From: AAAI-82 Proceedings. Copyright ©1982, AAAI (www.aaai.org). All rights reserved.

TRACKING KNOWN THREE-DIMENSIONAL OBJECTS*

Donald B. Gennery

Robotics and Teleoperator Group
Jet Propulsion Laboratory
Pasadena, California 91109

ABSTRACT

A method of visually tracking a known three-dimensional object is described. Predicted object position and orientation extrapolated from previous tracking data are used to find known features in one or more pictures. The measured image positions of the features are used to adjust the estimates of object position, orientation, velocity, and angular velocity in three dimensions. Filtering over time is included as an integral part of the adjustment, so that the filtering both smooths as appropriate to the measurements and allows stereo depth information to be obtained from multiple cameras taking pictures of a moving object at different times.

I INTRODUCTION

Previous work in visual tracking of moving objects has dealt mostly with two-dimensional scenes [1, 2, 3], with labelled objects [4], or with restricted domains in which only partial spatial information is extracted [5]. This paper describes a method of tracking a known solid object for which an accurate object model is available, determining its three-dimensional position and orientation rapidly as it moves, by using natural features on the object. Only the portion of the tracking problem concerning locking onto an object and tracking it when given initial approximate data is discussed here. The acquisition portion of the problem is currently being worked on and will be described in a later paper. Since the tracking proper portion discussed here has approximate information available from the acquisition data or from previous tracking data, it can quickly find the expected features in the pictures, and it can be optimized to use these features to produce high accuracy, good coasting through times of poor data, and optimum combining of information obtained at different times. (An earlier, similar method lacking many of the features described here was previously reported [6].)

The current method uses a general object model consisting of planar surfaces. The features found in the pictures are the brightness edges formed by the intersection of the planar faces of the object, caused by differences in illumination on the different faces. By comparing the positions of the actual features in the pictures to their predicted postions, discrepancies are generated that are used in a least-squares adjustment (based on a linearization using partial derivatives) to refine the current estimates of object position and orientation. Filtering over time is included in the adjustment to further reduce error by smoothing (including different amounts of smoothing automatically in different spatial directions as required by the accuracy of the data), to obtain velocities for prediction, and to enable information obtained at different times to be combined optimally. Thus stereo depth information is obtained when more than one camera is used, even though individual feartures are not tracked or matched between pictures, and even if the different cameras take pictures at different times. When only one camera is used, the approximate distance to the object is still determined, because of its known size. In order to avoid the singularity in the Euler angle representation (and for other reasons mentioned later), the current orientation of the object is represented by quaternions, and the incremental adjustment to orientation is represented by an infinitesimal rotation vector. (Corben and Stehle [7] provide a discussion of quaternions, and Goldstein [8] provides a discussion of infinitesimal rotation vectors.)

The tracking program works in a loop with the following major steps: prediction of the object position and orientation for the time at which a picture is taken by extrapolating from the previously adjusted data (or from acquisition data when starting), detection of features by projecting into the picture to find the actual features and to measure their image positions relative to the predictions; and the use of the resulting data to adjust the position, orientation, and their time derivatives so that the best estimates for the time of the picture are obtained. These steps will be described briefly in the following sections. A more detailed description will appear in a paper published elsewhere.

II PREDICTION

The prediction of position and orientation is based upon the assumption of random acceleration

* The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

and angular acceleration (that is, a constant power spectrum to frequencies considerably higher than the rate of picture taking). Since random acceleration implies constant expected velocity, the predicted position itself is obtained simply by adding to the position estimate from the previous adjustment the product of the previous adjusted velocity times the elapsed time since the previous picture, for each of the three dimensions. Similarly, the predicted orientation is obtained by rotating from the previous adjusted orientation as if the previous adjusted angular velocity vector applied constantly over the the elapsed time interval. (This orientation extrapolation is particularly simple when quaternions are used.) The predicted velocity and angular velocity are simply equal to the previous adjusted values. However, these predicted values must have appropriate weight in the adjustment, and, since the weight matrix should be the inverse of the covariance matrix (see, for example, Mikhail [9]), the computation of the covariance matrix of the predicted data will now be discussed.

