Vision-based Pose Computation: Robust and Accurate Augmented Reality Tracking

Jun Park, Bolan Jiang, and Ulrich Neumann
Computer Science Department
University of Southern California
{junp|bjiang|uneumann}@usc.edu

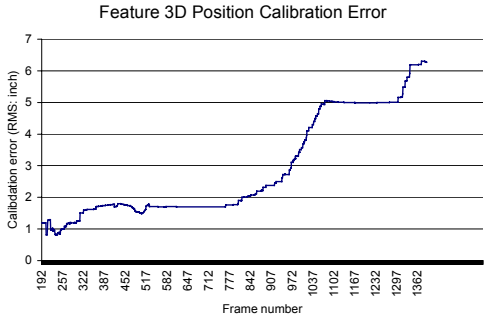
Abstract

Vision-based tracking systems have advantages for augmented reality (AR) applications. Their registration can be very accurate, and there is no delay between the motions of real and virtual scene elements. However, vision-based tracking often suffers from limited range, intermittent errors, and dropouts. These shortcomings are due to the need to see multiple calibrated features or fiducials in each frame. To address these shortcomings, features in the scene can be dynamically calibrated and pose calculations can be made robust to noise and numerical instability. In this paper, we survey classic vision-based pose computations and present two methods that offer increased robustness and accuracy in the context of real-time AR tracking.

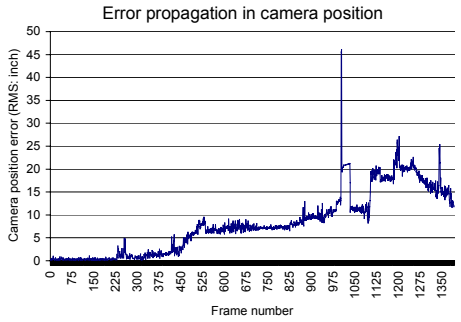
1. Introduction

Vision-based tracking systems have advantages for video see-through augmented reality (AR). First, the same video camera used to capture real scenes also serves as a tracking device. Second, the pose calculation is most accurate in the image plane, thereby minimizing the perceived image alignment error. Additionally, processing delays in the video and graphics subsystems can be matched, thereby eliminating dynamic alignment errors. Tracking in vision-based systems is often relative to objects of interest, allowing these to move in the environment [NEUM96].

The operating range of vision-based tracking systems is limited to areas where a minimum number of calibrated features (landmarks or fiducials) are in view. Partial occlusion of these features, even when the area of users' interest is in view, may cause failure or errors in tracking. More robust and dynamically *extendible tracking* can be achieved by dynamically calibrating the 3D positions of uncalibrated fiducials or natural features [PARK98], however the effectiveness of this approach depends on the behavior of the pose calculations. Experiments show that tracking errors propagate rapidly for extendible tracking when the pose calculation is sensitive to noise or otherwise unstable. Fig.1 shows how errors in camera position increase as dynamic pose and calibration errors propagate to new scene features. In this simulated experiment, the system started tracking with 6 calibrated features. The camera was then panned and rotated while the system estimated the positions of 94 initially uncalibrated features placed in a 100”x30”x20” volume. Fig. 1a shows the average errors of the dynamically estimated features. Fig. 1b shows the errors in the camera position computed from the estimated features. After about 500 frames (~16 seconds) the five inch accumulated error exceeds 5% of the largest operating volume dimension. This performance may be adequate to compensate for several frames of fiducial occlusion, but it does not allow significant tracking area extension.



a. Feature 3D position calibration error

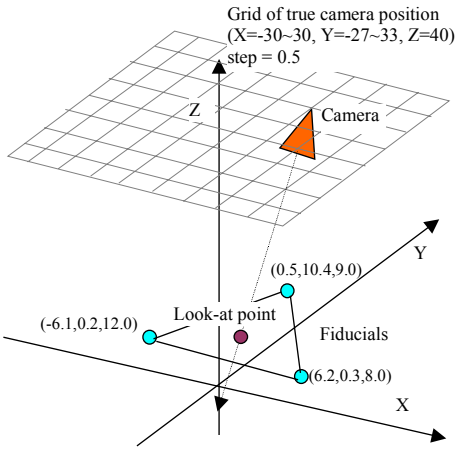


b. Errors in camera position

Fig.1 Propagated errors in dynamic tracking range extension experiment

A major source of errors in the above experiment is the pose calculation method. We used a popular 3-point analytical method [FISH81]. Fig. 2b plots the camera

position error produced by this method under simulated test conditions. We project calibrated points to the image plane and add Gaussian measurement noise (σ=0.5 pixels). The true camera position is placed at grid points on a plane with the look-at point maintained around the center of the triangle that is formed by the 3D point positions (Fig. 2a). The X and Y (horizontal plane) coordinates of the dots in Fig. 2b indicate the X and Y coordinates of the tested true camera positions. The vertical coordinate indicates the computed camera position error. The errors are well behaved (~0) in most cases. But this method has a known numerically unstable area (curved triangular hole in Fig. 2b) and also produces multiple solutions. (In our test, multiple solutions were ignored by selecting the closest solution to the true pose.) More accurate pose estimates are needed to reduce the error growth rate in extendible tracking.

