MONOCULAR OPTICAL FLOW FOR REAL-TIME VISION SYSTEMS

Stephen M. Benoit and Frank P. Ferrie

TR-CIM-96-09 October 1996

Artificial Perception Laboratory
Centre for Intelligent Machines
McGill University
Montréal, Québec, Canada

Submitted to *14th International Conf. on Pattern Recognition*, Feb. 1996

Postal Address: 3480 University Street, Montréal, Québec, Canada H3A 2A7
Telephone: (514) 398-6319 Telex: 05 268510 FAX: (514) 398-7348
Email: cim@cim.mcgill.ca

MONOCULAR OPTICAL FLOW FOR REAL-TIME VISION SYSTEMS

Stephen M. Benoit and Frank P. Ferrie

Abstract

This paper introduces a monocular optical flow algorithm that has been shown to perform well at nearly real-time frame rates (4 FPS on a 100 MHz SGI Indy workstation), using 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. 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.

Ce papier introduit un algorithme de flot optique monoculaire qui a été appliqué avec succès sur des scènes naturelles à des fréquences video presque temps réel. Le système utilise une approche de bas niveau s'apputant 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 est minimisée localement, et les contraintes de compatibilité des champs de flot appliquent une diffusion adaptive non-linéaire, permettant auxc regions dde grande compatibilité d'influencer leurs voisins. La convergence est habituellement atteinte á 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 concu avec un critère de flexibilité; les grands déplacements sont percus aussi facilement que ceux de moins d'un pixel, et les informations de plus haut niveau peuvent fournir une assistance à la procédure de mesure.

1. Introduction

In this paper 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 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. 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.

The paper begins in Section 2 with a brief outline of the related work. A short biological motivation for the algorithm and the algorithm itself are described in Section 3 along with details of our particular implementation. Experimental results presented in Section 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. The paper concludes in Section 5 with some final observations and pointers to future work.

2. Background

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 [7,8] maximize a similarity measure, such as minimizing the sum-of-squared differences (SSD),

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

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.

In this paper, we argue that the important information is not strictly a displacement 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.

One might hope that applying a Gaussian diffusion operator to a flow field would somehow spread the flow field information throughout the moving object, but in reality, this only blurs the actual flow field. In order to be effective, flow field consistency must be enforced during the measurement stage.

3. Optical Flow from Region-Matching and Flow Consistency

As suggested in section 2, 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 paper, we will consider the optical flow field to be a coarse field of image point correspondences between the two sequential images.

Our algorithm makes use of the suggested organization of scalar and geometric tasks in the hypercolumns of the primate visual cortex [1]. Image correspondences are performed by specialized clusters of scalar intensity-processing hypercolumns, but these hypercolumns are surrounded by orientation-detecting, or flow geometry hypercolumns. Neighboring elements in the geometric pathway are forced to adopt orientations that obey flow field consistency, and seem to control the diffusion of measurements from the scalar pathway. We suggest that the flow geometry, computed from the image correspondences, feeds back to steer the image correspondences. We also adhere to the observation of Yeshurun [10], that the useful output of the biological optical flow processing is very sparse compared to the input visual field density.

An overview of our algorithm is presented in the block diagram of Figure 1. 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 is used to predict the next set of region matching. In our laboratory environment, we use a Kalman prediction loop to provide top-down feedback, but for this paper, 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 the following sections.

3.1. **Comparing Computational Cost.**

Region matching, the fundamental measure used in this paper and the algorithms of Anandan and Singh, will have the

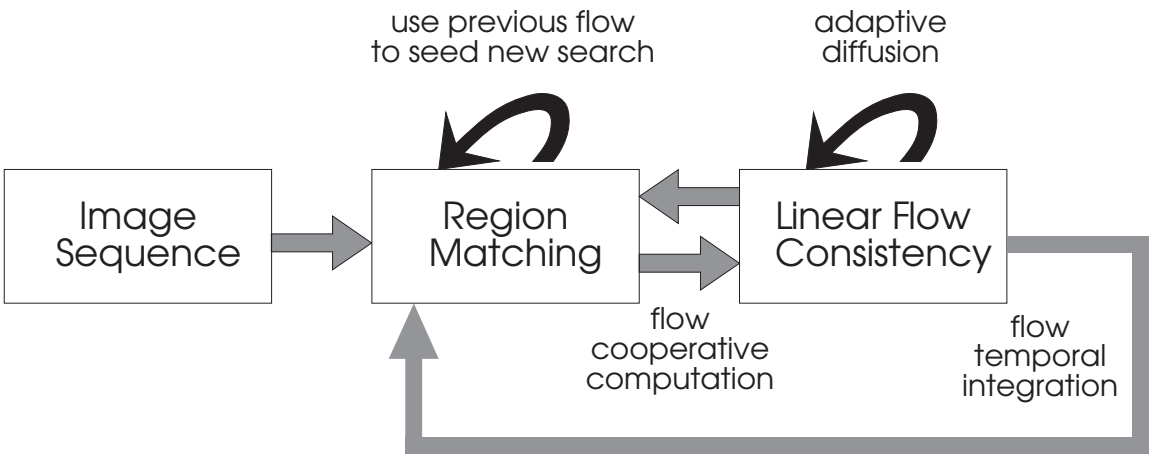


FIGURE 1. Block diagram of full optical flow process.

