On the visual mathematics of tracking

John (Yiannis) Aloimonos* and Dimitris P Tsakiris†

A mathematical theory for visual tracking of a three-dimensional target of known local shape moving rigidly in 3D is presented here, and it is shown how a monocular observer can track an initially foveated object and keep it stationary in the centre of the visual field assuming that the target is always visible during the tracking phase. Our attempt is to develop correspondence-free tracking schemes and get rid of the limitations inherent in the optical flow formalism. A general tracking criterion, the Tracking Constraint, *is derived, which reduces tracking to an appropriate optimization problem. A correspondence-free scheme is devised, that depends on global information about the scene in order to bypass the ill-posed problem of computing the spatial derivatives of the image intensity function, and amounts to the solution of a linear system of equations in order to estimate the 3D motion of the target. Finally, it must be emphasized that – as hinted in the title – this work is devoted entirely to the vision aspects of tracking, and does not get involved in control aspects. In addition, the principal difference between this technique and existing ones is that here tracking is performed in 3D as opposed to 2D. Experimental results with synthetic and real images demonstrate the feasibility of the approach and its robustness against noise.*

Keywords: tracking, tracking constraint, computer vision, 3D, optimization

INTRODUCTION

Visual tracking is an important application of computer vision. Several tracking systems have been developed which either focus on the tracking of targets moving on a plane, or attempt to reduce the 3-dimensional tracking problem to the tracking of a set of characteristic points of the target. These approaches may be seriously handicapped in complex visual situations due to segmentation and feature correspondence problems.

In general, visual tracking is a problem of especial difficulty, since it involves dynamic scene analysis which is not yet fully understood. Assuming knowledge of the shape* of the target, but without using any correspondences and without requiring computation of the optical flow field, we devise a mathematical theory for the most general tracking situation, namely for three-dimensional targets moving in three-dimensional space.

First we describe previous approaches to the tracking problem, we give a general description of a tracking system, and we provide the motivation for our approach to the 3D tracking problem by attempting to answer the question of why we need to estimate the 3D motion of 3D targets moving in 3D, and by showing the advantages of this approach over previous ones.

A kinematic model of the system is derived, and the *Tracking Constraint* is introduced, which reduces the problem of tracking to an appropriate optimization problem. Tracking strategies based on the recovery of the 3D motion of the target are devised under the assumption that we know the local shape of the target, and various experimental results are reported. Finally, future research is discussed.

Optical tracking system description and previous work

Tracking systems may employ active sensors like radar, sonar and laser in order to produce range data of a scene, with explicit depth information. But they involve scanning devices, which are potentially unreliable and consume excessive power, which is an important factor especially in space applications[1,2]. Thus passive electromegnetic sensing, and vision systems in particular, are an attractive alternative for such systems, providing very good spatial and temporal resolution and accuracy which cannot be provided by active sensors, and may be critical in several applications[3,4]. Moreover, active research in this field over the last 30 years has provided solutions to a number of practical problems. Applications of visual tracking include aircraft and missile tracking, robot manipulation of objects, navigation, traffic monitoring, cloud tracking in meteorology, cell

*Computer Vision Laboratory, Center for Automation Research, †Electrical Engineering Department, University of Maryland, College Park, MD 20742-3411, USA

*If Ω is the image of the object, by shape we mean that we know the depth function $z(x, y)$, $\forall (x, y) \in \Omega$, up to a constant factor.

0262–8856/91/004235–17

motion and tracking of moving parts of the body (e.g. the heart) in biomedicine.

In the past, the planar motion of 2D targets[5,6] and 3D targets[7–10] has received considerable attention, due to its important applications, such as the motion of parts on conveyor belts in industry and the analysis of satellite pictures of atmospheric disturbances in meteorology. Here we will consider mainly the more general problem of tracking 3D targets moving in 3-dimensional space.

Roach and Aggarwal[11] considered tracking of rigid convex polyhedra. For each new frame the objects are specified by segmentation. The centroid of each object is computed and from its displacement an estimate of the translational velocity of the object is derived. If the object is partially occluded or uncertainty in the scene prevents segmentation, then individual features such as corners are tracked. Gilbert *et al.*[4,12] presented the design of the US Army Videotheodolite. Segmentation of the image is performed at video rate, using a statistical clustering algorithm to separate the target (missile or aircraft) from the background and plume areas. A window around the target is specified, which simplifies subsequent segmentation. The segmented image is transformed into a binary image of the target, which is used to compute the centroid and orientation of the target. These are the input to the tracking processor, which computes the change in azimuth and elevation of the telescope necessary to continue tracking. The boundary of the projection of a known moving target (aircraft) is characterized by the coefficients of its expansion in a complex Fourier series by Wallace and Mitchell[13]. After normalization, these are compared to a library of coefficients corresponding to different projections of the target, and the target orientation is specified. Tracking involves identification of the target in each new frame, extraction of its boundary and estimation of its orientation using library data. The MIT Eye-Head robot is described by Cornog[14], and control strategies to fixate and track high contrast targets in 3D are presented. The targets are white spheres on a black background, and the image is segmented using thresholding, where the threshold varies with the illumination conditions. In the tracking phase, the previous position of the centre of the sphere is used as an initial point for radial search in order to specify its new position. Its deviation from the line of sight is used as an error signal to the controller that must specify the camera rotation. Error exceeding a threshold induces a saccadic repositioning of the system. A stereo system developed at JPL for tracking 3D targets whose models are known is described by Gennery[3,15] and Wilcox *et al.*[2]. In the acquisition phase, features such as vertices of polyhedra are detected and tracked in 2D, the stereo matching problem is solved, and thus the 3D locations of the features are inferred and matched to those of the object model. During the tracking phase, features are matched directly from images to the object model, without need for stereo matching between pairs of them from the cameras. The object position and orientation is predicted by extrapolation from previous data and is adjusted based on new data from feature detection.

The optical tracking systems presented up to now are composed of two major subsystems: the *vision* and the *control* modules. The first generates a desired camera trajectory in real time that the second has to implement, also in real time.

The *vision module* consists of the necessary hardware and software for the processing of the visual information. It gets the analog image, digitizes it, creates an array of intensities for all points of the field of view and from this information locates the target and computes the camera positioning that will achieve tracking. Locating the target is the part of the tracking process referred to as the *target acquisition* phase and corresponds in the human oculomotor system to the fixation of a target using saccades. It may encompass problems like recognition of the target and recovery of its approximate location in 3D, and these may require image segmentation, feature extraction, creation of a model of the target and matching this model to one of a database of target models. Very often range data from active sensors[2] or structured light[8] are used for the acquisition of the target, if its initial position is not preset by a human operator[10] or by a database of possible target locations. The more complex the scene, the more sophisticated the algorithms for this phase must be, and the more sophisticated they become, the more delay they introduce into the system. The *tracking* phase proper includes the computation and execution of a smooth and continuous camera motion that will achieve tracking by keeping the target foveated and corresponds in the human visual system to the smooth pursuit of a moving object. During this phase, the target tends to drift slowly away from the centre of the image and, after some time, it may be lost. Then a new acquisition phase is required. As in many vision applications, we face here the problem of processing the images in real time. In order to solve the vision problems mentioned earlier, the number of operations that must be performed per unit time may become impressively large. Conventional uniprocessor architectures usually cannot match these processing requirements, but suitable special or parallel architectures may be found for the specific processing task.

The *control module* of the system implements the desired trajectory provided from the vision module. The plant in this case is the camera, with the tracking mount and the actuators that drive it in the azimuth and elevation directions, and is usually modeled as a second-order system. From the system theoretic point of view, the problem of trajectory tracking, in the sense of deriving the control law that will force the plant to follow a desired trajectory, has received considerable attention and several techniques have been examined for its solution. Depending on the technique, the selection of an appropriate control strategy to implement the optimal camera trajectory may be a complicated task. In any case, the hard real-time regime of the problem will impose control cycles ranging from 0.1 to 1 ms, while the updating of the sensory information will be done at video rate (30 to 60 Hz). Although computer speed improvement over the last 25 years has been on the order of 10^6, the average control computer speed improvement was only on the order of Reference 16. This is especially important for visual servoing applications, where scene complexity may dramatically increase the processing time requirements and considerable time delays in the controller may be intolerable. A solution seems to lie in the use of a hierarchy of

control computers with assembly programmed ones handling input/output intensive and real time critical tasks and with high-level programmed ones handling task synchronization and supervisory control. Such hierarchical schemes were successfully used in the past[14].

The inertia of such a mechanical system will prevent instantaneous repositioning of the camera in case the target drifts away from the centre of the camera. Therefore prediction of future positions of the target is necessary, in order for the target to be kept foveated. Moreover, the image processing algorithms in the vision module will introduce additional delays, which must be compensated. As we saw earlier, a physiological analog of a prediction mechanism also exists in the human oculomotor system[54,55]. Various predictive schemes for visual tracking, based on linear regression and Kalman filtering, are presented by Hunt and Sanderson[17]. As we saw, one important reason for tracking is to reposition a high resolution, but narrow field of view, photoreceptor array (like the fovea), so that more information about the target becomes available. Then, accurate prediction of the target's future positions becomes more critical, since if the target is lost, the time-consuming acquisition phase must be repeated. In general, due to delays introduced from the image processing and control systems, a trade-off between the velocity of the moving target and the complexity of the visual situation, which will influence our choice of the vision algorithms, seems inevitable in a tracking system. The delays are difficult to quantify *a priori*, since they will depend a lot on the hardware which is available for computation and control.

Finally, the mechanical problems arising in the design of high-performance machines, like the choice of actuators and transmission components in order to achieve the best dynamic performance and many other engineering issues related to the manufacturing of those machines, are also present in the design of a tracking system and should be given appropriate consideration.

Main results and criticism of previous work

In the following we will concentrate on the tracking phase of the tracking of a 3D target and assume that during the acquisition phase the target was brought to the centre of the image plane. Previous approaches to the problem are composed of three main steps: first, they perform segmentation of the image or of appropriately selected parts of it in order to locate the target and extract the new position of some characteristic points of its image (centroid, feature points, etc.); second, they match these new characteristic points with their corresponding ones in previous or current (in the binocular case) frames, and thus extract information about the displacement of the target; and third, from this displacement they compute the necessary camera rotation that will achieve tracking.

In the sequel we describe these steps and the problems that they introduce in more detail.

In the first step, *image segmentation* is used in order to separate the target from its background. In most early tracking schemes, segmentation is done at every new frame in order to specify the target location. In later schemes, segmentation is done mainly during the acquisition phase and windows are specified around the target or around feature points of the target, which are updated during the tracking phase according to the estimated target motion. These windows facilitate segmentation by restricting it to a relatively small area of the image. In computer vision, image segmentation has been established as one of the most difficult problems, especially in complex scenes. Segmentation algorithms based on thresholding and stochastic image models for image generation have been developed, but their usefulness is usually limited for natural images, due either to their assumptions about the visual world or the need for continuous adjustment of thresholds in the case of varying imaging conditions.