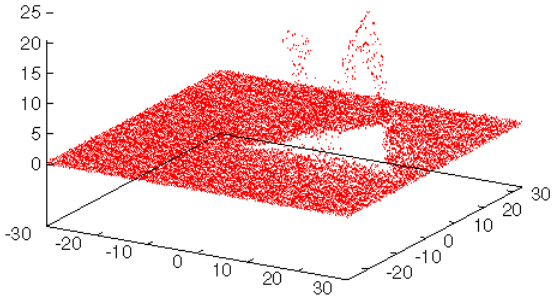


a - Experiment sequence

(1) true camera position is iterated on the grid points

(2) for a given true camera position:

Image coordinates of fiducials are computed, adding 0.5 pixel Gaussian noise, then based on the fiducial 3D positions and corresponding image coordinates, camera pose is computed.



b – Computed camera position error

Fig. 2 Accuracy test of 3-point pose method

Our criteria for pose calculation methods suited to dynamically extendible AR tracking are as follows:

- Real-time pose computation (<20 ms/estimate)
- Accurate solutions when given accurate data, yet robust solutions in the presence of measurement and calibration errors. The method should also facilitate outlier culling in the presence of gross errors (e.g., incorrectly identified features).
- Adaptive use of available information in a frame. When more information (features) are available in a frame, the method should use them to increase accuracy. When little information is visible, it should make the best estimate and reduce its confidence in the solution.

Methods that use all available information (N-point methods) are generally robust because errors and noise can be averaged out [DEME95]. In terms of the minimum number of features required for tracking, the lower, the better, but three or four visible features per frame are consistent with theoretical minimums.

2. Background

In this section, vision-based pose calculation methods are surveyed in light of our above criteria for application to extendible AR tracking.

Our test method is based on three calibrated point correspondences. Methods have been proposed to select the most likely solution among the multiple solutions [SHAR97]. We weighted several heuristics to rank the possible pose solutions [NEUM98], however, it is still unstable over a significant area and the heuristics fail in some situations (Fig. 2b). Fishler and Bolles suggested "Random Sample Consensus" as a method of smoothing data containing a significant percentage of gross errors [FISH81]. They applied this method to the 3-point pose method with success in removing the effects of gross errors. However, their method does not have a time limit (performing random trials), making it unsuitable for real time applications.

Horaud et al. developed a four point pose method using non-coplanar points. Geometric constraints are used to solve biquadratic polynomial equations with one unknown [HORA89]. They assert that their method is real-time, providing fewer solutions than 3-point-based methods, and is more stable (not dependent on the relative orientation of the image plane and scene plane). The problem of multiple solutions persists and near coplanar points and noise produce unstable results.

Ganapathy computes camera position and orientation using a non-iterative analytical method [GANA84]. His method also employs only 3 points (for external camera parameter estimation), and in general, there are multiple solutions. Although it can be extended to using n-points, it requires iterative optimization.

Uenohara et al. used a recursive method (Newton's method: multiple DSP chip implementation) for 6DOF

pose estimation and coplanar invariants for direct computation of 2D-image overlay [UENO96]. This is not suited for more general software-based 3D annotation.

There are many methods for recovering pose by iteratation (Newton-Raphson) [LOWE91] [YUAN89]. However, these require initial approximations and can be computationally expensive [DEME95]. Also the solutions can converge to local minima if the initial values are not close to the true solution [OLIE97]. Dementhon et al. designed an iterative algorithm that does not require initial estimates and performs in real-time [DEME95]. However, their method uses scaled orthographic projections and did not fully used the fact that rotation matrices are orthonormal.

Researchers also used curves or surfaces for pose estimation [FELD97][KRIE90], which are not relevant to our problem. Recently, researchers from LORIA combined 3D features and 2D correspondences to compute accurate camera pose changes [SIMO98]. Because the relationship between corresponding points is a function of camera motion, the camera pose accuracy can be improved using a cost function that counts the re-projection error and matching error. This work also uses a robust M-estimator for managing false matches. Although this method is robust and accurate, it is not real-time (about 25 seconds per frame) and it only stabilizes camera motion, not absolute pose.

In the field of Structure from Motion, researchers seek to recover camera motion and the model parameters simultaneously. For example, Azarbayejani et al. demonstrates a recursive algorithm to estimate object structure, camera motion, and camera focal length [AZAR95]. A minimum of 7 points is required, but in practice more points are needed for smoothing. The method is also sensitive to noise. Other methods that use the essential or fundamental matrix [HART95] rely on polynomial manipulations, which are also unstable [OLIE97].

Welch et al. designed a pose filter or estimator that accepts one point measurement (or constraint) at a time [WELC97]. The computational overhead of this method (called SCAAT) is small, facilitating real-time applications, and it exhibits robust behavior with noise. SCATT was developed for a high update rate (>1KHz) active beacon system that measured one point at a time. We also use the idea of processing one point at a time, however given that video images are snapshots containing multiple point measurements taken at the same time, but at a much lower rate (30Hz), we develop our filter specifically for video-based tracking.

Azuma et al. used gyros and accelerometers with active beacon tracking [AZUM94]. A Kalman filter predicts the user's head motion. Our second pose calculation method also uses a Kalman Filter to smooth its pose estimates.

3. New Methods

This section presents our two approaches to vision-based pose computation. One method is based on robust averages of 3-point solutions. The other is based on an iterative Extended Kalman Filter (iEKF), and SCAAT (Single Constraint At A Time) filter [WELC97]. Our methods are designed specifically for video frame rates and over-constrained measurements per frame that are typical of passive vision systems.