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,

$$C_{ours} = kN\,(18r + d)\,. \tag{2}$$

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

$$C_{Anandan}(i) = 2^i N\,(Rr + Dd)\,. \tag{3}$$

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

$$\begin{aligned} C_{Anandan} &= \sum_{i=1}^{log(N)} 2^i N\,(Rr + Dd) \\ &= \left(2^{log(N)+1} - 2^1\right) N\,(Rr + Dd) \\ &= 2\,(N-1)\,N\,(Rr + Dd) \\ &\approx 2N^2\,(Rr + Dd) \end{aligned} \tag{4, 5}$$

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.

3.2. **Region Matching.** 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 paper, 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

$$D_p(i,j) \triangleq \left\| P_{2p}(i,j) - P_{1p}(i,j) \right\| \tag{6}$$

$$S_p(i,j) \triangleq \left\| P_{2p}(i,j) + P_{1p}(i,j) \right\| \tag{7}$$

$$err_p(i,j) \triangleq \frac{D_p(i,j)}{S_p(i,j)} \tag{8}$$

$$err_p = \sum_{i,j} err_p(i,j) \tag{9}$$

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.

$$\text{for } S_p(i,j) < S_\theta, \text{ set } err_p(i,j) = err_{max} \tag{10}$$

$$\text{for } D_p(i,j) < D_\theta, \text{ set } err_p(i,j) = err_{min} \tag{11}$$

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. This type of noise also results from lossy image compression. These parameters were chosen to be $S_\theta = 16$, $D_\theta = 8$, 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 $err_{min} = 0.01$.

3.3. **Properties of Normalized error measure.** 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 2.

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, 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.

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. Noisy, low-intensity image patch error surfaces also look similar, as shown in Figure 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

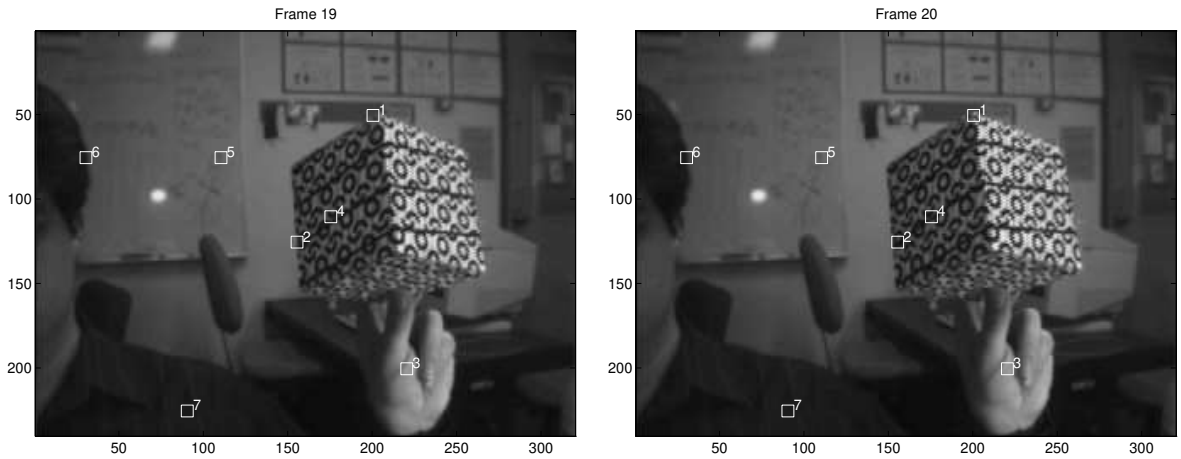


FIGURE 2. **Region Matching image frames.** Frame 19 of the hand-held moving cube **(a)**, and Frame 20 **(b)**. The numbered squares are the initial tile positions for region matching between the two images. The number corresponds to the region matching experiments, shown later.

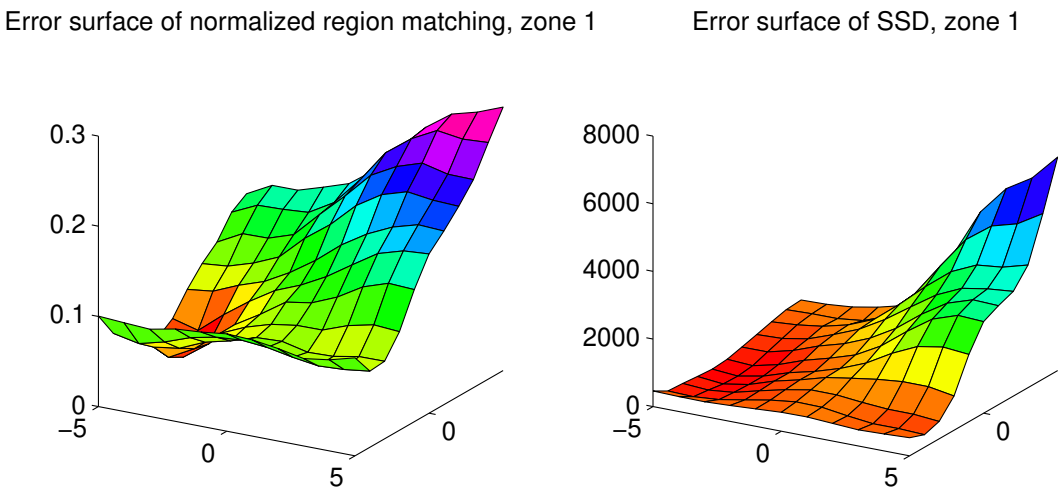


FIGURE 3. **Region Matching Zone 1.** The error surfaces generated using the normalized error measure versus the SSD error measure. The vertical axis is the error, the x and y axes are the pixel region shifting between the two frames to obtain the region match. This corresponds to a corner of the cube in the image sequence. Note the minima in the lower left of the two surfaces, where the true correspondence lies.

