Detection and tracking of independent motion

J. C. Clarke and A. Zisserman
Department of Engineering Science, University of Oxford,
Parks Road, Oxford, OX1 3PJ, UK
email: [johnc,az]@robots.oxford.ac.uk *

March 13, 1996

Abstract

We describe an efficient method of using a translating camera to detect and track independently translating objects and assess the likelihood of a collision. By analysing the underlying geometry it is shown that the tracking is reduced to two independent linear searches for a single feature in the image plane. Results are presented for both an off-line and a real time implementation using no special hardware. The method is completely automatic and shown to be accurate and robust.

1 Introduction

Any autonomous vehicle must be able to detect and avoid other moving objects. Previous work on the detection of independent motion has tended to combine the (computationally expensive) optical flow field with a ground plane assumption (Enkelman [5], Carlsson and Eklundh [3]) or weak geometric constraints (Nelson [11], Darrell and Pentland [4], Smith [13]). With the exception of Irani *et al.* [9] this work has ignored the benefits of tracking moving objects to improve the segmentation. In contrast to these optical flow methods Torr and Murray [14], Sinclair and Boufama [12] use the epipolar geometry to detect independent motion; we adopt this use of the rigidity constraint, and extend their work – in the case where all motions are pure translations – by incorporating tracking of both the background and the independent objects. The task is easily stated: given a sequence of images of a scene composed of rigid objects taken by a translating

*Funding for this work was provided by the EPSRC and Esprit Project 6448 VIVA. We are grateful to Paul Beardsley, Ian Reid and David Murray for helpful discussions and to Paul Beardsley for the use of his software.

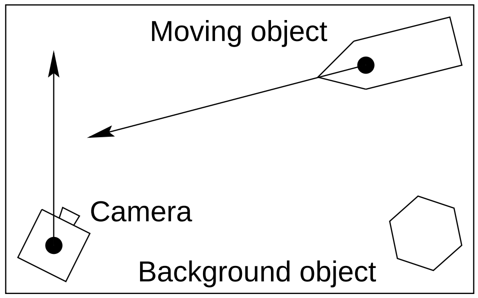


Figure 1: *The motion of the camera and the independent object is assumed to be a translation, but their directions need not be perpendicular or even in the same plane. Both are also free to vary their speed.*

camera (figure 1) detect and track any independently translating objects. If any are found, determine whether or not a collision will ensue. This paper shows how to do this accurately, automatically and robustly without requiring camera calibration. Although the method presented uses only image corners the theory can also be expressed as a special case of the trifocal constraint (Hartley [8], Torr *et al.* [15]) thus allowing the possibility of incorporating line segments.

The three dimensional geometry, and image projection of features on the background and moving objects, are described in sections 2 and 3. Next, section 4 gives a simple and robust test for imminent collisions. Finally sections 5 and 6 describe a complete implementation and a highly efficient frame rate feature tracker respectively. Some extensions of this work are suggested in the conclusion.

2 Model of the background motion

We are principally concerned with tracking points through a sequence of images. The problem of using a point's position in one set of images to predict its whereabouts in another is known as transfer; to perform the transfer we must complete three tasks:

Segmentation to determine which features are consistent with the motion of a single rigid object.

Initialisation to determine the invariants used to track an object.

Tracking to determine the current position of the object.

We shall show that for a pure translation given correspondences in two views, only one match in a third view is required to constrain the position of all the other features. Further, this single feature match can be found by a one dimensional search in the third image.

2.1 Algebraic modelling of transfer

The mechanics of image projection can be used to capture the geometric constraints imposed by the assumption of straight line motion, and predict the position of features in future images after a search for a single parameter. Using homogeneous coordinates the world position $\mathbf{X}_f$ of the f^{th} feature projects to its image position $\mathbf{x}_f^i$ at time i as

