Coarse Image Motion for Saccade Control

Philip F. McLauchlan, Ian Reid and David W. Murray
Robotics Research Group,
Department of Engineering Science, University of Oxford,
Oxford, U.K.

Abstract

We describe a 2D vision module that estimates the motion of moving objects for the purpose of driving saccadic head/eye motions to fixate them. Robustness and a fast reaction time are the main requirements of the module. The apparently moving background is segmented from the moving objects in the scene using a prediction of the background flow obtained from head odometry. Subsequently the velocities of the detected objects are determined using a least-square method to solve the aperture problem. The algorithm has been implemented at frame rate (25Hz) on a network of five transputers, and has a latency of approximately 0.06s.

1 Introduction

We are developing a reactive vision system to investigate real-time gaze control. The mount consists of a 4 degree-of-freedom stereo head (pan, elevation, independent vergence), of which elevation and vergence for one camera are currently implemented. Given the fast reaction times required of a useful gaze control system, we have required high quality engineering of the mount to supply the speed and precision required, with real time control and vision systems. The latter is made up of a combination of DataCube pipelined image processing boards and Transputers, with high speed links for communication with the mount. Details of the head design can be found in [1], the overall structure and aims of the project are described in [2], and details of the information processing system are given in [3].

This paper describes the implementation of a 2D vision module that will supply signals to drive saccadic camera motions for the purpose of obtaining fixation on a moving object. The image motion will be used to compensate for the inevitable processing and head motion delay by predicting the new position of the object. We wish to estimate large motions (up to 15 pixels per frame, corresponding to $30^o\ s^{-1}$ for the cameras we currently use) for this purpose. Given the inertia of the head we require motion to be computed with a latency of about 0.1s. Other modules will maintain fixation on (track) the object of interest, and our initial experiments with the head will investigate the interaction of these modules in conjunction with position and velocity head control.

Nelson [4] has described an algorithm designed to solve the background/moving object segmentation problem (but not the object velocity estimation problem) for the translating camera case as well as the case of rotation. Burt and his co-workers [5] have developed a multi-scale pyramidal motion segmentation algorithm designed for use in conjunction with control of the sensor and the parameters of the algorithm. François and Bouthemy [6] have designed an algorithm that uses qualitative information about the camera motion to aid motion segmentation, using a Markov Random Field (MRF) approach to segment the scene into regions with common affine flow field.

Our approach differs from the previous work in two main ways: firstly we must have an algorithm that can recover moving objects at the frame rate of 25Hz and, just as importantly, a latency below 0.1s. This eliminates iterative minimisation approaches like the MRF and the pyramidal multi-scale method of Burt. Secondly, we can take advantage of the precisely known head motion and use quantitative estimates of head rotation to predict the background motion and hence aid segmentation. Areas in the image not in agreement with the background motion can be flagged as being due to independent motion. This is *only* possible with a real-time gaze control system that allows effectively instantaneous determination of head odometry for each image, which explains why this method has not been tried before. On the assumption of zero camera translation and a static background, the image motion is constrained to be independent of the scene geometry. Because the rotation axes of the camera do not coincide with the optical centre, there is a small error which varies inversely with the distance of the object from the camera. The worst case error for an object at 2 metres distance is negligible for the vergence axis, 0.5 pixels for the elevation axis and 1.5 pixels per frame for the pan axis, undetectable by the coarse motion algorithm. Indeed it appears not to be necessary to align elevation and vergences axes precisely with the camera optical centres for the visual tasks we are concerned with.

We use normal image flow estimates directly for the first stage of the algorithm, segmenting the background from the moving objects. The normal flow is calculated from spatial and temporal image derivatives using the motion constraint equation [7]. The flow due to the known camera angular velocity is then "subtracted" from the normal flow vectors in the sense described in section 3. This yields image regions whose non-zero residual motion is incompatible with the background. These are analysed individually on the assumption that their motion can be approximated as a constant flow. A least-squares method is used to find the best-fitting full flow vector to the set of normal flow vectors, and also provides a measure of the error via the RMS residual. Thus the aperture problem is solved within each segmented region. The algorithm is not designed to segment two moving objects that happen to appear at adjacent positions in the image. In that case the motion will be flagged with a large error in the velocity estimate.