The covariance matrix of the previous adjusted data is denoted by S. This is a 12-by-12 matrix, since there are three components of position, three components of incremental rotation, three components of velocity, and three components of angular velocity (assumed to be arranged in that order in S). To a first approximation, the covariance matrix $\tilde{S}$ of the predicted data can be obtained by adding to S terms to represent the additional uncertainty caused by the extrapolation. These must include both of the following: terms to increase the uncertainty in position and orientation caused by uncertainty in the velocity and angular velocity that were used to do the extrapolation, and terms to increase the uncertainty in velocity and angular velocity caused by the random acceleration and angular acceleration occuring over the extrapolation interval. The former effect can be produced by using the following 12-by-12 transformation matrix:

$$A = \begin{bmatrix} I & 0 & \tau I & 0 \\ 0 & I & 0 & \tau I \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix}$$

where I is the 3-by-3 identity matrix and τ is the elapsed time interval of the extrapolation. Then the covariance matrix can be transformed by this matrix, and additional terms can be added for the latter effect, as follows:

$$\tilde{S} \approx ASA^T + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & a\tau I & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha\tau I \end{bmatrix}$$

where a and α are the assumed values of the power spectra of acceleration and angular acceleration, respectively, and the superscript T denotes the matrix transpose. (Mikhail [9] provides the necessary background information on matrix algebra.) The larger are the values of a and α, the larger will be the uncertainty in the predicted values as indicated by $\tilde{S}$, and thus the less smoothing over time will be produced in the adjustment. In practice, the above matrix multiplications are multiplied out so that the actual computations are expressed in terms of 3-by-3 matrices. This is computationally faster since A is so sparse.

However, for greater accuracy two additional effects are included in the implemented program. First, the effect on orientation of uncertainty in the previous orientation and angular velocity will be influenced by the rotation that has occured during the time τ. This causes some modification of the A matrix. Second, additional terms involving a and α are added to $\tilde{S}$ to reflect the influence that the random acceleration during the just elapsed time interval τ has on position and orientation. These refinements will be described in another paper.

III <u>DETECTION OF FEATURES</u>

Once the predicted object position and orientation are available for a picture, the vertices in the object model that are predicted to be visible (with a margin of safety) are projected into the picture by using the known camera model [10]. The lines in the picture that correspond to edges of the object are computed by connecting the appropriate projected vertices. Analytical partial derivatives of the projected quantities with respect to the object position vector and object incremental infinitesimal rotation vector are also computed.

Brightness edges are searched for near the positions of the predicted lines. The brightness edges elements are detected by a modified Sobel operator (including thresholding and thinning), which we have available both in software form and in special hardware that operates at the video rate [11]. The program only looks for edge elements every three pixels along the line, since the Sobel operator is three pixels wide. For each of these positions it searches approximately perpendicularly to the line. Currently it accepts the nearest edge element to the predicted line, if it is within five pixels. However, a more elaborate method has been devised. This new method varies the extent of the search according to the accuracy of the predicted data, accepts all edge elements within the search width, and gives the edge elements variable weight according to their agreement with the predicted line and their intensity. This method will be described in a later paper.

In principle, the position of each detected edge element could be used directly in the adjustment described in the next section. The observed quantity e_i would be the perpendicular distance from the predicted line to the detected edge element, the 1-by-6 partial derivative matrix B_i would be the partial derivatives of $-e_i$ with respect to the three components of object position and three components of incremental object rotation, and the weight w_i of the observation would be the reciprocal of the square of its standard deviation (accuracy). (Currently, this standard deviation is a given quantity and is assumed to be the same for all edge elements.)

However, for computational efficiency the program uses a mathematically equivalent two-step process. First, a corrected line is computed by a least-squares fit to the perpendicular discrepancies from the predicted line. In effect, the quantities obtained are the perpendicular corrections to the predicted line at its two end vertices, which form the 2-by-1 matrix E_i, and the corresponding 2-by-2 weight matrix W_i. B_i is then the 2-by-6 partial derivative matrix of $-E_i$ with respect to the object position and incremental orientation. Second, these quantities for each predicted line are used in the adjustment described in the next section.

IV <u>ADJUSTMENT</u>

Now the nature of the adjustment to position and orientation will be discussed. If no filtering were desired, a weighted least squares solution could be done, ignoring the predicted values except as initial approximations to be corrected. The standard way of doing this [9, 12] is as follows:

$$N = \sum_i B_i^T W_i B_i$$

$$C = \sum_i B_i^T W_i E_i$$

$$P = \tilde{P} + N^{-1} C$$

where B_i is the matrix of partial derivatives of the ith set of observed quantities with respect to the parameters being adjusted, W_i is the weight matrix of the ith set of observed quantities, E_i is a vector made up of the ith set of observed quantities, P is the vector of parameters being adjusted, and $\tilde{P}$ is the initial approximation to P. The covariance matrix of P, which indicates the accuracy of the adjustment, is then N^{-1}. For the case at hand, P is 6-by-1 and is composed of the components of position and incremental orientation, N is 6-by-6, and C is 6-by-1. The meanings of E_i, W_i, and B_i for this case were described in the previous section.

