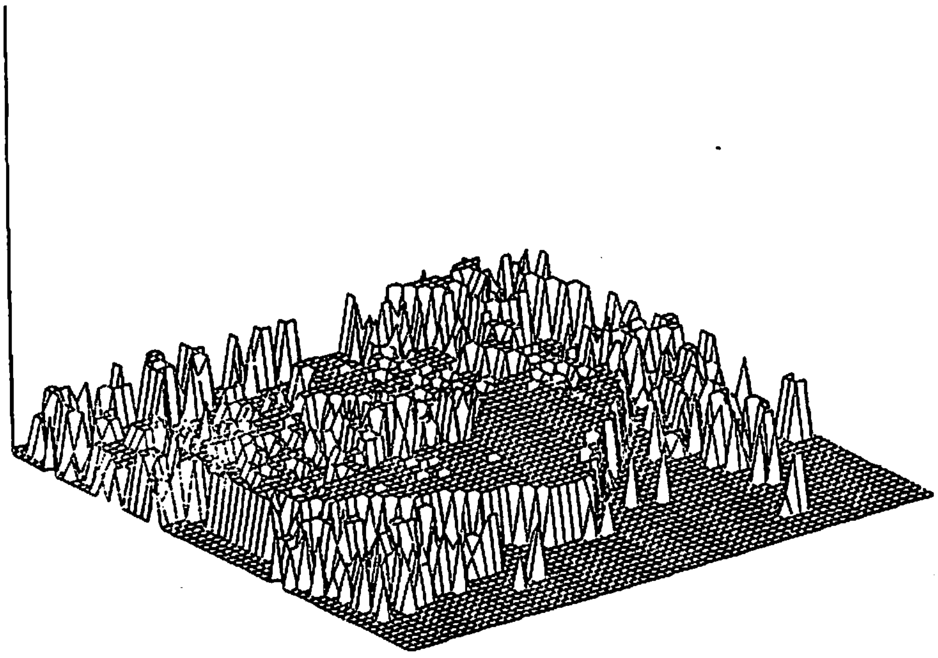
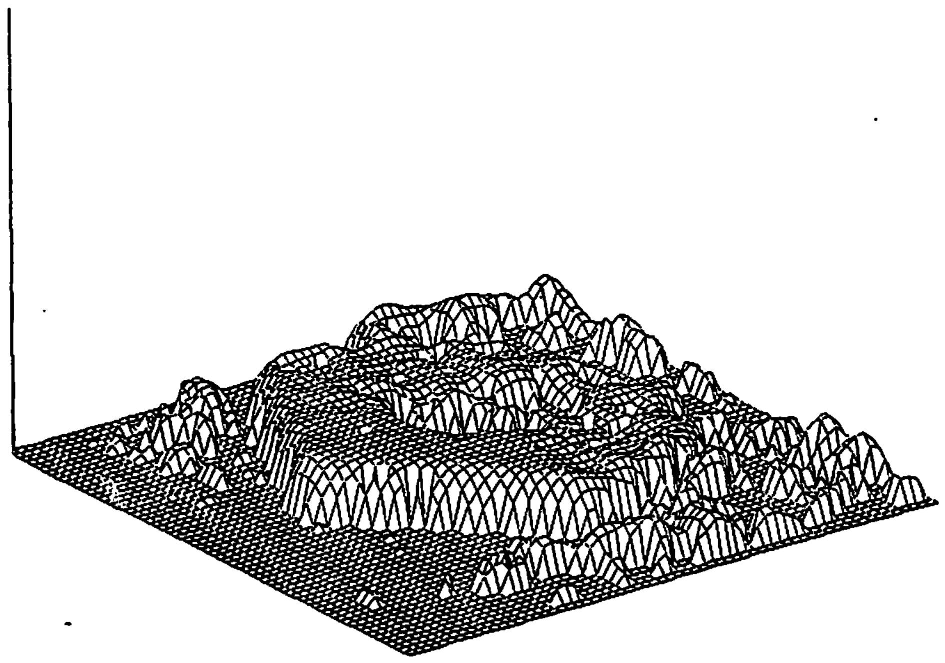


Figure 3.11: 3-D plots of the disparity map for the Pentagon images.



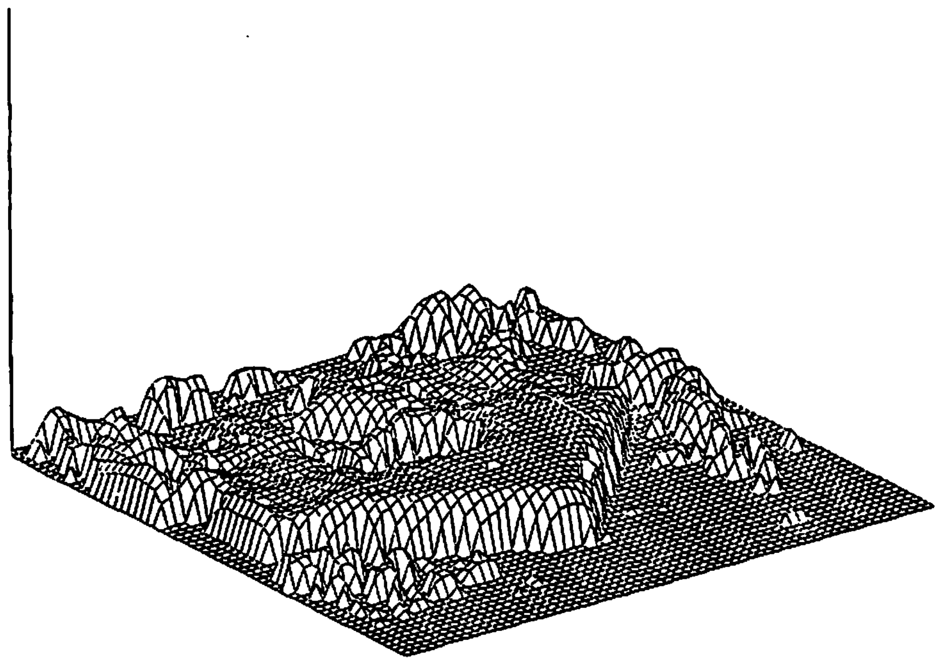


Figure 3.12: 3-D plots of the smoothed disparity map for the Pentagon images.

In considering the experimental results, however, one point should be kept in mind. It concerns the occluding pixels. At the location of the occluding pixels, the disparity values are undetermined. Hence, we have to detect the occluding pixels and establish some rules to infer the depth information at such location. This issue will be discussed in the next chapter.

Chapter 4

Motion Stereo

4.1 Introduction

Motion stereo is a method for deriving depth information from either a moving camera or objects moving through a stationary 3-D environment. Since motion stereo uses more than two images, it usually gives more accurate depth measurements than static stereo. In this chapter we give two neural network based approaches, the batch and recursive approaches, for computing disparities using a sequence of image frames.

A substantial amount of work has been devoted to methods for computing the disparity field based on a sequence of images. In a relatively early paper, Nevatia [67] uses multiple views between two stereo views to achieve certain accuracy without an increase in search time. The object is placed on a turntable and multiple views are taken by a camera every 0.5 degree apart. Two simple methods are suggested for region search. Both methods only reduce the search range but not increase the resolution. Also they do not make use of any information acquired in the previous view for the next search procedure. Williams presents a new approach for deriving depth from a moving camera in [68]. By moving the camera forward, the disparity is estimated using simple triangulation. For simplicity, all the object surfaces are assumed to be flat and oriented in either horizontal direction i.e. parallel to image plane or vertical direction i.e. parallel to ground plane. Therefore, only the distances for horizontal surface and the height for vertical surface need be found. To achieve subpixel accuracy, an image is interpolated according to the predicted disparity values obtained by a search process and occlusion effects. Based on the error between real and interpolated images, the correct orientation of each surface can be detected, and hence a segmented image consisting of refined synthetic surfaces can be obtained. For implementation purposes, an iterative segmentation procedure is employed and the systematic changes of distance and height embodied in synthetic segmented image at each iteration are used for finding the correct distance and height. Experimental results demonstrate the usefulness of this approach for simple natural image sequences. Since only planar surfaces with one of two different orientations are assumed to exist in natural images, areas corresponding to either non-planar surfaces or planar surfaces with other orientations are not correctly interpolated and therefore the estimated distances and heights for these areas are not reliable. Furthermore, this approach requires information about the focus of expansion (FOE) and the final result depends very much on the quality of initial segmentation.

Instead of computing depth in image space, Jain, Bartlett and O'Brien [69] developed a method for estimating the depth of feature points (corners) in the ego-motion complex logarithmic mapping (ECLM) space. They showed that the axial movement of the camera causes only horizontal but not the vertical change in the mapping of image points. Therefore, the depth of a feature point can be determined from the horizontal displacement in the ECLM for that point and from the camera velocity in the gaze direction. However, the mapping is very sensitive to noise, spatial quantization error and image blur, requiring some heuristics to establish the correspondence of points, such as thresholds for maximum possible changes in the vertical direction and an upper bound for the search range in the horizontal direction in the ECLM space. Also the FOE for arbitrary translation of the camera and the feature points (corners) are assumed to be known. Another motion stereo method using feature points (corners) for computing depth in image space can be found in [70].

Recently, Xu, Tsuji and Asada [71] have suggested a coarse-to-fine iterative method for motion stereo matching. By sliding a camera along a straight line, a sequence of images is taken at predetermined positions. The pair with short baseline is matched first to produce a coarse disparity map based on the zero-crossings. Then the coarse disparity map is used to reduce search range for the pair with the next longer baseline. This procedure is continued until the pair with the largest baseline is processed. One major advantage of this method is that occlusions can be predicted from the previous disparity map to avoid mismatches at present step. Although the computation time is less compared to other coarse-to-fine methods, this method gives only a sparse disparity map and can not be implemented on line.

Matthies, Szeliski and Kanade [12] have introduced two real time approaches, based on intensity values and features using a Kalman filtering technique. A sequence of lateral motion images is generated by a moving camera along a straight line from left to right (or right to left). The intensity based approach consists of four stages for each frame. First, a new measurement of disparity at each pixel is obtained by using a correlation matching procedure. Then the estimate of disparity is updated by a Kalman filter update equation based on the new measurement. Third, a generalized piecewise continuous spline technique is used to smooth the updated estimate. Finally, the disparity for each pixel in the next frame is given by the prediction procedure. As reported in [12], the intensity based approach is more efficient than the feature based approach. But a major problem in the intensity based approach is that once the updated estimate is smoothed in the third stage, the gain of the Kalman filter and the error variance of the estimate are no longer correct so that they can not be used in the next iteration.

In this chapter, we develop a neural network based batch approach for computing the disparity values. Since the batch approach needs to use the matching procedure only once, computational complexity is greatly reduced. We also present a new real time approach for deriving depth from sequential images. Basically, we use a recursive least squares (RLS) algorithm to update the bias inputs of the network instead of updating disparity values. When the next frame becomes available, the bias inputs are updated first and then a neural network is employed to estimate the disparity values under the epipolar, photometric and smoothness constraints. Unlike [12], no smoothing procedure is needed. Since the neural network can be run in parallel and the RLS algorithm can be implemented on line, this approach is extremely fast and hence useful for real time robot vision applications. As the derivatives of the intensity function are more reliable than the intensity values and are also dense, both approaches use the derivatives as measurement primitives for matching.

4.2 Depth from Motion

It is assumed that a sequence of images is taken by a camera moving with constant velocity from right to left along a straight line, as shown in Figure 4.1. Several assumptions are made for simplifying the problem. First, it is assumed that the optical axis of the camera is perpendicular to the moving direction and the horizontal axes of the image planes are parallel to the moving direction. The constraint imposed on the camera configuration is to restrict the search within the horizontal direction only, the so called epipolar constraint. Secondly, it is assumed that the camera takes pictures exactly every *t* seconds apart. Thus, all images are equally separated, i.e. each successive image pairs has the same baseline.

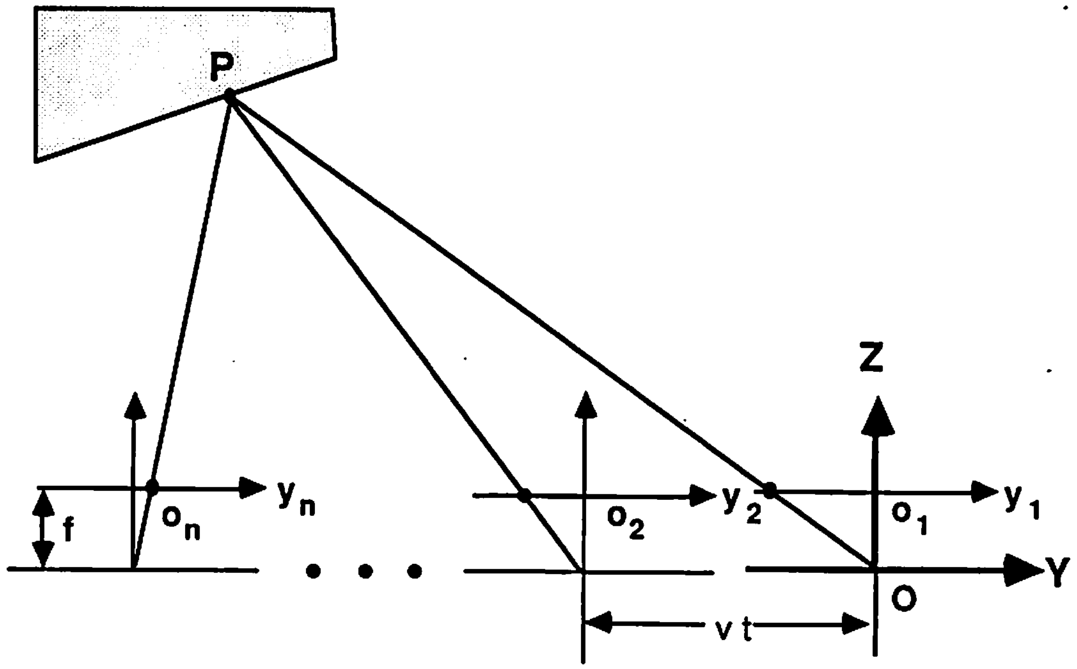


Figure 4.1: Camera geometry for motion stereo.

Let $OXYZ$ be the world coordinate system with Z axis directing along the camera optical axis and $o_i x_i y_i$ be the pth image plane coordinate system. The origin of the pth image system is located at $(0, -(p-1)vt, f)$ of the world system, where v is the velocity of camera, vt is the distance between two successive images and f is the focal length of the lens which takes a positive value in the world system. Under perspective projection, a point in the world, (X_o, Y_o, Z_o), projects into the pth image plane at

$$(x_i, y_i) = (\frac{f\,X_0}{Z_o}, \frac{f\,(Y_o + (p-1)\,v\,t)}{Z_o}) \tag{4.1}$$

Theoretically, the disparity D_o can be derived from two successive image frames

$$D_o = y_p - y_{p-1} = f\;v\;t\,\frac{1}{Z_o} \tag{4.2}$$

which is same as the formula used in static stereo. However, noise distortion, spatial quantization error and motion blur limit the estimation accuracy. In order to achieve high accuracy, a long sequence of image frames is needed.

4.3 Estimation of Derivatives

As only derivatives in the horizontal direction are required for matching, the epipolar constraint saves a lot of computations. By using a set of univariate discrete Chebyshev polynomials to approximate the intensity function in a window, we have

$$\hat{g}(i, j+y) = \underline{\mathbf{A}}^t\;\underline{\mathrm{CH}}(y) \tag{4.3}$$

where $\hat{g}(i, j+y)$ is the approximated continuous intensity function, t denotes the transpose operator,

$$\underline{\mathbf{A}}^t = [a_0, a_1, a_2, a_3, a_4]$$

is the coefficient vector and

$$\underline{\mathbf{CH}}^t(y) = [Ch_0(y),\ Ch_1(y),\ Ch_2(y),\ Ch_3(y),\ Ch_4(y)]$$

is polynomial vector defined over an index set $\Omega = \{-\omega, -\omega+1, ..., \omega-1, \omega\}$, i.e. over a window of size $2\omega+1$, as in Section2.1 of Chapter 3. The coefficients $\{a_n\}$ are estimated using

$$a_n = \frac{\sum_{y' \in \Omega} Ch_n(y')\, g(i, j+y')}{\sum_{u \in \Omega} Ch_n^2(u)} \tag{4.4}$$

where $\{g(i, j+y')\}$ are the observed intensity values.

The first order derivatives of the intensity function at subpixel position $(i, j+y)$ can be calculated by

$$\frac{\partial g(i,j)}{\partial j}\Big|_{j=j+y} = \frac{d\hat{g}(i, j+y)}{dy} = \underline{\mathbf{A}}^t\, \frac{d}{dy}\underline{\mathbf{CH}}(y) \tag{4.5}$$

$$for \quad -0.5 \le y < 0.5$$

For simplicity of notation, we use $g'(i, j+y)$ to represent the first order partial derivatives of the subpixel intensity function.

For the purposes of performance analysis, rewrite the coefficient vector as

$$\underline{\mathbf{A}} = \sum_{y' \in \Omega} \underline{\mathbf{A}}(y')\, g(i, j+y') \tag{4.6}$$

where $\underline{\mathbf{A}}(y')$ is a vector with the nth element

$$\frac{Ch_n(y')}{\sum_{u \in \Omega} Ch_n^2(u)}.$$

Replacing the vector $\underline{\mathbf{A}}$ in (4.5) with (4.6) we have

$$g'(i, j+y) = \sum_{y' \in \Omega} F_y(y; y')\, g(i, j+y') \tag{4.7}$$

where $F_y(y; y')$ is given by

$$F_y(y; y') = \underline{\mathbf{A}}^t(y')\, \frac{d}{dy}\underline{\mathbf{CH}}(y), \tag{4.8}$$

a filter for estimating the first order derivatives of the subpixel intensity function. Since only one camera is used, there is no bias amplitude in the observations. Suppose that the image is corrupted by zero mean white noise

$$\tilde{g}(i,j) = g(i,j) + n(i,j). \tag{4.9}$$

where $\tilde{g}(i,j)$ and $g(i,j)$ are the noisy observations and original intensity value, respectively. The variance is given by

$$\begin{aligned}
&\mathbf{Var}(\sum_{y'\in\Omega} F_y(y;y')\tilde{g}(i,j+y')) \\
&= \mathbf{E}\{(\sum_{y'\in\Omega} F_y(y;y')\, n(i,j+y'))^2\} \\
&= \sigma_n^2 \sum_{y'\in\Omega} F_y^2(y;y')
\end{aligned} \tag{4.10}$$

where σ_n^2 is the variance of noise. Making use of the orthogonality of the polynomial set, we have

$$\begin{aligned}
\sum_{y'\in\Omega} F_y^2(y;y') &= \sum_{y'\in\Omega} (\underline{\mathbf{A}}^t(y')\,\frac{d}{dy}\underline{\mathbf{CH}}(y))^2 \\
&= (\frac{d}{dy}\underline{\mathbf{CH}}^t(y)) \sum_{y'\in\Omega} \underline{\mathbf{A}}(y')\,\underline{\mathbf{A}}^t(y')\,\frac{d}{dy}\underline{\mathbf{CH}}(y) \\
&= (\frac{d}{dy}\underline{\mathbf{CH}}^t(y))\,\underline{\mathbf{W}}\,\frac{d}{dy}\underline{\mathbf{CH}}(y)
\end{aligned} \tag{4.11}$$

where $\underline{\mathbf{W}}$ is a diagonal matrix with elements

$$w_{i,i} = \frac{1}{\sum_{u\in\Omega} Ch_i^2(u)}$$

$$for \quad i = 0, 1, ..., 4.$$

Thus, the variance becomes

$$\mathbf{Var}(\sum_{y'\in\Omega} F_y(y;y')\tilde{g}(i,j+y'))$$

$$= \sigma_n^2 \left(\frac{d}{dy}\underline{\mathbf{CH}}^t(t)\right) \underline{\mathbf{W}} \frac{d}{dy}\underline{\mathbf{CH}}(y)$$

$$= \sigma_n^2 \sum_{n=1}^{4} \frac{(\frac{d}{dy} Ch_n(y))^2}{\sum_{v \in \Omega} Ch_n^2(v)} \tag{4.12}$$

which shows that the denominator of each term is a monotonically increasing function of the window size. Hence the variance of the output becomes smaller and smaller as the window size increases. Considering the effects of spatial quantization (as discussed in Chapter 3) and noise, a window with the size of $3 - 7$ pixels is recommended for natural images.

4.4 Batch Approach

The conventional batch algorithm needs many measurements requiring a lot of computations. For example, if there are M image frames, then the matching procedure has to be implemented $(M - 1)$ times to obtain $(M - 1)$ disparity measurements for each pixel. Such a batch approach does not have any advantage over the recursive approach. Instead of doing matching $(M - 1)$ times, the batch method presented in this section implements the matching algorithm only once by simultaneously using all the images pairs.

Theoretically, disparity takes continuous values. For implementation purposes, we sample the disparity range using bins of size W. We use $N_r \times N_c \times (D + 1)$ mutually interconnected binary neurons to represent the disparity measurements, where N_r and N_c are the image row and column sizes and $D \times W$ is the maximum disparity value. For the (i, j)th pixel, we use $(D + 1)$ mutually exclusive neurons $\{v_{i,j,0}, v_{i,j,1}, \ldots, v_{i,j,D}\}$. When $v_{i,j,k}$ is 1, the disparity measured at pixel (i, j) is $k\,W$.

4.4.1 Estimation of Pixel Positions

Let (i,j) be the position of the (i,j)th pixel in the first frame. In the successive frames, due to camera motion, the position of all pixels are shifted to the right by vt. Under the epipolar constraint, the shift happens only in the horizontal direction. For example, the (i,j)th pixel moves from position (i,j) in the first frame to position $(i, j + \frac{fvt}{Z_{i,j}})$ in the second frame. Let $S_{i,j}(p)$ be the total shift of pixel (i,j) from the first to the pth frame. Thus

$$\begin{aligned} S_{i,j}(p) &= (p-1)\,\frac{fvt}{Z_{i,j}} \\ &= (p-1)\,d_{i,j} \end{aligned} \tag{4.13}$$

where $d_{i,j}$ is the true disparity value for pixel (i,j). Note that the shift $S_{i,j}(p)$ is continuous due to the continuous variable $d_{i,j}$. A rounding operation has to be applied to $S_{i,j}(p)$ for locating the (i,j)th pixel in the subsampled image. After rounding, the position of the (i,j)th pixel in the pth frame is given by

$$(i, j + [\frac{S_{i,j}(p)}{W}]W). \tag{4.14}$$

where [ ] is a rounding operator. It can be simply written as

$$(i, j + kW) \tag{4.15}$$

where

$$k = [\frac{S_{i,j}(p)}{W}].$$

4.4.2 Batch Formulation

Assuming that the camera is moving along Y axis with constant velocity v and the images are taken exactly every t seconds apart, the error function for matching is defined as

$$\begin{aligned} E &= \sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=0}^{D}\sum_{p=1}^{M-1}\left(g_p'(i,j\oplus(p-1)kW)-g_{p+1}'(i,j\oplus pkW)\right)^2 v_{i,j,k} \\ &+\frac{\lambda}{2}\sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=0}^{D}\sum_{s\in S}(v_{i,j,k}-v_{(i,j)\oplus s,k})^2 \end{aligned} \qquad (4.16)$$

where $\{g_p'(\cdot)\}$ denote the intensity derivatives at $(\cdot)$ of the pth frame. Comparison of (4.16) to (3.27) results in

$$T_{i,j,k;l,m,n} = -48\lambda\delta_{i,l}\delta_{j,m}\delta_{k,n} + 2\lambda\sum_{s\in S}\delta_{(i,j),(l,m)\oplus s}\delta_{k,n} \qquad (4.17)$$

and

$$I_{i,j,k} = -\frac{1}{M-1}\sum_{p=1}^{M-1}\left(g_p'(i,j\oplus(p-1)kW)-g_{p+1}'(i,j\oplus pkW)\right)^2 \qquad (4.18)$$

where $T_{i,j,k;l,m,n}$ and $I_{i,j,k}$ are the interconnection strength and the bias input, respectively. Then a synchronous updating scheme described in Section 3.3 of the Chapter 3 is used to update the states of neurons. The final outputs of the neural network give the disparity estimates.

4.5 Recursive Approach

This approach basically consists of two steps: bias input update and stereo matching. Whenever a new frame of image becomes available, the bias inputs of the network are updated by the RLS algorithm and then new disparity measurements are obtained using the new bias inputs.

Suppose that images are corrupted by additive white noise and the measurement model is given by

$$\begin{aligned} \tilde{I}_{i,j,k}(p) &= h(p, g_p'(i,j\oplus(p-1)kW), g_{p+1}'(i,j\oplus pkW), n_{i,j,k}(p)) \\ &= -\left(\tilde{g}_p'(i,j\oplus(p-1)kW)-\tilde{g}_{p+1}'(i,j\oplus pkW)\right)^2 \end{aligned} \qquad (4.19)$$

$$for \quad p = 1, 2, ..., M-1$$

where h is a measurement function and $n_{i,j,k}(p)$ is noise. For p such measurements, find a function

$$\hat{I}_{i,j,k}(p) = \hat{I}_{i,j,k}(p, \tilde{I}_{i,j,k}(p), \tilde{I}_{i,j,k}(p-1), ..., \tilde{I}_{i,j,k}(1)) \tag{4.20}$$

that estimates the value of the bias input $I_{i,j,k}$ in some sense. The value of the function is the estimate. If the measurement function is linear and the measurement noise is white, then a Kalman filter is commonly used for finding an optimal estimate. In (4.19), as the measurement function is nonlinear and the measurement noise is no longer white but is dependent on measurements, the linear Kalman filter does not yield a good estimate. In contrast to the Kalman filter, the RLS algorithm does not make any assumption about measurement function and noise. Hence, the RLS algorithm can be used to update the bias inputs. When the pth frame becomes available, the bias input is updated by

$$I_{i,j,k}(p) = I_{i,j,k}(p-1) + \frac{1}{p}\,(\tilde{I}_{i,j,k}(p) - I_{i,j,k}(p-1)), \tag{4.21}$$

This RLS algorithm is equivalent to the batch least squares algorithm with the initial condition

$$I_{i,j,k}(0) = 0.$$

The matching algorithm is the same as that used for static stereo matching. The interconnection strengths is the same as in (3.30)

$$T_{i,j,k;l,m,n} = -48\lambda\delta_{i,l}\delta_{j,m}\delta_{k,n} + 2\lambda \sum_{s\in S} \delta_{(i,j),(l,m)\oplus s}\delta_{k,n} \tag{4.22}$$

and the bias input is given by (4.21). Since the bias inputs are recursively updated and contain all the information about the previous images, we do not have to implement the matching algorithm for every recursion if the intermediate result are not required. This method greatly reduces the computational load and therefore is extremely fast. Formally, the algorithm is as follows:

1. Update the bias inputs using the RLS algorithm.
2. Initialize the neuron states.
3. If there is a new frame to be processed, go back to step 1; otherwise go to step 4.
4. Update the neuron states using the matching algorithm.

This algorithm has several advantages over the correlation algorithm of [12]:

1. This algorithm recursively updates the bias inputs instead of the disparity values. The matching algorithm is implemented only once.
2. This algorithm incorporates the smoothness constraint into the matching procedure instead of using an extra smoothing procedure.
3. This algorithm uses the derivatives of the intensity function, which are more reliable than the intensity values, as measurement primitives. Hence it is suitable for natural images.