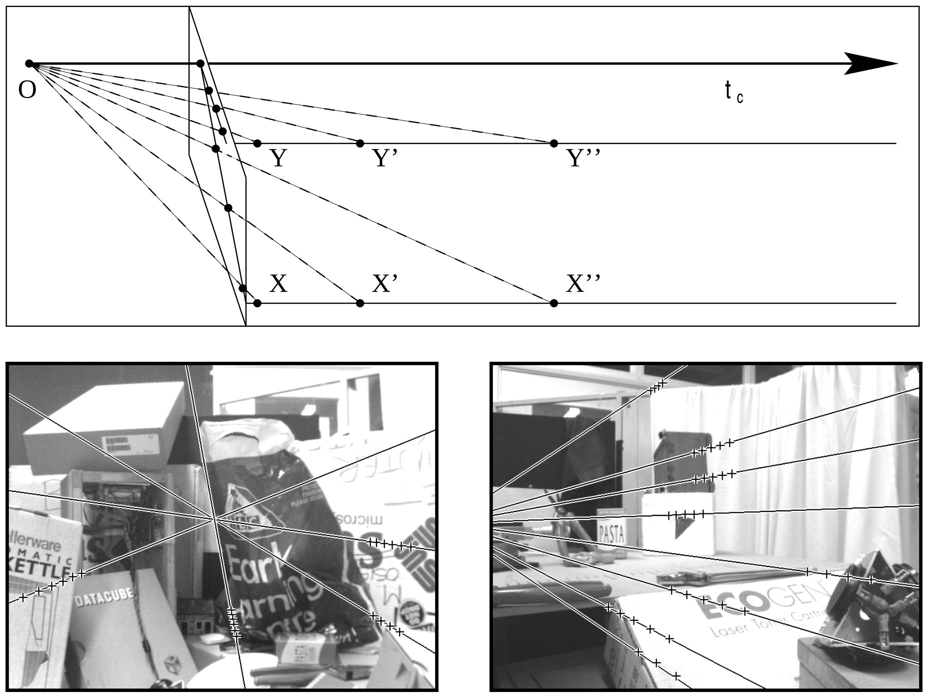
$$\mathbf{x}_f^i = \mathtt{P}^i \mathbf{X}_f \tag{1}$$

Here $\mathbf{x}_f^i$ is a 3 vector, $\mathbf{X}_f$ a 4 vector, and $\mathtt{P}^i$ a 3×4 matrix. Since homogeneous entities are equivalent up to scale $\mathbf{x}_f^i$, $\mathbf{X}_f$, and $\mathtt{P}^i$ have 2, 3, and 11 degrees of freedom respectively. A pair of views of a rigid scene separated in time is equivalent to stereo and so it is possible to find the values of $\mathbf{X}_f$ and $\mathtt{P}^i$ given sufficiently many feature correspondences (Faugeras [6], Armstrong [1]). It can be shown that by combining an uncalibrated camera model and the fact that the camera motion is a pure translation the projection matrix at the i^{th} view may be taken to have the form

$$\mathtt{P}_b^i = [\mathtt{I} | s_b^i \mathbf{e}_b] \tag{2}$$

in which s_b^i is the magnitude of the translation, and $\mathbf{e}_b$ is the background epipole or image plane projection of the direction of motion (figure 2). In this case $\mathbf{X}_f$ will be recovered up to an affinity (Moons *et al.* [10]) allowing the measurement of ratios of distances in parallel directions (in particular s_b^i). For a pure translation the epipole $\mathbf{e}_b$ has a fixed position on the image plane, and can be calculated from the motion of two image features. Once the epipole and all the features' world positions have been calculated only a single new parameter arises at the i^{th} image - the magnitude of translation s_b^i. Clearly s_b^i is determined by a single corner match, and the matching corner must lie on the epipolar line – see figure 2. Once s_b^i is known the position of all other features may be found from equations 1 and 2.

To summarise: **Segmentation:** two point matches determine $\mathbf{e}_b$ which is common to all background features. **Initialisation:** two point matches determine $\mathbf{e}_b$ and affine structure, s_b^0 and s_b^1 may be chosen arbitrarily. **Tracking:** a single point match along a feature's epipolar line determines s_b^i and hence the positions of all background features.



