International Journal of Computer Vision, 12:1, 5–16 (1994)

Computing Occluding and Transparent Motions

MICHAL IRANI, BENNY ROUSSO, AND SHMUEL PELEG
Institute of Computer Science, The Hebrew University of Jerusalem, 91904 Jerusalem, Israel

Abstract

Computing the motions of several moving objects in image sequences involves simultaneous motion analysis and segmentation. This task can become complicated when image motion changes significantly between frames, as with camera vibrations. Such vibrations make tracking in longer sequences harder, as temporal motion constancy cannot be assumed. The problem becomes even more difficult in the case of transparent motions.

A method is presented for detecting and tracking occluding and transparent moving objects, which uses temporal integration without assuming motion constancy. Each new frame in the sequence is compared to a dynamic internal representation image of the tracked object. The internal representation image is constructed by temporally integrating frames after registration based on the motion computation. The temporal integration maintains sharpness of the tracked object, while blurring objects that have other motions. Comparing new frames to the internal representation image causes the motion analysis algorithm to continue tracking the same object in subsequent frames, and to improve the segmentation.

1 Introduction

Motion analysis, such as *optical flow* (Horn & Schunck 1981), is often performed on the smallest possible regions, both in the temporal domain and in the spatial domain. Small regions, however, carry little motion information, and such motion computation is therefore inaccurate. Analysis of multiple moving objects based on optical flow (Adiv 1985) suffers from this inaccuracy.

Increasing the temporal region to more than two frames improves the accuracy of the computed optical flow. Methods for estimating local image velocities with large temporal regions have been introduced using a combined spatio-temporal analysis (Fleet & Jepson 1990; Heeger 1988; Shizawa & Mase 1990). These methods assume motion constancy in the temporal regions, that is, motion should remain uniform in the analyzed sequence.

The major difficulty in increasing the size of the spatial region of analysis is the possibility that larger regions will include more than a single motion. Existing approaches for the analysis of multiple motions can be classified into methods that compute the multiple motions without using segmentation (Bergen et al. 1991; Bergen et al. 1992b; Burt et al. 1991; Darrell & Pentland 1991; Shizawa 1992; Shizawa & Mase 1991), and those that separate the motions by segmentation (Meyer & Bouthemy 1992; Peleg & Rom 1990).

Analysis of multiple motions without segmentation has been suggested using the *dominant motion* approach (Bergen et al. 1991; Burt et al. 1991), which finds the parameters of a single translation in a scene with multiple motions without performing segmentation. The dominant-motion approach has also been used to compute two motions from three frames (Bergen et al. 1992b), with the assumption that the motions remain constant in the 3-frame sequence. The motions are computed between registered frame differences, rather than between the original frames. The use of frame differences makes it possible to avoid the segmentation problem, but introduces temporal derivatives which increase the order of the derivatives used in this method. Another method for computing multiple motions without segmentation uses the principle of superposition (Shizawa 1992; Shizawa & Mase 1991). It provides an elegant framework to construct motion transparency

constraints from conventional single-motion constraints, but requires the use of high-order derivatives. In another approach, a robust estimation technique for detecting multiple translating objects (Darrell & Pentland 1991) has been introduced. It assumes motion constancy over several successive frames in the analyzed sequence.

Analysis of multiple motions using segmentation has been suggested (Peleg & Rom 1990) for the simple case of two planar moving regions of constant depth. A more general approach with good experimental results has been presented in a region-based tracking method (Meyer & Bouthemy 1992). Kalman filters are used to predict and update the polygonal shape approximation and the 2-D motion parameters of the tracked regions. Motion-based segmentation (Francois & Bouthemy 1992), based on a statistical regularization approach using MRF models, is being used in that approach to initially separate the moving objects.

In this article we propose a method for detecting and tracking multiple moving objects using both a large spatial region and a large temporal region without assuming temporal motion constancy. When the large spatial region of analysis has multiple moving objects, the motion parameters and the locations of the objects are computed for one object after another using segmentation. The method has been applied successfully using parametric motion models in the image plane, such as affine and projective transformations. Both *transparent* and *occluding* objects are tracked using temporal integration of images registered according to the computed motions.

Section 2 describes the method for detecting the differently moving objects and computing their motion parameters between two successive frames. The tools used for the motion computation and the segmentation are described in that section. Section 3 describes the method for tracking the detected objects using temporal integration of image frames. Section 4 shows how this technique is used for tracking and reconstructing transparent moving objects.

2 Detection of Multiple Moving Objects in Image Pairs

To detect differently moving objects in an image pair, a single motion is first computed, and the object that corresponds to this motion is identified. We call this motion the *dominant motion*, and the corresponding object the *dominant object*. Once a dominant object has been detected, it is excluded from the region of analysis, and the process is repeated on the remaining region to find other objects and their motions. This section describes the methods used for object detection and motion computation between two images.

2.1 The Motion Model

We use 2-D parametric transformations to approximate the projected 3-D motions of the objects on the image plane. This assumption is valid when the differences in depth caused by the motions are small relative to the distances of the objects from the camera. The choice of a 2-D motion model enables efficient motion computations and is numerically stable (since the 2-D problem is highly overdetermined due to the small number of unknowns). Full 3-D motion computation may be difficult and ill conditioned (due to the very large number of unknowns—the 3-D motion parameters plus the depth at each point).

