Extendible Tracking by Line Auto-Calibration

Bolan Jiang, Ulrich Neumann
Computer Graphics and Immersive Technology Laboratory
Integrated Media Systems Center
University of Southern California
{bjiang|uneumann}@usc.edu

Abstract

One of the key requirements for an augmented reality system is a tracking system that determines the user's viewpoint accurately. Many vision-based tracking systems exhibit limited tracking range due to their dependence on pre-calibrated features. Previous extendible-tracking systems only make use of point features. This paper describes an extendible tracking method that integrates pre-calibrated landmarks and natural line features for camera tracking. Extendible and robust tracking is achieved by dynamically calibrating prior uncalibrated 3D structure of line features. The experimental results indicate the effectiveness of this approach for extending the tracking range and reducing dependence on the prepared environment.

1. Introduction

An augmented reality system inserts virtual objects into the user's view of the real environment. It should appear to the user that the virtual objects (text, images, or models) actually exist in the real environment. One of the key requirements to achieve this illusion is a tracking system that can accurately determine the user's viewpoint (or camera pose).

Many tracking systems utilizing vision techniques [1, 2, 4, 5, 6] can achieve accuracies suitable for an augmented reality system. However, these systems require prepared environments where the system operator can put and calibrate artificial landmarks. Tracking is only successful when the pre-calibrated landmarks are visible. The dependence on calibrated landmarks limits the range and usefulness of these tracking methods, especially in large-scale and outdoor environments. The extendible tracking methods proposed in [3, 11] reduce the dependence on wide-area calibration by auto-calibrating unknown point features in the environment. The tracking range can be extended from a small prepared area occupied by pre-calibrated landmarks to a neighboring unprepared area where none of the pre-calibrated landmarks are in the user's view.

The methods mentioned above only make use of point features. Straight lines are also prominent features in most man-made environments. Lines can be detected and tracked reliably, and they provide a great deal of information about the structure of the scene. Though pre-calibrated line features have been used in [10, 13], no extendible tracking method for using prior uncalibrated line features has been reported.

In this paper, we develop an Extended Kalman Filter (EKF) tracking framework that extends tracking range from a prepared area to a neighboring unprepared area by detecting and auto-calibrating a-priori unknown line features in the environment. In our system, both pre-calibrated landmarks and natural line features are used to estimate camera pose. Pre-calibrated landmarks initiate the tracking process, and extendible tracking is achieved by dynamically calibrating line features. This method is an extension to our previous work [11] on point feature auto-calibration. A line feature is modeled as an infinite straight line and its observed line segments in different views may correspond to different portions of the same line. The experimental results indicate that the dependence on pre-calibrated landmarks is reduced by line auto-calibration.

The remainder of this paper is as follows: Section 2 presents the representations of 3D structure of lines and their image projections. Section 3 presents the tracking algorithm that estimates camera pose and auto-calibrates line features. Section 4 presents the results of experiments.

2. Problem formalization

Several representations for a 3D line have been proposed. A 3D line can be represented by plüker coordinates [17] that use a 6-dimention vector

$(l_{12}, l_{23}, l_{31}, l_{14}, l_{24}, l_{34})$ to define a 3D line up to a scale factor if and only if

$$l_{12}l_{14} + l_{23}l_{24} + l_{31}l_{34} = 0 \quad (1)$$

Unfortunately, it is hard to maintain the constraint (1) in an EKF framework. A 3D line can also be represented by two points on the line [10], or by a point on the line and the direction of the line [9]. However, neither of these is a minimal representation. A minimal representation is proposed in [12], which makes use of the observed line segment on the first image. This representation assumes that the world coordinates align with the first camera coordinates and all the lines are observable in the first view. These two assumptions are not suitable for AR applications.

In the following section we present a new minimal representation for a 3D line and discuss how to compute its image projection from this representation.

