Camera Trajectory Estimation using Inertial Sensor Measurements and Structure from Motion Results

Sang-Hack Jung Camillo J. Taylor
GRASP Laboratory, CIS Department
University of Pennsylvania
Philadelphia, PA, 19104-6228

Abstract

This paper describes an approach to estimating the trajectory of a moving camera based on the measurements acquired with an inertial sensor and estimates obtained by applying a structure from motion algorithm to a small set of keyframes in the video sequence. The problem is formulated as an offline trajectory fitting task rather than an online integration problem. This approach avoids many of the issues usually associated with inertial estimation schemes. One of the main advantages of the proposed technique is that it can be applied in situations where approaches based on feature tracking would have significant difficulties. Results obtained by applying the procedure to extended sequences acquired with both conventional and omnidirectional cameras are presented.

1. Introduction

This paper considers the problem of estimating the trajectory of a moving camera by combining the measurements obtained from an inertial sensor with the results obtained from a structure from motion algorithm. In the proposed scheme, the structure from motion algorithm is used to estimate the position and orientation of the camera at a small set of keyframes in a video sequence, typically three or four in a sequence of several hundred. These results are then combined with the angular velocity and translational acceleration readings obtained from the inertial sensor to estimate the pose of the camera with respect to a fixed coordinate system.

The ability to obtain an accurate estimate for the camera trajectory could be used in a number of different contexts. Many stereo based reconstruction systems, for example, involve estimating camera motion prior to recovering scene structure. The camera trajectory estimate could also be used to combine range maps obtained from various vantage points into a single frame of reference, a critical step in model construction. [1]

In the motion picture industry, estimates for camera trajectory are often used in move matching applications so that virtual elements can be inserted into actual video sequences. These estimates are sometimes acquired by instrumenting the camera platform with sophisticated motion control systems. The proposed technique may offer a simpler alternative.

This pose estimation scheme can also be applied to the problem of recovering the trajectory of an omnidirectional camera as it is moved through an immersive environment like an office complex. The resulting omnidirectional video sequence, augmented with pose could then be explored interactively to provide users with virtual tours of the space.

The problem of recovering camera trajectories solely from image data has received a great deal of attention in the computer vision literature. Recently Hsu et al [5], Fitzgibbon and Zisserman [3], Chiuso et al. [2] and Pollefeys et al [7] have all proposed vision based systems for pose estimation. Commercial systems such as MatchMover by REALVIZ are also currently available.

All of these approaches rely quite heavily on the ability to track distinctive features in the scene throughout the sequence. There are many situations, such as moving a camera from room to room in an office complex, where the features of interest may become occluded or undergo radical changes in appearance as the camera moves. In these cases, the problem of constructing a reliable feature tracker becomes exceedingly difficult.

Another issue that can pose a serious problem for image based estimation schemes is the assumption that one can detect and track a subset of features that are fixed with respect to the base frame of reference. If one were interested in estimating the trajectory of a camera as it is moved through a crowded train station it would be quite challenging to automatically select an appropriate set of features in the presence of multiple

[1] Note that these range maps could, of course, come from sources other than a stereo system.

ISBN 0-7695-1272-0/01 $10.00 (C) 2001 IEEE

independently moving targets. The proposed approach avoids these problems by exploiting information from an independent source of positioning information, the inertial sensors.

The problem of estimating the trajectory of a moving platform based on measurements from an inertial sensor has been tackled by a number of researchers. Azuma and Bishop [1], You et al [9] and List [6] dealt with the problem of using inertial sensors to estimate and predict the motion of a head mounted display for an immersive virtual reality system.

In these cases, the primary goal has been to provide an accurate, *online* estimate for the moving platform based on the inertial data. Here one faces the difficulties attendant with integrating noisy, biased accelerometer measurements. The problem considered in this paper is quite different in that the main goal is to provide an *offline* trajectory estimation system. This allows us to reformulate the problem in terms of trajectory fitting rather than integration. This approach allows us to overcome many of the noise and drift issues that one usually associates with inertial sensors.