We have implemented the algorithm on our head/eye platform at frame rate (25 Hz) with a latency of 0.06s using a network of five transputers plus a transputer frame-grabber. Details of the real-time implementation can be found in [3].

The following sections 2 to 5 describe the various stages of the algorithm in more detail. There follow some results on moving imagery and a discussion of the future development of the project.

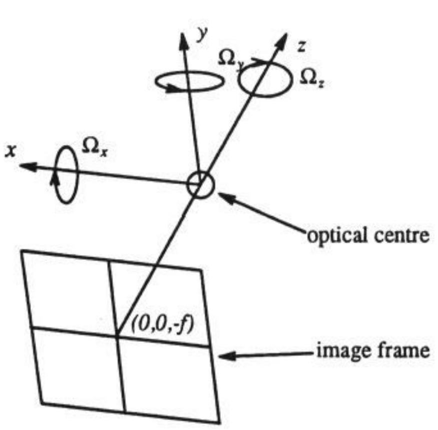

Figure 1: The camera coordinate frame.

2 Calculation of Normal Image Flow

Let the image be $E(x, y)$, x and y being the pixel coordinates. The motion constraint equation [7] is

$$\frac{dE}{dt} = \frac{\partial E}{\partial t} + u\frac{\partial E}{\partial x} + v\frac{\partial E}{\partial y} = 0, \quad \text{where } u = \dot{x},\ v = \dot{y} \tag{1}$$

Calculation of the spatial gradients $\partial E/\partial x$, $\partial E/\partial y$ and temporal gradient $\partial E/\partial t$ employs gaussian convolution and is described in section 5. The image flow $\mathbf{v} = (u\ v)^T$ is approximately the projected motion of the moving object, although Verri and Poggio have shown [8] that equation 1 does not hold exactly, but increases in accuracy as the spatial image gradients increase. We place a threshold on $|\nabla E| = \sqrt{(\partial E/\partial x)^2 + (\partial E/\partial y)^2}$, and only use points where the threshold is exceeded.

Equation 1 is the equation of a line in (u, v) space, corresponding to the set of image flow vectors consistent with equation 1. This *constraint line* is perpendicular to ∇E. The *normal flow* is defined as the vector $\mathbf{v}_\perp = (u_\perp\ v_\perp)^T$ that satisfies equation 1 and is parallel to ∇E:

$$\mathbf{v}_\perp = -\frac{\partial E}{\partial t}\frac{\nabla E}{|\nabla E|^2} \tag{2}$$

$\mathbf{v}_\perp$ is related to $\mathbf{v}$ by the equation $\mathbf{v} \cdot \mathbf{v}_\perp = |\mathbf{v}_\perp|^2$. That we can calculate only the component of the image flow parallel to the gradient is of course a result of the *aperture problem* [9].

3 Segmenting the Background

Let us place a coordinate frame at the camera's optical centre, and assume an ideal pinhole camera model with x, y, z axes as shown in figure 1, the z axis aligned with the principal axis of the lens, and the image origin at the point $(0, 0, -f)^T$. This frame is rotating with angular velocity $\mathbf{\Omega} = (\Omega_x, \Omega_y, \Omega_z)^T$, related to the pan, elevation and vergence angles. This gives rise to motion in the image:

$$\mathbf{u} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \frac{1}{f}\begin{pmatrix} xy\Omega_x - (f^2 + x^2)\Omega_y - fy\Omega_z \\ (f^2 + y^2)\Omega_x - xy\Omega_y + fx\Omega_z \end{pmatrix}. \tag{3}$$

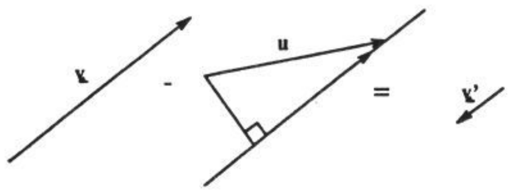

Figure 2: The method of nulling the background motion. Its projection onto the normal flow is subtracted from the normal flow.