3.1. Robust Averages of 3-Point Solutions (RA3)

To address the multiple solution problems and instabilities of the simple 3-point pose method (shown in Fig. 2), we take robust averages of 3-point solutions. First, the features whose 3D position uncertainties are below a threshold are selected. (Note that these features may have been calibrated off-line or dynamically on-line.) Second, the feature positions in the image are analyzed to select a set of evenly distributed features. Four features closest to the corners of the image, and two features closest to the center of the left and right halves of the image, are selected. A maximum of twenty triples from these six points is robust-averaged with outlier culling.

As can be seen from Fig. 2, there are pose outliers due to numerical instability. For robust least-square solutions, the M-estimator has been suggested [HUBE81] [SIMO98]. We used a real-time approximation of Huber's M-estimator. Details are found in Appendix A.

After the robust M-estimator is computed, a linear Kalman filter applies temporal smoothing. In our case, the measurement and the Kalman filter state have the same dimension, and the measurement equation and process equation are linear (Fig. 3). Currently, a simple dynamic equation with 0 acceleration is to effect smoothness.

This pose solution makes use of both spatial (by feature distribution) and temporal (by smoothing using a linear Kalman filter) information.

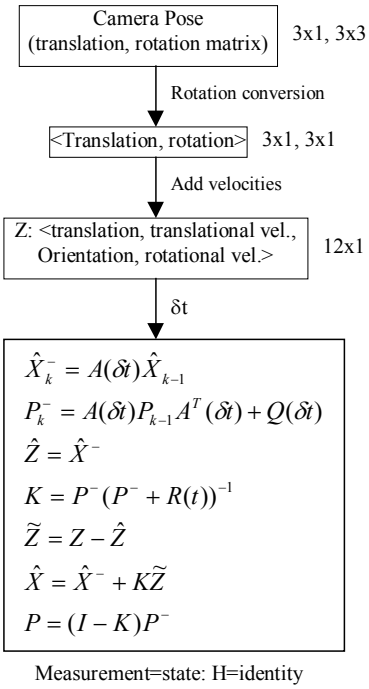


Fig. 3 Linear Kalman Filter for temporal smoothing

Fig. 4 Shows how the pose solutions is improved by averaging, robust M-estimation, and Kalman filtering. Gaussian noise (σ = 0.5-pixel) was added to the measurement of simulation data. The 3-point method is greatly enhanced by averaging (Fig. 4a). The M-estimator removes the effects of incorrect correspondences and performs outlier (e.g., gross error) culling. Even with correct data and no outliers, the result was improved in many frames (Fig. 4b) showing reduced sensitivity to noise. Lastly, the linear Kalman filtering smoothed the camera pose enhancing the camera position accuracy (Fig. 4c). In this and many other pose calculation methods, camera orientation is calculated based on camera position and feature correspondences, so the orientation accuracy depends on the position accuracy. Thus, we did not present the charts for orientations. The improvements are summarized in Table 1. with averages and standard deviations of errors. These error statistics clearly show the benefits of averaging, applying robust M-estimator, and Kalman filtering.

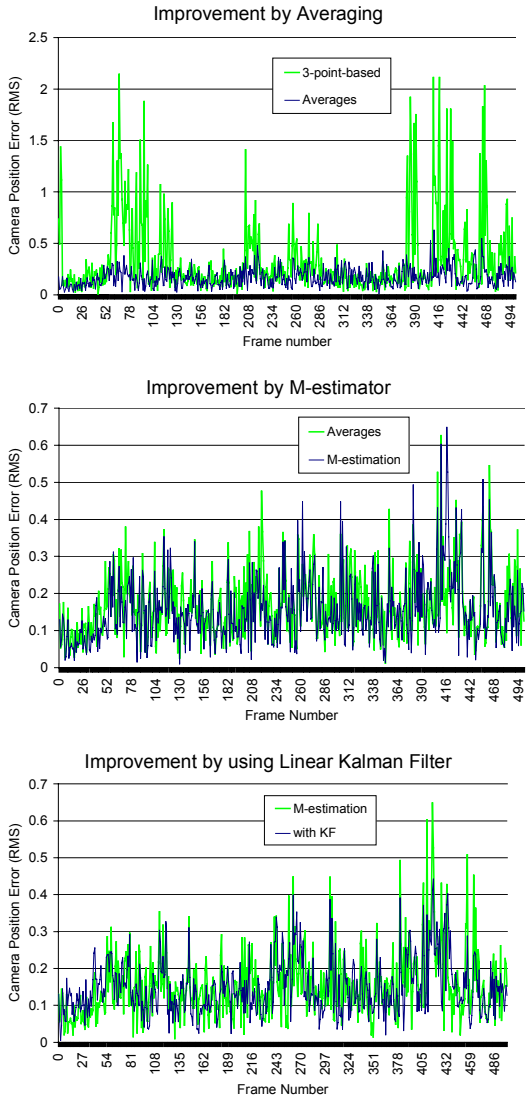


Fig. 4 Improvement by averaging (a - top), robust M-estimation (b - middle), and temporal smoothing with Kalman filtering (c - bottom).

	3-point	Avg.	M-est.	w/ KF
Mean	0.350313	0.169502	0.15565	0.150282
σ	0.379892	0.09245	0.091413	0.0753

Table 1. Improvement in averages and standard deviation in errors

3.2. Iterative Extended Kalman filter (iEKF)

This method iterates an Extended Kalman Filter (EKF) using all the usable point measurements available in a frame. We call this method iterative EKF (iEKF) to differentiate from Iterated Extended Kalman Filter (IEKF) which iterates in Taylor series expansion. As mentioned previously, this method is a variation of SCAAT, which was designed for sensory devices that read incomplete information sequentially and

measurement systems that fuse the incomplete information to update the state. A SCAAT filter uses the incomplete information in order to partially update the state in under-constrained cases without incorrect simultaneity assumption. However, measurements in passive vision systems are often over-determined providing enough information simultaneously to completely update the state. SCAAT was successful for tracking systems of high frequencies (e.g., UNC's Hiball tracker: >1KHz), and its performance on tracking systems of low frequencies (e.g., passive vision tracker: 7-30Hz) is not well known. The high frequency updates of the SCAAT filter may be compensated for by over-determination of the vision-based tracker.