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.

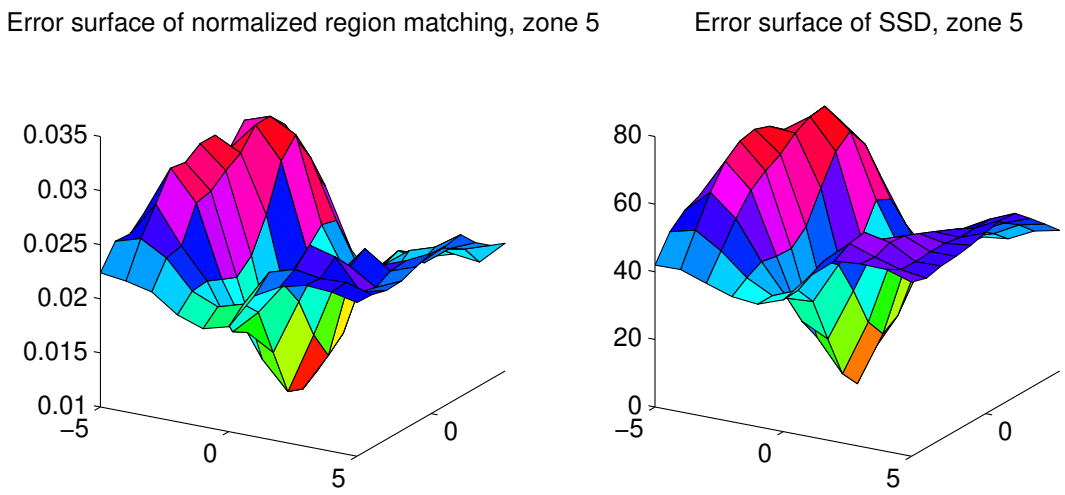


FIGURE 4. **Region Matching Zone 5.** The error surfaces generated using the normalized error measure versus the SSD error measure. The vertical axis is the error, the x and y axes are the pixel region shifting between the two frames to obtain the region match. This corresponds to the poorly-lit, our-of-focus whiteboard in the background in the image sequence. Note the SSD error surface does not tell us how ambiguous the overall matching is.

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 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. This type of noise would be present in the case of lossy image compression, where bandwidth restrictions limit the transmitted image quality.

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.

3.4. **Flow Field Consistency.** 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 [6]. 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.

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,

$$\vec{v}_p^{k+1} \leftarrow \frac{\sum_{n \in \mathbf{N}} w_n^k \vec{v}_n^k}{\sum_{n \in \mathbf{N}} w_n^k}, \tag{12}$$

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. In this situation, both the magnitude and direction similarity are considered equally important. The similarity measure is divided into two components, magnitude similarity $S_m(\vec{v}_1, \vec{v}_2)$ and direction similarity $S_d(\vec{v}_1, \vec{v}_2)$.

$$S_d(\vec{v}_1, \vec{v}_2) = \begin{cases} \frac{1+\vec{v}_1 \cdot \vec{v}_2}{2|\vec{v}_1||\vec{v}_2|} & \text{when } |\vec{v}_1||\vec{v}_2| \neq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

$$S_m(\vec{v}_1, \vec{v}_2) = \begin{cases} 1 - \frac{|\vec{v}_1 - \vec{v}_2|}{|\vec{v}_1| + |\vec{v}_2|} & \text{when } |\vec{v}_1| + |\vec{v}_2| \neq 0, \\ 1 & \text{otherwise.} \end{cases} \tag{14}$$

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

$$S(\vec{v}_1, \vec{v}_2) = \frac{1}{2} S_m(\vec{v}_1, \vec{v}_2) + \frac{1}{2} S_d(\vec{v}_1, \vec{v}_2). \tag{15}$$

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 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.

3.5. **Integrating Region Matching and Flow Field Consistency.** Singh and Allen proposed a novel framework to unify many contemporary optical flow algorithms that could take directional errors into account for later processing [9]. A key notion that is used in this paper is how velocity must be propagated from "regions of full information, such as corners, to regions of partial or no information." [9] 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 paper, 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.

3.6. **Implementation Issues.** Note that the region matching and flow field consistency constraints could have been implemented as one error function to minimize. It can be shown that by alternatively measuring the region-matching error and enforcing the flow field consistency, the end effect is to perform a coordinated gradient descent locally for each tile, while diffusing measurements to neighboring tiles.

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.

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.

4. Experimental Results

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

4.1. **Translating Sinusoids.** 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, $\vec{v} \equiv \frac{1}{\sqrt{v_x^2+v_y^2+1}}(v_x, v_y, 1)^T$, the error between the correct velocity $\vec{v}_c$ and an estimate $\vec{v}_e$ is

$$\psi_E = \arccos(\vec{v}_c \cdot \vec{v}_e) \tag{16}$$

The tests were performed on the *mysineB-6* (**Sinusoid 1**) data set, where the motions of the entire image plane are known to be (1.583, 0.863) pixels / frame. 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 1.

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

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

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.

4.2. **Yosemite Fly-Through 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]. A frame of the sequence and the measured flow field for our algorithm is shown with the results from Anandan and Singh in Figure 5. 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 error details for the experiment are shown in Figure 6. Note that our algorithm is competitive with the algorithms of Anandan and Singh.

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. The results for the thresholded and unthresholded experiments are summarized in table 2.

Technique	Valid Data	Average Error	Std. Dev.	$< 1°$ Error	$< 2°$ Error	$< 3°$ Error
Us Unthresholded	67.6%	17.16°	17.50°	1.96%	7.25%	13.35%
Us Threshold=0.025	3245	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 2. 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].