We wish to "subtract" the effects of this from the calculated normal flow $\mathbf{v}_\perp$, forming a new normal flow estimate $\mathbf{v}'_\perp$, which will represent the motion of the independent objects. This is done by subtracting from $\mathbf{v}_\perp$ the projection of $\mathbf{u}$ onto $\mathbf{v}_\perp$, as illustrated in figure 2:

$$\mathbf{v}'_\perp = \mathbf{v}_\perp \left(1 - \frac{\mathbf{u} \cdot \mathbf{v}_\perp}{|\mathbf{v}_\perp|^2}\right). \tag{4}$$

In this way we reconstruct the normal flow field that would have arisen if the camera had not been rotating. Combining equations 2 and 4 allows us to simplify the process to the following, computing image flow and background compensation in a single stage:

$$\mathbf{v}'_\perp = -\frac{\nabla E}{|\nabla E|^2}\left(\frac{\partial E}{\partial t} + \mathbf{u} \cdot \nabla E\right). \tag{5}$$

Note that the method can be applied whatever the form of the predicted flow $\mathbf{u}$.

The next step is to decide, on the basis of the new normal flow $\mathbf{v}'_\perp$, which image regions are background and which are not. The simplest method, and the one we use at present, is to threshold $|\mathbf{v}'_\perp|$, i.e. label residual velocities greater than a certain value as due to independently moving objects. In future versions of the algorithm we hope to integrate the results over time to obtain a more robust segmentation.

Lastly, we enforce spatial coherence on the moving objects found. This is done by searching for small square patches in the image dominated by non-background flow vectors. The patches are arranged in two sets each of which covers the image as shown in figure 3. Thus diagonally adjacent patches overlap over a quarter of their area, while laterally adjacent patches do not overlap. Then, if the total number of vectors in a patch is n_{total}, and the number of non-background vectors is $n_{\text{non-back}}$, we label a patch as non-background if:

1. $n_{\text{total}} > T1$, and
2. $n_{\text{non-back}}/n_{\text{total}} > T2$

where $T1$ and $T2$ are constant thresholds.

Adjacent non-background patches are then connected on the assumption that they are part of the same moving object. A region-growing procedure then finds all distinct sets of mutually connected patches. These are the image regions corresponding to the moving objects.

4 Calculating the Independent Motion

We assume that a moving object will give rise to a spatially constant image flow. The algorithm will fail to deliver accurate velocity information if:

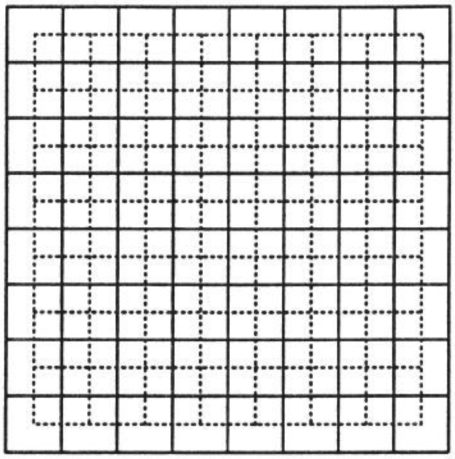

Figure 3: The arrangement of the square image patches. One set is drawn in solid lines, the other dotted.

1. The visible part of the object has a large extent in depth (z).
2. The component of motion perpendicular to the image plane is large.
3. The object is rotating significantly.
4. The velocity of the object is too great (see below).

In these cases the algorithm will only be able to label the regions as moving, without any specific velocity, essentially flagging a region of non-zero image difference [10]. We do not think that this is a drawback of the method, which is intended as a crude but robust way of labelling moving regions in an image; complex motions can be recovered after the moving region is stabilised in the centre of the image. We feel that robust, real-time algorithms should have simple, specific tasks to perform, and be able to report when they need help. We have developed a separate module that detects strong flow divergence [11] which is an "alarm" cue that an object may be on collision course with the cameras and hence deals with case 2 in the above list. We use a generalisation of the current method using a linear flow field approximation, as suggested by Campani and Verri [12] as a good approximation for the flow field generated by a moving planar surface.

