National Library of Canada
Acquisitions and Bibliographic Services Branch
395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada
Direction des acquisitions et des services bibliographiques
395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

MONOCULAR OPTICAL FLOW FOR REAL-TIME VISION SYSTEMS

Stephen M. Benoit

Department of Electrical Engineering
McGill University

March 1996

A Thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfilment of the requirements of the degree of
Master of Engineering

ISBN 0-612-12110-0

Canada

Abstract

This thesis introduces a monocular optical flow algorithm that has been shown to perform well at nearly real-time frame rates (4 FPS) on natural image sequences. The system is completely bottom-up, using pixel region-matching techniques. A coordinated gradient descent method is broken down into two stages; pixel region matching error measures are locally minimized, and flow field consistency constraints apply non-linear adaptive diffusion, causing confident measurements to influence their less confident neighbors. Convergence is usually accomplished with one iteration for an image frame pair. Temporal integration and Kalman filtering predicts upcoming flow fields and figure/ground separation. The algorithm is designed for flexibility: large displacements are tracked as easily as sub-pixel displacements, and higher-level information can feed flow field predictions into the measurement process.

Résumé

Cette thèse introduit un algorithme de flot optique monoculaire qui a été appliqué avec succès sur des séquences d'images de scènes naturelles à des fréquences video presque temps réel. Le système utilise une approche de bas niveau s'appuyant principalement sur des techniques de comparaison des regions de pixels. Une méthode de descente de gradient collaborative est séparée en deux étapes; l'erreur de comparaison des regions est minimisée localement, et les contraintes de compatibilité des champs de flot appliquent une diffusion adaptive non-linéaire, permettant uax regions de grande compatibilité d'influencer leurs voisins. La convergence est habituellement atteinte à la première itération pour une paire d'images. L'intégration temporelle et l'utilisation de filtres Kalman prédisent les champs de flot et la séparation objet versus arrière-scène. L'algorithme est conçu avec un critère de flexibilité; les grands déplacements sont perçus aussi facilement que ceux de moins d'un pixel, et les informations de plus haut niveau peuvent fournir une assistance à la procèdure de mesure.

ACKNOWLEDGEMENTS

I thank my supervisor, Frank P. Ferrie, for the key suggestions, direction and resources at the Center for Intelligent Machines that made this research possible. I would also like to acknowledge and thank Gilbert Soucy, research engineer at the Artificial Perception Lab for technical support and for providing expert advice on implementation feasibility and planning. Thanks also goes out to the members of the Artificial Perception Lab, including Peter Whaite, Francesco Callari and Tal Arbel for general guidance and exchange of ideas. To my parents, whose moral support was ever-present, I am very grateful.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

1. Brief Proposal

In this thesis we describe a fast monocular optical flow algorithm for real-time applications that features rapid convergence (within a single iteration), ease of temporal integration, and swift reaction to abrupt change in scene motion. Although the flow data are represented on a coarse grid, the quantitative and qualitative flow for natural scenes is as good as or better than algorithms in the same class.

The system is completely bottom-up, but does incorporate predictions from higher-level processes. In the current implementation a Kalman filter is used to predict upcoming flow fields and to perform figure/ground separation. The heart of the algorithm is a coordinated gradient descent method that alternately minimizes local correspondence errors and the consistency of adjacent flow field vectors. What results is a non-linear adaptive diffusion in which confident measurements are used to influence their less confident neighbours.

The principles used to formulate the algorithm follow directly from an approximate model of the hypercolumn organization present in the primate visual cortex. This model suggests a process in which scalar and vector information are processed independently and how they might be combined to produce a coarse flow field that is both accurate and robust. The optimization of region correspondences and flow field consistency, for example, are implemented as separate minimization stages as in the biological case instead of one lumped minimization.

2. Organization of this Thesis

Optical flow does not exist in a vacuum. This thesis therefore begins in Chapter 2 with a discussion of the motivating applications of motion from image sequences, and the techniques used to implement them. The reader is guided through a brief outline of the

biological motivation for the algorithm and then summarizes some of the goals of this work. The algorithm itself is described in Section 5.

The major components and concepts of the algorithm are treated in separate sections of Chapter 3. Our algorithm combines the layers of image region matching (Section 1), flow field consistency (Section 2), and figure/ground separation (Section 4). Our algorithm can integrate information from other sources, and these higher-level sources are discussed in Section 5. The details of our particular implementation, including real-time considerations and time versus quality trade-offs are described in Section 6.

Experimental results presented in Chapter 4 demonstrate the performance of the algorithm on both synthetic and real data using some of the well-known data sets cited in the literature [5], and show that it it comparable to some of the best results obtained, with the added advantage of rapid computation.

This thesis concludes in Chapter 5 by bringing the claims and the experimental results together. The novelty of the present work and the future direction for this work are also outlined.

CHAPTER 2

Background

This thesis unites concepts from biological models of the primate vision system and computer vision techniques to yield a rapid and robust optical flow algorithm. However, we also offer an architecture for integrating higher-level information to steer the lower-level sensors in a principled way. For this reason, we will start this chapter with an overview of the motion problem (Section 1), followed by a discussion on computer vision techniques (Section 2) that have been applied and the relevant applications (Section 3) that motivated motion measurement in image sequences. Exploring a biological model hypothesized for the primate visual system (Section 4) has led to our proposal for a real-time optical flow architecture (Section 5). While measuring motion from image sequences will remain an open problem, our future work (Section 6) will make use of our open architecture to introduce operator and volumetric reconstruction feedback to project perceived 3D motion and objects back into the image plane to close an exploration loop. These exploration issues will not be addressed in this thesis.

1. General Motion

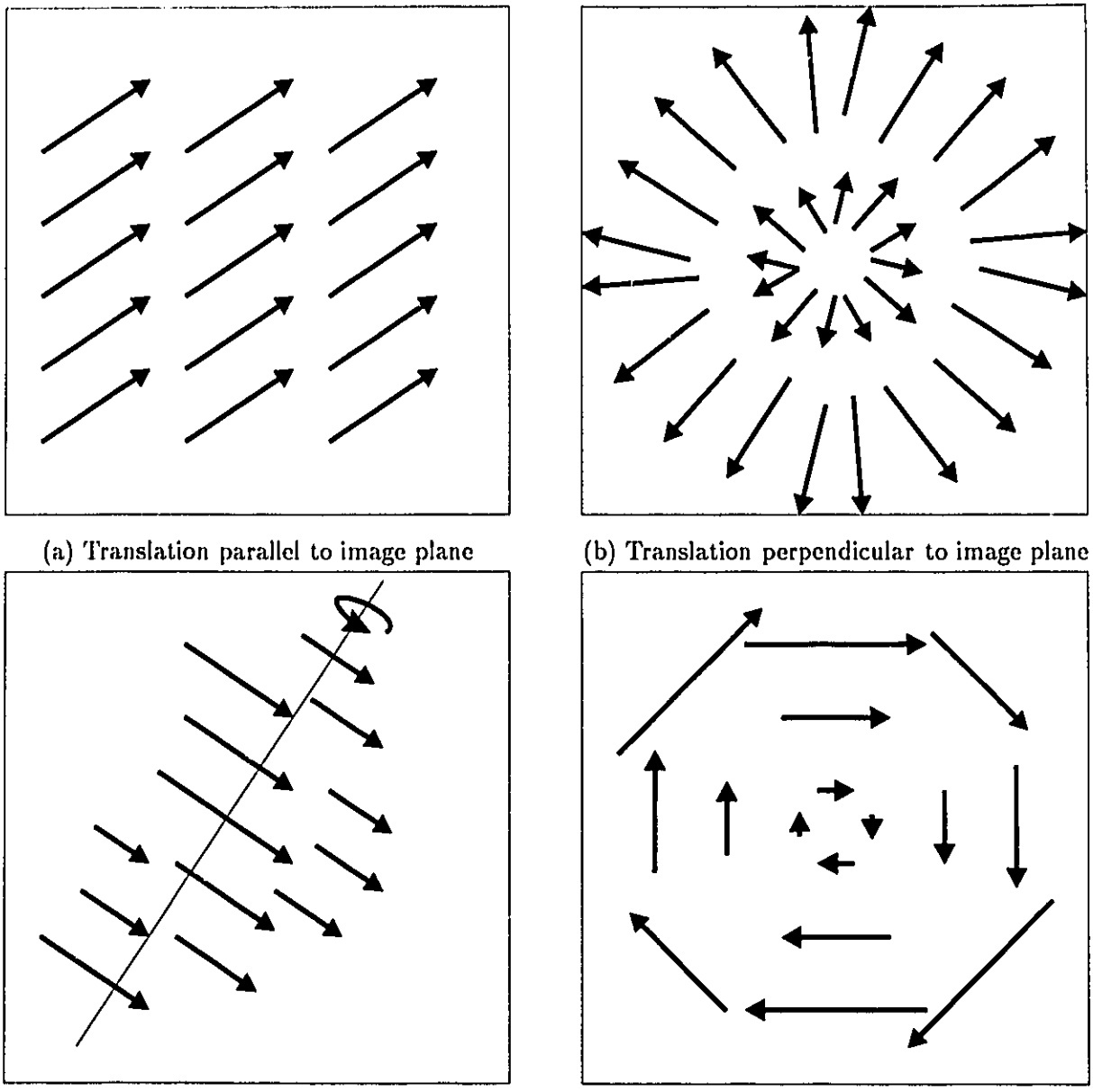
Humans perceive motion with so little effort, and with so much fluidity that they tend to take the mechanism for granted and focus on the inferences caused by the signals, rather than the signals themselves. A "trained observer" could be conditioned to locate important points in a scene and to describe the displacement of these points over time, but his or her visual processor integrates heuristics of the organization of the physical world to generate perception of physical motion more elaborate than the stimuli. The distinction between the stimuli and its perception is typically the focus of psychophysics. As far back as 1865, Mach had demonstrated and discussed the "existance of light and dark bands in the perceived pattern which had no analog in the actual input." [18, p. 219]. More recently, the studies

of illusory contours and other subjective interpretations of light patterns [9] have led to the elucidation of the mechanisms in the hidden layers between the retina and the cerebral cortex. In our case, we will be applying some of these implied mechanisms to a computer vision system, equipped with a gray-level sensing camera.

There has been debate on the choice of camera imaging model for motion analysis. Most camera optical paths are not perfectly modeled by a pinhole camera perspective warping. Projecting image plane data into 3D space using the constraints of strict perspective is a very daunting task. If, instead, one chooses a weak perspective model, such as scaled orthographic projection, the result is a toy model of perspective, much simpler both intuitively and mathematically. Translations in 3D parallel to the image plane are projected orthogonally onto the image plane. Translations perpendicular to the image plane yield a change in scale. This weak perspective approximation holds as long as the distance from the viewer to the object is significantly larger than the object. This toy imaging model might be implemented in human perception: it would explain why we get confused about an object's shape when perspective deformation dominates (as an object approaches a viewer).

Assuming the scaled-orthogonal imaging model, we can decompose 3D object motion into a set of representative motions, illustrated in Figure 2.1. We have already mentioned translation in the image plane and translation perpendicular to the image plane (scale change), but there is also rotation in the image plane and rotation about an axis in the image plane. The latter rotation decomposition with respect to the image plane is attributed to Kontsevich [17], and is a simpler form than the difficult to compute and decompose lumped 3D rotation and translation matrix of the camera (or object) motion. This thesis does not assume either strict or weak perspective imaging, and concentrates on the optical flow on the image plane itself. Nevertheless, it is important to note that general 3D motion projected onto the image plane, may have all of the motion components illustrated. This will be stressed later, when we show how some optical flow algorithms have difficulty computing some of these components.

Motion perception is a victim of noise. The image sensors (biological or synthetic) are prone to bursts of white noise and structured noise, aliasing and other imaging artifacts. The interpretation of static scenes is difficult enough, and measuring displacements of image components is also noisy in itself: small and large displacements alike suffer from aliasing. The older information technology techniques suggested a modular approach to computer vision in general, creating data structures to describe static images, and to have different modules infer knowledge from the image from different algorithms, e.g. edge-detection,



(a) Translation parallel to image plane

(b) Translation perpendicular to image plane

(c) Rotation about axis in image plane

(d) Rotation about axis perpendicular to image plane

FIGURE 2.1. Components of optical flow from weak perspective imaging.

shape-from-shading, and so on [20, 6]. But motion perception can not be confined to analysing two static images and then analysing the data structures of the two images: the tiniest noise signal introduced will make the problem intractable. Furthermore, natural scenes involve the motions of objects that are difficult to characterize as data structures. Given an image sequence, subjective issues such as scale and aperture allow many possible

interpretations and figure/ground separations, but the viewer usually selects one interpretation. To be of use computationally, this selection would need to be expressed in terms of a perceptual model or a model of biological sensor processing.

Regularization is introduced as a method of applying local constraints to an otherwise noisy system. Knowledge of the physical system under observation leads to approximate models of how neighboring elements should interact. In cases of noise, the system would integrate neighborhood activity into each measurement. Classic regularization techniques, such as those in Blake and Zisserman[7], deal with surface reconstruction from noisy measurements and use analogies of surface plates, weak springs and membranes to embody physics in a search for a best fit.

The distinction here is that a measurement process is guided by a regularization of the measurement results, and the process is iterated until a reasonable result is obtained. The unresolved problems here include: how many iterations are enough? How do we prevent degradation from over-regularizing or over-iterating? How do we prevent noise artifacts from under-regularizing or under-iterating? How do we balance the measurement signal with the regularized measurement signal? How important are neighborhood constraints with respect to individual measurements? For the most part, these questions are answered by tuning weighting parameters by experimental observations or noise measurements. There is clearly something important missing from the approach so far: a new regularizer is not the answer, but a better architecture to integrate measurements and physical or perceptual constraints.

We propose to go a large step further. Instead of embodying a model of the physics of surfaces into the solution, we model some of the *perception* of physics of surfaces and motion, which we will show is a more powerful constraint in motion perception, producing results consistent with human observers, including the cases of unstructured scenes.

1.1. Motivation With the advances in computer speed and the decrease in imaging hardware costs, the novel problem of extracting information from image sequences can be approached with more accessible tools. While the majority of optical flow and motion algorithms resolve the data into knowledge by off-line batch techniques, we have designed and constructed an optical flow algorithm sufficiently rapid to use temporal integration and user interaction as far more powerful constraints. We believe that this strategy and architecture will contribute to the more common computer vision tasks of surface reconstruction and tracking from image motion.

2. Computer Vision Motion Techniques

Optical flow can be considered as a dense set of point correspondences between successive images. How these correspondences are found varies according to key assumptions made about the imaged environment. The following sections describe the assumptions and characteristics of the dominant optical flow algorithm types. The comprehensive study by Barron et al. [4, 5] provides a level playing field to objectively test and compare optical flow algorithms against a set of standardized input image sequences.

2.1. Optical Flow by Differential Relations The key assumptions in differential relation optical flow such as Horn et al are that the objects in the scene have surfaces of near-uniform reflectivity (e.g. uniform color) and no specularity[15]. This allows the approximation of displacement proportional to intensity change. Image intensity leaks are considered to be negligible. For image intensity $E(x, y, t)$, this is expressed as

$$\frac{dE}{dt} = 0. \tag{2.1}$$

Using the chain rule for differentiation, Horn obtained

$$\frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t} = 0. \tag{2.2}$$

The unknowns are the velocities in the x and y direction, which are denoted as u and v such that

$$u = \frac{dx}{dt}, \qquad v = \frac{dy}{dt}. \tag{2.3}$$

The notion here is to approach optical flow as a large set of partial differential equations for a physical system, of the form

$$E_x u + E_y v + E_t = 0. \tag{2.4}$$

Still, this is under-determined, providing the component of the movement in the direction of the brightness gradient [15] and intensity leaks abound. Furthermore, textured surfaces fail under this assumption, and displacements are limited to a fixed scale; the same scale over which the intensity gradient change is computed. The assumptions made for this form of optical flow computation may seem almost contradictory, in that without texture, no motion is perceived, but the motion of boundaries is known in only one dimension, due

to the local nature of the measurement. Regularization improves the local measure by introducing neighbor interactions, but does not go all the way to propose a principled way to balance measurement with regularization. Local error minima are almost unescapeable, so initial estimates of displacement are essential for convergence.

2.2. Optical Flow by Feature Point Correspondence Feature point correspondence raises interesting issues, and literature exists to define what makes a good feature point. This decision is environment and task dependent, and is useful for very controlled or contrived situations, such as tracking target markers on a moving object. The reason for doing this is to bypass the optical flow problem and use precisely-measureable image features as a skeleton of displacement that can be filled in at a later stage. The advantage is that the problem becomes a data processing problem where features are matched in successive image frames, using a sparse data structure. The difficulty here is in determining how to fill in the motion for elements of the scene that do not contain feature points: we usually prefer seeing the motion of an entire car rather than the motion of some spots on the car: this gives more useful information for perceiving the bulk properties of the car, avoiding the car with one's bicycle, and so on. Collision avoidance may be desired, independent of the type of object in view.

The interdependance of what makes a good feature point, and how feature points interact as neighbors can not be respected in a feed-forward system. Most feature detectors use feed-forward image pre-processing, and not predictions of how feature points behave over time. This restriction prevents feature point correspondences from being used in more evolved vision architectures which include top-down feedback.

2.3. Optical Flow by Region Matching To borrow the terminology of Barron et al, a region-based matching algorithm defines the velocity $\vec{v}$ as the shift **d** that yields the best fit between image regions at different times [5]. The methods of Anandan [2, 3] or Singh [28, 29, 30] optimize a similarity measure, such as minimizing the sum-of-squared differences (SSD) between two images, I_1 and I_2,

(2.5)

$$SSD_{1,2}(x, y; d_x, d_y) = \sum_{j=-n}^{n} \sum_{i=-n}^{n} W(i,j)\left[I_1(x+i, y+j) - I_2(x+d_x+i, y+d_y+j)\right]^2,$$

where W denotes a 2-D window function, and (d_x, d_y) are usually restricted to a small integer number of pixels. Alternatively, a distance measure minimization is added to the optimization, in the expectation that small displacements make more sense than larger

displacements when the pixel patterns are matched equally well. Clearly, this is a bottom-up, or data-driven strategy that uses local information. The representative algorithms use different methods for reaching the optimal tile alignments, but neither use the rich information available at the flow geometry level, i.e. flow consistency constraints.

Anandan goes futher and measures confidence of each region-match, by measuring the curvature of the SSD error surface [3]. By doing this, Anandan encodes the directional certainty for each flow vector. At corners, this error surface is pointy, so the magnitude of the uncertainty is small. On edges, the error surface has a ridge, so the direction of uncertainty runs parallel to the ridge. Unfortunately, the directionality of this error measure is not used in regularization. This will be discussed in detail in Section 2.5.

One of the thorny issues is the size and shape of the W window function. The larger the sampling area, the more convex the error function, leading to fewer steps for convergence, at a higher cost of computation per step. On the down side, larger window sizes suffer at object boundaries, reducing the discriminability of finer details. In short, larger windows blur perceptually significant discontinuities. The linearity or non-linearity of the weighting function in W is also subjective: uniform weighting is computationally efficient, but perhaps less meaningful than a Gaussian distribution. We will propose that the size chosen for W is less sensitive when coupled with other constraints.

While optical flow techniques abound in the literature, all must deal with a fundamental limitation of the aperture problem: how can one share information between the local and global domains, even when they conflict? Furthermore, to what extent, and in what way can information from different scales be combined?

Anandan's algorithm traverses scale space by performing region matching at coarser scales and smoothing the resulting flow field, taking the directional uncertainty of each flow vector into account. The resulting flow field then seeds the search at the next-finer scale, and the hierarchical computation continues. This inheritance of smoothed results from coarser scales imposes a limitation on the amount of image-plane rotation that can be preserved. The practical upshot of this is that Anandan's algorithm is biased towards object translation and rotation about an axis in the image plane (which resembles translation over large patches), but has difficulty with rotation in the image plane.

In this thesis, we argue that the important information is not strictly individual error surface measurements that should be minimized, but rather, flow field consistency that should be enforced. By having the two layers of region matching and flow consistency

constraints interact, we can gain significant advantages in stability, robustness and overall accuracy. For example in the case of textured surfaces with a repeating pattern the algorithm would be less likely to become trapped in a local minimum.

2.4. Rigid Body Constraints Because optical flow fields are underdetermined by the sensory input, regularization is often introduced to force the resulting flow field to conform to certain material or perceptual properties. These constraints reduce the solution space while searching for a global optimum, and help produce results that satisfy the material physics or minimize noise over the entire data set. The way in which these constraints are applied, however, will significantly affect the overall outcome of the optimization. For example, interleaving measurement and regularization, versus weighting measurement with previous measurements and regularization, lead to different optimization paths.

Weber et al. show how a flow field can be segmented and predicted by using the Fundamental Matrix of each image region to determine local rigid body motion [36]. A region-growing method groups image regions of similar motion parameters, and associates a high cost for segmentations that select large numbers of groups. Using the Fundamental Matrix is a form of rigid body constraint (the matrix is formed under the assumption of a rigid body undergoing translation and rotation).

While the rigid body constraints embody physical models of the imaged environment, they are still difficult to determine, noisy to quantify into a useable form. The rigid body lives in 3D space, and the mapping from the image plane into 3D space is rarely adequate. For unstructured environments, a more appropriate level of regularization can be achieved with flow field constraints which are based on physical properties, but can be processed without resorting back to the 3D space.

2.5. Flow Field Constraints How does the flow algorithm know when it has reached a local, sub-optimal minima, and where should it go next to escape? One answer lies in the flow field consistency of its neighborhood: if the neighbors arrived at the same conclusion, there is no cause for alarm, but if one flow vector points in the wrong direction, it will ponder its neighbors' choice as an alternative. Algorithms that use flow fields as inputs generally expect that the flow fields behave in certain ways: the rigid body constraint discussed in Section 2.4 make strong assumptions about how objects behave in 3D, but projecting these constraints back into the image plane can be computationally challenging. An alternative approach is to deal with the flow vectors in the image plane, using simple rules of interaction that are suggested by perception experiments, and implied

by material physics. This becomes useful in the cases of sparse measurements, where only key features can be accurately determined, although a more dense measurement is desired.

One might hope that applying a Gaussian diffusion operator to a flow field would somehow spread the flow field information throughout the moving object. Giachetti et al. do this to spread a sparse optical flow field over the entire image plane, and perform some weighting based on measurement error [12]. This is acceptable when the desired effect is a linearized flow field for measuring a small number of motion parameters, such as focus of expansion, angular rotation and time to impact. In general, however, this only blurs the actual flow field and eliminates the finer details that allow figure/ground separation and tracking.

Anandan's algorithm [3] uses a modified Gaussian diffusion to spread flow vectors across each scale image before proceeding to the next-finer scale. Each vector becomes a weighted sum of its previous value and the average of its neighbors. The weighting is determined by the curvature of the error surface at that point. What is ignored is the error measure of the individual neighbors and their consistency with each other: one should not give the same weight to neighbors which are obviously wrong, and smear the error around. In order to be effective, flow field consistency must be enforced during the measurement stage.

Giachetti et al. also relax the rigid body constraints and permit shearing and other deformations to the image plane [12]. Prazdny's research into estimating egomotion [25] assumed that flow fields behaved in a linearly consistent manner, due to the imaging of planar obstacles: planar motion induces linearly smooth optical flow patterns.

Psychophysics insists that no matter how construed or noisy the input image sequence, the perception of fluid motion will dominate over local minima that lack local coherency or consistency. In fact, these uncertain areas are almost always influenced by neighboring, certain areas: when motion is perceived at boundaries of an object, the low-contrast interior region of the object is perceived to move with the same velocity, even when there is no apparent local motion. The hypothesized mechanism for this ability in the primate visual cortex is offered in Section 4.2.

Steering local detection of curvature or flow field by neighborhood flow consistency is covered by Parent and Zucker [23, 40]. Simple updating rules based on the substrate problem are used to attain solutions that are globally and locally consistent. The advantages of using flow field consistency include its ambiguity resolving, rapid convergence, and the abstraction from 3D physical models into the 2D flow field.

One of the problems associated with applying a Gaussian filter to diffuse the flow field is its linearity: discontinuities are smoothed over, regardless of their perceptual significance. Flow field consistency recognises that some combinations of neighboring flow vectors are more likely than others, and are given weights accordingly, preserving features based on measurement confidence and neighborhood compatibility, a decidedly non-linear strategy, rather than a blind feed-forward strategy.

Embedding flow field consistency into the measurement process allows confident neighbors to steer uncertain flow vectors for local consistency, but at the same time forces the propagation of local measures to achieve a more globally consistent set of flow vectors. Rapid convergence is possible due to the reduction in solution space, but also because flow field consistency does not lead the measurement stage to test hypotheses that violate material constraints.

3. Computer Vision Motion Applications

3.1. Surface Reconstruction from Points of Correspondence Two views of an object are not always sufficient to determine the structure of the object. We can always benefit from integrating new information into our perception of the world: new information reduces uncertainty, reduces noise, and solidifies our internalized representations of the world. More concretely, temporal integration of optical flow can certainly improve upcoming measurements of image motion, just as surface reconstruction can be improved by integrating multiple measurements, demonstrated by Heel [14].

