Extending Augmented Reality with Natural Feature Tracking

Jun Park[J.P.], Suya You[S.Y.], and Ulrich Neumann[U.N.]

Computer Science Department
University of Southern California
Los Angeles, California 90089-0781

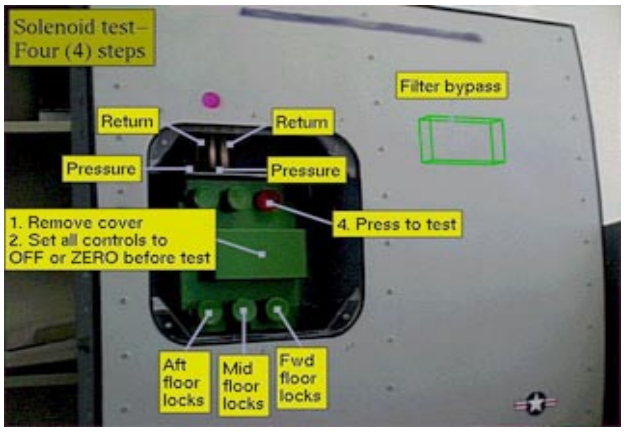
ABSTRACT

In traditional vision-based Augmented Reality tracking systems, artificially-designed fiducials (or landmarks) have been used as camera tracking primitives. The 3D positions of these fiducials should be pre-calibrated, which imposes limitations in ranges of tracking view. Fortunately, the advance of computer vision technologies combined with new point position estimation technology enable natural features to be detected, tracked, and calibrated to be used as camera tracking primitives. This paper describes how these technologies are used to track in an unprepared environment for Augmented Reality.

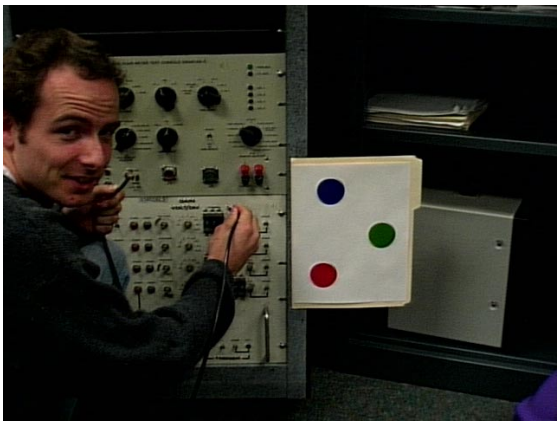
Keywords: Augmented Reality, Natural Feature tracking, Fiducial, Kalman Filter, Position Estimation, Tele-manipulation, Tele-training

1. INTRODUCTION

Augmented Reality (AR) is an interface technology in which a user's perception of the real world is enhanced or augmented by the addition of computer-generated objects or information (figure 1-a). The enhancement could take the form of label annotations, virtual object overlays, or shading modifications. AR promises to be a powerful technology in helping people and computers to cooperatively work in the real world in a wide range of applications including manufacturing, medicine, entertainment, and education. For example, in tele-manipulation or tele-training systems, an on-site worker/trainee may have questions answered by a remote expert through text, images, and dynamic manipulations of 3D virtual objects superimposed on shared images of the site environment. Figure 1-b is an example of a shared real environment where an on-site worker imposes a question of where to connect the cable.



a. Example of AR annotation supporting a maintenance task



b. Example of Tele-manipulation: On-site personnel asks where to connect the cable.

Fig. 1 Examples of AR annotation and Tele-manipulation

[J.P.] Email: junpark@cs.usc.edu, http://star.usc.edu/~junp, Telephone: (213) 740-4238, Fax: (213) 740-7285

[S.Y.] Email: suyay@graphics.usc.edu,

[U.N.] Email: uneumann@cs.usc.edu, Telephone (213) 740-4489, Fax: (213) 740-7285

Maintaining accurate spatial relationships between real and virtual objects requires tracking, and many systems have been developed to track the six-degree of freedom (6DOF) pose of an object (or a person) relative to a fixed coordinate-frame in the environment [FOXL96, GHAZ95, KIM97, MEYE92, STAT96, WARD92, WELC97, FOXL98]. These tracking systems involve a variety of sensing technologies, each with unique strengths and weaknesses, to determine a world coordinate pose, as required for virtual and augmented reality applications. However, a large class of AR applications require annotation on objects whose positions in a room or the world may vary freely without impact on the AR media linked to them. For example, AR applications in manufacturing, maintenance, and training [KLIN97, CAUD92, FEIN93] require virtual annotations that provide task guidance and specific component indications on subassemblies or portions of structure that may move within a room, building, or complex. A more appropriate tracking approach for these mobile applications is one that is based on viewing the object itself [NEUM96, UENO95, MELL95].

