Reprinted from

image AND vision COMPUTING

Image and Vision Computing 17 (1999) 189–199

Computation of instantaneous optical flow using the phase of Fourier components

David Vernon

Department of Computer Science, National University of Ireland, Maynooth, Kildare, Ireland

Received 4 April 1997; received in revised form 27 February 1998; accepted 3 March 1998

ELSEVIER

ELSEVIER

Image and Vision Computing 17 (1999) 189–199

Image AND Vision COMPUTING

Computation of instantaneous optical flow using the phase of Fourier components

David Vernon

Department of Computer Science, National University of Ireland, Maynooth, Kildare, Ireland

Received 4 April 1997; received in revised form 27 February 1998; accepted 3 March 1998

Abstract

A technique for computing the instantaneous optical flow of two images is presented. The velocity at each point in the image can be computed by treating a local region as a distinct sub-image which is translating with some velocity, and by identifying the Fourier components which exhibit the magnitude and phase changes which are consistent with this velocity. The velocity detection itself is accomplished using a Hough transform. The approach lends itself to the production of arbitrarily dense optical flow fields and the velocity vectors are computed to sub-pixel accuracy. Image data in a region are weighted as a function of its distance from the region centre to reduce the impact of 'edge effects' caused by the entry and exit of visual data at the region boundary, thereby violating the assumption of pure image translation. Results are presented for Gaussian weighting functions of three standard deviations, each representing increased attenuation of image data toward the edge of the image. The proposed approach is evaluated using Otte and Nagel's benchmark image sequence [Lecture Notes in Computer Science, Computer Vision—ECCV'94, 1994, pp. 51–60], for which ground-truth data are available, and both maximum and RMS errors of velocity magnitude and direction are computed.

Keywords: Fourier components; Optical flow; Velocity; Hough transform; Gaussian weighting function

1. Introduction

The measurement of optical flow has received a great deal of attention from the computer vision community, and the literature is replete with a very large number of publications on the topic. Surveys and comparisons of the different approaches can be found in comprehensive works by Barron et al. [1], and Otte and Nagel [2]. Most approaches to the measurement of optical flow in images normally exploit one of two primary techniques. The first involves the computation of the spatiotemporal derivatives (first-order or second-order), differentiating the (filtered or unfiltered) image sequence with respect to time and thus computing the optical flow field (e.g. Ref. [3]). The second involves either feature- or region-based matching (e.g. normalized cross-correlation) of local iconic information such as raw image data or segmented features or objects. Comparisons of the many variations of these approaches and the relationship between them can be found in Refs [1,2,4,5].

A lesser-used approach exploits the regularity in spatiotemporal-frequency representations of the image, e.g. the spatiotemporal Fourier Transform Domain, resulting from certain types of image motion. It can be shown that the spatiotemporal Fourier Transform of an image sequence in which the image content is moving with constant velocity results in a spatiotemporal-frequency representation which is equal to the spatial Fourier Transform of the first image multiplied by a δ-Dirac function in the temporal-frequency domain. This δ-Dirac function is dependent on the image velocity which can be computed if one knows the position of the δ-Dirac function and any spatial frequency [6]. Because this approach is based on image motion, rather than object motion, it normally assumes uniform (zero) background when evaluating object motion. Extensions of the technique have been developed to allow it to cater for situations involving noisy backgrounds [7], several objects [8,9], and non-uniform cluttered backgrounds [10].

The use of spatiotemporal frequency representations for the measurement of optical flow has been developed in depth by Fleet and Jepson [1,11–13], who have extended the spatiotemporal frequency framework to deal with situations where the normal assumption of a single pure (local) image translation is no longer valid. Their technique, which is based on the application of a bank of spatiotemporal band-limited velocity-tuned linear filters, is able to distinguish different velocities within a given neighbourhood, and is resilient to small affine geometric deformations of the image neighbourhood.

In this paper, we present an alternative formulation of the spatiotemporal frequency approach. This alternative uses the normal spatial Fourier Transform together with a

0262-8856/99/$ - see front matter
PII: S0262-8856(98)00094-8

Hough Transform, rather than the spatiotemporal Fourier Transform. The velocity at each point in the image is computed by treating a local region as a distinct sub-image which is translating with some velocity, and by identifying the Fourier components which exhibit the magnitude and phase changes which are consistent with this velocity. The velocity detection itself is accomplished using an appropriate Hough transform. This Hough transform embodies the relationship between velocity and phase change, and velocity is measured by locating local maxima in the Hough space. Because it bases the velocity computation on the Fourier phase information, the approach is able to estimate the velocity of any signal function except a uniform flat field (which has an ambiguous flow field in any case) and, consequently, the approach lends itself to the production of arbitrarily dense optical flow fields. In addition, the technique facilitates the computation of the velocity vectors with sub-pixel accuracy.

As we will see in the next section, the computation exploits a property of the Fourier transform of a signal translating with uniform velocity, and the underlying assumption is that we are dealing with translating signals, i.e. signals which are identical at times t_0 and t_1, except for a translation. In the estimation of velocity at a point, we base the estimate of the velocity on the translation which is apparent in a local region (or window). Unfortunately, the image data in such a region will, in fact, exhibit a change due not only to the signal shift, but also the translation of objects into the window and out of the window. Consequently, there is a change in the spectral content of the window and not just a phase change as is assumed in the model. In order to reduce the impact of this 'edge effect', image data in a region are weighted as a function of its distance from the region centre, i.e. the region is windowed or apodized. In this paper, a Gaussian weighting function is used and the Gaussian's standard deviation σ chosen such that the weighting at some distance from the region centre is 50% of that at the region centre, where w is the length (in pixels) of the side of the 2D region. Results are presented for Gaussian weighting functions of three standard deviations, each representing increased attenuation of image data toward the edge of the image (the three functions provide 50% weighting at $w/8$, $2w/8$ and $3w/8$ from the region centre). In the following, we will denote the three Gaussian functions as $\sigma w/8$, $\sigma 2w/8$ and $\sigma 3w/8$.