Prazdny used synthetic flow fields to reconstruct egomotion parameters and for planar surface reconstruction [25]. The technique is reasonably rapid, performs coarse range imaging, but works only with low noise flow fields. The flow field generated from our algorithm will be shown to be of sufficiently high quality to do coarse range imaging of this type.

3.2. Feature Point Tracking Common applications for motion tracking in general include feature-point tracking [31, 26]. Both methods are presently limited to laboratory environments and controlled experiments, but could be improved by applying denser motion measures from optical flow, and might even be moved to less structured environments. Region matching coupled with flow field consistency can offer more than any single regularizer in terms of noise reduction. Feature point tracking for articulated rigid bodies would not benefit, however, if individual limb motion measurement needs to meet a precision requirement. However, rigid bodies with unknown shapes or more general, non-rigid motion would greatly benefit from the interpolation of perceived motion throughout the object.

3.3. Structure and 3D Motion The classic 8-point algorithm [19] offered a rapid technique of using point correspondences from two projections to produce the 3D positions of the points as well as the relative orientation of the two viewing planes. Unfortunately, this algorithm is wrought with noise sensitivity and singularities. Noticeable improvements are afforded by conditioning the inputs [13], but no amount of regularization will recover the surface in the presence of structured noise ever-present in optical flow techniques.

However, with rapid optical flow computation, where measurements and constraints produce acceptable flow output, the higher-level 3D constraints can be projected back down to the sensor level, closing the acquisition and analysis loop.

Uncalibrated, and even noisy, images can yield reasonable 3D structure and motion information [8, 27], but the noise in the image plane is amplified many-fold when projected back into 3D space through noisy imaging parameters [34, 39, 32, 37, 22, 35]. The limiting factors to perform the projection from the image plane to 3D space are noise in image coordinate measurements (due to aliasing or artifacts or local minima), and the ill-conditioned inversion of the perspective imaging parameters, such as the Fundamental Matrix. The inversion of the Fundamental Matrix can be regularized to a point, but at the expense of computational efficiency. On the other hand, the image coordinate measurements can be improved significantly when neighborhood flow consistency is applied: the motions of image elements are governed by constraints of fluid, or consistent flow elements. Clean patches of image coordinate measurement allow local calculations of the Fundamental Matrix, with the possibility of merging patches, until entire objects are percieved with a unique set of Fundamental Matrix motion parameters [36].

4. Combining Computer Vision with Biological Clues

In order to anchor this discussion in meaning and relevance, the reader is offered the following section as a roadmap to vision system architectures. After this, it should be more clear where our work is situated in the larger perspective, and why we consider it significant to the vision community.

4.1. Three Vision Paradigms: Summary of Early Vision When discussing (computer or biological) vision algorithms, it is convenient to classify them according to their architecture. We have adopted a classification and naming scheme that differentiates between early-generation approaches to computer vision and biologically-motivated systems, inspired by [41]. There is a steady evolution from one level to the next, and a complexity gap separates each paradigm from its predecessor. The reader should note that this section

is not meant to be an all-inclusive discussion of vision algorithm taxonomy, but rather, an illustration of how vision algorithm architecture tends toward solution methods espoused by biological vision systems.

This should not be surprising: after all, biological and computer-based vision systems suffer similar constraints in terms of size and complexity of available hardware and energy consumption. In the ideal case, both systems either need or could benefit from timely information. While some of nature's tricks for compactness and speed involve brute force, unorthodox shortcuts and massive parallelism, this rapid execution in turn simplifies the computation. Perception at the biological level, as we understand it today, succeeds by performing simple computations in specialized layers, enabling neighboring regions to influence each other. As the information passes through successive layers, higher levels have the opportunity to send signals to the lower-level layers, providing hints to steer the overall processing. This top-down feedback can be significantly more useful to the lower levels than neighborly interaction for convergence.

The **First paradigm** is typical of the first generation vision algorithms which attempted to associate features or objects with image intensities, such as histogram equalization and thresholding, or convolving an edge-enhancing kernel with an image. Points of interest are tagged, and because strong assumptions are made concerning the imaged environment, the interpretation is strongly context-dependant. The key ingredients to the first paradigm architecture include strictly feed-forward stages, very local processing that usually only works in artificial environments. Distinguishing characteristics of these algorithms are several tuneable parameters (thresholds and the like) that must be set by experimentation in order to function. The biggest advantage of this paradigm is the simplicity of computer implementation, and rapid execution.

The **Second paradigm** architecture attempts to rectify the drawbacks of the first paradigm by combining more local information, and performing more iterations before reporting results. Like the first paradigm, this usually involves separating the processing into feed-forward stages, but within each stage, information is diffused locally for each iteration. An example of this is combining information across different scales by performing first paradigm measurements at different scales and using temporal integration (such as Kalman filtering) to determine relevant scales at different image locations. Diffusion parameters and discontinuity penalties enforce properties perceived in physical objects. Surfaces are seen as locally smooth patches with sharp edges, moving objects tend to stay in motion. These material observations are translated into constraints imposed during the iterations: at each

point, neighborhood interaction causes adhesion of clumps of data. This leads to a definite improvement in results, but it is costly and leaves open the issues of how many iterations can be performed at each stage before eroding the information one hoped to recover. The computation, therefore, involves many iterations of relatively simple rules. Unfortunately, the results become eroded beyond a certain number of iterations, and execution is usually slow.

The **Third paradigm** takes its inspiration from approximate models of the massively parallel and integrated organization of biological vision systems. The information flow here is not strictly feed-forward: each stage performs a large amount of neighborhood interaction that includes non-linear diffusion. Later stages are biased by their previous state, and neighbors can influence each other in cases of sensor or signal noise. Global constraints govern behaviour in cases of local ambiguity, providing perceptual continuity. Furthermore, these constraints are less restrictive than the constraints of the second paradigm; surface curvature consistency replaces the plate and rod models, relaxation labelling replaces the maximum-likelihood strategies. One key advantage to this approach is rapid convergence (in terms of number of iterations; each iteration can be very costly). The minimization of errors used in the second paradigm is most often blind to neighborhood interaction. In the second paradigm, blind minimizations lead to testing interpretations of the scene that are unstable or violate material property constraints. In the third paradigm, impossible states are discarded and not even tested: at each stage, the algorithm will only reach a state allowed by the previously attained state, and the consistency constraints are inviolable. The computation usually involves a large set of simple rules that would be executed in a parallel fashion. Typically the complexity of implementation prevents these algorithms from being real-time vision systems. This thesis presents an algorithm that runs contrary to this widely-held view, providing the robustness and high-quality results of the third-paradigm class algorithms but within a reasonable amount of execution time.

4.2. Biological Clues The primate visual system is arguably the most developed vision system that can be studied. Computer vision research has much to gain from studying the structure and organization of the only fully-functional vision system architectures, developed not by engineers, but by nature.

Some approximate biological vision models yield efficient algorithms by decoupling a complex minimization expression into individual components. Marr and Poggio [21] used a cooperative algorithm for computing stereo disparity, suggesting a rough model of early vision and cooperative layers that could be explained by the biological hardware available.

The cooperative process decouples a pattern-match minimization and surface continuity constraints into two layers that interact (feedback). The two layers are much simpler than a single layer that performed both actions. Since the simpler or more complex minimization attain the same goals, it seems reasonable that the simpler architecture is a more practical choice for both biological systems and synthetic algorithms. Hardware that consists of specialized layers, where each element in a layer resembles its neighbor, is generally faster and less expensive than single-layer monolithic implementations. More iterations can be performed in a given amount of time. For the relatively slow neural pathway signals (10 cm in 10 ms), this decoupled architecture is crucial.

Ullman suggested mapping biological computational elements to the algorithmic counterparts in [33]. Examining the receptive fields in the retina and the contrast-enhancing property of the Difference-of-Gaussian operators leads us to examine the signal path further, into the visual cortex. We can model the simple, complex, and hyper-complex cells in the visual cortex as combinations of these receptive fields. But there is yet another step needed to explain the neighborly interaction of perceived motion.

Staining neurons of the visual cortex for cytochrome oxidase identifies clusters of neurons that are highly active, contrasted with clusters of lower activity [1]. Typically, the stained cross sections yield clumps, or *blobs* of apparently higher neural activity, separated by regions of lower neural activity, sometimes referred to as *interblobs*. The regions of higher activity (blobs) are found to be scalar representatives of the visual fields, while the lower activity regions (interblobs) are orientation detectors [1]. The scalar zones are more accustomed to continuous activity and react abruptly to changes in input, while the oriented zones fatigue rapidly unless the images shift, allowing the neurons to rest until activated again, responding smoothly to changes in input. This suggests that shifting the input images by eye movement becomes necessary not only to refresh the retinal cells, but also to refresh the oriented cells.

Some of the clues for region-based motion (and stereo vision) correspondence that are already widely-known include the alternating left- and right-occular dominance hypercolumns in the visual cortex. In each hypercolumn of the interblob (oriented) regions, different orientations and scales are represented, higher neural firing rates indicating a stronger presence of each element. In each hypercolumn of the blob (scalar, non-oriented) regions, different spacial scales of light intensity are represented.

The parvocellular pathway, dominated by orientation detectors, has been suspected of being largely responsible for matching edge information from left and right visual fields, and

the medial temporal region for tracking this positional information over time. This would be a strong indication for the importance of edge information in sequential image analysis.

But what about the non-oriented, smoothly-varying or textured image fields that do not trigger the oriented-edge cells? Surely, scalar information would be more useful in these zones. The correspondence problem would be given another constraint to help the solution converge more rapidly, and reduce the solution space. The blob and interblob regions are interspersed, and can share information with each other. The blobs might perform image intensity region (or template) matching in the hypercolumns, and similarly, the interblobs might perform a comparison of flow orientations across several scales.

Because the blob regions respond abruptly to changes in input, neighboring neurons within the blobs may represent very different light intensities, as opposed to the interblob regions, where edge and flow orientations are strongly influenced by neighboring orientations. One would expect that within the blob regions, neighboring neurons do not diffuse their estimates as much as neurons within interblob regions. This characteristic could allow a form of window or region matching of light intensities or textures that takes place between corresponding receptive fields in the blob regions over time.

A key issue underlined by Yeshurun [**38**] in stereo and motion perception is the difference in size between a receptive field and the region represented by a hypercolumn. This observation implies that a motion patch in a hypercolumn represents changes from a cluster of neighboring receptive fields, and that the motion field will be less dense than the input image field. Traditionally, however, optical flow algorithms [**3, 28, 15**] expect motion field density about the same as the imput image density.

The combination of these relevant facts suggest that the scalar and oriented hypercolumns perform different tasks to produce optical flow motion representation, and the results from both are combined to produce a coarse flow field. The blob-interblob interaction may be modelled as intensity-based region matching diffusing its estimates, and the orientation of the flow elements as a constraint or boundary for the diffusion.

These cooperative processes play quite well into the decoupled matching and consistency minimization model suggested above. Each "layer" performs its task on its inputs, and sends a compressed summary to the next (or previous) layer.

5. Proposal for Real-Time Optical Flow

As suggested in Sections 4.2, 2.3 and 2.5, our algorithm performs region-based matching between successive image frames, at once minimizing a pixel pattern matching error and

imposing flow field constraints within a neighborhood. For the purposes of this thesis, we will consider the optical flow field to be a coarse field of image point correspondences between the two sequential images.

5.1. Organization of Algorithm An overview of our algorithm is presented in the block diagram of Figure 2.2. Each stage is represented as one block, with the execution of stages proceeding from left to right. Each stage performs iterations internally, as indicated by the dark, curved arrows. Furthermore, there is a backward and forward exchange of flow data between the tile-matching and the flow consistency stages. Note that flow information from a Kalman prediction loop is used to predict the next set of region matching. This top-down feedback is used in our laboratory environment, but for this thesis, we will be reporting results without using this high-level tracking and prediction, in order to compare our optical flow algorithm with other relevant algorithms. Each stage of this block diagram is described in more detail in Chapter 3.

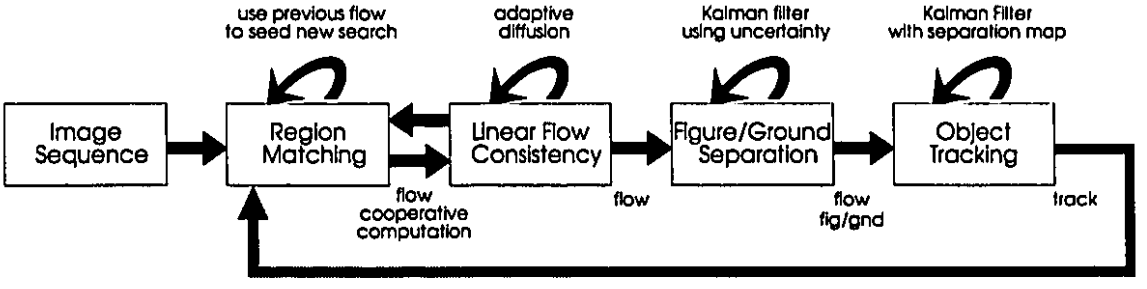


FIGURE 2.2. Block diagram of full optical flow process, including tracking feedback.

For the most part, however, we will demonstrate the core algorithm modelled by the block diagram in Figure 2.3. The higher-level information is ignored here, and the optical flow is examined and processed without considering the *interpretation* of the optical flow. For this simplified model, prediction of upcoming image motion is determined by the previous measured motion at each point.

5.2. Strengths and Shortcomings of Algorithm Our algorithm is firmly rooted in the third paradigm of vision architectures. Impossible (perceptually unlikely) interpretations are not allowed into the optimization process. This intelligent error minimization embodies the flow field consistency constraints described by material properties and psychophysics by restricting the correspondence search regions and combining the geometrical

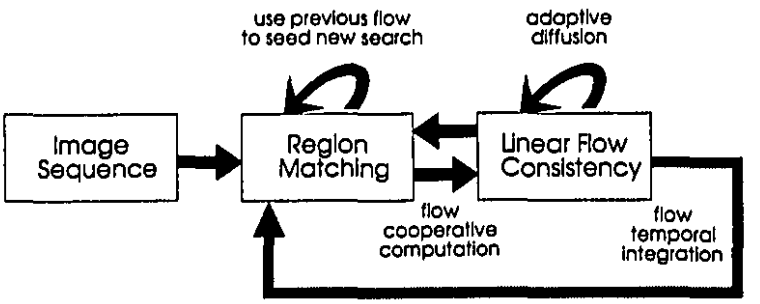


FIGURE 2.3. Block diagram of cooperative optical flow process, without higher-level information. This model will be used for most of this thesis.

information from these correspondences with non-linear diffusion to seed searches for further iterations. The stages of region-matching and flow consistency are separate stages, but they share information and steer each other. The principled method of integrating the flow field geometry with correspondences enables an open architecture where information can be introduced from a higher level of processing, leading to top-down feedback. The promise of this approach is rapid convergence for flow field measurement, better quality flow fields, and easy integration with other processing stages.

For the purpose of this thesis, we apply no underlying surface transport model. An example of such a surface transport model would be parametric solids, such as superellipsoids, undergoing general motion. The optical flow algorithm presented here does not use this constraint, neither in the measurement stage, nor in the flow field consistency stage. Such a powerful constraint would improve the quality and convergence of the flow field. However, allowances have been made to include this type of higher level information at a later stage, in order to predict upcoming flow fields.

At this point, it may seem that some issues have been simplified to fit our architecture, namely ignoring directional confidence that Anandan's algorithm could produce. We will show that these issues have not been neglected, but instead are embodied by the form of flow consistency constraints chosen (see Section 2.3). The savings in computation and sampling, however, are enormous, allowing either more iterations per sample set, or more sample sets per unit time.

5.3. Performance Expected Our proposal can meet the goals presented in Section 5.2 by demonstrating that convergence occurs within very few iterations, and that the quality of the resultant flow field is competitive with slower algorithms. These points will be dealt with in the Experiments section, Chapter 4.

This proposal would demonstrate the almost real-time capabilities of a good quality third-paradigm algorithm when applied to optical flow. Our premise is that the approximate biological model of visual cortex hypercolumns provides powerful constraints to motion processing that can be performed with simple updating rules and implemented in near real-time on conventional workstations. Our claim is that our integrated third-paradigm architecture with simple rules out-performs second-paradigm algorithms that embody more complex constraints in the error minimization in every measureable sense. This claim will be justified by comparing results with competing algorithms in Chapter 4 (Experiments).

5.3.1. *Comparing Computational Cost* Region matching, the fundamental measure used in this thesis and the algorithms of Anandan and Singh, will have the same computational cost for all three algorithms: it is dependant on the area of the window used in region matching. Also, all three algorithms have a form of neighborhood diffusion operation that regularizes the flow fields. In this case as well, the computational cost of the diffusion is dependant on the number of neighbors that are affected by any single element, analogous to an area encompassing the neighbors. What distinguishes the three algorithms, therefore, is how often pixel regions must be compared, and how often flow measures must be diffused between two image frames.

Chosing r to represent the cost of performing a region comparison, d to represent the cost of updating an estimate by examining all its neighbors, and N to represent the number of resulting flow vectors, we shall examine the cost entailed by our algorithm and those of Anandan and Singh. This cost analysis is not absolutely rigourous, and some allowances must be made for end-user adjustments, such as diffusion factors and the number of iterations applied. This section is offered as a sketch for comparison.

For our algorithm, we perform an arbitrary k iterations between image frames. For each iteration, there is one step of $18 \times N$ region matches of cost r and one step of N diffusion updates of cost d. As a cost expression, then,

$$C_{ours} = kN\,(18r + d)\,. \tag{2.6}$$

This leads us to the claim that our algorithm's cost is of order $O(kN)$.

For Anandan, there is an added cost of image pre-processing to construct the hierarchical image pyramid, but this is a fixed cost and will be put aside for this discussion. There are n levels in the image hierarchy, where n is typically proportional to $log(N)$. At each level i there are $2^i \times R \times N$ region matches of cost r, where R is the number of tests that

are applied for each measurement, typically 36. At the same level i, there are $D \times 2^i \times N$ diffusion updates of cost d, where D is the number of diffusion iterations (typically 10). At each level i, then, the cost becomes

$$C_{Anandan}(i) = 2^i N\,(Rr + Dd)\,. \tag{2.7}$$

Summing up these costs over all n levels of the image hierarchy, we obtain the total cost of

$$\begin{aligned} C_{Anandan} &= \sum_{i=1}^{log(N)} 2^i N\,(Rr + Dd) \\ &= \left(2^{log(N)+1} - 2^1\right) N\,(Rr + Dd) \\ &= 2\,(N-1)\,N\,(Rr + Dd) \\ &\approx 2N^2\,(Rr + Dd) \end{aligned} \tag{2.8, 2.9}$$

This approximate cost analysis suggests that Anandan's algorithm has a cost of order $O(N^2)$. Note also that D and R are usually constants of the order of 10, adding more cost, and that the image pre-processing is also costly.

Singh's algorithm is of similar hierarchical structure, and can be paritioned in a similar fashion, leading to a cost of order $O(N^2)$ as well.

How can we ensure that our algorithm will be faster (less costly) than those of Anandan or Singh? First, we can ensure that the number of iterations k is much less than the number of flow vectors N. Second, we perform only one diffusion operation per iteration, instead of D. The three algorithms offer comparable quality results, as will be shown in Chapter 4, but ours is of cost $O(kN)$ instead of $O(N^2)$.

6. Context and Future Work

A 3D volumetric reconstruction architecture is under study that would incorporate flow measurements and estimated range data, using simplifications inherent in the technique suggested by Kontsevich in [17]. A simple scaled-orthographic (weak perspective) camera model can be used to extract approximate 3D object descriptions, which can evolve in time to assist the optical flow algorithm. By rapidly estimating a volumetric model of the scene under study, the 3D motion of the object can be tracked well enough to predict upcoming flow fields, projecting 3D motion into the image plane. The flow field consistency constrains

the solution enough to reduce the workload of the volumetric fitting process. The 3D model of the object will constrain upcoming flow fields significantly more, leading to rapid temporal convergence of the whole system.

CHAPTER 3

Theory

1. Region Matching

1.1. How it's Usually Done The comprehensive study by Barron et al. [4, 5] on the performance of optical flow techniques describes, classifies and compares representative algorithms using similar conditions. To borrow from their terminology, a region-based matching algorithm defines the velocity $\vec{v}$ as the shift $\mathbf{d}$ that yields the best fit between image regions at different times [5]. The methods of Anandan [2, 3] or Singh [28, 29, 30] maximize a similarity measure, such as minimizing the sum-of-squared differences (SSD) between two images, I_1 and I_2,

(3.1)

$$SSD_{1,2}(x, y; d_x, d_y) = \sum_{j=-n}^{n} \sum_{i=-n}^{n} W(i,j)\left[I_1(x+i, y+j) - I_2(x+d_x+i, y+d_y+j)\right]^2,$$

where W denotes a 2-D window function, and (d_x, d_y) are usually restricted to a small integer number of pixels.

SSD as a measure of matching error is useful and convenient for computation, but has some drawbacks that affect its ease of use and appropriateness to the task. More precisely, SSD returns a number that is unnormalized and varies according to overall intensity, and camera noise can be amplified. The SSD returns a positive number for any measurement, it does not indicate the overall goodness of a match. The SSD error surface will return a single minima even when many answers are possible: the decision is then controlled by camera noise. The same SSD number from two locations says nothing about the similarity of goodness of fit at the two locations.

Anandan's algorithm performs this region matching at coarse scales and diffuses the resulting flow field to seed region matching at finer scales. The diffusion is controlled by the

curvature of the SSD error surface. As will be shown, the SSD error surface can have a very different shape depending on overall light intensity changes. The same correspondences under different lighting conditions will have different SSD error surfaces, meaning different SSD error curvatures, leading to different diffusion characteristics. The practical upshot is that the same flow pattern will be spread differently in the case of an overall lighting change, an undesirable property. The diffusion will be discussed further in Section 2.2. Because the coarse scales are represented with fewer pixels than the finer scales, Anandan is able to apply the same $W(i,j)$ at each scale, typically a 5×5 window.

1.2. How We Do It To find correspondences of clusters of pixels between the first and second images, our algorithm divides the first image into a grid of tiles, and proceeds to search for each tile's corresponding position in the second image, minimizing a pixel pattern-matching error metric between corresponding tiles.

The search for corresponding tile positions is assisted by providing an initial estimate of where each tile was predicted to move. This can be provided by a higher-level process in a larger vision system, and effectively tunes the measurement system to the expected motion events. For this thesis, the predicted flow field is the flow field calculated from the preceding image pair.

For each tile p, there is a pixel pattern $P_{1p}(i,j)$ in frame 1 at position T_{1p} and a corresponding pattern $P_{2p}(i,j)$ in frame 2 at position T_{2p}. We define a difference and summation operation between the two corresponding tiles as

$$D_p(i,j) \triangleq \|P_{2p}(i,j) - P_{1p}(i,j)\| \tag{3.2}$$

$$S_p(i,j) \triangleq \|P_{2p}(i,j) + P_{1p}(i,j)\| \tag{3.3}$$

$$err_p(i,j) \triangleq \frac{D_p(i,j)}{S_p(i,j)} \tag{3.4}$$

$$err_p = \sum_{i,j} err_p(i,j) \tag{3.5}$$

This difference and summation are performed between corresponding pixels, and the resulting error term err_p for the tile summarizes the average pixel-matching error. The error function expresses a difference of intensities, normalized by their mean. To combat sensor noise, thresholds are applied to D and S, to clip unwanted behavior at sensor input extremes. This applies mainly to when the input intensities are very low, and governed by noise. When the summation of the pixel intensities is too small, the data are essentially unusable. Also, when the differences between successive inputs are very low, the intensities

should be considered essentially the same.

(3.6) $$\text{for } S_p(i,j) < S_0, \text{ set } err_p(i,j) = err_{max}$$

(3.7) $$\text{for } D_p(i,j) < D_0, \text{ set } err_p(i,j) = err_{min}$$

The advantage of using these two constraints becomes clear when using natural scenes encoded by video cameras: even in a static scene, digitization or other noise can introduce speckles or streaks in an image sequence that most algorithms would prefer chasing. These parameters were chosen to be $S_0 = 16$, $D_0 = 8$, for the intensity thresholds of a 256-level digitized image. The error levels were never allowed to be exactly 1.0 or 0.0. Instead, we preferred the more numerically stable choices of $err_{max} = 0.99$ and $err_{min} = 0.01$.

1.3. Why It Should Work (Better) Our normalized error measure has several desirable features. It returns an absolute measure of goodness of match, from 0 to 1. Camera noise is clipped, or taken into consideration during individual pixel comparisons. Equally plausible candidates for region matching are not just local minima in an error surface, but have about the same error height. In SSD, local minima can correspond to equally plausible matches, but will have widely varying error heights, and the *numerically* lowest of these minima will influence the outcome of a search. For our normalized measure, the same number for different pixels imply the same quality of match. The same number for different regions implies the same overall quality of match for the regions.