2.1 Line representation

A 3D line can be represented by the intersection of two planes. Because two planes together have six degrees of freedom (DOF) while a 3D line only has four DOF, a representation as two planes is not minimal. If a plane passes through a fixed point T_l, in a local coordinate system whose origin is T_l and transformation to world coordinates is

$$X_w = R_l X_l + T_l, \quad (2)$$

the plane can be represented in the local coordinate by

$$X_l^T n_l = 0. \quad (3)$$

The vector n_l has a direction normal to the plane. Without loss of generality, we can assume the projection of n_l on the z-axis is fixed as unit length. So $n_l = (n_x, n_y, 1)$ and the plane can be represented by a two-dimension variable (n_x, n_y) in the local coordinate frame. This representation only has two DOF, which meets our requirements for a minimal line representation. So our representation of a 3D line has two parts: one is the variable part $(n_{x1}, n_{y1}, n_{x2}, n_{y2})$, representing the normals to two planes in their own local coordinates. The other part is the constant (T_1, R_1, T_2, R_2) that represents the transformation between each local coordinate system to world coordinates. Given (T_1, R_1, T_2, R_2), $(n_{x1}, n_{y1}, n_{x2}, n_{y2})$ uniquely determines a 3D line L.

2.2 Image projection of 3D line

As shown in Fig. 1, a 3D line L and the camera center O determine a plane π, whose normal is N. The image projection of the 3D line is the intersection of the plane π and the image plane.

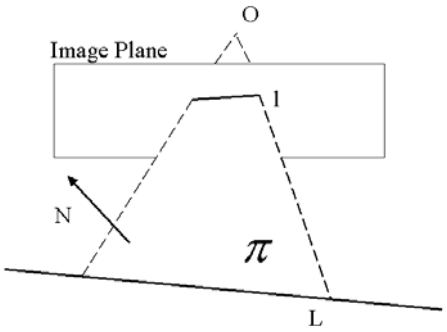

Fig. 1 Perspective projection of 3D line

Suppose that L is represented by the minimal representation of planes π_1 and π_2. After transformation to camera coordinates, $\pi_1 = \begin{pmatrix} n_1 \\ d_1 \end{pmatrix}$ and $\pi_2 = \begin{pmatrix} n_2 \\ d_2 \end{pmatrix}$, where n_1 and n_2 are three-dimension vectors, and d_1, d_2 are real numbers. Since plane π passes through the camera center O, it can be represented in camera coordinates as $\begin{pmatrix} n \\ 0 \end{pmatrix}$, where n is a three-dimension vector and has the same direction as N.

Due to the fact that π, π_1 and π_2 intersect at a line, π is a linear combination of π_1 and π_2. Noting that the last element of π is zero, it follows that

$$n = d_2 n_1 - d_1 n_2. \quad (4)$$

Because n has the same direction as the normal to plane π, any point X_c on π satisfies:

$$X_c^T n = 0, \quad (5)$$

including the points on the image projection l. So any point on image projection l with image coordinate (x_i, y_i), satisfies

$$x_i B_x + y_i B_y + B_z = 0, \quad (6)$$

where $\begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} = K^{-T} n$ and $K = \begin{bmatrix} fu & 0 & U0 \\ 0 & fv & V0 \\ 0 & 0 & 1 \end{bmatrix}$.

The 2D image line l can also be represented as

$$x_i \cos\theta + y_i \sin\theta + \rho = 0 \quad (7)$$

where $\theta \in [-\pi, \pi], \rho \in (-\infty, +\infty)$.

Comparing equation (6) and (7), we can get the image projection of the line L

$$l = [B_x / \sqrt{B_x^2 + B_y^2}, B_y / \sqrt{B_x^2 + B_y^2}, B_z / \sqrt{B_x^2 + B_y^2}]. \quad (8)$$

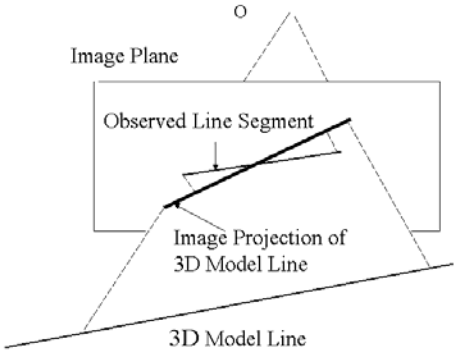

Fig. 2 The image projection of a 3D line and its corresponding observed line segment.