(a) (b)

Figure 2: *The epipolar geometry of a translating camera. Upper figure: as the camera's optical centre O translates along $\mathbf{t}_c$, points X and Y in the world appear to take up new positions (X', X'', ...). The feature's epipolar lines are shown in the image, these are the intersections of the image plane with the planes defined by $\mathbf{t}_c$ and the lines $\bar{O}X$ and $\bar{O}Y$ respectively. Because these epipolar planes always include the line $\mathbf{t}_c$ the epipole $\mathbf{e}_b$ must lie on every epipolar line. The epipole is the vanishing point for all lines parallel to $\mathbf{t}_c$, and remains fixed so long as the direction of $\mathbf{t}_c$ is constant. Lower figures: image sequences obtained as the camera translated towards a set of objects; a number of features at different instants are superimposed on their epipolar lines. In (a) the camera's optical axis is aligned with the direction of translation. In (b) the camera has been rotated to the right forcing the epipole to the left.*

3 Model of the independently translating object(s)

We shall show that given the background motion (i.e. $\mathbf{e}_b$ and s_b^i) independently translating objects have the same complexity as the background – the image projection for all points on the object is determined by a single new parameter at each frame and that parameter may be determined by a single feature match found using a one dimensional search.

As in the case of the background we use an uncalibrated camera and it can be shown that the projection matrix for points on the object may be chosen to have the form

$$\mathtt{P}_o^i = [\mathtt{I} | s_b^i \mathbf{e}_b + s_o^i \mathbf{t}_o] \tag{3}$$

in which $\mathbf{t}_o$ is the image plane projection of the object's direction of motion and s_o^i is the magnitude of that translation at the i^{th} frame. The **epipole** for the moving object lies on the ray $s_b^i \mathbf{e}_b + s_o^i \mathbf{t}_o$ which gives the direction of relative motion between the camera and the object, but unlike the background epipole $s_b^i \mathbf{e}_b + s_o^i \mathbf{t}_o$ varies according to the ratio $s_b^i : s_o^i$. Notice that the "ground-plane constraint" was not invoked i.e. the independent object and the camera may move in different planes.

To summarise: **Segmentation:** two point matches determine $s_b^i \mathbf{e}_b + s_o^i \mathbf{t}_o$ which is common to all points on the object. **Initialisation:** three point matches determine $\mathbf{t}_o$ and affine structure. **Tracking:** given $s_b^i \mathbf{e}_b$ and $\mathbf{t}_o$ the search is along the line parameterised by s_o^i for a single point match which then determines s_o^i and hence the position of all points on the object.

4 Collision prediction

In order to decide whether or not a collision will occur requires a knowledge of the viewer's size, the potential obstacle's size, and the direction of relative motion. In this section we consider the simpler problem of deciding whether or not the camera's optical centre will collide with a moving object, thus obviating any need for knowledge about the viewer's size.

As was shown in section 3 the direction of the combined motion is given by the epipole defined by points on the moving object, and has already been calculated as part of the tracking process. The test is simple: **a collision will occur if the epipole lies inside the image boundary of the moving object.** If the epipole lies inside then the camera's relative motion is toward the object and a collision will eventually occur, conversely if the epipole lies outside then no collision can occur. In our experiments we have used a rectangle to approximate the boundary of the moving object, but other possibilities are the convex hull, and methods incorporating image edges e.g. Smith [13].

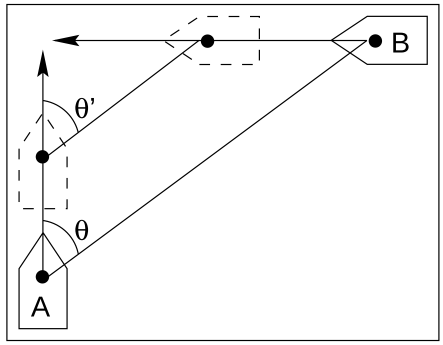


Figure 3: *The sailor's test for a collision. Two ships A and B will collide if the second makes a constant bearing θ as seen from the first, so the condition for a collision is that $\theta = \theta'$. This is a special case of the epipole test for a collision.*

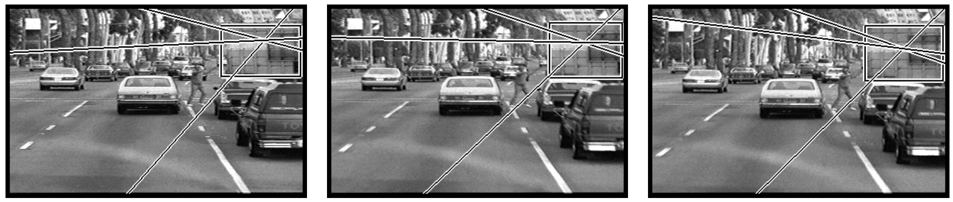
This test can be seen as a generalisation of the well known sailor's test for determining whether two ships will collide. Figure 3 shows how in nautical terms a collision will occur if the second vessel makes a constant bearing as seen from the first. However, notice that the only point on the second vessel which will not appear to diverge (hence changing bearing) is the epipole, and that the epipole will remain fixed only if both the camera and moving object have constant velocities. Figure 4 illustrates the epipole test with images taken from the motion picture *Speed*.