The second step is a generalization of the centroid algorithms used in tracking planar targets moving in 2D. In the 2D case the centroid of the image corresponds to a specific physical point on the target or in the area of the target and, most important, will continue to correspond to the *same* physical point during the entire motion of the target. Therefore, the claim that its motion describes the motion of the whole target is justified, since the motion of every other point of the rigid target is kinematicaly related to the motion of this characteristic point. If we attempt to use a similar analysis in 3D though, it is going to fail, since, as the target moves, the centre of brightness at each time instant will not correspond to the same physical point on the target (this can be readily seen by considering the tracking of a car moving towards the observer; initially the centroid will probably correspond to a point of the front windshield but as the car passes in front of the observer, the new centroid may correspond to a point on the door). Therefore, knowing the motion of the centroid does not necessarily provide the motion of every other point of the target; it does not satisfactorily characterize the motion of the target. Thus the computationally simple tracking methods of the 2D case, based on the recovery and tracking of the image centre of brightness, have to be replaced by harder ones in the 3D case, which use feature extraction, matching and feature tracking. In this case feature correspondence between consecutive frames has to be accounted for and the difficulty of this problem is proportional to the complexity of the scene. Alternatively, *ad hoc* methods for specific targets were developed, where the new orientation of the target is specified at each time instant by comparing its projection to a database of target projections.

The third step involves the computation of the camera angular velocity ω needed to achieve tracking. Usually ω is computed as $\omega = Ke$, where e is the displacement of one of the characteristic points computed in the second step and K is a constant. Clearly, the issue here is whether this displacement has been computed accurately, despite all the segmentation and point correspondence problems and, more importantly, whether the displacement of this point really characterizes the displacement of the target.

The fundamental question every visual tracking scheme has to answer is "Where is the new position of the target?" Previous methods attempted to do so by exploiting only "static" information, such as the one provided by each separate frame. We *directly* exploit information about the temporal variation of the image induced from the target motion. The target separation

problem then has a very elegant solution based on "dynamic" information from consecutive frames, which enables us to distinguish what moves (target) from what doesn't (background). The carriers of information are then the derivatives of the image intensity functions, not the displacements of heuristically specified characteristic points of the target. For example, in the case of a ball moving over a uniform background, the viewing situation is simple enough to allow segmentation and feature correspondence-based algorithms to perform satisfactorily. But in the case of a book moving in front of a bookcase full of books, the edge detection algorithm will produce such a large number of edges that the segmentation task will become impossible. On the other hand, the image spatiotemporal derivatives will capture the variation of the image generated from the target motion and this information will *dynamically segment* the image into moving and static parts.

In our approach, we consider a general criterion for 3D tracking, the *tracking constraint*, which if satisfied, guarantees tracking in the sense of keeping an initially foveated target stationary in the centre of the camera, without being restricted to a specific target or visual environment. We follow the continuous motion approach and use the concept of optical flow to formulate this constraint. The tracking constraint uses global information from entire areas of the image, not local information from specific points, and does not require feature extraction or matching. It requires, though, knowledge of the shape and motion of the target, and it is formulated in terms of shape and 3D motion instead of the optical flow field. Provided the shape is known, an algorithm is presented which estimates the 3D motion of the target from image data. This estimate of the target motion parameters, together with the known shape of the target, are used with the tracking constraint to produce the desired camera angular velocity that will track the target. Obviously, a requirement for knowledge of the shape in a general tracking scheme is far less restrictive than an *ad hoc* tracking scheme for a specific target or class of targets, since the latter may be used only for a specific target, while the former may be used for *any* tracking problem where the shape is known exactly or approximately.*

Comparison with other theories

The existence of a multitude of tracking systems demonstrates the fact that there does not exist to date a tracking system able to track successfully any moving object under any circumstances. Excluding the 3D model based tracking systems described in the previous sections and comparing our theory with 2D tracking systems that claim generality, we find that such 2D systems based on some sort of prediction (*ad hoc*) could fail in such a way that is not predictable from the image data. Put more simply, a general 2D tracking system can totally fail while there is no indication in the image data that such a thing is about to happen; and this is the striking difference between such systems and the one developed here. Our system is based on a strict computational theory that gives rise to algorithms (and an implementation). At the heart of the computational theory (see next section) lies a procedure for computing the three-dimensional motion of the target, without correspondence and under the assumption of the knowledge of its shape. For this to happen, several measurements on the target image (which can be "nice" or nondifferentiable) must be performed. When such measurements cannot be carried out, it is obvious that this tracking system will fail. One such case appears when the image of the target occupies a very small part of the image. On the one hand, there is not enough data for the procedure to work and on the other hand the assumption of the knowledge of shape becomes meaningless, when the target occupies a "few" pixels on the image plane. For such cases, for example tracking a missile with an airborne camera, 2D tracking seems more appropriate. The present system will work well when the target is not far away, for example in an indoor robotic environment (its image is "large" enough† to provide enough data).

Finally, and most importantly, the theory described here, considers tracking in 3D and consequently suggests several research avenues on how tracking interacts with modules such as shape from "*x*", structure from motion, etc.; such interactions are necessary in our effort to integrate various vision modules[20,21].

COMPUTATIONAL THEORY OF TRACKING

In this section we will devise the model we use in our analysis of the tracking process. The tracking process will be explicitly defined and a computational theory (in the sense of Marr[22]) for tracking will be derived from the modelling process.

Tracking requires that a mathematical model be established which describes the relationship between the target and its image taken by the camera and between information derived from this image and the kinematic parameters of the motion of the target. To this end, we will define the coordinate systems that we will need and provide their relationships.

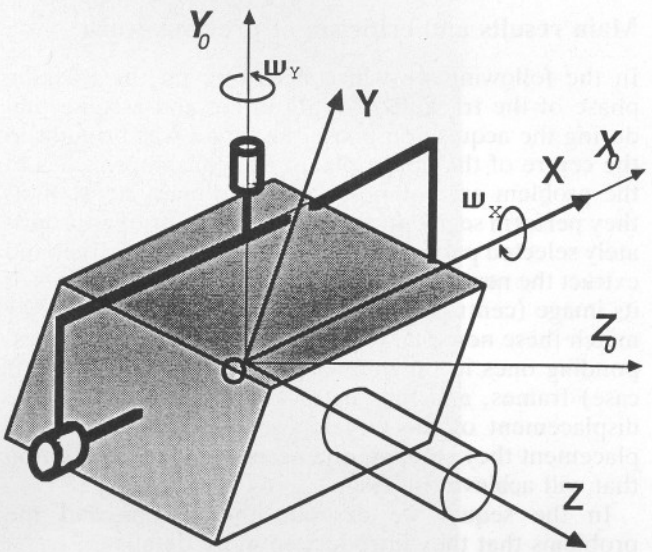

Figure 1. World and camera coordinate systems

*We require knowledge of shape only in order to solve the problem in a correspondenceless manner.

†Complete quantitative analysis of instability connected to the motion estimation problem is still lacking from the literature, and consequently from this part of our paper. For some preliminary results see Spetsakis and Aloimonos[18] and Adiu[19].

Suppose there exists an inertial frame in $\mathbb{R}^3$, the **world coordinate system** $OX_OY_OZ_O$, with respect to which all quantities are measured (see Figure 1). The ideal pinhole camera model is used. An image is a two-dimensional pattern of brightness projected on the image plane. Consider the **camera coordinate system** $OXYZ$, with Z along the optical axis, i.e. the perpendicular from the pinhole to the image plane pointing toward the scene. This induces a two-dimensional coordinate system, the **image-plane coordinate system** oxy on the image plane. The *focal length* f of the camera is defined as the distance Oo. Consider finally a **target coordinate system** $WX'Y'Z'$ attached to the rigid body that we want to track.

If P: $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ is the projection function, consider a point A in $\mathbb{R}^3$ with coordinates $\vec{\mathbf{X}} = (X, Y, Z)$ relative to the camera coordinate system, which is projected to the point P_A of the image plane with coordinates $\vec{\mathbf{x}} = (x, y)$. Then,

$$\vec{x} = \begin{pmatrix} x \\ y \end{pmatrix} = P(\vec{\mathbf{X}}) = \begin{pmatrix} P_1(\vec{\mathbf{X}}) \\ P_2(\vec{\mathbf{X}}) \end{pmatrix} \tag{1}$$

Assume the target is stationary with respect to the target coordinate frame, i.e. $\vec{X}' = 0 \Longrightarrow \vec{X}'(t) = \vec{X}'(t_0) \stackrel{\text{def}}{=} \vec{X}'$ and is moving rigidly with respect to the other coordinate frames.

Consider the target with its point W (see Figure 2) moving with translational velocity $\vec{T} = (T_1, T_2, T_3)$ and with spatial angular velocity $\vec{\omega} = (\omega_1, \omega_2, \omega_3)$ relative to the *camera* frame. From the system kinematics[23,24], it can be shown that:

(i) $\vec{X}(t) = \vec{r}(t) + R(t)\vec{X}'$

with $R(.) \in \mathbb{R} \times SO(3)$, where $SO(3)$ is the Special Orthogonal (Rotation) Group of order 3.

(ii) $\dot{\vec{X}}(t_0) = \vec{T} + \hat{\omega}\vec{X}(t_0)$ (2)

where

$$\hat{\omega} = \begin{pmatrix} 0 & -\omega_3 & \omega \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} =$$

$$= \frac{dR(t_0)}{dt} R^{\mathrm{T}}(t_0) \quad \text{and} \quad \vec{T} = \frac{d\vec{r}(t_0)}{dt} - \omega\vec{r}(t_0)$$

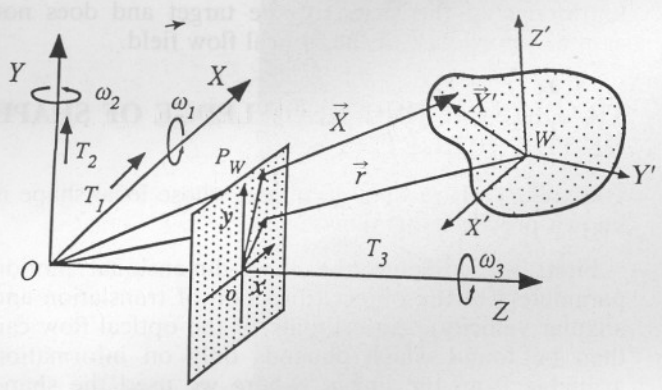

Figure 2. Camera, image-plane and target coordinate systems

The effect of a rotation of the camera on the image formation process is exactly the same as the effect of an opposite rotation of the target on the process. Put in another way, since the formation of an image on the image plane depends on the relative motion of the camera and the target, it is irrelevant to this process whether the camera is rotating in one direction or the target is rotating in the opposite one. This is formally proven elsewhere[24]. This is an important fact, since it will allow us to make the subsequent analysis in the camera coordinate frame instead of the world frame and thus achieve a remarkable simplification of the corresponding formulae.

Optical flow is the velocity field generated on the image plane from the projection of objects moving in 3D from the motion of the observer with respect to the scene or from apparent motion, when a series of images gives the illusion of motion. The image intensity $s(x, y, t)$ at a point (x, y) of the image plane at time t is related to the instantaneous velocity $(u^t(x, y), v^t(x, y))$ at that point by the optical flow constraint equation[25].

$$s_x(x, y, t)\, u^t(x, y) + s_y(x, y, t)\, v^t(x, y) + s_{t^*}(x, y, t) = 0, \tag{3}$$

The computation of the optical flow field is a very hard problem and although some very good research has been published[25–35], the problem is far from solved, especially for the case of complex visual environments involving discontinuities. For this reason we avoid the use of optical flow as the input to the perceptual process of tracking, although tracking is facilitated tremendously if the optical flow is known[24].

Let $\mathcal{R}$ be the projection of our target on the image plane S. Suppose that our target has a prominent feature that we want to track at a point W in $\mathbb{R}^3$, which can be identified during the target acquisition phase and which is projected to a point P_W of the image plane. Because we will devise tracking schemes that will use only global information from large areas of the image, we do not need an exact solution to the feature extraction problem, i.e., the exact location of the point P_W. All we need is a neighbourhood $\mathcal{B}_W(P_W, \epsilon_W) \subset \mathcal{R}$, with ϵ_W acceptably small. This is very important, since recovering the area $\mathcal{B}_W$ during the target acquisition phase is going to be much more robust to noise than attempting to recover a single point P_W.