The proposed approach is evaluated using two images from Otte and Nagel's benchmark image sequence [2], for which ground-truth data are available, and both maximum and RMS errors of velocity magnitude and direction are reported.

2. Overview of the approach

The discrete Fourier transform F ($f(x,y)$) of a 2D function $f(x,y)$ is given by:

$$F(f(x,y)) = \mathbf{F}(k_x, k_y)$$

$$= \sum_x \sum_y f(x,y) e^{i(k_x x + k_y y)}$$

and the inverse discrete Fourier transform is:

$$f(x,y) = F^{-1}(\mathbf{F}(k_x, k_y))$$

$$= \frac{1}{(2\pi)^2} \sum_{k_x} \sum_{k_y} |\mathbf{F}(k_x, k_y)| e^{i\phi(k_x,k_y)} e^{-i(k_x x + k_y y)}$$

where $|\mathbf{F}(k_x,k_y)|$ is the real-valued amplitude spectrum and $\phi(k_x,k_y)$ is the real-valued phase spectrum.

For a function translating with constant velocity (v_x,v_y), $f(x, y)$ becomes $f(x - v_x\delta t, y - v_y\delta t)$. By the shift property [14], its Fourier transform is given by:

$$F(f(x - v_x\delta t, y - v_y\delta t)) = |\mathbf{F}(k_x, k_y)| e^{i\phi(k_x,k_y)} e^{-i(k_x v_x \delta t + k_y v_y \delta t)}$$

Thus, a spatial shift of $(v_x\delta t, v_y\delta t)$ of a signal in the spatial domain, i.e. $f(x,y)$ shifted to $f(x - v_x\delta t, y - v_y\delta t)$, only produces a change in the phase of the Fourier components in the frequency domain. This phase change is $e^{-i(k_x v_x \delta t + k_y v_y \delta t)}$. In order to estimate the velocity of a signal translating with constant velocity in the image, we simply need to identify the set of frequency components k_x and k_y which have all been modified by the same phase shift, i.e. $e^{-i(k_x v_x \delta t + k_y v_y \delta t)}$. To accomplish this, we note that the phase spectrum for the shifted wave at time $t + \delta t$ is equal to the phase spectrum of the wave at time t multiplied by the phase change given above:

$$e^{i\phi_{t+\delta t}(k_x,k_y)} = e^{-i(k_x v_x \delta t + k_y v_y \delta t)} e^{i\phi_t(k_x,k_y)}$$

$$= e^{i(\phi_t(k_x,k_y) - (k_x v_x \delta t + k_y v_y \delta t))}$$

Hence:

$$\phi_{t+\delta t}(k_x,k_y) = \phi_t(k_x,k_y) - (k_x v_x \delta t + k_y v_y \delta t)$$

That is the phase at time $t + \delta t$ is equal to the initial phase at time t minus $(k_x v_x \delta t + k_y v_y \delta t)$. Since we require v_x and v_y, we rearrange as follows:

$$v_y = \frac{1}{k_y \delta t}(\phi_t(k_x,k_y) - \phi_{t+\delta t}(k_x,k_y) - k_x v_x \delta t) \quad (1)$$

This equation is degenerate if $k_y = 0$, in which case we substitute $k_y = 0$ into Eq. (1) and use an alternative re-arrangement as follows:

$$v_x = \frac{1}{k_x \delta t}(\phi_t(k_x,k_y) - \phi_{t+\delta t}(k_x,k_y)) \quad (2)$$

If we have two images taken at time $t = t_0$ and $t = t_0 + \delta t$, we can compute ϕ_{t0} and $\phi_{t0+\delta t}$. Treating the equation above as a Hough transform, with a 2D Hough transform space defined on v_x, v_y, then we can compute v_y for all possible values of v_x, and for all (known) values of k_x, k_y, $\phi_t(k_x, k_y)$, $\phi_{t+\delta t}(k_x, k_y)$ Local maxima in this v_x, v_y Hough transform space signify Fourier components which comprise signals in the spatial domain which are moving with constant velocity v_x, v_y. Note that, in both Eq. (1) and Eq. (2), $\phi(k_x, k_y)$ represents the absolute phase of

frequency (k_x, k_y). However, in the Fourier domain, the phase is bounded by $\pm 2\pi$ and phase values will 'wrap' as they cross this threshold. In effect, phase values are represented modulo 2π. In this implementation, we have allowed for this by solving Eq. (1) and Eq. (2) for the given phase values $\phi(k_x, k_y) + 2n\pi$, $\phi > 0$; $\phi(k_x, k_y) - 2n\pi$, $\phi < 0$, for all n such that $2n\pi < |k_x v_{x\max}\delta t| + |k_y v_{y\max}\delta t|$.