Because of the low measurement rate of video tracking, it may be impossible to use only one measurement at each time step (frame) and obtain reasonable estimates. We use many or all of the measurements available at each time step in a point by point iterative update of the state. For each new frame time, we predict the camera state based on the prior filter state and a model. Suppose there are *N* 2D feature measurements observable for a frame. We use these measurements to correct the prediction in an iterative way, using each measurement once. After using all the measurements for a frame, we zero the orientation elements of the state vector and update the external orientation in the form of quaternion.

We use six parameters (x, y, z, ϕ, θ, φ) to represent the state of camera. These six elements have their own motion curves we approximate as quadric curves. Let x_i represent a curve, then

$$x_i(t+\delta t) = x_i(t) + \dot{x}_i(t)\delta t + \ddot{x}_i(t)(\delta t)^2 / 2$$
$$\dot{x}_i(t+\delta t) = \dot{x}_i(t) + \ddot{x}_i(t)\delta t$$
$$i \in \{x, y, z, \phi, \theta, \varphi\}$$

We use terms $\ddot{x}_i(t)\frac{(\delta t)^2}{2}$ and $\ddot{x}_i(t)\delta t$ to model process noise. Because state transition matrix A implements the relationships

$$x_i(t+\delta t) = x_i(t) + \dot{x}_i(t)\delta t$$
$$\dot{x}_i(t+\delta t) = \dot{x}_i(t)$$

We can set the process noise matrix as follows

$$Q(\delta t)[i,i] = q_i(\delta t)^4 / 4$$
$$Q(\delta t)[i,j] = Q[j,i] = q_i(\delta t)^3 / 2$$
$$Q(\delta t)[j,j] = q_j(\delta t)^2$$
$$(i,i) \in \{(x,x),(y,y),(z,z),(\phi,\phi),(\theta,\theta),(\varphi,\varphi)\}$$
$$(i,j) \in \{(x,\dot{x}),(y,\dot{y}),(z,\dot{z}),(\phi,\dot{\phi}),(\theta,\dot{\theta}),(\varphi,\dot{\varphi})\}$$
$$(j,j) \in \{(\dot{x},\dot{x}),(\dot{y},\dot{y}),(\dot{z},\dot{z}),(\dot{\phi},\dot{\phi}),(\dot{\theta}.\dot{\theta}),(\dot{\varphi}.\dot{\varphi})\}$$
$$q_i \in \{q_x, q_y, q_z, q_\phi, q_\theta, q_\varphi\}$$

And q_i is a process noise constant that reflects the noise in motion. Large values for q_i mean more uncertainty of prediction and the filter will correct the estimate over a relatively large search space. On the other hand, small values for q_i mean less uncertainty of prediction and the filter will correct the estimate over a relatively small search space.

For example, if we set q_i = 1, it means to search camera position over a range of about 0.5mm and camera orientation in the range of about 0.03 degree. If we set q_i equal to 100, it means to search camera position in the range of about 5 mm and search camera orientation in the range of about 0.3 degree. Fig. 5 is the RMS error in camera position when q_i = 1 and 100 using 9 fiducials and 0.5 pixels measurement noise.

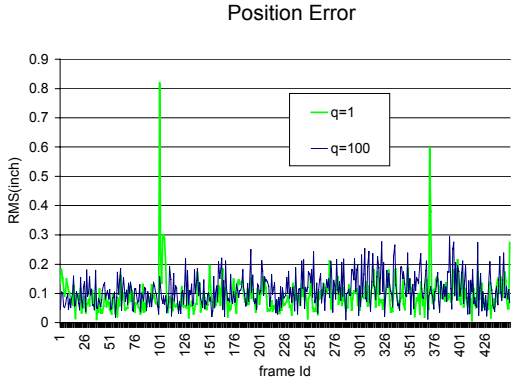


Fig. 5 RMS error of camera position with different static process noise

From Fig. 5, we see that when the process noise is large the peak error is small, but we have more jitters. When the process noise constant is small the peak error is large but with less jitters. We attempt to dynamically tune our process noise to reduce the error where possible. When there is a large change in camera state there is often large image feature motion in consecutive frames. If we measure large (above a threshold) feature motions between frames we use a large process noise. For small feature motions we use small process noise. Fig. 6 compares the results obtained by using the dynamic noise and small static noise. Fig. 7 compares the dynamic process noise and large static noise. We see that overall the error behavior is better for dynamic noise than for either of static cases.