Suppose that during the target acquisition phase, the area $\mathcal{B}_W$ was recovered and moved to the origin of the image plane. We define the *tracking problem* as follows: Find the camera rotation (ω_x, ω_y) that will hold $\mathcal{B}_W(P_W, \epsilon_W)$ stationary in a neighbourhood $\mathcal{B}(\mathbf{o}, \epsilon)$ of the origin $\mathbf{o}$ of the image plane.

Suppose now that we get an image $s(x, y, t)$ at time t and we want to solve the tracking problem defined above. We must specify a camera rotation (ω_x, ω_y) that will act from time t to $t + dt$ and that will keep $\mathcal{B}_W$ at the origin despite the motion of the target. Let the optical flow at time t be $(u^t(x, y), v^t(x, y))$ at a point $(x, y) \in S$ and at time $t + dt$ let it be $(u^{t+dt}(x, y), v^{t+dt}(x, y))$. Now, (u^{t+dt}, v^{t+dt}) will be the superposition of the optical flow induced by two motions: The motion of the target by $(\vec{T}, \vec{\omega})$ and the tracking motion of the camera by (ω_x, ω_y). Suppose that the motion of the target during the time interval $[t, t + dt]$ that will induce the

first component of (u^{t+dt}, v^{t+dt}) will be almost identical to its motion during the time interval $[t-dt, t]$ that induced the optical flow (u^t, v^t). Therefore this first component of (u^{t+dt}, v^{t+dt}) will be $u^t, v^t)$. The second component of (u^{t+dt}, v^{t+dt}) will be induced from the motion of the camera; let it be (u_{CAM}, v_{CAM}). Therefore:

$$u^{t+dt} \approx u^t + u_{CAM} \tag{4}$$

and:

$$v^{t+dt} \approx v^t + v_{CAM} \tag{5}$$

Assuming that the focal length is $f=1$, let us see exactly how the motion influences the induced optical flow for the case of perspective projection. It is easy to prove that:

$$u^t(x, y) = \frac{T_1 - xT_3}{Z} - \omega_1 xy + \omega_2(x^2+1) - \omega_3 y \tag{6}$$

$$v^t(x, y) = \frac{T_2 - yT_3}{Z} - \omega_1(y^2+1) + \omega_2 xy + \omega_3 x. \tag{7}$$

The more general case of an arbitrary projection P: $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ is treated below.

In order to compute the optical flow induced from the camera motion, we take advantage of the nature of optical flow as the velocity field induced by the *relative* motion of the camera and the target. This is related to our previous observation that a rotation of the camera has the same effect on the image formation process as an additional opposite rotation of the target. Therefore, we can see that the camera moving by ω_x, ω_y generates optical flow u_{CAM}, v_{CAM}, which is exactly the same to the one that would be generated by the target, if the target underwent an additional motion by $-\omega_x$, $-\omega_y$.

Then the respective camera-induced optical flow will be given from (6) and (7), with $\vec{T}=0$ and $\vec{\omega}=(-\omega_x, -\omega_y, 0)$:

$$u_{CAM} = \omega_x xy - \omega_y(x^2+1) \tag{8}$$

$$v_{CAM} = \omega_x(y^2+1) - \omega_y xy \tag{9}$$

Thus, the total optical flow at time $t+dt$ will be

$$u^{t+dt} = u^t + \omega_x xy - \omega_y(x^2+1) \tag{10}$$

$$v^{t+dt} = v^t + \omega_x(y^2+1) - \omega_y xy \tag{11}$$

Now, for the tracking to be successful, at time $t+dt$ the point P_W must be stationary at the origin, and therefore the optical flow at the origin must be zero, i.e.:

$$u^{t+dt}(0, 0) = v^{t+dt}(0, 0) = 0 \tag{12}$$

or:

$$u^t(0, 0) + u_{CAM}(0, 0) = v^t(0, 0) + v_{CAM}(0, 0) = 0 \tag{13}$$

Therefore, by appropriate choice of ω_x and ω_y, the optical flow u_{CAM} and v_{CAM} induced from the camera motion as given from (8) and (9) will be such that (13) holds.

In order to increase robustness, in addition to (13) we require that the optical flow on an entire neighbourhood $\mathcal{B}(0, \epsilon)$ of the origin (with $\epsilon \geqslant \epsilon_W$), be minimal. This is then our *tracking constraint* and is formulated as follows: the desired camera angular velocity (ω_x^*, ω_y^*) that will achieve tracking is the minimizer of:

$$\begin{aligned} f(\omega_x, \omega_y) &= \iint_{\mathcal{B}} [(u^{t+dt})^2 + (v^{t+dt})^2]\, dx\, dy \\ &= \iint_{\mathcal{B}} [(u^t + \omega_x xy - \omega_y(x^2+1))^2 \\ &\quad + (v^t + \omega_x(y^2+1) - \omega_y xy)^2]\, dx\, dy \end{aligned} \tag{14}$$

Notice that the tracking constraint is expressed in terms of the neighbourhood $\mathcal{B}$ of the origin of the camera, not some neighbourhood of a target feature whose specification would require segmentation of the image and feature extraction during the tracking phase. Moreover, the tracking problem is now reduced to an optimization problem and appropriate mathematical methods can be used to attack it.

An algorithm for tracking the motion of an object moving in 3D then suggests itself:

Step 1:
Estimate $(\vec{T}, \vec{\omega})$ from the image and then derive the optical flow induced from this motion, using equations (6) and (7).

Step 2:
Minimize expression (14) and derive the desired camera angular velocities that will achieve tracking.

Typically, Step 1 involves the optical flow constraint, and Step 2 the tracking constraint. Clearly, if we assume knowledge of the optical flow, then tracking is achieved through minimization of (14) or, if (14) is combined with (3), tracking is achieved through solution of a penalized least-squares problem (the reader interested in this approach is referred to Tsakiris[24]). However, due to the fact that the computation of the optical flow still remains a very hard problem, we chose to seek an alternative approach. The next section describes our algorithm for tracking based on the knowledge of the shape of the target and does not assume knowledge of the optical flow field.

TRACKING USING KNOWLEDGE OF SHAPE OF THE TARGET

Our theory of tracking an object whose local shape is known proceeds as follows:

First, we compute the three-dimensional motion parameters of the object (direction of translation and angular velocity). An estimate of the optical flow can then be found which depends only on information available from the image, where we used the shape information in order to eliminate the depth terms of equations (6) and (7). Based on this and on the concept

of linear features (defined below), we can estimate the target motion parameters by solving a system of linear equations. The system does not use point correspondences, only global information from the image;

Second, from the tracking constraint, we formulate a cost functional whose minimizers are the camera angular velocities ω_x and ω_y, which are necessary in order for the camera to track the moving target. Using our estimate of the target motion parameters, we can solve this minimization problem for ω_x and ω_y.

Linear features

Linear features of an image are sets of functionals of the type

$$F = \left\{ f_i(t) \,|\, f_i(t) = \iint_D s(x, y, t)\,\mu_i(x, y)\,dx\,dy, \quad i = 1, \ldots, n \right\},$$

where s is the image intensity at a point (x, y) of a subset D of the image at time t and μ_i: $\mathbb{R}^2 \rightarrow \mathbb{R}$ is the *measuring function* (also referred to as *moment weighting kernel*).

Our purpose for introducing linear features in the study of the 3D motion estimation problem is not to classify or reconstruct a pattern of some kind, as it is in classical pattern recognition[36–42, 13], but is related to the errors introduced into the optical flow constraint equation when it is used to process discretized information. In principle, the 3D motion parameters could be recovered (up to depth) by a formulation like the following: from (3), (6) and (7) we get for every point (x, y):

$$s_x\left[\frac{T_1}{Z} - x\frac{T_3}{Z} - \omega_1(y^2+1) + \omega_2 xy + \omega_3 x\right] + s_y\left[\frac{T_2}{Z} - y\frac{T_3}{Z} - \omega_1 xy + \omega_2(x^2+1) - \omega_3 y\right] + s_t = 0$$

and then:

$$s_x\frac{T_1}{Z} + s_y\frac{T_2}{Z} - (s_x x + s_y y)\frac{T_3}{Z} - [s_x(y^2+1) + s_y xy]\,\omega_1 + [s_x xy + s_y(x^2+1)]\,\omega_2 + [s_x x - s_y y]\,\omega_3 = -s_t.$$

Given enough points (x, y) and expressing the depth function Z in terms of shape and retinal coordinates, we will get a system of linear equations whose solution gives the 3D motion. However, the problem is that the coefficients of the system depend on the spatiotemporal derivatives of s at each point (x, y) and, since numerical differentiation is an ill-posed (and consequently ill-conditioned) problem[56], severe errors will be introduced if we attempt to evaluate these derivatives numerically.

Therefore an algorithm is presented below where, using linear features to collect global information from an area of the image, the computation of the spatial derivatives of the image is no longer necessary[43]. A problem common in this approach and in the use of linear features in pattern recognition is how to select the most appropriate measuring functions and what the length of the linear feature vector should be, so that the characterization of the image by this vector be "rich" enough in information. This point will be discussed further later.

Recovery of the three-dimensional motion

Suppose that an object exists in $\mathbb{R}^3$, with its surface given as a function $Z = g(X, Y)$ in the camera coordinate system $OXYZ$. The following analysis is done with respect to the camera coordinate frame, but can be readily extended to the world frame. However, this is not necessary as was proven from the kinematic analysis of the above problem. Consider a point A on the surface of the object which is projected to the point P_A of the image plane. Suppose that A has coordinates $\vec{X} = (X, Y, Z)$ with respect to the camera coordinate system at time t and that P_A has coordinates $\vec{x} = (x, y)$ with respect to the image-plane frame at the same instant.

Theorem

The optical flow $\vec{x} = (u, v)$ generated from the motion of a rigid target relative to our camera is a linear function of the target motion parameters of the form

$$\vec{x} = \begin{bmatrix} u \\ v \end{bmatrix} = \sum_{k=1}^{6} m_k \vec{u}_k(\vec{x}) \tag{15}$$

where m_k are the motion parameters (direction of translation and rotation) and $\vec{u}_k$ are known functions of the shape, the projection function and the retinal coordinates.

Proof.