Given two grey-level images of an object, $I(x, y, t)$ and $I(x, y, t + 1)$, we use the assumption of grey-level constancy:

$$I(x + p(x, y, t), y + q(x, y, t), t + 1) = I(x, y, t) \quad (1)$$

where $[p(x, y, t), q(x, y, t)]$ is the displacement induced on pixel (x, y) by the motion of the object between frames t and $t + 1$. Expanding the left-hand side of equation (1) to its first-order Taylor expansion around (x, y, t) and neglecting all nonlinear terms yields

$$I(x + p, y + q, t + 1) = I(x, y, t) + pI_x + qI_y + I_t, \quad (2)$$

where

$$I_x = \frac{\partial I(x, y, t)}{\partial x}, \; I_y = \frac{\partial I(x, y, t)}{\partial y}, \; I_t = \frac{\partial I(x, y, t)}{\partial t},$$

$$p = p(x, y, t), \; q = q(x, y, t)$$

Equations (1) and (2) yield the well-known constraint (Horn & Schunck 1981)

$$pI_x + qI_y + I_t = 0 \quad (3)$$

We look for a motion (p, q) which minimizes the error function at frame t in the region of analysis R:

$$\mathrm{Err}^{(t)}(p, q) = \sum_{(x,y)\in R} (pI_x + qI_y + I_t)^2. \quad (4)$$

We perform the error minimization over the parameters of one of the following motion models:

1. **Translation:** 2 parameters, $p(x, y, t) = a$, $q(x, y, t) = d$. In order to minimize $\text{Err}^{(t)}(p, q)$, its derivatives with respect to a and d are set to zero. This yields two linear equations in the two unknowns, a and d. Those are the two well-known optical-flow equations (Lucas & Kanade 1981; Bergen 1992a), where every small window is assumed to have a single translation. In this translation model, the entire *object* is assumed to have a single translation.
2. **Affine:** 6 parameters, $p(x, y, t) = a + bx + cy$, $q(x, y, t) = d + ex + fy$. Deriving $\text{Err}^{(t)}(p, q)$ with respect to the motion parameters and setting to zero yields six linear equations in the six unknowns: a, b, c, d, e, f (Bergen et al. 1991; Bergen et al. 1992a).
3. **Moving planar surface** (a pseudo projective transformation): 8 parameters (Adiv 1985, Bergen 1992a),

$$p(x, y, t) = a + bx + cy + gx^2 + hxy$$

$$q(x, y, t) = d + ex + fy + gxy + hy^2$$

Deriving $\text{Err}^{(t)}(p, q)$ with respect to the motion parameters and setting to zero, yields eight linear equations in the eight unknowns: a, b, c, d, e, f, g, h.

2.2 *Processing the First Object*

When the region of support of a single object in the image is known, its motion parameters can be computed using a multiresolution iterative framework (Bergen & Adelson 1987; Bergen et al. 1991; Bergen et al. 1992b). The basic components of this framework are:

— Construction of a Gaussian pyramid (Rosenfeld 1984), where the images are represented in multiple resolutions.
— Starting at the lowest resolution level:

1. Motion parameters are estimated by solving the set of linear equations to minimize $\text{Err}^{(t)}(p, q)$ (equation (4)) according to the appropriate motion model (section 2.1). When the region of support of the object is known, minimization is done only over that region.
2. The two images are registered by warping according to the computed motion parameters. Steps 1 and 2 are iterated at each resolution level for further refinements.
3. The motion parameters are interpolated to the next resolution level, and are refined by using the higher resolution images.

Motion estimation is more difficult when the region of support of an object in the image is not known, which is the common case. It was shown by Burt et al. (1991) that the motion parameters of a single object translating *in the image plane* can be recovered accurately by applying the above motion computation framework to the entire region of analysis, using a *translation* motion model. This can be done even in the presence of other differently moving objects in the region of analysis, and with no prior knowledge of their regions of support. A thorough analysis of hierarchical translation estimation is found in (Burt et al. 1991). This, however, is rarely true for *higher-order* 2-D parametric motion models (e.g., affine, projective, etc.), which are much more sensitive to the presence of other moving objects in the region of analysis.

Following is a procedure to compute higher-order (affine, projective, etc.) motion parameters of an object among differently moving objects in an image pair:

1. Compute the dominant 2-D translation in the region by applying a translation computation technique (section 2.1) to the entire region of analysis. This locks onto an existing translation in the region of analysis. In the case of a motion that is not a translation in the image plane, the computed translation is an approximation of the motion of an object segment.
2. Segment out the region that corresponds to the computed motion (the segmentation technique is described in section 2.4). This confines the region of analysis to a region containing only a single motion.
3. Apply a higher-order parametric motion computation (affine, projective, etc) to the segmented region only, to improve the motion estimation.
4. Iterate steps 2–3–4 until convergence.

The above procedure segments a single object and computes its motion parameters using two frames. This object will be referred to as the *dominant object*, and its motion as the *dominant motion*. The choice of the motion model is done gradually. First a translational motion model is used, then an affine motion model, and finally a projective motion model, with segmentation refinements in between. In many cases an affine model suffices, but since the scheme is automatic we apply the projective model as well. This scheme could

theoretically be further extended to yet higher-order parametric transformations *in the image plane.*

An example of a detected dominant object between two frames will be shown in figure 2e. This sequence contains two moving objects: a flying helicopter and a background moving due to camera motion. In this example, noise has strongly affected the segmentation. The problem of noise is overcome once the algorithm is extended to handle longer sequences using temporal integration (section 3).

2.3 Processing Other Objects

After detecting the dominant object between two images, attention is given to other objects. The dominant object is excluded from the region of analysis, and the detection process is repeated on the remaining parts of the image to find other objects and their motion parameters. More details are found in section 3.2.

2.4 Segmentation

Once a motion has been determined, we would like to identify the region having this motion. To simplify the problem, the two images are registered using the detected motion. The motion of the corresponding region is canceled after registration, and the tracked region is stationary in the registered images. The segmentation problem reduces therefore to identifying the stationary regions in the registered images.