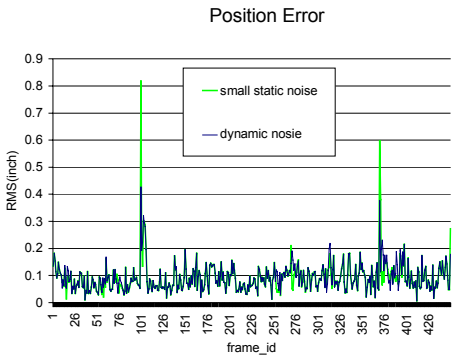


Fig. 6 RMS error for camera position with small static noise and dynamic noise

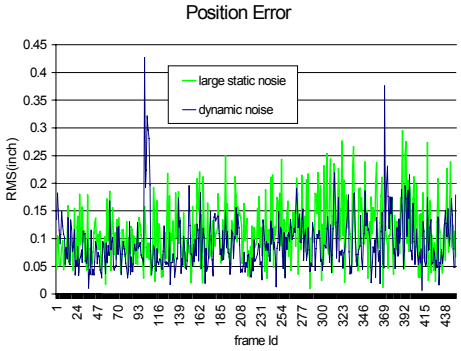


Fig. 7 RMS error for camera position with large static noise and dynamic noise

4. Experiments and Results

4.1. Synthetic Experiments

We performed synthetic data experiments to show that the proposed methods satisfy the pose computation method criteria of section 1.

For synthetic camera motion generation, a mechanical digitizer was used to generate 6DOF pose sequences or keyframe interpolations of viewpoints and look-at points were used. Gaussian noise of various standard deviations was added to the measurements.

The average computation times (with 6-14 points in view) for RA3 and iEKF were 3.6ms and 15 ms, respectively. Considering 30-70ms for image analysis and 25-40ms for virtual object rendering, computational overhead of both methods are small and the whole process runs at about 8-14Hz on a 450 MHz Pentium CPU.

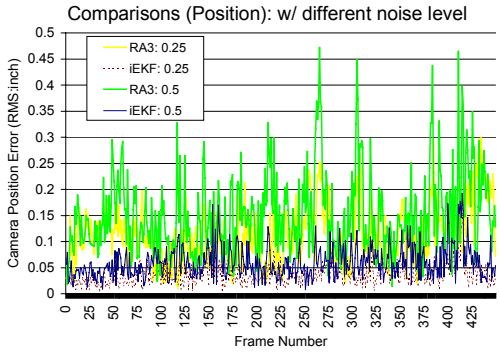
The accuracy was tested comparing the projections of 3D-points using true camera pose and estimated camera pose. Two 3D-points were projected in 500 test frames. Averages of projection errors are shown in Table 2 for two measurement noise levels.

Unit: pixel	RA3	iEKF
$\sigma = 0.25$	0.55	0.29
$\sigma = 0.5$	1.02	0.52

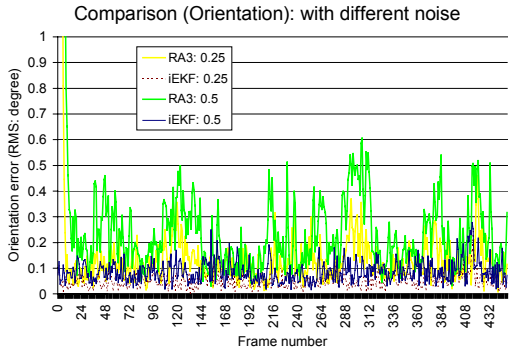
Table 2. Pose feature projection accuracy

4.1.1. Different Noise Levels

Computed camera pose solutions are compared with true values. Measurement noise levels were $\sigma = 0.25$ and $\sigma = 0.5$ pixel. The pose accuracy of iEKF was slightly better than RA3 as was its projected pixel error accuracy. Fig. 8 shows the camera position and orientation errors.



a – Camera position error for two noise levels



b – Camera orientation error for two noise levels

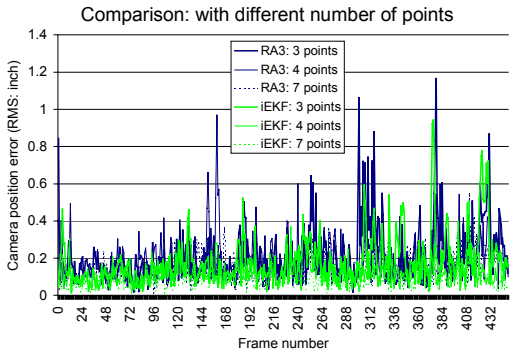
Fig. 8 Camera position and orientation errors for two measurement noise levels.

4.1.2. Processing Different Numbers of Points

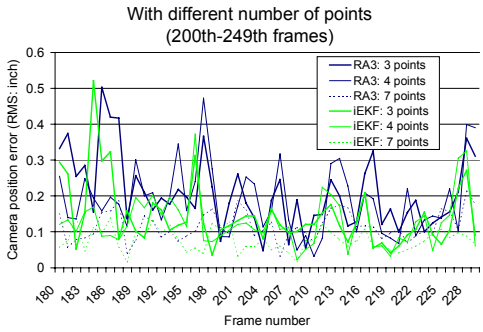
Processing different numbers of points affects the pose accuracy. Fig.9a shows pose solutions using 3, 4, and 7 points. Fig.9b is a zoomed segment of Fig. 9a for the segment between the 180th and 230th frame to better show how more points reduces pose errors.