If P: $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ is the projection function, then:

$$\vec{x} = P(\vec{X}) = \begin{pmatrix} P_1(\vec{X}) \\ P_2(\vec{X}) \end{pmatrix} \tag{16}$$

Suppose that the object in view performs a rigid motion. We have seen that each point A will move with a velocity $\vec{V}$, which is a function of its position $\vec{X}$ with respect to the camera coordinate system. More precisely, from (2):

$$\vec{\dot{X}} \triangleq \vec{V}(\vec{X}) = \vec{T} + \hat{\omega}\,\vec{X} = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \end{pmatrix} + \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{17}$$

where $\vec{T}$ is the translational velocity of the centre W of the target frame and $\vec{\omega}$ the spatial angular velocity of the target with respect to the camera frame. The

velocity $\vec{V}$ can be written as:

$$\vec{V}(\vec{X}) = T_1\begin{pmatrix}0\\0\\1\end{pmatrix} + T_2\begin{pmatrix}0\\1\\0\end{pmatrix} + T_3\begin{pmatrix}0\\0\\1\end{pmatrix} + \omega_1\begin{pmatrix}0\\-Z\\Y\end{pmatrix} + \omega_2\begin{pmatrix}Z\\0\\-X\end{pmatrix} + \omega_3\begin{pmatrix}-Y\\X\\0\end{pmatrix} \quad (18)$$

$$= \sum_{k=1}^{6} m_k V_k(\vec{X})$$

where m_i are the target motion parameters defined as follows:

$$m_1 = T_1, \quad m_2 = T_2, \quad m_3 = T_3, \quad m_4 = \omega_1, \quad m_5 = \omega_2, \quad m_6 = \omega_3$$

and:

$$\vec{V}_1(\vec{X}) = \begin{pmatrix}1\\0\\0\end{pmatrix}, \quad \vec{V}_2(\vec{X}) = \begin{pmatrix}0\\1\\0\end{pmatrix}, \quad \vec{V}_3(\vec{X}) = \begin{pmatrix}0\\0\\1\end{pmatrix},$$

$$\vec{V}_4(\vec{X}) = \begin{pmatrix}0\\-Z\\Y\end{pmatrix}, \quad \vec{V}_5(\vec{X}) = \begin{pmatrix}Z\\0\\-X\end{pmatrix}, \quad \vec{V}_6(\vec{X}) = \begin{pmatrix}-Y\\X\\0\end{pmatrix}$$

As the target moves in $\mathbb{R}^3$, its image also moves on the image plane. The projection P_A of the point A with image plane coordinates $\vec{x} = P(\vec{X})$, moves with velocity $\dot{\vec{x}}$, which is the optical flow at this point. From (16):

$$\dot{\vec{x}}(\vec{X}) = \begin{pmatrix}\dot{x}\\\dot{y}\end{pmatrix} \triangleq \begin{pmatrix}u\\v\end{pmatrix} = \frac{d}{dt}P(\vec{X}) = \frac{\partial P}{\partial \vec{X}}(\vec{X})\,\vec{V}(\vec{X}) = \begin{pmatrix}\frac{\partial P_1}{\partial X} & \frac{\partial P_1}{\partial Y} & \frac{\partial P_1}{\partial Z}\\ \frac{\partial P_2}{\partial X} & \frac{\partial P_2}{\partial Y} & \frac{\partial P_2}{\partial Z}\end{pmatrix}\begin{pmatrix}\dot{X}\\\dot{Y}\\\dot{Z}\end{pmatrix} \quad (19)$$

Moreover, from the projection function $\vec{x} = P(\vec{X})$ we can derive the inverse projection mapping P_g^{-1} from the image to the surface, i.e., $\vec{X} = P_g^{-1}(\vec{x})$, which is known up to a factor (depth), since the shape $Z = g(X, Y)$ of the target is known. We can then consider $\partial P/\partial \vec{X}$ and $\vec{V}$ and thus the optical flow $\dot{\vec{x}}$ itself as functions of the retinal coordinates $\vec{x}$. Then:

$$\frac{\partial P}{\partial \vec{X}}(P_g^{-1}(\vec{x})), \quad \vec{V} = \vec{V}(P_g^{-1}(\vec{x}))$$

and:

$$\dot{\vec{x}} = \begin{pmatrix}u\\v\end{pmatrix} = \frac{\partial P}{\partial \vec{X}}(P_g^{-1}(\vec{x}))\,\vec{V}(P_g^{-1}(\vec{x}))$$

where for $A \subset \mathbb{R}^2$ and $B \subset \mathbb{R}^3$ we have $P_g^{-1}: A \to B$ the inverse transformation of P.

From (18) and (19) we have:

$$\dot{\vec{x}} = \begin{pmatrix}u\\v\end{pmatrix} = \frac{\partial P}{\partial \vec{X}} \sum_{k=1}^{6} m_k \vec{V}_k(\vec{X}) =$$

$$\sum_{k=1}^{6} m_k \frac{\partial P}{\partial \vec{X}}(P_g^{-1}(\vec{x}))\,\vec{V}_k(P_g^{-1}(\vec{x})) = \sum_{k=1}^{6} m_k \begin{pmatrix}u_{k_1}(\vec{x})\\u_{k_2}(\vec{x})\end{pmatrix}$$

where we have defined:

$$\vec{u}_k(\vec{x}) \triangleq \begin{pmatrix}u_{k_1}(\vec{x})\\u_{k_2}(\vec{x})\end{pmatrix} = \frac{\partial P}{\partial \vec{X}}(P_g^{-1}(\vec{x}))\,\vec{V}_k(P_g^{-1}(\vec{x})) \quad (20)$$

Obviously, $\vec{u}_k(\vec{x})$ depends only on the shape of the target and is known. Then:

$$\dot{\vec{x}} = \begin{pmatrix}u\\v\end{pmatrix} = \sum_{k=1}^{6} m_k \vec{u}_k(\vec{x}) \quad (21)$$

and we have expressed the optical flow at a point of the image as a function of the shape of the target, its motion parameters m_i and the retinal coordinates. Moreover, it is given as a linear combination of the motion parameters.

Estimation of the motion paramaters m_i

The problem now is to compute the motion parameters m_i, $i = 1, \ldots, 6$, of (15), without actually computing the optical flow or the derivatives of the image intensity function. For this purpose, we use the concept of linear features presented earlier. Consider a set F of linear features over a neighbourhood $\mathcal{B}(0, \epsilon)$ of the origin of the image plane $F(t) = \{f_i(t) = \int\int_{\mathcal{B}} s(x, y, t))\,\mu_i(x, y)\,dx\,dy,\ i = 1, \ldots, n\}$. From the optical flow constraint equation we have $s_x u + s_y v + s_t = 0$ or $\partial s/\partial t = -\dot{\vec{x}}^{\mathsf{T}}\nabla s$. Consider the temporal derivative of the i^{th} linear feature:

$$\dot{f}_i = \int\int_{\mathcal{B}} \frac{\partial s}{\partial t}\,\mu_i\,dx\,dy = -\int\int_{\mathcal{B}} \mu_i(\dot{\vec{x}}^{\mathsf{T}}\nabla s)\,dx\,dy \quad (22)$$

$$\overset{(3.1)}{=} -\sum_{k=1}^{6} m_k \int\int_{\mathcal{B}} \mu_i(u_{k_1}s_x + u_{k_2}s_y)\,dx\,dy.$$

Defining $h_{ik} = -\int\int_{\mathcal{B}} \mu_i(u_{k_1}s_x + u_{k_2}s_y)\,dx\,dy$, we get $\dot{f}_i = \Sigma_{k=1}^{6} m_k h_{ik}$ and then:

$$\mathbf{H}\vec{m} = \vec{f}, \quad (23)$$

where $\mathbf{H}$ is the $n \times 6$ matrix of the coefficients h_{ik} and $\vec{m} = (m_1, \ldots, m_6)$. The vector of derivatives of linear features $\vec{f}$ and the matrix $\mathbf{H}$ depend on measurable quantities. We want to solve the linear system (23) and obtain the motion parameter vector $\vec{m}$. Since the number n of linear features will normally be greater than 6, the system will be inconsistent in the general case. We can select the linear features such that $rank(\mathbf{H}) = 6$. Therefore, a least-squares approximate solution becomes necessary.

In our case, the system (23) has the unique minimal norm solution:

$$\vec{m} = \mathbf{H}^{+}\vec{f}, \quad (24)$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose inverse of $\mathbf{H}$. This can be computed either directly from the formula $\mathbf{H}^{\dagger} = (\mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathrm{T}}$ if $\mathbf{H}$ has $rank(\mathbf{H}) = m$, or using Greville's algorithm, an iterative procedure that computes $\mathbf{H}^{\dagger}$ in m iterations. At the k^{th} iteration it computes $\mathbf{H}_{\mathbf{k}}^{\dagger}$, where $\mathbf{H}_{\mathbf{k}}$ is the submatrix of $\mathbf{H}$ consisting of its first k columns. The advantage of the second method is that we don't have to check the rank of $\mathbf{H}$, and we avoid the evaluation of enormous analytic expressions for $(\mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}\mathbf{H}^{\mathrm{T}}$, if n is large. Greville's algorithm is very well suited for our case, since it will compute $\mathbf{H}^{\dagger}$ in $m = 6$ steps, no matter what the length n of the linear feature vector is[44].*

In our target motion parameter estimation scheme no explicit calculation of the optical flow field is needed, as we have seen up to this point. In the next section we will see that neither is it needed for the tracking scheme developed there. Moreover, the only calculation involving the derivatives of the image intensity function s is done in order to obtain the parameters h_{ik}. Spatial differentiation of s is needed there, but since numerical differentiation is an ill-posed problem, we attempted to bypass it by using integration by parts in (22). Supposing that the area $\mathcal{B}$ is a rectangle $\mathcal{B} = [a, b] \times [c, d]$ in image plane coordinates, we have:

$$
\begin{aligned}
h_{ik} &= -\int_a^b \int_c^d \mu_i (u_{k_1} s_x + u_{k_2} s_y)\, dx\, dy \\
&= -\int_c^d [\mu_i(b, y)\, u_{k_1}(b, y)\, s(b, y) - \\
&\qquad \mu_i(a, y)\, u_{k_1}(a, y)\, s(a, y)]\, dy \qquad (25) \\
&\quad -\int_a^b [\mu_i(x, d)\, u_{k_2}(x, d)\, s(x, d) - \\
&\qquad \mu_i(x, c)\, u_{k_2}(x, c)\, s(x, c)]\, dx \\
&\quad +\int_a^b \int_c^d s(x, y) \left[\frac{\partial}{\partial x} (\mu_i(x, y)\, u_{k_1}(x, y)) + \right. \\
&\qquad \left. \frac{\partial}{\partial y} (\mu_i(x, y)\, u_{k_2}(x, y)) \right] dx\, dy.
\end{aligned}
$$

Therefore, numerical spatial differentiation of s was replaced by differentiation of μ_i, u_{k1} and u_{k2}, which can be done analytically.

Due to the problem degeneracy[38, 45–47],from the above process, as well as from every method for the recovery of motion from monocular data, we can only recover the direction of translation and the rotation of the target, due to the unknown depth scale. This becomes obvious from the following example, where we define $m_i = T_i/c$, $i = 1, 2, 3$, and c is essentially a measure of depth.

Example: the planar case

The above analysis is particularly simple in the case of a plane moving in 3D. This is a very interesting case, not only by itself or for the case of the polyhedral world, but also as a local approximation of more general surfaces[35]. We will show how (15) transforms under perspective projection. Suppose we want to track a point W on the plane and suppose that this point translates with velocity T and the plane rotates with angular velocity ω with respect to the camera frame. We want to estimate T and ω from visual data.