5 Implementation I: Off-line

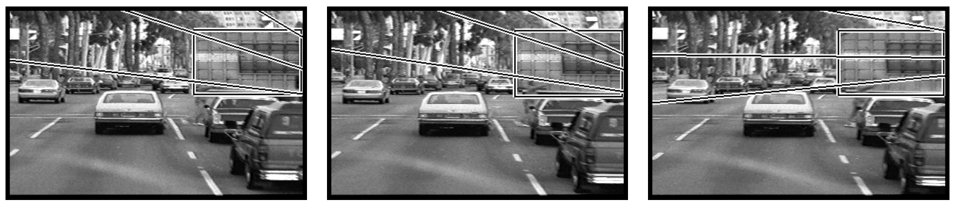
We have developed an off-line implementation of this method for detecting and tracking moving objects by modifying an automated corner matching program developed by Beardsley *et al.* [2]. Image corners are found to sub-pixel accuracy using the Plessey corner detector (Harris and Stephens [7]) and tracked through a sequence of images by matching them to their corresponding corner in the following frame using a two phase process incorporating both pixel value correlations as well as epipolar and structural constraints.

5.1 Robust calculation of the epipole

The assumption of pure translational motion forces the epipolar geometry to have only two degrees of freedom as figure 2 shows, consequently the epipole can be found by intersecting any two features' image plane trajectories. The small number of parameters makes the RANSAC robust minimisation technique a highly efficient way to estimate the epipole in the presence of (the inevitable) mismatched features and moving objects. The RANSAC algorithm for finding



(1a) (1b) (1c)



(2a) (2b) (2c)

Figure 4: *The test for a collision illustrated by six frames from the film* Speed. *The camera moves forward while the truck on the right moves into its path. Each image shows the truck's bounding rectangle and three epipolar lines intersecting at the epipole for the moving object. In (1a,b,c) the epipole lies directly on the object showing that a collision will occur, while in (2a,b,c) the epipole lies behind the object showing that the camera has slowed sufficiently to allow the object to pass by.*

the epipole proceeds by repeatedly using a random sample of two pairs of point matches to determine a putative epipole which is then evaluated for its support from all the feature matches. A point match is deemed to support a potential epipole if the feature's position in both images lies close to its putative epipolar line. The ultimately selected epipole is the one consistent with the most data, the remaining corner matches are ignored. A typical threshold of 1.5 pixels results in $\sim 80\%$ of the corner matches having a common epipole if a moving object is not present.

A linear least squares method improves the estimate of the epipole by finding the best estimate of the common intersection of the feature trajectories. The final minimisation typically reduces the average distance of a corner from its epipolar line from ~ 0.5 to ~ 0.4 pixels.

5.2 Tracking points on the background

The background is considered to be that part of the world which generates the most features on the image plane consistent with a single epipole. Given the background epipole and affine structure, points are matched as in Beardsley *et al.* [2] to provide estimates of s_b^i. As tracking proceeds the background epipole is updated as in section 5.1 and the background features' world structure is continuously refined using a least squares method that minimises the image plane errors. Once tracking is suitably advanced only features that have been tracked for more than a minimum number of frames (typically 4) are used when calculating the epipole. This continual re-estimation of the epipole implicitly tests the assumption of pure translation because the epipole's position should be constant.

5.3 The moving object(s)

Once the extent of the camera motion s_b^i is known there is only a single free parameter in equations 1 and 3 to model the moving object.

Detection: Moving objects are found by attempting to fit an epipole to the feature matches not consistent with the background epipole. If the techniques of section 5.1 find enough features (typically 5 – 10) consistent with a new epipole then an independently moving object is deemed present. If desired the process may be repeated to search for other moving objects until too few point matches remain, or no object is found.

Tracking a moving object: In addition to the background motion, tracking requires that the direction of the independent translation be known; these extra parameters must be found by fitting the model (equation 3) to earlier point matches; otherwise tracking an object is similar to tracking the background.