3. Camera tracking and line auto-calibration

An Extended Kalman Filter estimates camera pose and calibrates the 3D structure of unknown lines in the environment. The camera state is represented by position, incremental orientation, and their first derivatives: $[x, y, z, \dot{x}, \dot{y}, \dot{z}, \Delta\phi, \Delta\theta, \Delta\psi, \dot{\phi}, \dot{\theta}, \dot{\psi}]$. The accumulated orientation is maintained as a quaternion externally. The internal camera parameters are assumed known and fixed during the tracking session.

Basically the EKF has two main processes: prediction and measurement update. The prediction process uses prior state information and a dynamic model to predict the current camera state. More information about prediction and dynamic models can be found in [4, 11].

3.1 Measurement update

The measurement update uses the discrepancy between the predicted image line projection and the observed line segment to refine the estimates for camera pose and 3D line auto-calibration. The predicted image projection is computed from the predicted camera state and line state, as in section 2.2. As shown in Fig. 2, an observed line segment is represented by its two end points (x_1, y_1) and (x_2, y_2), and its corresponding image line projection is an infinite line represented by (m_x, m_y, m_z). Then

$$\begin{aligned} h_1 &= x_1 m_x + y_1 m_y + m_z \\ h_2 &= x_2 m_x + y_2 m_y + m_z \end{aligned} \tag{9}$$

where $m_x^2 + m_y^2 = 1$.

reflect the signed distances from the end points to the predicted image projection, and the vector $\begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$ serves as the measurement in our system.

Suppose that X^- and P^- are the predicted camera state and variance from the prediction process. For each observed line segment with a corresponding auto-calibrated line, we augment camera state and variance with line state and variance to get $\hat{X}^-$ and $\hat{P}^-$. Letting

$$\begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = f(\hat{X}^-, x_1, y_1, x_2, y_2), \tag{10}$$

the estimate for the camera pose and the 3D line is refined as:

$$\begin{aligned} \hat{X} &= \hat{X}^- - P^- C^T (C P^- C^T + D R D^T)^{-1} f(\hat{X}^-, x_1, y_1, x_2, y_2) \\ \hat{P} &= \hat{P}^- - \hat{P}^- C^T (C P^- C^T + D R D^T)^{-1} C \hat{P}^- \end{aligned} \tag{11}$$

where $C = \dfrac{\partial f}{\partial \hat{X}}$ and $D = \dfrac{\partial f}{\partial (x_1, y_1, x_2, y_2)}$.

R is the measurement noise matrix for the end points.

The measurement update is processed iteratively in that the estimate is refined by applying measurement correction from only one line segment at one time. For the lines observed in a single frame, the one with small uncertainty for 3D structure is used first for the measurement update. For each frame, at the beginning of measurement update, the lines with small uncertainty can lead fast converge of the camera state, and then the converged camera state can lead feast converge of lines with large uncertainty.

3.2 Initialization for 3D line auto-calibration

If a line is observed in two images, its 3D structure can be recovered by the two back projected planes from the observed line segments as:

$$\begin{aligned} \pi_1 &= P_1^T l_1 \\ \pi_2 &= P_2^T l_2 \end{aligned} \tag{12}$$

where P_1 and P_2 are the camera projection matrices, l_1 and l_2 are the observed line segments.

However, π_1 and π_2 are not the minimal representation for line L, some extra work is needed to transform π_1 and π_2 to the minimal representation π_1' and π_2', represented by $(n_{x1}, n_{y1}, n_{x2}, n_{y2})$ given (T_1, R_1, T_2, R_2).

Suppose any point X_w on the line L satisfies:

$$\begin{aligned} \pi_1 &: X_w^T n_1 + d_1 = 0 \\ \pi_2 &: X_w^T n_2 + d_2 = 0 \end{aligned}. \tag{13}$$

For any local coordinate (T_1, R_1) chosen for π_1', π_1 and π_2 can be represented in this local coordinate as:

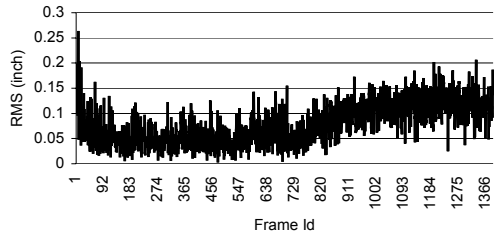

Fig. 3 (a) camera position error

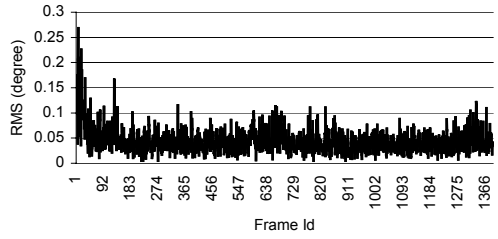

Fig. 3 (b) camera orientation error

Fig. 3 Errors for estimated camera state

$$X_1^T R_1^T n_1 + T_1^T n_1 + d_1 = 0$$
$$X_1^T R_1^T n_2 + T_1^T n_2 + d_2 = 0 \quad (14)$$

then plane π_1' is represented by

$$n_{x1} = L_x / L_z$$
$$n_{y1} = L_y / L_z \quad (15)$$

where

$$[L_x, L_y, L_z]^T = R_1^T(\lambda n_1 - \mu n_2)$$
$$\lambda = T_1^T n_2 + d_2 \quad . \quad (16)$$
$$\mu = T_1^T n_1 + d_1$$

The representation for the plane π_2' is computed similarly. After the initialization, the line estimate is refined by the measurement update in section 3.1.

In theory, (T_1, R_1, T_2, R_2) can be chosen arbitrarily. However, different (T_1, R_1, T_2, R_2) may cause different errors for the initial estimate of $(n_{x1}, n_{y1}, n_{x2}, n_{y2})$, which may lead to different performance during the tracking process.

In our implementation, we choose the estimated camera positions for two views as T_1 and T_2. We choose R_1 so that the z-axis of the local coordinate is aligned with the initial estimate of $\lambda n_1 - \mu n_2$. R_2 is determined similarly.

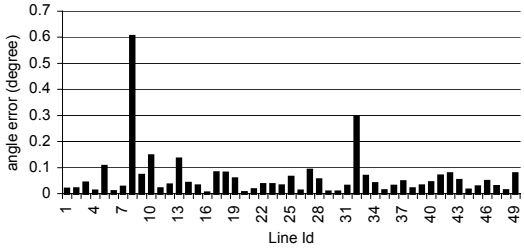

Fig. 4 (a) angles between auto-calibrated lines and the corresponding true lines

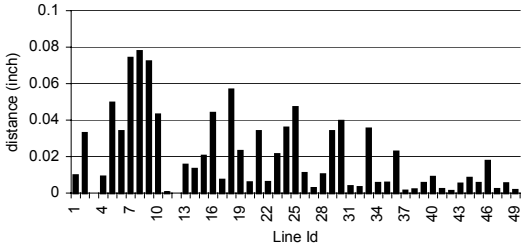

Fig. 4 (b) distances between auto-calibrated lines and the corresponding true lines

Fig. 4 Errors for auto-calibrated lines

To ensure a good initial estimate, the line is not estimated until the angle between two back projected planes exceed some threshold. The threshold is set between three and five degrees in our tests.

4. Experiments

Our line-based tracking method is tested with both simulations and real experiments.

4.1 Simulation

In a simulation test, 50 points are distributed evenly in a 100x30x20 in^3 volume. Each point is taken as the center of a unit sphere. A point is selected on the surface of each sphere with uniform distribution to define a line through the initial center point. The 3D structure of all 50 lines are assumed unknown. Six pre-calibrated points are placed on the left side of the volume. The camera movement starts with the pre-calibrated features in view. Then the camera is gradually translated and rotated away from the pre-calibrated features. The whole sequence has 1391 frames, and none of the pre-calibrated features are observable after the 560^{th} frame. For the image measurements, 0.5-pixel standard-deviation Gaussian noise is added to the end-point coordinates of the observed line segments. The estimated camera state is compared with the true camera state, and RMS errors for camera position and orientation

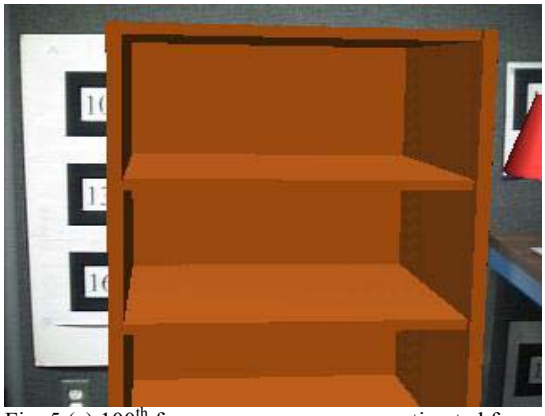

Fig. 5 (a) 100th frame, camera pose was estimated from pre-calibrated landmarks on the board behind the bookcase.