2. Approach

Figure 1 shows a schematic of the trajectory estimation problem. In this figure, there are three relevant frames of reference, the fixed world frame of reference, **W**, the cameras frame of reference **C** and the accelerometers frame of reference, **A**. The camera

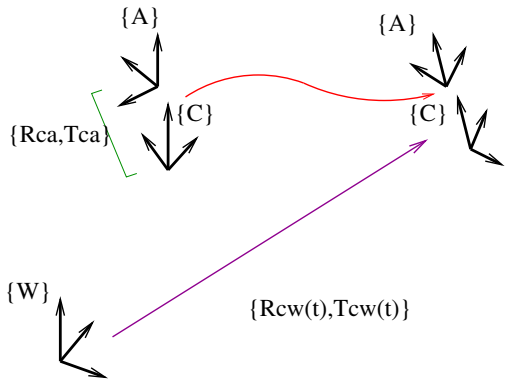

Figure 1: Schematic of the layout

and the inertial sensor are affixed to one another as shown in Figure 2 and the relationship between them is captured by the rotation matrix $R_{ca} \in SO(3)$ and a translation vector $\mathbf{T}_{ca} \in \Re^3$. The changing relationship between the camera and the world frame is represented by the parameters $R_{cw}(t) \in SO(3)$ and $T_{cw}(t) \in \Re^3$.

The vectors ω_a, $a_a \in \Re^3$ denote the instantaneous angular velocity and translational acceleration of the

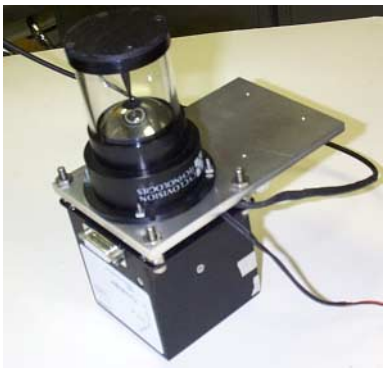

Figure 2: The omnidirectional camera and accelerometer setup

inertial sensor. Measurements for these parameters are returned by the inertial sensor. Note that these parameters are measured with respect to the sensors instantaneous frame of reference.

A number of effective schemes for recovering the pose of the camera and the structure of the scene from image correspondences have been proposed [4, 8, 10]. Typically, these schemes return estimates that are good up to an unknown scale factor, λ, unless additional metric information is available. In the sequel we will assume that one of these schemes is employed to recover estimates for the camera pose at selected keyframes in the sequence.

Once this has been done, the cameras trajectory is estimated in a two step process. The first step focuses on the problem of recovering the cameras orientation, $R_{cw}(t)$, based on the angular velocity measurements. The second step returns an estimate for the cameras translation, $T_{cw}(t)$.

2.1. Recovering Camera Orientation

Given the measurements for the accelerometers angular velocity one can compute the angular velocity of the camera as follows $\omega_c = R_{ca}\omega_a$. It is assumed that the structure from motion system can be used to obtain estimates for the cameras orientation at two keyframe instants, t_1 and t_2. One can then form an estimate for the cameras orientation between these instants by observing that the rotational displacement of the camera between subsequent inertial measurements, $\Delta R(t)$, can be approximated by exponentiating the skew symmetric matrix formed from the angular velocity vector in the usual manner:

$$\Delta R(t) = \exp(skew(\omega_c(t))\Delta t) \qquad (1)$$

This approximation corresponds to the camera moving with a constant angular velocity throughout the

ISBN 0-7695-1272-0/01 $10.00 (C) 2001 IEEE

measurement interval. Since the sampling period associated with the inertial sensor is typically quite small (5 ms for our experimental system) this approximation is usually acceptable.

This being the case, one can estimate the rotational displacement of the camera at a given time $t \in [t_1 t_2]$ by accumulating the rotational displacements in an appropriate fashion as shown in Equation 3.

$$R_{cw}(t) = R_{cw}(t_1)\Pi_{\tau=t_1}^{t}\Delta R(\tau) \quad (2)$$

$$= R_{cw}(t_1)\Pi_{\tau=t_1}^{t}\exp(skew(\omega_c(\tau)\Delta t)) \quad (3)$$

The series of matrix multiplications implied by this expression can be carried out efficiently using quaternion multiplication.

In a perfect world, the total rotational displacement between the keyframes instants, t_1 and t_2, calculated from the angular velocity measurements would correspond to the displacement between the two rotation estimates returned by the SFM algorithm. In practice, these estimates will differ slightly due to measurement errors and drift. For the system that has been implemented this discrepancy is quite small, less than half a degree over a trajectory of 30 seconds.