To illustrate the difference between the SSD region matching metric and our own error metric, representative regions have been chosen from a natural scene, and the SSD error surfaces and our error surfaces are compared. The images chosen are from a hand-held moving cube sequence, shown in Figure 3.1.

There are some noticeable similarities in the shape of the competing error surfaces: they are both concave around the minima for corner points, and have troughs at edge regions. But a serious drawback to SSD is illustrated in Figure 3.2, where the SSD error surface has a gentle slope near the minima, and the minima itself is hard to detect as compared with the normalized error surface. Note also the difference in scale between the two measures. The normalized error measure is designed to *locally* vary between 0 and 1 at each pixel-to-pixel comparison, but a comparison using squared differences will vary the scale widely between any two pixel locations. Neighboring individual pixel error measures for squared differences will therefore produce numbers that are not necessarily proportional to any perceivable similarity between the two pixel locations. An SSD error is the summation of these contributions, and this *combined* error scale varies from neighboring region to region.

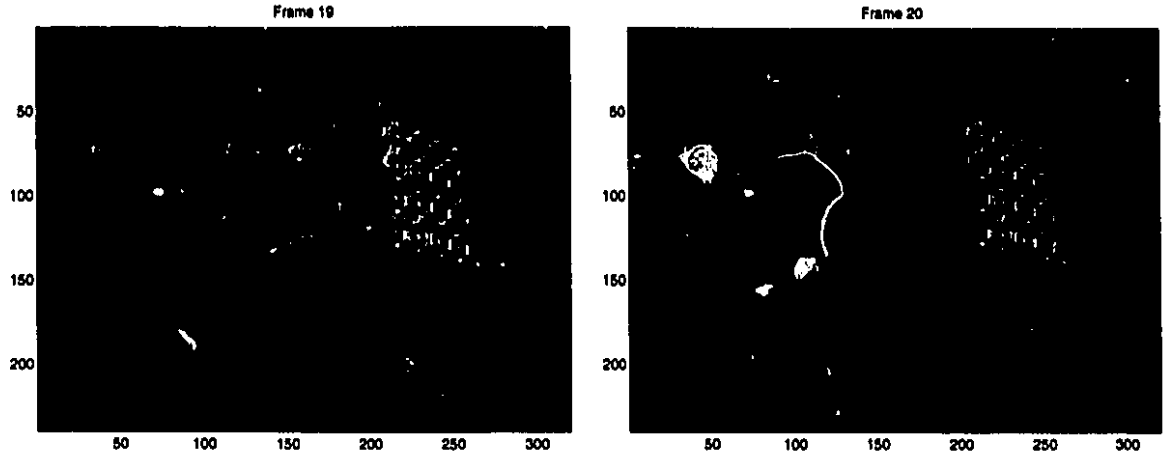


FIGURE 3.1. **Region Matching image frames.** Frame 19 of the hand-held moving cube (a), and Frame 20 (b). The numbered squares are the initial tile positions for region matching between the two images. The number corresponds to the region matching experiments, shown later.

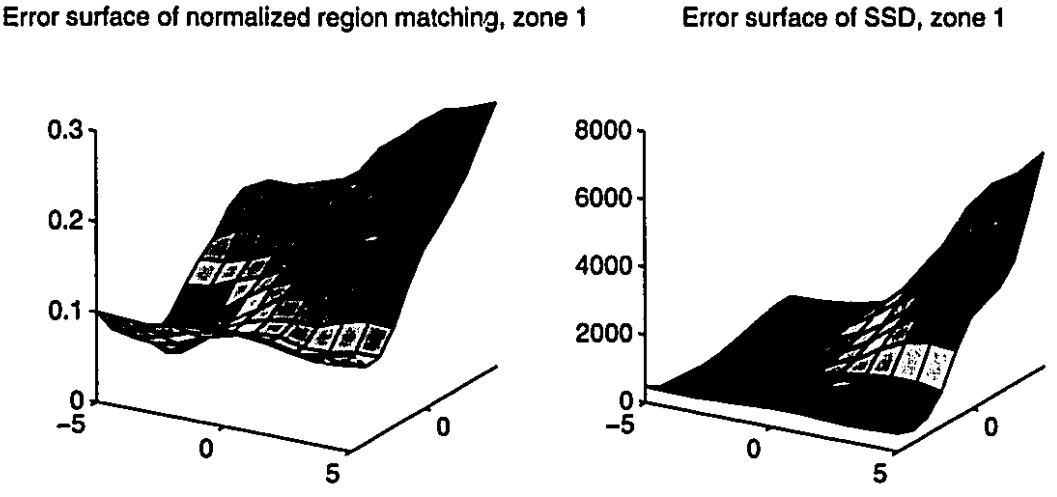


FIGURE 3.2. **Region Matching Zone 1.** The error surfaces generated using the normalized error measure versus the SSD error measure. The vertical axis is the error, the x and y axes are the pixel region shifting between the two frames to obtain the region match. This corresponds to a corner of the cube in the image sequence. Note the minima in the lower left of the two surfaces, where the true correspondence lies.

Of course, when overall light intensity does not vary much, such as the smooth grey-levels in the hand area, SSD and our normalized error metric perform very similarly, as

shown in Figure 3.3. Noisy, low-intensity image patch error surfaces also look similar, as shown in Figure 3.4. Note, however, the error axis of the SSD surface tells us nothing about how ambiguous or noisy the imaging conditions are. Our normalized error measure tells us that a large variation in position causes a small change in error. The image patch in question is a poorly-lit, out-of-focus whiteboard with writing on it: this should not be considered as reliable as a more textured image patch. The curvature of the SSD error surface cannot tell us this.

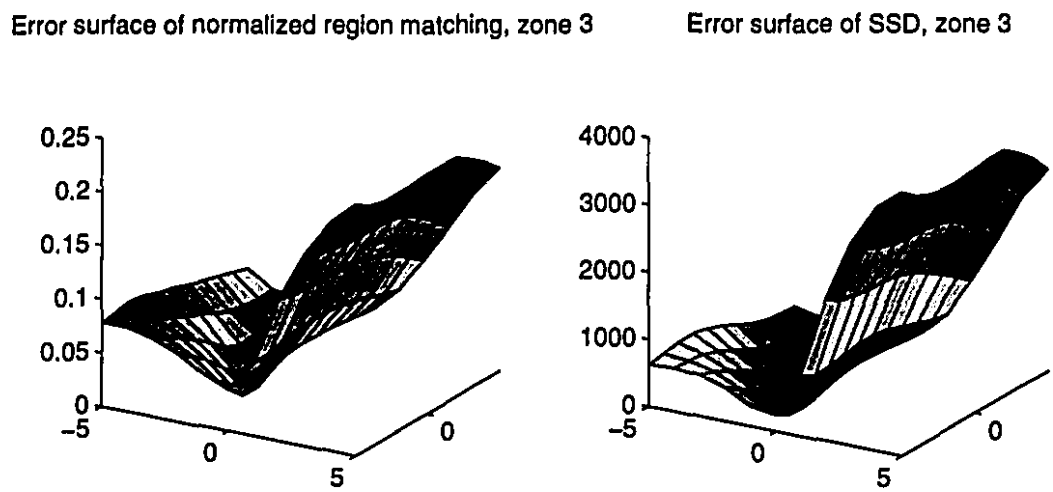


FIGURE 3.3. **Region Matching Zone 3.** The error surfaces generated using the normalized error measure versus the SSD error measure. The vertical axis is the error, the x and y axes are the pixel region shifting between the two frames to obtain the region match. This corresponds to the crease near the operators hand in the image sequence. Note the trough indicating the edge-like nature of the matching.

One other distinction between the normalized error surface and the SSD error surface can be demonstrated near an edge of high contrast. In this case, the operator's dark hair occludes the whiteboard behind him, shown as region 6 in Figure 3.1. The competing error surfaces are shown in Figure 3.5. Note the sharpness of the corner for the normalized error, and the smoother cusp for the SSD error. For the same window size, better positional accuracy can be achieved using the normalized error.

A neighborhood similarity error measure is provided as a function of local, individual pixel similarity errors. No assumptions are made about neighborhood intensity leakage that could bias derivative-based algorithms, like that of Horn and Shunck. This simple measure makes a strong statement about accomplishing region matching using inexpensive computational mechanisms that could be found in a biological vision system: the comparison of differences is easier to perform than the comparison of absolute values. By normalizing

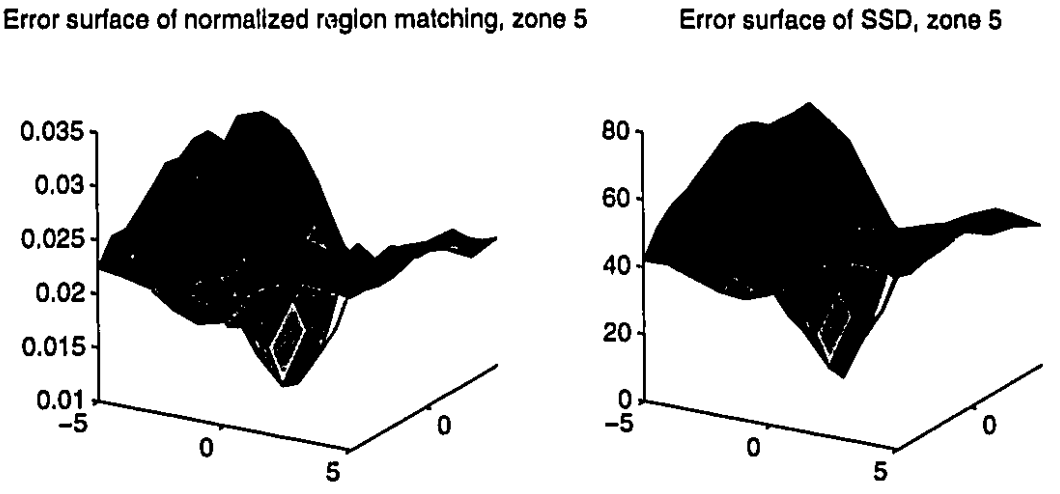


FIGURE 3.4. **Region Matching Zone 5.** The error surfaces generated using the normalized error measure versus the SSD error measure. The vertical axis is the error, the x and y axes are the pixel region shifting between the two frames to obtain the region match. This corresponds to the poorly-lit, our-of-focus whiteboard in the background in the image sequence. Note the SSD error surface does not tell us how ambiguous the overall matching is.

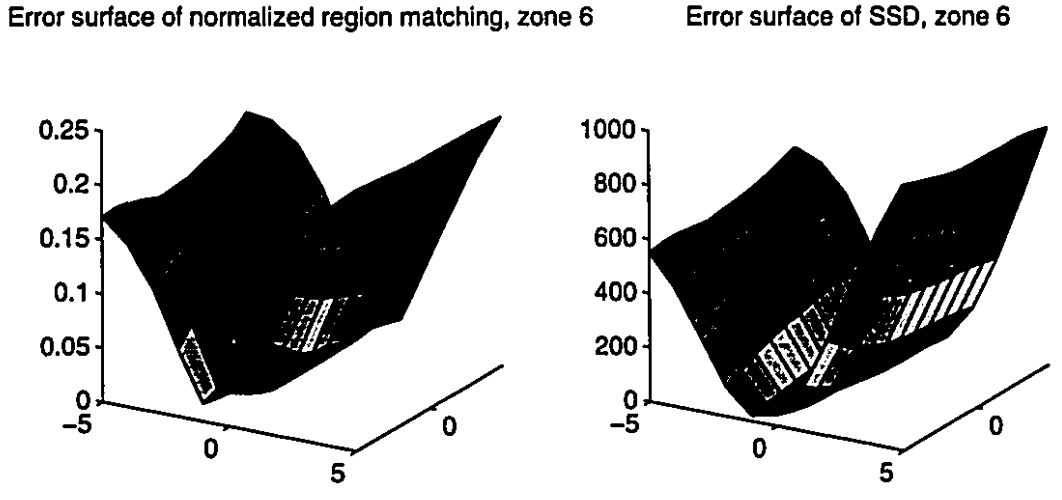


FIGURE 3.5. **Region Matching Zone 6.** The error surfaces generated using the normalized error measure versus the SSD error measure. The vertical axis is the error, the x and y axes are the pixel region shifting between the two frames to obtain the region match. This corresponds to the dark hair of the operator meeting the whiteboard on the left of the image sequence. Note the SSD error surface is not as sharp near the edge.

differences with the overall intensities, local illumination effects are eliminated: the emphasis is on local texture instead of local illumination.

By clipping the noisy extremities of intensity sums and differences, we make another strong statement by stating that no useful information can be extracted from indiscriminable intensity levels. When noise dominates, the algorithm tags the results as such.

Psyochophysical experiments can show that a human observer will perceive apparent motion of random-dot patches. Intensity-derivative methods cannot function in this environment. An algorithm using texture measurement, such as edge matching would fail as well. Only an algorithm performing region-matching could successfully track this type of motion.

No global-local reduction operations are performed. No scale-space assumptions are made, all measurements refer to the original set of images, not pre-processed, band-limited data.

1.4. Importance This error metric is rapidly executed, and measures the image samples directly, instead of a pre-processed smoothed or band-passed image: all the original data is available for measurement in an undistorted, unbiased, unfiltered form. Sensor noise is combatted at the lowest measurement stage, where noise is most expected to arise. The error measure minimization is very convex and stable.

Note that this error metric minimization has a weakness in the case of repeated textures over a large area. Local minima may be good matches of textures, but they do not describe the overall surface motion. This is a problem inherent in all forms of region-matching and correlational techniques, where geometric information from a higher level can help. A principled technique to correct these misinterpretations is discussed in Section 2. There, we will show how flow consistency steers region matching away from these local minima and toward a solution that is consistent with its neighbors.

2. Flow Field Consistency

2.1. What it is Flow field consistency is the behaviour of a flow field that obeys the constraints suggested by psychophysical experiments in motion perception. For example, texture flow fields are improved when the measurement process includes a texture flow curvature consistency constraint [23]. The issue of how to measure or enforce optical flow field consistency now deserves attention. Applying curvature consistency would probably improve the optical flow field, but due to the coarse sampling of tile alignments and equally coarse directional encoding, a linear flow consistency is more appropriate.

2.2. How it's Usually Done A form of flow field consistency is typically integrated into the measurement process by penalizing large displacements from the predicted search area, without considering neighborhood displacements or expectations. Alternatively, after a measurement stage is performed, the resulting flow field is diffused uniformly to neighboring tiles. Modest improvements are obtained when the diffusion is weighted by measurement error: this is the first step toward using confident measures to influence unconfident measures. But for the most part, this information is not used to re-seed the measurement stage, and the flow field diffusion is used unaltered by a higher-level process.

There are two distinctions to be made in Anandan's algorithm for flow consistency. First is the search for correspondences, which is goverened by the SSD surface curvature. Second is the diffusing of information between neighboring flow vectors.

Examining the first case for steering individual correspondence matches, Anandan's Hierarchcal algorithm uses a form of the Gauss-Seidel relaxation algorithm. The shape and orientation of the Gaussian filter is altered according to the direction of the flow uncertainty at each point, effectively tuning the search for correspondences along the troughs of the error surface during diffusion. This is intuitively correct for cases of edge-like structures in images, however does not tell us how to interpolate in the uncertain areas between edge-like structures.

Considering the second case for flow field consistency, Anandan's algorithm applies a linear Gaussian diffusion at each scale after region matching. The effective radius of the Gaussian smoothing function is determined by the curvature of the SSD error surface. Although the shape and orientation of the Gaussian filter could have been altered according to the direction of the flow uncertainty at each point, this information is ignored during diffusion. This feed-forward smoothing leads to difficulties at discontinuities and image plane rotations, but also allows poor measurements to bias otherwise good measurements.

One of the disadvantages of the Gaussian diffusion is its linearity. The smoothing was intended to enforce neighboring flow vectors to have equal directions and magnitudes, but no amount of linear diffusion will bring this about (except in the limit, where the entire flow field is blurred away to converge at uniformity). Not only are all neighbors considered equally valid (a rare event), but also, the flow field model (neighbors have similar magnitudes and directions) and the updating function (Gaussian diffusion) are incompatible.

2.3. How We Do It We acknowledge that neighboring elements in an image sequence can legitimately undergo very different motions. Discontinuities in flow are perceptually significant, and preserving these discontinuities during the diffusion stage is important. A form of relaxation is applied to our flow fields to update flow vectors in such a way that similar flow vectors reinforce each other, and different flow vectors will selectively ignore each other when neighbors conflict.

Relaxation labelling is an iterative procedure that models the parallelism of feature-preserving data diffusion and noise reduction in the brain. Each data element is assigned a parameter or label, with an associated probability or confidence measure. Compatibility functions are chosen to reflect desirable perceptual responses that describe the relationship between neighboring data. These compatibility, or *support* functions are used to refine the initial labelling $\{p_i^0\}$ at each iteration k [24]. Hummel and Zucker devlop a general scheme for the iteration [16]

$$p_i^{k+1}(\lambda) = f(p_i^k(\lambda); s_i^k(\lambda)) \tag{3.8}$$

where s_i^k is a measure of support for element i at iteration k.

For our purposes, the label λ is the parameter, the flow vector with a magnitude and a direction, and the *belief* in paramter λ at position i is $p_i(\lambda)$, obtained from an error measure (from region-matching). Our *prior*, or constraint for convergence must be expressed as a support function s.

Linear flow consistency implies that patches of an image should be moving in roughly the same direction as their neighbors. In cases of uncertainty, when a patch is moving in a direction contrary to all its neighbors, the neighbors will influence the outlier more than vice versa. An easily-implemented updating rule performs a weighted averaging of neighbor's displacements, each contribution weighted by a similarity measure. This similarity measure encodes a similarity of direction and magnitude between two given vectors.

Adaptive diffusion allows confident neighbors to influence uncertain tiles without affecting already confident tiles. To apply linear velocity consistency between all adjacent tiles, we define $\vec{v}_p$ as displacement of tile p between frames 1 and 2, i.e. $T_{2p} - T_{1p}$.

At each iteration k,

$$\vec{v}_p^{k+1} \leftarrow \frac{\sum_{n \in \mathbf{N}} w_n^k \vec{v}_n^k}{\sum_{n \in \mathbf{N}} w_n^k}, \tag{3.9}$$

where n is a neighbor of the tile from neighborhood N, and w is a weighting function that measures the similarity between a tile's motion vector and its neighbor n's motion vector.

The requirements of linear flow consistency call for function w to return a high weight when the vectors were similar, and a low weight when they were dissimilar. This is in fact the support function s introduced above. Fleet and Jepson offered a measure of flow vector similarity that treats flow vectors as 3 dimensional vectors, where the third dimension is unit time [11, 10]. In Chapter 4, we will use this error measurement to compare our experimental results with those of similar algorithms. Representing the velocities as 3-d space-time unit direction vectors, $\vec{v} \equiv \frac{1}{\sqrt{v_1^2+v_2^2+1}}(v_1, v_2, 1)^T$, the error between the correct velocity $\vec{v}_c$ and an estimate $\vec{v}_e$ is

$$\psi_E = \arccos(\vec{v}_c \cdot \vec{v}_e) \tag{3.10}$$

Note, however, that this error measure biases directional error over magnitude error. For our situation, both the magnitude and direction similarity are considered equally important. This can be justified by showing that the magnitude and direction of flow are independant. To expand this claim, we propose that time should not enter the flow vector similarity error measure. This way, we will be considering 2D displacement fields that are measured at arbitrary, possibly varying sampling rates. Since time is now omitted, the similarity measure now deals with 2D displacement vectors, where direction is independant of speed. Therefore, the similarity measure is divided into two components, magnitude similarity $S_m(\vec{v}_1, \vec{v}_2)$ and direction similarity $S_d(\vec{v}_1, \vec{v}_2)$. The direction similarity expression can be obtained from Equation 3.10, replacing the time component (1) with zero.

$$S_d(\vec{v}_1, \vec{v}_2) = \begin{cases} \frac{1+\vec{v}_1 \cdot \vec{v}_2}{2|\vec{v}_1||\vec{v}_2|} & \text{when } |\vec{v}_1||\vec{v}_2| \neq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.11}$$

$$S_m(\vec{v}_1, \vec{v}_2) = \begin{cases} 1 - \frac{|\vec{v}_1 - \vec{v}_2|}{|\vec{v}_1|+|\vec{v}_2|} & \text{when } |\vec{v}_1| + |\vec{v}_2| \neq 0, \\ 1 & \text{otherwise.} \end{cases} \tag{3.12}$$

Both functions have values ranging from 0.0 to 1.0. The overall similarity $S(\vec{v}_1, \vec{v}_2)$ is the linear combination

$$S(\vec{v}_1, \vec{v}_2) = \frac{1}{2}S_m(\vec{v}_1, \vec{v}_2) + \frac{1}{2}S_d(\vec{v}_1, \vec{v}_2) \tag{3.13}$$

This form of weighted averaging is a very non-linear diffusion process. The amount of diffusion between any two neighbors is governed by their local beliefs, in contrast to uniformly-weighted diffusion, which blurs away discontinuities in flow. After an update is performed for each tile, the region (tile pixel) matching error is recomputed in one pass.

The usefulness of the linear flow consistency constraints become clear in cases of motion involving repeating textures. When region matching becomes ambiguous, the flow consistency constraints dominate. This is in opposition to region-matching schemes that embed a displacement-minimization constraint, which would tend to halt patches with ambiguous region matching.

3. Integrating Region Matching and Flow Field Consistency

3.1. Singh's Framework for Optical Flow Computation Singh and Allen proposed a novel framework to unify many contemporary optical flow algorithms that could take directional errors into account for later processing [30]. A key notion that is used in this thesis is how velocity must be propagated from "regions of full information, such as corners, to regions of partial or no information." [30] They propose the *conceptual* separation of the region matching and flow diffusion stages in order to evaluate the constraints, but combine the two operations into one minimization step. They procede to label the information obtained from the first step of region matching as the *conservation information*, measured from the imagery and based on the assumption of conservation of some image property over time. The *neighborhood information* refers to the distribution of the velocity vectors in a small neighborhood.

While key elements of this framework have strong parallels in this thesis, there are also key differences. We decompose the region matching and neighborhood interaction stages computationally, as well as conceptually. The resulting steps suggest the properties of a coordinated conjugate descent, with the added advantage of rapid execution (through simpler stages) and less investigation of perceptually unlikely image events.

In Singh's region matching stage, the error measure (SSD) and estimation method (weighted least squares) are inextricably linked. Many displacements are tested, and the velocity estimate becomes the weighted average of all the displacements, weighted by the SSD similarity. Singh's method offers a covariance matrix to describe the directional uncertainty of the central pixel's motion. Our method instead tests region-matches in a few selected positions, and proceeds to a greedy error gradient descent.

Singh presents a neighborhood interaction stage that is overall consistent with our approach. Neighbors are weighted differently, according to their distance from the central pixel. Together, the neighbors form an opinion of how the motion of the central pixel should behave, including a covariance matrix to describe the directional uncertainty of the neighborhood's opinion. However, the neighborhood updating rule for velocity vectors is essentially a smoothing operator that does not reinforce, but *enforces* parallel flow vectors in a neighborhood. Singh's method decides *how far* and in *what direction* to spread the flow, but does not adapt the diffusion to reflect *how much* any neighbor is consistent with the central pixel. We argue that the extents and direction of diffusion can be decided by the number of iterations applied to the data set, whereas the neighborhood consistency constraint that decides *how much* any given neighbor influences another reflects the perceptual model chosen, and affects the outcome by far more. And the perceptual model we have chosen is that of flow field consistency, not the flow field similarity implied by Singh.

3.2. Practical Considerations for Optimization The error measures for region matching presented in equation set 3.5 are well-understood, as are the vector similarity measures described by equations 3.11, 3.12 and 3.13. Had these error terms been combined into one lumped error, we would have to choose an arbitrary weighting parameter that would significantly change the behaviour of the optimization and the shape of the output data. Either way, the algorithms would perform a coordinated gradient descent.

There are efficiency considerations, however, that support the decoupled optimization for our application. By decoupling the region matching and flow field consistency stages, fewer samplings in the region matching stage (2 dimensions of parameters) will be performed before making a choice. When the error measures are treated as coupled, many more possibilities (4 dimensions of parameters) must be tested. In fact, the coupled optimization will test many perceptually unlikely parameter sets that our decoupled optimization will not bother considering.

Besides accuracy, our goal is to make results available in real-time. This means that the optimization must be capable of producing useable results within a short number of iterations. The coupled optimization case does not degrade gracefully when interupted too early. Our decoupled implementation will at least have results that can be used for subsequent processing. Temporal integration becomes possible when we decouple the stages. Like the coupled optimization, the next iteration of optimization continues processing from whatever state it had achieved previously. But in the case of integrating suggestions from other sources, such as a Kalman filter following the overall image motion, the new error

expression would increase in complexity if coupled with the other error terms. By treating it as a separate, decoupled stage feeding into the region matching stage, we can construct a fairly flexible architecture for fusing data from different sources.