The perspective projection function is:

$$
\vec{x} = \begin{pmatrix} x \\ y \end{pmatrix} = P(\vec{X}) = \begin{pmatrix} \dfrac{fX}{Z} \\ \dfrac{fY}{Z} \end{pmatrix} \qquad (26)
$$

Consider the plane $Z = g(\mathrm{X}, \mathrm{Y}) = pX + qY + c$. The shape parameters p and q are supposed to be known, but not the depth factor c (since we consider the monocular case). The inverse projection transformation $P_g^{-1}(\vec{x})$ is:

$$
\vec{X} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = P_g^{-1}(\vec{x}) = \frac{c}{f - px - qy} \begin{pmatrix} x \\ y \\ f \end{pmatrix} \qquad (27)
$$

Then from (26) and (27):

$$
\frac{\partial P}{\partial \vec{X}}(\vec{X}) = \begin{pmatrix} \dfrac{f}{Z} & 0 & -\dfrac{fX}{Z^2} \\ 0 & \dfrac{f}{Z} & -\dfrac{fY}{Z^2} \end{pmatrix}
$$

$$
\Rightarrow \frac{\partial P}{\partial \vec{X}}(P_g^{-1}(\vec{x})) = \frac{f - px - qy}{\mathrm{c}} \begin{pmatrix} 1 & 0 & -\dfrac{x}{f} \\ 0 & 1 & -\dfrac{y}{f} \end{pmatrix}
$$

From (17) and (27) we have:

$$
\vec{V}(P_g^{-1}(\vec{x})) = \frac{c}{f - px - qy} \begin{pmatrix} \dfrac{T_1}{c}(f - px - qy) + \omega_2 f - \omega_3 y \\ \dfrac{T_2}{c}(f - px - qy) + \omega_3 x - \omega_1 f \\ \dfrac{T_3}{c}(f - px - qy) + \omega_1 y - \omega_2 x \end{pmatrix}
$$

Defining $m_1 = T_1/c$, $m_2 = T_2/c$, $m_3 = T_3/c$, $m_4 = \omega_1$, $m_5 = \omega_2$, $m_6 = \omega_3$ and:

$$
\vec{V}_1(\vec{x}) = \frac{c}{f - px - qy} \begin{pmatrix} f - px - qy \\ 0 \\ 0 \end{pmatrix}
$$

$$
\vec{V}_2(\vec{x}) = \frac{c}{f - px - qy} \begin{pmatrix} 0 \\ f - px - qy \\ 0 \end{pmatrix}
$$

$$
\vec{V}_3(\vec{x}) = \frac{c}{f - px - qy} \begin{pmatrix} 0 \\ 0 \\ f - px - qy \end{pmatrix}
$$

*It must be emphasized here that this approach could be related to a Kalman filtering approach to allow "maneuvering" objects (non-constant motion), but it is not pursued here.

$$\vec{V}_4(\vec{x}) = \frac{c}{f-px-qy}\begin{pmatrix} 0 \\ -f \\ y \end{pmatrix}$$

$$\vec{V}_5(\vec{x}) = \frac{c}{f-px-qy}\begin{pmatrix} f \\ 0 \\ -x \end{pmatrix}$$

$$\vec{V}_6(\vec{x}) = \frac{c}{f-px-qy}\begin{pmatrix} -y \\ x \\ 0 \end{pmatrix}$$

we get from (20):

$$\vec{u}_1(\vec{x}) = \begin{pmatrix} f-px-qy \\ 0 \end{pmatrix}, \quad \vec{u}_2(\vec{x}) = \begin{pmatrix} 0 \\ f-px-qy \end{pmatrix},$$

$$\vec{u}_3(\vec{x}) = \begin{pmatrix} -\dfrac{x(f-px-qy)}{f} \\ -\dfrac{y(f-px-qy)}{f} \end{pmatrix}$$

$$\vec{u}_4(\vec{x}) = \begin{pmatrix} -\dfrac{xy}{f} \\ -\dfrac{(f^2+y^2)}{f} \end{pmatrix}, \quad \vec{u}_5(\vec{x}) = \begin{pmatrix} \dfrac{(f^2+x^2)}{f} \\ \dfrac{xy}{f} \end{pmatrix},$$

$$\vec{u}_6(\vec{x}) = \begin{pmatrix} -y \\ x \end{pmatrix}$$

Then from (15):

$$\dot{\vec{x}} = \sum_{k=1}^{6} m_k \vec{u}_k(\vec{x})$$

with the parameters $\vec{u}_k$ depending on the shape parameters of the target and on retinal coordinates. Moreover the optical flow (u, v) is expressed as a linear combination of the motion parameters m_i.

Now we can choose a set of linear futures F and from (25) compute h_{ik}. Then we can form the matrix $\mathbf{H}$ and the vector $\vec{f}$ of (23) and estimate the parameters $\vec{m}$ of the 3D motion of the plane from:

$$\vec{m} = [\mathbf{H}^+]\vec{f}.$$

Appendix 1 presents the analysis for more complex shapes.

Tracking

From the algorithm presented in the previous section, we get an estimate of the motion parameters of our target. It remains to see how we use it in order to fulfill the tracking constraint. Using the tracking constraint (14) we reduce the problem of specifying the camera angular velocities that will achieve tracking to an unconstrained optimization problem, namely that of minimizing a cost functional of the form:

$$J(\omega_x, \omega_y) = \alpha\omega_x^2 + \beta\omega_y^2 + \gamma\omega_x + \delta\omega_y + \epsilon\omega_x\omega_y + \zeta \quad (28)$$

with respect to ω_x and ω_y.

It is easy to prove[24] that a cost functional of the form:

$$J(x, y) = \alpha x^2 + \beta y^2 + \gamma x + \delta y + \epsilon xy + \zeta$$

with $\alpha, \beta \geq 0$, α, β not simultaneously zero and $4\alpha\beta - \epsilon^2 \neq 0$ has a unique global minimizer (x^*, y^*) of the form:

$$x^* = \frac{\delta\epsilon - 2\beta\gamma}{4\alpha\beta - \epsilon^2}, \quad (29)$$

$$y^* = \frac{\gamma\epsilon - 2\alpha\delta}{4\alpha\beta - \epsilon^2}, \quad (30)$$

From the tracking constraint (14), the ω_x, ω_y we seek are the minimizers of:

$$J(\omega_x, \omega_y) = \iint_{\mathcal{B}} [(u^t(x, y) + u_{CAM}(x, y))^2 + (v^t(x, y) + v_{CAM}(x, y))^2]\, dx\, dy \quad (31)$$

In the sequel, we attempt to bring (31) into the form (28).

Under the assumption that we know the shape of our target, we derived an estimate of the target motion parameters m_i, $i = 1, \ldots, 6$. We derived also an expression for the *target-induced* optical flow of the form (15):

$$\begin{pmatrix} u^t(x, y) \\ v^t(x, y) \end{pmatrix} = \sum_{i=1}^{6} m_i \begin{pmatrix} u_{i1}(x, y) \\ v_{i2}(x, y) \end{pmatrix} \quad (32)$$

where u_{ij} are *known* functions of the shape of the target (see the example of the planar case in the previous section) and m_i the known estimates of the target motion parameters (where (m_1, m_2, m_3) is the direction of target translation and (m_4, m_5, m_6) is the target rotation). Similarly, the *camera-induced* optical flow will be given by an expression like (32) with $m_1 = 0$, $m_2 = 0$, $m_3 = 0$, $m_4 = -\omega_x$, $m_5 = -\omega_y$ and $m_6 = 0$:

$$\begin{pmatrix} u_{CAM}(x, y) \\ v_{CAM}(x, y) \end{pmatrix} = -\omega_x \begin{pmatrix} u_{41}(x, y) \\ u_{42}(x, y) \end{pmatrix} - \omega_y \begin{pmatrix} u_{51}(x, y) \\ u_{52}(x, y) \end{pmatrix} \quad (33)$$

where u_{41}, u_{42}, u_{51}, u_{52} are known functions of the retinal coordinates, and ω_x, ω_y are the unknown camera rotation parameters. By substituting u^t, v^t, u_{CAM}, v_{CAm} from (32) and (33) to (31), the only unknowns will be ω_x and ω_y and J can be brought to the form:

$$J(\omega_x, \omega_y) = \alpha\omega_x^2 + \beta\omega_y^2 + \gamma\omega_x + \delta\omega_y + \epsilon\omega_x\omega_y + \zeta \quad (34)$$

where:

$$\alpha = \iint_{\mathcal{B}} [u_{41}^2(x, y) + u_{42}^2(x, y)]\, dx\, dy$$

$$\beta = \iint_{\mathcal{B}} [u_{51}^2(x, y) + u_{52}^2(x, y)]\, dx\, dy$$

$$\gamma = -2\iint_{\mathcal{B}} [u^t(x,y)\, u_{41}(x,y) + v^t(x,y)\, u_{42}(x,y)]\, dx\, dy$$

$$\delta = -2\iint_{\mathcal{B}} [u^t(x,y)\, u_{51}(x,y) + v^t(x,y)\, u_{52}(x,y)]\, dx\, dy$$

$$\epsilon = 2\iint_{\mathcal{B}} [u_{41}(x,y)\, u_{51}(x,y) + u_{42}(x,y)\, u_{52}(x,y)]\, dx\, dy$$

$$\zeta = \iint_{\mathcal{B}} [(u^t(x,\ y))^2 + (v^t(x,\ y))^2]\, dx\, dy$$

with $u^t(x,\ y) = \Sigma_{i=1}^{6} m_i u_{i1}(x,\ y)$ and $v^t(x,\ y) = \Sigma_{i=1}^{6} m_i u_{i2}(x,\ y)$. Obviously $\alpha,\ \beta \geq 0$. Simulation results show that they are not simultaneously zero and $4\alpha\beta - \epsilon^2 \neq 0$ (see also the example below). Then, as we saw earlier, the unique minimizers ω_x^* and ω_y^* of (34) are given from (29) and (30), i.e.:

$$\omega_x^* = \frac{\delta\epsilon - 2\beta\gamma}{4\alpha\beta - \epsilon^2} \quad \text{and} \quad \omega_y^* = \frac{\gamma\epsilon - 2\alpha\delta}{4\alpha\beta - \epsilon^2} \tag{35}$$

Example: the planar case

Consider the same planar case as before and apply the above tracking scheme. If our tracking goal is to be accomplished, we expect it to produce a motion of the camera in the same direction as that of the target. Suppose that the point W is near the centre of the image-plane coordinate frame and is moving with a translational velocity T and the target rotates with angular velocity ω and assume that these are known from the estimation part of our algorithm. Then, in the small area $\mathcal{B}$ that we consider around the origin of the image plane, we will have $x \approx 0$ and $y \approx 0$. Therefore, the expressions we found for $\vec{u}_1, \ldots, \vec{u}_6$, will be as follows:

$$\vec{u}_1 \approx \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \vec{u}_2 \approx \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \vec{u}_3 \approx \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

$$\vec{u}_4 \approx \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \quad \vec{u}_5 \approx \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \vec{u}_6 \approx \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

and the optical flow in this area will be:

$$u^t(x,\ y) \approx \frac{T_1}{c} + \omega_2 \quad \text{and} \quad v^t(x,\ y) \approx \frac{T_2}{c} - \omega_1$$

Defining $A \triangleq \iint_{\mathcal{B}} dx\, dy$, the parameters of $J(\omega_x,\ \omega_y)$ in (34) will become:

$$\alpha \approx A, \quad \beta \approx A, \quad \gamma \approx 2\left(\frac{T_2}{c} - \omega_1\right) A,$$

$$\delta \approx -2\left(\frac{T_1}{c} + \omega_2\right) A, \quad \epsilon \approx 0$$

We see that $\alpha,\ \beta \geq 0$ and indeed $\alpha,\ \beta > 0$. Moreover, $4\alpha\beta - \epsilon^2 \approx 4A^2 \neq 0$. Then, from (35) we get:

$$\omega_x^* \approx -\frac{\gamma}{2\alpha} \approx -\frac{T_2}{c} + \omega_1 \quad \text{and} \quad \omega_y^* \approx -\frac{\delta}{2\beta} \approx \frac{T_1}{c} + \omega_2$$

From Figure 2 we can see that this is exactly the motion our camera should perform in order to track the target. In the next section we present experimental results that show how this tracking algorithm actually works in planar and other cases.