The velocity and angular velocity are included in the adjustment by considering twelve adjusted parameters consisting of the six-vectors P and V, where V is composed of the three components of velocity and the three components of angular velocity. The measurements which produce N and C above contribute no information directly to V. However, the predicted values $\tilde{P}$ and $\tilde{V}$ can be considered to be additional measurements directly on P and V with covariance matrix $\tilde{S}$, and thus weight matrix $\tilde{S}^{-1}$. (Giving the predicted values weight in the solution produces the filtering action, similar to a Kalman filter, because of the memory of previous measurements contained in the predicted information.) Therefore, the adjustment including the information contained in the predicted values in principle could be obtained as follows:

$$S = \left(\begin{bmatrix} N & 0 \\ 0 & 0 \end{bmatrix} + \tilde{S}^{-1} \right)^{-1}$$

$$\begin{bmatrix} P \\ V \end{bmatrix} = \begin{bmatrix} \tilde{P} \\ \tilde{V} \end{bmatrix} + S \begin{bmatrix} C \\ 0 \end{bmatrix}$$

However, using the above equation is inefficient and may present numerical problems, since the two matrices to be inverted are 12-by-12 and may be nearly singular. If $\tilde{S}$ is partitioned into 6-by-6 matrices as follows,

$$\tilde{S} = \begin{bmatrix} \tilde{S}_{PP} & \tilde{S}_{PV} \\ \tilde{S}_{PV}^T & \tilde{S}_{VV} \end{bmatrix}$$

the following mathematically equivalent form in terms of 6-by-6 matrices and 6-vectors can be produced by means of some matrix manipulation:

$$S_{PP} = (I + \tilde{S}_{PP} N)^{-1} \tilde{S}_{PP}$$

$$S_{PV} = (I + \tilde{S}_{PP} N)^{-1} \tilde{S}_{PV}$$

$$S_{VV} = \tilde{S}_{VV} - \tilde{S}_{PV}^T N (I + \tilde{S}_{PP} N)^{-1} \tilde{S}_{PV}$$

$$P = \tilde{P} + S_{PP} C$$

$$V = \tilde{V} + S_{PV}^T C$$

Not only is this form more efficient computationally, but the matrix to be inverted $(I + \tilde{S}_{PP} N)$ is guaranteed to be nonsingular, because both $\tilde{S}_{PP}$ and N are non-negative definite.

The first three elements of P from the above form the new adjusted position vector of the object. The last three elements form an incremental rotation vector used to correct the object orientation. This could be used directly to update the rotation matrix (as explained by Goldstein [8]), but, since the primary representation of orientation in the implemented tracker is in terms of quaternions, it is used instead to update the quaternion that represents orientation, and the rotation matrix is computed from that. This method also makes convenient the normalization to prevent accumulation of numerical error. (The relationship between quaternions and rotations is described by Corben and Stehle [7].) The covariance matrix S of the adjusted data is formed by assembling S_{PP}, S_{PV}, S_{PV}^T, and S_{VV} into a 12-by-12 matrix, similarly to $\tilde{S}$.

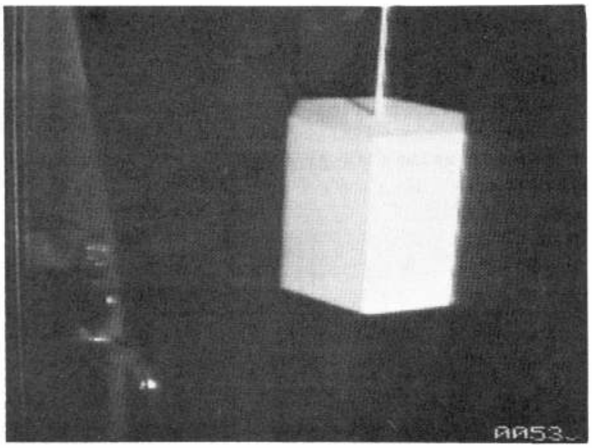

Figure 1. Digitized picture from left camera.