4.6 Matching Error

When multiple frames are used for matching, the spatial quantization error usually causes a large matching error. In this section, we derive a mean value for the matching error which can be used to detect the occluding pixels. It is assumed that the true disparity at pixel (i,j) in a smooth region can be expressed as

$$d_{i,j} = kW + \delta_{i,j}$$

where $\delta_{i,j}$ is uniformly distributed in $[-\frac{W}{2}, \frac{W}{2})$, and the first order derivative of the intensity function at point $(i, j+(p-1)kW)$ of the pth image can be expanded as a Talor series about the point $(i, j+(p-1)(kW+\delta_{i,j}))$ as

$$\begin{aligned} g'(i,j+(p-1)kW) &= g'(i,j+(p-1)(kW+\delta_{i,j}) - \\ &\quad (p-1)\delta_{i,j}g''(i,j+(p-1)(kW+\delta_{i,j}) + O(\delta_{i,j}^2). \\ &\quad for \quad p = 2,3,...,M \end{aligned} \tag{4.23}$$

where $g''(\cdot)$ denotes the second order derivative. The best estimate of the disparity value is given by

$$\hat{d}_{i,j} = kW.$$

Therefore, the matching error is

$$\begin{aligned} Error &= \frac{1}{M-1}\sum_{p=1}^{M-1}[g_p'(i,j+(p-1)kW) - g_{p+1}'(i,j+pkW)]^2 \\ &\simeq \frac{1}{M-1}\sum_{p=1}^{M-1}[g_p'(i,j+(p-1)(kW+\delta_{i,j})) - g_{p+1}'(i,j+p(kW+\delta_{i,j})) \\ &\quad -(p-1)\delta_{i,j}g_p''(i,j+(p-1)(kW+\delta_{i,j})) \\ &\quad +p\,\delta_{i,j}g_{p+1}''(i,j+p(kW+\delta_{i,j}))]^2 \end{aligned} \tag{4.24}$$

When images are corrupted by additive white noise

$$\tilde{g}_p(i,j) = g_p(i,j) + n_p(i,j)$$

where $\tilde{g}_p(i,j)$ is the observation, the first order derivative of the intensity function is given by

$$\begin{aligned} \tilde{g}_p'(i,j+y) &= \sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d}{dy}\underline{\mathbf{CH}}(y)\,\tilde{g}_p(i,j+y') \\ &= g_p'(i,j+y) + \sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d}{dy}\underline{\mathbf{CH}}(y)\,n_p(i,j+y') \end{aligned} \tag{4.25}$$

$$for \quad -0.5 \leq y < 0.5.$$

and the second order derivative is given by

$$\begin{aligned}\tilde{g}_p''(i,j+y) &= \sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d^2}{dy^2}\underline{\mathbf{CH}}(y)\,\tilde{g}_p(i,j+y') \\ &= g_p''(i,j+y) + \sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d^2}{dy^2}\underline{\mathbf{CH}}(y)\,n_p(i,j+y') \quad (4.26)\end{aligned}$$

$$for \quad -0.5 \leq y < 0.5.$$

Replacing $g_p'(\cdot)$ and $g_p''(\cdot)$ in (4.24) by (4.25) and (4.26) and noting that

$$g_1'(i,j) = g_p'(i,j+(p-1)(kW+\delta_{i,j}))$$

and

$$g_1''(i,j) = g_p''(i,j+(p-1)(kW+\delta_{i,j}))$$

$$for \quad p = 2,3,...,M,$$

the matching error becomes

$$\begin{aligned}Error = \; & \frac{1}{M-1}\sum_{p=1}^{M-1}[\sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d}{dy}\underline{\mathbf{CH}}(y)|_{y=\bar{y}_{p-1}} n_p(i,j+y_{p-1}+y') \\ & -\sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d}{dy}\underline{\mathbf{CH}}(y)|_{y=\bar{y}_p} n_{p+1}(i,j+y_p+y') \\ & -(p-1)\delta_{i,j}\sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d^2}{dy^2}\underline{\mathbf{CH}}(y)|_{y=\bar{y}_{p-1}} n_p(i,j+y_{p-1}+y') \\ & +p\,\delta_{i,j}\sum_{y'\in\Omega} \underline{\mathbf{A}}^t(y')\,\frac{d^2}{dy^2}\underline{\mathbf{CH}}(y)|_{y=\bar{y}_p} n_{p+1}(i,j+y_p+y') \\ & +\delta_{i,j}\,g_1''(i,j)]^2 \quad (4.27)\end{aligned}$$

where

$$y_p = [pkW]$$

$$\bar{y}_p = pkW - y_p$$

and [ ] is a rounding operator. Assuming that noise $\{n_p(i,j)|p = 1,2,...,M\}$ and $\delta_{i,j}$ are mutually independent and noting the orthogonality of the polynomial set, the mean error can be simplified as

$$\begin{aligned}\mathbf{E}\{Error\} &= \frac{1}{M-1}\sum_{p=1}^{M-1}\{\sum_{n=1}^{4}\frac{1}{\sum_{v\in\Omega} Ch_n^2(v)}[\sigma_{n_p}^2(\frac{d}{dy}Ch_n(y))^2|_{y=y_{p-1}} + \\ &\sigma_{n_{p+1}}^2(\frac{d}{dy}Ch_n(y))^2|_{y=y_p} + \frac{W^2p^2\sigma_{n_{p+1}}^2}{12}(\frac{d^2}{dy^2}Ch_n(y))^2|_{y=y_p} + \\ &\frac{W^2(p-1)^2\sigma_{n_p}^2}{12}(\frac{d^2}{dy^2}Ch_n(y))^2|_{y=y_{p-1}}] + \frac{W^2}{12}(g_1''(i,j))^2\} \quad (4.28)\end{aligned}$$

Since only one camera is used, it is reasonable to assume that

$$\sigma_{n_1}^2 = \sigma_{n_p}^2 \quad for \;\; p = 2,...,M.$$

The mean error at (i,j) can be computed as

$$\mathbf{E}\{Error(i,j)_k\} = C_1\sigma_{n_1}^2 + C_2(g_1''(i,j))^2 \qquad (4.29)$$

where C_1 and C_2 are determined by

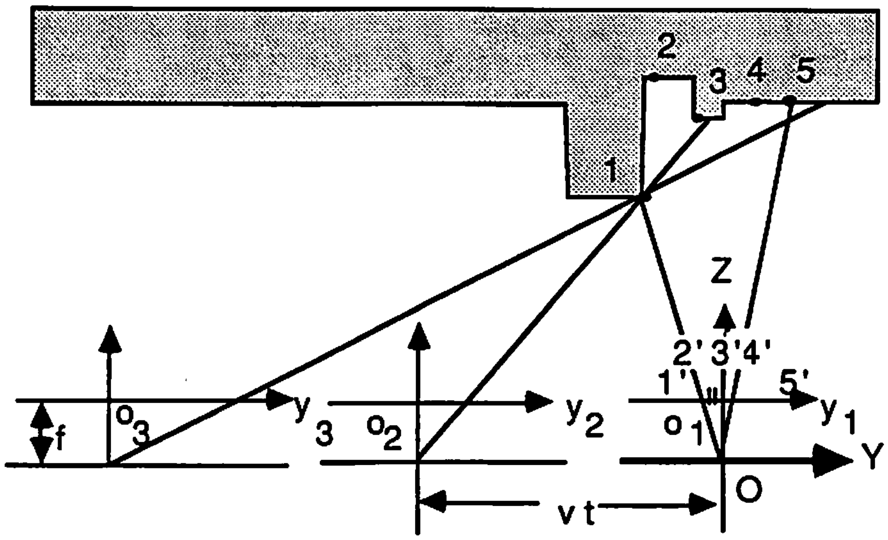
$$\begin{aligned}C_1 &= \frac{1}{M-1}\sum_{p=1}^{M-1}\{\sum_{n=1}^{4}\frac{1}{\sum_{v\in\Omega} Ch_n^2(v)}[(\frac{d}{dy}Ch_n(y))^2|_{y=y_{p-1}} + (\frac{d}{dy}Ch_n(y))^2|_{y=y_p}] \\ &+\frac{W^2}{12}\sum_{n=2}^{4}[p^2(\frac{d^2}{dy^2}Ch_n(y))^2|_{y=y_p} + (p-1)^2(\frac{d^2}{dy^2}Ch_n(y))^2|_{y=y_{p-1}}]\} \\ &and \\ C_2 &= \frac{W^2}{12}, \qquad (4.30)\end{aligned}$$

respectively. If the variance of noise and the second order derivatives of the intensity function are known or estimated from the images, then the mean error corresponding to the disparity value k at every point for a given ω (window size), M (frame number) and W (width of subsample interval) can be calculated.

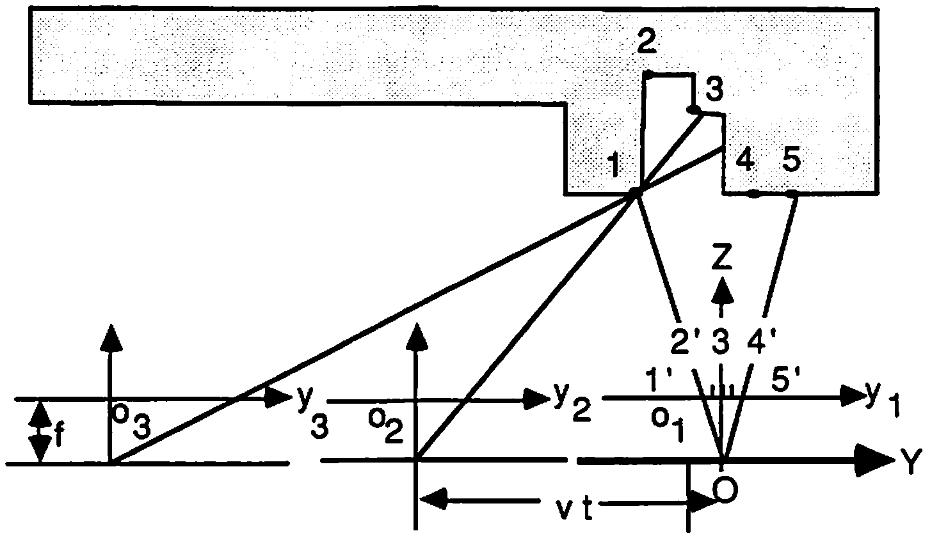
4.7 Detection of Occluding Pixels

Detection of occluding pixels is an important issue in motion stereo. As shown in Figure 4.2(a), when a camera moves from right to left, points 2, 3, 4 and 5 project into the first image plane at 2', 3', 4' and 5'. But points 2 and 3 on the object surface will not project into the second image plane because they are occluded by the front surface on which point 1 lies, Similarly, points 2, 3, 4 and 5 will not appear in the image plane 3. At the location of pixels 2', 3', 4' and 5', the match error is usually large which means no conjugate pixels can be found in the successive image frames. Hence, the disparity values are undetermined. Such pixels are called occluding pixels. When a smoothness constraint is used, although the matching algorithm always assign some values to the occluding pixels, the discontinuities of the disparity field may be shifted. As the number of frames increases, the number of occluding pixels also increases dramatically. For instance, if only two object points are occluded for the second image as shown in Figure 4.2(a), then about ten points are occluded for the sixth image which gives a ten pixel wide occluding region in the first image plane. On the other hand, if only the first two frames are used for matching, then pixels 4' and 5' are not occluding pixels and therefore the disparity values at such location are determinable. As the third frame does not provide any information about pixels 4' and 5', there is no need to update the bias inputs at the location of these pixels.

However, in some cases the number of occluding pixels does not increase as the number of frames increases. One typical example is illustrated in Figure 4.2(b), where pixels 4' and 5' are not occluding pixels when the third image frame is used. The above intuitive analysis essentially suggests a method for detecting occluding



(a) Pixels 2', 3' , 4' and 5' are occluding pixels.



(b) Pixels 4' and 5' are not occluding pixels.

Figure 4.2: Occluding pixels.

pixels. By using the mean values of matching error derived in the previous section the following detection rule can be used for detecting occluding pixels and hence we can prevent the RLS algorithm from updating the bias input at the location of occluding pixels.

<u>**Detection rule**</u>: An occluding pixel at location (i,j) is detected if

$$min(I_{i,j,k};\ 0 \leq k \leq D) > max(\mathbf{E}\{error(i,j)_k\};\ 0 \leq k \leq D) + b \tag{4.31}$$

where $b \geq 0$ is a constant for raising the threshold. When the noise variance is unknown, one can use a constant threshold instead of the mean error.

For the recursive approach, once an occluding pixel is detected, the bias inputs of neurons at such locations will not be updated anymore. But during the first iteration, the bias inputs of neurons at the locations of occluding pixels are first updated and then corrected accordingly. The correction procedure is as follows. From Figure 4.2 it can be seen that the pixels on the left side of the occluding region have high disparity values and the pixels on the right side have low disparity values. The width of the occluding region, i.e. the number of the occluding pixels is approximately given by

$$\hat{\Delta}_{i,j} = [\frac{D_{i,j-1} - D_{i,j+\Delta_{i,j}}}{W}] \tag{4.32}$$

where [ ] is a rounding operator, (i,j) denotes the location of the most left occluding pixel, $\Delta_{i,j}$ is the true width, $\hat{\Delta}_{i,j}$ is a estimate of the width, and $D_{i,j-1}$ and $D_{i,j+\Delta_{i,j}}$ are the disparity values of the nearest left and right nonoccluding pixels, respectively. Since a smoothness constraint is used, the occluding pixel usually takes either the high disparity value $D_{i,j-1}$ or low disparity value $D_{i,j+\Delta_{i,j}}$. The discontinuities of the disparity field can be detected by checking the disparity values in the y_1 direction for a transition from the high value to the low value.

Starting with the discontinuity pixel, we check all the left and right neighbor pixels. The search procedure will not be stopped until a $\hat{\Delta}_{i,j}$ wide or less occluding region including the discontinuity pixel is found. Then, for all occluding pixels the bias input of the $\Delta_{i,j}$th neuron is corrected by

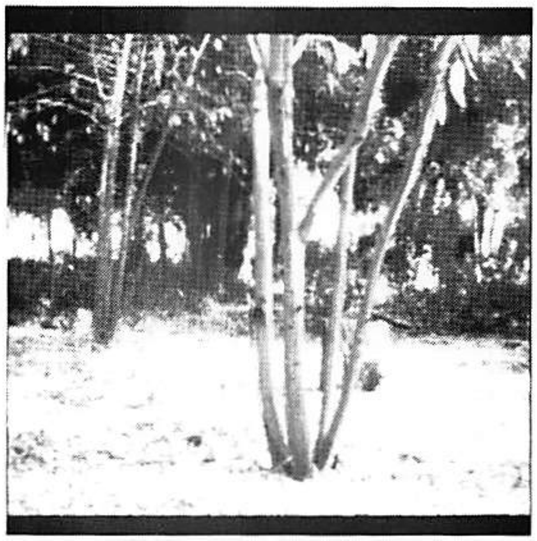
$$I_{i,l,D_{i,j+\Delta_{i,j}}}(1) = min(I_{i,l,k}(1);\; k = 0, 1, ..., D). \tag{4.33}$$

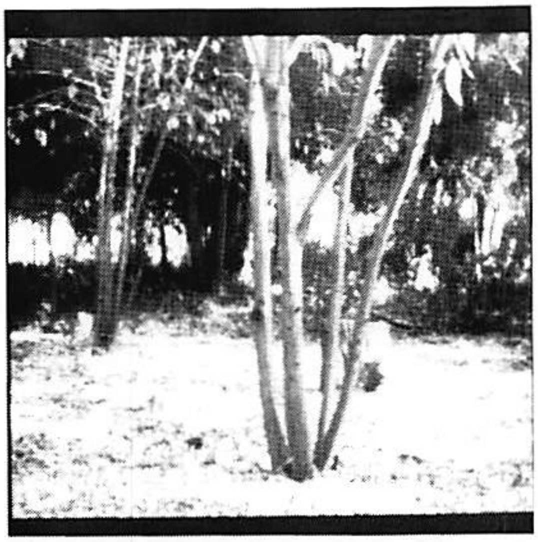
For the batch approach, the bias inputs at occluding pixels are estimated using only the first two image frames.

4.8 Experimental Results

We have tested both the batch and recursive algorithms on a sequence of 128 natural images taken by sliding a camera from right to left. The images are of size 256 × 233. We arbitrarily chose four successive frames (frames 61, 62, 63 and 64) shown in Figure 4.3 for testing, although there is no limit to the number of frames. No alignment in the vertical direction was made and the maximum disparity, about 2 pixels, was measured by hand. Same parameters were chosen for both algorithms. The subpixel width W was set at 0.2 and hence $D = 10$. The parameter λ was set at 7. The threshold for occluding pixel detection was set at 250 because the noise variance is unknown. Figure 4.4 (a) shows the batch result after 49 iterations. The disparity map is represented by an intensity image. Figure 4.4 (b) is a smooth version of Figure 4.4 (a) using a 5×5 mean filter. The recursive result is shown in Figure 4.4 (c) and its smooth version is in Figure 4.4 (d). The iterations for the recursive solution are 38.

(a) Frame 61.


(b) Frame 62.


(c) Frame 63.

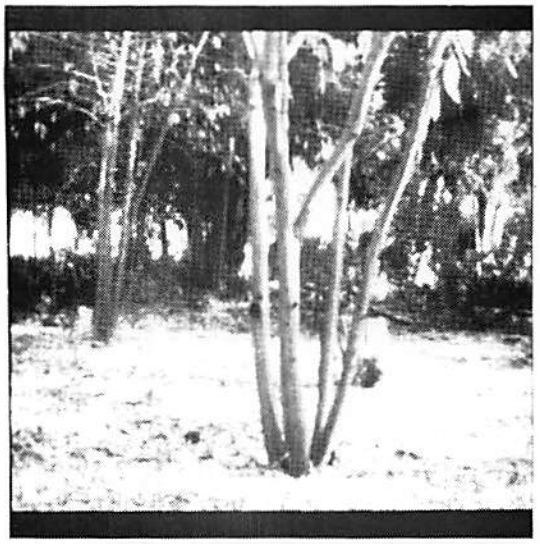
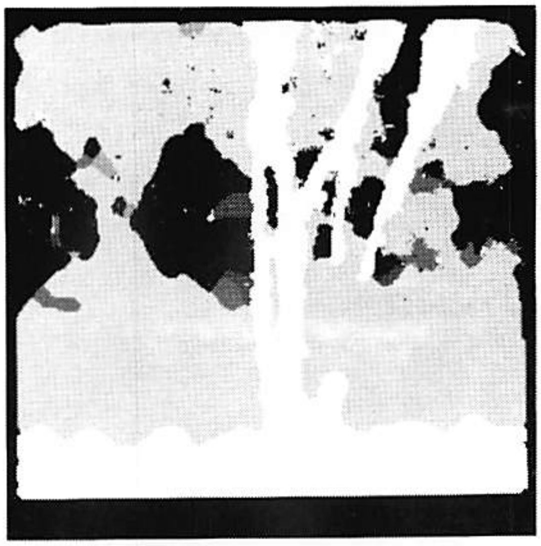
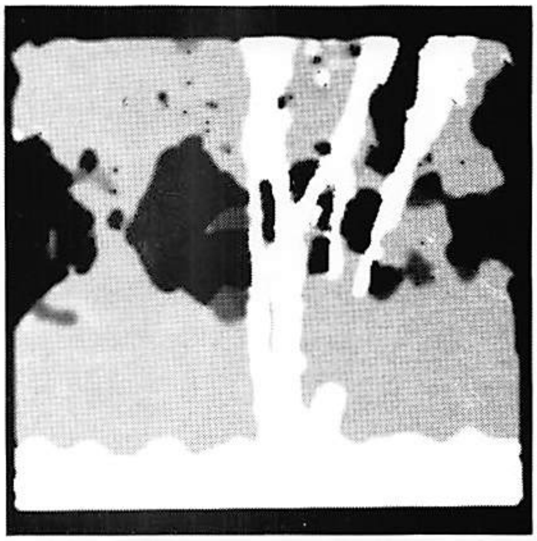

(d) Frame 64.

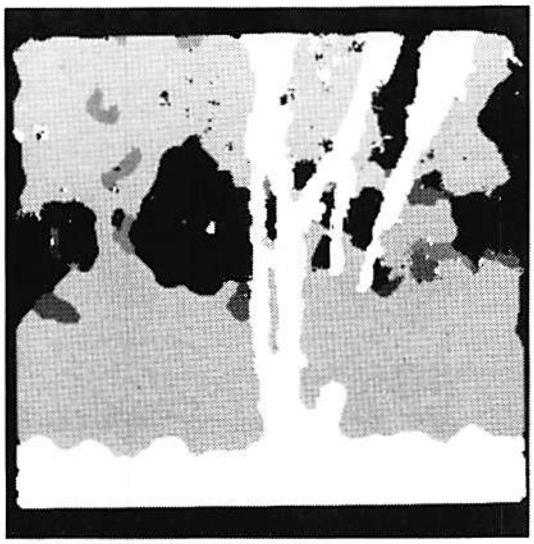
Figure 4.3: The Trees images.



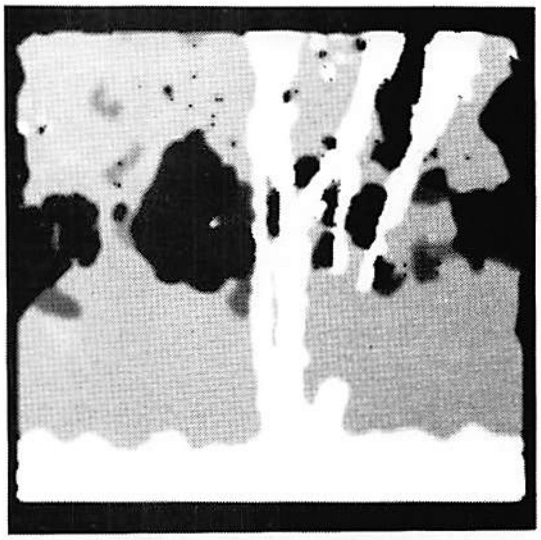
(a) Batch approach.



(b) Smoothed version of (a).



(c) Recursive approach.



(d) Smoothed version of (c).

Figure 4.4: Disparity maps.

4.9 Discussion

We have presented two neural network based approaches, known as the batch and recursive approaches, for motion stereo using the first order derivatives of intensity function as measurement primitives. For the recursive approach, the bias inputs of the neurons are recursively updated and if the intermediate results are not required the matching procedure is implemented only once. Unlike the existing recursive approach, the disparity field obtained by this approach is smooth and dense. Also no batch results are needed for setting the initial states of the neurons. Both batch and recursive methods gave very good results in comparison to Barnard's approach [45]. Experimental results show that the recursive approach needs fewer iterations than the batch approach. This is because the recursive approach uses a better bias input updating scheme especially for the occluding pixels. The good estimate of the bias inputs makes the network converge fast, although the updating step for bias inputs takes more computations. In view of parallelism and fast convergence, the recursive approach is useful for real time implementation, such as in a robot vision system. In our experiment, the threshold used was 250 which seems a little bit conservative. However, the maximum disparity is only about 2 pixels which means that the width of the occluding region is less than 2 pixels for two frames and there are only a few occluding pixels along the right boundaries of the trees. Hence the occluding pixels do not cause a serious problem in this experiment. This is also why the iteration number does not reduce a lot. We believe that if the maximum disparity is large and a long sequence of images is used, then the improvement on the occluding pixel detection will greatly reduce the number of iterations.

Chapter 5

Computation of Optical Flow

5.1 Introduction

Optical flow is the distribution of apparent velocities of motion brightness patterns in an image. Except for some special cases [64], optical flow corresponds to the motion field. Hence, computing the optical flow from sequential images gives important information about the spatial arrangement of the objects and the rate of change of this arrangement in a given scene. Optical flow can thus be used for segmenting images into regions and estimating the object motion in the scene. In this chapter we present a method for computing the optical flow based on rotation invariant measured primitives extracted from two successive image frames. An implementation using a neural network is also given.

Existing approaches to computation of optical flow can be divided into two types: image intensity based or token based approaches. The intensity based approach basically relies on the assumption that the changes in intensity are strictly due to the motion of the object and use the image intensity values and their spatial and temporal derivatives to compute the optical flow. Limb and Murphy [72] utilize the relation between spatial and temporal differential signals to estimate the displacement of an object in a television scene under the assumption that the displacement is constant within a block of image pixels. In order to make the constant displacement assumption more realistic to situations in which there are multiple moving objects, occluding objects and different parts of the same object moving with different displacements in scene, Netravali and Robbins [73] proposed a recursive algorithm for estimating the displacements of multiple moving objects in a television scene by minimizing a functional of the prediction error. Since the update at each pixel is based on gradient information at a single point, the method is more sensitive to noise than Limb's block approach. Subsequently, they extended their method by using the gradient information at a number of pixels in the neighborhood of the point [74]. Fennema and Thompson [75] developed a nonmatching method for computing the speed and direction of the one or more moving objects in a scene using a clustering technique based on spatial and temporal derivatives of the intensity. To determine the actual velocity, a modification of the Hough Transform was used. All these methods are applicable to rigid body translation only. Expanding the intensity function as a first order Taylor series, Horn and Schunck [76] derived an optical flow equation using brightness constancy and spatial smoothness constraints. An iterative method for solving the resulting equation was developed. This method usually fails to detect discontinuous locations of the velocity field due to oversmoothing and large displacements due to first order approximation. Recently, Gennert and Negahdaripour [77] have proposed to replace the brightness constancy constraint in the optical flow equation with a more general constraint, which permits a linear transformation between the image intensity values.

The token based approach is to consider the motion of tokens such as edges, corners and linear features in an image. The main reason to consider a token based approach is that the tokens are less sensitive to some of the difficulties associated with variations of the image intensity. Thompson and Barnard [78] developed a probabilistic relaxation labeling scheme for computing displacements based on corners, spots and similar structures in images. Hildreth [79] proposed a method based on the zero-crossing contours. Nagel [80] investigated an approach to track corners and estimate the displacement at the corners by using partial derivatives of the intensity function. Under the assumption of smooth, rigid body motion and nonspare distribution of feature points on a surface, Prager and Arbib [81] developed a MATCH algorithm for computing the optical flow. However, the token based approaches give information about object motion and shape only at edges, corners and linear features. An interpolation procedure has to be included when dense flow field is required.

Recently, several researchers have used neural network for computing optical flow based on either intensity values or tokens [82, 13]. In [82], a neural network proposed by Hopfield and Tank [6] was used to realize the Ullman's Minimal Mapping Theory [83] for computing the optical flow based on feature points of a nonrigid moving object. They also use the same neural network to implement the structural theory for solving correspondence and structure simultaneously for the rigid motion problems. Despite some mismatches, fast convergence of the neural network was always obtained. A similar analog neural network was used in [13] for computing the optical flow. To prevent the smoothness constraint from taking effect across strong velocity gradients, Koch proposed to incorporate a line process into the optical flow equation [13]. However, no attempt was made to detect large displacements. The resulting equation involving cubic and possibly higher terms is nonconvex. Instead of using simulated annealing algorithm which is very time consuming, a deterministic algorithm based on a mixed analog and digital network is used to obtain a suboptimal solution. As reported in [84], convergence of such a network was obtained within a couple of analog-digital cycles. Basically, the analog-digital network approach is the first to use Horn's optical flow equation to find a smoothest solution and then to update the line process by lowering the energy function of the network repeatedly. Several impressive examples using synthetic and video images were presented in [84].

In order to obtain a dense optical flow field, it seems that the intensity based approach is preferable. However, the intensity value may be corrupted by noise and also their partial derivatives are rotation variant. It is difficult to detect rotating objects in natural images, based on such measurement primitives. Under the assumption that the changes in intensity are strictly due to object motion, we may use rotation invariant principle curvatures of the intensity function to compute the optical flow. In this chapter, we use a neural network with maximum evolution function to compute optical flow based on the intensity values and their principle curvatures under local rigid motion assumption and smoothness constraints. For detecting motion discontinuities, a line process is commonly used [13] for locating motion discontinuities. However, without exactly knowing the occluding elements, the detected discontinuities may be shifted. In order to detect discontinuities accurately, we first detect the occluding elements from the initial motion measurements, then use a line process to locate the discontinuities by using the information about the detected occluding elements. The intensity values and their principle curvatures are estimated with subpixel accuracy by using a polynomial fitting technique. When a large window is applied, smooth estimates can be obtained from noisy observations. In order to ensure convergence of the network, deterministic decision rules are used. Since the neurons and lines are updated in parallel, this algorithm can be implemented in real time. To improve the accuracy, a batch approach using multiple frames is also presented. Experimental results using synthetic and natural images are given.