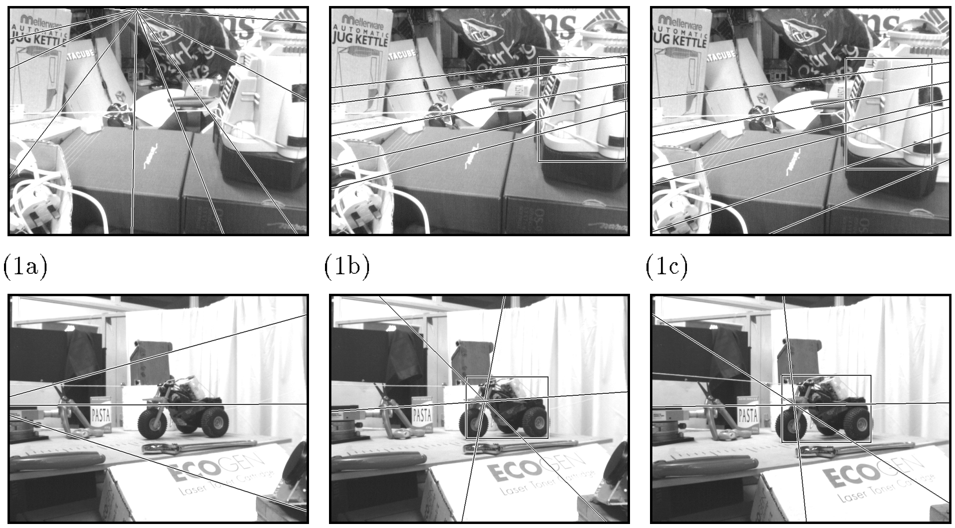
Segmentation of the moving object: The image plane extent of the moving object is defined by the smallest rectangle enclosing all the features that have been updated more than a fixed (typically 4) number of times.

5.4 Experimental results

Figure 5 shows typical results of the method, which produces a good segmentation and an accurate prediction of whether a collision will occur. In each test a small toy (robot or radio controlled buggy) translates towards the path of a translating camera mounted on an Adept industrial robot. The camera's optical axis is not aligned with the translation – in the first sequence the camera points $20°$ down, in the second $30°$ to the right. The algorithm requires 3 – 4 frames to initialise an accurate segmentation. About 120 features on the background and up to ~ 50 on the moving object were tracked.

6 Implementation II: Frame rate

This section describes a real time feature tracker able to detect independent motion operating on a Sun IPX computer equipped with an S2200 frame grabber and a video camera mounted on an Adept industrial robot. The original off-line implementation takes over four seconds to process a single 512x512 image on a



(1a) (1b) (1c)

(2a) (2b) (2c)

Figure 5: *Results from two tests of the off-line implementation. In each case the camera and moving object translate along approximately perpendicular paths. In both sequences image (a) shows the epipolar lines for background points, images (b) and (c) show the bounding rectangle and epipolar lines for features on the object. In images (1b) and (1c) the epipole is far from the object indicating safety, whereas in (2b) and (2c) the epipole is on the object hence there will be a collision; both reflect the ground truth.*

Sun IPX, so running at frame rate requires a speed up by a factor of at least 100. Almost all the processing time used by the off-line version is devoted to the Plessey corner detection algorithm, and so the problem is one of resource allocation; corner detection is slow and must only be performed where it will do the most good. This section shows that real time tracking can be performed on ordinary computers with no loss of performance.

The frame grabber alternately updates one of two 288 line interlaced fields making up a 576 line image. The computer is able to process one field for 20ms while the other is being refreshed. As we have already remarked the corner detection is extremely slow; 20ms is only long enough to detect corners in three 7×7 pixel regions, or one 19×19 pixel region.

6.1 Feature tracking

The key idea is to update tracks only once the corner has moved a significant distance (up to 50 pixels) in the image plane, since the feature's position may be accurately predicted there is little to gain from more frequent updates. The tracker must also divide its resources between maintaining existing tracks and searching for new features to track. Initially there are no tracks and all the time is spent searching for new features, in the steady state one third of the computer's time is devoted to servicing existing tracks.

6.1.1 Track initialisation

Searching for new features: Because we are limited to searching a very small region of the image at each field for new corners those regions must be chosen with care. Unfortunately it is not clear what an optimal sensing strategy would entail, and so we resort to the use of three heuristics in order to choose a 19×19 pixel region of the image plane to search for new corners:

Search the entire image plane. A region is chosen at random from the entire image plane. The probability distribution is biased towards regions where many corners have been found recently.