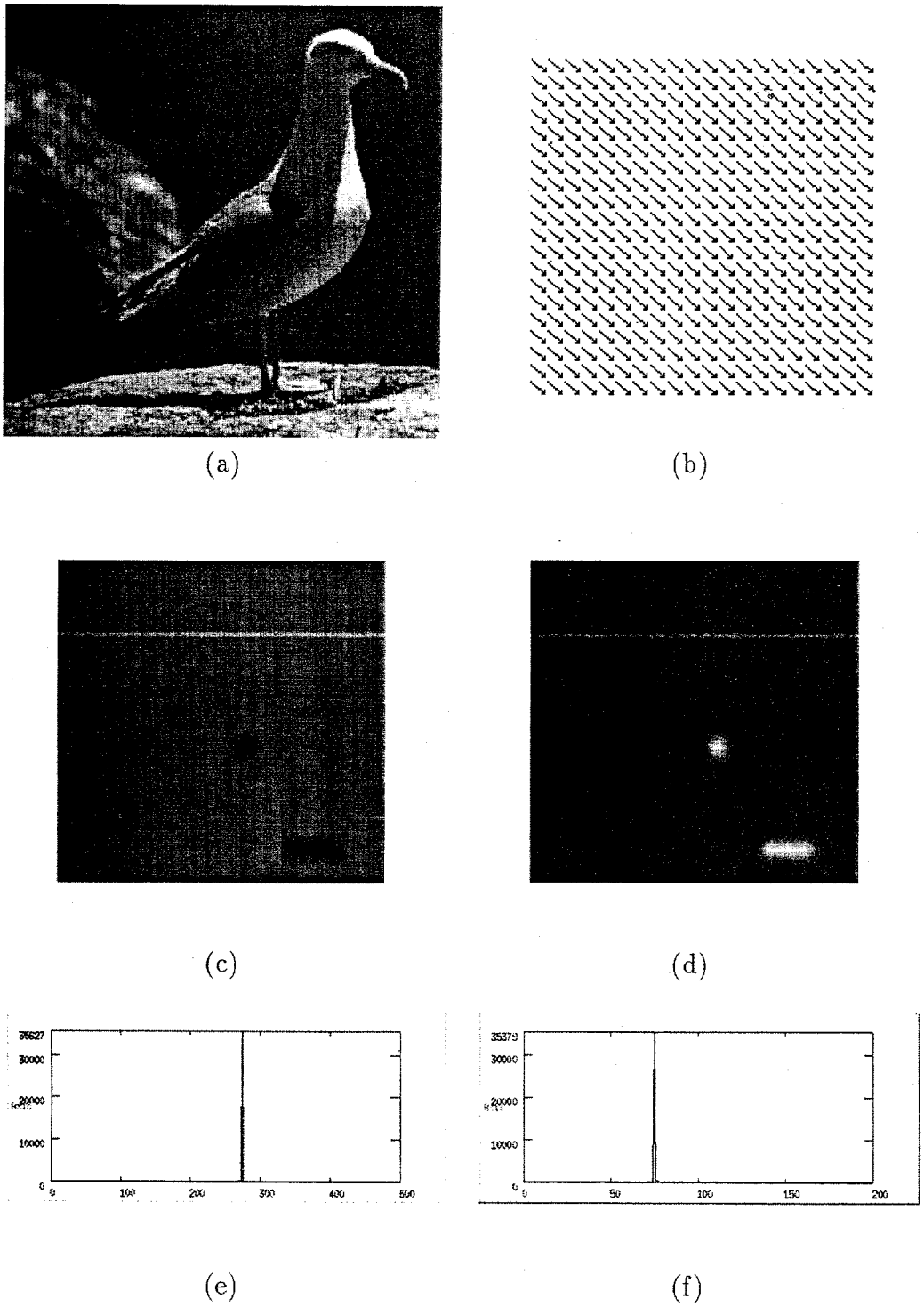

Fig. 1. (a) Simple test sequence: optical flow is computed using this image, and this image is translated by 2 pixels in the x and y direction. (b) The optical flow field is computed using phase information (Gaussian weighting function with 50% weight at $w/4$ from the window centre; w, the window size, equals 64 pixels). (c), (d) Absolute value of error in magnitude and direction of computed flow, respectively (error is proportional to darkness; maximum magnitude error = 0.141 pixels, maximum direction error = 0.025 radians). (e), (f) Histogram of magnitude and direction values of computed flow, respectively (values are scaled by 100).

We note in passing that the computational complexity of the algorithm for the estimation of the velocity of one region is $O(nm^2)$ where m is the dimension of the Fourier domain and n is the dimension of the Hough transform space (and is proportional to the measurable accuracy) since Eq. (1) must be computed for all x and y spatial frequencies, and for all possible values of v_x. Thus, the computational complexity of estimation of the total optical flow field is $O(nm^4)$, since the dimension of the Fourier domain m is the same as that of the original image.

The full algorithm is summarized below.

3. Results

Fig. 1 shows the result of applying the technique to two images where the second image is simply the first translated in the x and y direction by 2 pixels. It should be emphasized that this test is intended to do no more than demonstrate the accuracy and repeatability of the technique on real data with a known (and trivial) flow-field. Fig. 1(a) shows the test image. Fig. 1(b) shows the computed flow field using Gaussian weighting functions with 50% weight at $w/4$ pixels from window centre (w = window size). Note that all of the

```
/* compute the optical flow at coordinates i and j in images f1(i,j) and f2(i,j)   */
/* where i and j are the coordinates of the centre of a valid 64 x 64 pixel region */
/* i and j effectively sample the image with a sampling period sp                   */
/* (sp = 10 pixels in the results presented in this paper)                          */
/* The dimensions of f(i,j) are assumed to be given by a variables d_i and d_j      */

initial_i = 32; final_j = d_i - 32;
initial_j = 32; final_j = d_j - 32;

for (i = initial_i; i < final_i; i = i + sp)
   for (j = initial_j; j < final_j; j = j + sp)

      extract 64x64 pixel regions f1' and f2', centred at i,j, from f1 and f2

      apodized/window f1' and f2' by computing

         g1(x,y) = f1'(x,y) x G(x,y)  // G(x,y) is a 64x64 pixel Gaussian
         g2(x,y) = f2'(x,y) x G(x,y)  // = 0.5 at nw/8 pixels from centre
                                      // (n=1, 2, 3 and w = 64)

      compute G1(kx, ky), the Fourier transform of g1(x,y)
      compute G2(kx, ky), the Fourier transform of g2(x,y)

      compute the phases of G1 and G2:  P1(kx, ky) and P2(kx, ky)

      for (kx = inital_kx; kx < final_kx; kx = kx + 1)          // -32 < kx < 32
         for (ky = inital_k; ky < final_ky; ky = ky + 1)       // -32 < ky < 32

            compute phase difference: pd = P1(kx, ky) - P2(kx, ky);

            if (ky != 0)
               for (vx = 0; vx < 10; vx = vx + 0.1)           // vx is the x velocity
                  vy = (pd - (kx * vx)) / ky
                  Hough_accumulator[vx,vy] += 1
            else if (kx != 0)
               vx = pd / kx;
               for (vy = 0; vy < 10; vy = vy + 0.1)           // vy is the y velocity
                  Hough_accumulator[vx,vy] += 1

      Identify vx_max, vy_max such that
      Hough_accumulator[vx_max][vy_max] >= Hough_accumulator[vx][vy], for all vx, vy

      vx_max, vy_max is the velocity of the 64x64 region centred at i,j
```