5.2 Estimation of Intensity Values and Principle Curvatures

In order to reduce the effects of noise and quantization error, a bivariate polynomial fitting technique is used for the estimation of the intensity values and their principle curvatures from discrete observations. The estimation problem can then be formulated to find polynomial coefficients such that the square error

$$\epsilon_{i,j} = \sum_{x\in\Omega} \sum_{y\in\Omega} (\tilde{g}(i+x, j+y) - \hat{g}(i+x, j+y))^2 \tag{5.1}$$

between the estimate $\hat{g}(i,j)$ and the observation $\tilde{g}(i,j)$ is minimized. In (5.1), Ω is an index set $\{-\omega, -\omega+1, ..., \omega-1, \omega\}$.

5.2.1 Estimation of Polynomial Coefficients

Bivariate discrete Chebyshev polynomials can be constructed by using the tensor product technique. We assume that in each neighborhood of an image point (i,j) the underlying intensity function can be approximated by a fourth order polynomial and represent the intensity function in a window of size $2\omega+1$ by

$2\omega + 1$ is by

$$\hat{g}(i+x, j+y) = \underline{\mathbf{CH}}^t(x)\, \underline{\mathbf{Q}}\, \underline{\mathbf{CH}}(y) \tag{5.2}$$

where $\hat{g}(i+x, j+y)$ is the approximated continuous intensity function, t denotes the transpose operator,

$$\underline{\mathbf{CH}}^t(x) = [Ch_0(x)\; Ch_1(x)\; Ch_2(x)\; Ch_3(x)\; Ch_4(x)]$$

and

$$\underline{\mathbf{CH}}^t(y) = [Ch_0(y)\; Ch_1(y)\; Ch_2(y)\; Ch_3(y)\; Ch_4(y)]$$

are polynomial vectors and

$$\underline{\mathbf{Q}} = \begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,0} & a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix}$$

is the coefficient matrix. By minimizing the square error (5.1) and taking advantage of the orthogonality of the polynomial set, the coefficients $\{a_{m,n}\}$ are obtained as

$$a_{m,n} = \frac{\sum_{y\in\Omega} \sum_{x\in\Omega} Ch_m(x)\, Ch_n(y)\, g(i+x, j+y)}{\sum_{u\in\Omega} \sum_{v\in\Omega} Ch_m^2(u)\, Ch_n^2(v)} \tag{5.3}$$

where $\{g(i+x, j+y)\}$ are the observed intensity values. Once the coefficients are calculated, the subpixel intensity values can be estimated from (5.2). A smooth intensity function can be obtained if low order polynomials and large windows are used.

5.2.2 Computing Principle Curvatures

The principle curvatures at (i, j) are defined as the maximum and minimum values of the normal curvatures of intensity function at that point [85]. A notable property of principle curvature is that they are rotation invariant which is useful for detecting rotating objects. The principle curvatures can be expressed in terms of Gaussian and mean curvatures of intensity function as follows [85]

$$k_1(i,j) = \mathrm{M} + (\mathrm{M}^2 - \mathrm{G})^{\frac{1}{2}} \tag{5.4}$$

and

$$k_2(i,j) = \mathrm{M} - (\mathrm{M}^2 - \mathrm{G})^{\frac{1}{2}} \tag{5.5}$$

where $k_1(i,j)$ and $k_2(i,j)$ are the principle curvatures, G and M are the Gaussian and mean curvatures defined as

$$\mathrm{G} = \frac{\partial^2 g(i,j)}{\partial i^2} \frac{\partial^2 g(i,j)}{\partial j^2} - (\frac{\partial^2 g(i,j)}{\partial i \partial j})^2$$

and

$$\mathrm{M} = \frac{1}{2}(\frac{\partial^2 g(i,j)}{\partial i^2} + \frac{\partial^2 g(i,j)}{\partial j^2}),$$

respectively, under the assumption that

$$\frac{\partial^2 g(i,j)}{\partial i \partial j} = \frac{\partial^2 g(i,j)}{\partial j \partial i}.$$

Using the estimated continuous intensity function, the second order partial derivatives of the intensity function can be calculated with subpixel accuracy as follows

$$\begin{aligned}
\frac{\partial^2 g(i,j)}{\partial i^2}|_{i=i+x,j=j+y} &= \frac{\partial^2 \hat{g}(i+x,j+y)}{\partial x^2} = \frac{d^2}{dx^2}(\underline{\mathrm{CH}}^t(x))\,\underline{\mathrm{Q}}\,\underline{\mathrm{CH}}(y) \\
\frac{\partial^2 g(i,j)}{\partial i \partial j}|_{i=i+x,j=j+y} &= \frac{\partial^2 \hat{g}(i+x,j+y)}{\partial x \partial y} = \frac{d}{dx}(\underline{\mathrm{CH}}^t(x))\,\underline{\mathrm{Q}}\,\frac{d}{dy}(\underline{\mathrm{CH}}(y)) \\
\frac{\partial^2 g(i,j)}{\partial j^2}|_{i=i+x,j=j+y} &= \frac{\partial^2 \hat{g}(i+x,j+y)}{\partial y^2} = \underline{\mathrm{CH}}^t(x)\,\underline{\mathrm{Q}}\,\frac{d^2}{dy^2}(\underline{\mathrm{CH}}(y))
\end{aligned} \tag{5.6}$$

$$for \quad -0.5 \leq x, y < 0.5$$

For simplicity of notation, we use $g_{xx}(x,y)$, $g_{xy}(x,y)$ and $g_{yy}(x,y)$ to represent the second order partial derivatives of the subpixel intensity function. After substitution of the second order partial derivatives, **M**, (5.4) and (5.5) become

$$\begin{aligned} k_1(i+x, j+y) &= \frac{1}{2}\{[(g_{xx}(x,y) - g_{yy}(x,y))^2 + 4g_{xy}^2(x,y)]^{\frac{1}{2}}\} \\ &\quad + g_{xx}(x,y) + g_{yy}(x,y)\} \end{aligned} \tag{5.7}$$

and

$$k_2(i+x, j+y) = g_{xx}(x,y) + g_{yy}(x,y) - k_1(i+x, j+y) \tag{5.8}$$

$$for \quad -0.5 \leq x, y < 0.5.$$

Therefore, the principle curvatures of the subpixel intensity function can be obtained from the subpixel partial derivatives directly.

5.2.3 Analysis of Filters

Rewrite the coefficient matrix as

$$\underline{Q} = \sum_{x' \in \Omega} \sum_{y' \in \Omega} \underline{Q}(x', y')\, g(i+x', j+y') \tag{5.9}$$

where $\underline{Q}(x', y')$ is a matrix with the (m, n)th element

$$\frac{Ch_m(x')\, Ch_n(y')}{\sum_{u \in \Omega} \sum_{v \in \Omega} Ch_m^2(u)\, Ch_n^2(v)}.$$

By replacing the matrix $\underline{Q}$ in (5.2) with (5.9) we have

$$\hat{g}(i+x, j+y) = \sum_{x' \in \Omega} \sum_{y' \in \Omega} F(x, y; x', y')\, g(i+x', j+y') \tag{5.10}$$

where $F(x, y; x', y')$ is determined by

$$F(x, y; x', y') = \underline{\mathbf{CH}}^t(x)\, \underline{Q}(x', y')\, \underline{\mathbf{CH}}(y). \tag{5.11}$$

Similarly, (5.6) can be written as

$$\begin{aligned}
\frac{\partial^2 g(i,j)}{\partial i^2}\Big|_{i=i+x,j=j+y} &= \sum_{x'\in\Omega}\sum_{y'\in\Omega} F_{xx}(x,y;x',y')\, g(i+x',j+y') \\
\frac{\partial^2 g(i,j)}{\partial i \partial j}\Big|_{i=i+x,j=j+y} &= \sum_{x'\in\Omega}\sum_{y'\in\Omega} F_{xy}(x,y;x',y')\, g(i+x',j+y') \qquad (5.12) \\
\frac{\partial^2 g(i,j)}{\partial j^2}\Big|_{i=i+x,j=j+y} &= \sum_{x'\in\Omega}\sum_{y'\in\Omega} F_{yy}(x,y;x',y')\, g(i+x',j+y')
\end{aligned}$$

where

$$\begin{aligned}
F_{xx}(x,y;x',y') &= \frac{d^2}{dx^2}(\underline{\mathbf{CH}}^t(x))\, \underline{\mathbf{Q}}\, \underline{\mathbf{CH}}(y) \\
F_{xy}(x,y;x',y') &= \frac{d}{dx}(\underline{\mathbf{CH}}^t(x))\, \underline{\mathbf{Q}}\, \frac{d}{dy}(\underline{\mathbf{CH}}(y)) \qquad (5.13) \\
F_{yy}(x,y;x',y') &= \underline{\mathbf{CH}}^t(x)\, \underline{\mathbf{Q}}\, \frac{d^2}{dy^2}(\underline{\mathbf{CH}}(y))
\end{aligned}$$

are the corresponding filters.

It is assumed that the image is corrupted by additive white noise with zero mean and variance σ_n^2

$$\tilde{g}(i,j) = g(i,j) + n(i,j). \qquad (5.14)$$

where $\tilde{g}(i,j)$ and $g(i,j)$ are the observed and original intensity functions, respectively. The expected value of the output of the filter $F(x,y;x',y')$ can be

$$\begin{aligned}
&\mathbf{E}\{\sum_{x'\in\Omega}\sum_{y'\in\Omega} F(x,y;x',y')\, \tilde{g}(i+x',j+y')\} \\
&= \sum_{x'\in\Omega}\sum_{y'\in\Omega} F(x,y;x',y')\, g(i+x',j+y'). \qquad (5.15)
\end{aligned}$$

Accordingly, the variance is given by

$$\begin{aligned}
&\mathbf{Var}(\sum_{x'\in\Omega}\sum_{y'\in\Omega} F(x,y;x',y')\tilde{g}(i+x',j+y')) \\
&= \mathbf{E}\{(\sum_{x'\in\Omega}\sum_{y'\in\Omega} F(x,y;x',y')\, n(i+x',j+y'))^2\} \\
&= \sigma_n^2 \sum_{x'\in\Omega}\sum_{y'\in\Omega} F^2(x,y;x',y') \qquad (5.16)
\end{aligned}$$

By using (5.11) and noting the orthogonality of the polynomial set, it is straightforward to show that

$$\begin{aligned}
&\sum_{x'\in\Omega}\sum_{y'\in\Omega} F^2(x,y;x',y') \\
&= \sum_{x'\in\Omega}\sum_{y'\in\Omega} (\underline{\mathbf{CH}}^t(x)\,\underline{\mathbf{Q}}(x',y')\,\underline{\mathbf{CH}}(y))^2 \\
&= \sum_{m=0}^{4} \frac{Ch_m^2(x)}{\sum_{u\in\Omega} Ch_m^2(u)} \sum_{n=0}^{4} \frac{Ch_n^2(y)}{\sum_{v\in\Omega} Ch_n^2(v)}
\end{aligned} \tag{5.17}$$

Hence, the variance of the filter output is

$$\begin{aligned}
&\mathbf{Var}(\sum_{x'\in\Omega}\sum_{y'\in\Omega} F(x,y;x',y')\tilde{g}(i+x',j+y')) \\
&= \sigma_n^2 \sum_{m=0}^{4} \frac{Ch_m^2(x)}{\sum_{u\in\Omega} Ch_m^2(u)} \sum_{n=0}^{4} \frac{Ch_n^2(y)}{\sum_{v\in\Omega} Ch_n^2(v)}
\end{aligned} \tag{5.18}$$

Similarly, beginning with (5.13), one can show that the variances of the outputs of the filters $F_{xx}(x,y;x',y')$, $F_{xy}(x,y;x',y')$ and $F_{yy}(x,y;x',y')$ are

$$\begin{aligned}
&\mathbf{Var}(\sum_{x'\in\Omega}\sum_{y'\in\Omega} F_{xx}(x,y;x',y')\tilde{g}(i+x',j+y')) \\
&= \sigma_n^2 \sum_{m=2}^{4} \frac{(\frac{d^2}{dx^2}Ch_m(x))^2}{\sum_{u\in\Omega} Ch_m^2(u)} \sum_{n=0}^{4} \frac{Ch_n^2(y)}{\sum_{v\in\Omega} Ch_n^2(v)} \\
&\mathbf{Var}(\sum_{x'\in\Omega}\sum_{y'\in\Omega} F_{xy}(x,y;x',y')\tilde{g}(i+x',j+y')) \\
&= \sigma_n^2 \sum_{m=1}^{4} \frac{(\frac{d}{dx}Ch_m(x))^2}{\sum_{u\in\Omega} Ch_m^2(u)} \sum_{n=1}^{4} \frac{(\frac{d}{dy}Ch_n(y))^2}{\sum_{v\in\Omega} Ch_n^2(v)} \\
&\mathbf{Var}(\sum_{x'\in\Omega}\sum_{y'\in\Omega} F_{yy}(x,y;x',y')\tilde{g}(i+x',j+y')) \\
&= \sigma_n^2 \sum_{m=0}^{4} \frac{Ch_m^2(x)}{\sum_{u\in\Omega} Ch_m^2(u)} \sum_{n=2}^{4} \frac{(\frac{d^2}{dy^2}Ch_n(y))^2}{\sum_{v\in\Omega} Ch_n^2(v)}
\end{aligned} \tag{5.19}$$

It is clear from the above equations that the variances become smaller and smaller as the window size increases. For instance, when the window size is 5, the variance of the output of the filter $F(x,y;x',y')$ at $(x,y)=(0,0)$ is σ_n^2. When the window size is 11, the variance is significantly reduced to $0.044\sigma^2$. As pointed out in Chapter 3, large window causes some loss of local information due to smoothing which smears or erases local features. If one desires to retain the local features, then a small window may be used, but more noise remains and the estimated intensity function, their derivatives and hence their principle curvatures are less accurate.

5.3 Computing Optical Flow

5.3.1 Neural Network Formulation

The neural network consists of $N_r \times N_c \times (2D_i+1) \times (2D_j+1)$ mutually interconnected neurons, where N_r and N_c are the image row and column sizes, respectively. $D_i\ W$ and $D_j\ W$ represent the maximum velocities in the i and j direction, where W is the width of subsample interval. Let $V = \{v_{i,j,k,l}, 1 \leq i \leq N_r, 1 \leq j \leq N_c,\ -D_i \leq k \leq\ D_i,\ -D_j \leq l \leq\ D_j\}$ be a binary state set of the neural network with $v_{i,j,k,l}$ denoting the state of the (i,j,k,l)th neuron. When the neuron $v_{i,j,k,l}$ is 1, this means that the velocities in i and j directions at the point (i,j) are $k\ W$ and $l\ W$, respectively. Every point is represented by $(2D_i+1)\times(2D_j+1)$ mutually exclusive neurons and only one neuron is on and others are off. Let $T_{i,j,k,l;m,n,p,q}$ denote the strength (possibly negative) of the interconnection between neuron (i,j,k,l) and neuron (m,n,p,q). It is assumed that the interconnections are symmetric and the neurons have self-feedback. Each neuron (i,j,k,l) synchronously receives inputs from all the neurons and a bias input

$$u_{i,j,k,l} = \sum_{m=1}^{N_r} \sum_{n=1}^{N_c} \sum_{p=-D_i}^{D_i} \sum_{q=-D_j}^{D_j} T_{i,j,k,l;m,n,p,q} v_{m,n,p,q} + I_{i,j,k,l} \tag{5.20}$$

Each $u_{i,j,k,l}$ is fed back to corresponding neurons after maximum evolution

$$v_{i,j,k,l} = g(u_{i,j,k,l}) \tag{5.21}$$

where $g(x_{i,j,k,l})$ is a nonlinear maximum evolution function

$$g(x_{i,j,k,l}) = \begin{cases} 1 & if \;\; x_{i,j,k,l} = max(x_{i,j,p,q};\; -D_i \leq p \leq D_i,\; -D_j \leq q \leq D_j). \\ 0 & otherwise. \end{cases} \tag{5.22}$$

The uniqueness of the problem is ensured by the maximum evolution function.

Without adding any physical constraints to the solution, optical flow computed from a pair of image frames usually is noisy and inaccurate. For instance, although the correlation method may provide a solution based on the local match without any smoothness constraint, the resulting optical flow is not accurate and local error is undetectable. In our algorithm, a smoothness constraint is used for obtaining a smooth optical flow field and a line process is employed for detecting motion discontinuities. The line process consists of vertical and horizontal lines, L^v and L^h. Each line can be in either one of two states: 1 for acting and 0 for resting. The error function for computing the optical flow can be properly expressed as

$$\begin{aligned} E \;=\; & \sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=-D_i}^{D_i}\sum_{l=-D_j}^{D_j} \{A\,[(k_{11}(i,j) - k_{21}(i \oplus k, j \oplus l))^2 \\ & +(k_{12}(i,j) - k_{22}(i \oplus k, j \oplus l))^2] + (g_1(i,j) - g_2(i \oplus k, j \oplus l))^2\} v_{i,j,k,l} \\ & +\frac{B}{2}\sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=-D_i}^{D_i}\sum_{l=-D_j}^{D_j}\sum_{s \in S}(v_{i,j,k,l} - v_{(i,j)\oplus s,k,l})^2 \\ & +\sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=-D_i}^{D_i}\sum_{l=-D_j}^{D_j} \{\frac{C}{2}[(v_{i,j,k,l} - v_{i\oplus 1,j,k,l})^2(1 - L^h_{i,j,k,l}) \\ & +(v_{i,j,k,l} - v_{i,j\oplus 1,k,l})^2(1 - L^v_{i,j,k,l})] + D(L^h_{i,j,k,l} + L^v_{i,j,k,l})\} \end{aligned} \tag{5.23}$$

where $k_{11}(i,j)$ and $k_{12}(i \oplus k, j \oplus l)$ are the principle curvatures of the first image, $k_{21}(i,j)$ and $k_{22}(i \oplus k, j \oplus l)$ are the principle curvatures of the second image, $\{g_1(i,j)\}$ and $\{g_2(i \oplus k, j \oplus l)\}$ are the intensity values of the first and second images, respectively, S is an index set excluding $(0,0)$ for all neighbors in a $\Gamma \times \Gamma$ window centered at point (i,j), A, B, C and D are constants and the symbol $\oplus$ is the same as that defined in (3.29). The first term in (5.23) is to seek velocity values such that all points of two images are matched as closely as possible in a least squares sense. The second term is the smoothness constraint on the solution and the third term is a line process to weaken the smoothness constraint and to detect motion discontinuities. The constant B, C and D determine the relative importance of the three terms and the constant A in the first term determines the relative importance of the intensity values and their principle curvatures to achieve the best results.

By choosing the interconnection strengths and bias inputs as

$$\begin{aligned} T_{i,j,k,l;m,n,p,q} &= -[48B + C(4 - L^h_{i,j,k,l} - L^h_{i,j\oplus(-1),k,l} - L^v_{i,j,k,l} \\ &\quad - L^v_{i\oplus(-1),j,k,l})]\delta_{i,m}\delta_{j,n}\delta_{k,p}\delta_{l,q} + C[(1 - L^h_{i,j,k,l})\delta_{i,m}\delta_{j\oplus 1,n} \\ &\quad + (1 - L^v_{i,j,k,l})\delta_{i\oplus 1,m}\delta_{j,n} + (1 - L^h_{i,j\oplus(-1),k,l})\delta_{i,m}\delta_{j\oplus(-1),n} \\ &\quad + (1 - L^v_{i\oplus(-1),j,k,l})\delta_{i\oplus(-1),m}\delta_{j,n}]\delta_{k,p}\delta_{l,q} \\ &\quad + 2B\sum_{s\in S} \delta_{(i,j),(m,n)\oplus s}\delta_{k,p}\delta_{l,q} \end{aligned} \tag{5.24}$$

and

$$\begin{aligned} I_{i,j,k,l} &= -A[(k_{11}(i,j) - k_{21}(i \oplus k, j \oplus l))^2 + (k_{12}(i,j) - k_{22}(i \oplus k, j \oplus l))^2] \\ &\quad -(g_1(i,j) - g_2(i \oplus k, j \oplus l))^2 \end{aligned} \tag{5.25}$$

where $\delta_{a,b}$ is the Dirac delta function, and ignoring the term $D(L^h_{i,j,k,l} + L^v_{i,j,k,l})$ (which does not contain neurons $v_{i,j,k,l}$), the error function (5.23) is transformed into

$$\begin{aligned} E &= -\frac{1}{2}\sum_{i=1}^{N_r}\sum_{m=1}^{N_r}\sum_{j=1}^{N_c}\sum_{n=1}^{N_c}\sum_{k=-D_i}^{D_i}\sum_{p=-D_i}^{D_i}\sum_{l=-D_j}^{D_j}\sum_{q=-D_j}^{D_j} T_{i,j,k,l;m,n,p,q}\, v_{i,j,k,l}\, v_{m,n,p,q} \\ &\quad -\sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=-D_i}^{D_i}\sum_{l=-D_j}^{D_j} I_{i,j,k,l}\, v_{i,j,k,l} \qquad (5.26) \end{aligned}$$

which is same as (2.7), the energy function of the neural network. Therefore, computation of optical flow can be carried out by neuron evaluation. The size of the smoothing window used in (5.24) is 5.

5.3.2 Updating Schemes

However, the energy function (5.26) does not include the term $D(L^h_{i,j,k,l} + L^v_{i,j,k,l})$. If (5.20) and (5.21) are used to update the neurons, the lines can not be updated properly. It is necessary to update the neurons and lines separately.

Each neuron synchronously evaluates its state and readjustes according to (5.20) and (5.21). The synchronous updating scheme can be implemented in parallel.

The initial state of the neurons is set as

$$v_{i,j,k,l} = \begin{cases} 1 & if \;\; I_{i,j,k,l} = max(I_{i,j,p,q}; -D_i \le p \le D_i, -D_j \le q \le D_j). \\ 0 & otherwise \end{cases} \qquad (5.27)$$

where $I_{i,j,k,l}$ is the bias input. If there are two maximal bias inputs at point (i, j), then only the neuron corresponding to the smallest velocity is initially set at 1 and the other one is set at 0. This is consistent with the minimal mapping theory [83]. In the updating scheme, we also use the minimal mapping theory to handle the case of two neurons having the same largest inputs.

Again, self-feedback may increase the energy function E after a transition. For two neurons $v_{i,j,k,l}$ and $v_{i,j,k',l'}$ changing their states, let the state changes be denoted as

$$\Delta v_{i,j,k,l} = v_{i,j,k,l}^{new} - v_{i,j,k,l}^{old}$$

$$\Delta v_{i,j,k',l'} = v_{i,j,k',l'}^{new} - v_{i,j,k',l'}^{old}$$

and accordingly the energy change ΔE is

$$\begin{aligned}
\Delta E &= E^{new} - E^{old} \\
&= -(\sum_{m=1}^{N_r} \sum_{n=1}^{N_c} \sum_{p=-D_i}^{D_i} \sum_{q=-D_j}^{D_j} T_{i,j,k,l;m,n,p,q}\, v_{m,n,p,q} + I_{i,j,k,l})\Delta v_{i,j,k,l} \\
&\quad -(\sum_{m=1}^{N_r} \sum_{n=1}^{N_c} \sum_{p=-D_i}^{D_i} \sum_{q=-D_j}^{D_j} T_{i,j,k',l';m,n,p,q}\, v_{m,n,p,q} + I_{i,j,k',l'})\, \Delta v_{i,j,k',l'} \\
&\quad -\frac{1}{2} T_{i,j,k,l;i,j,k,l}\, (\Delta v_{i,j,k,l})^2 - \frac{1}{2} T_{i,j,k',l';i,j,k',l'}\, (\Delta v_{i,j,k',l'})^2 \\
&\quad -T_{i,j,k,l;i,j,k',l'}(\Delta v_{i,j,k,l} v_{i,j,k',l'}^{new} + \Delta v_{i,j,k',l'} v_{i,j,k,l}^{new})
\end{aligned} \tag{5.28}$$

When

$$v_{i,j,k,l}^{old} = 0, \qquad v_{i,j,k',l'}^{old} = 1$$

and

$$u_{i,j,k',l'} < u_{i,j,k,l},$$

we have

$$v_{i,j,k,l}^{new} = 1, \qquad v_{i,j,k',l'}^{new} = 0,$$

and

$$\Delta E = (u_{i,j,k',l'} - u_{i,j,k,l}) - \frac{1}{2}\, (T_{i,j,k,l;i,j,k,l} + T_{i,j,k',l';i,j,k',l'}). \tag{5.29}$$

Since the first term in (5.29) is negative and the second term $-\frac{1}{2}\ (T_{i,j,k,l;i,j,k,l} + T_{i,j,k',l';i,j,k',l'})$ is

$$48\ B + C\ (4 - L_{i,j,k,l}^h - L_{i,j\oplus(-1),k,l}^h - L_{i,j,k,l}^v - L_{i\oplus(-1),j,k,l}^v) > 0,$$

if the first term is less than the second term, then $\Delta E > 0$. A deterministic decision rule is used for updating neuron states.

For updating the lines, the following decision rule is used. Let $L^{\cdot;new}_{i,j,k,l}$ and $L^{\cdot;old}_{i,j,k,l}$ denote the new and old states of the line $L_{i,j,k,l}$, respectively. By (5.23), the energy changes due to the state changes of the vertical line $L^{v}_{i,j,k,l}$ and horizontal line $L^{h}_{i,j,k,l}$ can be determined

$$\begin{aligned}\Delta E^{v} &= E^{new} - E^{old} \\ &= \sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=-D_i}^{D_i}\sum_{l=-D_j}^{D_j}[\frac{C}{2}(v_{i,j,k,l} - v_{i\oplus 1,j,k,l})^2(L^{v;old}_{i,j,k,l} - L^{v;new}_{i,j,k,l}) \\ &\quad + D(L^{v;new}_{i,j,k,l} - L^{v;old}_{i,j,k,l})] \qquad (5.30)\end{aligned}$$

and

$$\begin{aligned}\Delta E^{h} &= E^{new} - E^{old} \\ &= \sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\sum_{k=-D_i}^{D_i}\sum_{l=-D_j}^{D_j}[\frac{C}{2}(v_{i,j,k,l} - v_{i,j\oplus 1,k,l})^2(L^{h;old}_{i,j,k,l} - L^{h;new}_{i,j,k,l}) \\ &\quad + D(L^{h;new}_{i,j,k,l} - L^{h;old}_{i,j,k,l})], \qquad (5.31)\end{aligned}$$

respectively. Then, the vertical line $L^{v}_{i,j,k,l}$ and the horizontal line $L^{h}_{i,j,k,l}$ take a new state if the energy changes ΔE^{v} and ΔE^{h} are less than zero, respectively.

The algorithm for computing the optical flow can then be summarized as

1. Set the initial state of the neurons.

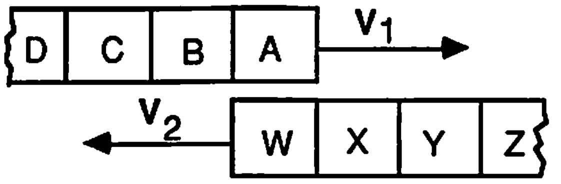
2. Update the state of the all lines synchronously.

3. Update the state of all neurons synchronously.

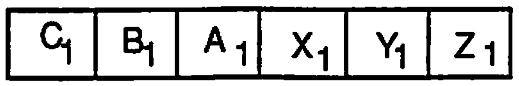
4. Check the energy function. If energy does not change anymore, stop; otherwise, go back to step 2.