Search near the background epipole. A region is chosen at random near the background epipole. New features will tend to appear around the epipole, and the probability distribution is based on the current estimate of the background epipole's 95% confidence region in an attempt to reduce the area of the confidence region as quickly as possible.

Search near any moving objects. A region is chosen at random in the vicinity of the current position of a previously detected moving object.

Whenever a full 20ms is available for searching for new corners a heuristic is chosen at random and applied; obviously in the initial state when neither the background epipole or any moving objects are known only the first can be used. If all the heuristics are applicable the typical weightings are 16.7%, 16.7%, 66.6% respectively.

Unguided tracking: If any corner(s) are found that do not match a currently tracked feature then new track(s) are created. The feature's trajectory is estimated initially by searching for the corner at the next frame using a 7×7 pixel window centered on the old position. This limits the maximum speed of features that the initialisation can cope with to 3 pixels $\times$ 25Hz = 75 pixels/second, as compared to the maximum measured in our experiments of ~ 99 pixels/second. Thereafter a constant acceleration Kalman filter is used on the image plane coordinates of the feature until the track can be confirmed.

6.1.2 Track confirmation

When a newly detected feature has moved more than a fixed distance (typically 4 – 8 pixels) it is evaluated to decide whether it corresponds to a background feature, a feature on a moving object, or is merely a spurious result of mismatches. Equations 2 and 3 are used to predict its position under the assumption that it belongs to the background or any particular moving object. If the discrepancy between the track's measured and predicted positions remains less than a threshold (1.5 pixels) it is considered a valid feature, and assigned to the relevant object. In the absence of a moving object approximately 76% of features tracked for up to 8 pixels will be pass the test as a background feature, the remainder are due to mismatches. It is important to delete tracks that can not be assigned to an object relatively quickly in order to prevent them from monopolising the corner detection.

6.1.3 Track maintenance

Each track is periodically updated by applying the corner detector to a 7×7 pixel window centered on the feature's predicted position found from equation 2 or 3. If a corner is found whose surrounding pixels correlate strongly enough with those when the feature was last found and the corner is close enough (within 1.5 pixels) to the feature's epipolar line then the track is updated, and the corner's position is used to update the estimate of the magnitude of the camera's or object's translation. In our work the magnitude of the translation is estimated by a constant velocity Kalman filter, but any filter appropriate to the expected world motion could be used. Tracks are serviced sufficiently frequently that the localisation error should not exceed 1 pixel.

6.1.4 Track deletion

Once a feature fails to be detected it becomes ineligible to be part of the epipole estimation (see below), but the tracker continues to attempt to find it. Only if a feature consistently fails to be detected at its predicted position for more than a set number of attempts (typically 4) is the track abandoned. A track is also removed immediately the feature moves out of the field of view.

6.1.5 Determination of the epipole

The tracker's most urgent task is to determine the background epipole. There are two reasons for this:

- Equations 1 and 2 can not be used to track background features until the epipole is known. The much less efficient unguided tracking must be used instead.
- Any independently moving object(s) can not be detected until the epipoles are known. Additionally, image features belonging to the moving object can not be tracked efficiently using equation 3 without knowing the background epipole.

The background epipole must be determined both robustly and accurately — obtaining good estimates of the epipolar geometry requires substantially more feature matches than the theoretical minimum (in our experiments at least ~ 15 are needed rather than just 2). Problems arise from mismatched image features as well as from the presence of moving objects in the scene.

Calculating the epipole: The background epipole is calculated as the common intersection of all background feature tracks; this is possible because all image features move along their epipolar lines as the camera translates. The calculation begins by using the RANSAC algorithm to determine a set of features consistent with a single epipole. This step removes the gross outliers caused by independent motion or mismatches. If a sufficient number (typically 6) of consistent features are found then the epipole is deemed correct and we proceed to improve the accuracy of the estimate.

First each feature's trajectory is estimated by fitting a straight line to all the observations of that feature; we also estimate the covariance matrix for the parameters of that line. Next we find the epipole by determining the best estimate of the intersection of all feature trajectories in a least squares sense. In this minimisation each feature trajectory is weighted by its uncertainty in order to enhance the accuracy of the solution. We also determine the covariance matrix for the epipole.