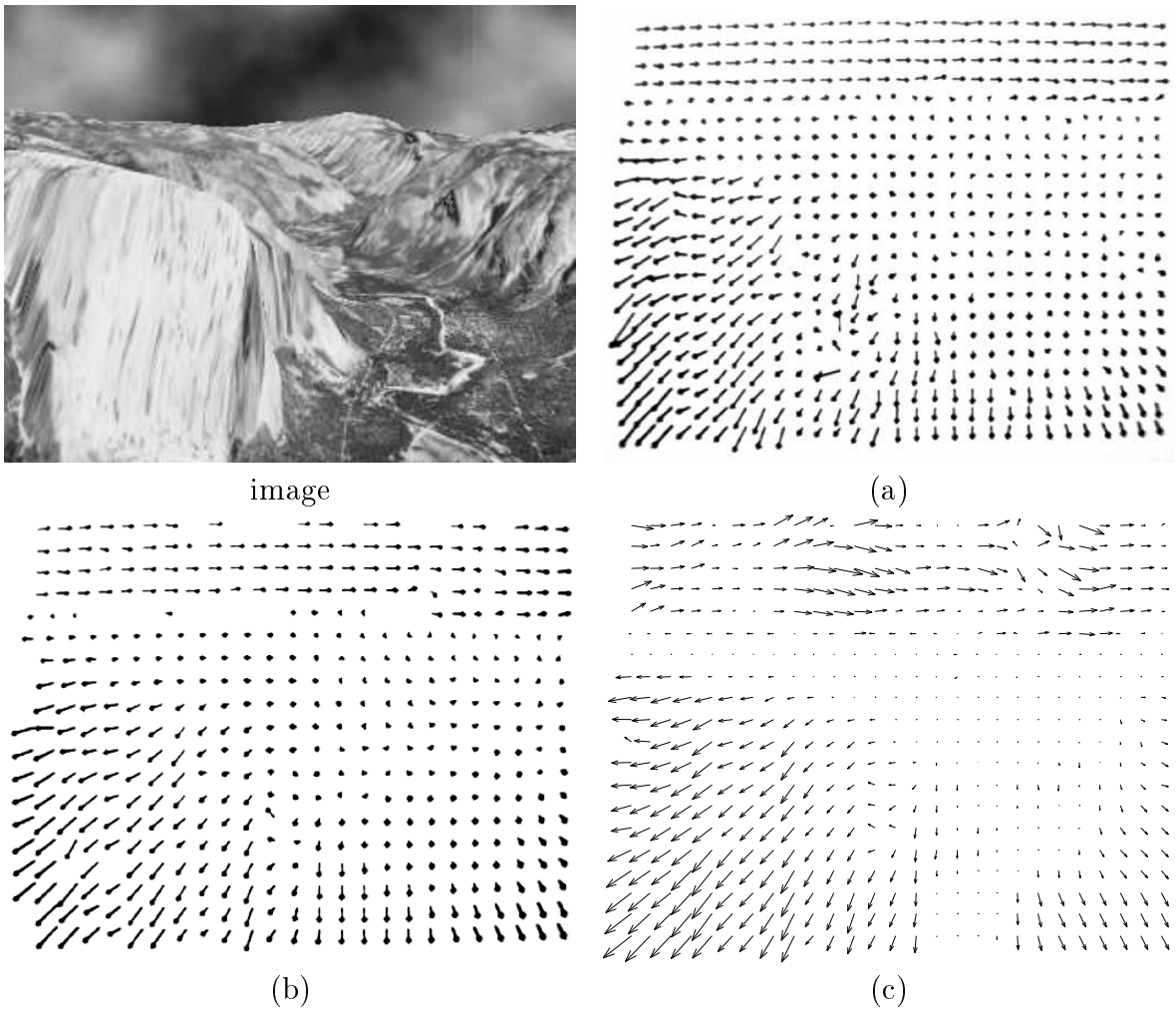


FIGURE 5. **Yosemite Sequence, Performance Comparison.** The flow field for Anandan's algorithm is shown in **(a)**. Singh's algorithm produced the flow field shown in **(b)**. Both plots were obtained from [4]. Our results were resampled and are shown in a similar format in **(c)**.

The correct flow field can be obtained from [5], but the measured flow field is shown in Figure 5. 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 2 as 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.

4.3. **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

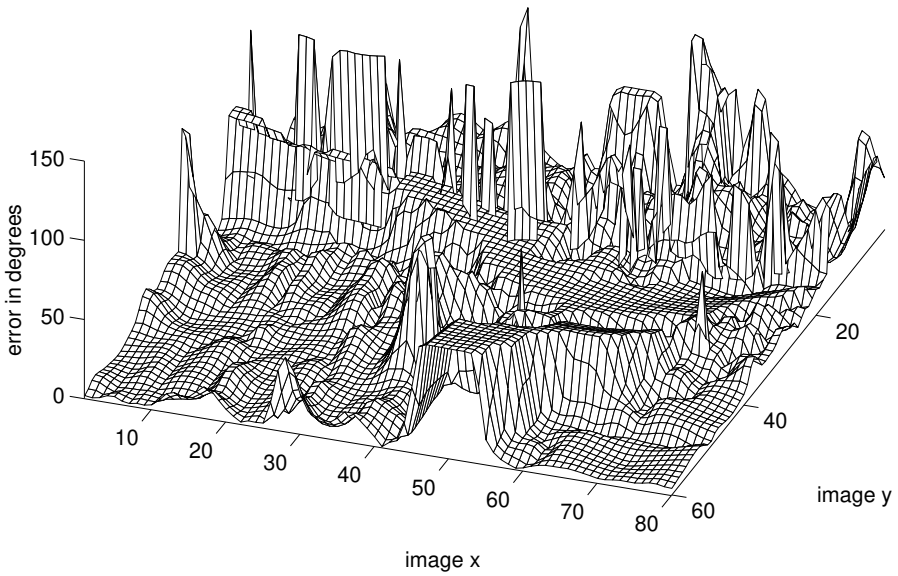


FIGURE 6. **Yosemite Sequence, error surface.** An angular error surface for frame 10.