4. Figure/Ground Separation

Identifying the location of objects in a scene using only the optical flow field is a non-trivial task in the general case of moving backgrounds and moving target objects. At this stage, it should be emphasized that this section on figure/ground separation is *not* used in the computation of optical flow. It is instead a convenient layer for other processing, such as demonstrating the validity of the flow field for real-world scenes.

We have also performed experiments using this simple form of tracking to limit the computation of optical flow in subsequent image frames. If the tracking filter is convinced that there is nothing moving beyond a window of $\frac{1}{N}$ of the image area for example, restricting the next iteration to that window would produce a speedup of factor N. The problem, of course, is that this elementary attentive mechanism will ignore all but the first object it caught sight of. Should that first object disappear from the tracking window's view, an opposing mechanism would need to relax the tracking filter enough to expand the tracking window, searching for other motion in the image sequence. These experiments will not be presented in this thesis. All the examples and experiments performed here were computed looking at the entire image.

Our system typically encounters scenes with moving target objects and a stationary background. The figure/ground separation problem is thus simplified: anything moving is not part of the background. A moving object can be isolated from its background by associating a certainty that a tile is observing image motion for each tile. Applying a Kalman filter to each tile's measure of occupation provides stability to this representation. For each frame pair k, the weight w_k is the number of tiles experiencing motion, i.e. those tiles moving faster than a threshold v_θ, typically 0.5 pixels / frame.

Tracking might be achieved adequately by examining differences of images, but difficulties abound. The assumptions made in difference of image tracking include small displacement between frames and highly-textured surfaces. What usually results is a rough contour that does not completely surround the target object. The countour occupies image areas that have been recently occupied and/or recently vacated: there is no information about where the object presently is. A figure/ground separation, by contrast, generates a filled

map locating all image regions occupied by the moving object, not the image artifacts that may or may not belong to the moving object.

A rapid scheme for associating a normalized measure of occupation for each tile involves applying a threshold to each tile's velocity. The heuristic used here implies that displacements of one pixel or more probably represent moving regions, while smaller displacements are more probably stationary background regions. The estimate of the measure of occupation can be expressed as

$$\hat{M}_{occ}(p) = \begin{cases} 0.9 & \text{when } |v(p)| > 0.5 \text{ pixels/frame} \\ 0.1 & \text{otherwise.} \end{cases} \tag{3.14}$$

So far, this certainty of occupation is very local, mapping one velocity vector to one occupation measure, and tends to be noisier than the flow field, due to the thresholding operation. A more useful quantity would integrate this information over time or over larger areas. For our purposes, we adopted integration over time using a Kalman filter. Each tile's certainty of occupation becomes a weighted average between the current estimate and the previous estimate. The weights are chosen to represent the number of tiles in motion in the present frame and those from the previous frame.

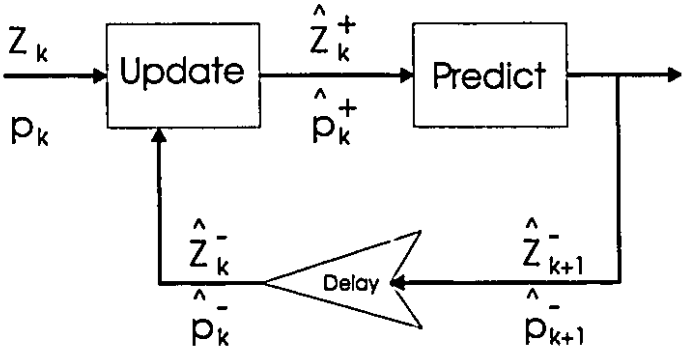


FIGURE 3.6. **Kalman Filter.** Block diagram of the classic Kalman Filter. Measurements at stage k are denoted as Z_k, while their error variances are p_k.

We propose a form of Kalman filtering that is applied to each region in the image. Traditionally, Kalman filtering uses the variance of measurement error to weight incoming measurements, and the structure of the filter is shown in Figure 3.6. The Kalman filter maintains an estimate $\hat{Z}_k$ and its variance $\hat{p}_k$. The measurements Z_k have a known error variance p_k, and the estimate is updated using equations 3.16.

$$\hat{Z}_k^+ = \frac{\frac{Z_k}{p_k} + \frac{\hat{Z}_k^-}{\hat{p}_k^-}}{\frac{1}{p_k} + \frac{1}{\hat{p}_k^-}} \tag{3.15}$$

$$\hat{p}_k^+ = \frac{1}{\frac{1}{p_k} + \frac{1}{\hat{p}_k^-}} \tag{3.16}$$

The Kalman filter's strength, however, is its ability to make predictions of upcoming measurements based on a model from previous measurements. If $f()$ is a function that predicts the next measurement from the present estimate, then we can user equations 3.18 to estimate the prediction and its error variance.

$$\hat{Z}_{k+1}^- = f(\hat{Z}_k^+) \tag{3.17}$$

$$\hat{p}_{k+1}^- = \left(\frac{\partial f}{\partial Z}\right)^2 \hat{p}_k^+ \tag{3.18}$$

In our case however, the prediction function f is identity, meaning that the upcoming position of the target object is exactly where it was last seen. Because our occupation does not have an error variance measure *per se*, we make use of the inverse relationship between population size and sampling variance. The weights $\frac{1}{p}$ become replaced by w, which is the number of tiles experiencing significant motion. Thus, the Kalman filtered certainty of occupation M_{occ} for tile p can be expressed as

$$M_{occ}^{k+1}(p) \leftarrow \frac{\hat{M}_{occ}^{k+1}(p)w^{k+1} + M_{occ}^{k}(p)w^{k}}{w^{k+1} + w^{k}}, \tag{3.19}$$

The end result is that a map of image regions is produced, isolating moving objects from the stationary background, integrated over time. This information can be used for 2-D tracking purposes.

This form of Kalman filtering allows moving objects to remain segmented from the background even when motion stops: the figure/ground separation map will not change until new motion is introduced. This allows targets that were identified by their motion to come to rest and retain their tag as a zone capable of motion.

If figure/ground separation were the only source of higher-level information, one could isolate each patch of motion and track them, predicting the motion for the next frame. This predicted flow-field can be used in the measurement stage to seed correspondence searches.

This notion includes a form of graceful degradation. By design, upcoming motion events are predicted by previously measured motion events. In the case where old motion patches halt, and new motion patches come into being, the older patches will not be reinforced and will lose importance, while newer motion will be reinforced and tracked instead.

One practical use of this figure/ground separation is tracking single moving objects against an unmoving background. A tracking window is computed to enclose the moving object. In order to double and quadruple processing speed, subsequent region-matching stages are limited to the tracking region. A live demonstration of this property has successfully shown that object tracking can be accomplished using only a partial optical flow field.

5. Using Higher-Level Information

Presently, most optical flow algorithms are designed and implemented as self-contained stages that take an image sequence as input and produce a flow field as ouput, perhaps including an error measure for the confidence of each flow vector. These algorithms exist as testbeds for specific characteristics of early vision and are designed as end-products: any connections to other processing stages are messy.

Our algorithm was designed to meet the goals of real-time performance and ease of integration into a larger vision system where each stage is assisted by adjacent stages. A simple region-matching approach becomes a cost-effective optical flow tool when it is integrated with flow field consistency.

But more important constraints become available after 3D information emerges through Structure from Motion processing. The 3D surfaces of objects in the image sequence can be tracked over time, and predicted 3D motion can be easily projected into a predicted 2D optical flow field, seeding the correspondence searches for the next set of images. This surface transport model would improve the quality of the flow field with the obvious benefit of a 3D scene motion representation. But if the transport model is blindly combinined into the flow field measurement, the added complexity would tend to slow down computation for relatively little improvement in the flow field. Instead, we propose adding the transport model as a higher-level stage as suggested in Figure 2.2.

Anandan's proposed architecture of optical flow does not deal with integrating other sources of motion. Each scale of region matching only uses information propagated from a coarser scale: there is no allowance for suggestions or predictions from previous image frame pairs. In short, the predictions are not temporal, they are from coarse to fine scales within

one instant. This choice discards valuable accumulated tracking information that could speed up subsequent region matching stages. Although Anandan's algorithm computes optical flow from images very well, it is memoryless, and thus cannot make predictions to make computations easier.

In effect, top-level information is formed from low-level measurements and constraints, while the low-level measurement stage is steered or directed by the top-level information. This not only resolves local-global issues in a principled way, but also integrates bottom-up and top-down data flow. This architecture would lead to improved optical flow measurement, which in turn would lead to better structure from motion. This information, integrated over time, can be fit to compact, parametric surface models that describe the moving objects in a scene, while tracking and predicting their motion.

6. Implementation Considerations

6.1. Time/Quality Trade-off Note that the region matching and flow field consistency constraints could have been implemented as one error function to minimize. It would appear that by alternatively measuring the region-matching error and enforcing the flow field consistency, the end effect is to coordinate a gradient descent locally for each tile, while diffusing measurements to neighboring tiles.

6.2. Fixed Time per Iteration But by decoupling the stages as is evidenced by the primate visual cortex architecture, we achieve brief steps that can be implemented compactly and executed quickly. This way, each iteration is brief, and can either be repeated over the same image pair, or pipelined to another processing stage while new information is gathered. Fast implementation of the optical flow algorithm becomes possible.

6.3. Pre-computation of tile positions The tiles were uniformly distributed over the image, and tests were performed using overlapping arrangements and non-overlapping arrangements, with various tile sizes, ranging from 3×3 pixels to 8×8 pixels, with 4×4 yielding a reasonable tradeoff of time to compute versus quality. During the region-matching stage, the algorithm tests a fixed number of tile displacements, searching around the predicted correspondence, but also testing for the case of sudden stopping. The latter case occurs most often when an object in the image sequence translates a distance the dimension of a tile. At one instant, the tile sees the object; at the next, the background.

CHAPTER 4

Experimental Results and Discussion

Twelve image sequences are presented here, consisting of four synthetic sets, three natural-appearance synthetic sets, four well-known natural image sequences, and one image sequence typical of the algorithm's intended environment.

The experimental results are followed by a summary Discussion in Section 6, which will unify the claims made in the Proposal (Chapter 2, Section 5) and the measurements performed on the algorithm.

In the experimental section, we will frequently refer to the related works of Anandan and Singh. The optical flow algorithms of P. Anandan and A. Singh are the most closely related works to this algorithm. Anandan employs a Laplacian pyramid and a coarse-to-fine SSD-based matching strategy, while Singh employs a comparable hierarchical coarse-to-fine strategy. Our work differs significantly in that we also employ the resulting geometry of the estimated flow field to reduce the noise in the flow field, refine the measurement process for further iterations and predict upcoming flow field events. Where appropriate, we will also mention how our results compare to more general classes of optical flow algorithms.

1. Synthetic Sequences

1.1. Positive Results This data set was obtained from the Barron et al. archive, and consists of the superposition of sinusoids. Error here is reported using the same error metric as reported in [4] and [5], namely the angular deviation from the correct flow direction. Representing the velocities as 3-d space-time unit direction vectors, $\vec{v} \equiv \frac{1}{\sqrt{v_1^2+v_2^2+1}}(v_1, v_2, 1)^T$, the error between the correct velocity $\vec{v}_c$ and an estimate $\vec{v}_e$ is

$$\psi_E = \arccos(\vec{v}_c \cdot \vec{v}_e) \tag{4.1}$$

The tests were performed on the *mysineB-6* (**Sinusoid 1**) and *mysineC-16* (**Sinusoid 2**) data sets, where the motions of the entire image plane are known to be (1.583, 0.863) pixels / frame and (1.0, 1.0) pixels / frame, respectively. The algorithm used a grid of 10×10 tiles, each tile of 6×6 pixels, applying 5 iterations between each image pair. The results shown are accumulated over the entire image sequence, not just a single frame pair. The results for our algorithm and those of Anandan and Singh are summarized in table 4.1. Sample frames and generated flow fields from the two sequences are shown in Figures 4.1 and 4.3. The flow fields from the algorithms of Anandan and Singh are shown in Figure 4.2.

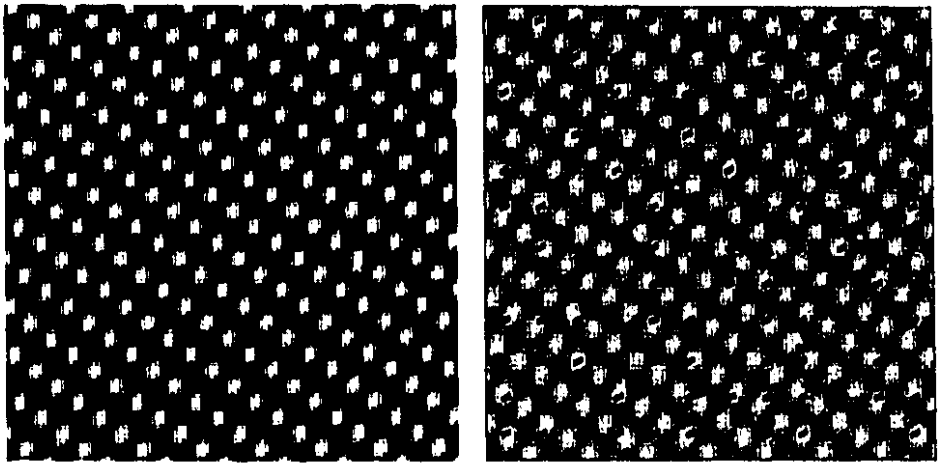
Technique	Average Error	Standard Deviation
Us	5.21°	$\approx 0.000°$
Anandan	30.80°	5.45°
Singh ($n = 2$, $w = 2$, $N = 2$)	2.24°	0.02°
Singh ($n = 2$, $w = 2$, $N = 4$)	91.71°	0.04°

TABLE 4.1. Results of **Sinusoid1** test data. Experimental results for Anandan and Singh are taken from [4] and [5].

Technique	Average Error	Standard Deviation
Us	0.0452°	0.3607°
Anandan	-	-
Singh	-	-

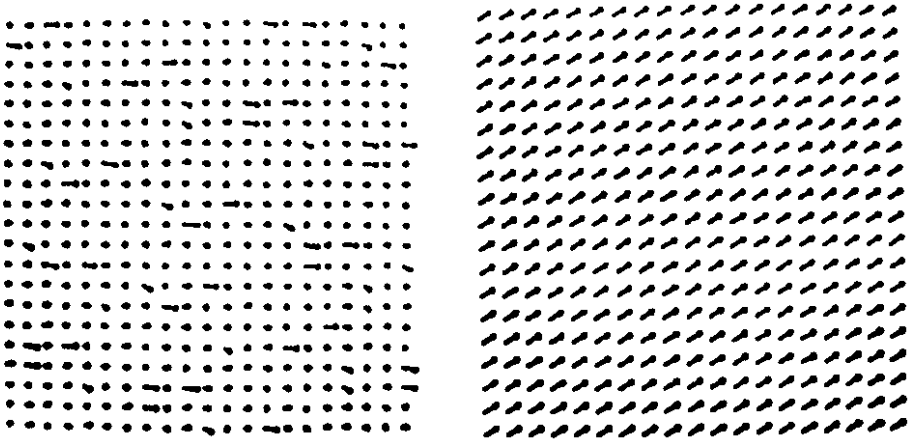
TABLE 4.2. Results of **Sinusoid2** test data. Experimental results for Anandan and Singh are unavailable from [4] and [5], but are described as "unchanged".

Our algorithm thus responds very strongly to this class of stimulus, namely uniform translations. Note that the displacements for **Sinusoid 1** are not integer displacements, and rival other region-matching methods. Our algorithm's success for this class of input can be explained by the flow field consistency enforcement. The local information provided by region matching is propagated to neighbors who improve their estimates with the new information. With weighted averaging of neighbors, non-integer displacements can be obtained despite the integer-based region-matching.



(a) (b)

FIGURE 4.1. **Sinusoid 1.** An image frame from the sequence (a), and superimposed reconstructed flow field, (b). Note that the flow image has been subdued for pictorial purposes only.

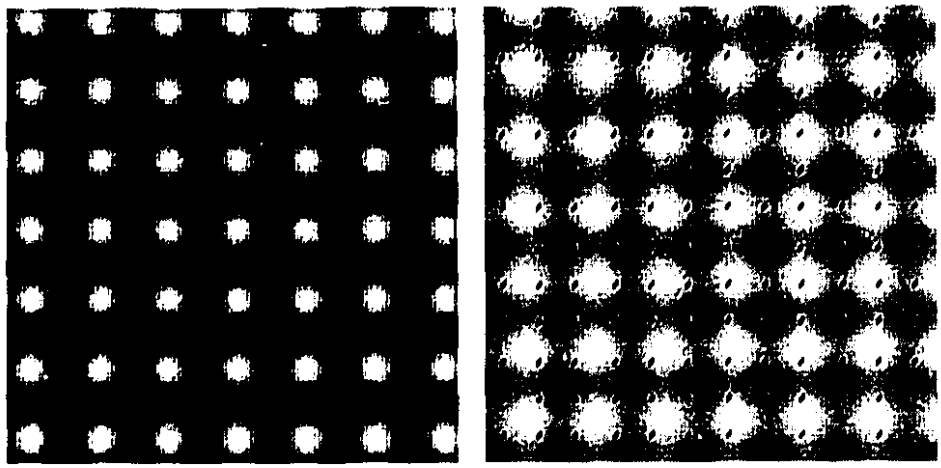


(a) (b)

FIGURE 4.2. **Sinusoid 1, other algorithms.** The flow field for Anandan's algorithm is shown in (a). Singh's algorithm produced the flow field shown in (b). Both plots were obtained from [4].

1.2. Negative Results Neither of the following results are considered catastrophic failures, since most optical flow algorithms encounter similar or worse results. The purpose of this section is to show how our algorithm deals with ambiguous synthetic scene changes.

One of the most difficult environments for an optical flow algorithm involves the ambiguity of the aperture problem. The case examined here is where our algorithm has no



(a) (b)

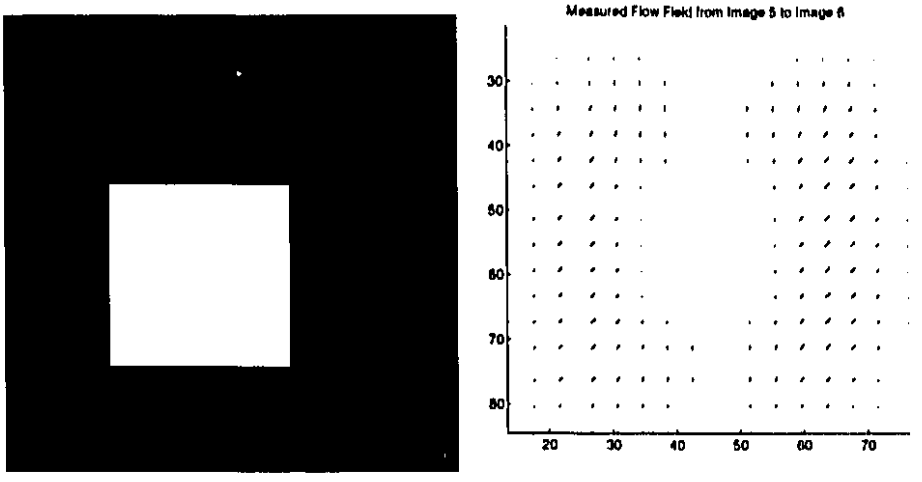
FIGURE 4.3. **Sinusoid 2.** An image frame from the sequence (a), and superimposed reconstructed flow field, (b). As before, the flow image has been subdued for pictorial purposes only.

textures to track, and is a perfect case to demonstrate how the algorithm degrades gracefully. Shown in Figure 4.4 are the experimental data obtained from a translating uniformly white square on a uniformly black background. As can be seen, some motion is perceived in the upper-right direction, but the distribution of magnitudes does not correspond to what a human observer perceives. The test was repeated on a translating square with an intermediate gray-level boundary around the square, producing similar results shown in Figure 4.5. For comparison, the results obtained in the Barron et al. experiments for Anandan and Singh are shown in Figure 4.6.

The explanation for this behaviour is not complicated: the region-matching information is inconclusive in areas of no texture. The algorithm does perceive the motion of the boundaries, but flow field consistency dominates over the image matching. Still, the flow field consistency needs actual measurements to anchor the image motion, which is unavailable.

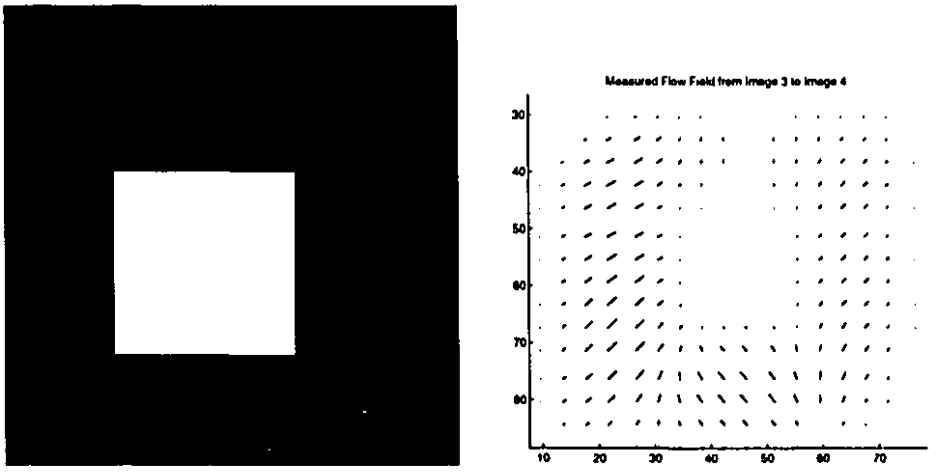
It is important to note that our algorithm's output appears better than the optical flow algorithms presented in [5], but at least as good as those presented in [4]. We consider our test results to be less than adequate compared to human observers, but superior to other algorithms tested.

One could argue that human observers would perceive the motion of geometric features of this synthetic scene, namely the boundary between the square and its background. Gestalt psychologists could explain how the motion of a boundary infers the motion of the



(a) (b)

FIGURE 4.4. **Translating Square 1.** An image frame from the sequence (a), and reconstructed flow field, (b).



(a) (b)

FIGURE 4.5. **Translating Square 2.** An image frame from the sequence (a), and reconstructed flow field, (b).

region enclosed by the boundary in the absense of textural stimuli. This reasoning would suggest that global constraints or regularization would dominate local texture measures in these image sequences, hence the non-ideal.

2. Natural-Appearance Synthetic Sequences

2.1. Yosemite Sequence The Yosemite sequence, created by Lynn Quam, was chosen as a complex text case with a range of velocities, occluding edges and severe aliasing [4].

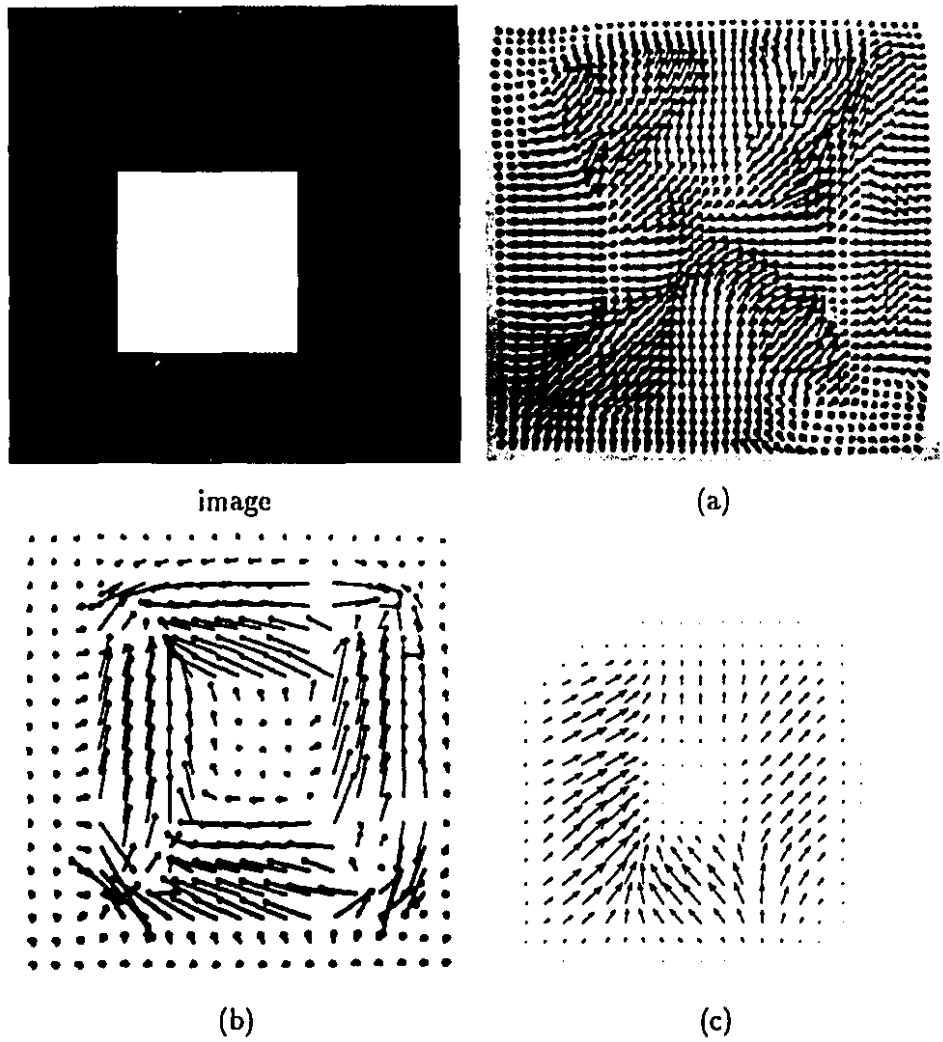
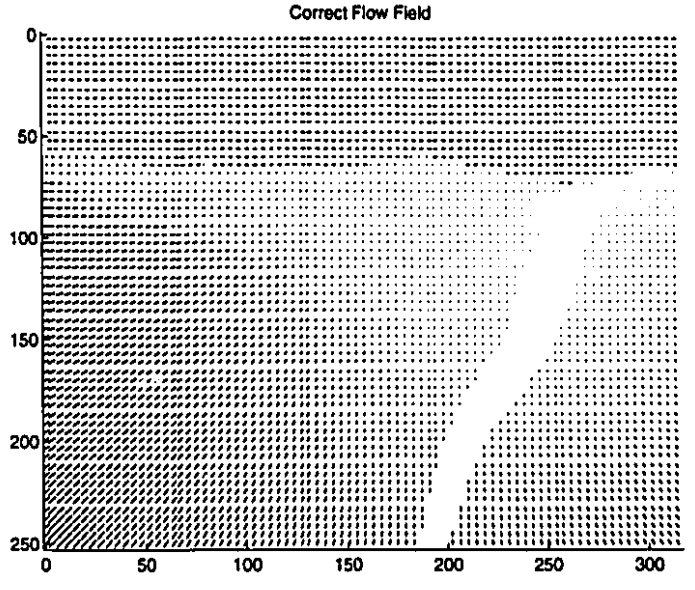


FIGURE 4.6. **Translating Square 2, other algorithms.** The flow field for Anandan's algorithm is shown in **(a)**. Singh's algorithm produced the flow field shown in **(b)**. Both plots were obtained from [4]. Our results were resampled and are shown in a similar format in **(c)**.