We can correct for this (small) disagreement by forming two estimates for the rotation at time t one constructed by starting at time t_1 and integrating forward, $R^1_{cw}(t)$, and another constructed by starting at time t_2 and integrating backward, $R^2_{cw}(t)$. The final estimate, $R_{cw}(t)$ is obtained by forming the moral equivalent of a weighted average of the two rotations using the following expression:

$$R_{cw}(t) = \exp(\frac{(t-t_1)}{(t_2-t_1)}\ln(R^2_{cw}(t)(R^1_{cw}(t))^T))R^1_{cw}(t) \quad (4)$$

The term $\ln(R^2_{cw}(t)(R^1_{cw}(t))^T)$ corresponds to the matrix logarithm of the discrepancy between the two rotation estimates.

The advantage of this scheme is that it produces a smooth rotational trajectory that agrees with the given rotations at the endpoints and jibes with the observed angular velocities.

2.2. Recovering Camera Translation

Once the camera orientation has been recovered, it remains to estimate the translational component of the trajectory. The most straightforward approach to computing the translation would be to simply integrate the accelerometer readings twice. Unfortunately this approach is completely unworkable due to the errors typically associated with accelerometer measurements.

Firstly, the accelerometer readings often contain non-zero bias terms. While these biases may appear to be small, when they are integrated twice the result is an error component that increases as t^2. For example, a bias of $0.1ms^{-2}$ in an accelerometer reading will lead to a positioning error of 5m after 10 seconds. Secondly, the accelerometer values are corrupted with random noise which when integrated corresponds to a random walk through space that would be added to the actual camera path.

Another issue that can complicate integration based schemes is the fact that for most camera trajectories, the accelerations induced by the cameras motion are small when compared with the omnipresent gravitational field. This can present a problem since efforts to compensate for the gravity vector by subtracting it from the raw signal can introduce significant errors. More specifically, small errors in the estimate for the cameras orientation with respect to the world frame can cause the system to misestimate the direction of the gravitational field, which will lead to errors in the reconstituted acceleration measurements.

These problems can be overcome by noting that the procedure in question is intended for use as an offline trajectory estimation system. This allows us to reformulate the problem as a trajectory fitting problem rather than an integration problem.

We begin by modeling the camera trajectory as a piecewise polynomial curve as shown in Figure 3. The trajectory is divided into a number of epochs as shown in Figure 3 and within each epoch the cameras motion along each axis is modeled using the simple second order polynomial[2] given by Equation 5.

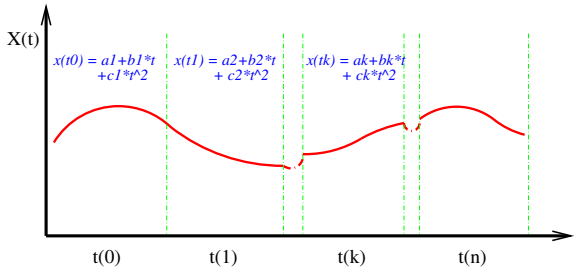

Figure 3: The spline representation of a single axis.

$$\tau = (t - iT)/T \quad t \in [iT, (i+1)T]$$

$$\begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix} = \begin{pmatrix} a_i^x + b_i^x\tau + c_i^x\tau^2 \\ a_i^y + b_i^y\tau + c_i^y\tau^2 \\ a_i^z + b_i^z\tau + c_i^z\tau^2 \end{pmatrix} \quad (5)$$

Continuity constraints are imposed at the boundaries between the epochs to ensure that the resulting

[2] We can also use cubic spline which gives equivalent results under smooth motion.

ISBN 0-7695-1272-0/01 $10.00 (C) 2001 IEEE

trajectory is continuous in both position and velocity. Equations 6 and 7 reflect the continuity constraints on position and velocity respectively.

$$\begin{pmatrix} a_i^x + b_i^x + c_i^x \\ a_i^y + b_i^y + c_i^y \\ a_i^z + b_i^z + c_i^z \end{pmatrix} = \begin{pmatrix} a_{i+1}^x \\ a_{i+1}^y \\ a_{i+1}^z \end{pmatrix} \quad (6)$$

$$\begin{pmatrix} b_i^x + 2c_i^x \\ b_i^y + 2c_i^y \\ b_i^z + 2c_i^z \end{pmatrix} = \begin{pmatrix} b_{i+1}^x \\ b_{i+1}^y \\ b_{i+1}^z \end{pmatrix} \quad (7)$$

Note that these continuity constraints take the form of homogeneous equations in the spline parameters. Given these constraints, the resulting trajectory has a total of $3n + 6$ degrees of freedom where n denotes the total number of epochs.