Experimental results

Synthetic experiments

Our target is a plane in R^3 with surface normal vector $(-p,\ -q,\ 1)$. We simulate the image formation process using kinematic models of the system motion. We consider the target covered by a grid and for each of the points of the grid we specify the respective projection on the image plane from the perspective projection function, the target kinematics and the position of the point in the target coordinate frame. The image on the image plane is then a textured shape. We consider binary images generated from perspective projection of the moving target. In this paper only the diagram of the distance $\mathbf{o}P_W^*$ *versus* time (see Figure 2) is presented as a measure of the quality of tracking. The simulation of the tracking process consists of a sequence of a target motion followed by a camera tracking action at each time instant. Therefore, motion at even time instants on the diagrams corresponds to target motion, while motion at odd time instants corresponds to camera tracking motion.

The tracking algorithm given above was implemented. It assumes that a feature extraction process found the feature P_W on the image plane during the acquisition phase, but due to noise and computational errors, its position was specified only up to a neighbourhood $\mathcal{B}_W$ and it attempts to stabilize $\mathcal{B}_W$ in the centre of the image plane. The exact formulation of this algorithm for the case of a plane moving in 3D was given as an example above. Very good results were obtained for this tracking scheme fed with "perfect" estimates of the target motion (accurate values for the direction of translation and rotation) for various motions of the target. The results degenerate "gracefully" as the quality of estimates decreases due to very fast motion of the target or noisy images. A sequence of "noisy" target motion estimates was generated in order to test the performance of the algorithm under inaccurate motion estimates. The algorithm performs well with 30% to 60% noise in target motion estimation; even in the case when some of the parameters are

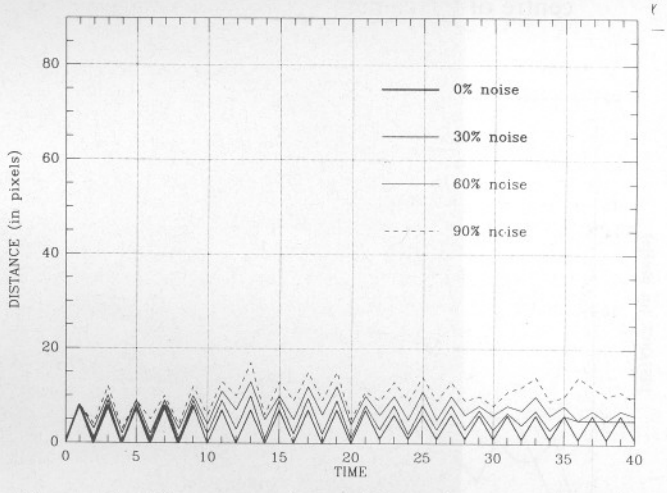

Figure 3. Target tracker with noisy target motion parameter estimation. General motion of the target (translational and rotational)

*P_w is the projection on the image plane of the point W of the target; o is the origin of the image plane coordinate system.

estimated with 90% deviation from real values, it performs satisfactorily enough. The case of a characteristic target motion is presented in Figure 3 ($T_1 = 20$, $T_2 = 30$, $T_3 = 5$, $\omega_1 = 2$, $\omega_2 = 3$, $\omega_3 = 5$), where the target translational velocities are given in (length units) per (time unit) and the angular velocities in (degrees) per (time unit). The image resolution is 91×91.

Several sets of linear features were considered before establishing experimentally that the best 3D motion parameter estimation using the above algorithm is obtained with moments of the image, i.e. with linear features of the form:

$$\iint_{\mathcal{B}} x^i y^j s(x, y, t)\, dx\, dy, \quad 0 \leqslant i + j \leqslant k.$$

The quality of the motion parameter estimation depends on the order k of the moments considered. Moments up to order $k = 10$ were considered and in certain cases significant improvement in performance was obtained from experiments with moments of order up to $k = 4$ or less.

The results that follow simulate the target motion parameter estimator given above, combined with the tracking algorithm for an image with resolution 21×21:

(a) Only translational motion of the target ($T_1 = 30$). Experimental results for moments up to order $k = 2$ and $k = 10$ are presented in Figure 4. Small deviation of the target from the centre of the camera is observed.
(b) Only rotation of the target ($\omega_2 = 5$). Experimental results for moments up to order $k = 2$ and $k = 10$ are presented in Figure 5.
(c) General motion of target ($T_1 = 40$, $\omega_2 = 2$, $\omega_3 = 5$). Experimental results for moments up to order $k = 5$ are presented in Figure 6a.
(d) General motion of target ($T_1 = 20$, $T_2 = 0$, $T_3 = 5$, $\omega_1 = 0$, $\omega_2 = -2$, $\omega_3 = 10$). Experimental results for moments up to order $k = 2$ are presented in Figure 6b. The target moves along the line of sight and the tracking algorithm does not cause it to deviate from the centre of the camera.

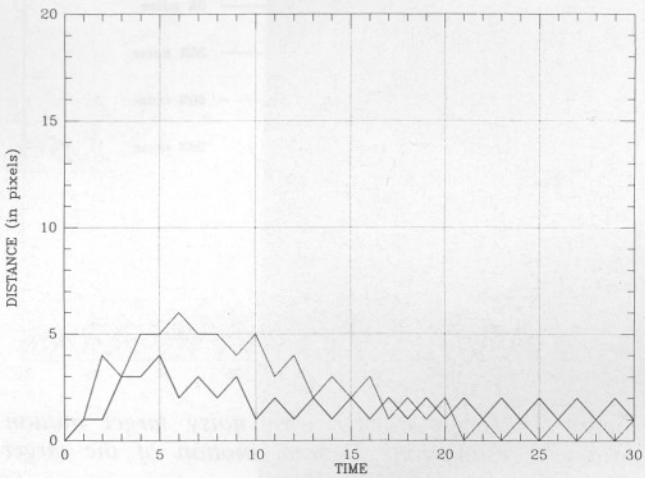

Figure 4. Motion parameter estimator (using moments up to orders $k = 2$ and $k = 10$) and target tracker. Translational target motion

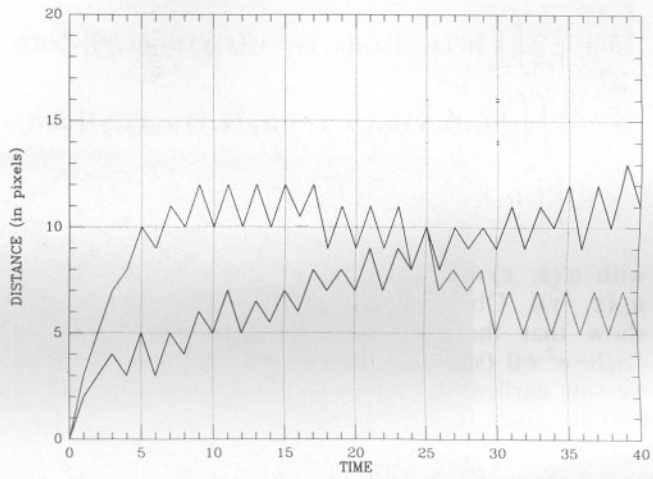

Figure 5. Motion parameter estimator (using moments up to orders $k = 2$ and $k = 10$) and target tracker. Rotational target motion

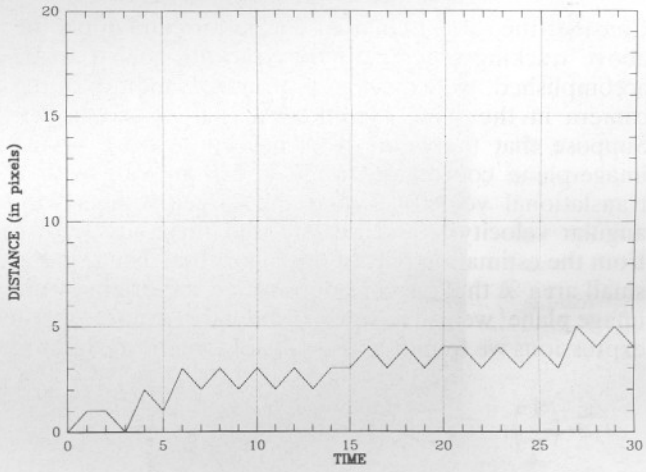

Figure 6. Experimental results (see text) ———: a; ———: b

Implementation issues

Errors are introduced in this tracking scheme from the following sources:

(1) Discretization effects, which will affect mainly the temporal derivatives of the linear features.
(2) Numerical instabilities in the solution of the linear system $\mathbf{H}\vec{m} = \vec{f}$.
(3) The coarse resolution of the image used in simulations. Due to this a translation may have the same visual effect as a specific rotation, and *vice versa*. This may affect the motion estimation algorithm, but ultimately it will not affect tracking. Using the fine resolution of a real system, this effect is expected to disappear.

The use of moments as linear features has the advantage that they are easily implementable in hardware[48], therefore the above scheme can be used for real-time applications. Moreover, multiprocessor architectures such as the fine-grain Connection Machine[49] are an interesting alternative for the real-time computation of the matrix $\mathbf{H}$ and the vector of linear features $\vec{f}$ (see above), which are the computational bottlenecks of the above algorithm.

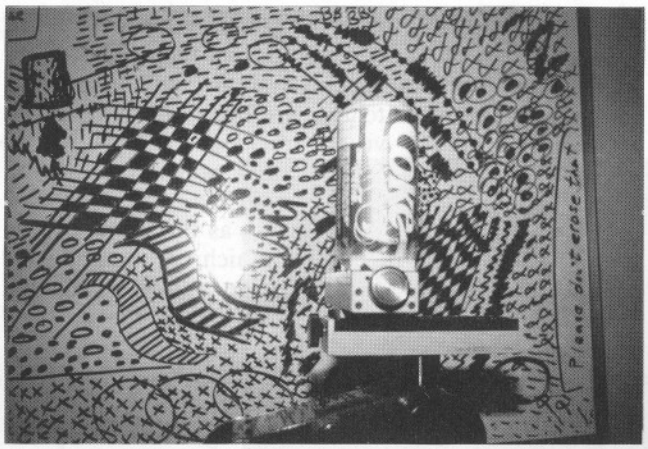

Figure 7. A Mitsubishi programmable robot arm holding the target (soft drink can)

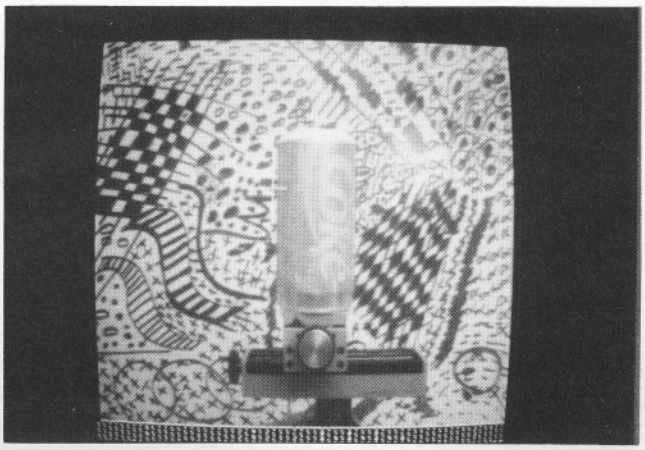

Figure 10. Image of target as seen by the tracker

A robotic implementation

In order to demonstrate the successfulness of the approach, we implemented the tracking algorithm in a robotic environment built in our laboratory. The task was the following: a programmable Mitsubishi robot arm held a soft drink can which it could move towards any direction. (Figure 7 shows the robot holding the can against a background of a white board that we intentionally painted with texture, for reasons to be explained later.) On the hand, near the first robot, an American Merlin robot arm was placed (see Figure 8), having six rotational joints and a sphere of operation approximatley two meters in diameter. A Sony DC-37 CCD miniature television camera with a 16 mm focal length lens was attached to the arm. The task then was to have the American Merlin robot arm track the can held by the first robot, which could move towards any direction (see Figure 9, which shows part of the "seeing" arm on the left "watching" the arm that holds the object to be tracked). Figure 10 shows the image of the object to be tracked as seen by the tracker. Since the theory works for both differentiable and non-differentiable image functions, in order to reduce the amount of data that needs to be processed, we work directly with the discontinuities (zero-crossings). This is why we painted the background with texture. The reason was to make it harder to identify the "moving" contours. Edge detection of an image such as in Figure 10, with some thresholding, provides us with an edge map such as in Figure 11. However, with the process of dynamic segmentation we identify the moving edges (Figure 12), and such successive frames constitute the input to our tracking process, assuming that the edge ponts lie on a cylindrical surface (and using the mathematics described in the Appendix). Because the analysis of the images taken by the tracker had to be done on a VAX 785 and primitive image operations on a VICOM image processor, all linked through a network, a real time implementation was impossible due to the heavy communication bottleneck. For this reason we chose to present results from the experi-