Finding the Full Motion

Given regions of non-background normal flow vectors we wish to find the image velocities of moving objects. This involves solving the aperture problem, which until this stage we have been able to avoid. For each labelled moving object we find a single full flow vector $\mathbf{v}$ which is closest, in a least-squares sense, to the constraint lines generated by the background-compensated normal flow vectors (see equation 5) making up the object. The distance between the constraint line represented by $\mathbf{v}'_\perp$ and $\mathbf{v}$ is $(1 - \mathbf{v} \cdot \mathbf{v}'_\perp / |\mathbf{v}'_\perp|^2)|\mathbf{v}'_\perp|$. Thus we minimise the expression

$$\sum_{i=1}^{N} \left(1 - \frac{\mathbf{v} \cdot \mathbf{v}'_{\perp i}}{|\mathbf{v}'_{\perp i}|^2}\right)^2 |\mathbf{v}'_{\perp i}|^2 \tag{6}$$

in $\mathbf{v} = (u, v)^T$ where N is the number of flow vectors and $\mathbf{v}'_{\perp i}$ is the value of $\mathbf{v}'_{\perp}$ for the ith point. The result is a pair of simultaneous equations from which u and v can be obtained as:

$$u = \frac{S_y S_{xy} - S_x S_{yy}}{S_{xy}^2 - S_{xx} S_{yy}}, \quad v = \frac{S_x S_{xy} - S_y S_{xx}}{S_{xy}^2 - S_{xx} S_{yy}}$$

where $S_x = \sum v'_{\perp ix}$, $S_y = \sum v'_{\perp iy}$, $S_{xx} = \sum v'^2_{\perp ix}/|\mathbf{v}'_{\perp i}|^2$, $S_{xy} = \sum v'_{\perp ix} v'_{\perp iy}/|\mathbf{v}'_{\perp i}|^2$ and $S_{yy} = \sum v'^2_{\perp iy}/|\mathbf{v}'_{\perp i}|^2$. The root-mean-square residual is then

$$\text{RMS residual} = \frac{1}{\sqrt{N}} \left[S_{xy2} + \frac{S_x^2 S_{yy} - 2 S_x S_y S_{xy} + S_y^2 S_{xx}}{S_{xy}^2 - S_{xx} S_{yy}} \right]^{1/2} \quad \text{pixels}$$

where $S_{xy2} = \sum |\mathbf{v}'_{\perp i}|^2$.

5 Implementation

We use a 512×256 pixel field of an interlaced frame subsampled to 64×32. All subsequent processing takes place at the coarse resolution. The subsampled images are smoothed using a gaussian convolution. The temporal image gradient $\partial E/\partial t$ is then simply the difference between the current and previous images, and the spatial gradients $\partial E/\partial x$ and $\partial E/\partial y$ are calculated by applying the gradient masks $[-1\ 0\ 1]$ and $[-1\ 0\ 1]^T$ to the average of the previous and current smoothed images, as in [4]. The parameter and threshold values used were:

1. Standard deviation of gaussian convolution: 1.5 pixels. The larger this value the larger the velocities that can be measured. This current value gives us the ability to measure velocities up to about 12 pixels per frame (1.5×8). Larger masks would allow us to measure greater velocities, but at greater computational cost.
2. Threshold on $|\nabla E|$: 4 grey levels per pixel.
3. Threshold on residual velocity $|\mathbf{v}'_{\perp}|$: 0.3 pixels per frame.
4. Segmentation patch size: 8×8 pixels. This specifies the smallest size of moving object that can be located by the method.
5. Threshold on total patch vectors $T1$: 20.
6. Proportion threshold on non-background to total vectors ratio $T2$: 0.7. Thus 70% of a patch must be non-background for itself to be labelled as non-background.

The transputer implementation runs at frame-rate (25Hz) on the even field of each interlaced image, and the latency (pipeline delay) is 0.06 sec.

6 Results

Initial tests were made using a camera carried on a robot arm. The camera was rotated about its x-axis by known amounts between frames, while an object, a white head, was moved to the right. The problem is to recover the motion of the object in the apparently moving background. The first two frames in the sequence at 512×512 pixel resolution are shown in figure 4a. They may be fused stereoscopically.