Many of these vision-based AR systems depend on artificial fiducials (also called landmarks) or a prior-known model data to perform the dynamic alignment between a real and a virtual camera. These approaches are appropriate in environments where known a priori features are continually in view. However, it becomes difficult to satisfy these constraints in cases of large-scale working areas, partial occlusion, or scenes where little or no data is available. In such cases, tracking depends on the utilization of naturally occurring scene features. This paper broaches the issues of using natural feature tracking and 3D pose estimation for extended-area AR applications. Our goal is to enable AR tracking for unprepared large-area working environments.

Our approach is to start with a set of temporarily placed fiducials, whose 3D positions have been calibrated off-line, as an initial calibration reference (figure 1). The reference is used to compute initial camera poses. As the camera moves, the system continuously tracks the fiducials as well as any natural features detected in the images. The camera pose, computed from the reference fiducials, is used to calibrate the natural scene features as they are tracked over an extended set of frames. Once calibrated, the natural scene features are used to compute pose in the absence of reference fiducials. The novel aspects of our techniques are, first we use a lazy-evaluation of structure from motion to calibrate tracked natural features, and secondly we use the calibrated natural features as tracking primitives to estimate camera pose. Consequently, our system tracks more robustly in large spaces and under conditions where some or all reference fiducials become occluded.

The detection and tracking of natural features is performed by a novel adaptive motion estimation approach [NEUM98a]. We select the most reliable features based on a tracking confidence metric, and track features in a multi-stage process that includes feedback of feature tracking confidence. This approach produces reliable image motion estimation for a modest computation. The 3D feature positions are estimated with Kalman filters [NEUM98c].

In combination, these techniques produce a robust AR tracking system that is capable of extending its operating area into unknown and unprepared scene regions. The natural feature tracking is still an order of magnitude too slow (~1Hz) for real-time applications. For this paper, real-time video sequences are recorded and processed automatically (no user intervention) and then reassembled into a video sequence. We are currently investigating dedicated hardware and DSP processor implementations of the algorithms.

2. SYSTEM OVERVIEW

Figure 2 depicts our system architecture. Let f be an image frame, N_{fd} be a set of image frames where 3 or more fiducials are detected, and N_{nd} be the other set of image frames where fewer than 3 fiducials are detected. For the image frames where three or more pre-calibrated fiducials are detected ($f \in N_{fd}$), the camera pose is calculated based on these fiducials and the 3D positions of detected natural features are estimated using recursive filters. For the image frames where fewer than three or none of the fiducials are detected ($f \in N_{nd}$), our system utilizes the estimated 3D positions of the natural features for camera pose calculation. Sections 3 and 4 describe fiducial-based tracking and natural feature-based tracking in more detail.

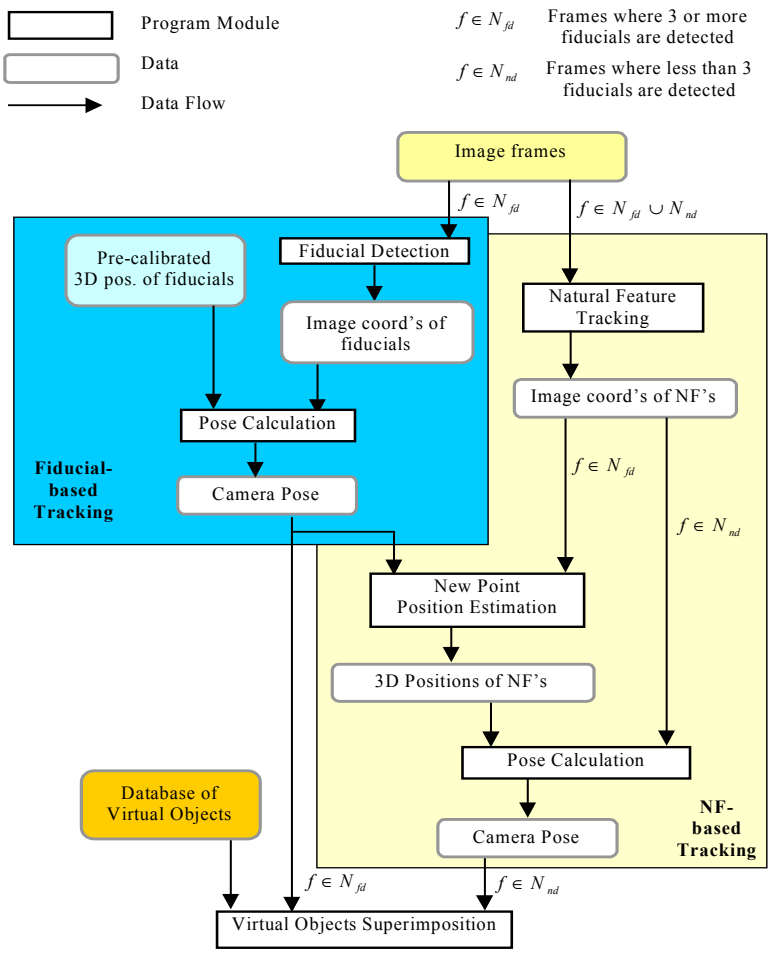


Fig. 2 System Components for AR system

3. FIDUCIAL-BASED TRACKING

Fiducial-based tracking (the part of the system on the dark rectangle in figure 2) is not a new concept. Given that it is required as an initial step for natural feature-based tracking, we provide a summary and discuss the issues regarding fiducial detection and pose calculation.