The duration of the epochs, T, and the degree of the spline polynomial should reflect the designers expectations about the nature of the camera motion. Increasing the number of epochs provides the system with the ability to model more complex trajectories. Conversely, decreasing the number of epochs reduces the number of degrees of freedom in the model, which usually improves the conditioning of the resulting estimation problem. An appropriate balance must be struck between these competing considerations.

In the proposed scheme the SFM algorithm is used to estimate the position of the cameras at a set of keyframes in the sequence. These provide constraints on the resulting camera trajectory, which must be satisfied.

$$\begin{pmatrix} x(t_j) \\ y(t_j) \\ z(t_j) \end{pmatrix} = \begin{pmatrix} a_i^x + b_i^x \tau_j + c_i^x \tau_j^2 \\ a_i^y + b_i^y \tau_j + c_i^y \tau_j^2 \\ a_i^z + b_i^z \tau_j + c_i^z \tau_j^2 \end{pmatrix} = \lambda \begin{pmatrix} \hat{x_j} \\ \hat{y_j} \\ \hat{z_j} \end{pmatrix} \quad (8)$$

The scalar parameter, λ, in Equation 8 reflects the fact that the SFM algorithm can only recover the locations of the camera up to a constant, unknown scale factor. The parameter t_j refers to the time of keyframe j while τ_j refers to the normalized time parameter used in the spline representation, $\tau_j = (t_j - iT)/T$. Note that these constraint can be rewritten as linear homogeneous equations in the spline parameters and λ as shown below.

$$\begin{pmatrix} a_i^x + b_i^x \tau_j + c_i^x \tau_j^2 - \lambda \hat{x_j} \\ a_i^y + b_i^y \tau_j + c_i^y \tau_j^2 - \lambda \hat{y_j} \\ a_i^z + b_i^z \tau_j + c_i^z \tau_j^2 - \lambda \hat{z_j} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad (9)$$

Given this model for the cameras motion it is a simple matter to compute what the expected acceleration in each epoch would be as a function of the spline parameters.

$$a_w = \begin{pmatrix} \ddot{x}(t) \\ \ddot{y}(t) \\ \ddot{z}(t) \end{pmatrix} = \frac{2}{T^2} \begin{pmatrix} c_i^x \\ c_i^y \\ c_i^z \end{pmatrix} \quad (10)$$

Once again the predicted accelerations are a linear function of the trajectory parameters.

These predictions can be related to the measurements obtained from the accelerometer system. Equation 11 shows how the estimated acceleration in the world frame $\hat{a}_w$ would be calculated from the readings.

$$\hat{a}_w(t) = R_{cw}(t) R_{ca} (a_a(t) - b - \omega_a(t) \times (\omega_a(t) \times T_{ca})) - g \quad (11)$$

In this equation the vector b refers to the unknown but constant accelerometer bias. The term $\omega_a(t) \times (\omega_a(t) \times T_{ca})$ accounts for the centripetal acceleration experienced by the camera center due to the angular velocity of the accelerometer. The constant vector g represents the gravitational field.

Given Equations 10 and 11, the trajectory estimation task can be reformulated as the problem of choosing values for the spline parameters, the bias term, b, and the scale factor, λ, that minimize the total discrepancy between the predicted acceleration $a_w(t)$ and the measured values $\hat{a}_w(t)$ over the entire sequence.

$$\min \Sigma_{k=1}^{m} \| a_w(t_k) - \hat{a}_w(t_k) \|^2 \quad (12)$$

The summation in Equation 12 is carried out over all of the accelerometer readings indexed by k; t_k denotes the time at which the kth inertial sensor measurement was acquired.

Note that the vector difference $(a_w(t_k) - \hat{a}_w(t_k))$ is actually an *affine* function of the parameters of interest. More specifically, $a_w(t_k)$ is a linear function of the spline parameters c_x^i, c_y^i, and c_z^i (Equation 10) and $\hat{a}_w$ is an affine function of the bias acceleration, b (Equation 11). This implies that the minimization task set forth in Equation 12 actually corresponds to a linear least squares problem with a set of homogeneous linear constraints on the parameters. This system can be solved quite easily using standard techniques from linear algebra[3] to obtain estimates for the spline parameters, the unknown accelerometer bias, b, and the overall scale parameter λ.