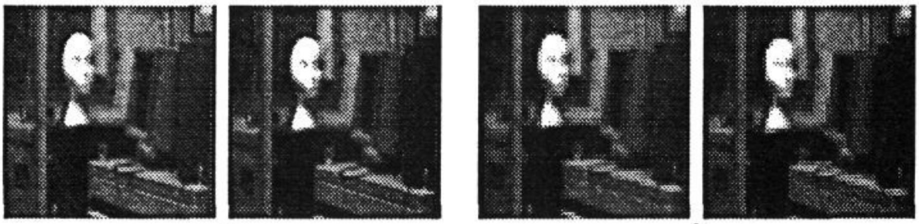
a b

Figure 4: a) The first two frames of an image sequence. b) The same frames after subsampling.

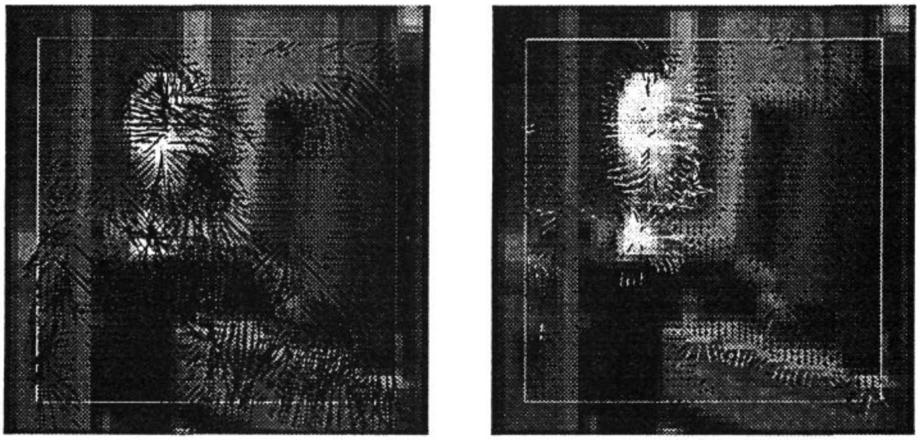
a b

Figure 5: a) Normal image flow vectors. b) The result of subtracting the background flow.

When block-averaged and subsampled down to 64×64 pixels the result is as shown in figure 4b. From these the normal flow vectors are calculated. The vectors in figure 5a found from the images in figure 4b are displayed six times their actual length to aid visibility.

To predict the background flow in this simple case we use a small field of view approximation to equation 3, justified since the field of view of the camera is only 20^o. The quadratic terms in x and y disappear. Ω_y and Ω_z are both zero, so we have

$$\mathbf{u} = \begin{pmatrix} 0 \\ f\Omega_x \end{pmatrix}$$

Ω_x is the known x-axis rotation, and f is known to be approximately $25mm$. Thus the predicted flow $\mathbf{u}$ is a constant vector (w.r.t. x and y) in the y-direction.

The result of subtracting $\mathbf{u}$ from the normal flow vectors is shown in figure 5b. The background vectors are clearly smaller than those in figure 5a. while those due to the moving object have remained about the same size. The vectors shown as black are those that have been labelled as background according to the $|\mathbf{v}'_\perp|$ threshold criterion (page 4). The vectors labelled as part of a moving object are shown white.

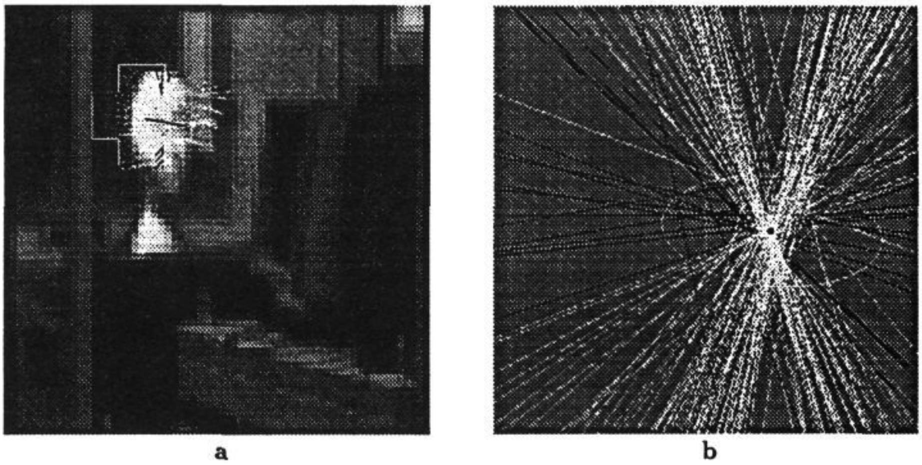

Figure 6: a) The result of segmenting the motion field. b) The least-squares fit to the constraint lines.