Fiducials have the advantage that they can be designed to maximize the ability of an AR system to detect and distinguish between them, they can be inexpensive, and they can be placed arbitrarily on objects. Our fiducial design is a colored circle or triangle [NEUM96][CHO97], but other designs such as concentric circles or coded squares are equally valid [STAT96, KLIN97, MELL95]. We use the three primary and three secondary colors along with the triangle and circle shapes to provide twelve unique fiducial types. Fiducials are detected by segmenting the image into regions of similar intensity and color, and by testing the regions for the color and geometric properties of fiducials. Detection strategies are often dependent on the characteristics of the fiducials [REKI97, TREM97, UENO95]. Currently we are working on more robust and intensity-tolerant fiducial detection system using fuzzy and rule-based algorithm.

To compute the camera pose, the 2D fiducials must correspond to elements in a database of known 3D fiducial positions and types. Computing correspondences is hard in the general case [MELL95, UENO95], and trivial if the fiducial types are unique for each element in the database. For this work of fiducial-based tracking, we consider only unique feature types that are trivial to correspond since our focus is on the extendible tracking portion of the architecture.

In computing camera pose, three or more corresponded fiducials are required [FISC81]. The approach we use has known instabilities in certain poses, and in general provides multiple solutions (two or four) from a fourth-degree polynomial. Methods have been proposed to select the most likely solution [SHAR97]. We weight several tests to rank the possible pose solutions in terms of their apparent correctness [NEUM98b].

To reject still spurious solutions, the system involves the use of the assumptions of smooth camera transitions. Let P_i be the most recent pose that was assumed (or proven) to be correct and P_j be a new pose, $i < j < i + n$ where n is a number to restrict j to a vicinity of i.

$$\textit{if (distance}\,(P_i, P_j) < \textit{threshold)}$$

P_j is accepted

If there is no P_j satisfying the condition with $i < j < i + n$, the system resets P_i to P_j, rejecting the assumption that P_j was a correct pose.

4. NATURAL FEATURE BASED TRACKING

Our novel approach of natural feature-based tracking (the part of the system on the bright rectangle in figure 2) is to combine natural feature tracking and new point position estimation. Calibrated natural features become tracking primitives for camera pose calculation.

Natural features are first detected from the initial image and ranked by a confidence metric. Then a user-specified number of features are selected and tracked in the following images, providing the image coordinates of the natural features for each image frame. These image coordinates of natural features, combined with camera pose calculated by fiducial-based tracking, are used to calibrate the natural features by new point position estimation process. The correspondence of natural features between images is solved by feature tracking algorithm as explained in section 4.1. The estimated 3D positions and their corresponding image coordinates of the natural features can be used to calculate camera pose in case part or all of the fiducials are occluded or undetectable.

Our contribution is the unique combination of components and the architecture of our system. Natural feature tracking process combined with new point position estimation process enables us to track the camera pose in an uncalibrated environment. Starting from a small set of calibrated fiducials, the camera tracking range can be extended dynamically and automatically. The user can also interact with the new environment, adding or modifying the virtual objects and real objects, since our tracking system is dynamically extendible to regions without calibrated features.

One of the issues related with natural feature-based tracking is the strategy of selecting natural features to be used for computing camera pose. Currently, we have the strategy of dynamically choosing four features closest to the each corner of the image. This results in evenly distributed feature sets. We also may consider the uncertainties of 3D position estimation and 2D variances of the features. Lastly, the information on the normal of the plane where 3 features lie may help avoiding numerical instability that occurs in certain poses.

4.1 NATURAL FEATURE TRACKING

To perform detection and tracking of naturally occurring features, we developed a novel motion tracking approach [NEUM98a]. The system integrates three main motion analysis functions, e.g., feature selection, motion tracking, and estimate verification, in a closed-loop cooperative manner to deal with the complex natural imaging conditions. First, in the feature selection module, two kinds of tracking features (points and regions) are selected and evaluated for their suitability for reliable tracking and motion estimation. The selection and evaluation processes are based on a tracking evaluation function that measures the suitability of features (λ) and the confidence of tracking (δ) fed back from tracking module

$$C = k_1\lambda + k_2\delta \tag{1}$$

where k_i is the weighted coefficient for each component. Once selected, the features are ranked according to their evaluation values and then fed into the tracking module.

