3D TRACKING USING 2D-3D LINE SEGMENT CORRESPONDENCE AND 2D POINT MOTION

Woobum Kang
*Kyoto University**
Gokasho, Uji, Kyoto 611-0011 Japan

Shigeru Eiho
The Kyoto College of Graduate Studies for Informatics
7 Monzen-cho, Tanaka, Sakyo-ku, Kyoto 606-8225 Japan

Keywords: 3D tracking, CAD model, edge, feature point.

Abstract: In this paper, we propose a 3D tracking method which integrates two kinds of 2D feature tracking. Our tracker searches 2D-3D correspondences used to estimate camera pose on the next frame from detected straight edges and projected 3D-CAD model on the current frame, and tracks corresponding edges on the consecutive frames. By tracking those edges, our tracker can keep correct correspondences even when large camera motion occurs. Furthermore, when the estimated pose seems incorrect, our tracker brings back to the correspondences of the previous frame and proceeds tracking of corresponding edges. Then, on the next frame, our tracker estimates the pose from those correspondences and can recover to the correct pose.
Our tracker also detects and tracks corners on the image as 2D feature points, and estimates the camera pose from 2D-3D line segment correspondences and the motions of feature points on the consecutive frames. As the result, our tracker can suppress the influence of incorrect 2D-3D correspondences and can estimate the pose even when the number of detected correspondences is not enough.
We also propose an approach which estimates both the camera pose and the correspondences. With this approach, our tracker can estimate the pose and the correspondence on the initial frame of the tracking.
From experimental results, we confirmed our tracker can work in real-time with enough accuracy for various applications even with a less accurate CAD model and noizy low resolution images.

1 INTRODUCTION

Image-based markerless 3D tracking is one of the important issues. One of the well-known approaches for the 3D tracking, called as model-based approach, estimates the camera pose from 2D-3D correspondences between 2D feature and 3D model. As there are many approaches (Liu et al., 1990), (Christy and Horaud, 1999) to estimate the pose from various kinds of 2D-3D feature correspondence (line, point, etc.), we can estimate the pose correctly if sufficient number of correspondences are obtained on every frame of tracking. However, this is difficult in the real situation. Various corresponds estimation approaches for model-based 3D tracking have been proposed. Lowe (Lowe, 1992) proposed an edge-based iterative pose and correspondence estimation approach provided that approximate pose is obtained as the initial guess. Drummond et al. (Drummond and Cipolla, 2002) proposed real-time 3D tracking method using the 2D-3D edge point correspondence.

The weaknesses of 2D-3D model based approach are: (1)They cannot estimate or misestimate when the number of correspondences is not sufficient due to motion blur or measurement error of both 3D model and intrinsic parameters of the camera. (2)In the methods such as (Lowe, 1992) and (Drummond and Cipolla, 2002), which estimate 2D-3D correspondences by projecting the 3D model using the pose on the previous frame and nearest 2D feature search, once the tracker estimates an incorrect pose, it cannot obtain correct correspondences on the latter frames.

Vacchetti et al. (L. Vacchetti and Fua, 2004b) proposed a tracking method using 2D-3D feature point correspondences and feature point motions for the pose estimation. They also proposed a tracking method(L. Vacchetti and Fua, 2004a) which integrates their feature-point-based method(L. Vacchetti and Fua, 2004b) and the edge-based methods proposed in (Drummond and Cipolla, 2002), (Comport

[1]Most part of this research was done when the authors belonged to Kyoto University. The first author currently works for KEYENCE Corp.

Kang W. and Eiho S. (2006).
3D TRACKING USING 2D-3D LINE SEGMENT CORRESPONDENCE AND 2D POINT MOTION.
In *Proceedings of the First International Conference on Computer Vision Theory and Applications*, pages 278-285
DOI: 10.5220/0001365402780285

et al., 2003). By integrating 2D feature motions on the consecutive frames, they cover the first weakness of model-based approach. However, their tracker can only use the feature points on the surface of tracked object and does not use those on the back ground.

We here propose a markerless 3D tracking method which combines image-based 2D feature tracking. We estimate the pose from 2D-3D correspondences between straight edges and 3D line segments of the CAD Model and feature point motions on the consecutive frames, where the corners detected on the image are used as the feature points.

Different from the method proposed on (L. Vacchetti and Fua, 2004b), our method does not restrict the position of feature points and can handle large camera motions by using a strong image-based straight edge tracking method. Moreover, by introducing special 2D-3D correspondence update process, our tracker can keep 2D-3D correspondences and can track corresponding edges even if quite a wrong pose is estimated for some numerical failures. And our tracker can recover to the correct pose on the latter frame by estimating the pose from stored correspondences obtained just before the incorrect pose estimation.

On the initial frame of our 3D tracking, it is necessary to obtain the 2D-3D correspondences and the pose. We also propose a method which estimates both the pose and the correspondence by using the approximate initial guess of the pose. With this method, our tracker can estimate the pose and the correspondence on the initial frame automatically.