An interesting point is that useful iEKF tracking is possible using only 2 points for a limited number of frames (Fig. 9c). This suggests that even if less than 3 points are available on average in a video sequence, iEKF may still track the camera state with reasonable

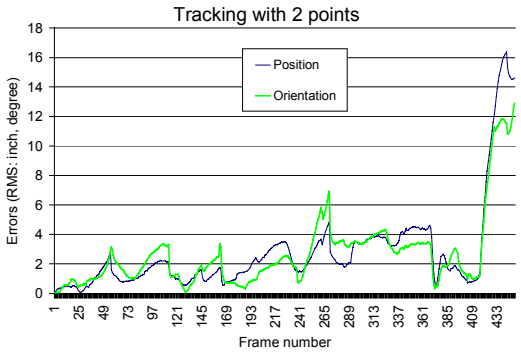
accuracy. (More analysis and testing needs to confirm this.)



a – Using 3, 4, and 7 points



b - Close view of (a) in 180th~230th frame



c – Tracking using 2 points: iEKF

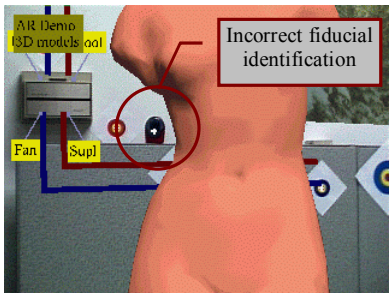
Fig. 9 Tracking with different number of points

4.1.3. Managing Outliers

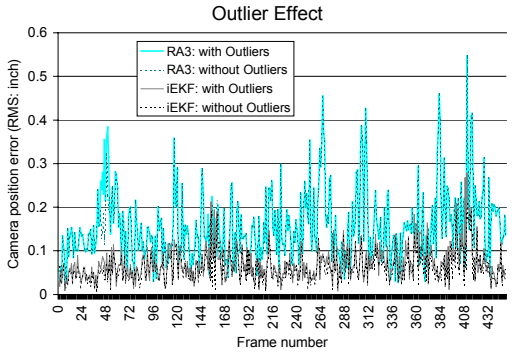
Vision-based methods are subject to have outlier problems. Fig. 10a is an example of a gross error resulted from incorrect fiducial identification (the cross mark on a pencil sharpener in the center of the image indicates that the pencil sharpener was detected as a fiducial).

Measurement outliers were added (in addition to σ=0.5 pixel Gaussian noise) to test the robustness of the methods (one feature outlier in 42nd~50th frame and 150th~162nd frame, two feature outliers in 343rd~344th frame; outlier displacements were 100~250 pixels).

RA3 method implements an approximated robust M-estimator (Appendix A.) and manages the outlier cases. Although it is difficult to implement robust statistical method directly to the iEKF method, the projection of the current feature can be compared with the measurement to reject outliers using a priori covariance [PRES93][BROI90]. As a result, both methods are robust in the presence of outliers (Fig. 10b).



a - An example of incorrect fiducial identification



b – Both methods are not affected by outliers

Fig. 10 With presence of outliers

4.1.4. Sudden Camera Motion

Sudden camera motion is generated to test the convergence stability of the methods. Both methods are stable providing pose solutions close to the true values. Fig. 11 shows the camera X coordinates. The pose solutions of the two methods are very close to the true pose even with sudden camera motion. Results of other coordinates are similar.

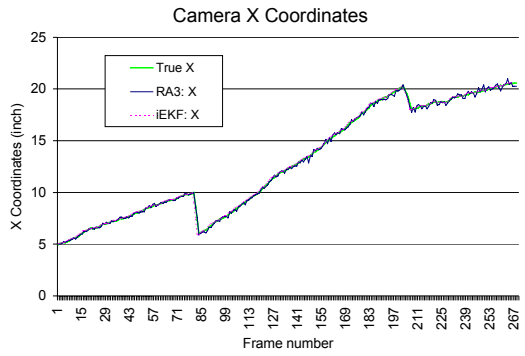


Fig. 11 Stability under sudden camera motion: camera X coordinates

4.1.5. Dynamic Calibration

The new methods were tested with the dynamic calibration test of Fig. 1. Tracking was started with 6 calibrated features, and dynamic calibration of 94 uncalibrated features was done in a 100"x30"x20" volume. The propagated errors were significantly reduced for both methods compared to the simple 3-point method. These results indicate that it may be feasible to use autocalibration over a long term and large area with modest error propagation. More tests and a real system are needed to verify or demonstrate the viability of extendible tracking for real applications.

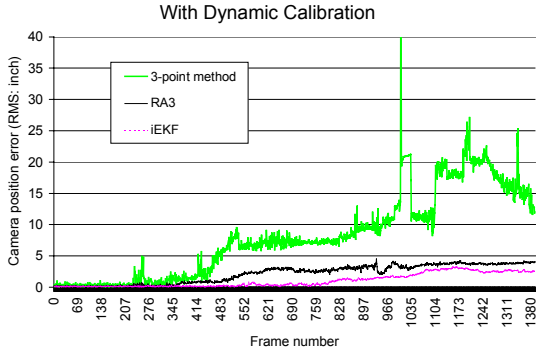
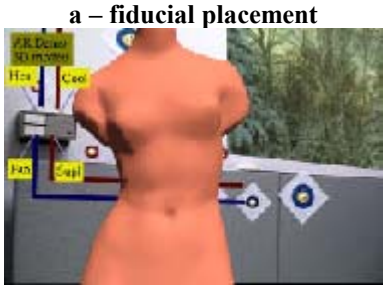


Fig. 12 Propagated camera position error with dynamic calibration

4.2. Real Data Experiment

A sequence of images was captured and digitized off-line to compare the two new methods. The image contains 15 multi-ring fiducials [CHO99] and the virtual objects include a torso of Venus, a virtual window, and annotations (Fig. 13). The re-projection errors between the measurement and projection of fiducials were computed (Fig. 14). The errors were predominantly under 1.0 pixel with both methods. Note that all the features are calibrated off line. No autocalibration is done in this test.

a – fiducial placement



b - virtual object overlay

Fig. 13 Real environment and virtual object overlay for experiment with real data