error threshold for our algorithm to detect. The algorithm used only one iteration per image pair, with a grid of 80×60 tiles, each tile at 6×6 pixels. An image frame of the sequence is shown with the flow fields obtained from our method, and those of Anandan and Singh, in Figure 7.

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

4.4. **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 the highest image velocities were found to be just under 3 pixels per second. A sample frame of the image sequence and the measured flow field are shown in Figures 8, together with the results from the algorithms of Anandan and Singh. 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, also shown in Figure 9 (b).

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

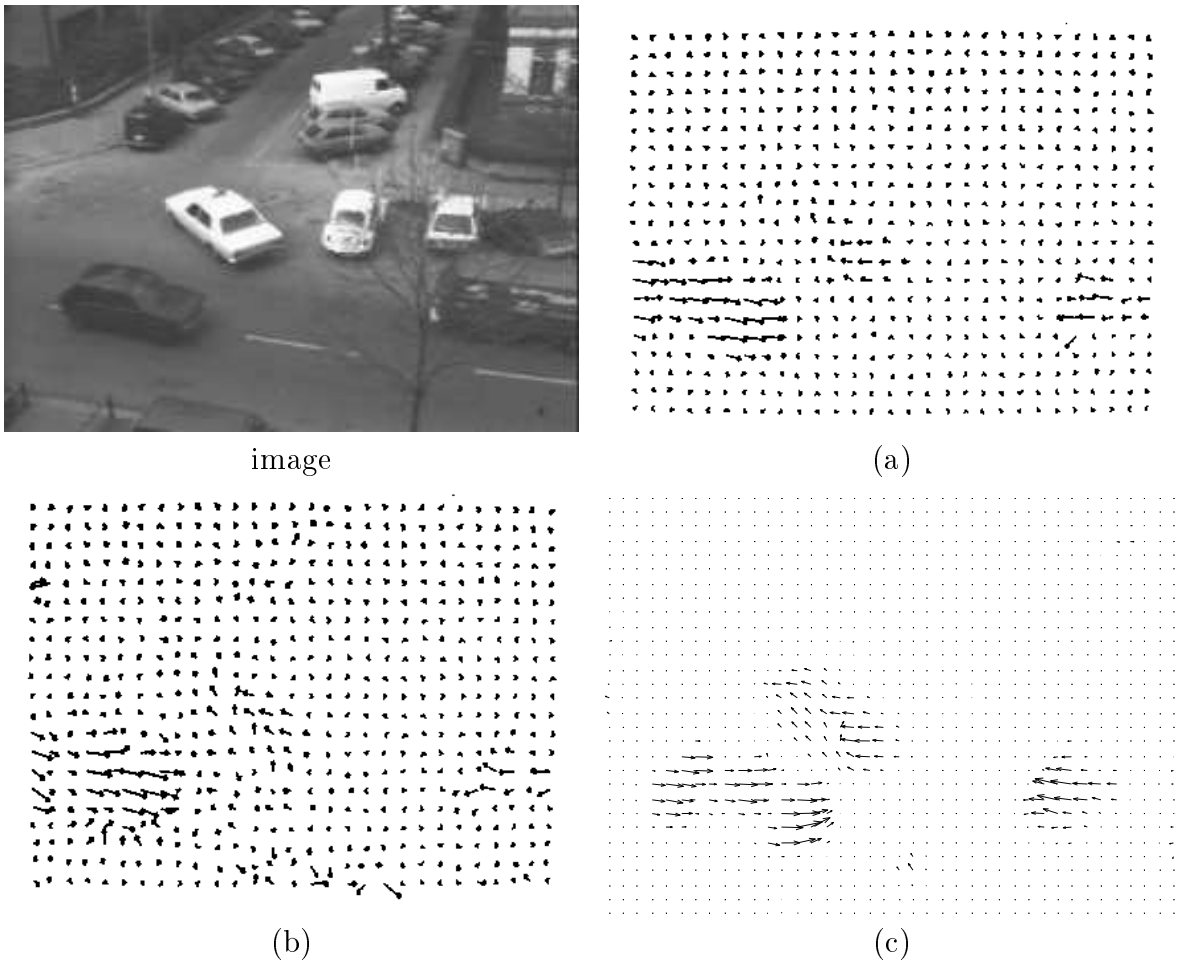


FIGURE 7. **Hamburg Taxi Sequence, other algorithms.** The flow field results by Anandan's algorithm are shown in **(a)**. The flow field by Singh is shown in **(b)**. Both of these diagrams appear in Barron et al. [5]. Our results were resampled and are shown in a similar format in **(c)**.

4.5. **Hand-held target.** 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 11) will be used in later stages 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 have been successfully tracked. As before, a relief map is shown in Figure 10 to illustrate the crisp boundaries of the target object and coherency of the flow field magnitude.