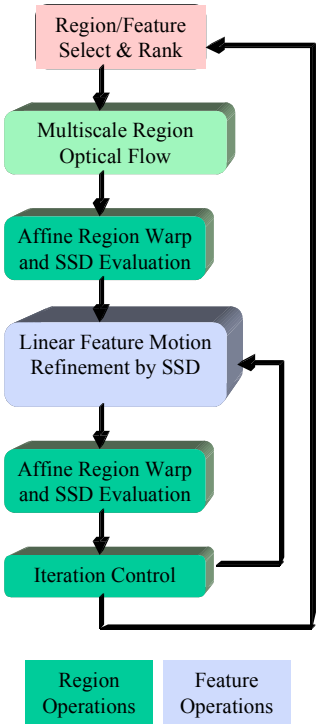


Fig. 3 – Feedback from tracking evaluation controls the interative refinement of motion estimates within each region.

The tracking method we employed is a differential based local optical flow estimate technique which utilizes normal motion information in local neighborhood to perform a least squares minimization to find the best fit to motion vectors. Unlike traditional implementations, we developed a multi-stage robust estimation strategy (more fully described in [NEUM98a]). The basic components (Fig. 3) of the multi-stage strategy are three steps: image warping, motion residual estimation, and motion refinement. For every region, an estimated motion is computed and fit to a model. A verification and evaluation are imposed to measure the confidence of the estimation and the model fit. If the estimation error is large, the motion estimate will be refined until the estimation error converges or the region is discarded as unreliable for tracking. In order to handle local geometric distortions due to large view variations and long sequence tracking, two motion models, a translation model and an affine model, are used for tracking point and region features, respectively. These models are the basis of the motion verification and evaluation processing. In regions, for example the optical-flow motion estimate is computed and fit to an affine model that is used to warp the region into an confidence evaluation frame $(x_c, y_c)^T$.

$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} v_2 x_{t_0} + v_3 y_{t_0} + v_1 \\ v_4 x_{t_0} + v_5 y_{t_0} + v_6 \end{bmatrix} \tag{2}$$

The confidence evaluation frame is compared to the true target image to obtain a measure of tracking error

$$\varepsilon = \frac{\left\| R_t(\mathbf{x},t) - R_c(\mathbf{x},t) \right\|^2}{\max\{\left\| R_t(\mathbf{x},t) \right\|^2, \left\| R_c(\mathbf{x},t) \right\|^2\}} \tag{3}$$

where $R_t(\mathbf{x},t)$ and $R_c(\mathbf{x},t)$ are the true target frame and reconstructed confidence evaluation frame, respectively. Motion error feedback is an essential component of an architecture for robust tracking. The error information is fed back to the tracking module for motion correction and to the feature detection module for continuous feature re-evaluation. This re-evaluation of features keeps the tracking system working in an "optimum" state by selecting and maintaining the most reliable features. Although two different verification strategies are used for the two types of tracked features (points and regions) and their motion models (translation and affine), they both generate an evaluation frame that measures the estimation residual. The feedback tracking confidence is defined as

$$\delta = \frac{1}{1+\varepsilon} \tag{4}$$

The closed-loop stabilization of the tracking system is inspired by the use of feedback for correcting errors in a non-linear control system. The process acts as "selection-hypothesis-verification-correction" strategy that make it possible to discriminate between good and poor estimation features, and maximizes the quality of the final motion estimation.

4.2 NEW POINT POSITION ESTIMATION

This is a summary of the explanation given in [NEUM98b, NEUM98c]. Two different recursive filters have been designed and tested to estimate 3D positions of new points based on the camera pose and measurements of image coordinates of features [NEUM98b].

EKF (Extended Kalman Filter) has the 3D position of a new point as its state. Since there is no dynamics involved with the state and the dimension of the state is three, the computation is quite low compared with other AR system modules such as fiducial detection and natural feature tracking.

The EKF process is composed of two groups of equations: *predictor* (time update) and *corrector* (measurement update). The *predictor* updates the previous ($(k-1)^{th}$) state and its uncertainty to the predicted values at the current (k^{th}) time step. Since the 3D fiducial position does not change with time, the predicted position at the current time step is the same as the position of the previous time step.

$$\begin{aligned} \hat{x}_k^- &= \hat{x}_{k-1} \\ P_k^- &= P_{k-1} + Q \\ \hat{z}_k &= \vec{h}(\hat{x}_k^-, c_k, p_c) \end{aligned} \tag{5}$$

Corrector equations correct the predicted state value $\hat{x}_k^-$ based on the residual of actual measurement z_k and measurement estimate $\hat{z}_k$. The Jacobian matrix linearizes the non-linear measurement function.