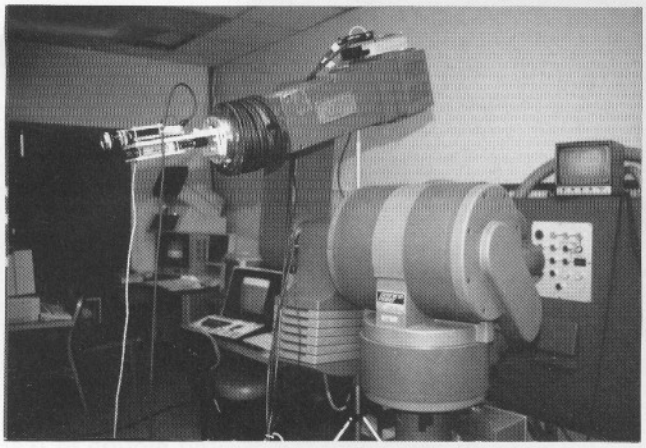

Figure 8. An American Merlin robot arm equipped with TV camera (tracker)

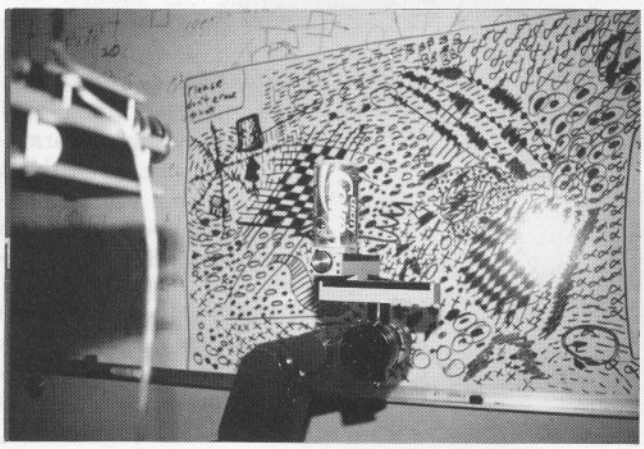

Figure 9. The tracker (left) and the target (right)

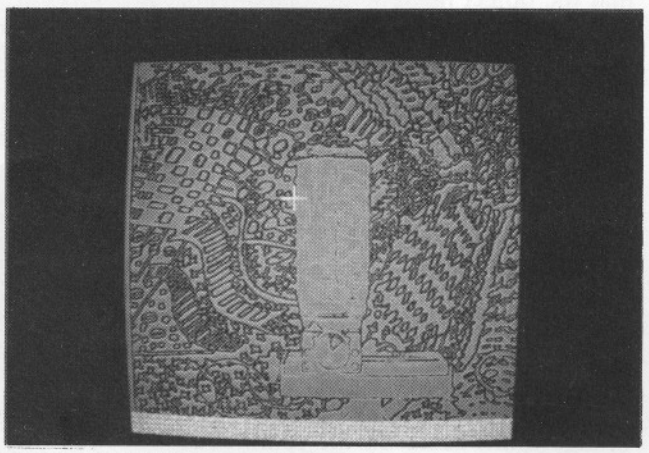

Figure 11. Edge map of scene

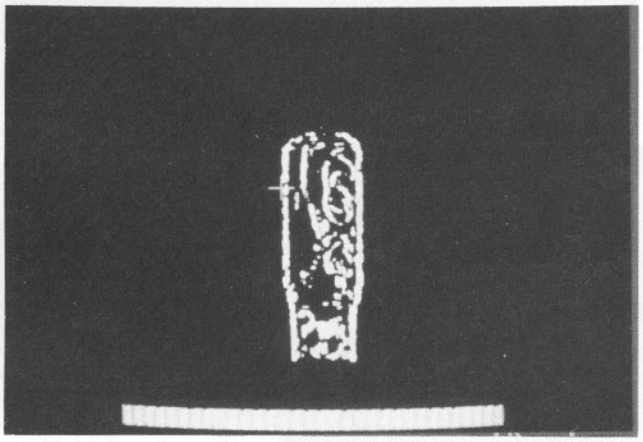

Figure 12. Moving edges (target)

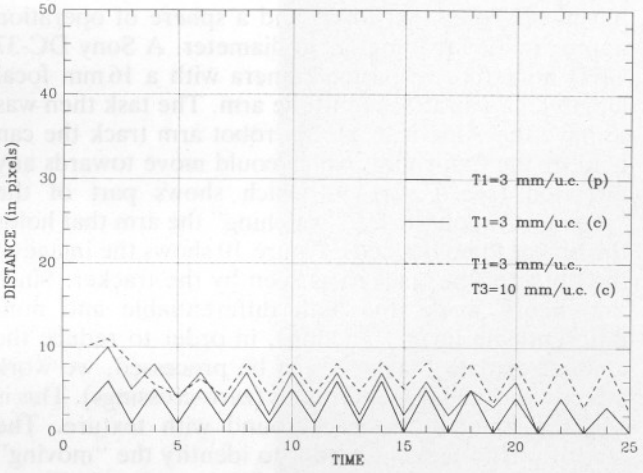

Figure 13. Target tracker (using moments up to $k = 5$)

ments in the same format as in the case of the synthetic simulations. Figure 13 shows some of the results. The vertical axis represents the distance oP_w as before, and the horizontal axis represents time. As before, the tracking process consists of a sequence of target motion, followed by a camera tracking action in each time instant. Thus, motion at even time instants on the diagrams corresponds to target motion, while motion at odd time instants corresponds to camera tracking motion. Figure 13 shows results from three experiments. In the first (solid line), the actual motion was translation along the x-axis and under the assumption that the object in view was planar (instead of cylindrical) in order to examine the deviation of the results as a function of deviation from the local shape assumption. Clearly the results are not sensitive to this assumption. The next two experiments (dotted and dashed lines) show results under horizontal motion (dotted) and motion in the x–y plane (dashed). It is quite clear that the results are very satisfactory.

CONCLUSIONS AND FUTURE RESEARCH

The problem examined in this paper was the tracking phase of visual tracking of three-dimensional targets moving in three dimensions, which is the most general tracing situation in practice. We devised a general class of algorithms based on the continuous motion formalism, which, contrary to previous approaches, use dynamic segmentation* of the image sequence and are correspondence-free. Therefore, two major shortcomings of most previous tracking systems, namely segmentation and feature matching, are bypassed and a novel formulation of the visual tracking problem as an optimization problem is presented, which leads to the *tracking constraint*. This expresses the camera angular velocity that will achieve tracking as the minimizer of an appropriate cost functional, which requires that for the tracking to be successful, the optical flow in a neighbourhood of the origin of the image plane has to be minimized.

The tracking constraint and the optical flow constraint are used to create an algorithmic scheme for tracking where we assume knowledge of the shape of the target. The optical flow can be expressed as a linear combination of the motion parameters (direction of translation and rotation), which can be estimated using linear features of the image, which in turn provide us with global, correspondence-free information about the temporal variation of the image. The linear feature formulation allows us not to compute numerically the spatial derivatives of the image intensity function, which is an ill-posed problem, but use instead the spatial derivatives of the corresponding measuring functions, which can be computed analytically.

Simulation results demonstrate the success of this algorithm in tracking 3D targets, even under very noisy information like very coarse image resolution and erroneous motion parameter estimation. Experimental results in a robotic environment demonstrate the promise of the approach.

Hardware implementation and testing of our algorithms in a real-time visual environment is one of our primary future research goals. The image moment invariants which we used as linear features have the advantage of being easily implementable on special hardware or computable on multiprocessor computer systems such as the Connection Machine. Therefore the computational bottleneck, which in our experiments was the computation of the linear features vector and the matrix **H** (see above), can be eliminated and real-time application of the algorithms becomes possible. These algorithms will generate a desired trajectory for the camera actuators to follow, and therefore appropriate modelling and control schemes are necessary in order to implement this desired motion in real time.

A 3D shape estimation scheme was presented[50] which assumes knowledge of the 3D motion and estimates shape, following a very similar approach to the one used in our 3D motion estimation algorithm. A cooperative scheme based on these two algorithms may be able to achieve tracking in the case when only approximate shape and approximate motion information are available *a priori*. Interaction between these two algorithms, together with additional visual information, may be able to improve the shape and motion estimates as time evolves, until some kind of equilibrium is reached.

Provided we have an estimate of the target 3D motion from the algorithm presented above, this information can be used not just for tracking, but as part of an augmented system able to acquire a target

*Using the temporal derivative to identify the moving target.

(track and approach it) and/or perform obstacle avoidance. Apart from obvious obstacle avoidance applications in robot navigation, this may have other interesting applications such as grabbing a rotating satellite based on visual information or combining visual and tactile information in an intelligent control scheme[50,51] that will allow a robot to approach, grasp and manipulate a moving object in a flexible manufacturing environment.

Even though linear features have been proven a very important tool both in classical pattern recognition and in novel correspondence-free motion and shape estimation algorithms, like the ones presented here, no theoretical investigation has been undertaken on the relative merits of specific measuring functions μ_i and the optimal length k of the linear feature vector. The experimental attempt has been to make this vector as rich in image-related information as possibsle, but the exact meaning of "richness" has never been stated formally. A formal measure of the richness of information depending on the selection of μ_i and k will possibly enable the use of our powerful optimization tools in order to derive optimality results applicable to each specific visual situation.*

Finally, the whole previous approach was related to rigid targets. Humans are perfectly able to track the motion of non-rigid ones, such as clouds, flags, sea waves and other humans, even though their shape or motion parameters may not be exactly known. This is related to the understanding of non-rigid motion from visual information and several researchers are already studying this problem e.g. see Reference 53. Another interesting problem is to extend this theory to deal with local shape functions that change continuously through time. However, this is a hard problem due to ambiguities between 3D motions that generate the same local shape changes.

ACKNOWLEDGEMENTS

The authors would like to thank Dr P S Krishnaprasad for helpful comments and discussions. Also, the thoughtful comments of the referees greatly improved the contents and presentation of the paper.