Table 1
Summary of mean magnitude and mean direction of ground-truth data and measured velocities

Sequence	Gaussian weighting	Magnitude		Direction	
		Mean	Standard deviation	Mean	Standard deviation
Image translation benchmark	Ground truth	2.828	0.0	0.785	0.0
	$3w/8$	2.756	0.009	0.760	0.003
	$2w/8$	2.757	0.005	0.760	0.001
	$w/8$	2.745	0.018	0.764	0.006
Otte and Nagel benchmark	Ground truth	1.586	0.627	0.318	0.217
	$3w/8$	1.171	0.503	0.597	0.297
	$2w/8$	1.207	0.523	0.588	0.291
	$w/8$	1.157	0.526	0.579	0.288

results shown in this paper were computed with a window size of 64×64 pixels. Fig. 1(c) and (d) shows the absolute value of the error of the magnitude and direction of the computed flow field. Fig. 1(e) and (f) shows a histogram of the magnitude and direction values of the computed flow field. Table 1 summarizes the mean magnitude and mean direction of ground-truth data, and the measured velocities; Table 2 provides a summary of the RMS and maximum errors of the measured velocities. The chief point to note about these results is that the correct flow field is computed to within 0.1 pixels (magnitude) and 0.03 radians (direction).

Figs. 2–5 demonstrate the results of applying the technique to two images in Otte and Nagel's ground-truth test sequence [2]. This sequence comprises images of a real scene acquired with a camera mounted on a moving robot arm. The camera motion is a pure 3D translation toward the scene which comprises a stationary ground plane, four stationary pillars and a fifth pillar which is translating to the left. The scene exhibits strong occlusion by both the stationary and moving pillars. The magnitude and direction of the complete optical flow fields of the sequence were computed by Otte and Nagel on the basis of camera calibration data and the robot motion parameters. The sequence and ground-truth flow fields are available by anonymous ftp at www-kogs.iitb.fhg.de/~mut/kogs/node6/htm#database.

Fig. 2(a) and (b) shows images number 40 and 41 in the sequence. Fig. 2(c) is the true optical flow field extracted directly from the ground-truth data (sampled every 10 pixels). Fig. 2(d)–(f) show the optical flow field computed in the manner described above, and using Gaussian weighting functions with 50% weight at $3w/8$, $2w/8$ and $w/8$ pixels from the window centre, respectively (w = window size). Flow vectors are plotted every 10 pixels and their magnitude has been scaled by a factor of four.

Fig. 3(a) shows the true magnitude of the optical flow field extracted from Otte and Nagel's ground-truth data, while Fig. 3(b)–(d) shows the magnitude of the computed optical flow field using Gaussian weighting functions with 50% weight at $3w/8$, $2w/8$ and $w/8$ pixels from the window centre, respectively. These images were generated by computing the optical flow vectors every 10 pixels (in both directions) and then by interpolating (bi-linearly) between these computed values.

Fig. 4 shows the true direction of the optical flow field extracted from Otte and Nagel's ground-truth data, while Fig. 4 (b)–(d) shows the direction of the computed optical flow field using Gaussian weighting functions with 50% weight at $3w/8$, $2w/8$ and $w/8$ pixels from the window centre, respectively. Again, these images were generated by bi-linear interpolation between the optical flow vector values estimated every 5 pixels.

Fig. 5 (a)–(c) shows the absolute difference between ground-truth magnitude values and those computed using phase information [i.e. the difference between Fig. 3(a) and (b), (c) and (d), respectively]. In the same manner, Fig. 5 (d)–(f) shows the absolute difference between ground-truth direction values and those computed using phase information [i.e. the difference between Fig. 4 (a) and (b), (c) and (d), respectively].

Again, Table 1 summarizes the mean magnitude and mean direction of ground-truth data and measured velocities; Table 2 provides a summary of the RMS and maximum errors of the measured velocities.

Table 2
Summary of errors in measured velocities

Sequence	Gaussian weighting	RMS error		Maximum error	
		Magnitude (pixels)	Direction (radians)	Magnitude (pixels)	Direction (radians)
Image translation benchmark	$3w/8$	0.072	0.024	0.141	0.025
	$2w/8$	0.070	0.021	0.141	0.025
	$w/8$	0.085	0.021	0.141	0.025
Otte and Nagel benchmark	$3w/8$	0.591	0.380	−8.215	3.119
	$2w/8$	0.516	0.355	−3.605	−3.141
	$w/8$	0.583	0.346	2.649	−3.141