A frame of the sequence and the correct flow field appear in Figure 4.7. This experiment used a grid size of 80 × 60 tiles, each tile consisting of 8 × 8 pixels. Five iterations were performed on each frame pair.

The sequence was tested in two ways, first using every flow vector, regardless of confidence, and the second time, vectors falling above an error threshold were ignored. This is made possible by the measurement of region-matching error during the minimization. The applied error threshold was 0.025, and affected 32.4% of the flow vectors.

(a)



(b)

FIGURE 4.7. **Yosemite Sequence.** An image frame from the sequence (a), and correct flow field, (b).

The results for the thresholded and unthresholded experiments are summarized in table 4.3.

The flow field obtained is shown in Figure 4.8, and can be compared with the results of Anandan and Singh shown in Figure 4.9. The error details from the experiment are shown in Figures 4.10 and 4.11. Note that our algorithm is competitive with the algorithms of Anandan and Singh. As explained earlier, our algorithm represents flow information as a coarse data set, versus the conventional dense data set (represented in table 4.3 as

Technique	Valid Data	Average Error	Std. Dev.	< 1° Error	< 2° Error	< 3° Error
Us Unthresholded	100%	17.16°	17.50°	1.96%	7.25%	13.35%
Us Threshold=0.025	67.6%	15.13°	15.57°	2.43%	9.37%	16.86%
Anandan	100%	13.46°	15.64°	1.1%	4.1%	8.0%
Singh (st 1, $n = 2$, $w = 2$)	100%	15.28°	19.61°	1.3%	3.7%	7.0%
Singh (st 1, $n = 2$, $w = 2$, $\lambda_1 \leq 6.5$)	11.3%	12.01°	21.43°	12.3%	24.4%	34.6%
Singh (st 2, $n = 2$, $w = 2$)	100%	10.44°	13.94°	-	-	-
Singh (st 2, $n = 2$, $w = 2$, $\lambda_1 \leq 0.1$)	97.7%	10.03°	13.13°	2.4%	7.4%	12.6%

TABLE 4.3. Results of Yosemite test data. Mean and standard deviation experimental results for Anandan and Singh are taken from [5], while the low angular error distribution were obtained from [4].

percentages of the image surface used. In particular, our algorithm has a tighter distribution of low-error flow data than either Anandan or Singh in most cases. We concede, of course, that Singh's mean error and standard deviation is better than ours in the thresholded case.

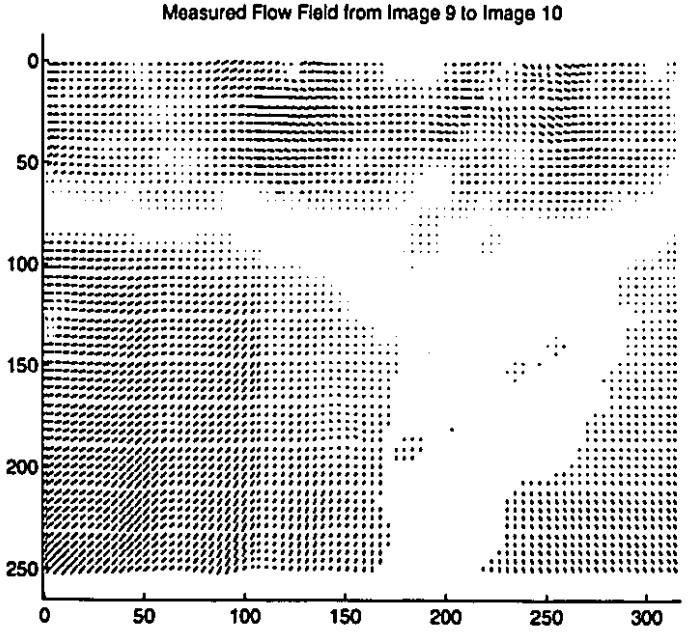


FIGURE 4.8. **Yosemite Sequence.** Measured flow field

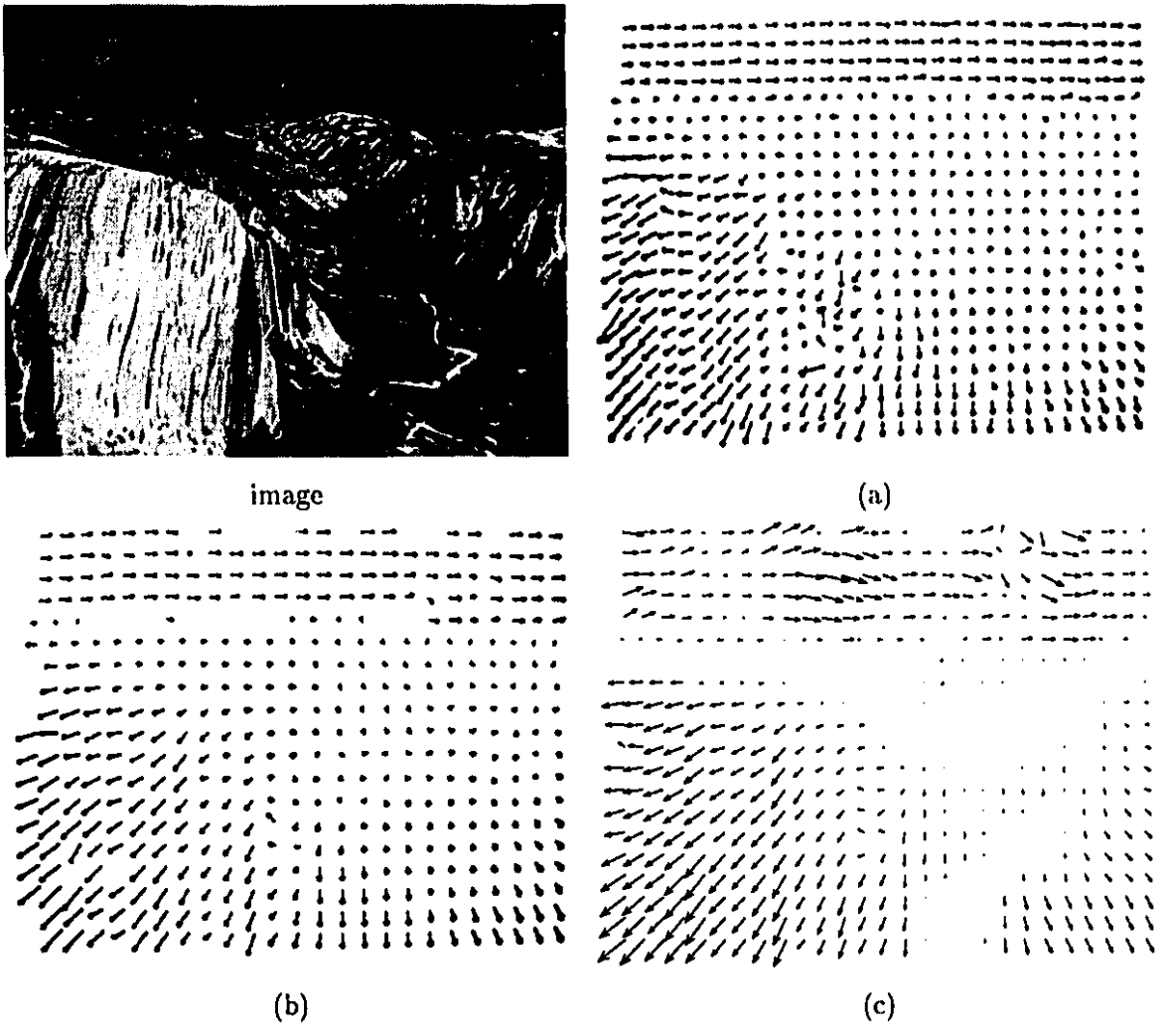


FIGURE 4.9. **Yosemite Sequence, other algorithms.** The flow field for Anandan's algorithm is shown in **(a)**. Singh's algorithm produced the flow field shown in **(b)**. Both plots were obtained from [4]. Our results were resampled and are shown in a similar format in **(c)**.

What is not shown here is the qualitative performance of Anandan's or Singh's algorithms on this image sequence. There are outliers in both algorithms' flow fields that are not allowed to occur in our algorithm. This qualitative information can be found in [5].

2.2. Translating Tree Sequence This image sequence simulates translational camera motion with respect to a textured planar surface, shown in Figure 4.12. In this case, the camera moves normal to its line of sight along its X-axis, with velocities all parallel with the image x-axis, with speeds between 1.73 and 2.26 pixels/frame. Our algorithm was performed with 4 iterations, using 16×16 pixel tiles in a grid of 20×20 tiles.

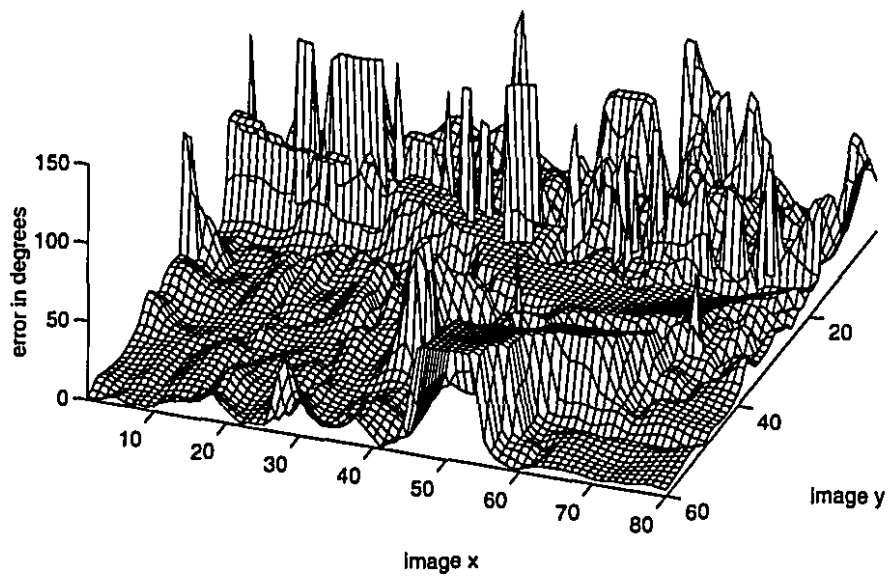


FIGURE 4.10. **Yosemite Sequence, error surface.** An angular error surface for frame 10.

The flow field obtained is shown in Figure 4.13, and can be compared with the flow fields from Anandan's and Singh's algorithms in Figure 4.14.

Technique	Valid Data	Average Error	Std. Dev.	$< 1°$ Error	$< 2°$ Error	$< 3°$ Error
Us	100%	2.44°	3.44°	37.8%	64.0%	81.8%
Anandan	100%	4.54°	3.10°	5.7%	19.1%	36.0%
Singh (st 1, $n = 2$, $w = 2$)	100%	1.64°	2.44°	-	-	-
Singh (st 1, $n = 2$, $w = 2$, $\lambda_1 \leq 5.0$)	41.4%	0.72°	0.75°	79.7%	93.8%	97.7%
Singh (st 2, $n = 2$, $w = 2$)	100%	1.25°	3.29°	-	-	-
Singh (st 2, $n = 2$, $w = 2$, $\lambda_1 \leq 0.1$)	99.6%	1.11°	0.89°	57.4%	84.6%	98.5%

TABLE 4.4. Results of Translating Tree test data. Mean and standard deviation experimental results for Anandan and Singh are taken from [5], while the low angular error distribution were obtained from [4].

From these results, our algorithm is clearly competitive with the other region-matching optical flow methods, both from the low mean error and tight error deviation, but also in terms of tight clustering toward zero error, shown in the error histogram of Figure 4.15. It

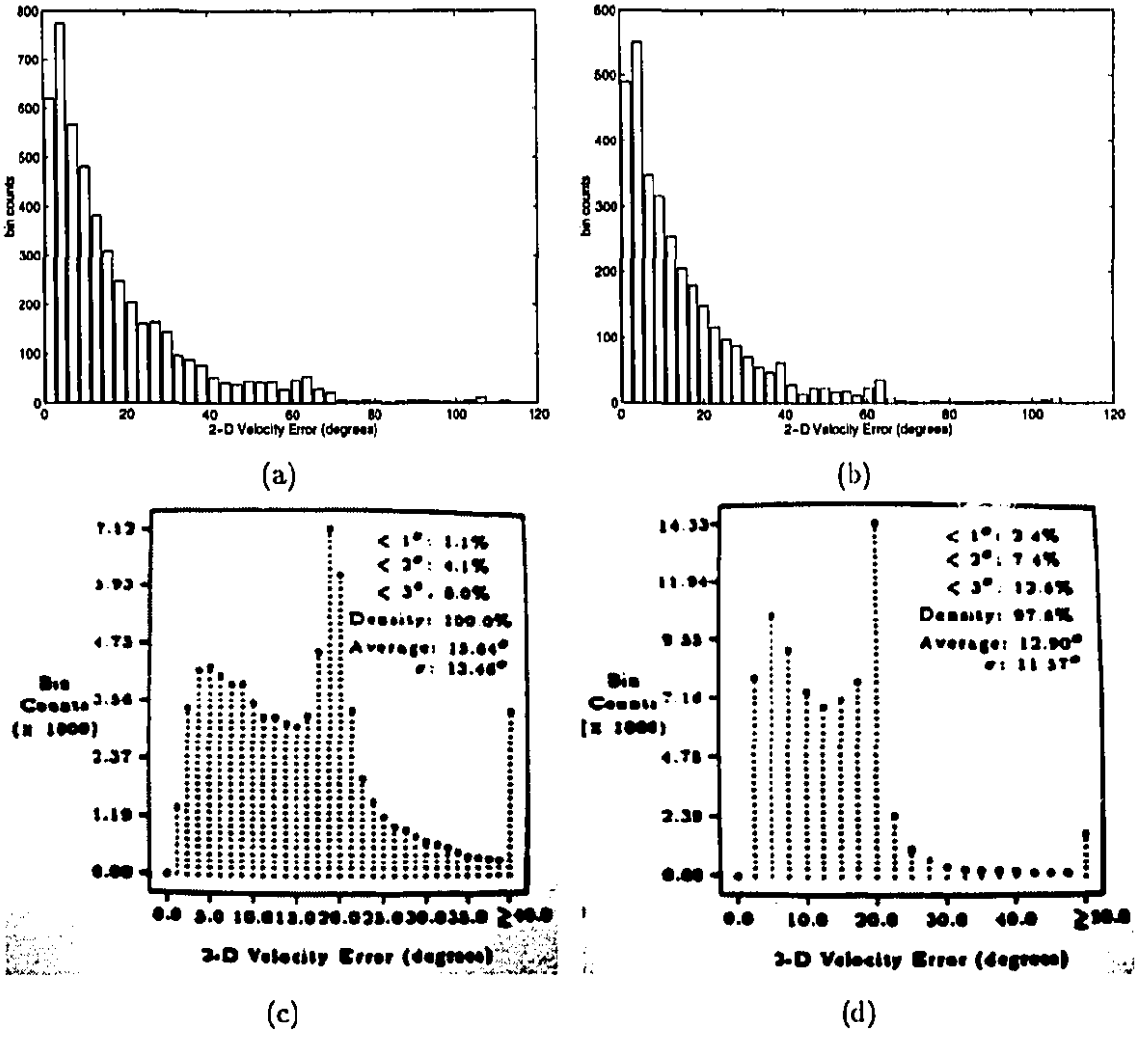
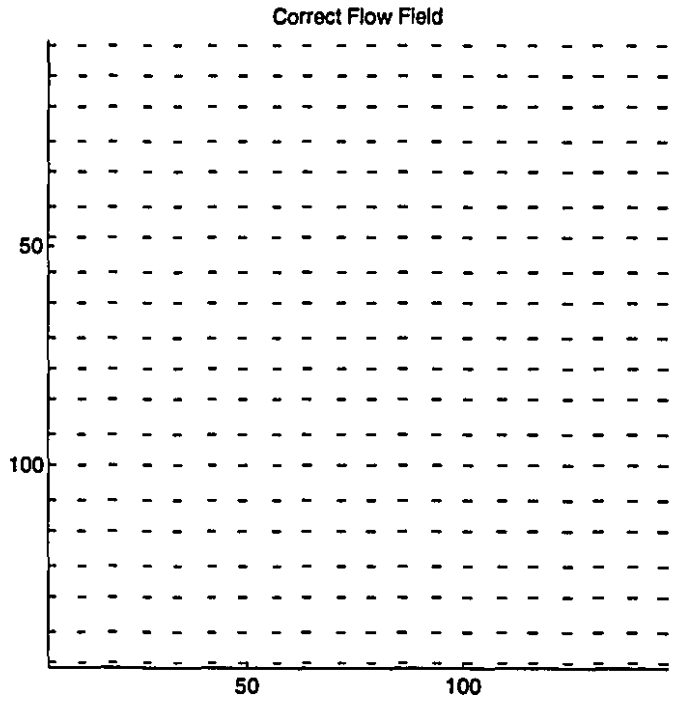


FIGURE 4.11. **Yosemite Sequence, error histograms.** Shown are angular error distribution for the unthresholded (a) and thresholded (b) experiments. The error distributions from (c) Anandan (unthresholded) and (d) Singh ($\lambda_1 \geq 0.1$) were obtained from [4].

is important to note that the apparent motion can not be measured directly in many areas of the image, in particular the low-contrast, untextured background. Because our method allows neighborhood flow field consistency to dominate in these under-determined zones, a globally meaningful flow field emerges which is influenced by the successful matching of higher-contrast, textured regions. The algorithm, it should be emphasized, is not using any underlying motion transport model, such as planar motion, to determine the displacement of the pixel regions in the image, and yet returns a flow field that one would expect of a higher-level ("global" motion) interpretation.

(a)



(b)

FIGURE 4.12. **Translating Tree Sequence.** An image frame from the sequence **(a)**, and correct flow field, **(b)**.

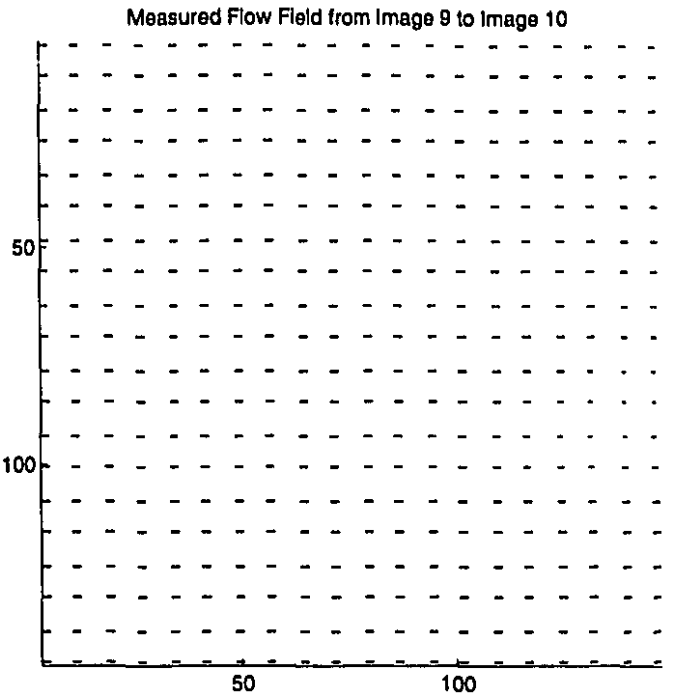


FIGURE 4.13. **Translating Tree Sequence.** Measured flow field

2.3. Diverging Tree Sequence This image sequence simulates translational camera motion with respect to a textured planar surface, shown in Figure 4.16. In this case, the camera moves along its line of sight. The focus of expansion is at the center of the image, with velocities varying from 1.279 pixels/frame on the left side and 1.86 pixels/frame on the right. Our algorithm was performed with 4 iterations, using 16×16 pixel tiles in a grid of 20×20 tiles.

The flow field obtained is shown in Figure 4.17, and can be compared with the flow fields from Anandan's and Singh's algorithms in Figure 4.18.

From these results, our algorithm is clearly competitive with the other region-matching optical flow methods, both from the low mean error and tight error deviation, but also in terms of tight clustering toward zero error, shown in the error histogram of Figure 4.19. Again, note that the apparent motion can not be measured directly in many areas of the image, in particular the low-contrast, untextured background. The flow-field consistency does not use an underlying motion transport model, such as planar motion. The algorithm, yet returns a flow field that one would expect of a higher-level ("global" motion) interpretation. The flow-field consistency is sufficient and necessary to recover diverging flow fields such as this.

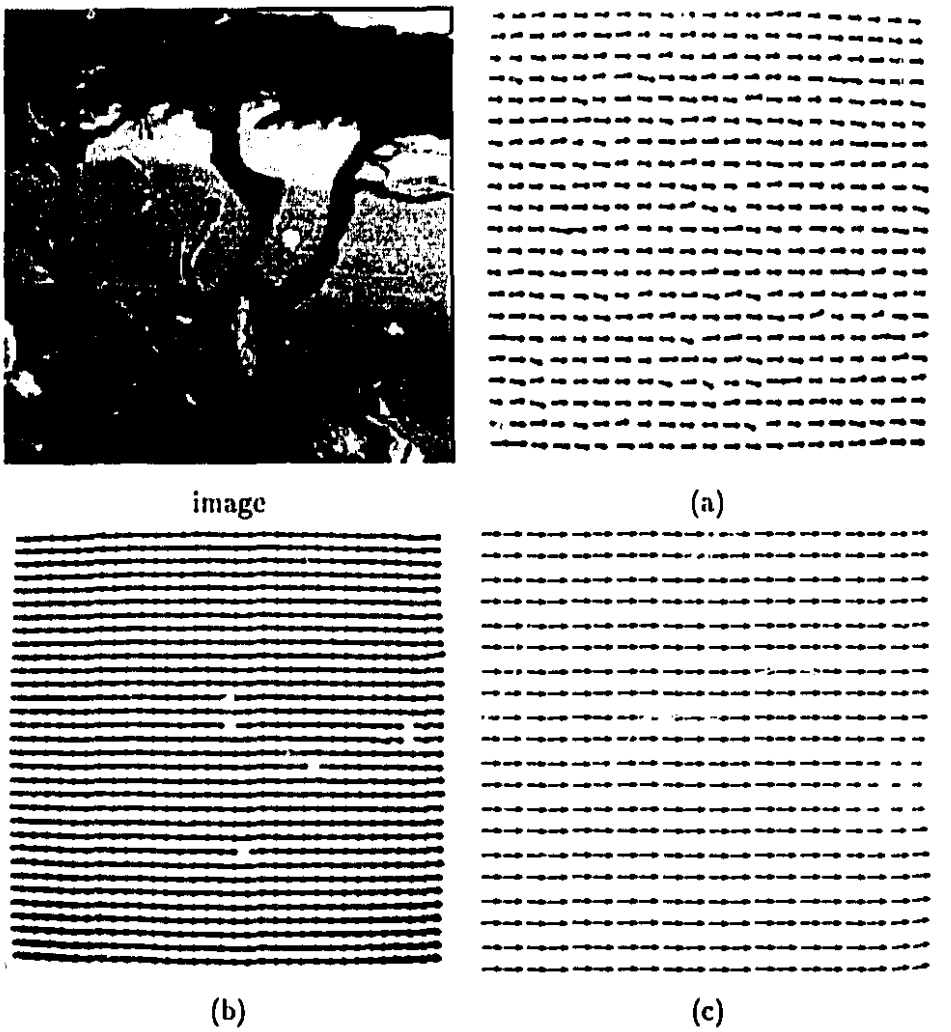


FIGURE 4.14. **Translating Tree Sequence, other algorithms.** The flow field for Anandan's algorithm (unthresholded) is shown in (a). Singh's algorithm (step 2, $\lambda_1 \geq 0.1$) produced the flow field shown in (b). Both plots were obtained from [4]. These were the best of the results produced by the two algorithms. Our results were resampled and are shown in a similar format in (c).