REFERENCES

1 **Kern, R, Kugel, U and Hettlage, E** 'Control of a pointing, acquisition and tracking subsystem for intersatellite laser links (ISL^2)', *SPIE Vol 810, Optical Systems for Space Applications* (1987)

2 **Wilcox, B, Gennery, D B, Bon, B and Litwin, T** 'Real-time model-based vision system for object acquisition and tracking', *SPIE Vol 754* (1987)

3 **Gennery, D B** 'Stereo vision for the acquisition and tracking of moving three-dimensional objects', in: **A Rosenfeld**, *Techniques for 3-D Machine Perception*, North Holland (1986)

4 **Gilbert, A L, Giles, M K, Flachs, G M, Rogers, R B and U, Y H** 'A real-time video tracking system', *IEEE Trans. PAMI* Vol 2 No 1 (1980)

5 **Bouthemy, P and Benveniste, A** 'Modelling of atmospheric disturbances in meteorological pictures', *IEEE Trans. PAMI* Vol 6 No 5 (1984)

6 **Venetsanopoulos, A N and Cappellini, V** 'Real-time image processing', in: **S G Tzofestos (ed.)**, *Systems: Techniques and Applications*, Marcel Dekker, USA (1986)

7 **Legters, G R and Young, T Y** 'A mathematical model for computer image tracking', *IEEE Trans. PAMI* Vol 4 No 6 (1982)

8 **Nagalia, S** 'Real-time acquisition of objects in motion', Center for Automation Research, CS-TR-1398 (1984)

9 **Rajala, S A, Riddle, A N and Snyder, W L** 'Application of the one-dimensional Fourier transform for tracking moving objects in noisy environments', *CVGIP* Vol 21 (1983) pp 280–293

10 **Schalkoff, R J and McVey, E S** 'A model and tracking algorithm for a class of video targets', *IEEE Trans. PAMI* Vol 4 No 1 (1982)

11 **Roach, J W and Aggarwal, J K** 'Computer tracking of objects moving in space', *IEEE Trans. PAMI* Vol 1 No 2 (1979)

12 **Gilbert, A L** 'Video data conversion and real-time tracking', *IEEE Comput.* (1981) pp 50–56

13 **Wallace, T P and Mitchell, O R** 'Analysis of three-dimensional movement using Fourier descriptors', *IEEE Trans. PAMI* Vol 2 No 6 (1980) pp 583–588

14 **Cornog, K** 'Smooth pursuit and fixation for robot vision', MS Thesis, MIT AI Laboratory, USA (1985)

15 **Gennery, D B** 'Tracking known three-dimensional objects', *Proc. AAAI 2nd Nat. Conf. on AI* Pittsburgh, PA (1982) pp 13–17

16 **Bernstein, H J** 'Constraints in real-time data acquisition and control', TR-195, Courant Institute of Mathematical Sciences, NYU, USA (December 1985)

17 **Hunt, A E and Sanderson, A C** 'Vision-based predictive robotic tracking of a moving target', Report of the Robotics Institute, Carnegie-Mellon University, USA

18 **Spetsakis, M E and Aloimonos, J** 'Optimal computing of structure from motion using point correspondences in two frames', *Proc. 2nd IEEE ICCV*, Tarpon Springs, FL, USA (1988)

19 **Adiv, G** 'Inherent ambiguities in recovering 3D motion and structure from a noisy flow field', *Proc. IEEE CVPR* (1985) pp 70–77

20 **Aloimonos, J and Shulman, D** *Integration of Visual Modules: An Extension of the Marr Paradigm*, Academic Press, Boston (1989)

21 **Poggio, T, Gamble, E B and Little, J J** 'Parallel integration of visual modules', *Science* Vol 242 (1989) pp 436–440

22 **Marr, D** *Vision*, W H Freeman and Co., USA (1982)

23 **Arnold, V I** *Mathematical Methods of Classical Mechanics*, Springer-Verlag, New York (1978)

24 **Tsakiris, D P** 'Visual tracking strategies', MSc Thesis, Electrical Engineering Department, University of Maryland (1988)

25 **Horn, B K P and Schunk, B G** 'Determining optical flow', *Artif. Intell.* Vol 17 (1981) pp 185–203

*It must be noted, however, that finding optimal linear features (optimal in the sense that the resulting error will be minimal) is an open problem in Functional Analysis (G W Stewart, private communication).

26 **Aloimonos, J and Schulman, H** 'Learning early vision computations', *J. Opt. Soc. America A* Vol 6 (1987) pp 908–919

27 **Brockett, R W** 'Gramians, generalized inverses and the least-squares approximation of optical flow', *Proc. IEEE Conf. on Robotics and Automation* (1987) pp 1834–1841

28 **Cornelius, N and Kanade, T** 'Adapting optical flow to measure object motion in reflectance and X-ray image sequences', *Proc. ACM SIGGRAPH/ SIGGART Interdisciplinary Workshop on Motion: Representation and Perception*, Toronto, Canada (1983) pp 50–58

29 **Hildreth, E C** 'Computations underlying the measurement of visual motion', *Artif. Intell.* Vol 23 (1984) pp 309–354

30 **Nagel, H-H and Enkelmann, W** 'An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences', *IEEE Trans. PAMI* Vol 8 No 5 (1986)

31 **Poggio, T, Torre, V and Koch, C** 'Computer vision and regularization theory', *Nature*, Vol 317 (1985) pp 314–319

32 **Schunk, B G** 'The image flow constraint equation', *CVGIP* Vol 35 (1986) pp 20–46

33 **Terzopoulos, D** 'Multilevel reconstruction of visual surfaces: variational principles and finite-element representation', in: **A Rosenfeld (ed.)** *Multiresolution Image Processing and Analysis*, Springer-Verlag (1984) pp 237–310

34 **Terzopoulos, D** 'Regularization of inverse visual problems involving discontinuities', *IEEE Trans. PAMI* Vol 8 No 4 (1986) pp 413–424

35 **Waxman, A and Wohn, K** 'Contour evolution, neighbourhood deformation and global image flow: planar surfaces in motion', *Int. J. Robotics Res.* Vol 4 No 3 (1985) pp 95–108

36 **Abu-Mostafa, Y S and Psaltis, D** 'Recognitive aspects of moment invariants', *IEEE Trans. PAMI* Vol 6 No 6 (1984) pp 698–706

37 **Amari, S** 'Feature spaces which admit and detect invariant signal transformations', *Proc. 4th Int. Joint Conf. on Pattern Recognition*, Tokyo, Japan (1978) pp 452–456

38 **Dudani, S A, Breeding, K B and McGhee, R B** 'Aircraft identification by moment invariants', *IEEE Trans. Comput.* Vol 26 No 1 (1977)

39 **Hu, M K** 'Visual pattern recognition by moment invariants', *IRE Trans.* Vol 8 (1962) pp 179–187

40 **Rosenfeld, A and Kak, A** *Digital Picture Processing*, Academic Press, USA (1982)

41 **Teague, M R** 'Image analysis via the general theory of moments', *J. Opt. Soc. Am.* Vol 70 No 3 (1980) pp 920–929

42 **Teh, C H and Chin, R T** 'On image analysis by the methods of moments', *IEEE Trans. PAMI* Vol 10 No 4 (1988)

43 **Ito, E and Aloimonos, J** 'Determining three dimensional transformation parameters from images: theory', *Proc. IEEE Conf. on Robotics and Automation* (1987)

44 **Ben-Israel, A and Greville, T N E** *Generalized Inverses: Theory and Applications*, John Wiley, UK (1974)

45 **Bandopadhay, A.** 'A computational study of rigid motion perception', PhD Thesis, Department of Computer Science, University of Rochester (1986)

46 **Longuet-Higgins, H G and Prazdny, K** 'The interpretation of a moving retinal image', *Proc. R. Soc. Lond. B* Vol 208 (1980) pp 385–397

47 **Prazdny, K** 'Determining the instantaneous direction of motion from optical flow generated by a curvilinearly moving observer', *CVGIP* Vol 17 (1981) pp 238–248

48 **Cagney, F and Mallon, J** 'Real-time feature extraction using moment invariants', *SPIE Vol 716* (1986)

49 **Hillis, W D** 'The connection machine: a computer architecture based on cellular automata', *Physica* (1984) pp 213–218

50 **Aloimonos, J, Weiss, I and Bandopadhay, A** 'Active vision', *Int. J. Comput. Vision* (1988) pp 333–356

51 **Fu, K S** 'Learning control systems and intelligent control systems: an intersection of artificial intelligence and automatic control', *IEEE Trans. Aut. Contr.* Vol 16 (1971) pp 70–72

52 **Saridis, G N** 'Intelligent robotic control', *IEEE Trans. Aut. Contr.* Vol 28 No 5 (1983)

53 **Schulman, D and Aloimonos, J** '(Non)rigid motion interpretation: a regularized approach', *Proc. R. Soc. Lond. B* Vol 233 (1988) pp 217–234

54 **Bahill, A T and LaRitz, T** 'Why can't batters keep their eyes on the ball?', *American Scientist* Vol 72 (1984) pp 219–253

55 **Bahill, A T and McDonald, J D** 'Model emulates human smooth pursuit system producing zero-latency target tracking', *Biol. Cybern.* Vol 48 (1983) pp 213–222

56 **Poggio, T and Koch, C** 'Ill-posed problems in early vision: from computational theory to analog networks', *Proc. R. Soc. Lond. B* Vol 226 (1985) pp 303–323

APPENDIX I

Consider again a pinhole camera as the camera model and a quadratic surface as the shape model. Then:

$$\vec{\mathbf{x}} = \begin{pmatrix} x \\ y \end{pmatrix} = P(\vec{\mathbf{X}}) = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$$

$$Z = g_{\mathbf{q}}(X, Y) = \mathbf{q}_0 X^2 + \mathbf{q}_1 XY + \mathbf{q}_2 Y^2 + \mathbf{q}_3 x + \mathbf{q}_4 y + \mathbf{q}_5$$

Solving the above with respect to $\vec{x}$, we get:

$$\vec{\mathbf{X}} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = P^{-1}_{g(\mathbf{q})}(\vec{\mathbf{x}}) = \lambda(\mathbf{q}, \vec{\mathbf{x}}) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

where:

$$\lambda(\mathbf{q}, \vec{\mathbf{x}}) = \frac{2q_5}{\beta - (\beta^2 - 4\alpha)^{1/2}}, \quad \alpha = \mathbf{q}_0 X^2 + \mathbf{q}_1 XY + \mathbf{q}_2 Y^2, \quad \beta = \mathbf{q}_3 X + \mathbf{q}_4 Y - 1$$

(Since there are two solutions, the solution closer to the camera is chosen). Since $P^{-1}_{g(\mathbf{q})}$ is the inverse projection, substituting into (19) and (15) we get:

$$\vec{u_k}(\vec{\mathbf{x}}) = \sum_k m_k \frac{\partial P}{\partial \mathbf{X}} (P^{-1}_{g(\mathbf{q})}(\vec{\mathbf{x}}))\, \vec{V}_k(P^{-1}_{g(\mathbf{q})}(\vec{\mathbf{x}}))$$

We can now express explicitly in terms of $\mathbf{q}$ and m the optic flow when the object is moving rigidly. Following the same reasoning as in the planar case, we get:

$$\dot{\vec{\mathbf{x}}} = \begin{bmatrix} 1 & 0 & -x \\ 0 & 1 & -y \end{bmatrix} \left(\hat{\omega} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} + \frac{1}{\lambda(\mathbf{q}, \vec{\mathbf{x}})} \mathbf{T} \right)$$

or:

$$\dot{\vec{\mathbf{x}}} = \begin{bmatrix} -xy & (1+x^2) & -y & \dfrac{1}{\lambda(\mathbf{q}, \vec{\mathbf{x}})} & 0 & \dfrac{-x}{\lambda(\mathbf{q}, \vec{\mathbf{x}})} \\ -1-y^2 & xy & x & 0 & \dfrac{1}{\lambda(\mathbf{q}, \vec{\mathbf{x}})} & \dfrac{-y}{\lambda(\mathbf{q}, \vec{\mathbf{x}})} \end{bmatrix} \begin{bmatrix} \omega \\ \mathbf{T} \end{bmatrix},$$