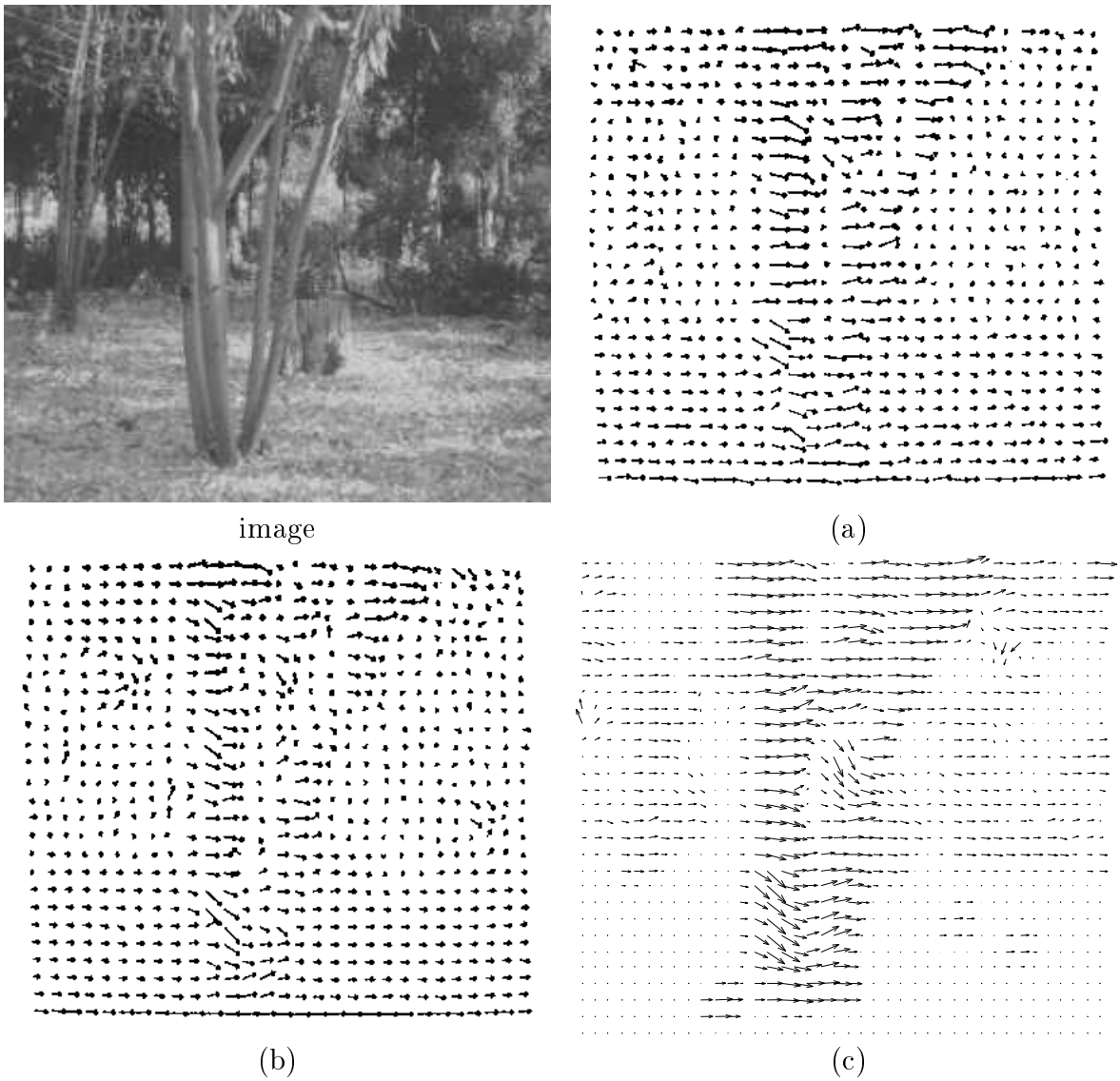


FIGURE 8. **SRI Tree Sequence, other algorithms.** The flow field results by Anandan's algorithm are shown in **(a)**. The flow field by Singh is shown in **(b)**. Both of these diagrams appear in Barron et al. [5]. Our results were resampled and are shown in a similar format in **(c)**.

5. CONCLUSIONS

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 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