In this implementation, pixels are classified as moving or stationary, using simple analysis based on local normalized differences. A more elaborate statistical scheme (Hsu et al. 1984) is also possible. A simple grey-level difference between the registered images is not sufficient for the classification, for two reasons:

1. Regions having uniform intensity may be interpreted locally as both moving and stationary. In order to classify correctly regions having uniform intensity, a multiresolution scheme (Rosenfeld 1984) is used; as in low-resolution pyramid levels the uniform regions are small. Classification is first performed on the lowest resolution level and is then interpolated to be used as an initial classification for the next resolution level. Higher resolution information is used to update the initial classification.
2. Intensity difference cased by motion is also affected by the magnitude of the gradient in the direction of the movement. Therefore, rather than using a simple grey-level difference as a motion measure for classifying the pixels, the grey-level difference *normalized* by the gradient magnitude is used as a local motion measure (5).

A pixel with a high motion measure is very likely to be moving. However, a low motion measure does not necessarily indicate that the pixel is stationary, as in the case of a motion along an edge or in uniform regions. In order to detect stationarity, the reliability of the motion measure is computed. A pixel is classified as stationary only if its motion measure (5) is very low, and the reliability of this measure (7) is high.

Following are the definitions of the motion measure and its reliability of the motion measure and its reliability as used in the segmentation procedure: Let $I(x, y, t)$ and $I(x, y, t + 1)$ be the intensities of pixel (x, y) of the two *registered* images at times t and $t + 1$, and let $\nabla I(x, y, t)$ be the spatial intensity gradient at time t. The *motion measure* $M(x, y, t)$ used is the weighted average of the normal flow magnitudes over a small neighborhood $N(x, y)$ of (x, y) (typically a 3×3 neighborhood). The weights are taken to be $|\nabla I(x_i, y_i, t)|^2$:

$$M(x, y, t) \stackrel{\text{def}}{=} \frac{\sum_{(x_i, y_i) \in N(x, y)} |I(x_i, y_i, t+1) - I(x_i, y_i, t)| \cdot |\nabla I(x_i, y_i, t)|}{\sum_{(x_i, y_i) \in N(x, y)} |\nabla I(x_i, y_i, t)^2 + C} \tag{5}$$

where the constant C is used to avoid numerical instabilities.

The reliability of the motion measure at each pixel is determined by the numerical stability of the two well-known optical-flow equations (Lucas & Kanade 1981; Bergen et al. 1992a):

$$\begin{bmatrix} (\Sigma\, I_x^2) & (\Sigma\, I_x I_y) \\ (\Sigma\, I_x I_y) & (\Sigma\, I_y^2) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} (-\Sigma\, I_x I_t) \\ (-\Sigma\, I_y I_t) \end{bmatrix} \tag{6}$$

where for each pixel (x, y) the sum is taken over the neighborhood $N(x, y)$. The *reliability* $R(x, y, t)$ is expressed by the *inverse* of the condition number of the coefficient matrix in (6):

$$R(x, y, t) \stackrel{\text{def}}{=} \frac{\lambda_{\min}}{\lambda_{\max}} \tag{7}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest eigenvalues, respectively.

The propagation of the motion measure from the lowest resolution level to the highest resolution level in the pyramid is performed as follows: First, all pixels at the lowest resolution level are initialized as "unknown to be moving or stationary." Then for each pixel at each resolution level in the pyramid both the local motion measure (5) and the reliability (7) are computed. If the computed motion measure is high (i.e., pixel is moving) or if it is low with high reliability (i.e., pixel is stationary), then the motion measure of the pixel at that resolution level is set to be the new computed motion measure. Otherwise, if the local information available at the current resolution level does not suffice for classification, then the motion measure from the previous lower resolution level is maintained.

This algorithm yields a *continuous* function, which is an indication of the *magnitude of the displacement* of each pixel between the two images. Taking a threshold on this function yields partitioning of the image to moving and stationary regions. We usually choose the threshold to be about 1 (i.e., a displacement of about one pixel), to allow for noise. The motion measures M for several experiments will be shown in figures 2, 3, 4.

3 Tracking Detected Objects Using Temporal Integration

The algorithm for the detection of multiple moving objects described in section 2 is extended to track detected objects throughout long image sequences. This is done by temporal integration, without assuming temporal motion constancy. For each tracked object a dynamic internal representation image is constructed. This image is constructed by taking a weighted average of recent frames, registered with respect to the tracked motion (to cancel its motion). This image contains, after a few frames, a sharp image of the tracked object, and a blurred image of all the other objects. Each new frame in the sequence is compared to the internal representation image of the tracked object rather than to the previous frame. Similar temporal-integration approaches, but which were applied only to stationary background, are described in (Donohoe et al. 1988; Karmann & Brandt 1989).

3.1 Tracking the Dominant Object

Let $\{I(t)\}$ denote the image sequence, and let $M(t)$ denote the segmentation mask of the tracked object computed for frame $I(t)$, using the segmentation method described in section 2.4. Initially, $M(0)$ is the entire region of analysis. The internal representation image of the tracked object is denoted by $Av(t)$, and is constructed as follows:

$$\begin{aligned} \mathrm{Av}(0) &\stackrel{\mathrm{def}}{=} I(0) \\ \mathrm{Av}(t+1) &\stackrel{\mathrm{def}}{=} (1-w)\cdot I(t+1) \\ &\quad + w \cdot \mathrm{register}\,[\mathrm{Av}(t), I(t+1)] \end{aligned} \tag{8}$$

where register (P, Q) denotes the registration of images P and Q by warping P towards Q according to the motion of the tracked object computed between them, and $0 < w < 1$ (currently $w = 0.7$). $\mathrm{Av}(t)$ therefore maintains sharpness of the tracked object, while blurring other objects in the image. An example of the evolution of an internal representation image of a tracked object is shown in figure 1.