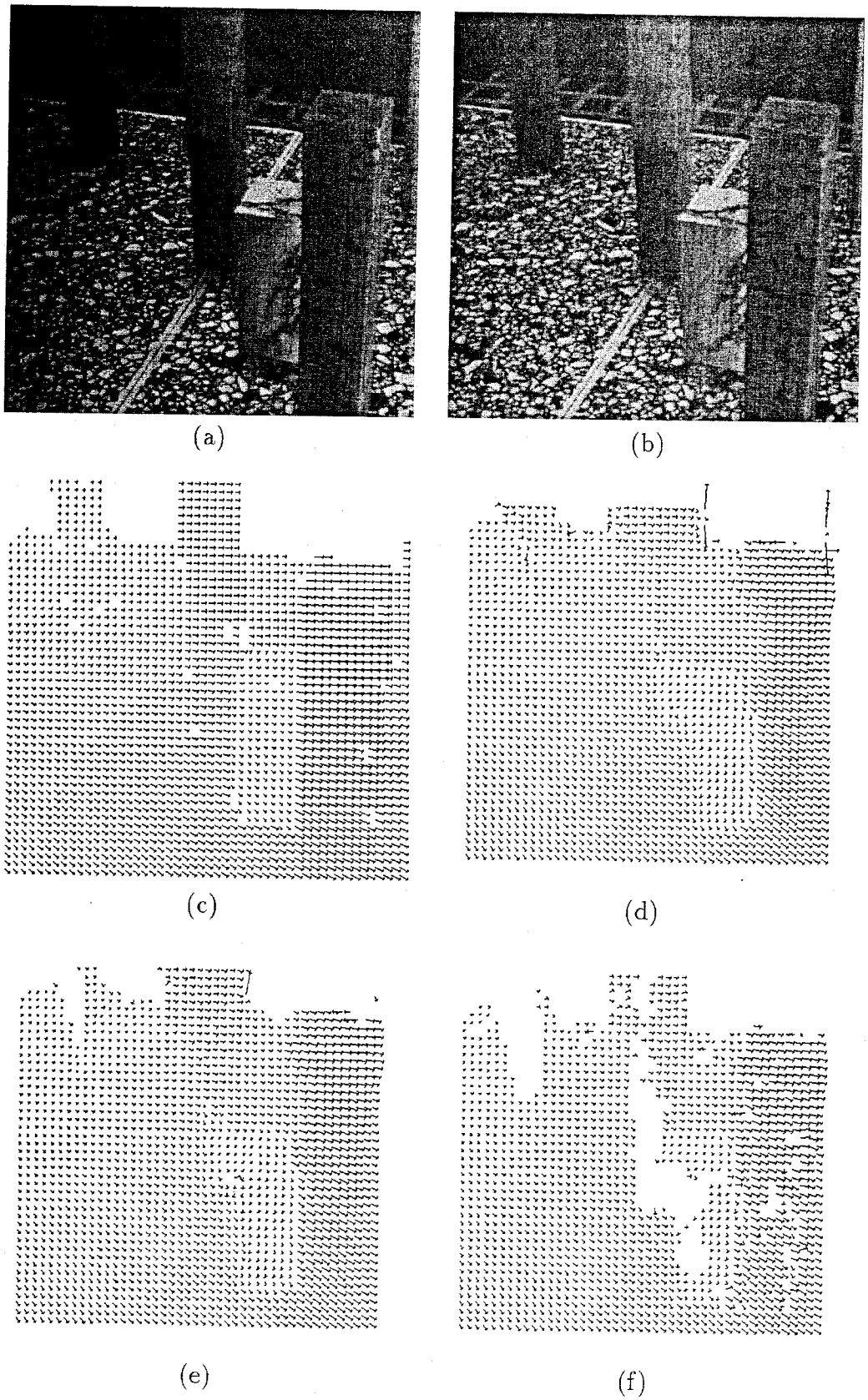

Fig. 2. (a), (b) Images number 40 and 43 of Otte and Nagel's ground-truth motion sequence. (c) True optical flow field given by Otte and Nagel's ground-truth data (sampled every 10 pixels). (d)–(f) Optical flow field computed using phase information (Gaussian weighting function with 50% weight at $3w/8$, $2w/8$, $w/8$ pixels from the window centre, respectively; w, the window size, equals 64 pixels).

Note that all of the results presented in this paper are the unprocessed output of the algorithm (apart from interpolation); each velocity vector has been estimated independently, and the vector field has not been subjected to median or mean filtering.

4. Discussion

The results demonstrate that the approach described produces a dense, accurate, repeatable and reasonably complete flow field. However, in the case of the flow field produced with 'wider' weighting functions, i.e. the Gaussians with standard deviation $\sigma 3w/8$ and $\sigma 2w/8$, the velocity represents an average or aggregate velocity in the windows. Normally, this is not a problem, especially where the velocity profile in the window is either constant or varying linearly, since the average will represent a good estimate at the centre of the window over which the estimate is taken. On the other hand, if there is a discontinuity in the velocity profile in the window, such as is the case where there exist two or more objects moving in the window, then this estimate will be inaccurate and will represent an aggregation of all the velocities. The reason for this becomes clear when we reflect on the manner in which we are computing the velocity estimate, i.e. by computing the phase changes of the Fourier frequency components. We have made the tacit assumption that there is just one single object moving in the image or, equivalently, that the image function translates with unique and uniform velocity. Thus, the Fourier components all exhibit the (frequency-dependent) phase change associated with this velocity. When there are two velocities, the Fourier component is the resultant of the two individual components of each object (assuming common spectral