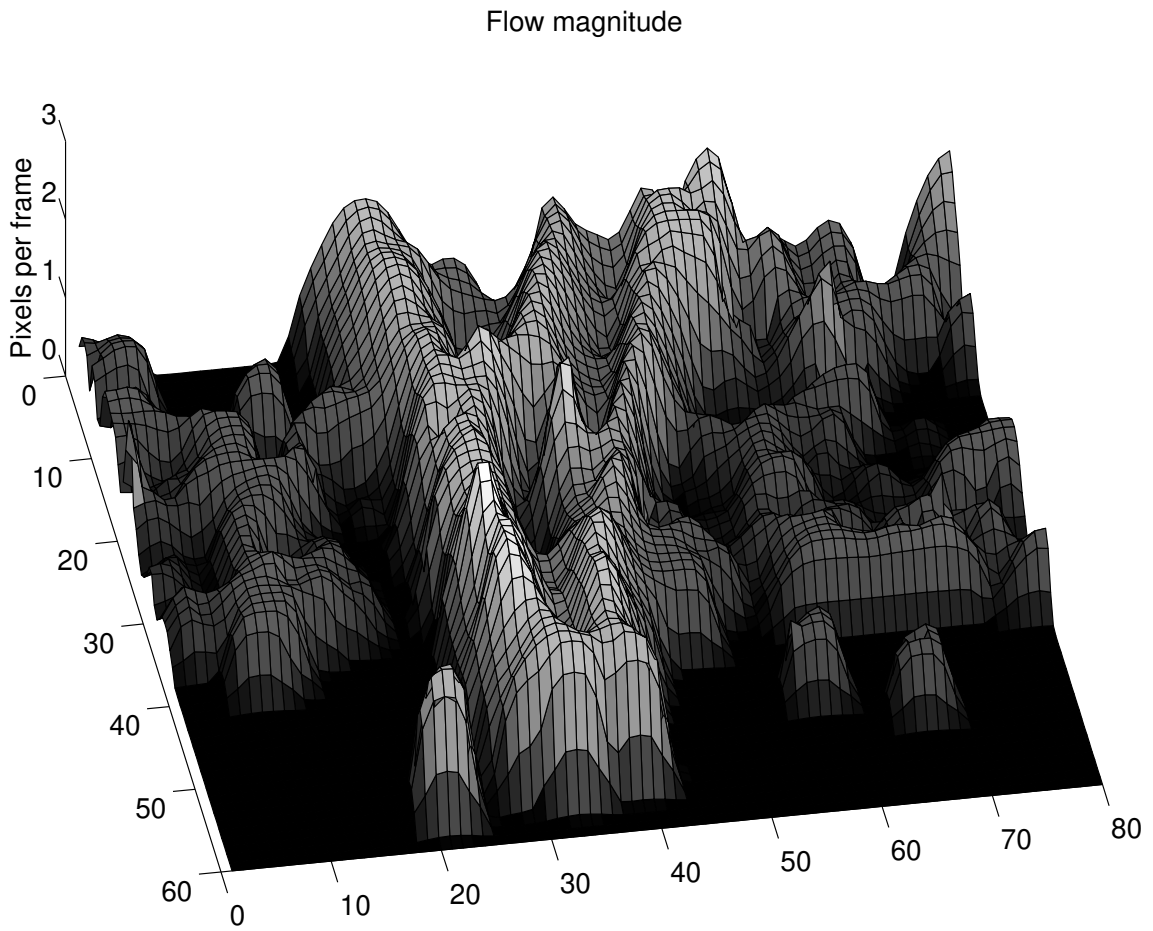


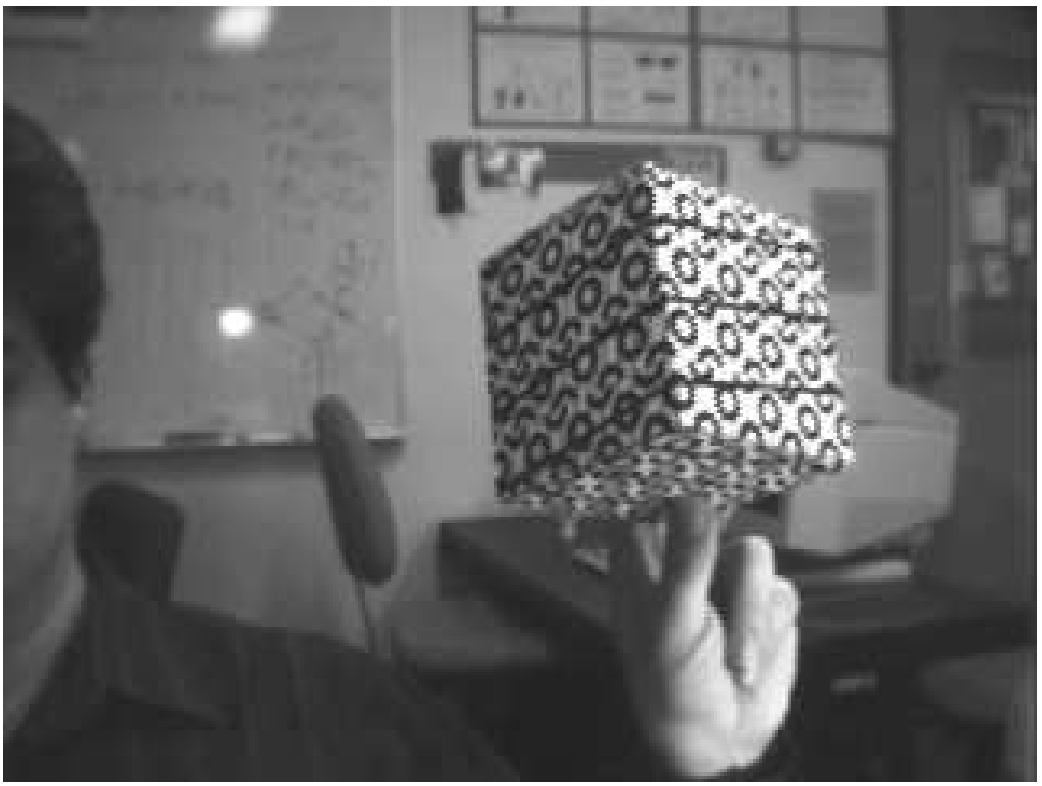
FIGURE 9. **SRI Tree Sequence, Kinetic Depth.** The magnitude of the flow field is rendered here as a relief map. Note how well-defined the foreground tree trunk and its V-shaped branches are.

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).