Fig. 5 (b) 1300th frame, camera pose was estimated from both pre-calibrated landmarks and auto-calibrated line features

Fig. 5 (c) 2100th frame, camera pose was estimated from auto-calibrated line features. None of the pre-calibrated landmarks were observable.

Fig.5 Snapshots from the indoor real experiment

are computed. As shown in Fig. 3, the errors for camera positions are below 0.3 inch and the errors for camera orientation are below 0.3 degree. Although drift increases

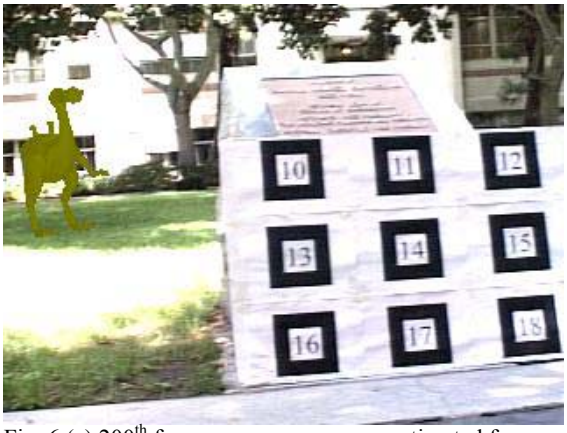

Fig. 6 (a) 200th frame, camera pose was estimated from pre-calibrated landmarks on the board.

Fig. 6 (b) 400th frame, camera pose was estimated from the pre-calibrated landmarks and auto-calibrated line features.

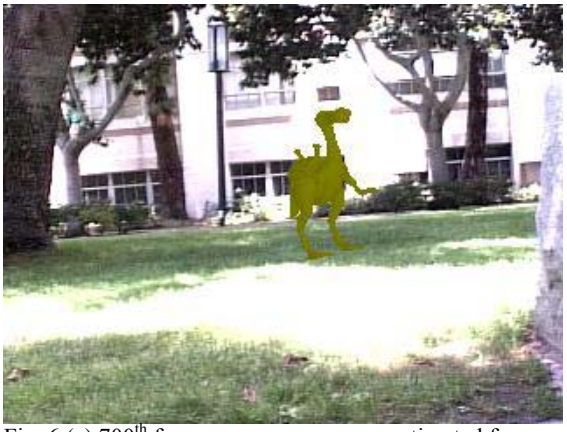

Fig. 6 (c) 700th frame, camera pose was estimated from auto-calibrated line features. None of the pre-calibrated landmarks were observable.

Fig. 6 Snapshots from the outdoor real experiment

after pre-calibrated features are out of view, the drift rate is modest. As shown in Fig. 4, all of 50 auto-calibrated lines,

the angle errors are well under 0.6 degree. And the distance errors are well under 0.08 inch.

4.2 Real experiment

In real experiments, a calibration board is put in the environment of the interests. The camera movement is started where a calibration board is in view. And then camera is moved to an unprepared neighboring region where none of the pre-calibrated landmarks are observable. Only the nine black-square landmarks on the board are pre-calibrated. The video sequences for the real experiments are recorded and digitized in real time. Line features are detected offline by the Vista system [19], and the observed lines segments on two consecutive frames are matched by the algorithm in [15].

Our line-based tracking method has been tested both indoor and outdoor. For the indoor sequence, a virtual bookcase, a lamp, and a chair are inserted into an office. For the outdoor sequence, a dinosaur is put on the grassland among the buildings. The sample images in Fig. 5 and Fig. 6 show that all the virtual objects remaining in their correct locations for both sequences.