5.4 Detection of Motion Discontinuities

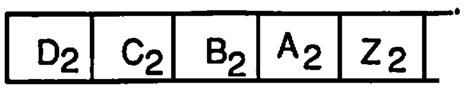
Motion discontinuities in optical flow often result from occluding contours of moving objects. In this case, the moving objects are projected into image plane as adjacent surfaces and their boundaries are undergoing either split or fusion or parallel motion. These motion situations give rise to discontinuities along their boundaries. We must detect these discontinuities to prevent the algorithm from using the physical constraint of surface smoothness in computating the optical flow from one surface to the other. Basically there are two approaches for detecting motion discontinuities. The first approach locates significant changes of optical flow from the computed optical flow field [86, 87, 88, 79]. In [88], a zero-crossing edge detector is used to find the discontinuities in a dense optical flow generated from a sparse one by interpolation. This scheme allows surfaces to translate and rotate as well. Instead of directly detecting discontinuities, region growing techniques are employed in [86, 87] to group elements of similar velocities and the discontinuities are then implicitly given by the boundaries between the regions. Hildreth [79] utilizes the properties of the initial perpendicular components of velocity to locate the discontinuities along zero-crossing contours derived from the image. These schemes are limited to pure translation motion. The second approach is to infer discontinuities from the initial motion measurements without fully computing the optical flow field [64, 13]. In fact this approach interleaves detection of discontinuities and computation of optical flow. By incorporating a line process into the optical flow equation, Koch [13] gives an explicit formulation in for detecting the discontinuities. Although these methods can infer discontinuities, location of discontinuities due to fusion motion may be shifted. In order to locate the discontinuities more accurately, we design a method for detecting the occluding elements based on the initial motion measurements. Once the information about the occluding elements is available, the neural network will correctly locate motion discontinuities.



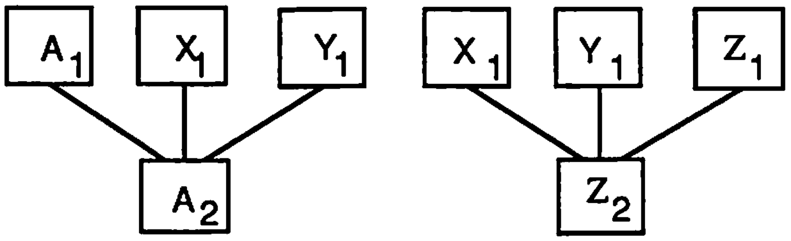
(a) Two moving objects.



(b) First Frame.



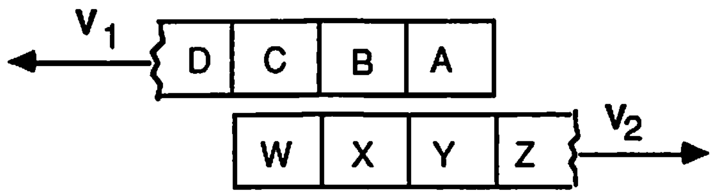
(c) Second frame.



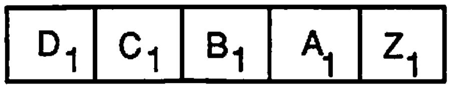
(d) Fusion compitition.

Figure 5.1: Fusion motion.

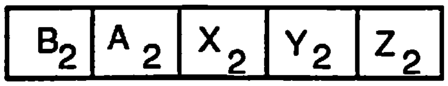
To formalize the analysis, we have to distinguish split motion, fusion motion and non-split-fusion motion. As explained in [83], the split motion occurs when



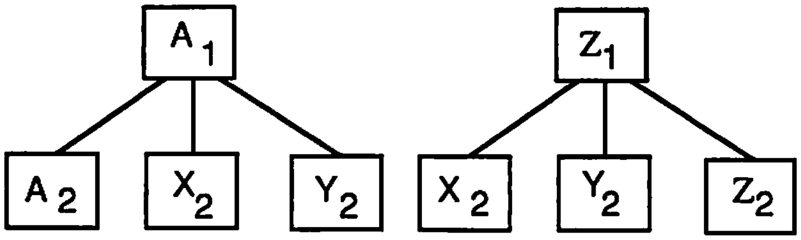
(a) Two moving objects.



(b) First frame.



(c) Second frame.



(d) Split compitition.

Figure 5.2: Split motion.

a single element is replaced by multiple elements, i.e. in two successive frames the single element is shown first followed by multiple elements, while the fusion motion results when multiple elements are presented first followed by a single element. The non-split-fusion motion does not give new elements or eliminate old elements, i.e. the number of the elements between the two frames does not change. Two simple examples are given in Figures 5.1 and 5.2 to illustrate split motion and fusion motion, respectively. Each example is composed of two frames and each frame contains two moving surfaces, one is in front and the another in back. For the first frame, the elements of the surface in the front are denoted by A_1, B_1, etc and the elements of the surface in back by W_1, X_1, etc. For the second frame, A_2, B_2, etc and W_2, X_2, etc denote the elements of the surface in front and the surface in back, respectively. The elements can be either image pixels or lines. In Figure 5.1 two surfaces are moving towards each other. The elements X and Y of the surface in back are visible in the first frame, while in the second frame they are occluded by the front surface. Hence, the elements A_1, X_1 and Y_1 (or X_1, Y_1 and Z_1) are replaced by the element A_2 (or Z_2), and a fusion motion is observed. In Figure 5.2 two surfaces are moving away from each other which gives a split motion. The element A_1 (or Z_1) are presented by A_2, X_2 and Y_2 (or X_2, Y_2 and Z_2). One typical example showing fusion, split and non-split-fusion motion is given in Figure 5.3. Edge **a** of a small square surface is undergoing a fusion motion, edge **b** is a split motion and edges **c** and **d** are undergoing non-split-fusion motion. For computation of optical flow, the fusion motion usually causes problem but the split motion does not. As shown in Figure 5.1(d), in the fusion motion case only one of three elements can find correspondence. The optical flow at the two unmatched points is undetermined. Unlike the fusion motion, the split motion has only one element to be matched with one of three elements as shown in Figure 5.2(d). Hence the optical flow at that point can be determined according to the measurement primitive. In the non-split-fusion motion case, the number of elements does not change and the optical flow is determinable at these points. If the optical flow is perfectly determined, then the line process can successfully and correctly locate motion discontinuities. Thus we will concentrate on the fusion motion case for detecting the occluding elements.

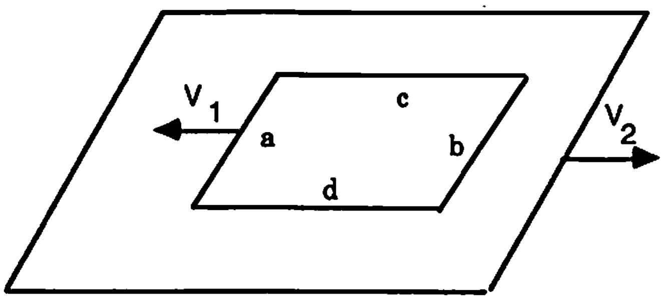
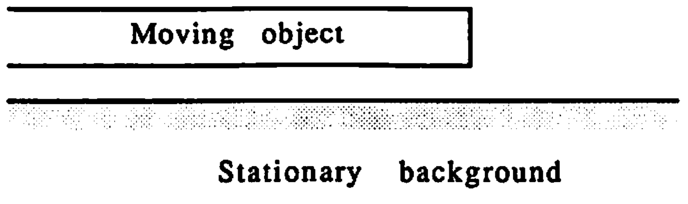


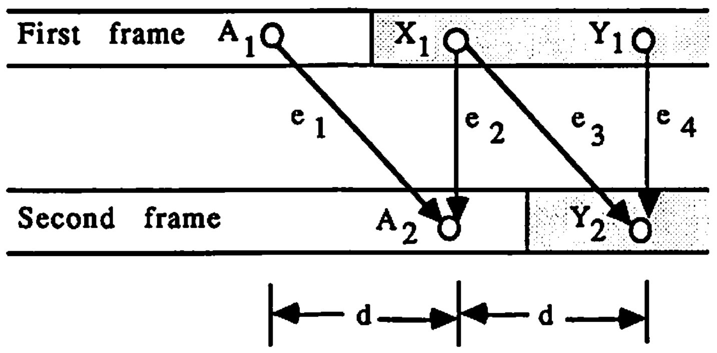
Figure 5.3: A typical example.

Suppose that the surfaces are translating with constant velocities. Let us consider the case in which a surface is moving against a stationary background as shown in Figure 5.4. Let X_1 denote the occluding element, A_2 and X_2 the corresponding elements of A_1 and X_1, respectively. Let (i, j) be the coordinates of element A_1, d_i and d_j the i and j components of optical flow at (i, j), respectively. We assume that X_1 and Y_1 are located at $(i + d_i, j + d_j)$ and $(i + 2 \times d_i, j + 2 \times d_j)$, respectively. By defining the match errors $e_1(i, j)$, $e_2(i, j)$, $e_3(i, j)$ and $e_4(i, j)$ as

$$e_1(i, j) = I_{i,j,d_i,d_j}$$



(a) One moving object.



(b) Detection scheme.

Figure 5.4: Detection of the occluding element.

$$e_2(i,j) = I_{i+d_i,j+d_j,0,0}$$

$$e_3(i,j) = I_{i+d_i,j+d_j,d_i,d_j}$$

$$e_4(i,j) = I_{i+2\times d_i,j+2\times d_j,0,0}$$

where $I_{.,.,.,.}$ are bias inputs given in (5.25), the following relations hold under orthographic or perspective projection without motion along the optical axis,

$$\begin{aligned} e_1(i,j) &\leq e_2(i,j) \\ e_4(i,j) &\leq e_3(i,j). \end{aligned} \quad (5.32)$$

Note that if the above relations do not hold then the element X_1 is not an occluding element. Hence, it is natural to use the relations (5.32) for detecting the occluding elements.

<u>**Detection rule**</u>: An occluding element is detected at $(i+d_i, j+d_j)$ if the optical flow has nonzero values at (i,j) and

$$\begin{aligned} \bar{e}_2(i.j) - \bar{e}_1(i,j) &> T \\ \bar{e}_4(i.j) - \bar{e}_3(i,j) &> T \end{aligned} \tag{5.33}$$

where the theshold T is a nonnegative number and $\bar{e}_k(i,j)$ are the avarage values of the matching errors within a $\Gamma_T \times \Gamma_T$ window S_T

$$\bar{e}_k(i,j) = \frac{1}{\Gamma_T^2} \sum_{s \in S_T} \bar{e}_k((i,j)+s)$$
$$for \quad k = 1,2,3, \ and\, 4.$$

For digitized natural images, T usually takes nonzero value to reduce the effects of quantization error and noise. Using a large value for T can eliminate false occluding elements but may miss the true ones. Since optical flow at an occluding point has zero values, the *a priori* knowledge about the occluding elements can be embedded in the bias inputs by setting, (for instance at point $(i+d_i, j+d_j)$)

$$I_{i+d_i,j+d_j,0,0} = min(I_{i+d_i,j+d_j,k,l};\ -D_i \le k \le D_i, -D_j \le l \le D_j). \tag{5.34}$$

Accordingly, the neural network will prefer zero optical flow at these points and therefore the line process can locate motion discontinuities precisely.

In the case of two moving objects, without any information about the moving objects such as local features it is difficult to detect the occluding elements if only two image frames are used. As discussed in the next section, by using more than two frames the occluding elements can be detected based only on the initial motion measurements.

5.5 A Batch Solution

A natural extension is to compute optical flow over a long time interval, i.e. using multiple frames. We design a neural network based batch approach when the motion is pure translation. Suppose that we have M frames and the objects are translating with constant velocities. The batch solution is given by setting the bias input as

$$\begin{aligned} I_{i,j,k,l} &= -\sum_{r=1}^{M-1}\{A[(k_{r1}(i\oplus(r-1)k, j\oplus(r-1)l) - k_{(r+1)1}(i\oplus rk, j\oplus rl))^2 \\ &\quad +(k_{r2}(i\oplus(r-1)k, j\oplus(r-1)l) - k_{(r+1)2}(i\oplus rk, j\oplus rl))^2] \\ &\quad +(g_r(i\oplus(r-1)k, j\oplus(r-1)l) - g_{r+1}(i\oplus rk, j\oplus rl))^2\} \end{aligned} \quad (5.35)$$

With minor modifications, the detection criterion used for the two frames can be extended to multiple frames. Define the matching errors as

$$\begin{aligned} \mathrm{e}_{1,r}(i,j) &= A[(k_{11}(i,j) - k_{(r+1)1}(i\oplus r_{0i}, j\oplus r_{0j}))^2 \\ &\quad +(k_{12}(i,j) - k_{(r+1)2}(i\oplus r_{0i}, j\oplus r_{0j}))^2] \\ &\quad +(g_1(i,j) - g_{r+1}(i\oplus r_{0i}, j\oplus r_{0j}))^2 \end{aligned} \quad (5.36)$$

$$\begin{aligned} \mathrm{e}_{2,r}(i,j) &= A[(k_{11}(i\oplus r_{0i}, j\oplus r_{0j}) - k_{(r+1)1}(i\oplus r_{0i}, j\oplus r_{0j}))^2 \\ &\quad +(k_{12}(i\oplus r_{0i}, j\oplus r_{0j}) - k_{(r+1)2}(i\oplus r_{0i}, j\oplus r_{0j}))^2] \\ &\quad +(g_1(i\oplus r_{0i}, j\oplus r_{0j}) - g_{r+1}(i\oplus r_{0i}, j\oplus r_{0j}))^2 \end{aligned} \quad (5.37)$$

$$\begin{aligned}
\mathrm{e}_{3,r}(i,j) &= A[(k_{11}(i \oplus r_{0i}, j \oplus r_{0j}) - k_{(r+1)1}(i \oplus r_{1i}, j \oplus r_{1j}))^2 \\
&+ (k_{12}(i \oplus r_{0i}, j \oplus r_{0j}) - k_{(r+1)2}(i \oplus r_{1i}, j \oplus r_{1j}))^2] \\
&+ (g_1(i \oplus r_{0i}, j \oplus r_{0j}) - g_{r+1}(i \oplus r_{1i}, j \oplus r_{1j}))^2 \qquad (5.38)
\end{aligned}$$

$$\begin{aligned}
\mathrm{e}_{4,r}(i,j) &= A[(k_{11}(i \oplus r_{1i}, j \oplus r_{1j}) - k_{(r+1)1}(i \oplus r_{1i}, j \oplus r_{1j}))^2 \\
&+ (k_{12}(i \oplus r_{1i}, j \oplus r_{1j}) - k_{(r+1)2}(i \oplus r_{1i}, j \oplus r_{1j}))^2] \\
&+ (g_1(i \oplus r_{1i}, j \oplus r_{1j}) - g_{r+1}(i \oplus r_{1i}, j \oplus r_{1j}))^2 \qquad (5.39)
\end{aligned}$$

where

$$r_{0i} = rd_i,$$

$$r_{0j} = rd_j,$$

$$r_{1i} = (r+1)d_i$$

and

$$r_{1j} = (r+1)d_j.$$

The detection rule in the case of one moving object is similar to (5.33) i.e. <u>Detection rule</u>: An occluding element is detected at $(i+(r+1)d_i, j+(r+1)d_j)$ if the optical flow at (i,j) is zero and

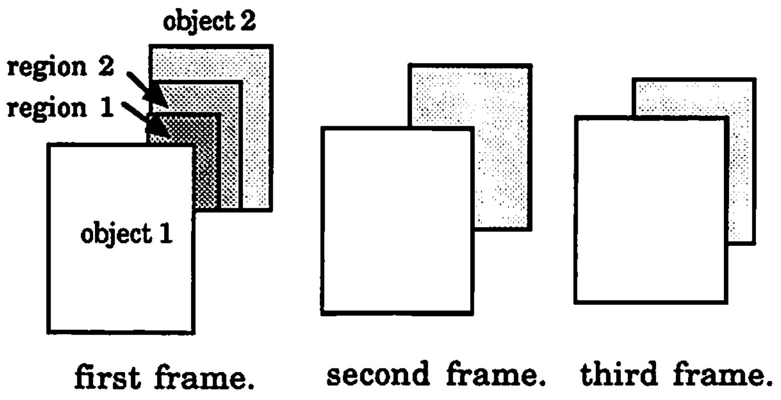
$$\bar{\mathrm{e}}_{2,k}(i.j) - \bar{\mathrm{e}}_{1,k}(i,j) > T,$$

$$\bar{\mathrm{e}}_{4,k}(i.j) - \bar{\mathrm{e}}_{3,k}(i,j) > T \qquad (5.40)$$

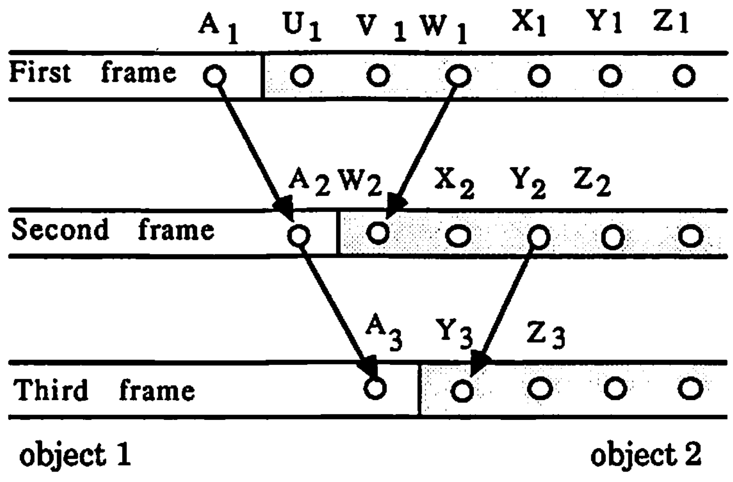
$$for \quad 0 < k \leq r \leq M-1$$

where $\bar{\mathrm{e}}_{.,k}(i.j)$ are avarage values of the matching error.

In the case of two moving objects as shown in Figure 5.5 (a), occluding elements can be detected by using multiple frames. In the second frame, region 1 is occluded



(a) Two moving objects.



(b) Detection scheme.

Figure 5.5: Detection of occluding elements using multiple frames.

and in the third frame both regions 1 and 2 are occluded. Hence, all the elements of regions 1 and 2 are occluding elements. The detection scheme is as follows. As shown in Figure 5.5 (b) (for simplicity, we only illustrate the detection scheme for 1-D case), U_1 and V_1 belong to region 1 and W_1 and X_1 to region 2. Using the first two frames, optical flow at A_1 and W_1 can be determined if the matching errors between A_1 and A_2 and W_1 and W_2 are smaller than some threshold. Similarly the optical flow at A_2 and Y_2 can be computed by using the frames 2 and 3. If optical flow at points W_1, Y_1, W_2 and Y_2 are same but different from that of point A_1, then W_1 and X_1 are occluding elements and therefore U_1 and V_1 are occluding elements too.

5.6 Experimental Results

A number of synthetic and natural image sequences were tested. For each point, we use two memories in the range $-D_i$ to D_i and $-D_j$ to D_j to represent velocities in i and j directions, respectively, instead of using $(2D_i+1)$ and $(2D_j+1)$ neurons. Due to local connections of the neurons, the neuron input $U_{i,j,k,l}$ is computed only within a small window.

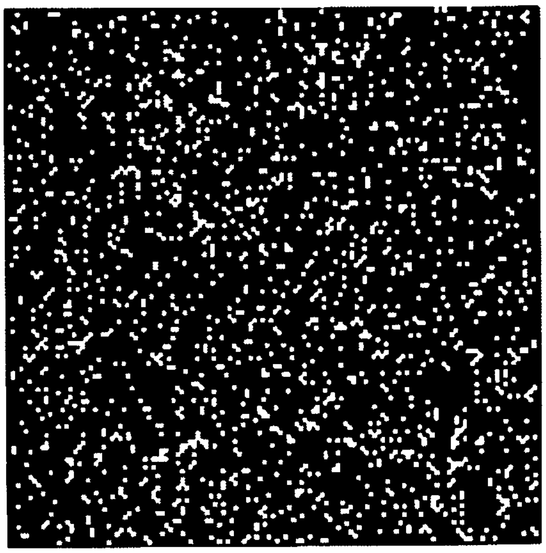
5.6.1 Synthetic Image Sequence

Two experiments based on synthetic image sequences, made of purely translating random dot images and rotating disk images, are presented here. The purely translating random dot images were created by the pseudo random number generating method described in [66] used for generating random dot stereograms. Each dot consists of only one element. Figure 5.6 shows 10% intensity translating random dot images (only 10% of the pixels are white and 90% are black). Intensity values of the white and black elements were set at 255 and 0, respectively. The images are of size 128×128 and contains two square patches which are moving against a stationary background. A center 20×20 patch is moving in the south-east direction with velocity $(v_i = 3, v_j = 3)$, meanwhile a center 40×40 patch partially overlapped by the small patch is moving in the south-southeast direction with velocity $(v_i = 2, v_j = 1)$. Figure 5.6(c) is the resulting optical flow after 10 iterations. For display purposes, only a 70×70 center part of the optical flow field is shown in the figure. The velocity values are normalized by the maximum velocity value. We used $A = 15$, $B = 10,000$, $C = 800$, $D = 1$, $D_i = D_j = 6$ and $W = 0.5$ (the width of subpixel). Note that the flow field is dense.

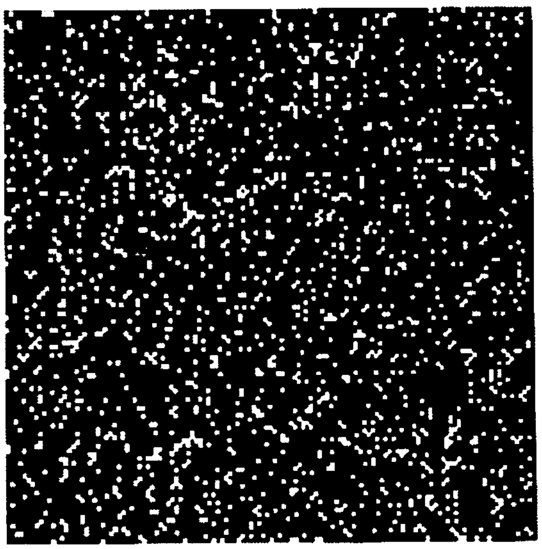
Another test was run on the rotating disk sequence with intensity values in range (0–255). Figures 5.7 (a) and (b) show the first and second frames of intensity images, respectively. Since the rotating disk is not globally rigid and the principle curvatures are estimated over a small window, we assume that the disk is rigid locally. The optical flow field shown in Figure 5.7(c) was obtained after 14 iterations. The parameters used were $A = 30$, $B = 5$, $C = 0$, $D = 0$, $D_i = D_j = 3$ and $W = 0.1$.

5.6.2 Natural Image Sequence

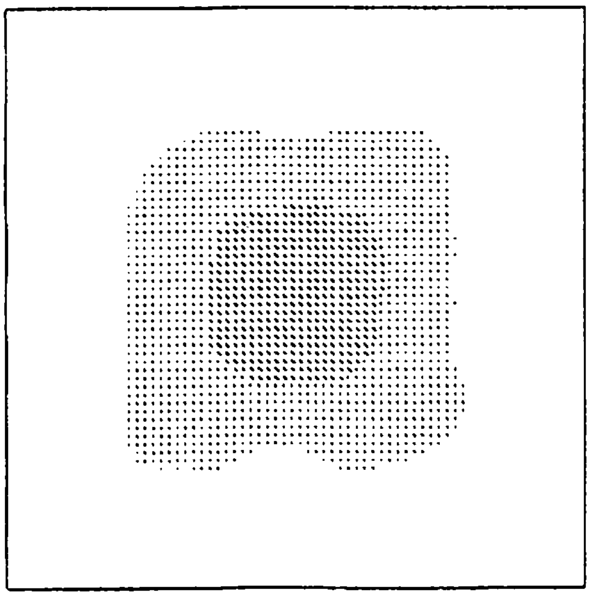
A sequence of pick-up truck images taken from a static camera was used to test both the nonbatch approach (using two image frames) and the batch approach (using multiple image frames). Figure 5.8 shows one frame of the image sequence, a pick-up truck moving from right to left against a stationary background. Since



(a) The first image.

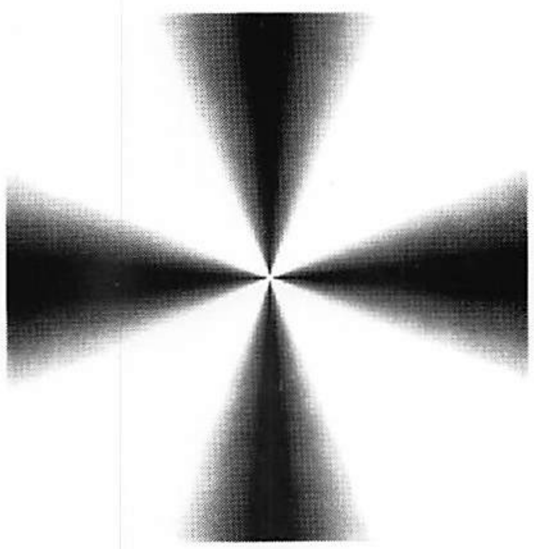


(b) The second image.

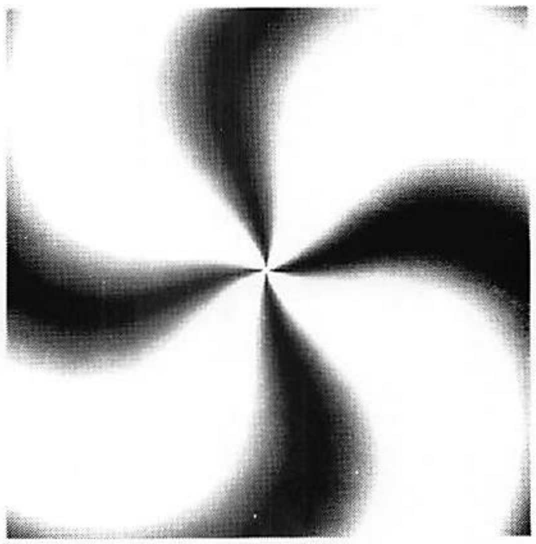


(c) Optical flow.

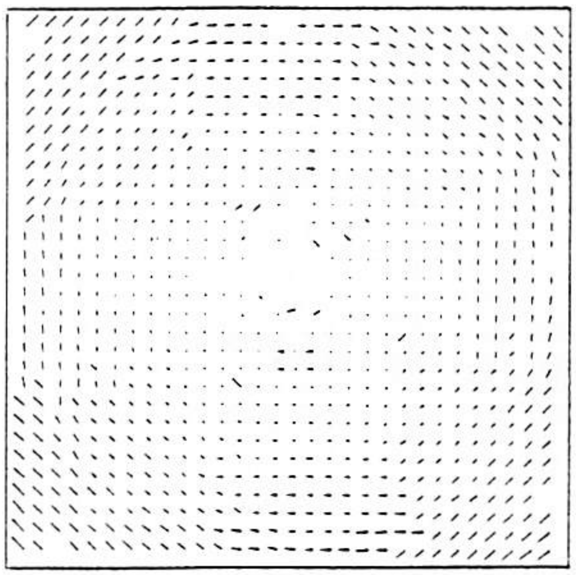
Figure 5.6: 10% density translating random dot images.



(a) The first image.



(b) The second image.



(c) Optical flow.

Figure 5.7: A rotating disk.

the shutter speed was low, the truck was heavely blurred by the motion. The motion blur smeared the edges and erased local features, especially the features on the wheel. Hence, it is difficult to detect the rotation of the wheels. The images are of size 480×480. Figure 5.9 shows four successive frames of the pick-up truck image. Since the rear part of the truck is missing in the first frame, we reversed the order of image sequence so that there is a complete truck image in the first frame. Accordingly, the direction of the computed optical flow should be reversed. For the two frame approach, we used the fourth frame as the first frame and the third frame as the second frame. For simplicity of computation, the image size was reduced to 120×120 by subsampling. By setting $A = 2$, $B = 250$, $C = 50$, $D = 20$, $D_i = 7$, $D_j = 1$ and $W = 1$, the optical flow was obtained after 36 iterations. A 48×113 sample of the computed optical flow corresponding to the part framed by black lines in Figure 5.8 is given in Figure 5.10. Note that although most of the boundary locations are correct, the boundaries due to the fusion motion such as the rear part of the truck and the driver's cab are shifted by the line process.