[3] Matlab provides a function, lsqlin(), for solving problems of this form, which was used in this work.

ISBN 0-7695-1272-0/01 $10.00 (C) 2001 IEEE

3. Experimental Results

3.1. Omnidirectional Video Sequences

In order to investigate the efficacy of the proposed approach, it was applied to data sets collected with an omnidirectional camera system.

The structure from motion system used in these experiments makes use of point and line correspondences in the omnidirectional frames. These correspondences are specified manually in a selected set of keyframes using a point and click interface. The system then automatically computes estimates for the camera pose and scene structure based on these estimates.

In the first sequence (see Fig. 4) the camera system was moved 6.1 meters over 26 seconds down a hallway. The sequence consists of 165 frames and 5218 accelerometer measurements. For this experiment the trajectory was divided into 80 epochs.

The camera trajectory was estimated based on three keyframes in the sequence. In order to verify the accuracy of the reconstructed trajectory, four more keyframes were chosen from the sequence and the SFM algorithm was applied to estimate the camera pose at four more keyframes. These estimates were then compared to the reconstructed camera trajectory. The mean disparity between the camera orientation predicted by the camera trajectory and the estimates returned by the SFM system was 0.77 degrees, the maximum disparity was 1.15 degrees. The mean disparity between the position estimates was 4.5cm with a maximum error of 16.4cm. Figure 5 shows plots of the reconstructed camera trajectory. The estimates for the keyframe locations returned by the SFM algorithm are denoted by circles.

The same procedure was applied to the omnidirectional sequence shown in Figure 6 which contains large rotational motions . In this case the mean disparity between the camera orientation obtained from the trajectory estimate and the results obtained from the pose estimation system was 1.2 degrees with a maximum error of 1.65 degrees. The mean disparity between the position estimates was 8.2cm with a maximum error of 22.1cm. The trajectory results are shown in Figure 7.

Note that in both sequences the camera travels a significant distance so that the first and last frames of the sequence are remarkably dissimilar. The extent of the motion and the presence of independently moving objects, such as the people seen in Figure 6, makes it difficult to apply standard feature tracking algorithms to these data sets.

Figure 4: Frame 0, 140, 210(=last) from the first omnidirectional video sequence

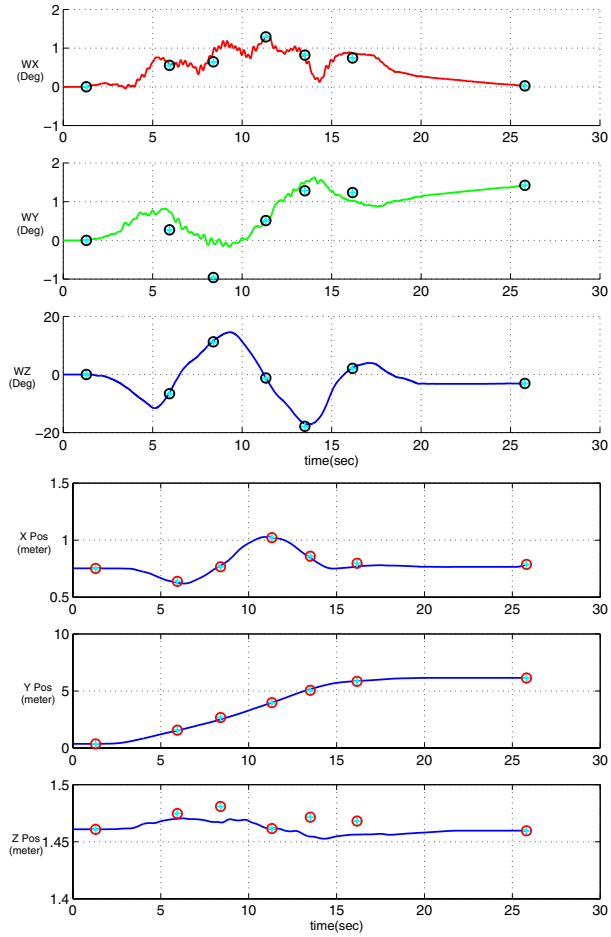

Figure 5: Camera trajectory plots of the first experiment - the rotation and translation parameters for each axis with keyframes marked by circles.