$$\begin{aligned} K_k &= P_k^- H_k^T (H_k P_k^- H_k^T + R)^{-1} \\ \tilde{z}_k &= z_k - \hat{z}_k \\ \hat{x}_k &= \hat{x}_k^- + K_k \cdot \tilde{z}_k \\ P_k &= (I - K_k H_k) P_k^- \end{aligned} \tag{6}$$

More details about EKF can be found in references such as [MAYB79, MEND95].
RAC (Recursive Average of Covariances) filter was designed to work in the 3D space with the lines as measurements. RAC filter has been refined recently to be more robust, accurate, and fast-converging.
The intersection[1] of two lines connecting the camera positions and the feature locations in the image create the initial estimate of the 3D position of the fiducial. The intersection threshold is a design parameter and depends on the fiducial design or the natural scene and objects. These initial estimates often have quite good accuracy especially when the angles between the two lines are big enough (e.g. > 5 degree).

Both filters appear stable in practice. The EKF is known to have good characteristics under certain conditions [BROI90], however the RAC gives comparable results, and it is simpler, operating completely in 3D-world space with 3D lines as measurements. The RAC approach eliminates the linearization processes required in the EKF with Jacobian matrices. For results of synthetic data and real data experiments, refer to [NEUM98b][NEUM98c].

5. INTERACTIVE VIRTUAL OBJECT MANIPULATION

Virtual objects or annotations often greatly enhance communication efficiency between on-site and off-site people sharing images of the site environment. Thus, the capability to create virtual objects/annotations interactively and dynamically is crucial for efficient communication. Users can specify points to be annotated by simply clicking a mouse on the corresponding screen positions with inputs of annotation information. Where no knowledge about the environment is available, the 3D positions of the user-specified points on screen need to be estimated on-line and in real-time.

There are two cases in estimating the 3D positions of the points. In the first case, there happens to be a natural feature that has been tracked or is being tracked at the user-specified point. In this case, the 3D position of the natural feature, which is already available, is the 3D position of the point. The 3D-2D correspondence between the 3D position of the natural feature and the 2D screen coordinate of the user-specified point can be solved by a vicinity test. Let $\boldsymbol{X}_f$ be the 3D position estimate of a natural feature, $\boldsymbol{Z}_f$ be the projection of $\boldsymbol{X}_f$ given the current camera pose. Let $\boldsymbol{Z}_u$ be the user specified point in the image.

if (distance($\boldsymbol{Z}_f$, $\boldsymbol{Z}_u$) < threshold)

$\boldsymbol{X}_f$ and $\boldsymbol{Z}_u$ correspond

where ***threshold*** is calculated from the projected uncertainty of the 3D position estimate and pre-defined pixel unit threshold (e.g., threshold for errors in mouse clicking).

In the other case where there is no natural feature corresponding to the user-specified point, a user needs to specify the point in more than one frame. A user can inform the point id to the system when he/she specifies a point to solve the 2D-2D

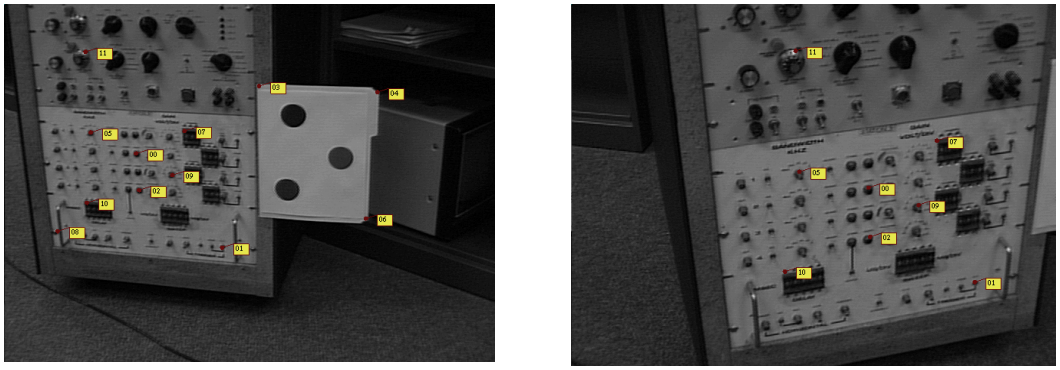
[1] Since two lines in 3D space may not actually intersect, the point midway between the points of closest approach is used as the intersection.

correspondence problem. The system can perform a vicinity test between the lines connecting the camera positions and the specified points, and may reject if the distance is larger than a threshold. The 3D position estimate is the intersection[1] of the two lines and can be refined using a recursive filter. The threshold value depends on the pre-defined pixel-unit threshold and the depth of the camera. For example, with 10 pixel-unit threshold and 50 inch of depth, and given that pixel-unit focal length is 1138, $\textbf{\textit{threshold}} = \frac{10 * 50}{138} \approx 0.44$ in inch scale.