Following is a summary of the algorithm for detecting and tracking the dominant object in an image sequence:

For each frame in the sequence (starting at $t = 0$) do:

1. Compute the dominant motion parameters between the internal representation image of the tracked object $\mathrm{Av}(t)$ and the new frame $I(t+1)$, in the region $M(t)$ of the tracked object (section 2).
2. Warp the current internal representation image $\mathrm{Av}(t)$ and current segmentation mask $M(t)$ toward the new frame $I(t+1)$ according to the computed motion parameters.
3. Identify the stationary regions in the registered images (section 2.4), using the registered mask $M(t)$ as an initial guess. This will be the segmented region $M(t+1)$ of the tracked object in frame $I(t+1)$.
4. Compute the updated internal representation image $\mathrm{Av}(t+1)$ using equation (8), and continue processing the next frame.

When the motion model approximates the temporal changes of the tracked object well enough, shape changes relatively slowly over time in registered images. Therefore, temporal integration of registered frames produces a sharp and clean image of the tracked object, while blurring regions having other motions. The temporal averaging according to equation (8) implies that the weights of images are reduced exponentially in time, giving the highest weights to the most recent frames. Less recent frames, which are blurred by repeated warping and for which the motion

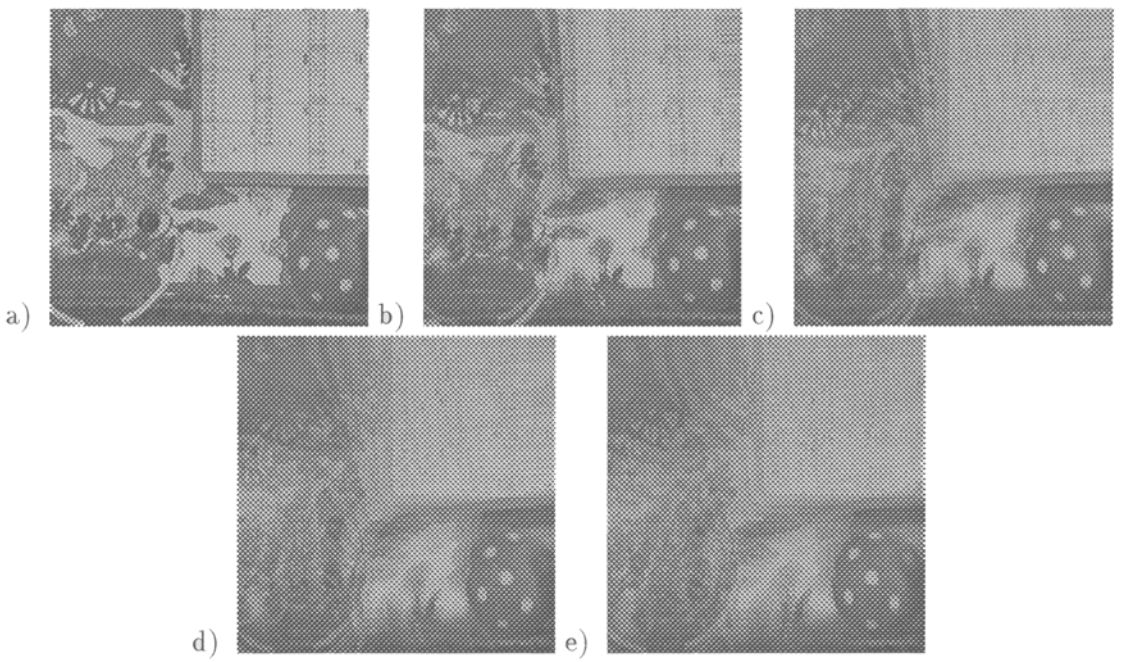

Fig. 1. An example of the evolution of an internal representation image of a tracked object. The scene contains four moving objects. The tracked object is the ball rolling from right to left. (a) Initially, the internal representation image is the first frame in the sequence. (b) The internal representation image after 2 frames. (c) The internal representation image after 3 frames. (d) The internal representation image after 4 frames. (e) The internal representation image after 5 frames: the tracked object (the ball) remains sharp, while all other regions blur out.

parameters might loose their accuracy, are "forgotten" at an exponential rate.

Figure 1 shows an example of the evolution of an internal representation image of a tracked object. The scene contains four moving objects. The tracked object is the ball, which is rolling from right to left. The ball remains sharp, while all other regions gradually blur out.

Comparing each new frame to the internal representation image (e.g., figure 1e) rather than to the previous frame gives the Algorithm a strong bias to keep tracking the same object. Since additive noise is reduced in the average image of the tracked object, and since image gradients outside the tracked object decrease substantially, both the segmentation and the motion computation improve significantly.

In the example shown in figure 2, temporal integration is used to detect and track the dominant object. This sequence contains two moving objects: a flying helicopter and a background moving due to camera motion. Figures 2c and 2d show the motion-measure maps obtained by the segmentation process (section 2.4) for the first frame and after several frames, respectively. Comparing the segmentation shown in figure 2e to the segmentation in figure 2f emphasizes the improvement in segmentation using temporal integration.

Another example of detecting and tracking the dominant object using temporal integration is shown in figure 3. In this sequence, taken by an infrared camera, the background moves due to camera motion, while the car and the pedestrian move independently. It is evident from figure 3c that the tracked object is the background, as all other regions in the image are blurred by their motion.