Figure 5.11 displays the occluding pixels detected at $T = 100$ based on the initially computed optical flow of Figure 5.10. By embedding the information about the occluding pixels into the bias inputs, using the initially computed optical flow as the initial conditions and choosing $A = 2$, $B = 150$, $C = 80$, $D = 20$, $D_i = 7$, $D_j = 1$ and $W = 1$, the final result shown in Figure 5.12 was obtained after 13 iterations. The accuracy of boundary location is significantly improved.

For the batch approach, we used four image frames shown in Figure 5.9. Theoretically there is no limit to the number of frames that can be used in the batch approach. For the same reason mentioned before, the fourth frame was taken as the first frame, the third frame as the second frame, etc. Figure 5.13 displays the occluding pixels detected at $T = 100$ from four frames. Figure 5.14 shows the optical flow computed from four frames using the occluding pixel information. The parameters used were $A = 4$, $B = 1000$, $C = 50$, $D = 20$, $D_i = 7$, $D_j = 1$ and $W = 1$, and 15 iterations were required. As expected, the output is much cleaner and the boundaries are more accurate than that of the two frame based approach.

5.7 Discussion

We have presented two neural network based approaches for computing optical flow. For the two frame based approach, we made no assumptions or requirements on the solutions except smoothness. For the batch approach, we assumed that the object is undergoing a pure translation. Experimental results show that principle curvatures are useful for matching and our approaches based on such measurement primitives work very well especially for some low quality natural images such as the truck images. The algorithm for detecting motion discontinuities also works very well. However, this detection algorithm is limited to pure translation motion, a common problem for most detection algorithms. More research is needed to handle more complicated cases such as translation and rotation.

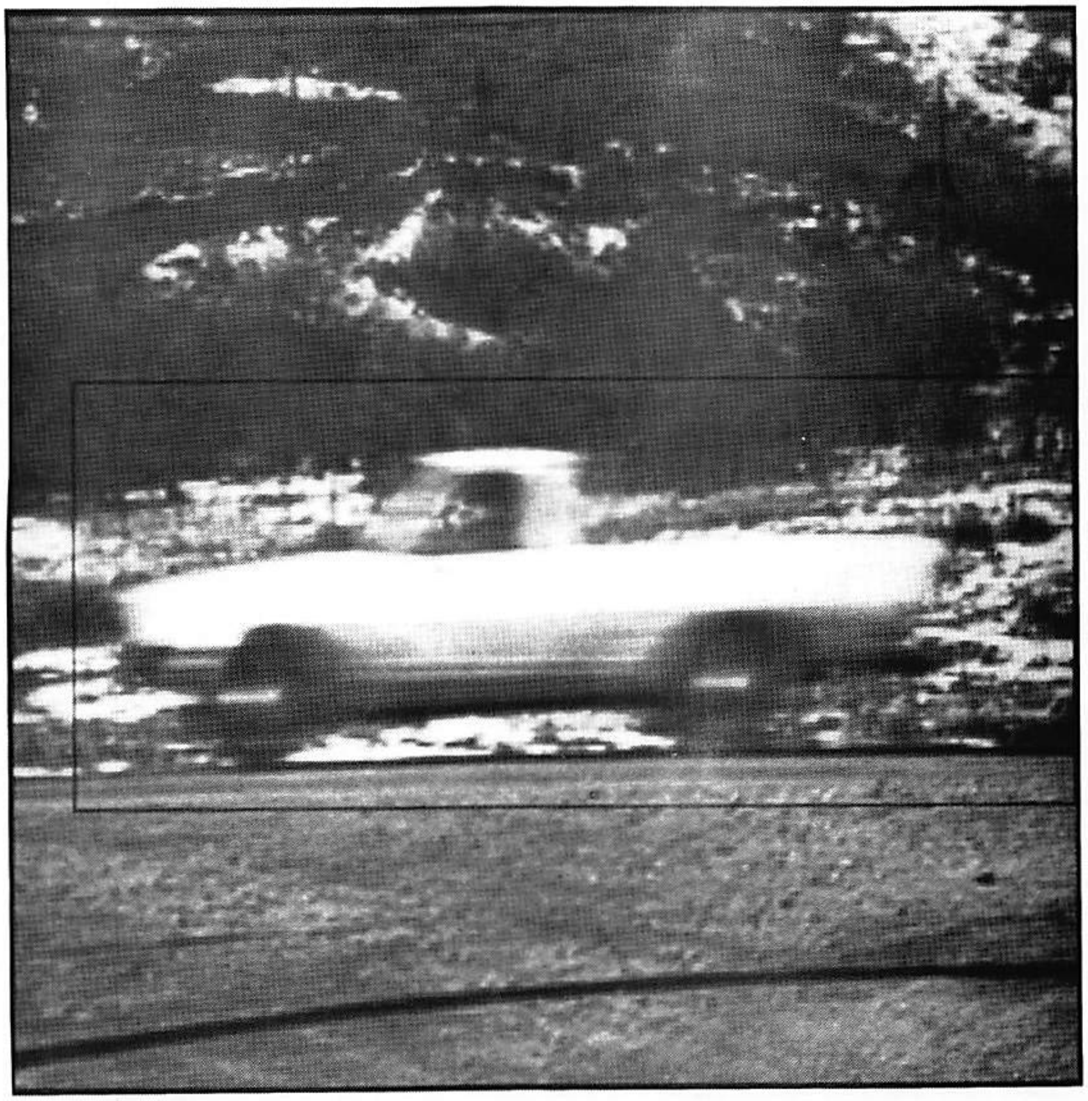
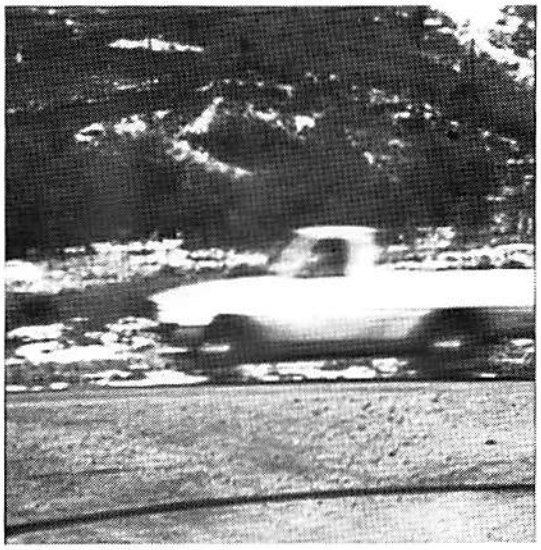
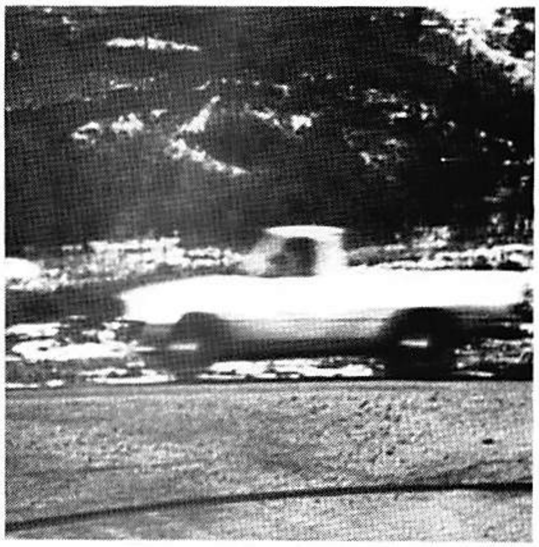
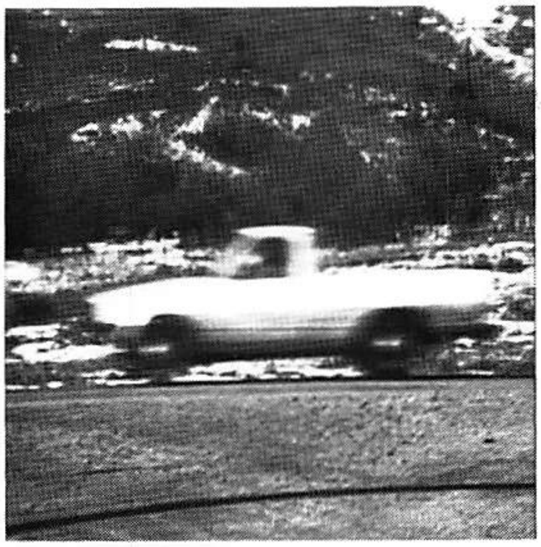


Figure 5.8: Pick-up truck image.


(a) The first frame.


(b) The second frame.


(c) The third frame.

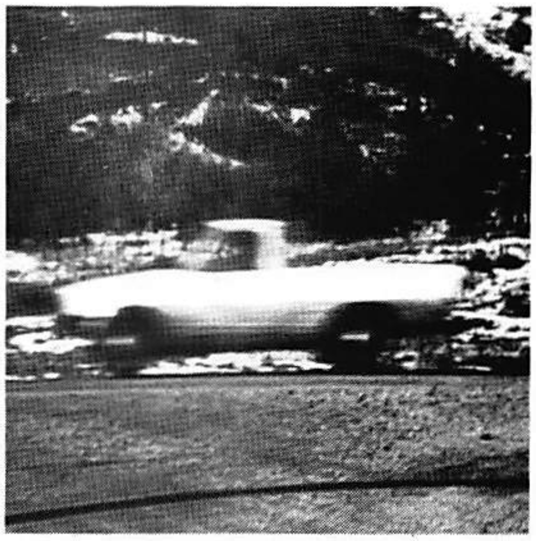

(d) The fourth frame.

Figure 5.9: A sequence of pick-up truck images.

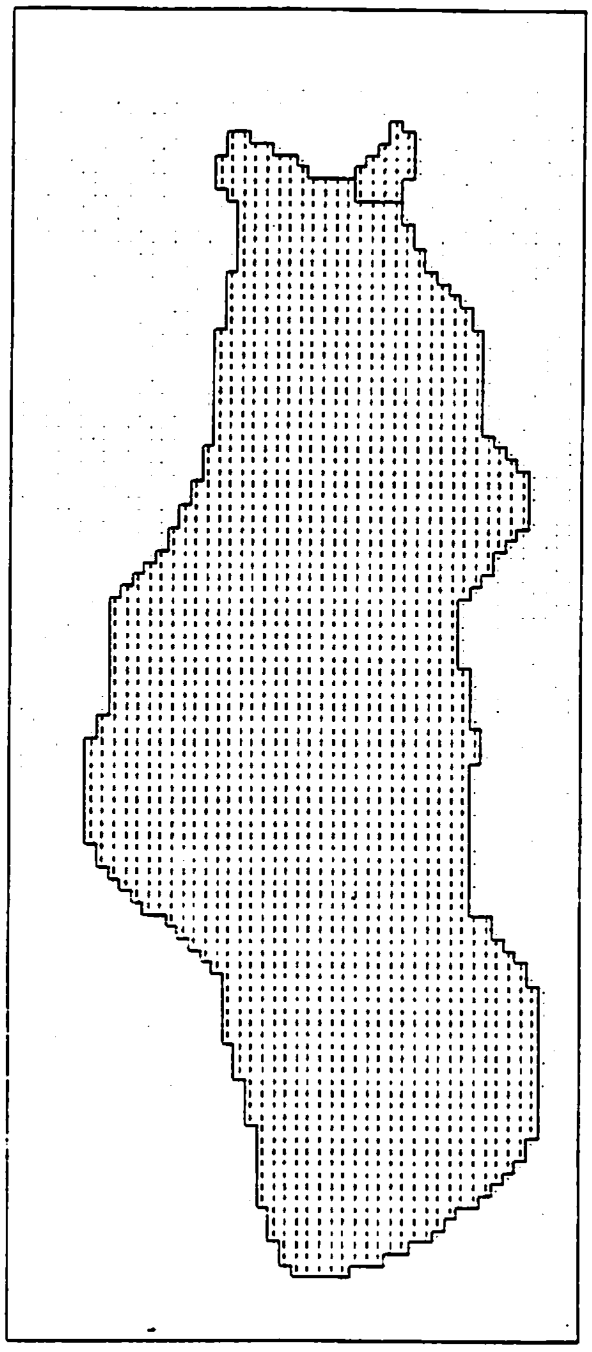


Figure 5.10: Optical flow computed from two images.

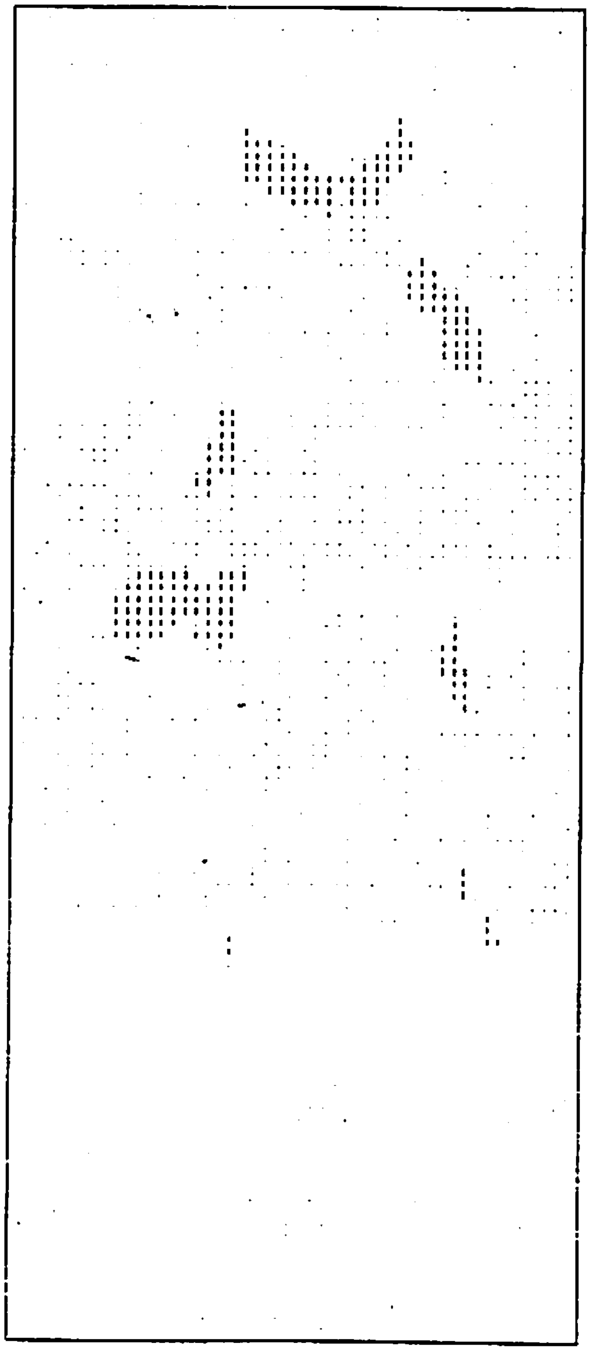


Figure 5.11: Occluding pixels detected from two frames

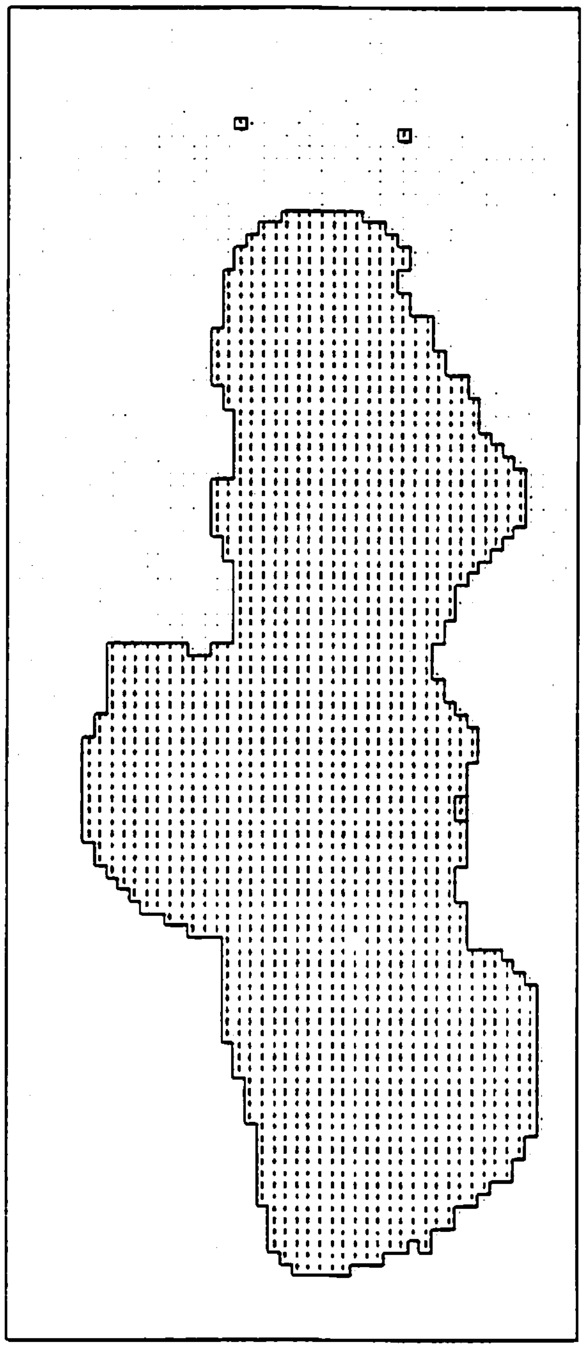


Figure 5.12: Optical flow computed from two frames using the information about the occluding pixels.

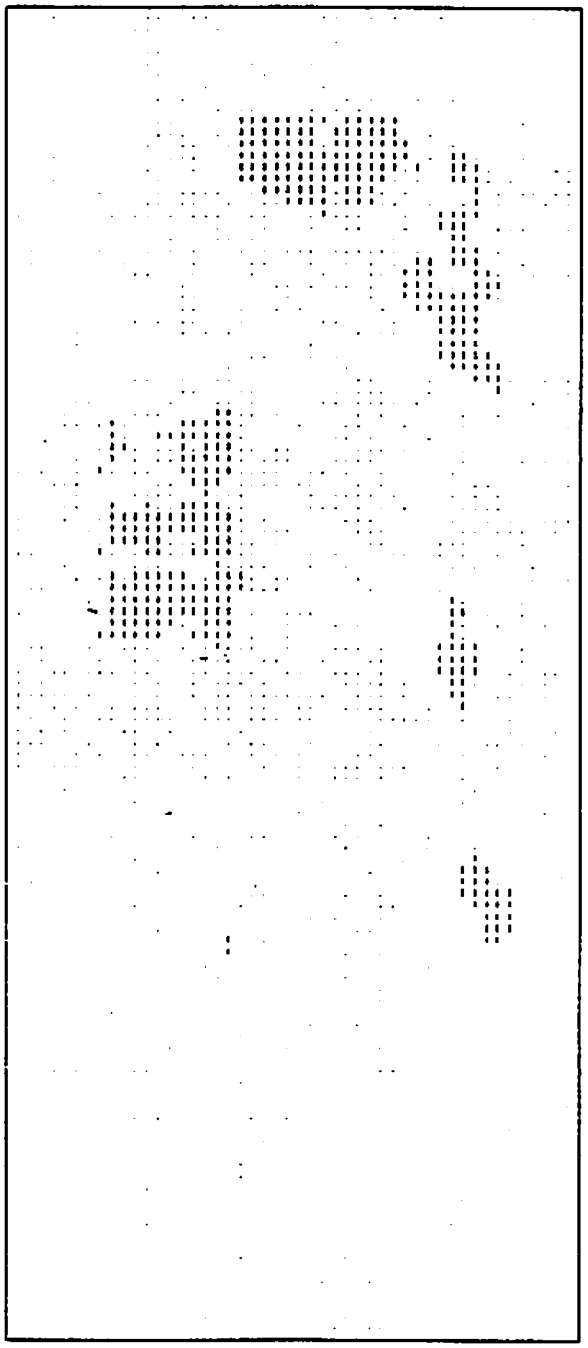


Figure 5.13: Occluding pixels detected from four frames.

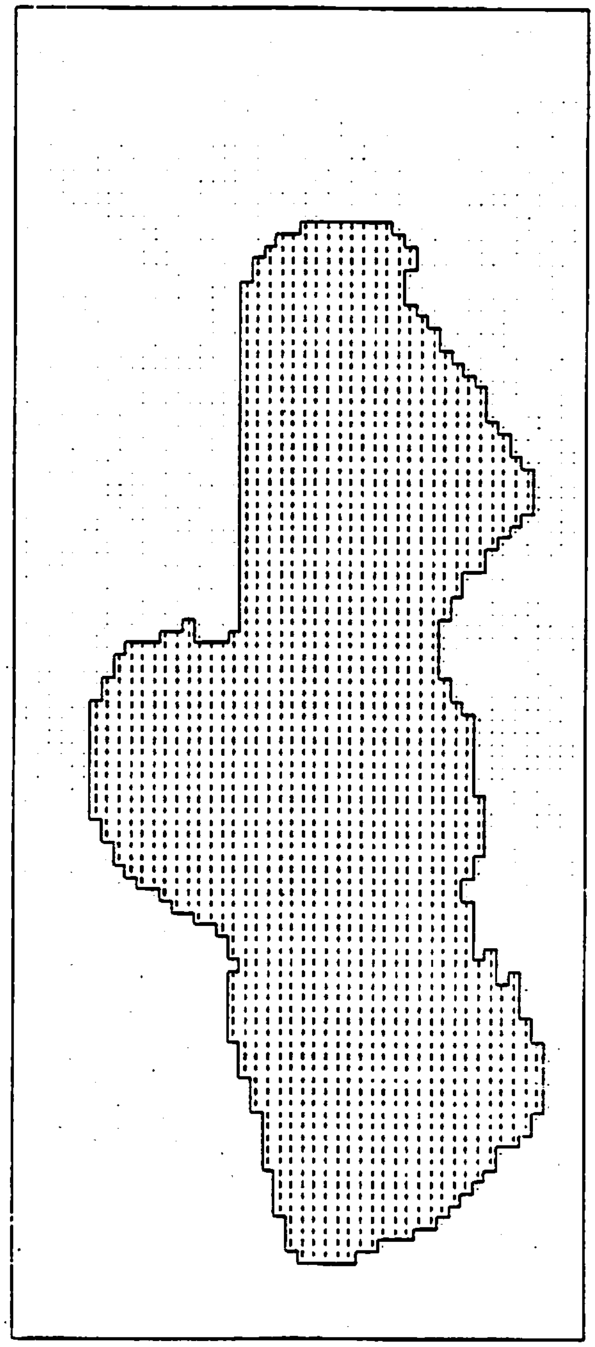


Figure 5.14: Optical flow computed from four frames using the information about the occluding pixels.

Chapter 6

Image Restoration

6.1 Introduction

Restoration of a high quality image from a degraded recording is an important problem in early vision processing. Restoration techniques are applied to remove (1) system degradations such as blur due to optical system aberrations, atmospheric turbulence, motion and diffraction; and (2) statistical degradations due to noise. Over the last 20 years, various methods such as the inverse filter [89], Wiener filter [89], Kalman filter [90], SVD pseudoinverse[89, 91] and many other model based approaches, have been proposed for image restoration. One of the major drawbacks of most of the image restoration algorithms is the computational complexity, so much so that many simplifying assumptions such as wide sense stationarity (WSS), availability of second order image statistics have been made to obtain computationally feasible algorithms. The inverse filter method works only for extremely high signal to noise ratio images. The Wiener filter is usually implemented only after wide sense stationary assumption has been made for images. Furthermore, knowledge of power spectrum or correlation matrix of the undegraded image is required. Oftentimes, additional assumptions regarding boundary conditions are made so that fast orthogonal transforms can be used. The Kalman filter approach can be applied to nonstationary image but is computationally very intensive. Similar statements can be made for SVD pseudoinverse filter method. Approaches based on noncausal models such as the noncausal autoregressive or Gauss Markov random field models [92, 93] also make assumptions such as WSS and periodic boundary conditions. It is desirable to develop a restoration algorithm that does not make WSS assumptions and can be implemented in a reasonable time. An artificial neural network system that can perform extremely rapid computations seems to be very attractive for image restoration in particular and image processing and pattern recognition [94] in general.

In this chapter, we use a neural network model containing redundant neurons to restore gray level images degraded by a known shift invariant blur function and noise. It is based on the model described in [6, 38, 29] using a simple sum number representation [95]. The image gray levels are represented by the simple sum of neuron state variables which take binary values of 1 or 0. The observed image is degraded by a shift–invariant function and noise. The restoration procedure consists of two stages: estimation of parameters of the neural network model and reconstruction of images. During the first stage, the parameters are estimated by comparing the energy function of the neural network with the constrained error function. The nonlinear restoration algorithm is then implemented using a dynamic iterative algorithm to minimize the energy function of the neural network. Owing to the model's fault–tolerant nature and computation capability, a high quality image is obtained using this approach. In order to reduce computational complexity, a practical algorithm, which produces results equivalent to the original one suggested above, is developed under the assumption that the neurons are sequentially visited. We illustrate the usefulness of this approach by using both synthetic and real images degraded by a known shift–invariant blur function with or without noise. We also discuss the problem of choosing boundary values and suggest two methods to reduce the ringing effect. Comparisons with other restoration methods such as the SVD pseudoinverse filter, the minimum mean square error (MMSE) filter and the modified MMSE filter using Gaussian Markov random field model are given using real images. The advantages of the method developed in this paper are (1) WSS assumption is not required for the images, (2) can be implemented rapidly and (3) is fault tolerant. We also present a schematic diagram for optical implemetation of this approach.

6.2 Image Representation Using a Neural Network

We use a neural network containing redundant neurons for representing the image gray levels. The model consists of $L^2 \times M$ mutually interconnected neurons, where L is the size of image and M is the maximum value of the gray level function. Let $V = \{v_{i,k},\ \ where\ \ 1 \leq i \leq L^2, 1 \leq k \leq M\}$ be a binary state set of the neural network with $v_{i,k}$ (1 for firing and 0 for resting) denoting the state of the (i,k)th neuron. Let $T_{i,k;j,l}$ denote the strength (possibly negative) of the interconnection between neuron (i,k) and neuron (j,l). We require symmetry

$$T_{i,k;j,l} = T_{j,l;i,k} \quad for \quad 1 \leq i,j \leq L^2 \ \ and \ \ 1 \leq l,k \leq M$$

We also allow for neurons to have self–feedback, i.e. $T_{i,k;i,k} \neq 0$. In this model, each neuron (i,k) randomly and asynchronously receives inputs $\sum T_{i,k;j,l}v_{j,l}$ from all neurons and a bias input $I_{i,k}$

$$u_{i,k} = \sum_{j}^{L^2} \sum_{l}^{M} T_{i,k;j,l}v_{j,l} + I_{i,k} \tag{6.1}$$

Each $u_{i,k}$ is fed back to corresponding neurons after thresholding

$$v_{i,k} = g(u_{i,k}) \tag{6.2}$$

where $g(x)$ is a nonlinear function whose form can be taken as

$$g(x) = \begin{cases} 1 & if \;\; x \geq 0 \\ 0 & if \;\; x < 0. \end{cases} \tag{6.3}$$

In this model, the state of each neuron is updated by using the latest information about other neurons.

The image is described by a finite set of gray level functions $\{x(i,j);\, 1 \leq i,j \leq L\}$ with $x(i,j)$ (positive integer number) denoting the gray level of the pixel (i,j). The image gray level function can be represented by a simple sum of the neuron state variables as

$$x(i,j) = \sum_{k=1}^{M} v_{m,k} \tag{6.4}$$

where $m = (i-1) \times L + j$. Here the gray level functions have degenerate representations. Use of this redundant number representation scheme yields advantages such as fault–tolerance and faster convergence to the solution [95].