6. RESULTS AND DISCUSSION

The goal of our experiment was to show how the system utilizes a temporarily placed fiducial set as initial calibration reference in order to, for example, put manipulation instructions to an on-site worker (figure 1) from off-site. As the camera moved, the system continuously tracked the fiducials as well as natural features detected in the images. The camera pose, computed from the reference fiducials, was used to calibrate the natural scene features as they were tracked. Calibrated natural features were used to compute camera pose in the absence of fiducials. As the camera pose was computed either by fiducials or by natural features, the system overlaid virtual objects/annotations placed by an off-site expert to answer the question. The image stream generated by a camera was directly digitized using a video editing system. The sampling rate was 15Hz resulting in about 250 frames from a 17 second video sequence.

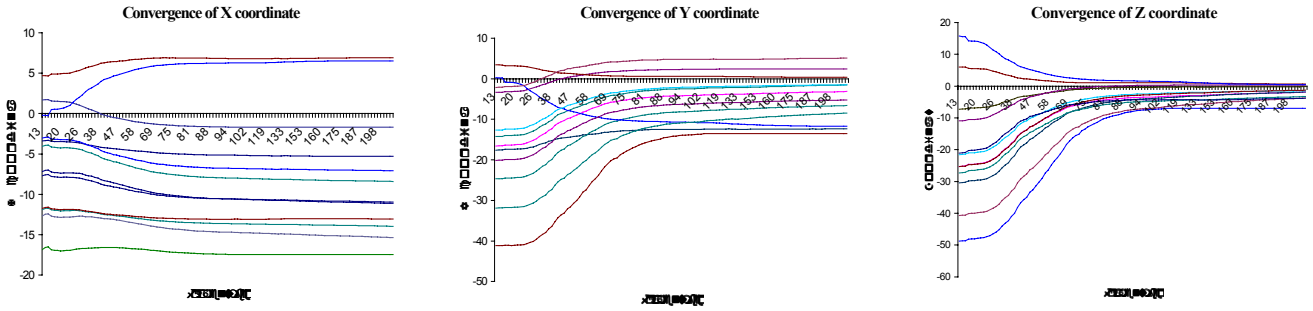
Figure 4 shows the result of natural feature tracking. Twenty features were detected in the first frame, twelve out of which were selected for tracking, rejecting others for being too close to fiducials or the already selected features. Even in 250th frame, all the features were accurately tracked except for those out of screen.



Feature Detection : 4th frame Feature Tracking : 250th frame

Fig. 4 Results of Feature Detection and Tracking

Figure 5 shows the result of new point position estimation. Each chart indicates the convergence of X, Y, and Z coordinates of 3D positions of the natural features. They converge fast (at about 90th frame, i.e., in 6 seconds) and are stable after the convergence. It is noticeable that the initial estimates of Z coordinate is less accurate than X and Y coordinates, which is intuitive.



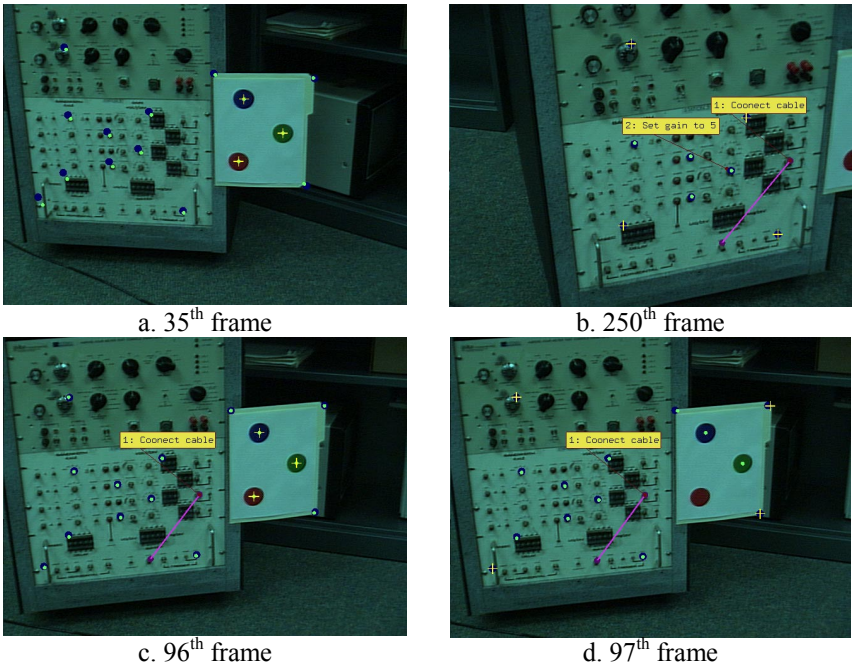
X coordinates of natural features Y coordinates of natural features Z coordinates of natural features

Fig. 5 Convergence of 3D positions of Natural Features

Figure 6 shows the result of camera tracking and virtual object/annotation overlay. The bigger dark circles indicate the projections of the calibrated (in case of fiducials) or estimated (in case of natural features) 3D positions. The smaller bright circles indicate the measurements resulted from natural feature tracking. The bright crosses indicate fiducials or natural features that were used for tracking. In case, fewer than three fiducials are detected, 4 features (either tracked natural features or detected fiducials) close to each corner of the image were selected to compute camera pose.