Figure 6: Frame 0, 80, 166(=last) from the second omnidirectional video sequence

ISBN 0-7695-1272-0/01 $10.00 (C) 2001 IEEE

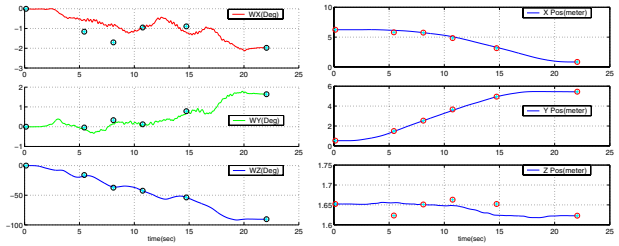

Figure 7: Camera trajectory plots of the second experiment - the rotation and translation parameters for each axis with keyframes marked by circles.

3.2. Conventional Video Sequences

Another set of experiments was carried out using a more conventional camera system (see Fig. 8). For these experiments the camera pose was estimated using the method described in [10]. An 11 second sequence containing 32 frames was considered. The camera trajectory was estimated based on 3 keyframes (see Fig. 9). This trajectory was then compared to the estimates obtained for 29 other keyframes. The mean error in orientation was 0.42 degrees with a maximum error of 0.8 degrees. The mean error in position was 1.05cm with a maximum error of 3.34cm.

Figure 8: Frame 0, 11, 22, 32(=last) from the conventional video sequence

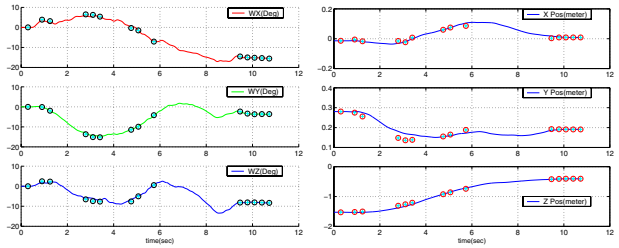

Figure 9: Camera trajectory plots of the conventional camera experiment - the rotation and translation parameters for each axis with keyframes marked by circles.

4. Conclusions

This paper describes an approach to estimating the trajectory of a moving camera by applying a structure from motion algorithm to a small set of keyframes in the sequence and combining these results with the measurements obtained from an onboard accelerometer system. The trajectory estimation process is broken into two stages, the first stage recovers estimates for the cameras orientation by considering the angular velocity measurements returned by the unit. The second stage recovers the trajectory of the cameras center of projection. The trajectory is modeled as a spline which is constrained to pass through the keyframe locations. The parameters of the spline are then chosen so that the predicted accelerations agree with the measurements obtained by the accelerometers. This scheme avoids many of the problems usually associated with integrating accelerometer measurements and can be applied in situations where feature based tracking approaches would have significant difficulties.

Acknowledgments : This material is based upon work supported by the National Science Foundation under a CAREER Grant (Grant No. 9875867).

References

[1] R. Azuma and G. Bishop. Improving static and dynamic registration in an optical see-through hmd. In *Proc. of SIGGRAPH*, pages 197–204, 1994.

[2] Alessandro Chiuso, Paolo Favaro, Hailin Jin, and Stefano Soatto. 3-d motion and structure from 2-d motion causally integrated over time: Implementation. In *European Conference on Computer Vision*, volume 2, pages 734–750, 2000.

[3] A.W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *European Conference on Computer Vision*, pages 311–326, June 1998.

[4] Richard I. Hartley. In defense of the eight-point algorithm. *IEEE Trans. Pattern Anal. Machine Intell.*, 19(6):580, June 1997.

[5] Stephen Hsu, Supun Samaraskera, Rakesh Kumar, and Harpreet S. Sawhney. Pose estimation, model refinement, and enhanced visualization using video. In *CVPR*, volume 1, page 488, June 2000.

[6] Uwe H. List. Nonlinear prediction of head movements for helmet-mounted display. In *Technical Report AFHRL-TP-83-45*, 1984.

[7] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *International Conference on Computer Vision*, pages 90–95, 1998.

[8] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[9] S. You, U. Neumann, and R. Azuma. Hybrid inertial and vision tracking for augmented reality registration. In *Proc. of IEEE Virtual Reality*, pages 260–267, 1999.

[10] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *International Conference on Computer Vision*, pages 666–673, 1999.

ISBN 0-7695-1272-0/01 $10.00 (C) 2001 IEEE