By using the lexicographic notation, the image degradation model can be written as

$$\underline{Y} = H\underline{X} + \underline{N} \tag{6.5}$$

where H is the "blurring matrix" corresponding to a blur function, $\underline{N}$ is the signal independent white noise, $\underline{X}$ and $\underline{Y}$ are the original and degraded images, respectively. Furthermore, H and $\underline{N}$ can be represented as

$$H = \begin{bmatrix} h_{1,1} & h_{1,2} & \cdot\ \cdot\ \cdot & h_{1,L^2} \\ h_{2,1} & h_{2,2} & \cdot\ \cdot\ \cdot & h_{2,L^2} \\ \cdot & \cdot & \cdot\ \cdot\ \cdot & \cdot \\ \cdot & \cdot & \cdot\ \cdot\ \cdot & \cdot \\ \cdot & \cdot & \cdot\ \cdot\ \cdot & \cdot \\ h_{L^2,1} & h_{L^2,2} & \cdot\ \cdot\ \cdot & h_{L^2,L^2} \end{bmatrix} \tag{6.6}$$

and

$$\underline{N} = \begin{bmatrix} \underline{N}_1 \\ \underline{N}_2 \\ \cdot \\ \cdot \\ \cdot \\ \underline{N}_L \end{bmatrix} = \begin{bmatrix} n_1 \\ n_2 \\ \cdot \\ \cdot \\ \cdot \\ n_{L^2} \end{bmatrix}, \quad \underline{N}_i = \begin{bmatrix} n(i,1) \\ n(i,2) \\ \cdot \\ \cdot \\ \cdot \\ n(i,L) \end{bmatrix} = \begin{bmatrix} n_{(i-1)\times L+1} \\ n_{(i-1)\times L+2} \\ \cdot \\ \cdot \\ \cdot \\ n_{i\times L} \end{bmatrix} . \tag{6.7}$$

respectively. Vectors $\underline{X}$ and $\underline{Y}$ have similar representations. Equation (6.5) is similar to the simultaneous equations solution of [95], but differs in that it includes a noise term.

The shift–invariant blur function can be written as a convolution over a small window; for instance, it takes the form

$$h(k,l) = \begin{cases} \frac{1}{2} & if\ \ k=0,\ l=0 \\ \frac{1}{16} & if\ \ |k|,\ |l| \le 1,\ \ (k,l) \ne (0,0) \end{cases} \tag{6.8}$$

accordingly, the "blur matrix" H will be a block Toeplitz or block circulant matrix (if the image has periodic boundaries). The block circulant matrix corresponding to (6.8) can be written as

$$H = \begin{bmatrix} H_0 & H_1 & \underline{0} & \cdots & \underline{0} & H_1 \\ H_1 & H_0 & H_1 & \cdots & \underline{0} & \underline{0} \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ H_1 & \underline{0} & \underline{0} & \cdots & H_1 & H_0 \end{bmatrix} \tag{6.9}$$

where

$$H_0 = \begin{bmatrix} \frac{1}{2} & \frac{1}{16} & 0 & \cdots & 0 & \frac{1}{16} \\ \frac{1}{16} & \frac{1}{2} & \frac{1}{16} & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \frac{1}{16} & 0 & 0 & \cdots & \frac{1}{16} & \frac{1}{2} \end{bmatrix} \tag{6.10}$$

$$H_1 = \begin{bmatrix} \frac{1}{16} & \frac{1}{16} & 0 & \cdots & 0 & \frac{1}{16} \\ \frac{1}{16} & \frac{1}{16} & \frac{1}{16} & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \frac{1}{16} & 0 & 0 & \cdots & \frac{1}{16} & \frac{1}{16} \end{bmatrix} \tag{6.11}$$

and $\underline{0}$ is null matrix whose elements are all zeros.

6.3 Estimation of Model Parameters

The neural model parameters, the interconnection strengths and bias inputs, can be determined in terms of the energy function of the neural network. As defined in [6], the energy function of the neural network can be written as

$$E = -\frac{1}{2}\sum_{i=1}^{L^2}\sum_{j=1}^{L^2}\sum_{k=1}^{M}\sum_{l=1}^{M} T_{i,k;j,l}\, v_{i,k}\, v_{j,l} - \sum_{i=1}^{L^2}\sum_{k=1}^{M} I_{i,k}\, v_{i,k} \tag{6.12}$$

In order to use the spontaneous energy–minimization process of the neural network, we reformulate the restoration problem as one of minimizing an error function with constraints defined as

$$E = \frac{1}{2}\|\underline{Y} - H\hat{\underline{X}}\|^2 + \frac{1}{2}\lambda\|D\hat{\underline{X}}\|^2 \tag{6.13}$$

where $\|\underline{Z}\|$ is the L_2 norm of $\underline{Z}$ and λ is a constant. Such constrained error function is widely used in the image restoration problems [89] and is also similar to the regularization techniques used in early vision problems [96]. The first term in (6.13) is to seek an $\hat{\underline{X}}$ such that $H\hat{\underline{X}}$ approximates $\underline{Y}$ in a least squares sense. Meanwhile, the second term is a smoothness constraint on the solution $\hat{\underline{X}}$. The constant λ determines their relative importance to achieve suppression of noise and ringing.

In general, if H is a low pass distortion, then D is a high pass filter. A common choice of D is a second order differential operator which can be approximated as a local window operator in the 2-D discrete case. For instance, if D is a Laplacian operator

$$\nabla = \frac{\partial^2}{\partial i^2} + \frac{\partial^2}{\partial j^2} \tag{6.14}$$

it can be approximated as a window operator

$$\frac{1}{6}\begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}. \tag{6.15}$$

Then D will be a block Toeplitz matrix similar to (6.9).

Expanding (6.13) and then replacing x_i by (6.4), we have

$$\begin{aligned} E &= \frac{1}{2}\sum_{p=1}^{L^2}(y_p - \sum_{i=1}^{L^2} h_{p,i}x_i)^2 + \frac{1}{2}\lambda\sum_{p=1}^{L^2}(\sum_{i=1}^{L^2} d_{p,i}x_i)^2 \\ &= \frac{1}{2}\sum_{i=1}^{L^2}\sum_{j=1}^{L^2}\sum_{k=1}^{M}\sum_{l=1}^{M}\sum_{p=1}^{L^2} h_{p,i}\, h_{p,j}\, v_{i,k}\, v_{j,l} + \frac{1}{2}\sum_{p=1}^{L^2} y_p^2 - \sum_{i=1}^{L^2}\sum_{k=1}^{M}\sum_{p=1}^{L^2} y_p\, h_{p,i}\, v_{i,k} \\ &+ \frac{1}{2}\lambda\sum_{i=1}^{L^2}\sum_{j=1}^{L^2}\sum_{k=1}^{M}\sum_{l=1}^{M}\sum_{p=1}^{L^2} d_{p,i}\, d_{p,j}\, v_{i,k}\, v_{j,l} \end{aligned} \tag{6.16}$$

By comparing the terms in (6.16) with the corresponding terms in (6.12) and ignoring the constant term $\frac{1}{2}\sum_{p=1}^{L^2} y_p^2$, we can determine the interconnection strengths and bias inputs as

$$T_{i,k;j,l} = -\sum_{p=1}^{L^2} h_{p,i}\, h_{p,j} - \lambda\sum_{p=1}^{L^2} d_{p,i}\, d_{p,j} \tag{6.17}$$

and

$$I_{i,k} = \sum_{p=1}^{L^2} y_p\, h_{p,i}. \tag{6.18}$$

where $h_{i,j}$ and $d_{i,j}$ are the elements of the matrices H and D, respectively. Two interesting aspects of (6.17) and (6.18) should be pointed out: (1) the interconnection strengths are independent of subscripts k and l and the bias inputs are independent of subscript k, and (2) the self-connection $T_{i,k;i,k}$ is not equal to zero which requires self-feedback for neurons.

From (6.17), one can see that the interconnection strengths are determined by the shift–invariant blur function, differential operator and constant λ. Hence,

$T_{i,k;j,l}$ can be computed without error provided the blur function is known. However, the bias inputs are functions of the observed degraded image. If the image is degraded by a shift-invariant blur function only, then $I_{i,k}$ can be estimated perfectly. Otherwise, $I_{i,k}$ is affected by noise. The reasoning behind this statement is as follows. By replacing y_p by $\sum_{i=1}^{L^2} h_{p,i}\, x_i + n_p$, we have

$$\begin{aligned} I_{i,k} &= \sum_{p=1}^{L^2} (\sum_{i=1}^{L^2} h_{p,i}\, x_i + n_p)\, h_{p,i} \\ &= \sum_{p=1}^{L^2} \sum_{i=1}^{L^2} h_{p,i}\, x_i\, h_p + \sum_{p=1}^{L^2} n_p\, h_{p,i} \end{aligned} \tag{6.19}$$

The second term in (6.19) represents the effects of noise. If the signal to noise ratio (SNR), defined by

$$SNR = 10 \log_{10} \frac{\sigma_s^2}{\sigma_n^2}, \tag{6.20}$$

where σ_s^2 and σ_n^2 are variances of signal and noise, respectively, is low, then we have to choose a large λ to suppress effects due to noise. It seems that in the absence of noise, the parameters can be estimated perfectly, ensuring exact recovery of the image as error function E tends to zero. However, the problem is not so simple, as the restoration performance depends on both the parameters and the blur function when a mean square error or least square error such as (6.13) is used. A discussion about the effect of blur function is given in section 10.

6.4 Restoration

Restoration is carried out by neuron evaluation and an image construction procedure. Once the parameters $T_{i,k;j,l}$ and $I_{i,k}$ are obtained using (6.17) and (6.18), each neuron can randomly and asynchronously evaluate its state and readjust accordingly using (6.1) and (6.2). When one quasi-minimum energy point is reached, the image can be constructed using (6.4).

Although a step function is used as activation function, the self-feedback may still cause the energy function E to increase with a transition. This is explained below. Define the state change $\Delta v_{i,k}$ of neuron (i,k) and energy change ΔE as

$$\Delta v_{i,k} = v_{i,k}^{new} - v_{i,k}^{old} \quad and \quad \Delta E = E^{new} - E^{old}$$

Consider the energy function

$$E = -\frac{1}{2}\sum_{i=1}^{L^2}\sum_{j=1}^{L^2}\sum_{k=1}^{M}\sum_{l=1}^{M} T_{i,k;j,l}\, v_{i,k}\, v_{j,l} - \sum_{i=1}^{L^2}\sum_{k=1}^{M} I_{i,k}\, v_{i,k}, \tag{6.21}$$

Since only one neuron is updated at each step, the energy change ΔE due to a change $\Delta v_{i,k}$ is given by

$$\Delta E = -(\sum_{j=1}^{L^2}\sum_{l=1}^{M} T_{i,k;j,l}\, v_{j,l} + I_{i,k}\ )\, \Delta v_{i,k} - \frac{1}{2}\, T_{i,k;i,k}\, (\Delta v_{i,k})^2 \tag{6.22}$$

which is not always negative. For instance, if

$$v_{i,k}^{old} = 0, \qquad u_{i,k} = \sum_{j=1}^{L^2}\sum_{l=1}^{M} T_{i,k;j,l}\, v_{j,l} + I_{i,k}\ > 0,$$

and the threshold function is as in (6.3), then $v_{i,k}^{new} = 1$ and $\Delta v_{i,k} > 0$. Thus, the first term in (6.22) is negative. But

$$T_{i,k;i,k} = -\sum_{p=1}^{L^2} h_{p,i}^2 - \lambda \sum_{p=1}^{L^2} d_{p,i}^2 < 0.$$

with $\lambda > 0$ leading to

$$-\frac{1}{2}\, T_{i,k;i,k}\, (\Delta v_{i,k})^2 > 0.$$

When the first term is less than the second term in (6.22), then $\Delta E > 0$.

Thus, depending on whether convergence to a local minimum or a global minimum is desired, we can use a deterministic or stochastic decision rule. The restoration algorithm is summarized below.

Algorithm 1:

1. Set the initial state of the neurons.
2. Update the state of all neurons randomly and asynchronously according to the decision rule.
3. Check the energy function; if energy does not change, go to step 4; otherwise, go back to step 2.
4. Construct an image using (6.4).

6.5 A Practical Algorithm

The algorithm described above is difficult to simulate on a conventional computer owing to high computational complexity even for images of reasonable size. For instance, if we have an $L \times L$ image with M gray levels, then L^2M neurons and $\frac{1}{2}L^4M^2$ interconnections are required and L^4M^2 additions and multiplications are needed at each iteration. Therefore, the space and time complexities are $O(L^4M^2)$ and $O(L^4M^2K)$, respectively, where K, typically $10 - 100$, is the number of iterations. Usually, L and M are $256 - 1024$ and 256, respectively. However, simplification is possible if the neurons are sequentially updated .

In order to simplify the algorithm, we begin by reconsidering (6.1) and (6.2) of the neural network. As noted earlier the interconnection strengths given in (6.17) are independent of subscripts k and l and the bias inputs given in (6.18) are independent of subscript k, the M neurons used to represent the same image gray level function have the same interconnection strengths and bias inputs. Hence, one set of interconnection strengths and one bias input are sufficient for every gray level function, i.e. the dimensions of the interconnection matrix T and bias input matrix I can be reduced by a factor of M^2. From (6.1) all inputs received by a neuron, say, the (i,k)th neuron can be written as

$$\begin{aligned} u_{i,k} &= \sum_{j}^{L^2} T_{i,\cdot;j,\cdot} \left(\sum_{l}^{M} v_{j,l}\right) + I_{i,\cdot} \\ &= \sum_{j}^{L^2} T_{i,\cdot;j,\cdot}\, x_j + I_{i,\cdot} \end{aligned} \tag{6.23}$$

where we have used (6.4) and x_j is the gray level function of the jth image pixel. The symbol "$\cdot$" in the subscripts means that the $T_{i,\cdot;j,\cdot}$ and $I_{i,\cdot}$ are independent of k. Equation (6.23) suggests that we can use a multivalue number to replace the simple sum number. Since the interconnection strengths are determined by the blur function, the differential operator and the constant λ as shown in (6.17), it is easy to see that if the blur function is local, then most interconnection strengths are zeros and the neurons are locally connected. Therefore, most elements of the interconnection matrix T are zeros. If the blur function is shift invariant taking the form in (6.8), then the interconnection matrix is block Toeplitz so that only a few elements need to be stored. Based on the value of inputs $u_{i,k}$, the state of the (i,k)th neuron is updated by applying a decision rule. The state change of the (i,k)th neuron in turn causes the gray level function x_i to change as

$$x_i^{new} = \begin{cases} x_i^{old} & if \ \ \Delta v_{i,k} = 0 \\ x_i^{old} + 1 & if \ \ \Delta v_{i,k} = 1 \\ x_i^{old} - 1 & if \ \ \Delta v_{i,k} = -1 \end{cases} \tag{6.24}$$

where $\Delta v_{i,k} = v_{i,k}^{new} - v_{i,k}^{old}$ is the state change of the (i,k)th neuron. The superscripts "new" and "old" are for after and before updating, respectively. We use x_i to represent the gray level value as well as the output of M neurons representing x_i. Assuming that the neurons of the network are sequentially visited, it is straightforward to show that the updating procedure can be reformulated as

$$u_{i,k} = \sum_{j}^{L^2} T_{i,.;j,.}\, x_j + I_{i,.} \tag{6.25}$$

$$\Delta v_{i,k} = g(u_{i,k}) = \begin{cases} \Delta v_{i,k} = 0 & if\ \ u_{i,k} = 0 \\ \Delta v_{i,k} = 1 & if\ \ u_{i,k} > 0 \\ \Delta v_{i,k} = -1 & if\ \ u_{i,k} < 0 \end{cases} \tag{6.26}$$

$$x_i^{new} = \begin{cases} x_i^{old} + \Delta v_{i,k} & if\ \ \Delta E < 0 \\ x_i^{old} & if\ \ \Delta E \geq 0 \end{cases} \tag{6.27}$$

Note that the stochastic decision rule can also be used in (6.27). In order to limit the gray level function to the range 0–255 after each updating step, we have to check the value of the gray level function x_i^{new}. Equations (6.25), (6.26) and (6.27) give a much simpler algorithm. This algorithm is summarized below.

Algorithm 2:

1. Take the degraded image as the initial value.

2. Sequentially visit all numbers (image pixels). For each number, use (6.25), (6.26) and (6.27) to update it repeatedly until no further change, i.e. if $\Delta v_{i,k} = 0$ or energy change $\Delta E \geq 0$, then move to next one.

3. Check the energy function; if the energy does not change anymore, a restored image is obtained; otherwise, go back to step 2 for another iteration.

The calculations of the inputs $u_{i,k}$ of the (i,k)th neuron and the energy change ΔE can be simplified furthermore. When we update the same image gray level function repeatedly, the input received by the current neuron (i,k) can be computed by making use of the previous result

$$u_{i,k} = u_{i,k-1} + \Delta v_{i,k}\, T_{i,.;i,.} \tag{6.28}$$

where $u_{i,k-1}$ is the input received by the $(i,k-1)$th neuron. The energy change ΔE due to the state change of the (i,k)th neuron can be calculated as

$$\Delta E = -u_{i,k}\,\Delta v_{i,k} - \frac{1}{2}\,T_{i,\cdot;i,\cdot}\,(\Delta v_{i,k})^2 \tag{6.29}$$

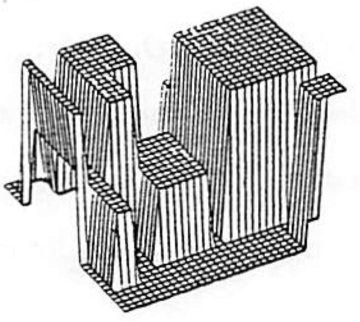
If the blur function is shift invariant, all these simplifications reduce the space and time complexities significantly from $O(L^4M^2)$ and $O(L^4M^2K)$ to $O(L^2)$ and $O(ML^2K)$, respectively. Since every gray level function needs only a few updating steps after the first iteration, the computation at each iteration is $O(L^2)$. The resulting algorithm can be easily simulated on mini–computers for images as large as 512×512.

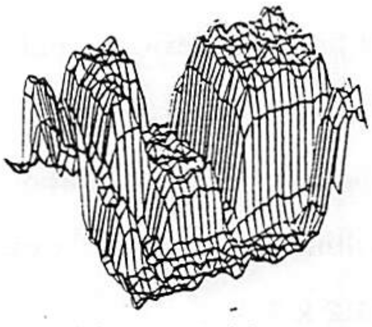
6.6 Computer Simulations

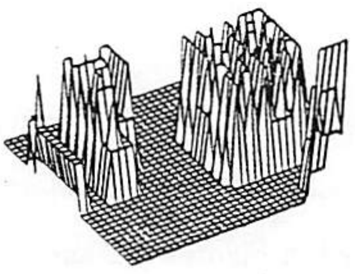
The practical algorithm described in the previous section was applied to synthetic and real images on a Sun-3/160 Workstation. In all cases only the deterministic decision rule was used. The results are summarized in Figures 6.1 and 6.2.

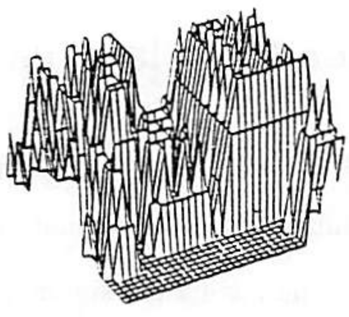
Figure 6.1 shows the results for a synthetic image. The original image shown in Figure 6.1(a) is of size 32×32 with 3 gray levels. The image was degraded by convolving with a 3×3 blur function as in (6.8) using circulant boundary conditions; 22 dB white Gaussian noise was added after convolution. A perfect image was obtained after 6 iterations without preprocessing. We set the initial state of all neurons to equal 1, i.e. firing and chose $\lambda = 0$ due to the well conditioning of the blur function.

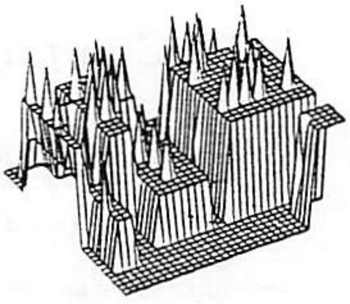
Figure 6.2(a) shows the original girl image. The original image is of size 256×256 with 256 gray levels. The variance of the original image is 2797.141. It was degraded by a 5×5 uniform blur function. A small amount of quantization

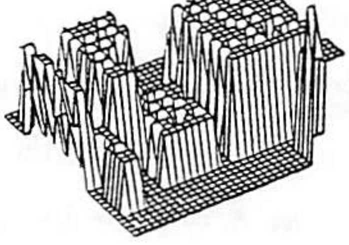

(a) Original girl image.

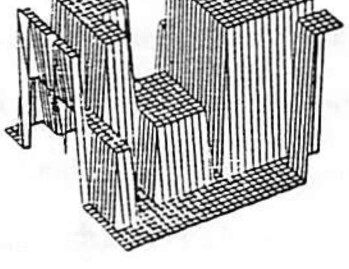

(b) Degraded image.


(c) The 1st iteration.


(d) The 2nd iteration.


(e) The 3rd iteration.


(f) The 4th iteration.


(g) The 5th iteration.

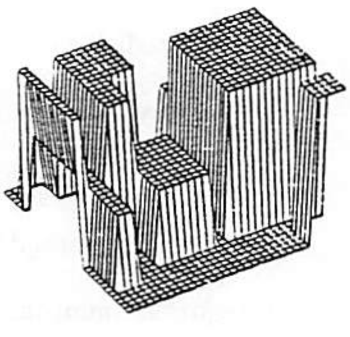
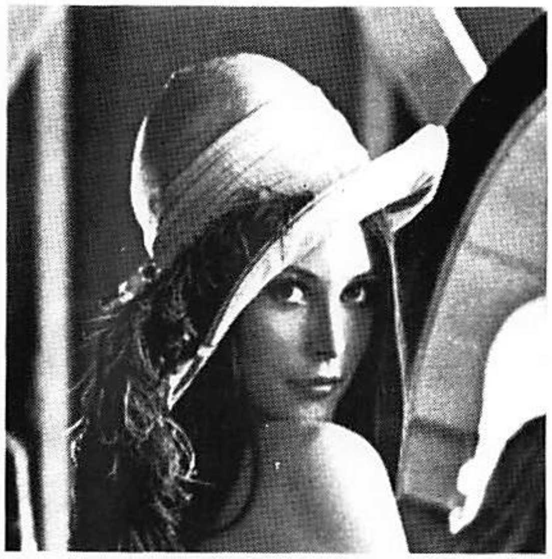

(h) The 6th iteration.

Figure 6.1: Restoration of noisy blurred synthetic image.

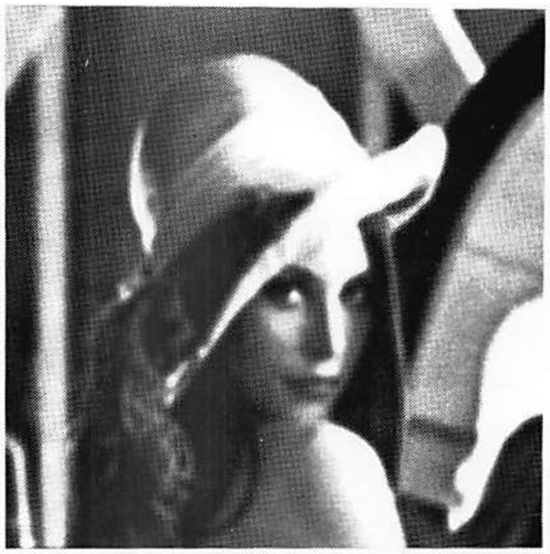
noise was introduced by quantizing the convolution results to 8 bits. The noisy blurred image is shown in Figure 6.2(b). For comparison purpose, Figure 6.2(c) shows the output of an inverse filter [97], completely overridden by the amplified noise and the ringing effects due to the ill conditioned blur matrix H. Since the blur matrix H corresponding to the 5×5 uniform blur function is not singular, the pseudoinverse filter [97] and the inverse filter have the same output. The restored image obtained by using our approach is shown in Figure 6.2(d). In order to avoid the ringing effects due to the boundary conditions, we took 4 pixel wide boundaries, i.e. the first and last four rows and columns, from the original image and updated the interior region (248×248) of the image only. The noisy blurred image was used as an initial condition for accelerating the convergence. The constant λ was set at 0 because of small noise and good boundary values. The restored image in Figure 6.2(d) was obtained after 213 iterations. The square error (i.e. energy function) defined in (6.13) is 0.02543 and the square error between the original and the restored image is 66.5027.

6.7 Choosing Boundary Values

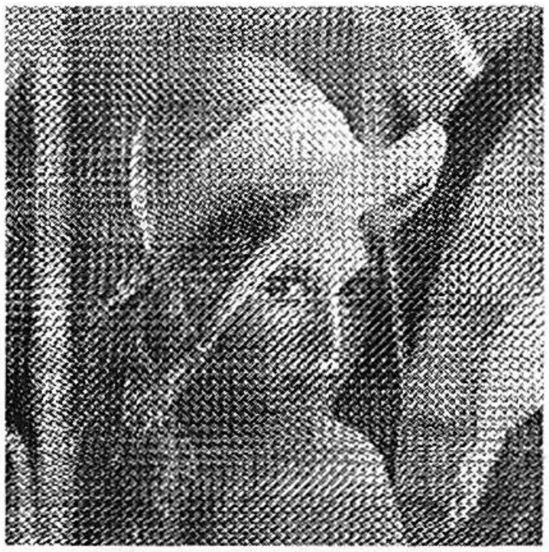
As mentioned in [98], choosing boundary values is a common problem for techniques ranging from deterministic inverse filter algorithms to stochastic Kalman filters. In these algorithms boundary values determine the entire solution when the blur is uniform [99]. The same problem occurs in the neural network approach. Since the 5×5 uniform blur function is ill conditioned, improper boundary values may cause ringing which may affect the restored image completely. For example, appending zeros to the image as boundary values introduces a sharp edge at the



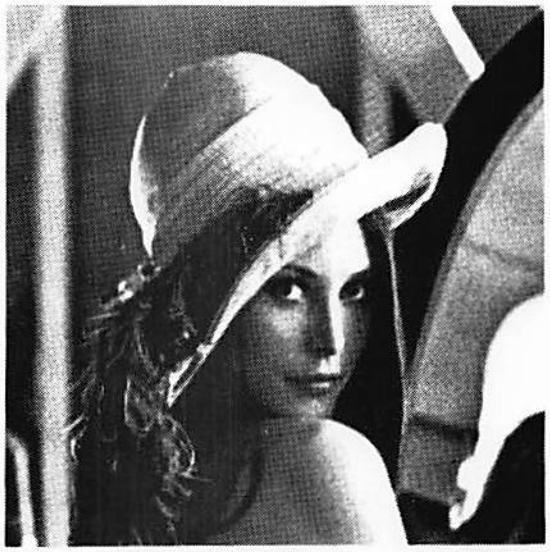
(a) Original girl image.



(b) Image degraded by 5×5 uniform blur and quantization noise.



(c) The restored image using inverse filter.



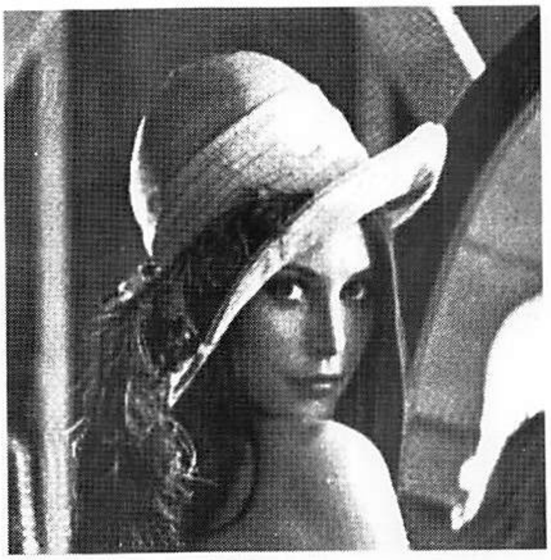
(d) The restored image using our approach.