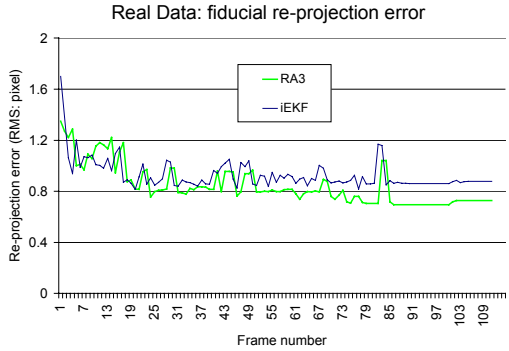


Fig. 14 Re-projection errors in real data experiment

5. Conclusion and Discussion

We described the desirable criteria for vision-based pose computation methods for supporting dynamic tracking extension. In summary these are real-time, robust, accurate, and n-point-based.

RA3 is fast (about 4 times faster than iEKF); robust under sudden camera motion and in the presence of outliers; accurate with about 1.0 pixel re-projection error (σ=0.5 pixel measurement noise); and capable of using a wide range of points (3-6 in our tests).

iEKF is also fast enough for real time applications, though it is slower than RA3; robust under sudden motion and in the presence of outliers; accurate within about 0.52 pixel of re-projection error (σ=0.5 pixel measurement noise); and fully n-point-based.

iEKF also has no minimum-points requirement for a given frame. It can estimate pose if only one point is

available in a frame (SCAAT case). As with SCAAT, only one feature is processed at a time, making the pose computation and uncertainty management relatively simple.

RA3 is fast and robust under noise measurements because of the robust M-estimator. Applying averaging has also advantage when the result (pose) is applied to a Kalman filter for temporal smoothing because the averaged result is Gaussian by central limit theorem. However, the accuracy is lower than that of iEKF.

The dynamic calibration experiment showed that both methods reduced the propagated error significantly, offering greater freedom of mobility and accuracy to the users of vision-based tracking systems.

Mpeg video files of the real data experiment are available for comparisons in our web page (http://star.usc.edu/~junp/iwar99.html).

6. References

[AZAR95] A. Azarbayejani and A. Pentland, "Recursive Estimation of Motion, Structure, and Focal Length", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 6, June 1995

[AZUM94] Ronald Azuma, Gary Bishop, "Improving Static and Dynamic Registration in an Optical See-through HMD", Proceedings of Siggraph94, Computer Graphics, pp.197-204

[BROI 90] T.J. Broida, S. Chandrashekhar, and R. Chellappa, "Recirsive 3-D Motion Estimation from a Monocular Image Sequence", IEEE Transactions on Aerospace and Electronic Systems Vol. 26, No. 4, July 1990

[CHO99] Youngkwan Cho, Scalable Fiducial-Tracking Augmented Reality, Ph.D. Dissertation, Computer Science Department, University of Southern California, January 1999

[DEME95] D. Dementhon and L. Davis, "Model Based Object Pose in 25 Lines of Code", International Journal of Computer Vision, 15:123-141, 1995

[FELD97] J. Feldmar, N. Ayache, and F. Betting, "3D-2D Projective Registration of Free-Form Curves and Surfaces", Computer Vision and Image Understanding, 65(3):403-424, 1997

[FISH81] Martin A. Fischler and Robert C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", Communications of the ACM, Vol.24, No.6, June 1981, pp.381-395

[GANA84] Sundaram Ganapathy, "Decomposition of Transformation Matrices for Robot Vision", Proceedings of Int. Conf. Ronotics and Automation, 1984, pp. 130-139

[HART95] R.L. Hartley, "A Linear Method for Reconstruction from Lines and Points", ICCV, pp.882-887, 1995

[HORA89] Radu Horaud, Bernard Conio, and Oliver Leboulleux, "An Analytic Solution for the Perspective 4-Point Problem", Computer Vision, Graphics, and Image Proceeding 47, 33-44 (1989)

[HUBE81] Huber, Peter J., "Robust Statistics", Wiley Series in Probability and Mathematical Statistics,1981

[KRIE90] D. Kriegman and J. Ponce, "On Recognizing and Positioning Curved 3D Objects from Image Contours", IEEE Transactions on PAMI, 12(12):1127-137, December 1990

[KUMA94] R. Kumar and A. Hanson, "Robust Methods for Estimating Pose and a Sensitivity Analysis", CVGIP:Image Understanding, 60(3):313-342, 1994

[LOWE91] Lowe, D.G., "Fitting Parameterized Three-Dimensional Models to Images", IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol. 13, pp.441-450, 1991

[MAYB79] P.S. Maybeck, *Stochastic Models, Estimation, and Control*, Volume 1, Acamedic press, Inc., 1979

[NEUM96] U. Neumann, Y. Cho, "A Self-Tracking Augmented Reality System," Proceedings of ACM Virtual Reality Software and Technology '96, pp. 109-115

[NEUM98] U. Neumann, J. Park, "Extendible Object-Centric Tracking for Augmented Reality", 1998 IEEE Virtual Reality Annual International Symposium, pp.148-155, 1998

[OLIE97] John Oliensis, "A Critique of Structure from Motion Algorithms", NECI Technical Report, April 1997, http://www.neci.nj.nec.com/homepages/oliensis/poleiccv.ps

[PARK98] J. Park, U. Neumann, "Natural Feature Tracking for Extendible Robust Augmented Realities", International Workshop on Augmented Reality (IWAR) '98.