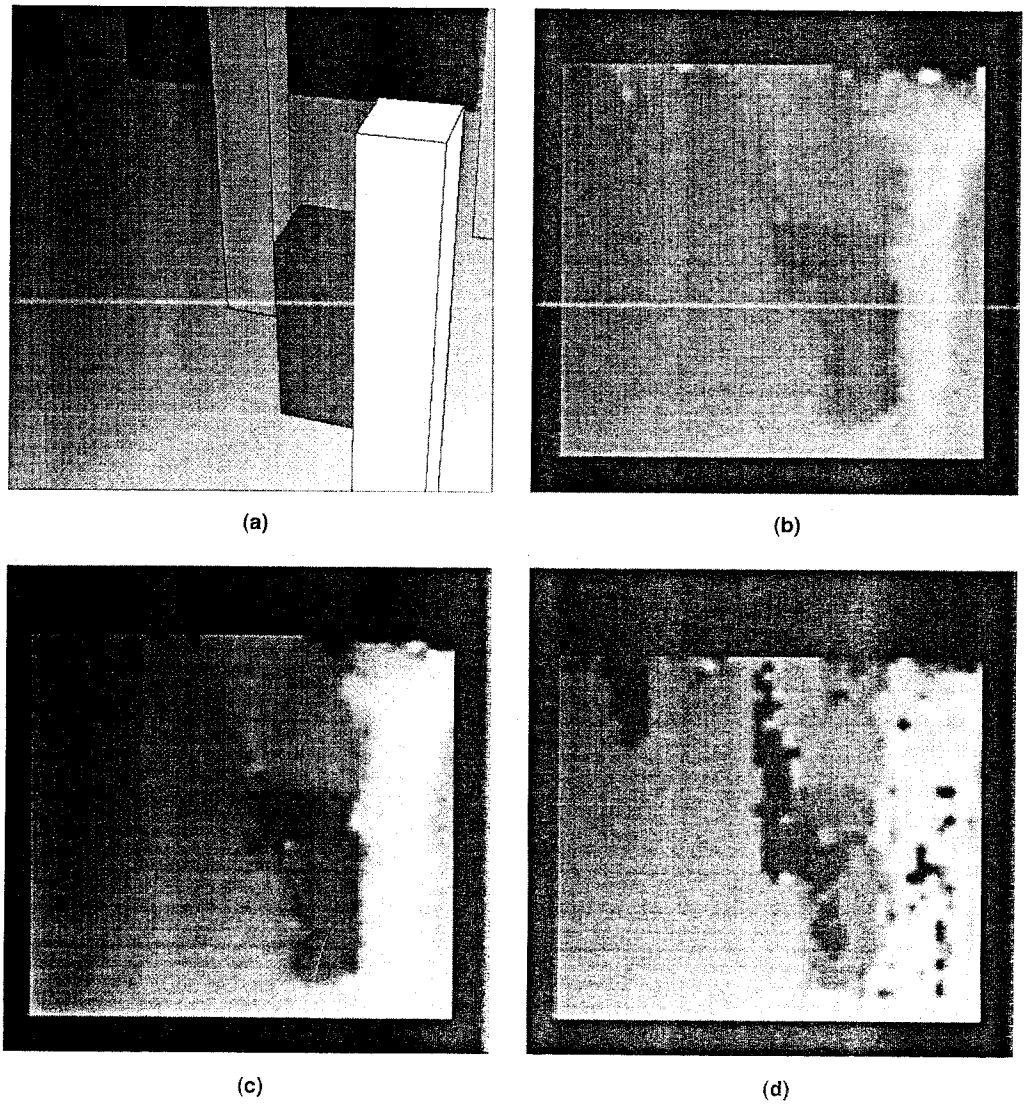

Fig. 3. (a) Magnitude of the optical flow field extracted from Otte and Nagel's ground truth data. (b)–(d) Magnitude of the optical flow field computed using phase information (Gaussian weighting function with 50% weight at $3w/8$, $2w/8$, $w/8$ pixels from window centre, respectively; w, the window size, equals 64 pixels).

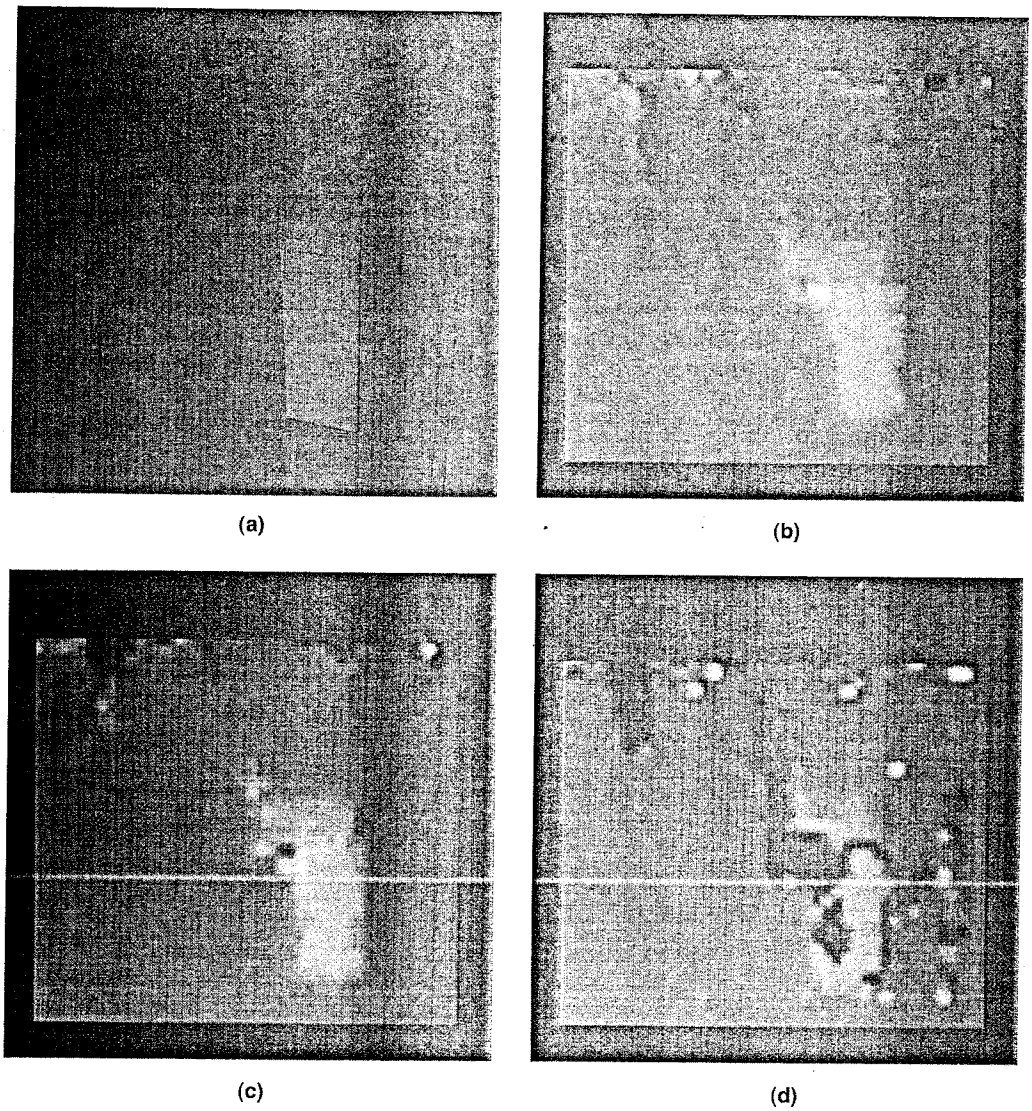

Fig. 4. (a) Direction of the optical flow field extracted from Otte and Nagel's ground-truth data. (b)–(d) Magnitude of the optical flow field computed using phase information (Gaussian weighting function with 50% weight at $3w/8$, $2w/8$, $w/8$ pixels from the window centre, respectively; w, the window size, equals 64 pixels).

support), and the technique described in this paper computes the velocity based on the phase change of the resultant, and not of the individual components. Since the phase change of the resultant Fourier component will depend on the phase changes of the individual components and their magnitude, the computed velocity will be some aggregate of the velocities of the two components. If the magnitude of the components are equal, the phase change of the resultant will be the average of the phase change of the individual components.