3.2 Tracking Other Occluding Objects

After detecting and tracking the first object, attention is directed at other objects. This is done by applying the tracking algorithm once more, this time to the rest of the image, after excluding the first-detected object

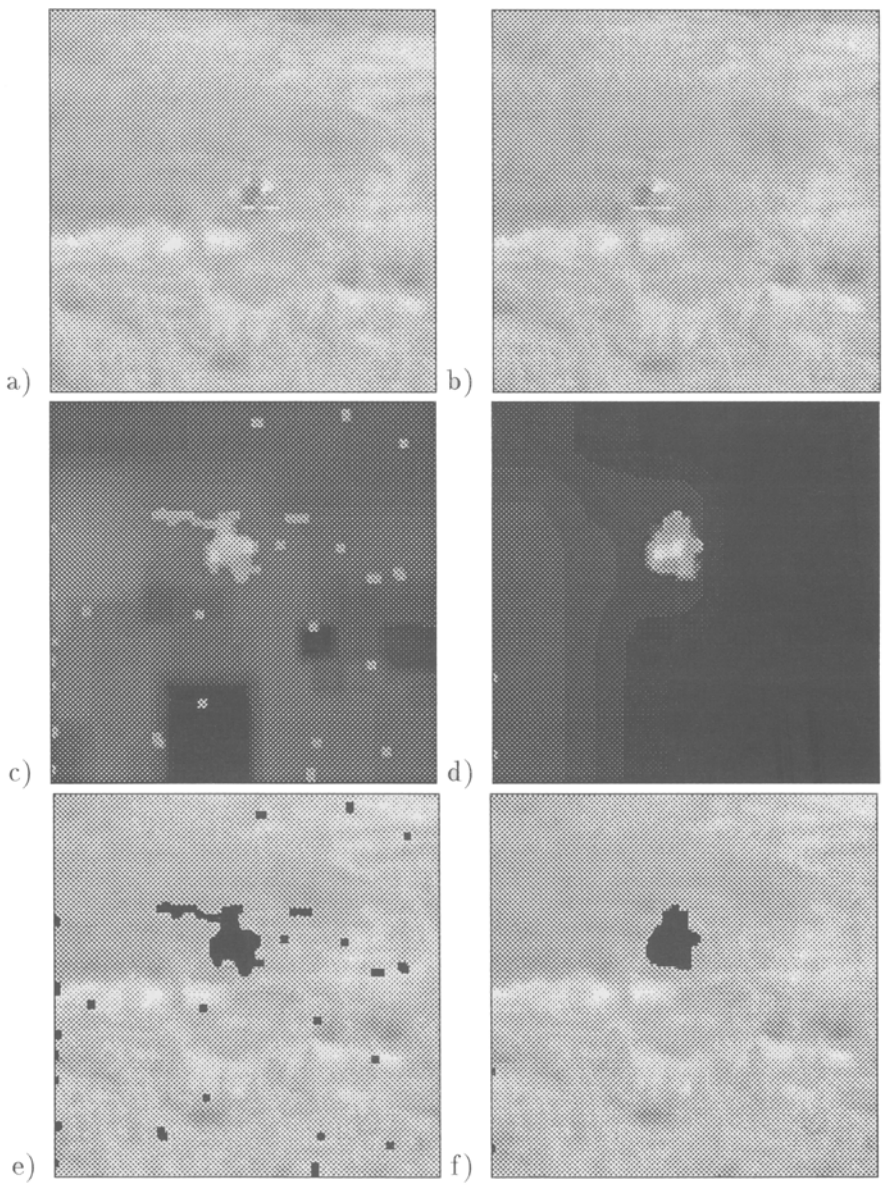

Fig. 2. Detecting and tracking the dominant object using temporal integration. (a–b) The first and last frames in the sequence: both the background and the helicopter are moving. (c) The motion-measure map between the first two frames: bright regions indicate a high motion measure. (d) The motion-measure map after a few frames with the temporal integration process. (e) The segmented dominant object (the background) between the first two frames: black regions are those excluded from the dominant object. (f) The segmented tracked object after a few frames using temporal integration.