Figure 6-a shows the early stage of tracking and natural feature 3D position estimation. There are noticeable difference (about 5-7 pixels) between the projection of the estimated 3D position and screen coordinates resulted from natural feature tracking. As the frame number increases, the estimates of 3D positions and the screen coordinates resulted from natural feature tracking become closer (about 1-2 pixels) as can be seen in figure 6-b.

Figures 6-c and 6-d show how the tracking primitives for camera pose calculation were switched from fiducials to natural features. There are also frames where the mixture of fiducials and natural features were used for pose calculation.



a. 35th frame b. 250th frame

c. 96th frame d. 97th frame

Fig. 6 Result of Virtual Object/Annotation Overlay

Point id	Virtual Annotation	Virtual Object
0	Text: "2: Set gain to 5"	
1	Text: "1: Connect cable"	Line: to point 2
2		Line: to point 1

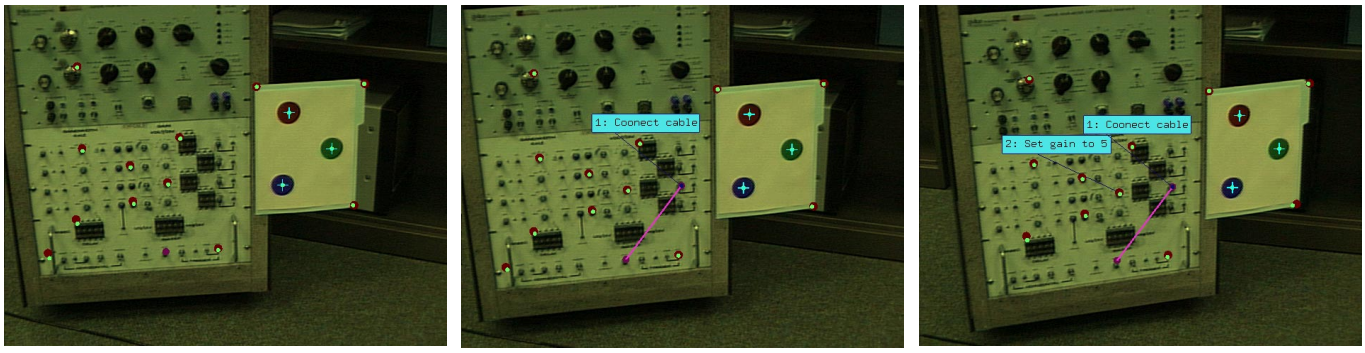
Table 1 Virtual objects/annotations with point id's

The user created the script of virtual objects/annotations described in table 1 in advance. Mouse clicking by the user, at a certain frames specified the screen coordinates of the points. Table 2 shows the user's specification of the screen coordinates of the point with the frame numbers and point id's. The 3D positions of the points were estimated by the methods described in section 5. The point 0 happened to correspond with a natural feature, hence the 3D position of the point were obtained from the 3D position of the natural feature. Point 1 and 2 did not correspond with natural features and required user's specification in two frames.

Frame number	122	128	171	184	213
Point id	1	2	2	1	0
Screen coord.	286, 237	220, 330	225, 350	307, 254	284, 267

Table 2 User's specification of the screen coordinates of the points to create virtual objects/annotations.

The 3D positions of the points combined with the specification of the virtual objects/annotations in table 1 were used to create virtual objects/annotations. As soon as the 3D positions of the points were available, the virtual objects/annotations were displayed. Figure 7 shows the addition of the virtual objects/annotations right after the frames where the 3D positions of the features were determined.



172nd frame (addition of point 2) screen coordinate is about (225, 350)	185th frame (addition of point 1) screen coordinate is about (307, 254)	214th frame (addition of point 0) screen coordinate is about (284, 267)

Fig. 7 Addition of the virtual objects/annotations after the user's specification of the screen coordinates of the points

The experiment of the current work was done in off-line but automatically. Most of the computation time goes to the natural feature tracking, which is still an order of magnitude too slow (~1Hz) for real-time applications. Dedicated hardware and DSP processor implementations of the algorithms will hopefully allow on-line and real-time applications. We plan to set up a strategy reflecting the uncertainties of 3D position and 2D variances of the features to choose features with more accurate 3D positions and 2D image coordinates. We also hope to build a more robust pose calculation algorithm, which adapts multiple features and is more tolerant to numerical instability.