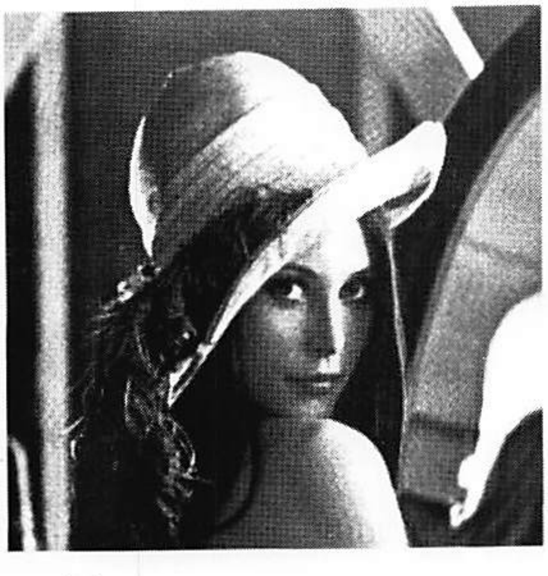
Figure 6.2: Restoration of noisy blurred real image.

image border and triggers ringing in the restored image even if the image has zero mean. Another procedure is to assume a periodic boundary. When the left (top) and right (bottom) borders of the image are different, a sharp edge is formed and ringing results even though the degraded image has been formed by blurring with periodic boundary conditions. The drawbacks of these two assumptions for boundary values were reported in [98, 90, 100] for the 2–D Kalman filtering technique. We also tested our algorithm using these two assumptions for boundary values; the results indicate the restored images were seriously affected by ringing.

In the last section, to avoid the ringing effect we took 4 pixel wide borders from the original image as boundary values for restoration. Since the original image is not available in practice, an alternative to eliminate the ringing effect caused by sharp false edges is to use the blurred noisy boundaries from the degraded image. Figure 6.3(a) shows the restored image using the first and last four rows and columns of the blurred noisy image in Figure 6.2(b), as boundary values. In the restored image there still exists some ringing due to the naturally occurring sharp edges in the region near the borders in the original image, but not due to boundary values. A typical cut of the restored image to illustrate ringing near the borders is shown in Figure 6.4. To remove the ringing near the borders caused by naturally occurring sharp edges in the original image, we suggest the following techniques.

First, divide the image into three regions: border, subborder and interior region as shown in figure 6.5. For 5×5 uniform blur case, the border region will be 4 pixels wide due to the boundary effect of the bias input $I_{i,k}$ in (6.18), and the subborder region will be 4 or 8 pixels wide. In fact, the width of subborder region will be image dependent. If the regions near the border are smooth, then the


(a) Blurred noisy boundaries.


(b) Method 1.

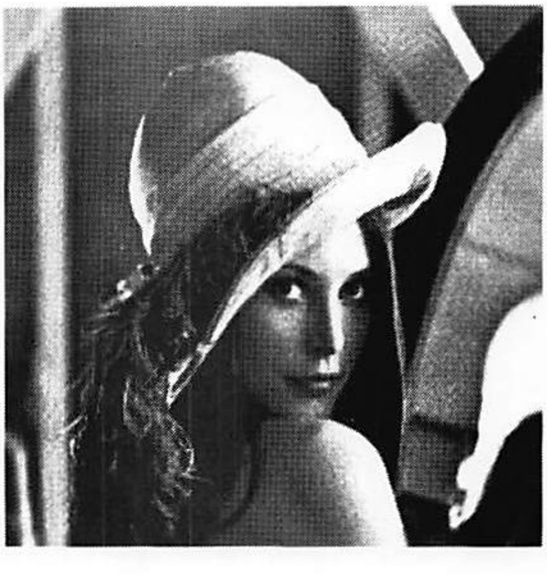

(c) Method 2.

Figure 6.3: Results using blurred noisy boundaries.

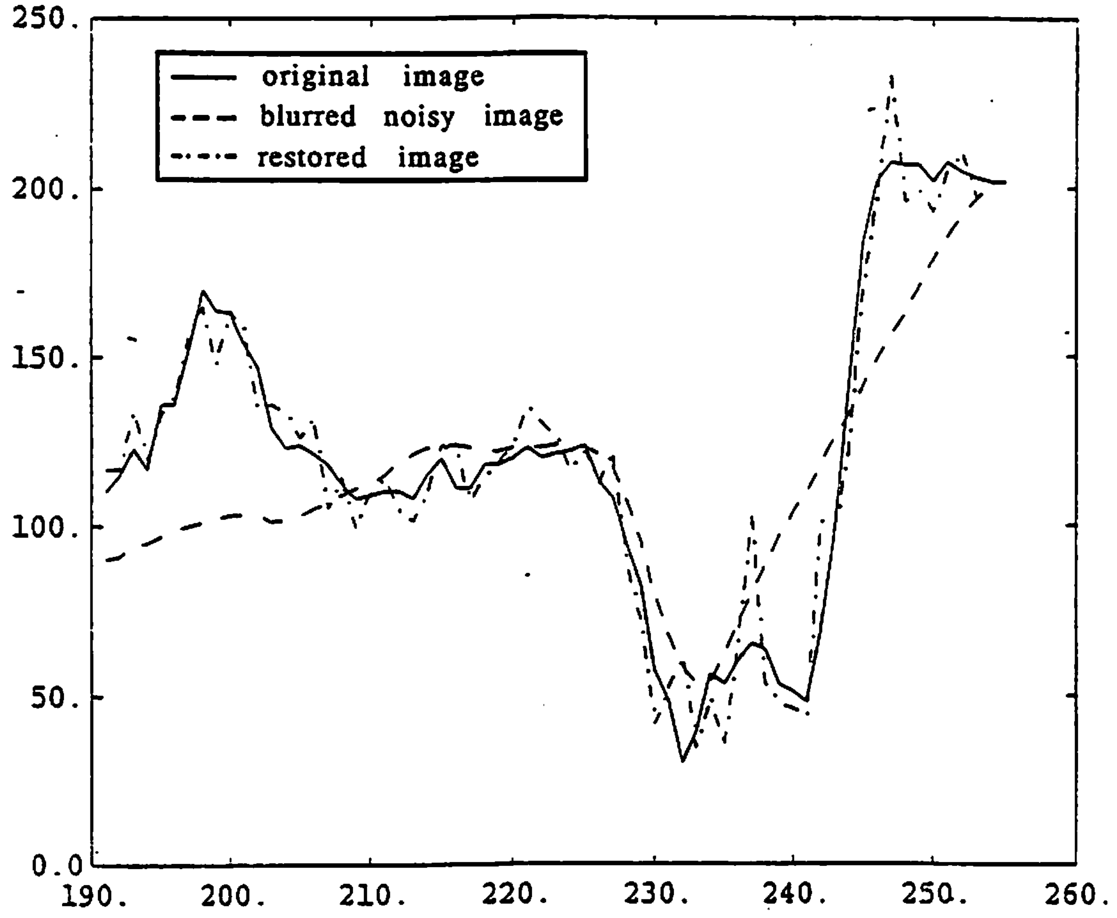


Figure 6.4: One typical cut of the restored image using the blurred noisy boundaries.

width of subborder region will be small or even zero. If the border contains many sharp edges the width will be large. For the real girl image, we chose the width of the subborder region to be 8 pixels. We suggest using one of the following two methods.

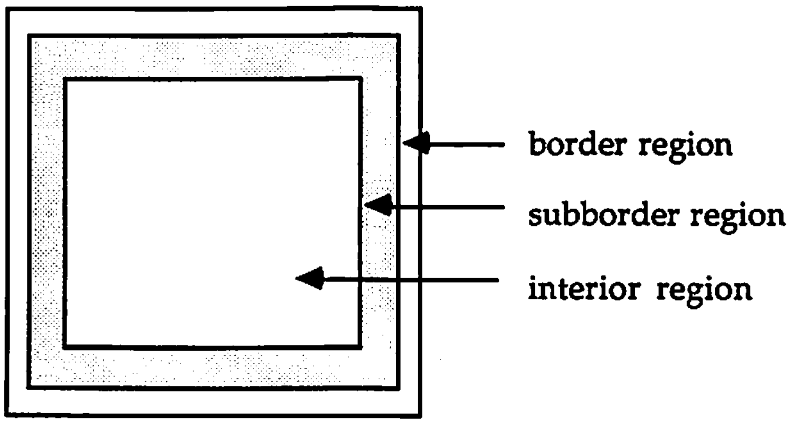


Figure 6.5: Border, subborder and interior regions of the image.

Method 1: In the case of small noise, such as quantization error noise, the blurred image is usually smooth. Therefore, we restricted the difference between the restored and blurred image in the subborder region to a certain range to reduce the ringing effect. Mathematically, this constraint can be written as

$$\|\hat{x}_i - y_i\| \leq T \quad for\ i \in\ subborder\ region, \tag{6.30}$$

where T is a threshold and $\hat{x}_i$ is the restored image gray value. Figure 6.3(b) shows the result of using this method with $T = 10$.

Method 2: This method simply sets λ in (6.13) to zero in the interior region and nonzero in the subborder region, respectively. Figure 6.3(c) shows the result of using this method with $\lambda = 0.09$. In this case, D was a Laplacian operator.

Owing to checking all restored image gray values in the subborder region, method 1 needs more computation than method 2. However, method 2 is very sensitive to the parameter λ while method 1 is not so sensitive to the parameter λ. Experimental results show that both methods 1 and 2 reduce the ringing effect significantly by using the suboptimal blurred boundary values.

6.8 Comparisons to Other Restoration Methods

Comparing the performance of different restoration methods needs some quality measures which are difficult to define owing to the lack of knowledge about the human visual system. The word "optimal" used in the restoration techniques usually refers only to a mathematical concept, and is not related to the response of the human visual system. For instance, when the blur function is ill conditioned and the SNR is low, the MMSE method improves the SNR, but the resulting image is not visually good. We believe that human objective evaluation is the best ultimate judgment. Meanwhile, the mean square error or least square error can be used as a reference.

For comparison purposes, we give the outputs of inverse filter, SVD pseudoinverse filter, MMSE filter and modified MMSE filter in terms of the Gaussian Markov random field (GMRF) model parameters [101, 93].

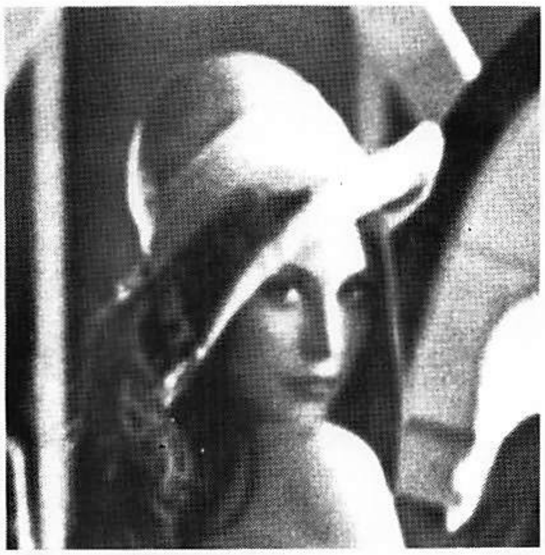
6.8.1 Inverse Filter and SVD Pseudoinverse Filter

An inverse filter can be used to restore an image degraded by a space invariant blur function with high signal to noise ratio. When the blur function has some singular points, an SVD pseudoinverse filter is needed; However both filters are very sensitive to noise. This is because the noise is amplified in the same way as the signal components to be restored. The inverse filter and SVD pseudoinverse filter were applied to an image degraded by the 5×5 uniform blur function and quantization noise (about 40 dB SNR). The blurred and restored images are shown in Figure 6.2(b) and 6.2(c), respectively. As we mentioned before the outputs of these filters are completely overridden by the amplified noise and ringing effects.

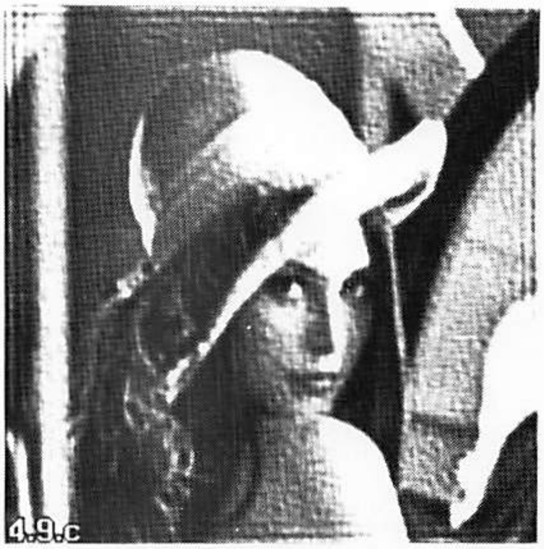
6.8.2 MMSE and Modified MMSE Filters

The MMSE filter is also known as the Wiener filter (in frequency domain). Under the assumption that the original image obeys a GMRF model, the MMSE filter (or Wiener filter) can be represented in terms of the GMRF model parameters and the blur function. In our implementation of the MMSE filter, we used a known blur function, unknown noise variance and the GMRF model parameters estimated from the blurred noisy image by a maximum likelihood (ML) method [101]. The image shown in Figure 6.6(a) was degraded by 5×5 uniform blur function and 20 dB SNR additive white Gaussian noise. The restored image is shown in Figure 6.6(b).

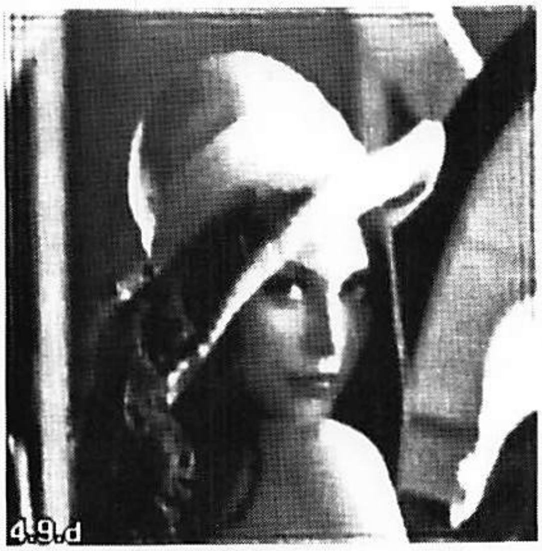
The modified MMSE filter in terms of the GMRF model parameters is a linear weighted combination of a Wiener filter with a smoothing operator (such as median filter) and a pseudoinverse filter to smooth the noise and preserve the edge of the restored image simultaneously. Details of this filter can be found in [93]. We applied the modified MMSE filter to the same image used in the MMSE filter above with the same model parameters. The smoothing operator is a 9×9 cross shape median filter. The resulting image is shown in Figure 6.6(c).



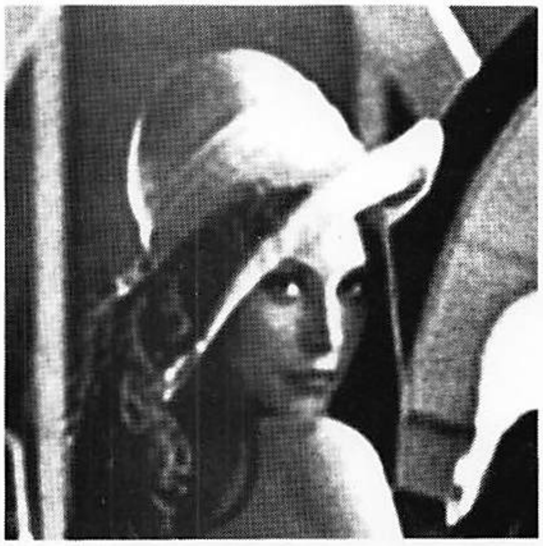
(a) Image degraded by 5×5 uniform blur and 20 dB SNR additive white Gaussian noise.



(b) The restored image using the MMSE filter.



(c) The restored image using the modified MMSE filter.



(d) The restored image using our approach.

Figure 6.6: Comparison to other restoration methods.

Method	MMSE	MMSE (o)	Modified MMSE	Neural network
MSE	1.384 dB	2.139 dB	1.893 dB	1.682 dB

Table 6.1: Mean square error (MSE) improvement.

The result of our method is also shown in Figure 6.6(d). The D we used in (6.13) is a Laplacian operator as in (6.14). We chose $\lambda = 0.0625$ and used 4 pixel wide blurred noisy boundaries for restoration. The total number of iterations was 20. The improvement of mean square error between the restored image and the original image for each method is shown in Table 6.1. In the table the "MMSE (o)" denotes that the parameters were estimated from the original image. The restored image using "MMSE (o)" is very similar to Figure 6.6(a). As we mentioned before, the comparison of the outputs of the different restoration methods is a difficult problem. The MMSE filter visually gives the worst output which has the smallest mean square error for MMSE(o) case. The result of our method is smoother than that of the MMSE filter. Although the output of the modified MMSE filter is smooth in flat regions, it contains some artifacts and snake effects at the edges, a consequence of using a large sized median filter.

6.9 Optical Implementation

To take advantage of the parallelism of the optics, we use a semi-synchronous neural network instead of an asynchronous one. The difference between the semi-synchronous and asynchronous neural networks is that at each clock cycle the former updates L^2 neurons selected from L^2 different gray level functions whereas the latter updates one neuron only. Without loss of generality, we assume that the neurons are semi-sequentially updated which is not similar to natural neuron transition rules. The semi-sequential updating means that if an $M \times L^2$ matrix is used to represent neurons (each element for one neuron and each column for one gray level function), then each row will be sequentially updated.

Figure 6.7 schematically shows a system capable of performing the semi-synchronous neural network consisting of the equations (6.1) and (6.2) with a deterministic decision rule. It is based on the idea described in [94] and [102] using an optical matirx-matrix product. The basic idea is as follows.

As we noted early, the matrix-matrix product (6.1) can be written as

$$u_{i,k} = \sum_{j}^{L^2} T_{i,\cdot;j,\cdot} \left(\sum_{l}^{M} v_{j,l}\right) + I_{i,\cdot} \qquad (6.31)$$

which is a vector-matrix product. We use a $M^2 \times (L^2+1)$ laser diode array shown in Figure 6.8 to represent the current state of neurons stored in the storage. To calculate (6.31), the light emitted by the laser diode array is collected vertically and then spread to a spatial light module (SLM) which represents the interconnection strengths and the bias inputs shown in Figure 6.8. After collecting the light emerging from each row of the SLM horizontally, an $L^2 \times 1$ photo detector array is used to detect the output. The elements of the output vector $\underline{U}_i$ are then entered in parallel to a threshold array to calculate (6.2). Another threshold array is used for computing the energy changes as soon as the output of the first threshold array becomes available. By feeding all outputs of the first and second threshold arrays to a decision array, the final result, a row of new state of neurons, is obtained and stored in the ith row of the storage chosen by a control. As a consequence, the ith row of the laser diode array is updated. In this system only the matrix-matrix (or matrix-vector) product is implemented optically and other

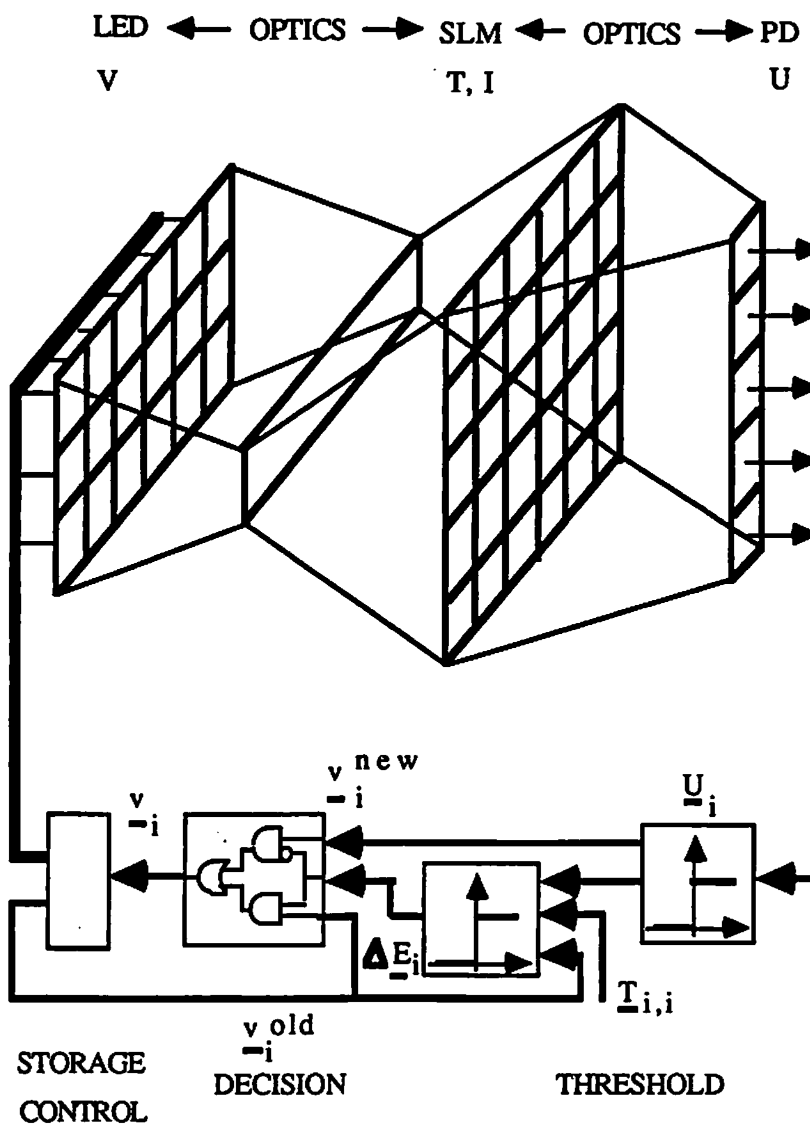


Figure 6.7: A schematic diagram for optical implementation of semi-synchronous neural network.

computations are done by the electronic circuits. To overcome the difficulty of the negative number representaton in optics, a coding scheme using two non-negative number to represent a bipolar number can be used here [103] [94].

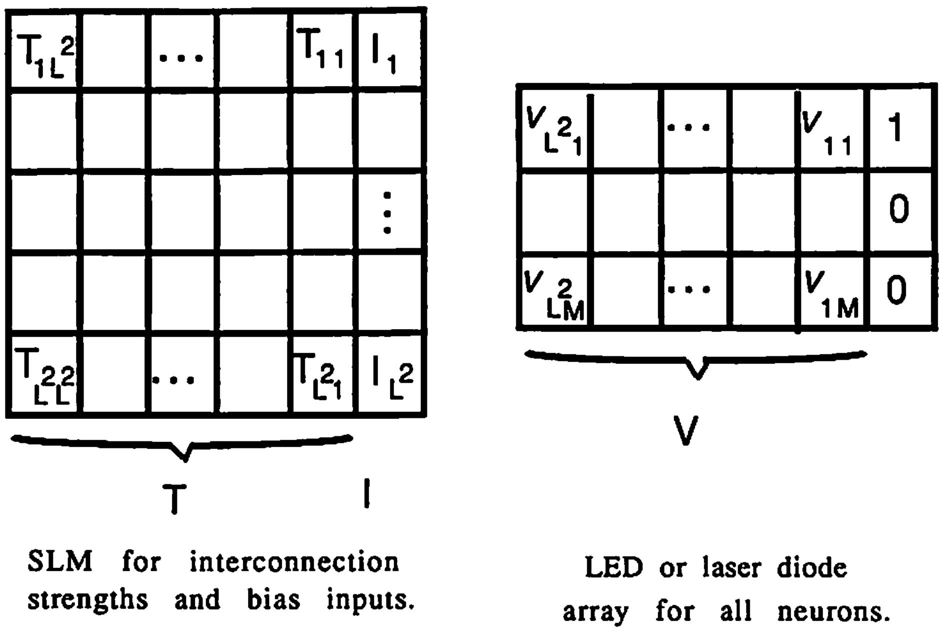

T
I
V
SLM for interconnection strengths and bias inputs.
LED or laser diode array for all neurons.

Figure 6.8: LED or laser diode array and SLM.

Since the dimensions of the array of neurons and the SLM are very large, we use holograms to implement the matrix-matrix (or matrix-vector) product. It is estimated [104] that with current technology as many as $(10^6 - 10^7)$ gates can be interconnected by using holograms. Figure 6.9 shows a schematic of a large size of matrix-matrix product using holograms. The first and the third holograms in the figure are computer generated holograms (CGH) which are used for interconnections to realize the summations in (6.31). The second hologram is used for the SLM to implement multiplication. Suppose that the blur function is shift invariant, all elements in the same column of the laser diode array are connected to the same $(2 \times K - 1)^2$ elements of the SLM array, where K is the window size of the blur function. Therefore, this is an $M \times L^2$ elements to $(2 \times K - 1)^2 \times L^2$ elements interconnection. The horizontal collection i.e the second summation at the last step of the matrix-matrix product is accomplished by the third hologram using a space variant interconnection method.

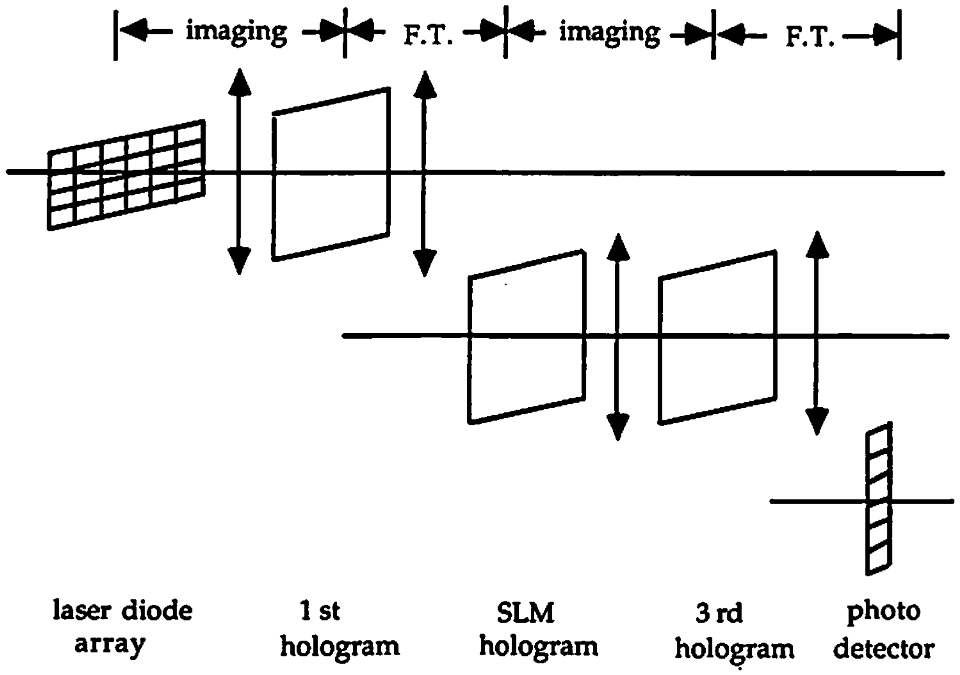


Figure 6.9: Schematic of large size matrix-matrix product using holograms.

In the optical implementation, L^2 neurons are updated at each clock cycle, thus, one iteration needs only M clock cycles for ML^2 neurons. The time complexity of optical implementaion is $O(MK)$, where K is the number of iterations. The space complexity is $O(ML^2)$ due to the total number of neurons.