As the window in which the velocity is being estimated encounters, and crosses, a velocity discontinuity, as in the case of an occluding boundary, the computed velocity will change from a true estimate of the velocity of the first object to a true estimate of the velocity of the second object, passing through estimates of the aggregate velocity of the two. This is particularly evident in the velocity fields associated with the 'wider' weighting functions, $\sigma 2w/8$ and $\sigma 3w/8$ [see Fig. 2 (d) and (e)], since they incorporate more components from both objects as the window passes over them. The flow-field associated with the $\sigma w/8$ weighting function represents an attempt to reduce the effective support, computing just one velocity. However, the reduced support introduces its own problems. The velocity estimate is more prone to produce spurious results and, given its reduced 'window of visibility', it encounters problems in identifying a velocity in regions where the image function is approximately constant and, hence, there is no detectable phase change [see Fig. 2(f)].

We can overcome this somewhat by reducing the threshold of the frequency magnitude which a component must exhibit in order to be used in the computation. Although this does result in a more complete flow field, these values tend not to be quite as reliable.

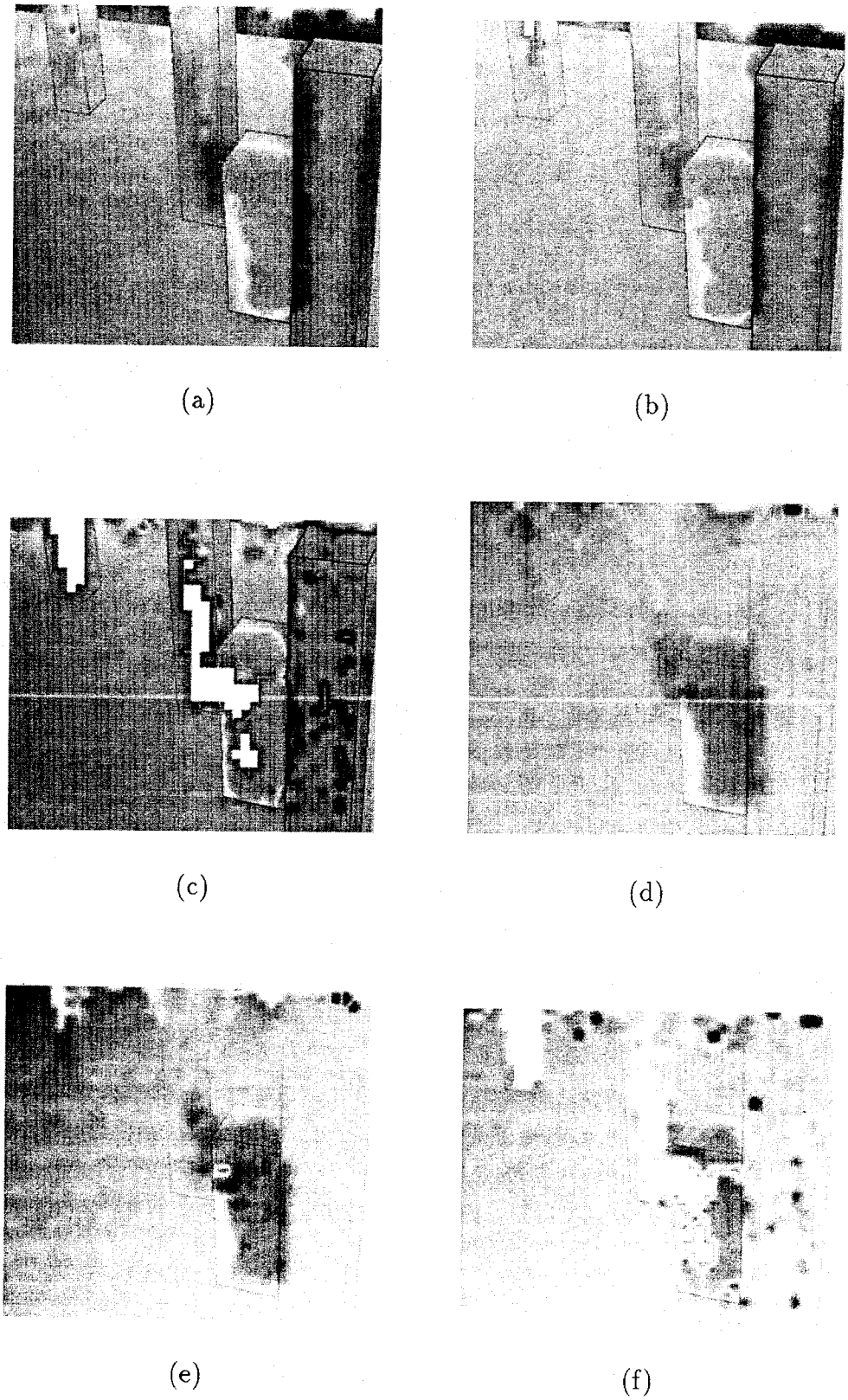

Fig. 5. (a)–(c) Absolute difference between computed and ground-truth magnitude values, (d)–(f) between computed and ground-truth direction values (Gausssian weighting function with 50% weight at $3w/8$, $2w/8$, $w/8$ pixels from the window centre, respectively; w, the window size, equals 64 pixels). Note darkness is proportional to error.