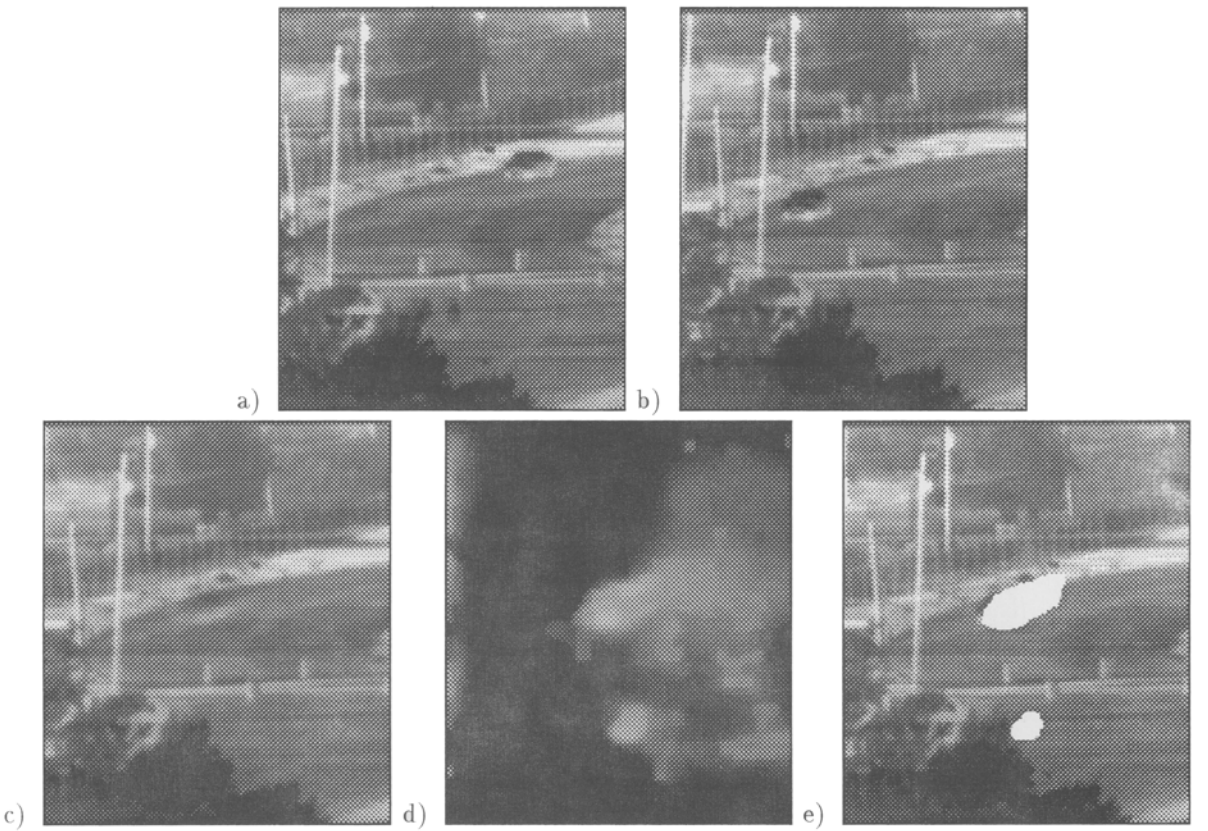

Fig. 3. Detecting and tracking the dominant object in an infrared image sequence using temporal integration. (a–b) The first and last frames in the sequence: both the background and the car are moving, and a person is walking at the bottom part of the road (appears as a dark spot at the lower part of figure 3a). (c) The internal representation image of the tracked object (the background): the background remains sharp with less noise, while the car and the pedestrian blur out. (d) The motion-measure map after several frames: bright regions indicate high motion measure. (e) The segmented tracked object (the background): white regions are those excluded from the tracked region.

from the region of analysis. To increase stability, the displacement between the centroids of the remaining regions of analysis in successive frames is given as the initial guess for the computation of the dominant translation. This increases the chance of detecting small objects that move fast (i.e., objects that have a small overlap between successive frames), like the car in the infrared sequence (figures 3 and 4).

The scheme is repeated recursively, until no more objects can be detected. In cases when the region of analysis consists of many disconnected regions and the motion-analysis algorithm does not converge, the analysis is repeated on the largest connected component in the region.

In the example shown in figure 4, the second dominant object is detected and tracked. It is evident from figure 4b that the tracked object is the car, as all other regions in the image are blurred by their motion.

The detection and tracking of several moving objects can be performed in parallel, with a delay of one or more frames between the computations for different objects.

4 Tracking and Reconstructing Objects in Transparent Motion

We consider a region to have transparent motions if it contains several differently moving image patterns that

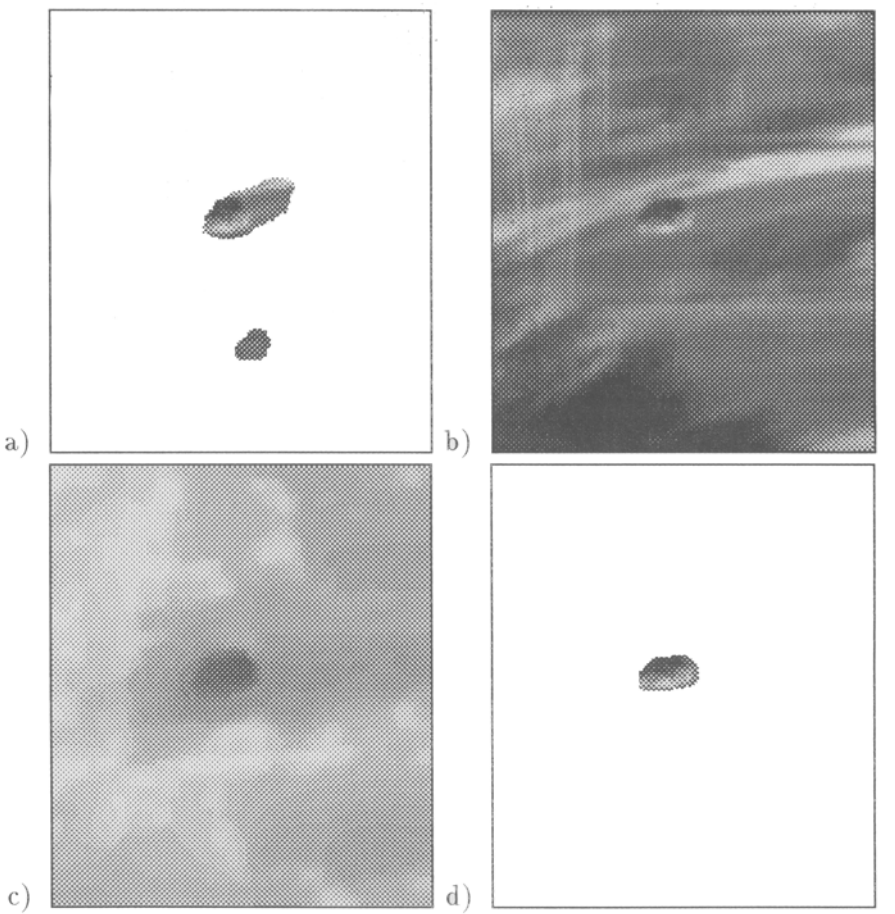

Fig. 4. Detecting and tracking the second object using temporal integeration. (a) The initial region of analysis after excluding the first dominant object (from figure 3e). (b) The internal representation image of the second tracked object (the car): the car remains sharp while the background and the pedestrian blur out. (c) The motion-measure map after several frames: bright regions indicate high motion measure. (d) Segmentation of the tracked car.

appear superimposed. For example, moving shadows, spotlights, reflections in water, transparent surfaces moving past one another, etc. In this section, we show how the tracking algorithm presented in section 3.1 can be used to detect, track, and reconstruct objects in the case of transparent motions.