3. Natural Sequences

3.1. Hamburg Taxi Sequence The Hamburg taxi sequence has four principal moving objects, including a taxi turning the corner, a car in the lower left moving from left to right, and a van in the lower right moving from right to left. A pedestrian is also walking on the sidewalk in the upper left, but the motion was too far below the error threshold for our algorithm to detect. Alongside the image of the flow field superimposed on the scene in Figure 4.20 are velocity and occupation maps. The algorithm used only one iteration

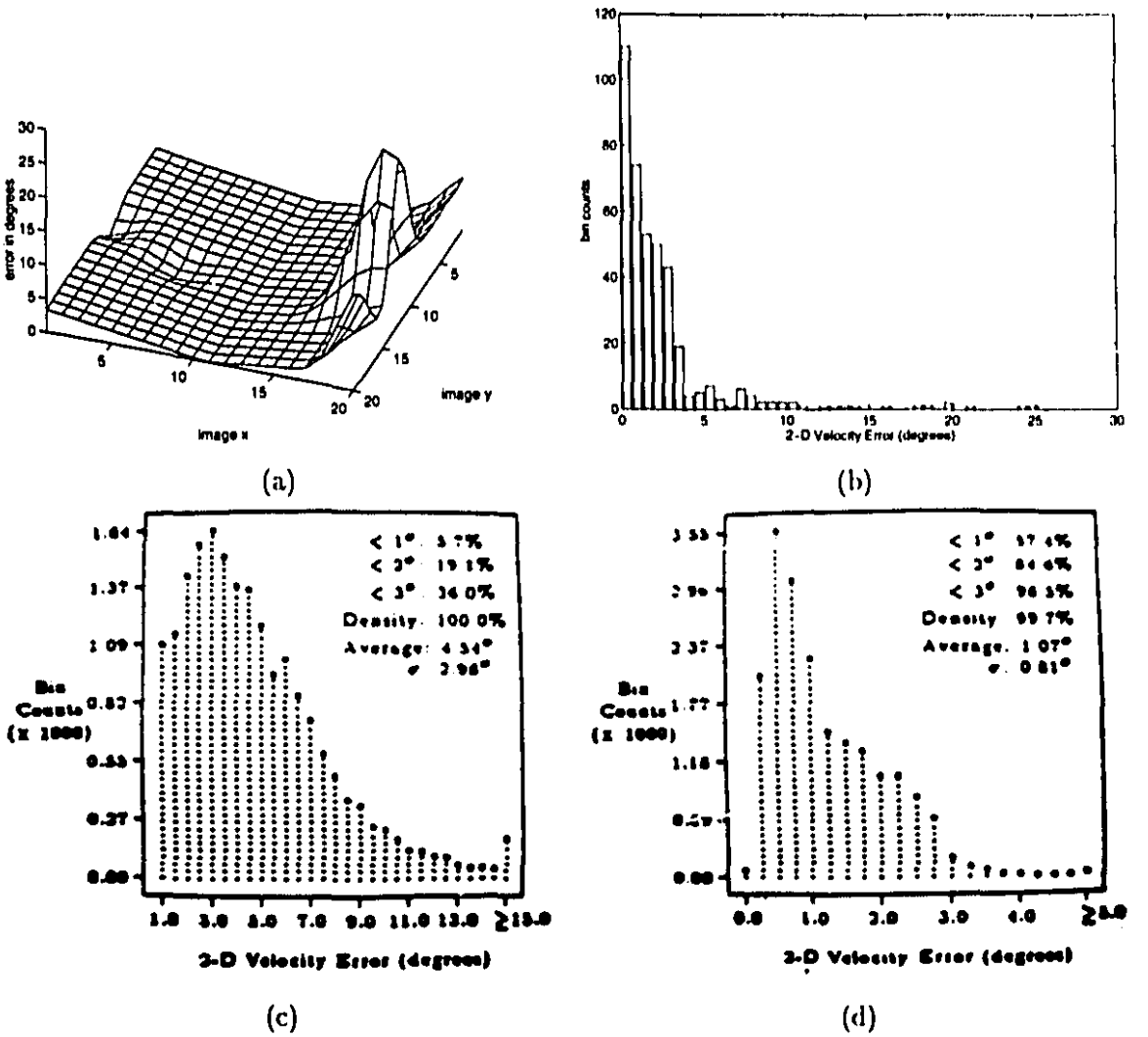
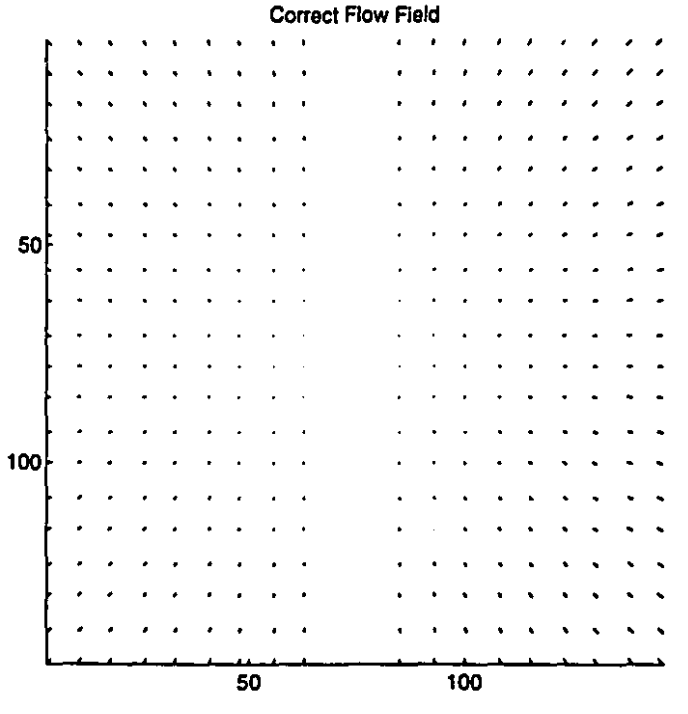


FIGURE 4.15. **Translating Tree Sequence, error surface and histograms.** An angular error surface for frame 10 (a). Also shown is the angular error distribution for the unthresholded (b) experiment. The error distributions from (c) Anandan (unthresholded) and (d) Singh ($\lambda_1 \geq 0.1$) were obtained from [4].

per image pair, with a grid of 80 × 60 tiles, each tile at 6 × 6 pixels. For clarity, an image frame and the flow field are shown separated in Figure 4.21. The results of Anandan's and Singh's algorithms on the same data set are shown in Figure 4.22.

Qualitatively, the background is shown to be immobile, despite the large amount of white noise and aliasing present in the image sequence. Anandan's output does not show the vertical displacement of the taxi, while Singh's output shows less coherent motion

(a)



(b)

FIGURE 4.16. **Diverging Tree Sequence.** An image frame from the sequence (a), and correct flow field, (b).

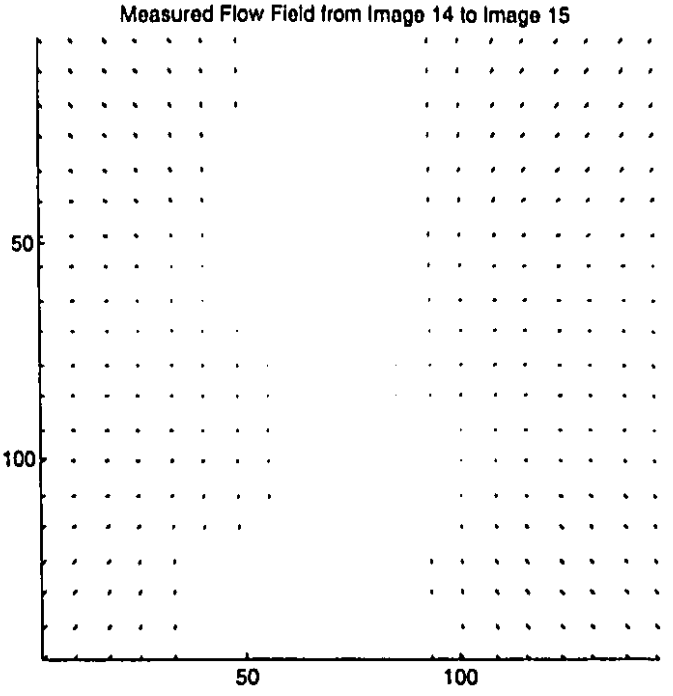


FIGURE 4.17. **Diverging Tree Sequence.** Measured flow field

Technique	Valid Data	Average Error	Std. Dev.	$< 1°$ Error	$< 2°$ Error	$< 3°$ Error
Us	100%	9.21°	5.06°	1.0%	2.8%	8.0%
Anandan	100%	7.64°	4.96°	2.5%	8.5%	16.3%
Singh (st 1, $n = 2$, $w = 2$)	100%	17.66°	14.25°	0.6%	1.9%	4.0%
Singh (st 1, $n = 2$, $w = 2$, $\lambda_1 \leq 5.0$)	3.3%	7.09°	6.59°	7.1%	19.7%	30.7%
Singh (st 2, $n = 2$, $w = 2$)	100%	8.60°	5.60°	-	-	-
Singh (st 2, $n = 2$, $w = 2$, $\lambda_1 \leq 0.1$)	99.0%	8.40°	4.78°	0.8%	3.4%	7.3%

TABLE 4.5. Results of Diverging Tree test data. Mean and standard deviation experimental results for Anandan and Singh are taken from [5], while the low angular error distribution were obtained from [4].

(without thresholding) and noise along the bottom of the image frame where there is no motion.

3.2. SRI Tree Sequence This is a low-contrast image sequence, where the camera translates perpendicularly to the line of sight. There is a large amount of occlusion, and

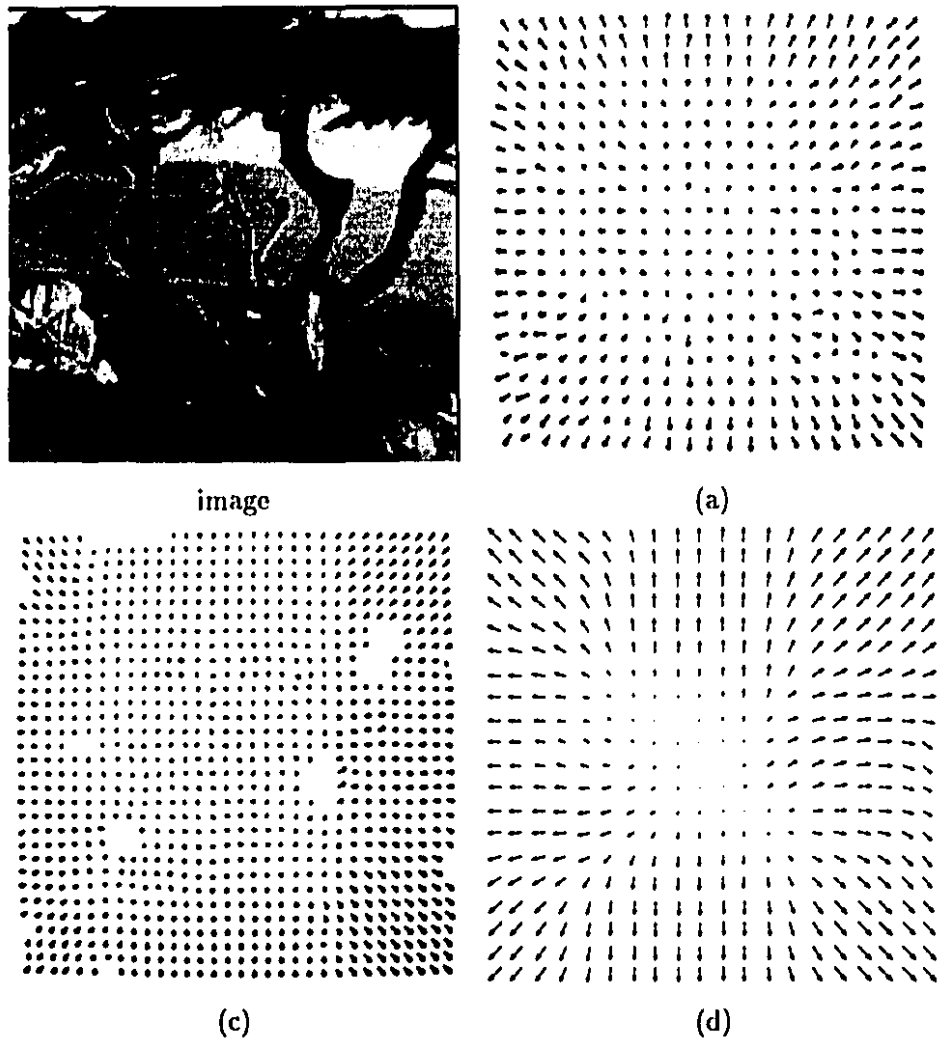


FIGURE 4.18. **Diverging Tree Sequence, other algorithms.** The flow field for Anandan's algorithm (unthresholded) is shown in (a). Singh's algorithm (step 2, $\lambda_1 \geq 0.1$) produced the flow field shown in (b). Both plots were obtained from [4]. These were the best of the results produced by the two algorithms. Our results were resampled and are shown in a similar format in (c).

the highest image velocities were found to be just under 3 pixels per frame. A sample frame of the image sequence and the measured flow field are shown in Figure 4.23.

The algorithms of both Anandan and Singh do reasonably good jobs on the SRI tree sequence, as shown in Figure 4.24. But both have discontinuous flow fields in locations where the motion is fluid, whereas our output has a consistent flow field.

In this special case of camera translation, we can use the kinetic depth effect (proximity proportional to velocity) to show an approximate depth map of the scene, shown in Figure 4.25.

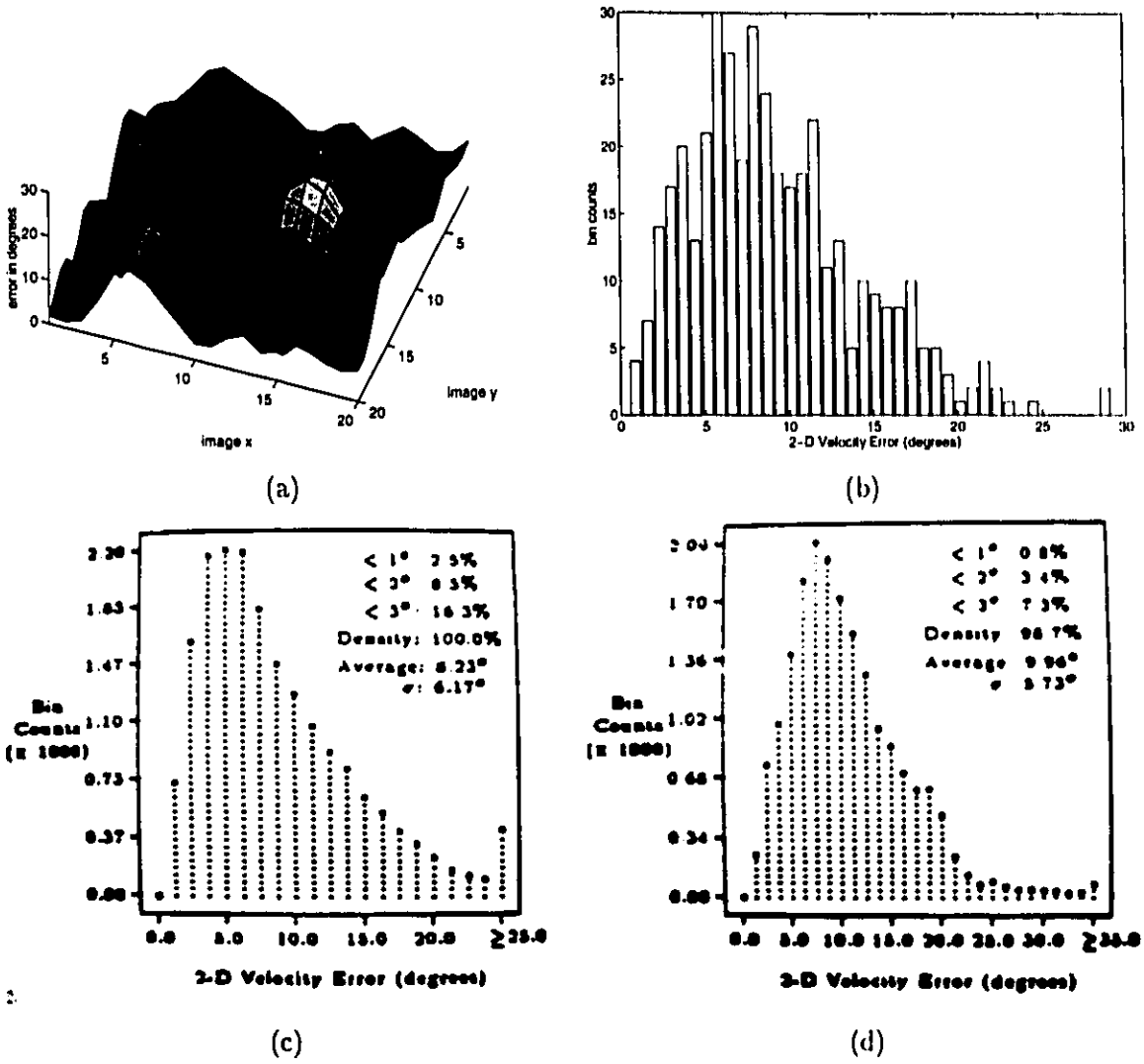


FIGURE 4.19. **Diverging Tree Sequence, error surface and histograms.** An angular error surface for frame 10 (a). Also shown is the angular error distribution for the unthresholded (b) experiment. The error distributions from (c) Anandan (unthresholded) and (d) Singh ($\lambda_1 \geq 0.1$) were obtained from [4].

3.3. Rubic Cube Sequence The Rubic Cube sequence portrays the famous toy on a rotating platter, the sides of which have a pattern that can be used for position encoding. Our algorithm suffers in zones without appropriate-scale textures. Although one can argue that the black-and-white squares of the rubic cube constitute a very strong, regular texture, we must also note that the scale of this texture is very large in comparison to the rest of the image. While we employed 10×10 pixel region matching with a 64×64 grid and 4

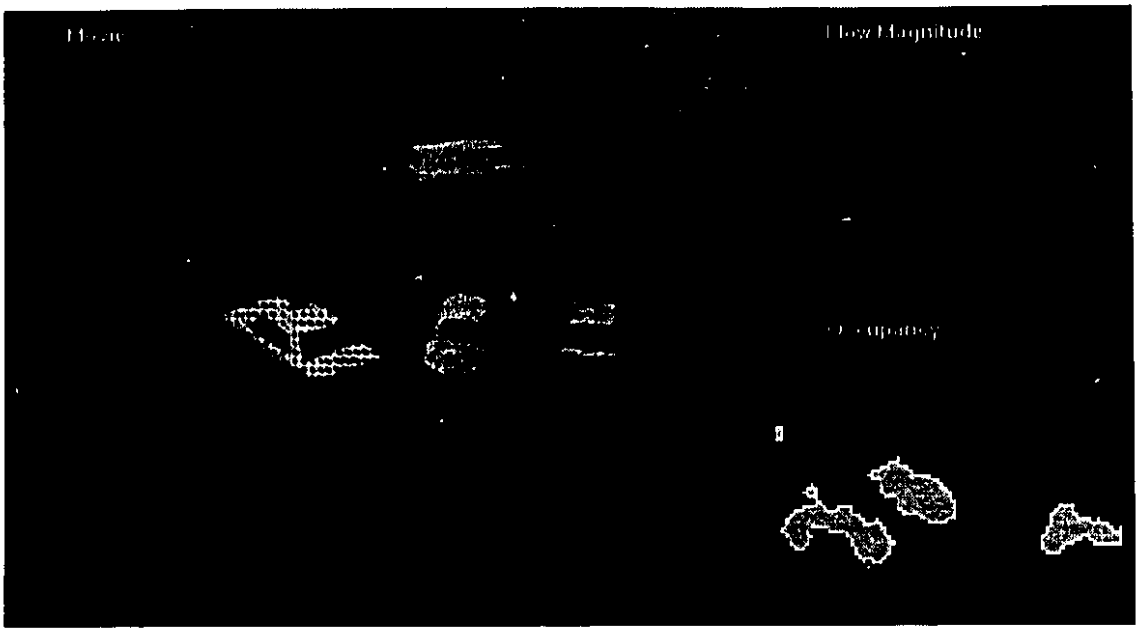
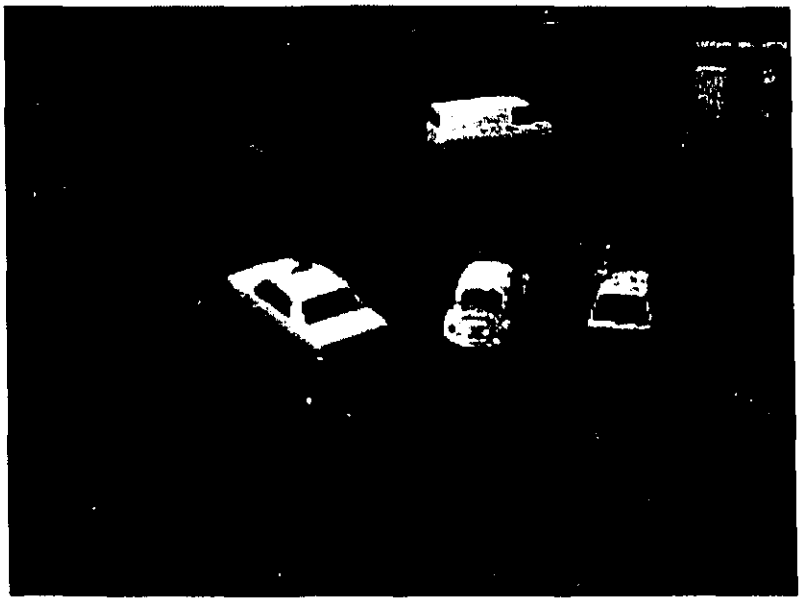


FIGURE 4.20. **Hamburg Taxi Sequence, workstation view.** The flow magnitude is rendered in the upper right window as intensity proportional to image velocity. In the lower right window, the figure/ground separation is rendered as a binarized image.

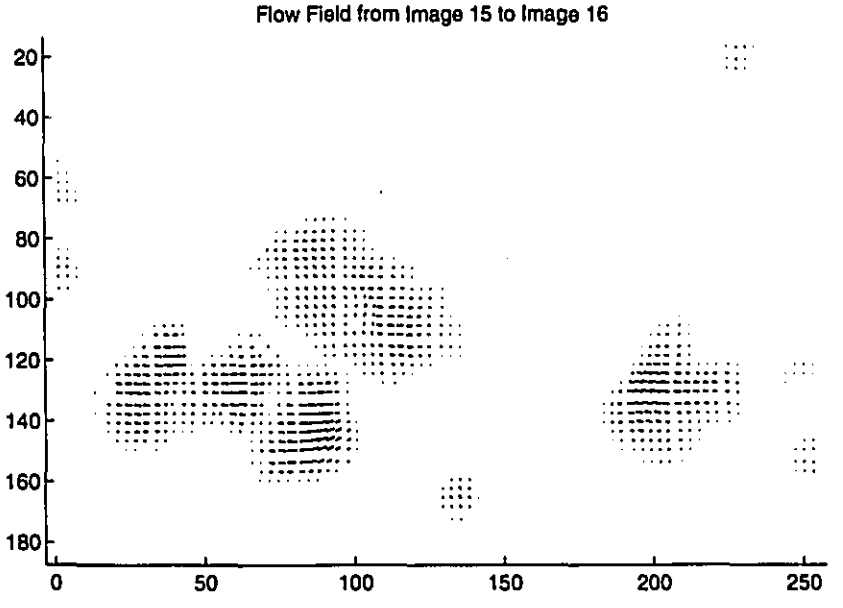
iterations, we can observe that motion is correctly found at and around the regions of higher texture, illustrated in Figure 4.26.

Qualitatively, our results on this image sequence rival those of Singh and Anandan, shown in Figure 4.27, which have many zones with apparently random flow vector orientations and magnitudes. One advantage to our algorithm is that large flow vectors will only result from actual large displacements: the flow field consistency stage would not allow neighboring vectors to behave so randomly unless there were sufficient low-level evidence for such displacements.