The proper solution to this problem would be to identify explicitly the two (or more) velocities of those objects translating in the window and then to assign the appropriate velocity to that being estimated in the window. This requires that the resultant Fourier component and its phase change, which forms the basis of the velocity computation, be decomposed into its constituent components, where each component corresponds directly to a distinct object, and these components and their phase changes then be used to identify the velocity.

What of the accuracy of the technique? The accuracy is dependent on the resolution of the Hough accumulator since it effectively samples the velocity space. We have adopted, somewhat arbitrarily, a velocity sampling period of 0.1 pixel (i.e. the interpixel-distance in the velocity Hough space is equivalent to 0.1 pixels/frame). Fig. 1(e) and (f), together with Tables 1 and 2, show clearly that the technique is indeed capable of consistent computation of the flow to within this tolerance, at least for the trivial flow field shown in Fig. 1. For a more demanding assessment of the technique, we have used Otte and Nagel's [2] benchmark sequence. This sequence has the major benefit that ground truth optical flow is available [i.e. the magnitude and direction of the optical flow of (almost) every point in the image]. To compare the optical flow computed with the algorithm presented in this paper and ground-truth, the optical flow was estimated every 10 pixels (for the three Gaussian weighting functions), and then a complete optical flow image was produced for both magnitude and direction by interpolating bi-linearly among these points (see Figs 3 and 4). These were then compared to the ground-truth magnitude and direction images by estimating the RMS error and the maximum error (see Table 2). In addition, the point-by-point difference between the magnitude and direction of the computed optical flow and the ground-truth optical flow is shown in Fig. 5.

Finally, the mean and standard deviation of the magnitude and direction of the ground-truth flow field and the three computed flow fields are given in Table 1.

Referring to these images and tables, a number of points can be noted.

First, it is clear that the main errors occur, as one would expect bearing in mind the discussion above, at the occluding contours and, in particular, at the contour where the two objects are moving with significant velocities (as, e.g. in the case with the white block and large dark block in the foreground). Again, as expected, this error is greater for the wider weighting functions and, because the velocity estimate is based on a larger effective support, the error propagates into a bigger region around the occluding contour.

Second, the mean magnitudes and directions of the three computed flow fields are consistent and do not vary significantly (1.171, 1.207 and 1.157 pixels, and 0.597, 0.588, 0.579 radians, for mean magnitudes and directions, respectively). However, they do differ from the ground-truth mean magnitude and direction values of 1.586 and 0.318, respectively. Interestingly, the standard deviations of all four fields are reasonably consistent. Clearly, there is a bias in the measurement (assuming, as we must, that the ground-truth data are correct). This apparent bias is evident in the vector-field [compare Fig. 2 (d), (e) and (f) with (c)] and shows up in the RMS error estimates. On the other hand, it also has to be said that the algorithm reliably and consistently produces the correct magnitude and direction (to better than 0.1 pixel) when tested on artificial test images.

5. Conclusions

Notwithstanding the apparent bias of the results of the approach when tested with Otte and Nagel's ground truth data, the technique presented produces dense, consistent and accurate instantaneous optical flow fields. The use of a Gaussian weighting function which provides a 50% weighting at $w/4$ pixels from the centre of the window in which the velocity estimate is being computed provides a good compromise between distortion introduced due to edge-effects and the inability to compute an estimate due to the small support of the estimate.

The major problem of the approach is that it produces an aggregate velocity estimate in regions comprising objects with two distinct velocities (e.g. in the local region around occluding contours). A solution to this problem is the subject of current research.

References

[1] J.L. Barron, D.J. Fleet, S. Beauchemin, Performance of optical flow techniques, International Journal of Computer Vision 12 (1) (1994) 43–77.

[2] M. Otte and H.-H. Nagel, Optical flow estimation: advances and comparisons, in: J.O. Eklundh (ed.), Lecture Notes in Computer Science, Computer Vision—ECCV '94, Springer, Berlin, 1994, pp. 51–60.

[3] J.H. Duncan, T.-C. Chou, On the detection and the computation of optical flow, IEEE Trans. Pattern Analysis and Machine Intelligence 14 (3) (1992) 346–352.

[4] M. Tistarelli, Multiple constraints for optical flow, in: J.O. Eklundh (ed.), Lecture Notes in Computer Science, Computer Vision—ECCV '94, Springer, Berlin, 1994, pp. 61–70.

[5] L. Jacobson, H. Wechsler, Derivation of optical flow using a spatio-temporal-frequency approach, Computer Vision, Graphics, and Image Processing 38 (1987) 29–65.

[6] M.P. Cagigal, L. Vega, P. Prieto, Object movement characterization from low-light-level images, Optical Engineering 33 (8) (1994) 2810–2812.

[7] M.P. Cagigal, L. Vega, P. Prieto, Movement characterization with the spatiotemporal Fourier transform of low-light-level images, Applied Optics 34 (11) (1995) 1769–1774.

[8] S.A. Mahmoud, M.S. Afifi, R.J. Green, Recognition and velocity computation of large moving objects in images, IEEE Trans. Acoustics, Speech, and Signal Processing 36 (11) (1988) 1790–1791.

[9] S.A. Mahmoud, A new technique for velocity estimation of large moving objects, IEEE Trans. Signal Processing 39 (3) (1991) 741–743.

[10] S.A. Rajala, A.N. Riddle, W.E. Snyder, Application of one-dimensional Fourier transform for tracking moving objects in noisy environments, Computer Vision, Graphics, and Image Processing 21 (1983) 280–293.

[11] D.J. Fleet, A.D. Jepson, Hierarchical construction of orientation and velocity selective filters, IEEE Trans. Pattern Analysis and Machine Intelligence 11 (3) (1989) 315–325.

[12] D.J. Fleet, A.D. Jepson, Computation of component image velocity from local phase information, International Journal of Computer Vision 5 (1990) 77–104.

[13] D.J. Fleet, A.D. Jepson, Stability in phase information, IEEE Trans. Pattern Analysis and Machine Intelligence 15 (12) (1993) 1253–1268.

[14] A. Rosenfeld and A. Kak, Digital Picture Processing, Volume 1, Academic Press, New York, 1982, p. 12.