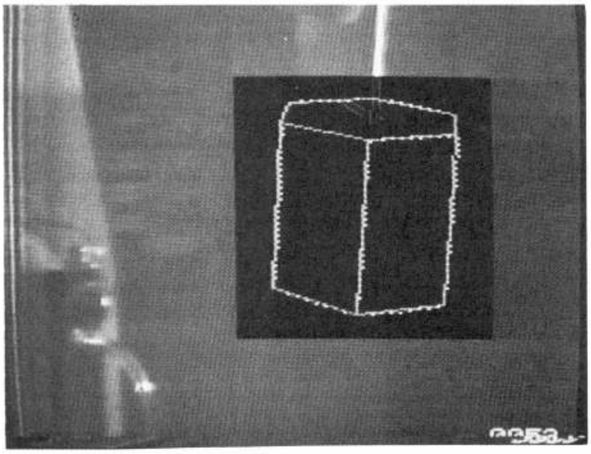

Figure 2. Results from Figure 1.

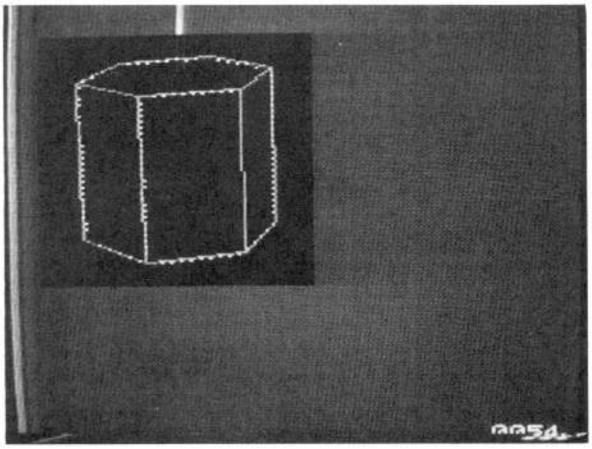

Figure 3. Results from next picture from right camera.

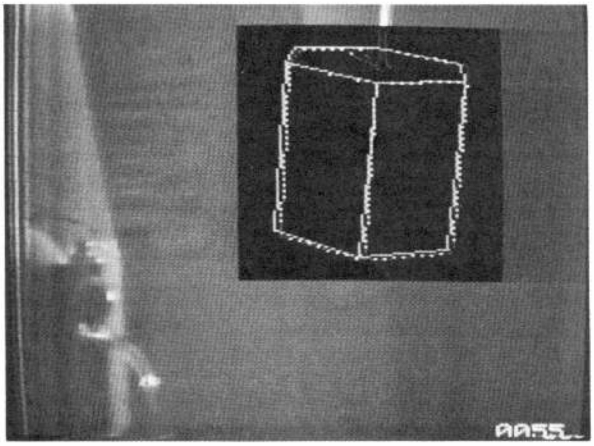

Figure 4. Results from next picture from left camera.

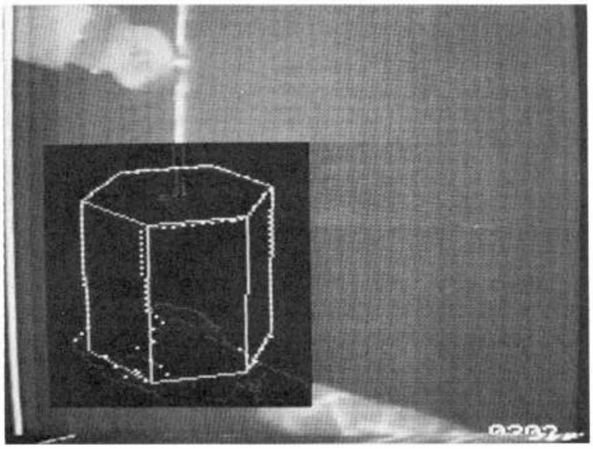

Figure 5. Results from right camera with obscuring object.

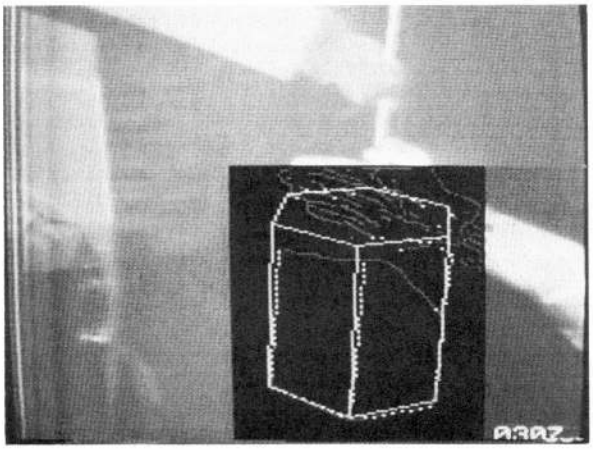

Figure 6. Results from left camera five pictures later.