An additional remark is warranted by the lack of apparent motion on the top surface of the platter. By using higher-level knowledge of the scene, human observers have little difficulty perceiving the rotation of the entire platter when it is constrained by the coaxial rotation of the Rubic cube. Our algorithm does interpolate the motion between the platter rim and the Rubic cube, but does not completely connect the two motion regions because of the lack of local support for this connection through moving textures on the top platter surface. Our optical flow algorithm, without higher-level scene knowledge, will not fill-in the rest of the otherwise ambigous holes.



(a)



(b)

FIGURE 4.21. **Hamburg Taxi Sequence.** Frame 15 from the image sequence is shown in (a). The flow field between frames 15 and 16 is shown in (b).

3.4. NASA Coke Can Sequence Tha NASA Coke Can sequence shows the slow image expansion motion caused by the camera moving along its line of sight toward the Coke can near the center of the image. The typical image velocities are below 1 pixel/frame.

While we would expect most flow vectors to radiate away from the center of the image, only one fifth of the vectors in the field do this, as illustrated in Figure 4.28. Note that the

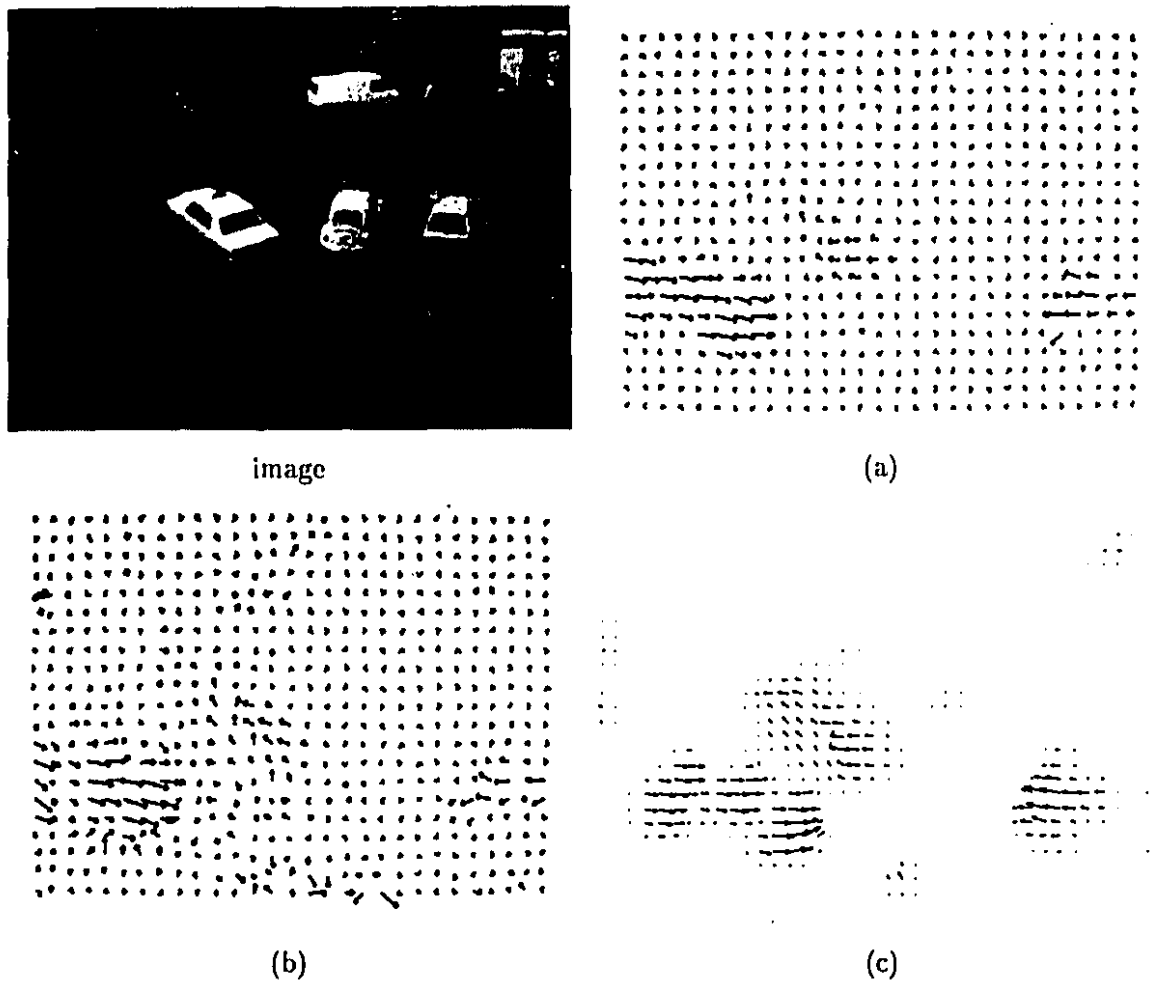


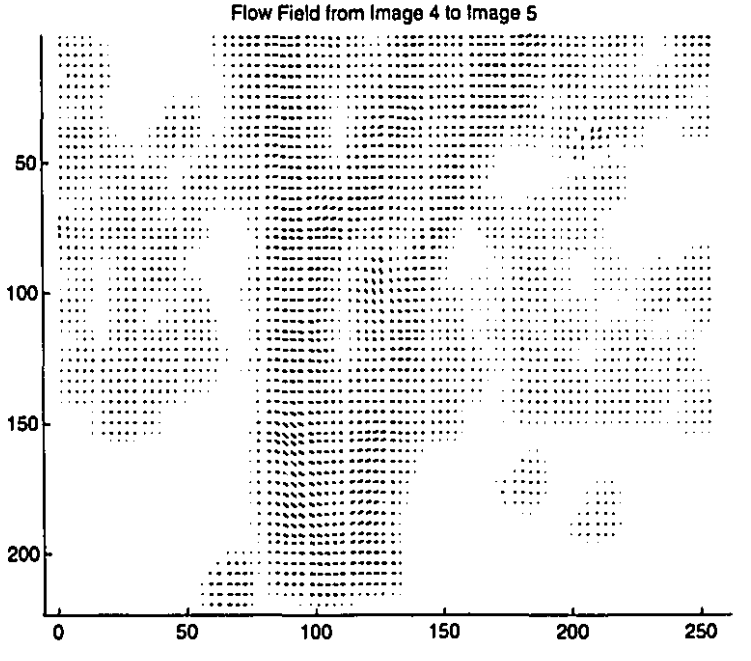
FIGURE 4.22. **Hamburg Taxi Sequence, other algorithms.** The flow field results by Anandan's algorithm are shown in (a). The flow field by Singh is shown in (b). Both of these diagrams appear in Barron et al. [5]. Our results were resampled and are shown in a similar format in (c).

regions where no motion was percieved (white space in the flow field) correspond to image patches with very little texture as well as image intensities near the edges of the sensor's dynamic range (i.e.: near intensity level 0 or 255). Our algorithm treats these limits of the dynamic range as noisy and unreliable, acreditting very little weight to these regions.

This image sequence did not cause our algorithm the instabilities seen in the results of Singh or Anandan, shown in Figure 4.29. We concede, however, that the flow vectors generated are not as impressive as the previously shown image sequences.

Again, the results of Singh and Anandan exhibit instabilities evidenced by the apparently random flow vector orientations and magnitudes, which our flow consistency stage

(a)



(b)

FIGURE 4.23. **SRI Tree Sequence.** Frame 5 from the image sequence is shown in (a). The flow field between frames 4 and 5 is shown in (b).

disallows. The strength of our algorithm is apparent when applied to these difficult scenes: flow vectors in textured patches are acceptable to good, but never unexpectedly random in regions of little information. The thrust of flow field consistency is that neighborhood

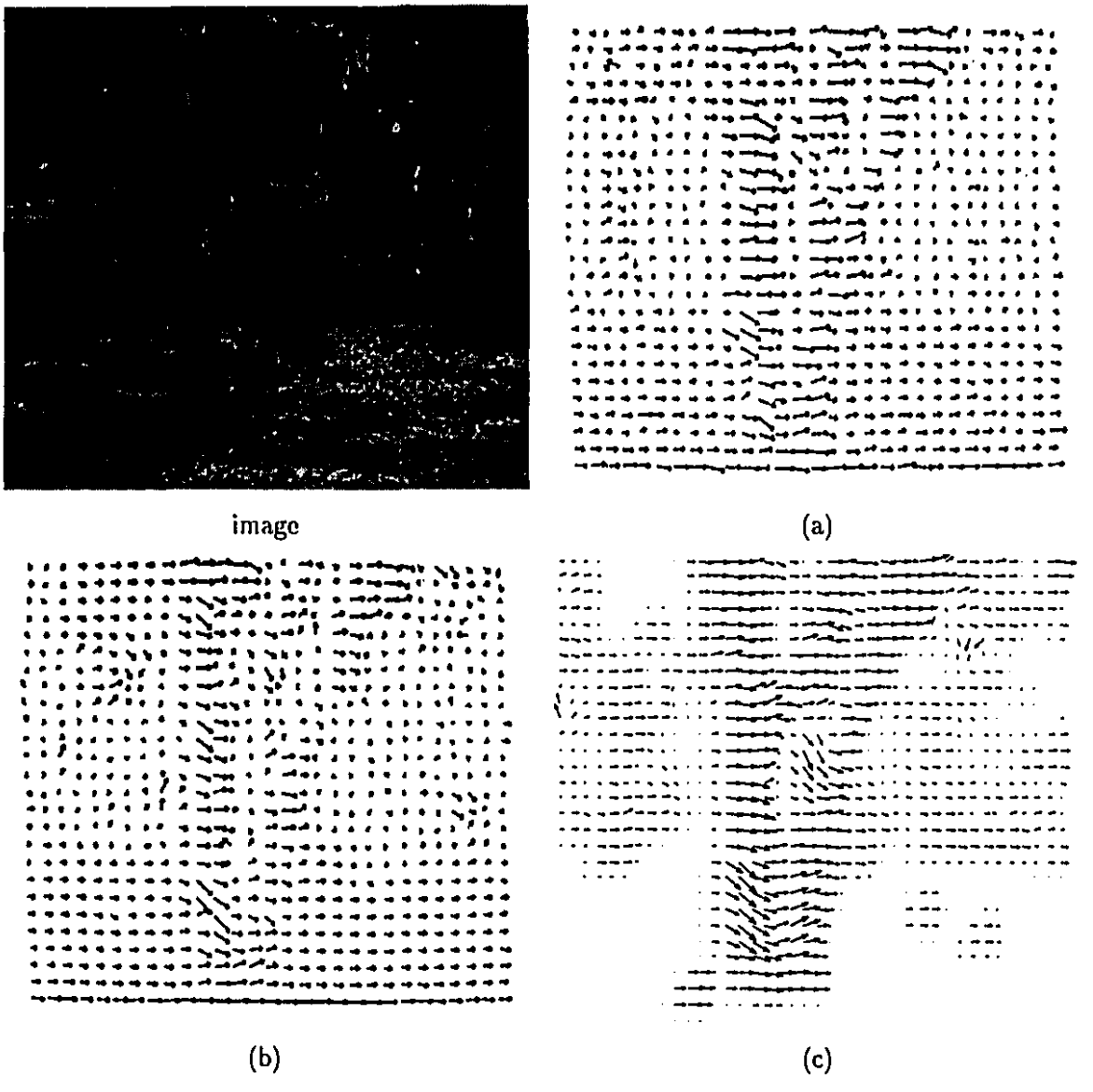


FIGURE 4.24. **SRI Tree Sequence, other algorithms.** The flow field results by Anandan's algorithm are shown in (a). The flow field by Singh is shown in (b). Both of these diagrams appear in Barron et al. [5]. Our results were resampled and are shown in a similar format in (c).

behaviour dominates in regions of greater uncertainty, producing a flow field that can be gently extrapolated from more confident neighbors.

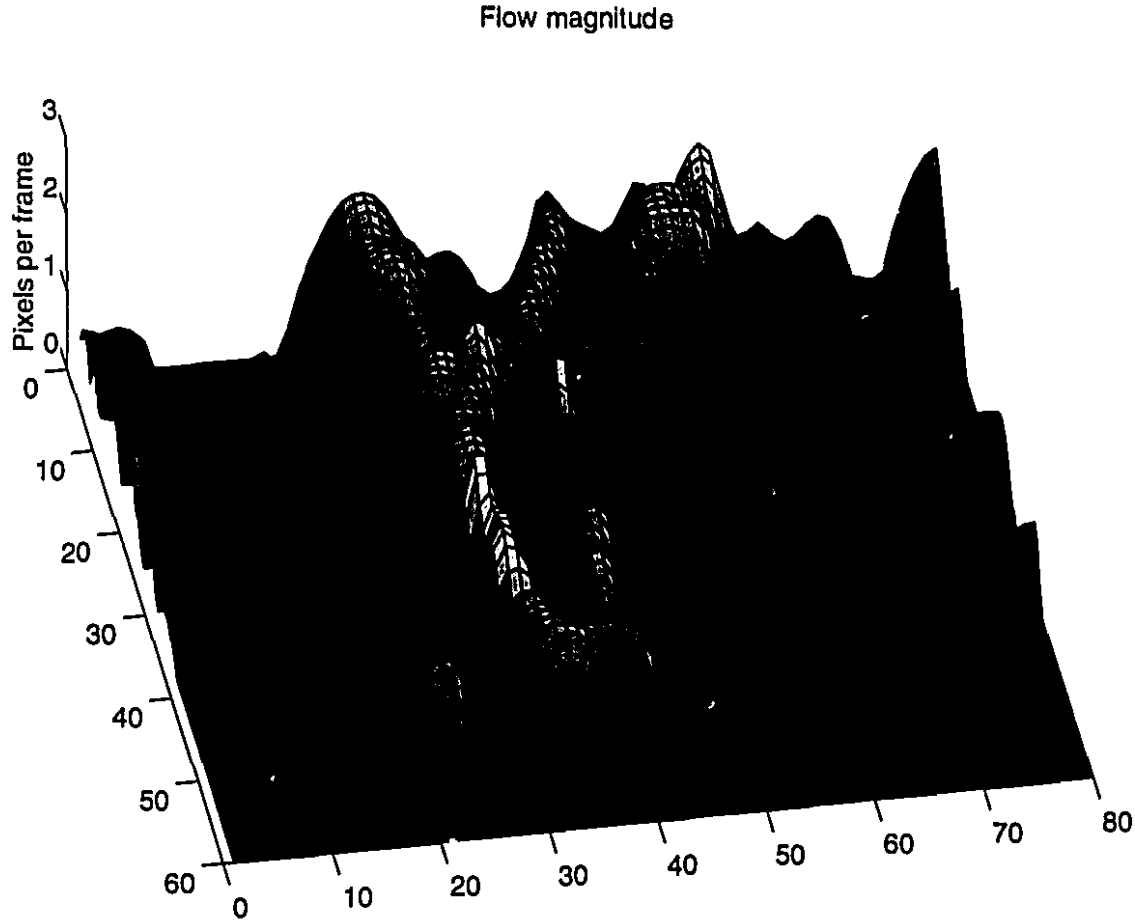
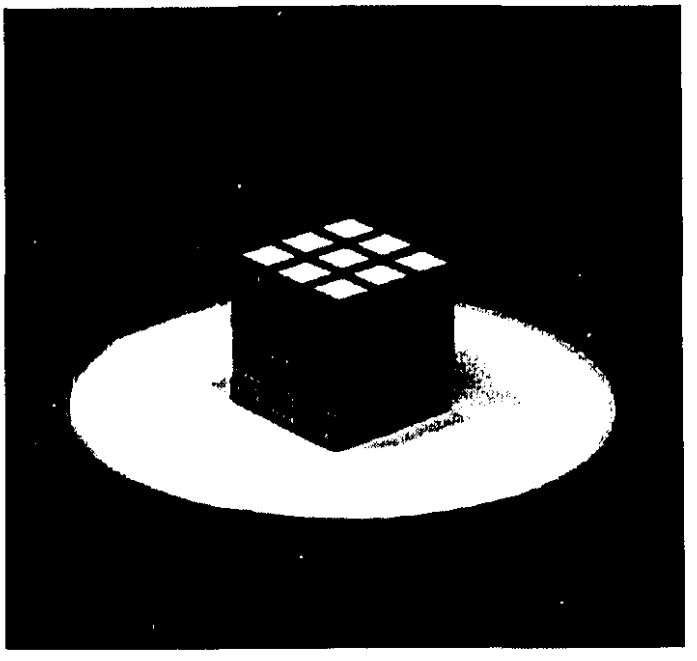


FIGURE 4.25. **SRI Tree Sequence, Kinetic Depth.** The magnitude of the flow field is rendered here as a relief map.

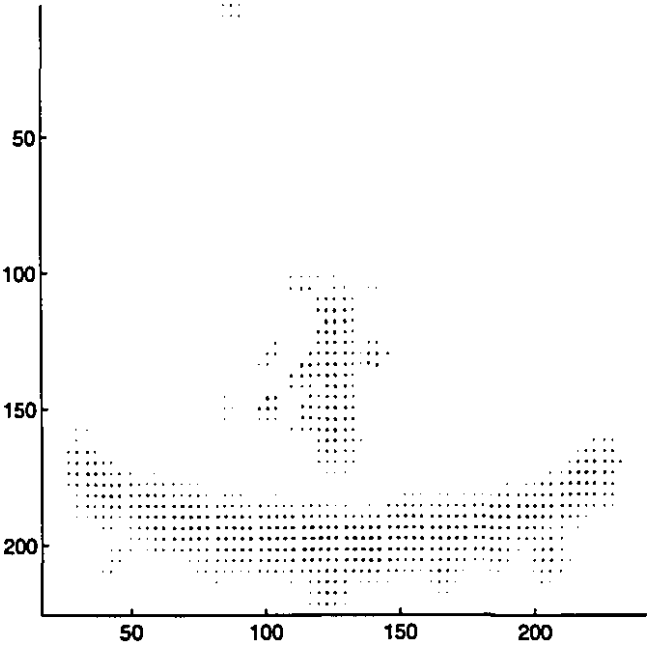
4. Image Plane Rotation

Not all optical flow algorithms can deal with image plane rotations, particularly when the rotating object occupies the bulk of the image and the angular velocity is large. Hierarchical coarse-to-fine or scale-space algorithms use coarse scale displacement measurements to seed the finer scales. An image plane rotation is poorly predicted from the coarser scales because the diffusion averages out the opposing flow vectors to zero: instead of predicting a rotation, it would predict no motion at all. This difficulty is not found in the flow-field consistency architecture of our algorithm. In Figure 4.30, we demonstrate the algorithm's success when dealing with image-plane rotation. A textured cube is rotated counter-clockwise by hand.

Note that the pixel displacements in this image range as high as 4 pixels per frame.



(a)



(b)

FIGURE 4.26. **Rubic Cube Sequence.** Frame 1 from the image sequence is shown in (a). The flow field between frames 2 and 3 is shown in (b).

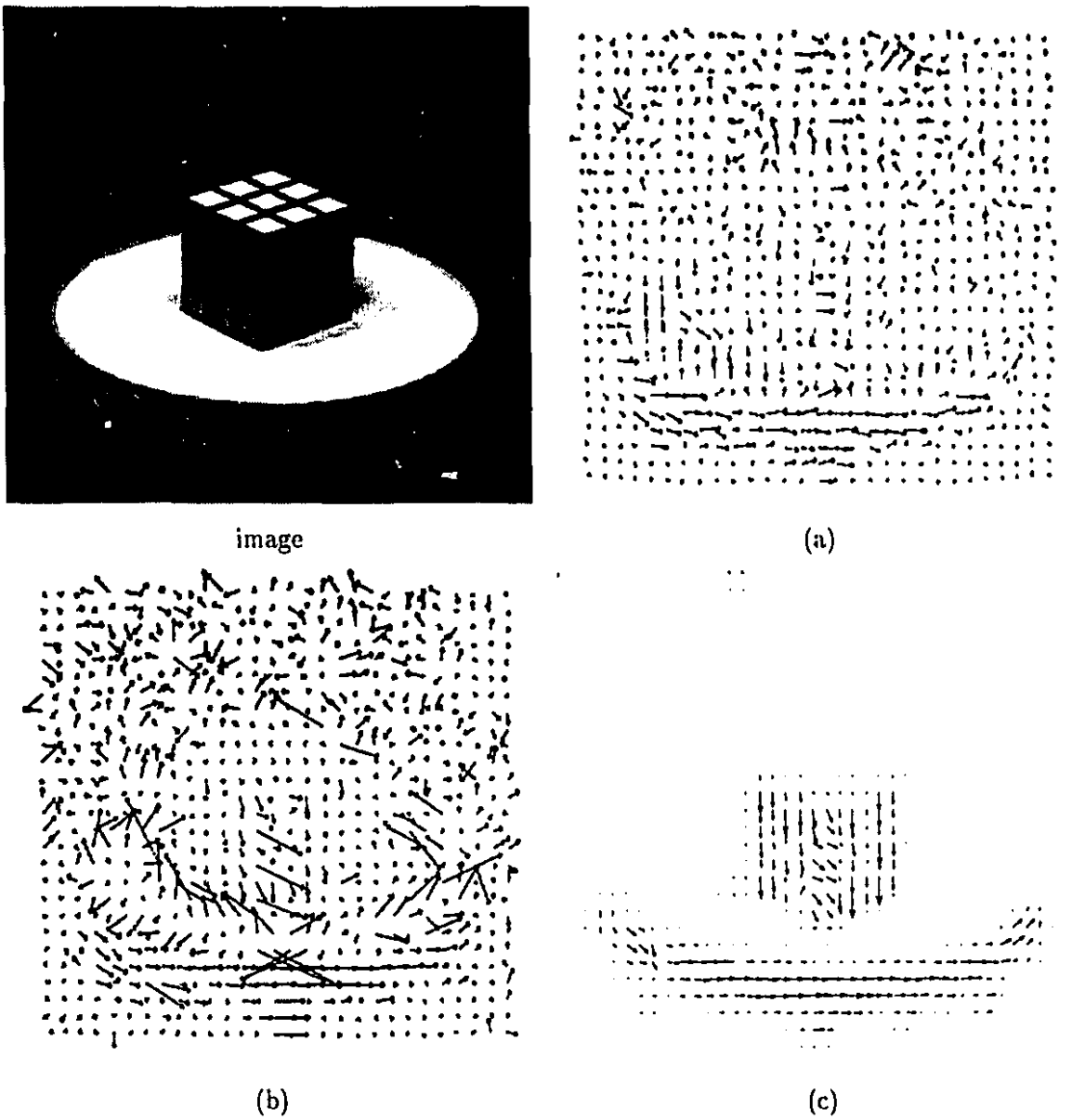
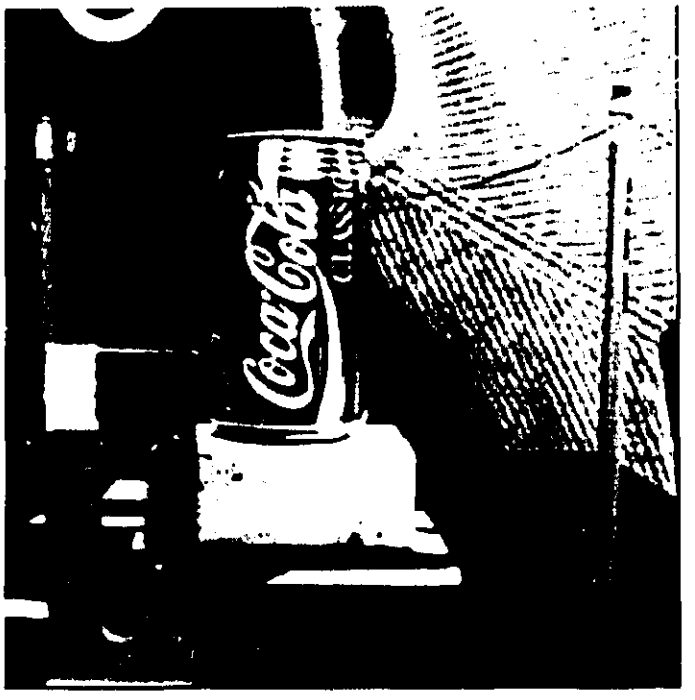


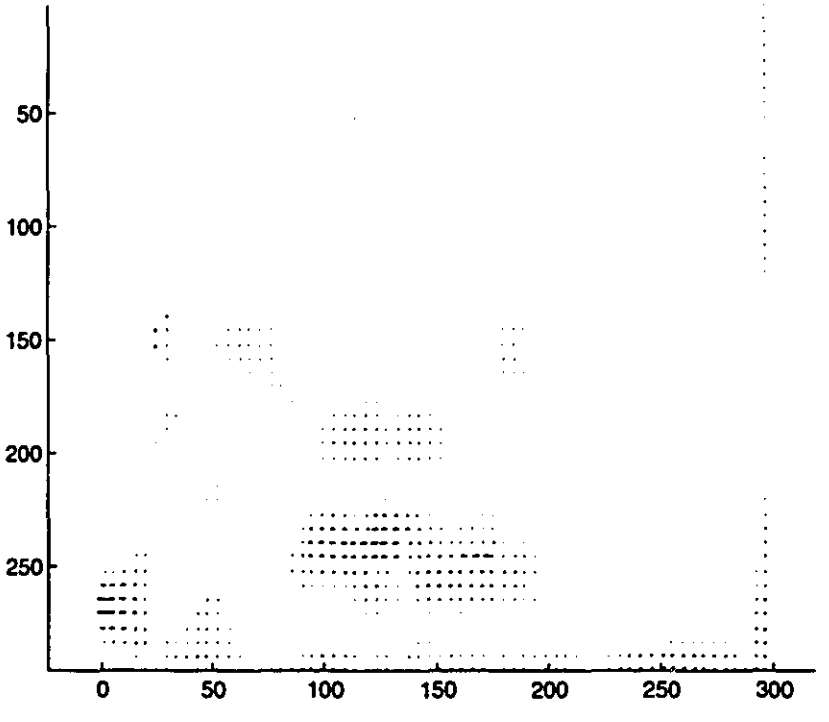
FIGURE 4.27. **Rubic Cube Sequence, other algorithms.** The flow field results by Anandan's algorithm are shown in (a). The flow field by Singh is shown in (b). Both of these diagrams appear in Barron et al. [5]. Our results were resampled and are shown in a similar format in (c).