Previous analysis of transparency (Bergen et al. 1992b; Darrell & Pentland 1991; Shizawa 1992; Shizawa & Mase 1990, 1991) assumed constant motion over several successive frames, which excludes most sequences taken from an unstabilized moving camera. Some methods (Bergen et al. 1992b; Shizawa 1992; Shizawa & Mase 1991) elegantly avoid the segmentation problem. They require, however, high-order derivatives (the order increases with the number of objects), which increases the sensitivity to noisy data.

In our work we do not assume any motion constancy, and we temporally integrate the image frames rather than use temporal derivatives. The temporal integration provides robustness and numerical stability, and also allows us to reconstruct each of the transparent moving objects separately.

Transparent motions yield several motion components at each point, and segmentation cannot be used to isolate one of the transparent objects. In practice, however, due to varying image contrast, in many regions

one object is more prominent than other objects, and segmentation can be used to extract pixels that support better a single motion in the region of analysis. We use the temporal integration scheme described in section 3.1 to track the dominant transparent object. The temporal averaging restores the dominant transparent object in the integrated image, while it blurs out the other transparent objects, making them less noticeable. Comparing each new frame to the integrated image of the tracked object rather than to the previous frame gives the algorithm a strong tendency to keep tracking the same transparent object, as it is the only object in the integrated image that is still similar to its image in the new frame (figure 5).

For recovering the second transparent object, the temporal-integration tracking technique is applied once more to the sequence, after some delay. Let $\mathrm{Av}_1(t)$ denote the integrated image of the first transparent object. Starting at frame $I(t)$, the algorithm is applied only to pixels for which the value of $|I(t) - \mathrm{Av}_1(t)|$ is high. This difference image has high values in regions that contain prominent features of transparent objects in $I(t)$ which faded out in the integrated image $\mathrm{Av}_1(t)$, and low values in regions that correspond to the first dominant transparent object. Therefore, we use the values of the absolute-difference image as an initial mask for the search of the next dominant object in the temporal-integration algorithm from section 3.1. The tracking

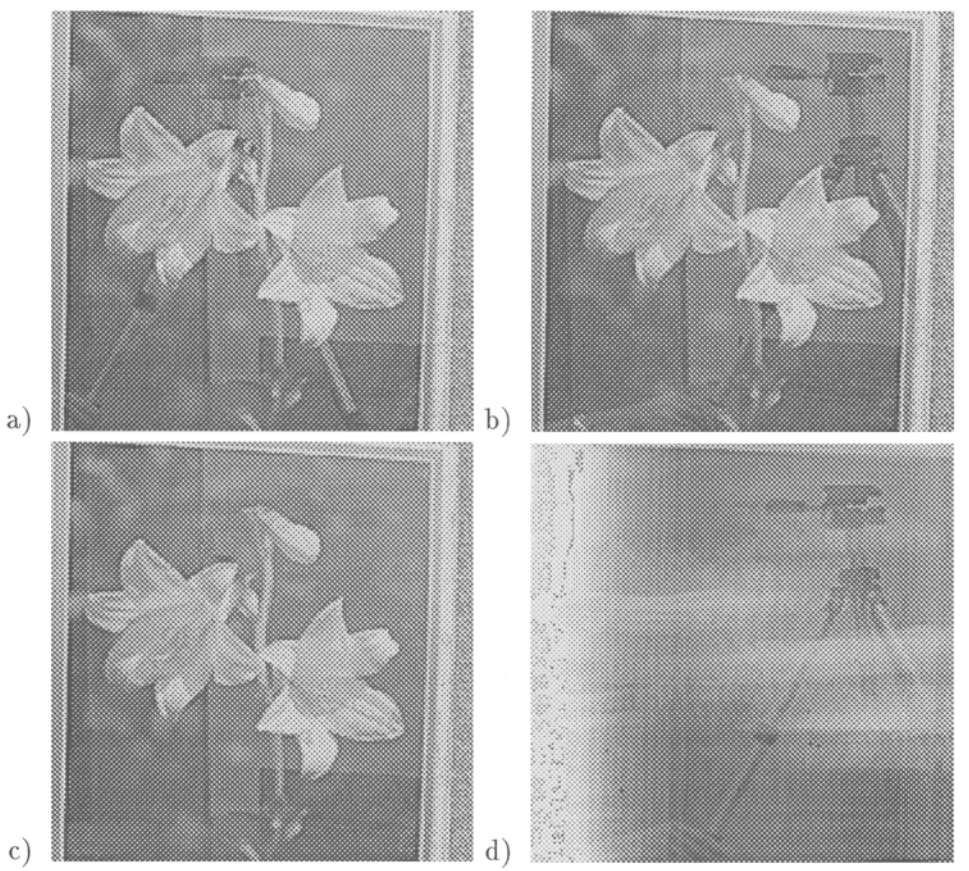

Fig. 5. Reconstruction of "transparent" objects. (a–b) The first and last frames in a sequence: a moving tripod is reflected in a glass covering a picture of flowers. (c) The integrated image of the first tracked object (the picture of flowers) after 14 frames: the picture of flowers was reconstructed; the reflection of the tripod faded out. (d) The integrated image of the second tracked object (the reflection of the tripod) after 14 frames: the reflection of the tripod was constructed; the picture of flowers faded out.

algorithm is applied once again to the *original* image sequence, and not to frame differences as in (Bergen et al. 1992b). Now that the algorithm tracks the second dominant object, the new internal representation image $Av_2(t)$ restores the second dominant transparent object, and blurs out the other transparent objects, including the first dominant object.