Figure 6a illustrates the flow segmentation and the results of the least-square velocity fit. Four adjacent square patches were found to be non-background and these were connected and the least-squares fit method described in section 4 applied to the non-background (white) normal flow vectors. The long black vector, displayed at ten times its actual length, is the estimated object velocity, correctly found to be a sideways motion. In figure 6b are the constraint lines (black for background vectors, white for object vectors) with the black dot at the best fit velocity. The ellipse gives an indication of the fit error and the amount of anisotropy in the fit, given by the ellipse eccentricity. If r is the RMS residual of the least-squares fit then the major and minor axes of the ellipse are set to ϵr and $\frac{r}{\epsilon}$, where ϵ is such that the ellipse takes on the shape of the cross-section through the quadratic sum-of-squares function in equation 6. Thus the ellipse shows the goodness of fit and how it changes with orientation in (u, v) space, allowing the common case of the motion being constrained mainly in one direction (for instance due to a moving straight edge) to be detected as a large major axis, indicating the lack of constraint in that direction. The ellipse is magnified 8 times to aid visibility.

Results for the Head/Eye Platform

The second sequence was produced by the real-time transputer implementation. Shown in figure 7 are 16 frames from a sequence showing two people walking past each other. There was no camera motion in this case. We used only the even field of each frame to avoid motion effects of interlacing. The outline of each detected moving region is shown along with the velocity vector and the error ellipse, both magnified six times. The sequence shows that the people are initially detected as one moving region, the nearer person moving to the right dominating the result (frames 1-4), while as they separate the estimated error increases greatly since there is an equal amount of image data travelling in opposite directions (frames 5-8). At that point the algorithm separates the two people and the velocities are estimated with greater subjective precision, i.e. the algorithm "knows" when it has good velocity data. The real-time implementation allows us to obtain such sequences routinely.

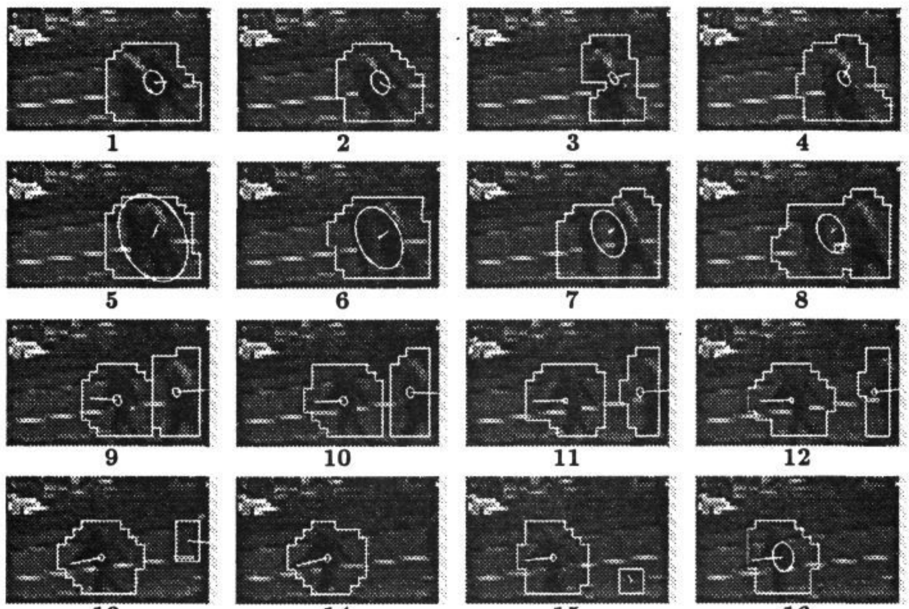

Figure 7: Results for real-time sequence of two people passing on a pavement next to the road outside out laboratory.