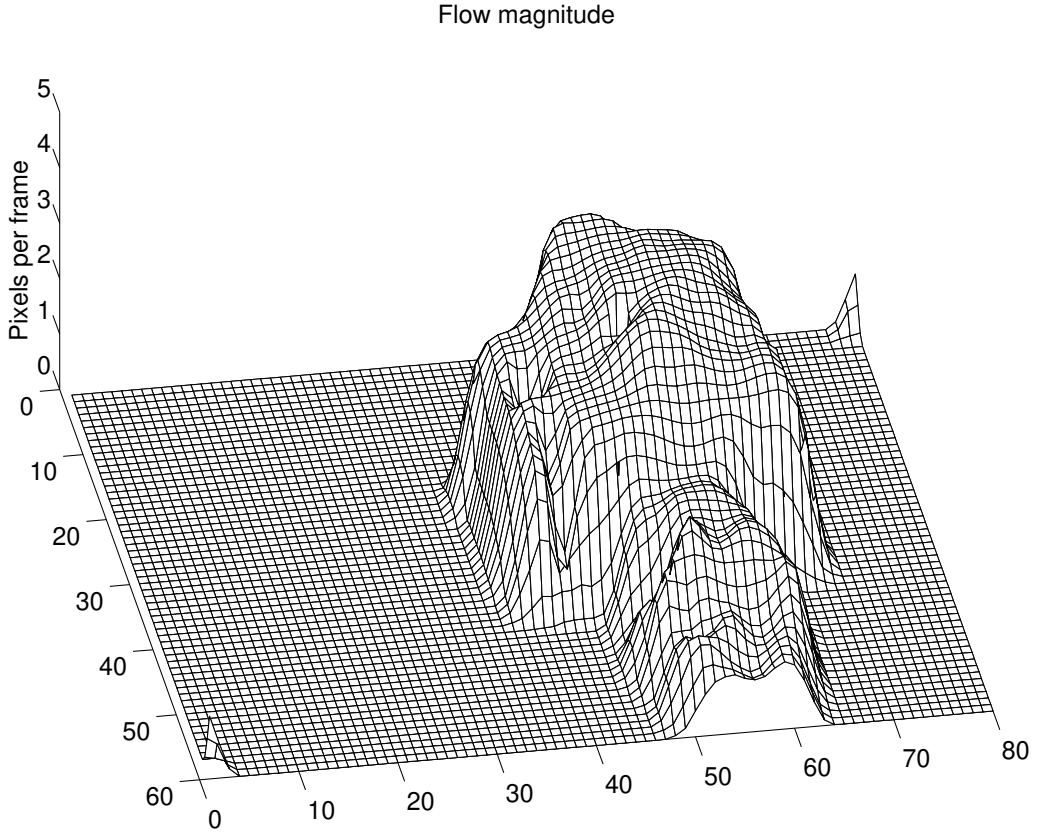
One of our interests is in the application of optical flow analysis to 3-D shape recovery. The example shown in Figures 10 and 11 shows how the algorithm can be used to recover the shape of an object waved in front of the camera. Further improvements are possible by fitting a parametric model to the accumulated data and predicting upcoming flow fields by projecting the object model's motion onto the image plane, accelerating the acquisition process and improving the quality of the flow data. With continuing improvements in flow estimation and computing technologies, practical machine vision applications based in optical flow analysis should find increasing popularity.

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.



(a)



(b)

FIGURE 10. **Hand-Held Target Sequence, Kinetic Depth.** Frame 20 from the image sequence is shown in **(a)**. The magnitude of the flow field is rendered in **(b)** as a relief map. Note how the physical edges of the cube are present in the relief map.

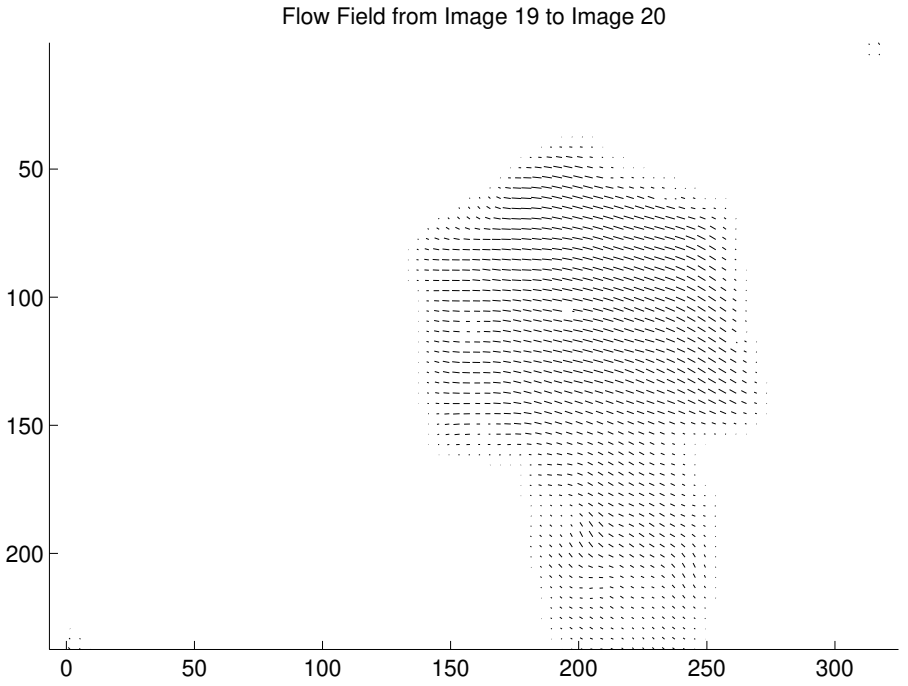


FIGURE 11. **Hand-Held Target Sequence, Flow Field.** The measured flow field between frames 19 and 20.

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

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

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

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

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

E-mail address: {benoits,ferrie}@cim.mcgill.ca