6.10 Discussion

This chapter has introduced a new approach for the restoration of gray level images degraded by a shift invariant blur function and additive noise. The restoration procedure consists of two steps: parameter estimation and image reconstruction. In order to reduce computational complexity, a practical algorithm (Algorithm 2), which has equivalent results to the original one (Algorithm 1), is developed under the assumption that the neurons are sequentially visited. The image is generated iteratively by updating the neurons representing the image gray levels via a simple sum scheme. As no matrices are inverted, the serious problem of ringing due to the ill conditioned blur matrix H and noise overriding caused by inverse filter or pseudoinverse filter are avoided by using suboptimal boundary conditions. For the case of a 2-D uniform blur plus small noise, the neural network based approach gives high quality images compared to some of the existing methods. We see from the experimental results that the error defined by (6.13) is small while the error between the original image and the restored image is relatively large. This is because the neural network decreases energy according to (6.13) only. Another reason is that when the blur matrix is singular or ill conditioned, the mapping from $\underline{X}$ to $\underline{Y}$ is not one to one, therefore, the error measure (6.13) is not reliable anymore. In our experiments, when the window size of a uniform blur function is 3×3, the ringing effect was eliminated by using blurred noisy boundary values without any smoothness constraint. When the window size is 5×5, ringing effect was reduced with the help of the smoothing constraint and suboptimal boundary conditions.

Chapter 7

Conclusions and Future Research

7.1 Conclusions

An artificial neural network was presented and applied to some computer vision problems such as static and motion stereo, computation of optical flow and image restoration. To ensure convergence of the network, the deterministic decision rule was used in all the algorithms. Experimental results using natural images confirm that neural networks provide simple but very efficient means to solve computer vision problems, especially at the low level. Experimental results also provide a strong support to the hypothesis that the first order derivatives of the intensity function are appropriate measurement primitives for stereo matching and the principle curvatures are useful for computing optical flow. The utilization of multiple frames for computing depth and optical flow gives much better results and is useful for robot vision applications. Since no matrices are inverted during restoration, the serious problem of ringing due to the ill conditioned blur matrix is avoided and hence the neural network algorithm gives high quality images compared to some of the existing methods. Although this artificial neural network has been applied to only a few low level computer vision problems so far, it is potentially useful for many computer vision problems.

7.2 Future Research

As pointed out in the discussion section of previous chapters, some topics for future study became apparent during the course of this research. For instance, how to detect motion discontinuities when the motion is translational and rotational. How to restore the blurred and noisy images when the SNR is very low. How to use stochastic decision rule to find a global optimal solution to these problems, etc. However, the long-term goal of this research is to develop an artificial neural network vision system for the recognition of objects in a 3-D scene, which is useful for robot manipulation and vehicle navigation. (A simple example of using a network called schemas for robot hand control can be found in [105].) The system will make use of parallelism at all levels to achieve real-time vision in complex enviroments. There exist many vision systems such as ACRONYM [106], 3D MOSAIC [107], VISIONS [108] etc. emphasizing different aspects of the 3-D object recognition problem. But none of them were able to match human performance. A typical system has the following structure shown in Figure 7.1. The first part is to preprocess the given digital image to remove system and statistical degradations i.e. image restoration. The second part is to extract features such as edges, lines, shapes, optical flow, etc. and to segment the image into connected regions "homogeneous" in some sense. The third part is to resegment the image based on various geometric criteria, to measure various properties of and relations among the regions, and to establish a relational structure i.e. a labeled scene graph according to the properties and relations of the image parts (regions). The last part is to match models and to recognize the objects by finding subgraphs of the scene graph satisfying the constraints defined by the object graph. To construct such vision system using neural networks, many interesting and promising research topics should be pursued. Here we discuss a few of them.

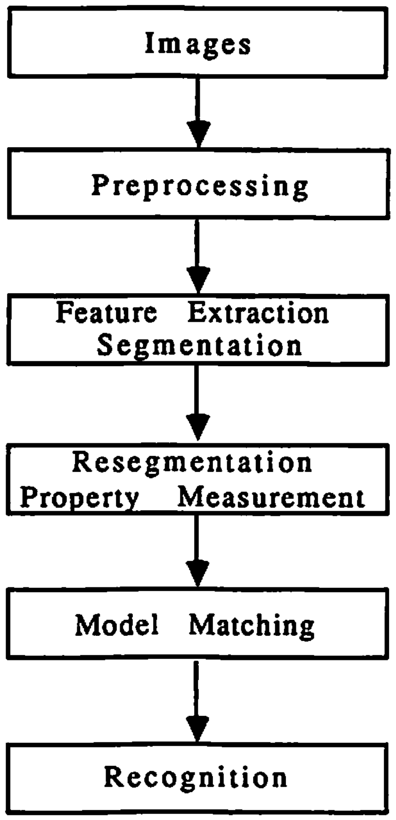


Figure 7.1: Vision system structure.

For feature extraction, naturally edge detection is the first thing to think of. There are many types of edges, such as step, roof, line and ramp edges. One possible way to detect edges by using a neural network is to formulate the edge detection problem as a pattern recognition problem. For example, step edges with different orientations may be considered as different patterns. The neural network can then be trained to remember all these patterns. When observations are fed into the network, an edge will be detected if the observation matches one of the patterns.

Although neural network based approach to texture segmentation is not new [109], it is still required to develop a rotation invariant method. As mentioned in the previous chapter, human eye is very sensitive to intensity changes. We may use the intensity derivatives to construct some rotation invariant statistics and then segment the image based on such statistics. Moreover, segmentation based on geometric structure of surfaces is also a good research topic.

Another important issue is to compute shapes, or Shape from x. Our static stereo and motion stereo algorithms are for computing shapes from stereo and motion images, respectively. Shape from shading is to find the needle map and then to recover the depth map or directly recover depth without computing the reflectance map. If the needle map is known, a neural network can be used to recover the depth $Z(i,j)$ by minimizing the following cost function

$$\begin{aligned}\mathbf{E} \;=\; & \sum_{i=1}^{N_r}\sum_{j=1}^{N_c}\{[\frac{1}{2}(Z(i,j+1)-Z(i,j)+Z(i+1,j+1)-Z(i+1,j))-p(i,j)]^2 \\ & +[\frac{1}{2}(Z(i+1,j)-Z(i,j)+Z(i+1,j+1)-Z(i,j+1))-q(i,j)]^2 \\ & +\lambda[(Z(i,j+1)-Z(i,j))^2+(Z(i+1,j)-Z(i,j))^2]\} \qquad (7.1)\end{aligned}$$

where $p(i,j)$ and $q(i,j)$ are the gradients at point (i,j), λ is a control parameter. The depth $Z(i,j)$ is represented by a simple sum of neuron state variables which may take real values. Although computing the needle map is a difficult problem because of the nonlinearity, it is still possible to use a higher order neural network to solve it.

The neural network is also potentially useful for shape from texture. Under paraperspective projection, shape from texture can be formulated as a linear least squares problem [110], which the neural network is good to handle. When perspective projection is introduced, shape from texture becomes more difficult because of the nonlinear equation to be solved. However, by using the Fourier transform or the Wigner distribution, shape from texture can be simplified as a linear least squares problem [111, 112].

One very challenging problem is how to develop a neural network algorithm to compute shape from a combination of the information from different image cues such as stereo, motion, texture, shading and contour. If one thinks of these cues as some constraints, then this can be done using a neural network. Since the neural network can take into account multiple and mutual constraints simultaneously, it is a promising approach to the integration problem.

Since graph matching is a natural process in a neural network, definitely neural network based approach will give a good solution. Recently, von der Malsburg [113] has presented a shift invariant method for labeled graph matching using a multi-layer neural network. This method uses local features such as edges, corners, gray level etc. and their neighborhood relationships to construct a labeld graph. Distortions such as partly hidden are not considered. For 3-D matching, the major problem is that the observed image parts may not correspond to the object parts due to changes in perspective such as occlusion and segmentation error such as incorrect merging and splitting. We have to develop an error tolerant matching algorithm to solve this problem. Principally, this algorithm will use object dependent features to recognize objects based on subgraph matching techniques.

References

[1] J. Hadamard, "Sur les Problèmes aux Dérivées Partielles et leur Signification Physique", *Princeton University Bulletin*, vol. 13, 1902.

[2] V. A. Morozov, *Methods for Solving Incorrectly Posed Problems*, Springer-Verlag, New York, 1984.

[3] M. A. Arbib and A. R. Hanson, "Vision, Brain, and Cooperative Computation: An Overview", In M. A. Arbib and A. R. Hanson, editors, *Vision, Brain, and Cooperative Computation*, pp. 1–83, The MIT Press, 1987.

[4] S. Amari, "Characteristics of Randomly Connected Threshold-Element Networks and Network Systems", *Proceedings IEEE*, vol. 59, pp. 35–47, Jan. 1971.

[5] S. Amari, "Neural Theory of Association and Concept Formation", *Biological Cybernetics*, vol. 26, pp. 175–185, 1977.

[6] J. J. Hopfield and D. W. Tank, "Neural Computation of Decisions in Optimization Problems", *Biological Cybernetics*, vol. 52, pp. 141–152, 1985.

[7] D. Marr and T. Poggio, "Cooperative Computation of Stereo Disparity", *Science*, vol. 194, pp. 283–287, October 1976.

[8] N. M. Grzywacz and A. L.Yuille, "Motion Correspondence and Analog Networks", In *Proc. Conf. on Neural Networks for Computing*, pp. 200–205, Snowbird, UT, 1986, American Institute of Physics.

[9] C. V. Stewar and C. R. Dyer, "A Connectionist Model for Stereo Vision", In *Proc. IEEE First Annual Intl. Conf. on Neural Networks*, San Diego, CA, June 1987.

[10] G. Z. Sun, H. H. Chen, and Y. C. Lee, "Learning Stereopsis with Neural Networks", In *Proc. IEEE First Annual Intl. Conf. on Neural Networks*, San Diego, CA, June 1987.

[11] A. F. Gmitro and G. R. Gindi, "Optical Neurocomputer for Implementation of the Marr–Poggio Stereo Algorithm", In *Proc. IEEE First Annual Intl. Conf. on Neural Networks*, San Diego, CA, June 1987.

[12] L. Matthies, R. Szeliski, and T Kanade, "Kalman Filter-Based Algorithms for Estimating Depth from Image Sequences", In *Proc. DARPA Image Understanding Workshop*, pp. 199–213, Cambridge, MA, April 1988.

[13] C. Koch, "Analog Neuronal Networks for Real-Time Vision Systems", In *Proc. Workshop on Neural Network Devices and Applications*, Los Angeles, CA, February 1987.

[14] Y. T. Zhou and R. Chellappa, "Stereo Matching Using a Neural Network", In *Proc. Intl. Conf. on Acoustics, Speech, and Signal Processing*, pp. 940–943, New York, NY, April 1988.

[15] Y. T. Zhou and R. Chellappa, "A Neural Network Approach to Stereo Matching", In *SPIE's 32nd Annual Intl. Tech. Symposium on Optical and Optoelectronic Applied Science and Engineering*, San Diego, CA, August 1988.

[16] Y. T. Zhou and R. Chellappa, Stereo Matching Using a Neural Network, Technical Report USC SIPI Report 124, Signal and Image Processing Institute, Univ. of Southern California, March 1988.

[17] Y. T. Zhou and R. Chellappa, "Computation of Optical Flow Using a Neural Network", In *Proc. IEEE Intl. Conf. on Neural Networks*, volume vol. 2, pp. 71–78, San Diego, CA, July 1988.

[18] Y. T. Zhou and R. Chellappa, "Computation of Optical Flow Using a Neural Network", In *Intl. Neural Network Society First Annual Meeting*, Boston, MA, Sept. 1988.

[19] Y. T. Zhou, R. Chellappa, and B. K. Jenkins, "A Novel Approach to Image Restoration Based on a Neural Network", In *Proc. IEEE First Annual Intl. Conf. on Neural Networks*, pp. 269–279, San Diego, CA, June 1987.

[20] Y. T. Zhou, R. Chellappa, and B. K. Jenkins, Image Restoration Using a Neural Network, Technical Report USC SIPI Report 112, Signal and Image Processing Institute, Univ. of Southern California, July 1987.

[21] Y. T. Zhou, R. Chellappa, A. Vaid, and B. K. Jenkins, "Image Restoration Using a Neural Network", *IEEE Trans. Acoust,Speech,Signal Processing*, vol. 36, pp. 1141–1151, July 1988.

[22] W. S. McCulloch and W. Pitts, "A Logical Calculus of the Ideas Immanent in Nervous Activity", *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.

[23] D. O. Hebb, *The Organization of Behavior*, Wiley, New York, 1949.

[24] F. Rosenblett, "Two Theorems of Statistical Separability in the Perceptron", In *Mechanisation of Thought Processes: Proceedings of a Symposium Held at the National Physical Laboratory*, pp. 421–456, HM Stationery Office, London, November, 1959.

[25] F. Rosenblett, *Principles of Neurodynamics*, Spartan, New York, 1962.

[26] O. G. Selfridge, "Pattern Recognition in Modern Computers", In *Proc. of the Western Joint Computer Conference*, 1955.

[27] B. Widrow and M. E. Hoff, "Adaptive Switching Circuits", In *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record*, volume Part 4, pp. 96–104, August 1960.

[28] J. A. Anderson, "Neural Models with Cognitive Implications", In D. LaBerge and S. J. Samuels, editors, *Basic Processes in Reading Perception and Comprehension*, pp. 27–90, Erlbaum, 1977.

[29] S. Amari, "Learning Patterns and Pattern Sequences by Self-Organizing Nets of Threshold Elements", *IEEE Trans. on Computer*, vol. C-21, pp. 1197–1206, Nov. 1972.

[30] S. Amari and M. A. Arbib, "Competition and Cooperation in Neural Nets", In J. Metzler, editor, *Systems Neuroscience*, pp. 119–165, Academic Press, 1977.

[31] M. A. Arbib, *Brains, Machines, and Mathematics*, McGraw-Hill, New York, New York, 1964.

[32] C. von der Malsburg, "Self-Organization of Orientation Sensitive Cells in the Striate Cortex", *Kybernetik*, vol. 14, pp. 85–100, 1973.

[33] K. Fukushima, "Cognitron: A Self-Organizing Multilayered Neural Network", *Biological Cybernetics*, vol. 20, pp. 121–136, 1975.

[34] S. Grossberg, "Adaptive Pattern Calssification and Universal Recording: Part I. Parallel Development and Coding of Neural Feature Detectors", *Biological Cybernetics*, vol. 23, pp. 121–134, 1976.

[35] T. Kohonen, *Associative Memory: A System Theoretical Approach*, Springer, New York, 1977.

[36] J. A. Feldman and D. H. Ballard, "Connectionist Models and Their Properties", *Cognitive Science*, vol. 6, pp. 205–254, 1982.

[37] D. E. Rumelhart and J. L. McClelland, "An Interactive Activation Model of Context Effects in Letter Perception: Part 2. The Contextual Enhancement Effect and Some Tests and Extensions of the Model", *Psychological Review*, vol. 89, pp. 75–112, 1982.

[38] J. J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", *Proc. Natl. Acad. Sci. USA*, vol. 79, pp. 2554–2558, April 1982.

[39] J. J. Hopfield, "Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons", *Proc. Natl. Acad. Sci. USA*, vol. 81, pp. 3088–3092, May 1984.

[40] J. Sejnowski and C. R. Rosenberg, NETtalk: A Parallel Network That Learns to Read Aloud, Technical Report JHU-EECS-86-01, Johns Hopkins Univ., 1986.

[41] J. P. LaSalle, *The Stability and Control of Discrete Processes*, Springer-Verlag, New York, New York, 1986.

[42] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equations of State Calculations by Fast Computing Machines", *J. Chem. Phys.*, vol. 21, pp. 1087–1091, 1953.

[43] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Stimulated Annealing", *Science*, vol. 220, pp. 671–680, 1983.

[44] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and Bayesian Restoration of Images", *IEEE Trans. on Patt. Anal. and Mach. Intel.*, vol. PAMI-6, pp. 141–149, November 1984.

[45] S. T. Barnard, "A Stochastic Approach to Stereo Vision", In *Proc. Fifth National Conf. on Artificial Intelligence*, Philadelphia, PA, August 1986.

[46] W. E. L. Grimson, *From Images to Surfaces*, The MIT press, Cambridge, MA, 1981.

[47] G. Medioni and R. Nevatia, "Segment-Based Stereo Matching", *Computer Vision, Graph. and Image Processing*, vol. 31, pp. 2–18, 1985.

[48] J. E. W. Mayhew and J. P. Frisby, "Psychophysical and Computational Studies towards a Theory of Human Stereopsis", *Artificial Intelligence*, vol. 17, pp. 349–385, August. 1981.

[49] D. Marr and E. C. Hildreth, "Theory of Edge Detection", *Proc. Royal Society of London*, vol. B-207, pp. 187–217, February 1980.

[50] R. Nevatia and K. R. Babu, "Linear Feature Extraction and Description", *Computer Graph. and Image Processing*, vol. 13, pp. 257–269, 1980.

[51] W. E. L. Grimson, "Computational Experiments with a Feature Based Stereo Algorithm", *IEEE Trans. on Patt. Anal. and Mach. Intel.*, vol. PAMI-7, pp. 17–34, January 1985.

[52] W. Hoff and N. Ahuja, "Extracting Surfaces from Stereo Images: An Integrated Approach", In *Proc. First International Conf. on Computer Vision*, pp. 284–294, London, England, June 1987.

[53] B. Julesz, *Foundations of Cyclopean Perception*, The University of Chicago Press, Chicago, IL, 1971.

[54] H. R. Wilson and S. C. Giese, "Threshold Visibility of Frequency Grating Patterns", *Vision Research*, 17, pp. 1177–1190, 1977.

[55] H. R. Wilson and J. R. Bergen, "A Four Mechanism Model for Threshold Spatial Vision", *Vision Research*, 19, pp. 19–32, 1979.

[56] P.Dev, "Perception of Depth Surfaces in Random-dot Stereogram: a Neural Model", *Int. J. Man-Machine Studies*, 7, pp. 511–528, 1975.

[57] T. Poggio, "Vision by Man and Machine", *Scientific American*, vol. 250, pp. 106–116, April. 1984.

[58] G. G. Lorentz, *Approximation of functions*, Holt, Rinehart and Winston, New York, 1966.

[59] P. Beckmann, *Orthogonal Polynomials for Engineers and Physicists*, Golem, Boulder, CO, 1973.

[60] R. M. Haralick, "Digital Step Edges from Zero Crossings of Second Directional Derivatives", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. PAMI-6, pp. 58–68, January 1984.

[61] T. J. Laffey, R. M. Haralick, and L. T. Watson, "Topographic Classification of Digital Image Intensity Surfaces", In *The proc. IEEE Workshop on Computer Vision: Theory and Control*, Rindge, New Hampshire, August 1982.

[62] D. Marr, *Vision*, W. H. Freeman and Company, New York, 1982.

[63] S. D. Blostein and T. Huang, "Quantization Errors in Stereo Triangulation", In *Proc. First International Conf. on Computer Vision*, pp. 325–334, London, England, June 1987.

[64] B. K. P. Horn, *Robot Vision*, The MIT Press, Cambridge, Massachusetts, 1986.

[65] D. Marr, T. Poggio, and E. C. Hildreth, "The Smallest Channel in Early Human Vision", *J. Opt. Soc. Am.*, vol. 90, pp. 868–870, 1979.

[66] B. Julesz, "Binocular Depth Perception of Computer–Generated Patterns", *Bell System Technical J.*, vol. 39, pp. 1125–1162, Sept. 1960.

[67] R. Nevatia, "Depth Measurement by Motion Stereo", *Computer Graph. and Image Processing*, vol. 6, pp. 619–630, 1976.

[68] T. Williams, "Depth from Camera Motion in a Real World Scene", *IEEE Trans. on Patt. Anal. and Mach. Intel.*, vol. PAMI-2, pp. 511–516, November 1980.

[69] R. Jain, S. L. Bartlett, and N. O'Brien, "Motion Stereo Using Ego-Motion Complex Logarithmic Mapping", *IEEE Trans. on Patt. Anal. and Mach. Intel.*, vol. PAMI-9, pp. 356–369, May 1987.

[70] H. Itoh, A Miyauchi, and S. Ozawa, "Distance Measuring Method Using Only Simple Vision Constrained for Moving Robots", In *Seventh Intl. Conf. Pattern Recognition*, pp. 192–195, Montreal, July 1984.

[71] G. Xu, S. Tsuji, and M. Asada, "A Motion Stereo Method Based on Coarse-to-Fine Control Strategy", *IEEE Trans. on Patt. Anal. and Mach. Intel.*, vol. PAMI-9, pp. 332–336, March 1987.

[72] J. O. Limb and J. A. Murphy, "Estimating the Velocity of Moving Objects from Television Signals", *Computer Graph. and Image Processing*, vol. 4, pp. 311–329, 1975.

[73] A. N. Netravali and J. D. Robbins, "Motion-Compensated Television Coding: Part I", *The Bell System Technique Journal*, vol. 58, pp. 631–670, 1979.

[74] A. N. Netravali and J. D. Robbins, "Motion-Compensated Coding: Some New Results", *The Bell System Technique Journal*, vol. 59, pp. 1735–1745, 1980.

[75] C. L. Fennema and W. B. Thompson, "Velocity Determination in Scene Containing Several Moving Objects", *Computer Graph. and Image Processing*, vol. 9, pp. 301–315, 1979.

[76] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow", *Artificial Intelligence*, vol. 17, pp. 185–203, August 1981.

[77] M. A. Gennert and S. Negahdaripour, Relaxing the Brightness Constancy Assumption in Computing Optical Flow, Technical Report AI Memo No. 975, MIT Artificial Intelligence Lab., June 1987.

[78] W. B. Thompson, "Lower-Level Estimation and Interpretation of Visual Motion", *Computer*, pp. 20–28, August 1981.

[79] E. C. Hildreth, *The Measurement of Visual Motion*, The MIT Press, Cambridge, Massachusetts, 1983.

[80] H. H. Nagel, "Displacement Vectors Derived from Second Order Intensity Variations in Image Sequence", *Computer Vision, Graph. and Image Processing*, vol. 21, pp. 85–117, 1983.

[81] J. M. Prager and M. A. Arbib, "Computing the Optical Flow: The MATCH Algorithm and Prediction", *Computer Vision, Graphics and Image Processing*, vol. 24, pp. 271–304, 1983.

[82] N. M. Grzywacz and A. L.Yuille, Massively Parallel Implementations of Theories for Apparent Motion, Technical Report AI Memo No. 888, CBIP Memo No. 016, MIT Artificial Intelligence Lab. and Center for Biological Information Processing, June 1987.

[83] S. Ullman, *The Interpretation of Visual Motion*, M.I.T. Press, Cambridge, MA, 1979.

[84] J. Hutchinson, C Koch, J Luo, and C. Mead, "Computing Motion Using Analog and Binary Resistive Networks", *IEEE Computer Magazine*, pp. 52–63, March 1988.

[85] B. O'Neill, *Elementary Differential Geometry*, Academic Press, New York, NY, 1966.

[86] J. L. Potter, "Velocity as a Cue to Segmentation", *IEEE Trans. Systems, Man, and Cybernetics*, vol. SMC-5, pp. 390–394, 1975.

[87] W. B. Thompson, "Combining Motion and Contrast for Segmentation", *IEEE Trans. on Patt. Anal. and Mach. Intel.*, vol. PAMI-2, pp. 543–549, 1980.

[88] W. B. Thompson, K. M. Mutch, and V. A. Berzins, "Dynamic Occlusion Analysis in Optical Flow Fields", *IEEE Trans. on Patt. Anal. and Mach. Intel.*, vol. PAMI-7, pp. 374–383, July 1985.

[89] H. C. Andrews and B. R. Hunt, *Digital Image Restoration*, Prentice-Hall, Englewood Cliffs, New Jersey, 1977.

[90] J. W. Woods and V. K. Ingle, "Kalman Filtering in Two Dimensions: Further Results", *IEEE Trans. Acoust,Speech,Signal Processing*, vol. ASSP-29, pp. 188–197, April. 1981.

[91] W. K. Pratt, *Digital Image Processing*, John Wiley and Sons, New York, 1978.

[92] R. Chellappa and R. L. Kashyap, "Digital Image Restoration Using Spatial Interaction Models", *IEEE Trans. Acoust,Speech,Signal Processing*, vol. ASSP-30, pp. 461–472, June. 1982.

[93] H. Jinchi and R. Chellappa, "Restoration of Blurred and Noisy Image Using Gaussian Markov Random Field Models", In *Proc. Conf. on Information Science and System*, pp. 34–39, Princeton Univ., NJ, 1986.

[94] N. H. Farhat, D. Psaltis, A. Prata, and E. Paek, "Optical Implementation of the Hopfield Model", *Applied Optics*, vol. 24, No.10, pp. 1469–1475, 15 May 1985.

[95] M. Takeda and J. W. Goodman, "Neural Networks for Computation: Number Representations and Programming Complexity", *Applied Optics*, vol. 25, No. 18, pp. 3033–3046, Sept. 1986.

[96] T. Poggio, V. Torre, and C. Koch, "Computational Vision and Regularization Theory", *Nature*, vol. 317, pp. 314–319, Sept. 1985.

[97] W. K. Pratt, O. D. Faugeras, and A. Gagalowicz, "Visual Discrimination of Stochastic Texture Fields", *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, pp. 796–814, Nov. 1978.

[98] J. W. Woods, J. Biemond, and A. M. Tekalp, "Boundary Value Problem in Image Restoration", In *Proc. Intl. Conf. on Acoustics, Speech, and Signal Processing*, pp. 692–695, Tampa, FL, March 1985.

[99] M. M. Sondhi, "The Removal of Spatially Invariant Degradations", *Proc. of IEEE*, vol. 60, pp. 842–853, July 1972.

[100] J. Biemond, J. Rieske, and J. Gerbrand, "A Fast Kalman Filter for Images Degraded by Both Blur and Noise", *IEEE Trans. Acoust,Speech,Signal Processing*, vol. ASSP-31, pp. 1248–1256, October. 1983.

[101] R. Chellappa and H. Jinchi, "A Nonrecursive Filter for Edge Preserving Image Restoration", In *Proc. Intl. Conf. on Acoustics, Speech, and Signal Processing*, pp. 652–655, Tampa, FL, March 1985.

[102] Y. S. Abu-Mostafa and D. Psaltis, "Optical Neural Computers", *Scientific American*, pp. 88–96, May 1987.

[103] J. W. Goodman, "The Optical Data Processing Family Tree", *Opt. News*, vol. 25, 1984.

[104] A. A. Sawchuk and B. K. Jenkins, "Dynamic Optical Interconnections for Parallel Processors", In *Proc. of Optical Computing Symposium on Opto-electronics and Laser Applications in Science and Engineering*, volume SIPE vol. 625, Los Angeles, CA, January 1986.

[105] M. A. Arbib, T. Iberall, and D. Lyons, "Schemas That Integrate Vision and Touch for Hand Control", In M. A. Arbib and A. R. Hanson, editors, *Vision, Brain, and Cooperative Computation*, pp. 489–510, The MIT Press, 1987.

[106] R. A. Brooks, *Model-Based Computer Vision*, M.I.T. Press, Cambridge, MA, 1981.

[107] M. Herman and T Kanade, "The 3D MOSAIC Scene Understanding System: Incremental Reconstruction of 3D Scenes from Complex Images", In *Proc. DARPA Image Understanding Workshop*, Norwood, NJ, 1984.

[108] R. Belknap, E. Riseman, and A. Hanson, "The Information Fusion Problem and Rule-Based Hypotheses Applied to Complex Aggregations of Image Events", In *Proc. DARPA Image Understanding Workshop*, Miami Beach, FL, 1985.

[109] A. R. Hanson and E. M. Riseman, "Segmentation of Natural Scenes", In A. R. Hanson and E. M. Riseman, editors, *Computer Vision Systems*, Academic Press, 1978.

[110] J. Aloimonos, "Shape from Texture", *Biological Cybernetics*, vol. 58, pp. 345–360, 1988.

[111] R. Bajcsy and L Lieberman, "Texture Gradient as a Depth Cue", *Computer Graphics and Image Processing*, vol. 5, pp. 52–67, 1976.

[112] Y. C. Jau and R. T. Chin, "Shape from Texture Using the Wigner Distribution", In *The Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 515–523, Ann Brbor, MI, 1988.

[113] C. von der Malsburg, "Pattern Recognition by Labeled Graph Matching", *Neural Networks*, vol. 1, Number 2, pp. 141–148, 1988.