7 Conclusions

The algorithm we have developed can quite robustly calculate the position, extent and velocity of isolated moving objects against a stationary background in the presence of known head rotation, in real time (25Hz) and with short latency (0.06s). As it stands this will allow us to perform experiments with the head. We need to analyse the accuracy of the algorithm for different velocities and types of object. The basic test that the algorithm must pass is to be accurate enough that using the position and velocity data it provides the head will be able to saccade, fixate on the object, and start tracking it in a small foveal window. We are currently working on a Kalman filter-based tracking module that will allow the coherence of the temporal image to be made explicit, and to determine the trajectory of moving objects. With this module we wish to maintain an updateable memory of the moving objects in the scene, so that while tracking one object knowledge is retained of others, enabling change of attention if an another object becomes more "interesting" according to the size, velocity, persistence, etc. of the object. We have also extended the method to deal with more general types of object motion. For instance, an object moving towards the cameras gives rise to a strongly divergent flow field. This divergence can be used as an "alarm" signal that the object is about to hit the camera. The time-to-contact can be determined from the divergence (see [13]). Using a linear flow approximation [12] enables us to calculate the flow divergence. This work will be reported in future pulications.

This algorithm should be seen in context as a single module of a complex vision system, much of which will be bootstrapped from the results of this module. As such, robustness has been of greatest importance, and given higher priority than the ability to deal with complex scenes. The important outcome in the context of a real-time gaze control system is not the "perfection" of the flow vectors, but the number of correct *actions* that the module gives rise to.

Acknowledgements

This work was supported by Esprit (Project 5390: Real Time Gaze Control) and the SERC (grant number GR/G30003). We have had fruitful discussions with our RTGC partners, especially Paul Sharkey here at Oxford, and Donald Weir and Nigel Gent at Hirst Research Centre. The work has benefitted greatly from discussions within the Oxford Robotics group, and we would especially like to thank Phil Torr, Paul Beardsley, Rupert Curwen, Andrew Zisserman and Mike Brady.

References

[1] P.M. Sharkey, I.D. Reid, P.F. McLauchlan, and D.W. Murray. Real-time control of an active stereo head/eye platform. In *Proc. 2nd ICARCV, Singapore*, 1992.

[2] D.W. Murray, F. Du, P.F. Mclauchlan, I.D. Reid, P.M. Sharkey, and M. Brady. Design of stereo heads. In A. Blake and A. Yuille, editors, *Active vision*. MIT Press, 1992.

[3] I.R. Reid, P.M. Sharkey, P.F. McLauchlan, and D.W. Murray. A modular head-eye platform for real-time reactive vision. Technical Report 1941/92, Dept. of Engineering Science, Oxford University, 1992.

[4] R.C. Nelson. Qualitative detection of motion by a moving observer. *IJCV*, 7(1):33–46, 1991.

[5] P.J. Burt. Image motion analysis made simple and fast, one component at a time. In *Proc. 2nd BMVC*, pages 1–8, 1991.

[6] E. François and P. Bouthemy. Multiframe-based identification of mobile components of a scene with a moving camera. In *Proc. CVPR*, 1991.

[7] B.K.P. Horn. *Robot vision*. MIT Press, 1986.

[8] A. Verri and T. Poggio. Against quantitative optical flow. In *Proc. 1st ICCV*, pages 171–180, 1987.

[9] E. Hildreth. *The Measurement of Visual Motion*. MIT Press, 1984.

[10] Y.Z. Hsu, H. Nagel, and G. Rekers. New likelihood test methods for change detection in image sequences. *CVGIP*, 26:73–106, 1982.

[11] J.J. Koenderinck. Optic flow. *Vision Res.*, 26(1):161–180, 1986.

[12] M. Campani and A. Verri. Computing optical flow from an overconstrained system of linear algebraic equations. In *Proc. 3rd ICCV*, pages 22–26, 1990.

[13] D.N. Lee. The optic flow field: the foundation of vision. *Phil. Trans. R. Soc. Lond.*, 290, 1980.