5. Laboratory Sequences

This scene is typical of the events we wish to measure. An end user presents a target object to the workstation's video camera and moves the object while viewing the result on-screen in real-time. The rich flow field (see Figure 4.31) will be used in later stages



(a)



(b)

FIGURE 4.28. **NASA Coke Can Sequence.** Frame 4 from the image sequence is shown in **(a)**. The flow field between frames 2 and 3 is shown in **(b)**.

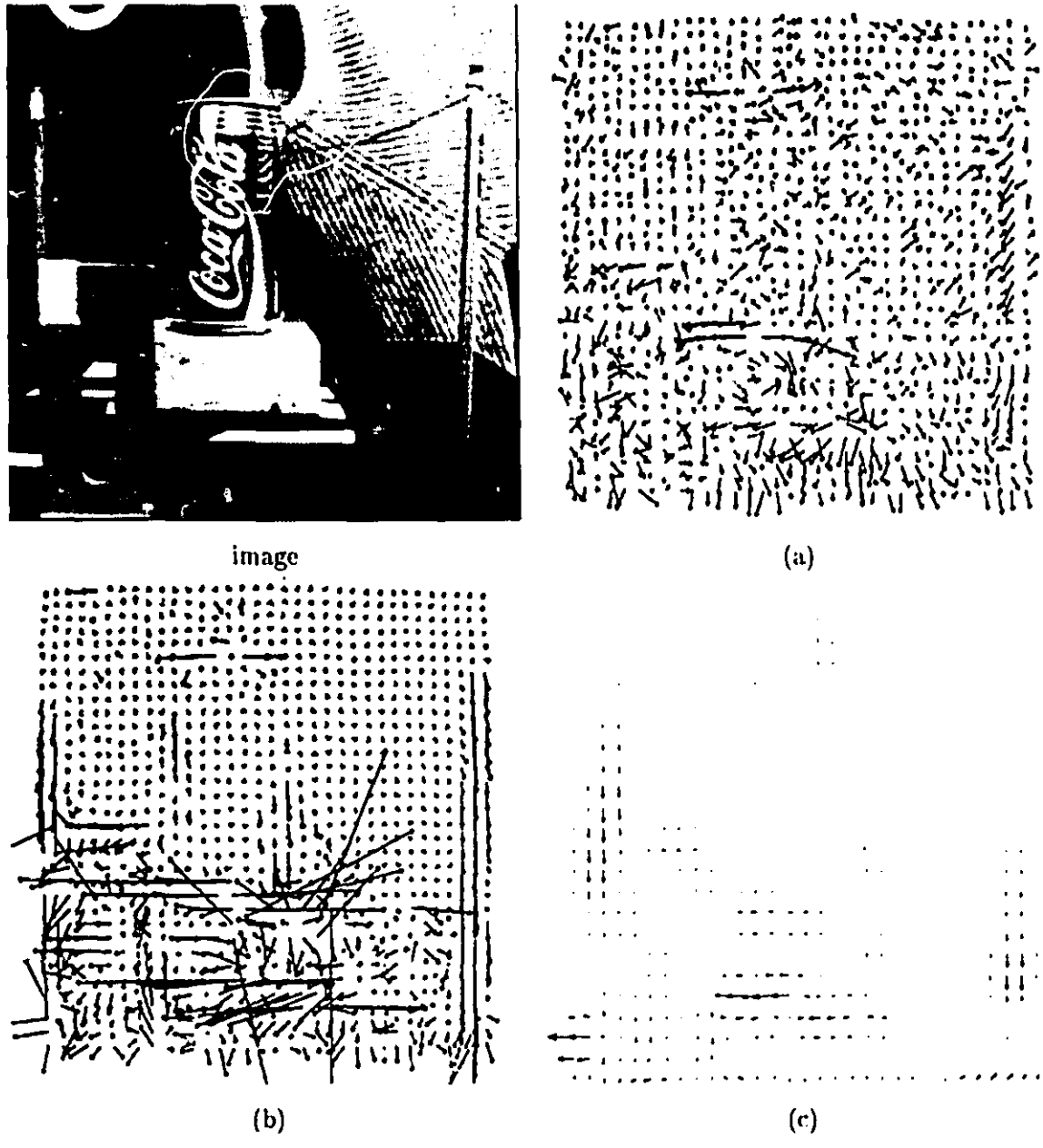
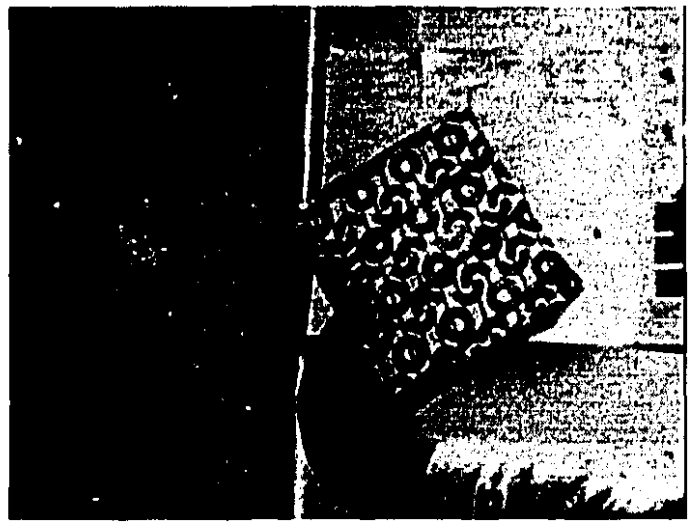
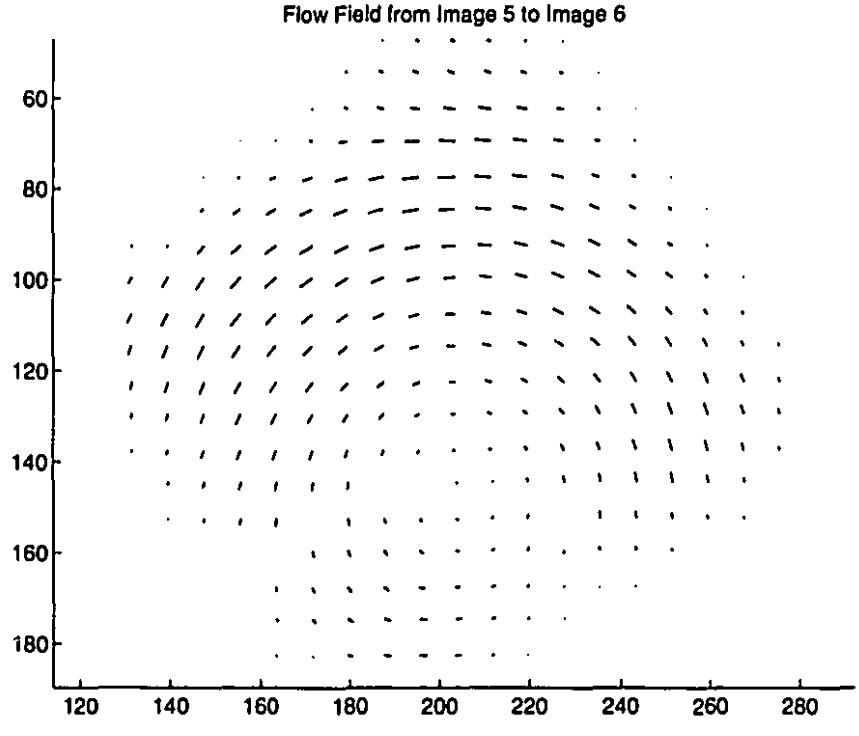


FIGURE 4.29. **NASA Coke Can Sequence, other algorithms.** The flow field results by Anandan's algorithm are shown in (a). The flow field by Singh is shown in (b). Both of these diagrams appear in Barron et al. [5]. Our results were resampled and are shown in a similar format in (c).

for qualitative shape description. This scene demonstrates the algorithm under its best conditions, i.e. low camera noise and high-contrast textures. As in all the above examples, only one iteration of the algorithm was applied to each image pair. Image velocities for this particular frame pair approached 5 pixels / frame, but velocities as high as 10 pixels / frame



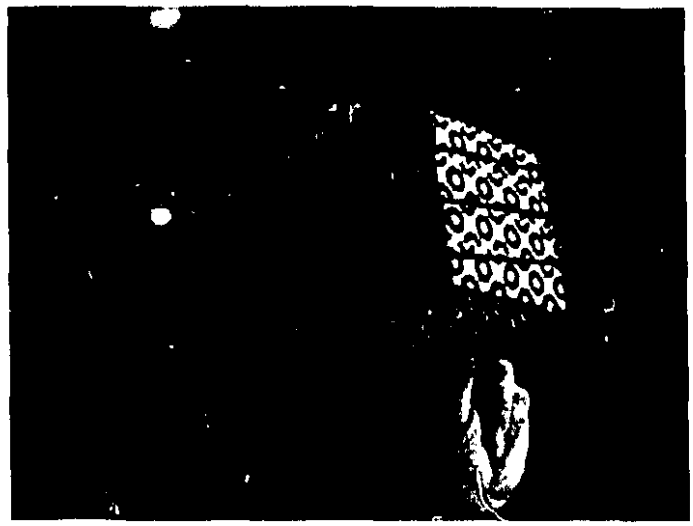
(a)



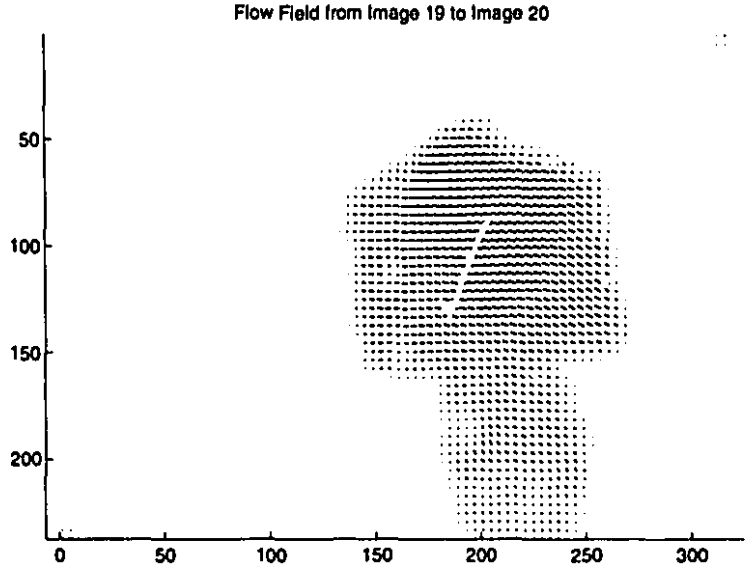
(b)

FIGURE 4.30. **Lab Cube rotation Sequence.** Frame 5 from the image sequence is shown in (a). A closeup of the flow field between frames 5 and 6 is shown in (b). All flow vectors not shown in this image were null.

have been successfully tracked. As before, a relief map is shown in Figure 4.32 to illustrate the crisp boundaries of the target object and coherency of the flow field magnitude.



(a)



(b)

FIGURE 4.31. **Hand-Held Target Sequence.** Frame 20 from the image sequence is shown in (a). The flow field between frames 19 and 20 is shown in (b).

6. Summary

The experimental results on synthetic image sequences (Section 1) and natural-appearance synthetic sequences (Section 2) are comparable to those of Anandan and Singh, despite the difference in computational cost. The natural scenes (Section 3) demonstrated our algorithm's tendency to avoid perceptually unlikely flow field configurations, whereas the hierarchical algorithms allowed runaway flow vectors to influence the surrounding flow field, producing misleading flow responses.

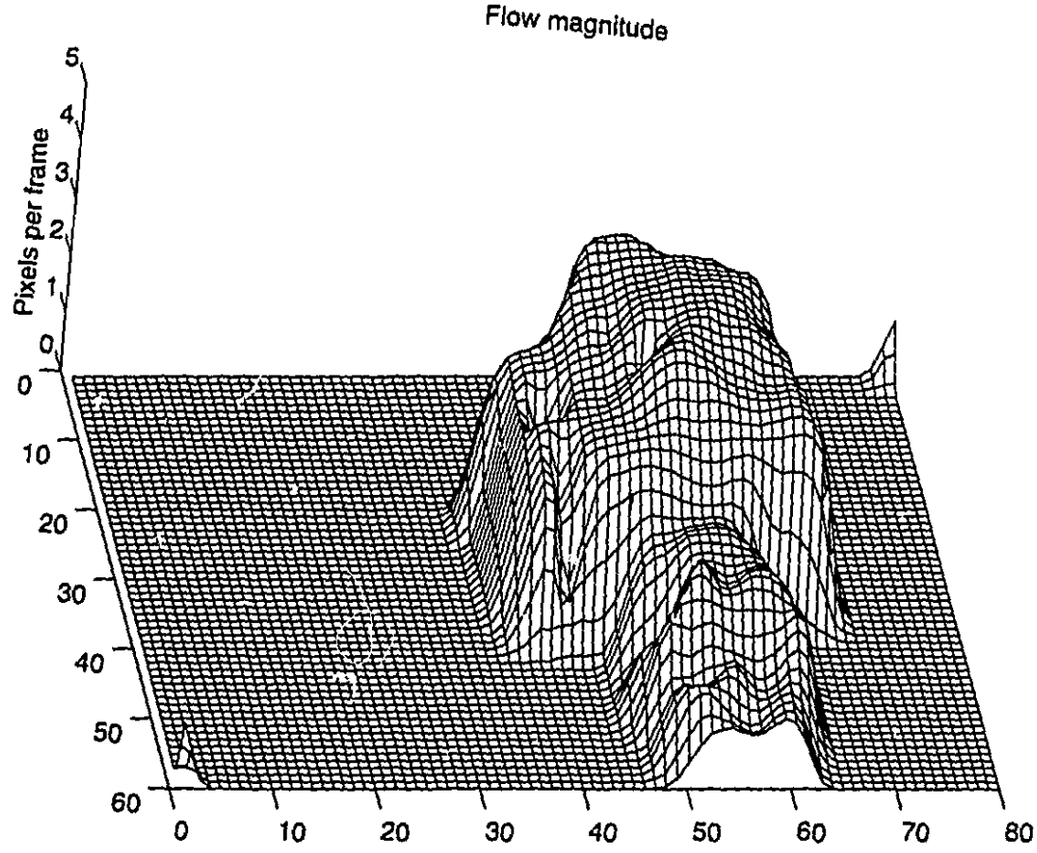


FIGURE 4.32. **Hand-Held Target Sequence, Kinetic Depth.** The magnitude of the flow field is rendered here as a relief map.

As described in Chapter 3, Section 2.2, the hierarchical algorithms diffusion stages prevent the perception of image-plane rotation, a percept that we consider as important as translation in the image plane and translation along the line of sight (illustrated in Chapter 2, Figure 2.1). We demonstrate experimentally how our algorithm's flow-consistency stage allows the perception of image plane rotation in Section 4.

Typically, natural image sequences will contain displacements of much more than one pixel per frame. We demonstrate how our algorithm can adapt to these wide ranges of image velocity, maintaining enough sensitivity to distinguish an operator's hand from the cube he is holding, as discussed in Section 5.

CHAPTER 5

Conclusion

From the results in Chapter 4, we have demonstrated the strengths of our algorithm on a variety of image sequences with results that are consistently as good as or better than other region-matching based optical flow algorithms. Furthermore, the results are obtained at near real-time frame rates. The experiments were performed on a 100 MHz R4600 SGI workstation at about 4 frames per second, but within a few short years, these same experiments could be performed on platforms at real-time frame rates, of the order of 15 frames per second.

As evidenced by both the synthetic and natural image sequences, our algorithm consistently finds optical flow fields that are close to what human observers would perceive. On synthetic data sets, we obtain quantitatively competitive results; on natural scenes, the results are qualitatively superior to other algorithms of the same class, and many algorithms of any class.

1. Unique Contributions

The algorithm presented here embodies principles of optical flow measurement, noise reduction and flow consistency believed to be present in the Primate brain. By following this model, region-based (pixel pattern) matching performs rapid searches, while a non-linear diffusion process enforces flow field consistency. This is really a compact decomposition of a coordinated gradient descent applied to many regions in parallel. The architecture employed is simple and executes rapidly on general-purpose workstations. Useful solutions usually emerge after only one iteration. Just as significantly, the algorithm is designed to be steered, or tuned, by higher-level information. The adaptability, quality, convergence and speed of this algorithm make it stand out as an easily-integrated, multi-purpose tool.

2. Importance

To our knowledge, no other algorithm is as efficient or effective at making use of measurement predictions. Our results demonstrate the advantages of flow field consistency constraints in region-based optical flow computation. Based on a biological model, this algorithm makes use of the rich geometrical information available from the estimated flow field at each iteration. Real-time optical flow is now achievable, and higher-level information can tune or direct the attentions of the low-level measurement process.

3. Relevance

The results obtained using our algorithm suggest that the biologically-motivated strategy of interleaving scalar region correspondence with flow field consistency operations leads to a stable inference of optical flow field that can serve as a stable basis for further interpretation. Performance is comparable to the best algorithms in terms of both quantitative and qualitative performance with the additional advantages of speed and adaptability. The algorithm is also flexible - large displacements are tracked as easily as sub-pixel displacements, and high-level information can feed flow field predictions into the measurement process (e.g. Kalman filtering).

4. Future Work

Our group does not consider optical flow computation a goal in itself. Our original intent was to explore the acquisition of three dimensional surface information from moving objects using a video camera signal as input. The advent of low-cost sensors coupled with high-performance computing power has rekindled the interest in both the determination of the optical flow field and its interpretation in tems of scene structure.

The context of our future work is the characterization of three-dimensional shape given prior knowledge in the form of a parametric model. In this scenario an operator presents a target object to a video camera and moves it according to the computer's suggestions for new view points (using a strategy derived from the autonomous exploration paradigm of our group). Our goal is to correctly recover the 3-D motion and structure of the object from the resulting flow and to minimize the ambiguity of this interpretation using constraints derived from the structure of the model and feedback provided to the operator.

The problem of fitting optical flow fields to 3-D parametric objects is still an open field of study, but would be of great benefit in closing the loop for higher-level information in our optical flow algorithm. Once the approximate 3-D shape and position of the object

is known, its motion can be tracked and projected back into the camera's image plane, predicting the next optical flow field. This ensures top-down feedback, where the sensor is directed, or tuned, by higher-level information of scene motion.

The top-down feedback would be akin to embedding Gestalt constraints far stronger than the local consistency constraints at the sensing level. This feedback is compact and subtle, is present in human perception, and would be a powerful instrument for machine perception.

REFERENCES

[1] J. Allman and S. Zucker. Cytochrome oxidase and functional coding in primate striate cortex: An hypothesis. *Cold Spring Harbor Symp. Quant. Biology*, 55:979-982, 1990.

[2] P. Anandan. *Measuring Visual Motion from Image Sequences.* PhD thesis, Univ. of Massachusetts, Amherst, MA, 1987. COINS TR 87-21.

[3] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, (2):283-310, 1989.

[4] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. Technical report, Robotics and Perception Laboratory, Department of Computing and Information Science, Queen's University, Kingston, Ontario, July 1992.

[5] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43-77, February 1994.

[6] H. Barrow and J. Tenenbaum. Interpreting line drawings as three-dimensional surfaces. *Artificial Intelligence*, 17:75-116, 1981.

[7] A. Blake and A. Zisserman. *Visual Reconstruction.* Cambridge: The MIT Press, 1987.

[8] T. J. Broida and R. Chellappa. Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):90-99, January 1986.

[9] S. Coren and J. Girgus. *Seeing is Deceiving: The Psychology of Visual Illusions.* Lawrence Erlbaum Associates, Hillsdale, N.J., 1978.

[10] D. Fleet. *Measurement of Image Velocity.* Kluwer Academic Publishers, Norwell, MA, 1992.

[11] D. Fleet and A. Jepson. Computation of component image velocity from local phase information. *International Journal of Computer Vision*, 5:77–104, 1990.

[12] A. Giachetti and V. Torre. Optical flow and deformable objects. In *Proceedings of ICCV*, pages 706–711. IEEE, 1995.

[13] R. Hartley. In defence of the 8-point algorithm. In *Proceedings of ICCV*, pages 1064–1070. IEEE, 1995.

[14] J. Heel. Temporally integrated surface reconstruction. In *Proc 3 Int Conf Comput Vision*, pages 292–295, Piscataway, NJ, USA, 1990. IEEE, IEEE Service Center.

[15] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[16] R. Hummel and S. Zucker. On the foundations of relaxation labelling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5:267–187, 1983.

[17] L. Kontsevich. Pairwise comparison technique: a simple solution for depth reconstruction. *Opt. Soc. Am. A*, 10(6):1129–1135, June 1993.

[18] M. Levine. *Vision in Man and Machine*, chapter 6, page 574. McGraw-Hill Book Company, 1985.

[19] H. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981.

[20] D. Marr. Early processing of visual information. *phil. trans. r. soc. lond*, 275:483–519, Oct. 1976.

[21] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 194:283–287, 5 October 1976.

[22] H. Nagel. Representation of moving rigid objects based on visual observations. *Computer*, pages 29–39, August 1981.

[23] P. Parent and S. Zucker. Curvature consistency and curve detection. *J. Opt. Soc. Amer., Ser. A*, 2(13), 1985.

[24] P. Parent and S. Zucker. Trace inference, curvature consistency, and curve detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8):823–839, August 1989.

[25] K. Prazdny. Egomotion and relative depth map from optical flow. *Biological Cybernetics*, 36:87–102, 1980.

[26] I. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(1):56–73, January 1987.

[27] A. Shashua. Projective structure from two uncalibrated images: Structure from motion and recognition. Technical Report A.I. Memo No. 1363, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1992.

[28] A. Singh. An estimation-theoretic framework for image-flow computation. In *Proceedings of ICCV*, pages 168–177. IEEE, 1990.

[29] A. Singh. *Optic flow computation: a unified perspective*. IEEE Computer Society Press, 1992.

[30] P. Singh, A. amd Allen. Image -flow computation: An estimation-theoretic framework and a unified perspective. *CVGIP: Image Understanding*, 56:152–177, 1992.

[31] R. Tsai and W. Huang, T. amd Zhu. Estimating three-dimensional motion parameters of a rigid planar patch, ii: Singular value decomposition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-30(4):525–534, August 1982.

[32] G. Tseng and A. Sood. Analysis of long image sequence for structure and motion estimation. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1511–1526, November/December 1989.

[33] S. Ullman. Analysis of visual motion by biological and computer systems. *Computer*, 14(8):57–69, 1981.

[34] Viéville and O. Faugeras. Motion analysis with a camera with unknown, and possibly varying intrinsic parameters. In *Proceedings of ICCV*, pages 750–756. IEEE, 1995.

[35] J. Webb and J. Aggarwal. Structure from motion of rigid and jointed objects. *Artificial Intelligence*, 19:107–130, 1982.

[36] J. Weber and J. Malik. Rigid body segmentation and shape description from dense optical flow under weak perspective. In *Proceedings of ICCV*, pages 251–256. IEEE, 1995.

[37] J. Weng and N. Huang, T.S. amd Ahuja. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(5):451–479, May 1989.

[38] H. Yeshurun. Size limits on stereo and motion perception: Back to the hypercolumn? Lecture, October 1995.

[39] G. Young, R. Chellappa, and T. Wu. Monocular motion estimation using a long sequence of noisy iamges. In *Proceedings ICASSP*, volume 4, pages 2437–2440, Piscataway, NJ, USA, 1991. IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE.

[40] S. Zucker. Early orientation selection: Tangent fields and the dimensionality of their support. *Computer Vision, Graphics, and Image Processing*, 32:74–103, 1985.

[41] S. Zucker. *Early Vision*, chapter E, pages 394–419. John Wiley & Sons, Inc., second edition, 1992.