V RESULTS

Figures 1, 2, 3, and 4 show the tracker in action. The object being tracked (a hexagonal prism) is 203 mm tall and is moving upwards at about 16 mm/sec. Pictures from two cameras were taken alternately. The values used for the acceleration parameters were $a = 1$ mm^2/sec^3 and $\alpha = 0.0001$ radian^2/sec^3. The assumed standard deviation of the edge measurements was one pixel. The software version of the edge detector was used. The program, which runs on a General Automation SPC-16/85 computer, was able to process each picture in this example in 1.6 seconds, so that the complete loop through both cameras required 3.2 seconds. (When the hardware edge detector is used, the time per picture in a case such as this is only 0.5 second.)

Figure 1 shows the raw digitized image corresponding to Figure 2. For successive pictures from the left, right, and left cameras, respectively, Figures 2, 3, and 4 show the following information. In a window that the program puts around the predicted object for applying the software edge detector, the raw digitized picture has been replaced by the detected brightness edges (showing as faint lines). (With the hardware edge detector the entire picture would be so replaced.) Superimposed on this are the predicted lines corresponding to edges of the object (showing as brighter lines). The bright dots are the edge elements which were used in the adjustment. (These may be somewhat obscured in the figures when they lie directly on the predicted lines.)

The program is able to tolerate a moderate amount of missing and spurious edges. This is because it looks for edges only near their expected positions, because the typical abundance of edges produces considerable overdetermination in the adjustment, and because of the smoothing produced by the filtering. Figures 5 and 6 (similar to Figures 2, 3, and 4) show an example of an obscuring object passing in front of the tracked object without causing loss of track. Figure 5 is from the right camera, and Figure 6 is from the left camera five pictures later (so that there are two pictures from the left camera and two from the right camera between these in time that are not shown).

ACKNOWLEDGMENTS

The programming of the tracker was done primarily by Eric Saund, with portions by Doug Varney and Bob Cunningham. Bob Cunningham assisted in conducting the tracking experiments.

REFERENCES

[1] H.-H. Nagel, "Analysis Techniques for Image Sequences," Fourth International Joint Conference on Pattern Recognition, Tokyo, November 1978, pp. 186-211.

[2] W. N. Martin and J. K. Aggarwal, "Dynamic Scene Analysis," Computer Graphics and Image Processing **7** (1978), pp. 356-374.

[3] A. L. Gilbert, M. K. Giles, G. M. Flachs, R. B. Rogers, and Y. H. U, "A Real-Time Video Tracking System," IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-2** (1980), pp. 47-56.

[4] H. F. L. Pinkney, "Theory and Development of an On-Line 30 Hz Video Photogrammetry System for Real-Time 3-Dimensional Control," Proceedings of the ISP Symposium on Photogrammetry for Industry, Stockholm, August 1978.

[5] J. W. Roach and J. K. Aggarwal, "Computer Tracking of Objects Moving in Space," IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-1** (1979), pp. 127-135.

[6] E. Saund, D. B. Gennery, and R. T. Cunningham, "Visual Tracking in Stereo," Joint Automatic Control Conference, sponsored by ASME, University of Virginia, June 1981.

[7] H. C. Corben and P. Stehle, Classical Mechanics (Second Edition), Wiley, 1960.

[8] H. Goldstein, Classical Mechanics (Second Edition), Addison-Wesley, 1980.

[9] E. M. Mikhail (with contributions by F. Ackermann), Observations and Least Squares, Harper and Row, 1976.

[10] Y. Yakimovsky and R. T. Cunningham, "A System for Extracting Three-Dimensional Measurements from a Stereo Pair of TV Cameras," Computer Graphics and Image Processing **7** (1978), pp. 195-210.

[11] R. Eskenazi and J. M. Wilf, "Low-Level Processing for Real-Time Image Analysis," Jet Propulsion Laboratory Report 79-79.

[12] D. B. Gennery, "Modelling the Environment of an Exploring Vehicle by Means of Stereo Vision," AIM-339, STAN-CS-80-805, Computer Science Dept., Stanford University, 1980.