In figure 5, the reconstruction of two transparent moving objects in a real image sequence is shown. In this sequence a moving tripod is reflected in a glass covering a picture of flowers. Figure 5c shows reconstruction of the picture of flowers after the tripod has faded out in the integrated image of the first tracked object. Figure 5d shows reconstruction of the tripod after the picture of flowers has faded out in the integrated image of the second tracked object.

5 Concluding Remarks

Temporal integration of registered images proves to be a powerful approach in motion analysis, enabling human-like tracking of moving objects. The tracked object remains sharp in its integrated image, while other objects blur out. Comparing each new frame to the integrated image of the tracked object rather than to the previous frame gives the algorithm a strong tendency to keep tracking the same object. In case of occluding objects, this improves the accuracy of segmentation and motion computation. In case of transparent moving objects, this also yields an isolated reconstructed image for each of the transparent tracked objects. Other objects can then be tracked.

Once good motion estimation and segmentation of a tracked object are obtained, it becomes possible to enhance the object images, like reconstruction of occluded regions and improvement of image resolution (Irani & Peleg 1991, 1992).

Acknowledgments

This research was supported by the Israel Science Foundation. M. Irani and B. Rousso were partially supported by a fellowship from the Leibniz Center.

References

Adiv, G. 1985. Determining three-dimensional motion and structure from optical flow generated by several moving objects, *IEEE Trans. Patt. Anal. Mach. Intell.* 7(4):384–401, July.

Bergen, J.R., and Adelson, E.H. 1987. Hierarchical, computationally efficient motion estimation algorithm, *J. Opt. Soc. Amer. A.*, 4:35.

Bergen, J.R., Anandan, P., Hanna, K.J., and Hingorani, R. 1992a. Hierarchical model-based motion estimation, *Europ. Conf. Comput. Vis.* pp. 237–252, Santa Margarita Ligure, May.

Bergen, J.R., Burt, P.J., Hanna, K., Hingorani, R., Jeanne, P., and Peleg, S. 1991. Dynamic multiple-motion computation. In Y.A. Feldman and A. Bruckstein, ed., *Artificial Intelligence and Computer Vision: Proceedings of the Israeli Conference*, pp. 147–156. Elsevier: New York.

Bergen, J.R., Burt, P.J., Hingorani, R., and Peleg, S. 1992b. A three-frame algorithm for estimating two-component image motion, *IEEE Trans. Patt. Anal. Mach. Intell.* 14:886–895, September.

Burt, P.J., Hingorani, R., and Kolczynski, R.J. 1991. Mechanisms for isolating component patterns in the sequential analysis of multiple motion, *IEEE Workshop on Visual Motion*, pp. 187–193, Princeton, October.

Donohoe, G.W., Hush, D.R., and Ahmed, N. 1988. Change detection for target detection and classification in video sequences, *Intern. Conf. Acous. Speech Sig. Process.*, pp. 1084–1087, New York.

Darrell, T., and Pentland, A. 1991. Robust estimation of a multi-layered motion representation, *IEEE Workshop on Visual Motion*, pp. 173–178, Princeton, October.

Fleet, D.J. and Jepson, A.D. 1990. Computation of component image velocity from local phase information, *Intern. J. Comput. Vis.* 5(1):77–104.

Francois, E., and Bouthemy, P. 1992. Multiframe-based identification of mobile components of a scene with a moving camera, *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, pp. 282–287, Champaign, June.

Heeger, D.J. 1988. Optical flow using spatiotemporal filters, *Intern. J. Comput. Vis.* 1:279–302.

Horn, B.K.P., and Schunck, B.F. 1981. Determining optical flow, *Artificial Intelligence* 17:185–203.

Hsu, Y.A., Nagel, H.-H., and Rekers, G. 1984. New likelihood test methods for change detection in image sequences, *Comput. Vis. Graph., Image Process.* 26:73–106.

Irani, M., and Peleg, S. 1991. Improving resolution by image registration, *Comput. Vis., Graph. Image Process.* 53:231–239, May.

Irani, M., and Peleg, S. 1992. Image sequence enhancement using multiple motions analysis, *Proc. Conf. Comput. Vis. Patt. Recog.*, Champaign, June.

Karmann, K.P., and Brandt, A.V., 1989. Moving object recognition using an adaptive background memory, *Proc. 3rd Intern. Workshop on Time-Varying Image Process. Mov. Object Recog.*, pp. 289–296, Florence, May.

Lucas, B.D., and Kanade, T. 1981. An iterative image registration technique with an application to stereo vision, *Proc. Image Understanding Workshop*, pp. 121–130.

Meyer, F., and Bouthemy, P. 1992. Region-based tracking in image sequences, *Proc. 2nd Europ. Conf. Comput. Vis.*, pp. 476–484, Santa Margarita Ligure, May.

Peleg, S., and Rom, H. 1990. Motion based segmentation, *Proc. Intern. Conf. Patt. Recog.* 1:109–113, Atlantic City, June.

Rosenfeld, A. ed. 1984. *Multiresolution Image Processing and Analysis*. Springer-Verlag: New York.

Shizawa, M. 1992. On visual ambiguities due to transparency in motion and stereo, *Proc. 2nd Europ. Conf. Comput. Vis.*, pp. 411–419, Santa Margarita Ligure, May.

Shizawa, M., and Mase, K. 1990. Simultaneous multiple optical flow estimation, *Proc. 10th Intern. Conf. Patt. Recog.*, pp. 274–278, Atlantic City, June.

Shizawa, M., and Mase, K. 1991. Principle of superposition: A common computational framework for analysis of multiple motion, *IEEE Workshop on Visual Motion*, pp. 164–172, Princeton, October.