Mpeg video files of the real experiments are available on http://deimos.usc.edu/~bjiang/isar01.html.

5. Conclusion

We present an EKF-based extendible tracking method that can extend tracking from a small prepared region to neighboring unprepared regions of the environment by auto-calibrating the 3D structure of a-priori unknown line features. We present a minimal representation for 3D lines, and a measurement update model for the EKF based on this representation. The experimental results indicate that the dependence on the prepared environment can be reduced by auto-calibrating 3D structure of the line features, and that camera pose and line structure can be recovered accurately.

Acknowledgements

This work received funding from ONR and the MURI program. Funding and research facilities are also provided by the NSF through its ERC funding of the Integrated Media Systems Center. We recognize the assistance of many members in the CGIT Laboratory at USC. Special thanks go to Suya You for discussion and Douglas Fidaleo for assistance in preparing the 3D models used in our demonstrations.

References:

[1] R.Azuma Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments 6 (4), 355-385 (August 1997).

[2] U.Neumann and Y. Cho, A Self-Tracking Augmented Reality System. Proceedings of ACM Virtual Reality Software and Technology, 109-115 (July 1996).

[3] U.Neumann and J.Park, Extendible Object-Centric Tracking for Augmented Reality. Proceedings of IEEE Virtual Reality Annual International Symposium 1998, 148-155 (March 1998).

[4] J.Park, B. Jiang, and U. Neumann, Vision-based Pose Computation: Robust and Accurate Augmented Reality Tracking. Proceedings of International Workshop on Augmented Reality (IWAR)'99 (October 1999).

[5] A. State, G. Hirota, D. Chen, B. Garrett, and M. Livington, Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking. Proceedings of SIGGRAPH '96, 429-438 (August 1996)

[6] G. Welch and G. Bishop, SCAAT: Incremental Tracking with Incomplete Information. Proceedings of SIGGRAPH '97, 333-344 (August 1997)

[7] D. Morris and T. Kanade, A unified factorization algorithm for points, line segments and planes with uncertainty models, ICCV 696-702, 1998.

[8] C.J. Taylor, D. Kriegman, Structure and Motion from Line Segments in Multiple Images, PAMI 17:11 1021-1032, 1995.

[9] J. Weng, T. Huang and N. Ahuja, Motion and Structure from Line Correspondences: Closed-Form Solution, Uniqueness, and Optimization, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 14, No. 3, 318-336, March 1998.

[10] R. Kumar and A. Hanson, Robust Methods for Estimating Pose and a Sensitivity Analysis, CVGIP : Image Understanding, Vol 60, No. 3, November, 313-342, 1994.

[11] B. Jiang, S. You and U. Neumann, Camera Tracking for Augmented Reality Media, Proceedings of IEEE International Conference on Multimedia and Expo 2000, 1637-1640, 30 July - 2 August 2000, New York, NY.

[12] T. Vieville and O. Faugeras, Feed-Forward Recovery of Motion and Structure from a Sequence of 2D-Lines Matches, Proceedings of International Conference on Computer Vision, 517-520, 1990.

[13] D. Stricker, G. Klinker, and D. Reiners, A Fast and Robust Line-based Optical Tracker for Augmented Reality Applications, Proceedings of International Workshop on Augmented Reality, 129-145, 1998.

[14] S. You and U. Neumann, Fusion of Vision and Gyro Tracking for Robust Augmented Reality Registration, Proceedings of IEEE Virtual Reality, March 2001.

[15] N. Chiba and T. Kanade, A Tracker for Broken and Closely spaced lines, ISPRS, Vol. XXXII, No. 5, pp. 676-683, 1998.

[16] S. Soatte, R. Frezza, and P. Perona, Motion estimation via dynamic vision, IEEE Transaction on Automatic Control, 41(3): 393-414, March 1996.

[17] A. Beutelspacher and U. Rosenbaum, Projective Geometry, Cambridge University Press, ISBN 0-521-48364-6.

[18] Y. Liu, T. Huang and O. Faugeras, Detemination of Camera Location from 2-D to 3-D Line and Point Correspondences, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No. 1, January 1990.

[19] http://www.cs.ubc.ca/nest/lci/vista/vista.html