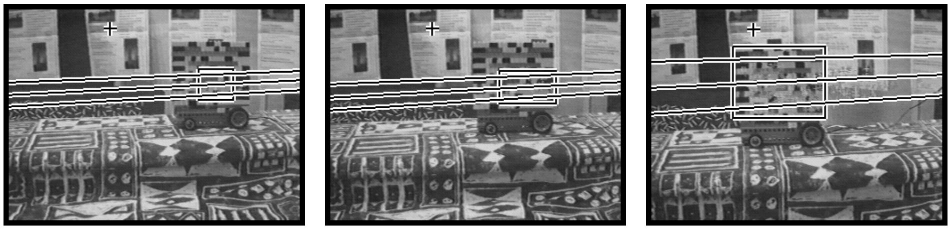
A similar method is used to calculate the epipole for features belonging to a moving object. In our experiments the background epipole is usually found within 3 seconds, and is updated twice per second.

6.2 Experimental results

The real time tracker has been evaluated using the same equipment as the off line implementation (see section 5.4). Figures 6 and 7 show the performance of the real time tracker. The tracker does not have a special initialisation phase, and is able to detect and track an independently moving object. The background epipole is recovered extremely accurately. Only one third of the processing time is devoted to maintaining existing tracks, the remainder is available to the search for new corners. The performance is specially impressive because it has been realised on very limited hardware.

7 Conclusion

By concentrating on the geometry of the simplest case of independent motion we have shown how reasonable assumptions about the world geometry can have powerful consequences in the image plane. We have shown that for a translating camera and each independent object there is only one parameter that must be determined afresh at each stage, and that this may be found by only a one dimensional search for a single feature in each new image. We have also demonstrated a highly efficient corner tracker that is able to track up to two hundred features on the background and independently moving objects at frame



(a) (b) (c)

Figure 6: *Results from a test of the real time implementation showing the evolution of a motion segmentation. The camera and moving object translate along approximately perpendicular paths. In each image a cross marks the background epipole, the current segmentation of the moving object is shown by a rectangle, and three epipolar lines for the moving object are drawn. Notice that the epipole for the moving object lies very far from the object so no collision will occur. Images (a), (b) and (c) are taken 1, 3, and 9 seconds after the object was detected.*

rate while continuously searching for new image features. Remaining tasks are to remove the assumption of translational motion (perhaps by introducing the ground plane assumption), and to address the problem of camera shake.

References

[1]M. Armstrong, A. Zisserman, and P.A. Beardsley. Euclidean structure from uncalibrated images. In E. Hancock, editor, *Proc. 5th British Machine Vision Conf., York*, pages 509–518. BMVA Press, 1994.

[2]P. A. Beardsley, A. Zisserman, and D. W. Murray. Sequential Update of Projective and Affine Structure from Motion. In *Proc. 3rd European Conf. on Computer Vision, Stockholm*, pages 85–96, 1994.

[3]S. Carlsson and J. Eklundh. Object detection using model based prediction and motion parallax. In O. Faugeras, editor, *Proc. 1st European Conf. on Computer Vision, Antibes, France*, pages 207–306, Berlin, 1990. Springer-Verlag.

[4]T. Darrell and A. Pentland. Robust estimation of a multi-layered motion representation. In *Proc. IEEE Workshop on Visual Motion*, 1991.

[5]W. Enkelmann. Obstacle Detection by Evaluation of Optical Flow Fields from Image Sequences. In O. Faugeras, editor, *Proc. 1st European Conf. on Computer Vision, Antibes, France*, pages 134–138, Berlin, 1990. Springer-Verlag.

[6]O. D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In G. Sandini, editor, *Proc. 2nd European Conf. on Computer Vision, Santa Margharita Ligure, Italy*, pages 563–578. Springer-Verlag, 1992.

[7]C. J. Harris and M. Stephens. A combined corner and edge detector. In *Proc. 4th Alvey Vision Conf., Manchester*, pages 147–151, 1988.

[8]R. Hartley. Lines and points in three views – a unified approach. In *ARPA Image Understanding Workshop, Monterrey*, 1994.

[9]M. Irani, B. Rousso, and S. Peleg. Detecting and Tracking Multiple Moving Objects Using Temporal Integration. In G. Sandini, editor, *Proc. 2nd European Conf. on Computer Vision, Santa Margharita Ligure, Italy*, pages 282–287. Springer-Verlag, 1992.