7. REFERENCES

[BROI90] T. J. Broida, S. Chandrashekhar, R. Chellappa, "Recursive Estimation from a Monocular Image Sequence," IEEE Transactions on Aerospace and Electronic Systems, Vol. 26, No. 4, pp. 639-655, July 1990

[CAUD92] T. P. Caudell, D. M. Mizell, "Augmented Reality: An Application of Heads-Up Display Technology to Manual Manufacturing Processes," Proceedings of the Hawaii International Conference on Systems Sciences, pp. 659-669, 1992

[CHO97] Youngkwan Cho and Jun Park, "Fast Color Fiducial Detection and Dynamic Workspace Extension in Video See-through Self-Tracking Augmented Reality (Color Figures)", Proceedings of the Fifth Pacific Conference on Computer Graphics and Applications, Oct. 1997

[FEIN93] S. Feiner, B. MacIntyre, D. Seligmann, "Knowledge-Based Augmented Reality," Communications of the ACM, Vol. 36, No. 7, pp 52-62, July 1993

[FISC81] Fischler, M.A., Bolles, R.C. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," Graphics and Image Processing, Vol. 24, No. 6, 1981, pp. 381-395.

[FOXL96] E. Foxlin, "Inertial Head-Tracker Sensor Fusion by a Complementary Separate-Bias Kalman Filter", Proceedings of VRAIS'96, pp. 184-194

[FOXL98] E. Foxlin, M. Harrington, G. Pfeifer, "Constellation : A Wide-Range Wireless Motion-Tracking System for Augmented Reality and Virtual Set Applications", Proceedings of Siggraph98, Computer Graphics, pp. 371-378

[GHAZ95] M. Ghazisadedy, D. Adamczyk, D. J. Sandlin, R. V. Kenyon, T. A. DeFanti, "Ultrasonic Calibration of a Magnetic Tracker in a Virtual Reality Space," Proceedings of VRAIS'95, pp. 179-188

[KIM97] D. Kim, S. W. Richards, T. P. Caudell, "An Optical Tracker for Augmented Reality and Wearable Computers," Proceedings of VRAIS'97, pp. 146-150

[KLIN97] G. Klinker, K. Ahlers, D. Breem, P. Chevalier, C. Crampton, D. Greer, D. Koller, A. Kramer, E. Rose, M. Tuceryan, R. Whitaker, "Confluence of Computer Vision and Interactive Graphics for Augmented Reality," Presence: Teleoperator and Virtual Environments, Vol. 6, No. 4, pp. 433-451, August 1997

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

[MELL95] J. P. Mellor, "Enhanced Reality Visualization in a Surgical Environment," Master's Thesis, Dept. of Electrical Engineering, MIT, 1995

[MEND95] J.M. Mendel, *Lessons in Estimation Theory for Signal Processing, Communications, and Control*, Prentice Hall PTR, 1995

[MEYE92] K. Meyer, H. L. Applewhite, F. A. Biocca, "A Survey of Position Trackers," Presence: Teleoperator and Virtual Environments, Vol. 1, No. 2, pp. 173-200, 1992

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

[NEUM98a] U. Neumann, S. You, "Integration of Region Tracking and Optical Flow for Image Motion Estimation," to appear in proceedings of IEEE ICIP-98, Oct. 1998.

[NEUM98b] U. Neumann, J. Park, "Extendible Object-Centric Tracking for Augmented Reality", 1998 IEEE Virtual Reality Annual International Symposium

[NEUM98c] U. Neumann, J. Park, " Tracking for Augmented Reality on Wearable Computers ", British Journal on Virtual Reality, under process

[REKI97] J. Rekimoto, "NaviCam: A Magnifying Glass Approach to Augmented Reality," Presence: Teleoperator and Virtual Environments, Vol. 6, No. 4, pp. 399-412, August 1997

[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

[STAT96] A. State, G. Hirota, D. T. Chen, B. Garrett, M. Livingston, "Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking," Proceedings of Siggraph96, Computer Graphics, pp. 429-438

[TREM97] A. Tremeau, N. Borel, "A Region Growing and Merging Algorithm to Color Segmentation," Pattern Recognition, Vol. 30, No. 7, pp. 1191-1203, 1997

[UENO95] M. Uenohara, T. Kanade, "Vision-Based Object Registration for Real-Time Image Overlay," Proceedings of Computer Vision, Virtual Reality, and Robotics in Medicine, pp. 13-22, 1995

[WARD92] M. Ward, R. Azuma, R. Bennett, S. Gottschalk, H. Fuchs, "A Demonstrated Optical Tracker with Scalable Work Area for Head-Mounted Display Systems," Proceedings of the 1992 Symposium on Interactive 3D Graphics," pp. 43-52

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