2 OVERVIEW OF OUR 3D TRACKING METHOD

Our 3D tracker estimates the camera pose relative to the world coordinate system on every frame derived from a single camera. We assume that intrinsic parameters of the camera are known. We estimate the camera pose from (1) 2D-3D line segment correspondences between straight edges detected on the image and CAD model, and (2) motions of the feature points on the consecutive frames. We use corners on the image extracted with Tomasi-Kanade method(Shi and Tomasi, 1994) as the feature points. Our tracker tracks these points on the consecutive frames by calculating their optical flow using Lucas-Kanade method(Lucas and Kanade, 1981), and uses their motions to estimate the pose. We assume that the pose and the 2D-3D line segment correspondences on the initial frame are provided.

We explain the outline of our method using fig.1. (I) shows projected 3D CAD model (thin line), straight edges corresponding to the model line segments (bold line), and detected feature points. When the new frame comes in, our tracker tracks the edges corresponding to the model line segments and feature points detected on the previous frame shown as figures (II) and (III). By tracking those edges on the consecutive frames, our tracker can estimate the correct position of those edges even when the large camera motion occurs. After the 2D feature tracking, our tracker eliminates outliers of motions of feature points by fitting fundamental matrix with LMedS method. Then, as figure (IV), our tracker estimates the pose from both 2D-3D line segment correspondences and feature point motions on the consecutive frames. After the pose estimation, shown as the figures (V) and (VI), 2D-3D line segment correspondences are updated by checking the distances between straight edges detected on the current frame and projected 3D CAD model lines.

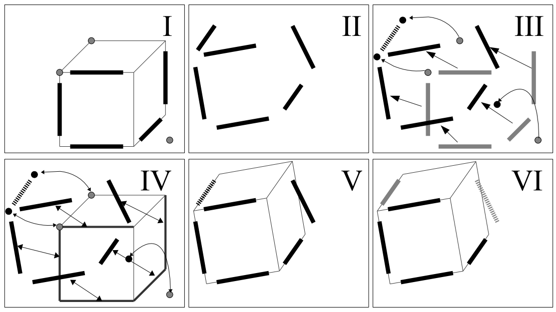

Figure 1: Straight edge and feature point tracking, pose estimation, and 2D-3D correspondence update. [I:Projected CAD model (thin line), edges corresponding to the model (bold line), and detected feature points. II:Edges detected on the next frame of I. III:Our tracker tracks edges corresponding to the model (solid bold line) and motions of feature points. IV and V:Our tracker estimates the pose from 2D-3D line segment correspondences and feature point motions. VI:Our tracker eliminates incorrectly corresponding edges (dashed gray line) and searches newly found corresponding edges (solid gray line).]

3 STRAIGHT EDGE TRACKING AND CORRESPONDENCE UPDATE

3.1 Straight Edge Tracking and Detection

Straight edges corresponding to the 3D model line segments are tracked on the consecutive frames. This straight edge tracking is performed from two steps, estimation and matching.

On the estimation step, our edge tracker estimates the motion of the edge by calculating optical flow of all edge pixels for every tracked straight edge. And the tracker fits a line from calculated destinations of edge pixels and eliminates outliers. If those destinations are not on a line, our tracker regards the estimation was failed and stops tracking for the edge.

On the matching step, our tracker searches the edges near the estimated destination. If straight edges exist around there, the tracker regards one on the nearest as the corresponding one. If there is no corresponding edge but most destinations of the edge pixels are on a line, the tracker constructs a straight edge from estimated destinations of edge pixels and regards it as the corresponding one. Unlike the straight edge tracking method proposed by Chiba et al. (Chiba and Kanade, 1998), our edge tracker can track even when the corresponding edge is not detected properly due to motion blur and illumination changes.

On every frame, our tracker detects straight edges on the image by using Canny edge detector(Canny, 1986) for the edge detection. Like the method proposed by Lowe(Lowe, 1987), our tracker detects straight edges by splitting connected edges until all split edges become straight. Then, to reduce the fragments, edges on the neighbor and on a line are merged.

3.2 Correspondence Update Process

After estimating the pose, our tracker updates 2D-3D correspondences by checking the distances between projected 3D model line segments and straight edges. In the model projection, hidden lines are removed in order to suppress incorrect 2D-3D correspondences. We define the distance and the overlapping ratio between straight edge and corresponding projected model line segment using the distances d_1, d_2 and the lengths l, l' as shown in fig.2. Where d_1, d_2 is defined as the point-to-line distance between each end point of the projected 3D line segment and the line obtained by extending the straight edge. We define the distance d and the overlapping ratio γ as follows.

$$\begin{cases} d &= \frac{1}{2}\left(d_1^2 + d_2^2\right)^{\frac{1}{2}} \\ \gamma &= l/l' \end{cases} \tag{1}$$

Correspondence update is done by the following two steps; (1) elimination of incorrect correspondences from those currently used for the pose estimation, and (2) addition of new correspondences.