[10]T. Moons, L. van Gool, M. van Diest, and A. Oosterlinck. Affine structure from perspective image pairs under relative translations between object and camera. Technical Report KUL/ESAT/M12/9306, Departement Elektrotechniek, Katholiele Universiteit Leuven, Belgium, 1993.

[11]R. C. Nelson. Qualitative detection of motion by a moving observer. *International Journal of Computer Vision*, 7(1):33–46, November 1991.

[12]D. Sinclair and B. Boufama. Independent motion segmentation and collision prediction for road vehicles. In *Proc. 3rd European Conf. on Computer Vision, Stockholm*, volume 1, pages 161–166, May 1994.

[13]S. M. Smith. Asset-2: Real-time motion segmentation and shape tracking. In *Proc. 5th Int'l Conf. on Computer Vision, Boston*, 1995.

[14]P. H. S. Torr and D. W. Murray. Statistical Detection of Independent Movement from a Moving Camera. *Image and Vision Computing*, 11(4):180–187, 1993.

[15]P. H. S. Torr, A. Zisserman, and D. W. Murray. Motion clustering using the trilinear constraint over three views. In R. Mohr and C. Wu, editors, *Europe-China Workshop on Geometrical Modelling and Invariants for Computer Vision*, pages 118–125, 1995.

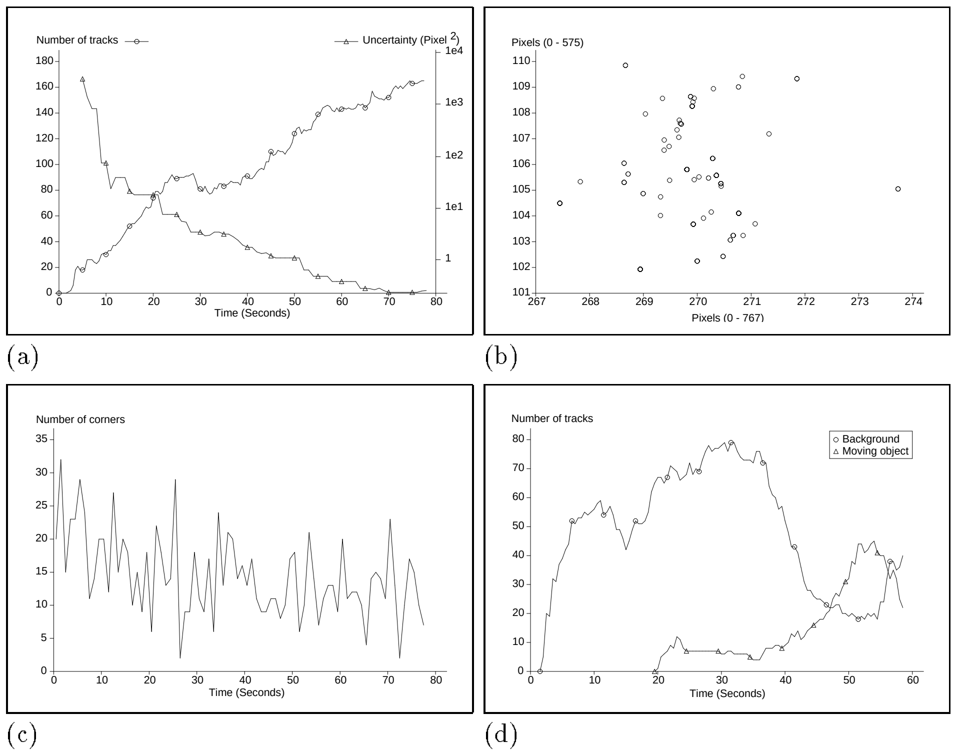


Figure 7: *Performance of the real time tracker. Graph (a) shows the total number of tracks consistent with a single epipole (left axis) and the area of the 95% confidence region for that epipole (right axis). The position of the epipole is shown in (b), it is seen to be very stable over time. The total dispersion in the estimates of the epipole is less than 0.6° in each dimension. The number of new corners found per second is given in (c). In (d) the number of tracks consistent with the background and a moving object are overlayed. The object gradually fills the field of view occluding background features before it passes out of view itself and the number of background tracks recovers.*