[PRES 93] William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, *Numerical Recipes in C : The Art of Scientific Computing*, 2nd edition (January 1993), Cambridge University Press; ISBN: 0521431085, p.705

[SHAR97] R. Sharma, J. Molineros, "Computer Vision-Based Augmented Reality for Guiding Manual Assembly," Presence: Teleoperator and Virtual Environments, Vol. 6, No. 3, pp. 292-317, June 1997

[SIMO98] G. Simon, V. Lepetit, and M.-O. Berger, "Computer Vision Methods for Registration: Mixing 3D Knowledge and 2D Correspondences for Accurate Image Composition", International Workshop on Augmented Reality (IWAR) '98

[UENO96] M. Uenohara and T. Kanade, "Vision-Based Object Registration for Real-Time Image Overlay", Journal of Computers in Biology and Medicine, 1996

[WELC97] G. Welch, G. Bishop, "SCAAT: Incremental Tracking with Incomplete Information," Proceedings of Siggraph97, Computer Graphics, pp. 333-344

[YUAN89] Yuan, J.S.C. "A General Photogrammetric Methods for Determining Object Position and Orientation", IEEE Trans. On Robotics and Automation, Vol. 15, pp.129-142

Appendix A. Real-time approximation of Robust M-estimator

Generally, robust M-estimators require searches to find the M-estimations, which is not appropriate for real-time applications. However, an approximation can be obtained utilizing a priori averages with less computation.

If we apply the M-estimation technique using Huber's function $\rho(x)$, the minimization equation is as follows.

$$\min_p \sum_i \rho(r_i)$$
$$= \min_p \sum_i \rho(p - p_i)$$
$$= \min_p \left\{ \sum_{|p-p_i| \le c} \frac{(p-p_i)^2}{2} + \sum_{|p-p_i|>c} c \cdot (|p-p_i| - \frac{c}{2}) \right\}$$
$$= \min_p \left\{ \sum_{|p-p_i| \le c} \frac{(p-p_i)^2}{2} + \sum_{p-p_i>c} c \cdot (p-p_i - \frac{c}{2}) + \sum_{p-p_i<-c} c \cdot (-p+p_i - \frac{c}{2}) \right\}$$

This equation is difficult to evaluate analytically requiring search because of the conditions p-p_i>c etc.: Conditions depend on the unknown value of p. However, because $p \approx \overline{p}$, we can approximate the condition using the average.

$$\min_p \left\{ \sum_{|p-p_i| \le c} \frac{(p-p_i)^2}{2} + \sum_{p-p_i>c} c \cdot (p-p_i - \frac{c}{2}) + \sum_{p-p_i<-c} c \cdot (-p+p_i - \frac{c}{2}) \right\}$$
$$\approx \min_p \left\{ \sum_{|\overline{p}-p_i| \le c} \frac{(p-p_i)^2}{2} + \sum_{\overline{p}-p_i>c} c \cdot (p-p_i - \frac{c}{2}) + \sum_{\overline{p}-p_i<-c} c \cdot (-p+p_i - \frac{c}{2}) \right\}$$
$$\Rightarrow \frac{\partial}{\partial p} \left\{ \sum_{|\overline{p}-p_i| \le c} \frac{(p-p_i)^2}{2} + \sum_{\overline{p}-p_i>c} c \cdot (p-p_i - \frac{c}{2}) + \sum_{\overline{p}-p_i<-c} c \cdot (-p+p_i - \frac{c}{2}) \right\} = 0$$
$$\left\{ \sum_{|\overline{p}-p_i| \le c} \frac{\partial}{\partial p} \frac{(p-p_i)^2}{2} + \sum_{\overline{p}-p_i>c} \frac{\partial}{\partial p} c \cdot (p-p_i - \frac{c}{2}) + \sum_{\overline{p}-p_i<-c} \frac{\partial}{\partial p} c \cdot (-p+p_i - \frac{c}{2}) \right\} = 0$$
$$\left\{ \sum_{|\overline{p}-p_i| \le c} (p-p_i) + \sum_{\overline{p}-p_i>c} c + \sum_{\overline{p}-p_i<-c} (-c) \right\} = 0$$
$$p \approx \frac{1}{N} \left\{ \sum_{|\overline{p}-p_i|} p_i + \sum_{\overline{p}-p_i>c} (\overline{p} - c) + \sum_{\overline{p}-p_i<-c} (\overline{p} + c) \right\}$$

where $c = k \cdot \sigma$.

Consequently, we can use pseudo measurement suggested by Huber. To devise robust algorithm that can be easily patched into existing programs, Huber suggest pseudo-observation, for example, in least square fitting:

Let $\hat{y}_i$ fitted value of y_i's

$r_i = y_i - \hat{y}_i$

s_i Standard error of y_i or r_I

Pseudo-observations y^*_i is defined as

$$y_i^* = \begin{cases} y_i & if \quad |\overline{y}_i - \hat{y}_i| \le c \cdot s_i \\ \hat{y}_i - c \cdot s_i & if \quad \overline{y}_i - \hat{y}_i < -c \cdot s_i \\ \hat{y}_i + c \cdot s_i & if \quad \overline{y}_i - \hat{y}_i > c \cdot s_i \end{cases}$$

Constant c regulates the amount of robustness: good choose is btw 1 and 2, e.g., c=1.5

However, there needs a method to differentiate between gross errors (e.g., from multiple solutions) and contaminated outliers (e.g., from noise measurement). This can be achieved by throwing away measurements with $|r| > c_o \cdot s_i$ where e.g., $c_o = 4$.