On the elimination step, our tracker calculates the distance for every 2D-3D line segment pair currently regarded as corresponding each other, and estimates the standard deviation of distances $\hat{\sigma}_d$ using MAD(Median Absolute Deviation)(G.A.F. Seber, 1981). Then, the correspondences whose distances

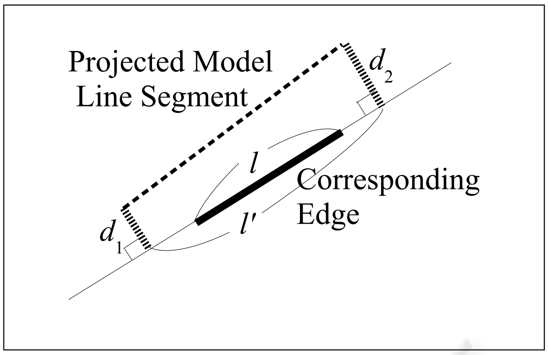

Figure 2: Distance and overlapping ratio between a 3D model line segment and corresponding straight edge.

d are larger than $\beta\hat{\sigma}_d$ are regarded as incorrect ones and eliminated, where β is a constant and its value is around 2-3. As threshold $\beta\hat{\sigma}_d$ becomes very large if the large displacements occur between the projected models and its corresponding edges, our tracker keeps 2D-3D correspondences of the previous frame if quite a different pose is estimated.

On the addition step, our tracker searches new correspondences by checking the distances between projected model line segments and straight edges, both of which has no corresponding edge or no corresponding model line segment. If their distance is $d < d_{corres}$ and the overlapping ratio is $\gamma > \gamma_{corres}$, our tracker adds this pair to the correspondences which are used in the pose estimation, where d_{corres} and γ_{corres} are constants, and we set d_{corres} around 2-3[pixel] and $\gamma_{corres} = 0.3$.

4 POSE ESTIMATION FOR THE TRACKING

4.1 Pose from Known 2D-3D Line Segment Correspondence

In this method, we estimate the pose by minimizing the following objective function:

$$f(R_n, \boldsymbol{t}_n) = \sum_{l=1}^{L} w_l \phi_l^2 + g(R_n) \tag{2}$$

where $\phi_l(R_n, \boldsymbol{t}_n)$ is error term in 2D-3D correspondence, $g(R_n)$ is constraint term for rotation matrix R_n, and w_l is weighting coefficients introduced to eliminate outliers.

We define the correspondence error between a 2D line segment (straight edge) and corresponding 3D line segment as follows: Considering fragments of straight edges, we define the error as the sum of point-to-line distances between each end point of projected 3D line segment and extended straight edge as fig. 2.

When one of the end points of a 3D line segment is represented as $\boldsymbol{X} = [X, Y, Z]^T$ in the world coordinate system and represented as $\boldsymbol{X'} = [X', Y', Z']^T$ in the camera coordinate system, the relationship between $\boldsymbol{X}$ and $\boldsymbol{X'}$ is described as follows:

$$\boldsymbol{X'} = R\boldsymbol{X} + \boldsymbol{t} \tag{3}$$

Therefore, squared point-to-line distance between projected end point and 2D line $l : ax + by + c = 0(a^2 + b^2 = 1)$ is described as follows.

$$d^2 = \left(a\frac{X'}{Z'} + b\frac{Y'}{Z'} + c\right)^2 \tag{4}$$

By substituting $1/Z'$ to a scale parameter μ in order to simplify the objective function, the distance is described as follows;

$$\begin{aligned} d'^2 = \mu^2\left(aX' + bY' + cZ'\right)^2 &= \mu^2\left(\boldsymbol{n}^T\boldsymbol{X}'\right)^2 \\ &= \mu^2\left[\boldsymbol{n}^T\left(R\boldsymbol{X} + \boldsymbol{t}\right)\right]^2 \end{aligned} \tag{5}$$

where $\boldsymbol{n} = [a, b, c]^T$. Then, 2D-3D line segment correspondence error is defined as squared sum of d'. When a 2D line segment l_i is corresponding to the 3D line segment L_j, its correspondence error ϕ becomes:

$$\phi(l_i, L_j, R, \boldsymbol{t}) = \sum_{k=1}^{2}\mu_{jk}^2\left[\boldsymbol{n}_i^T\left(R\boldsymbol{X}_{jk} + \boldsymbol{t}\right)\right]^2 \tag{6}$$

where $\boldsymbol{X}_{jk}(k = 1, 2)$ is end point of the 3D line segment L_j and μ_{jk} is its scale parameter.

Weighting coefficient w_l is determined from correspondence error ϕ_l. We set w_l from Tukey's ρ function;

$$w = \begin{cases} \left[1 - \left(\frac{x}{C}\right)^2\right]^2 & |x| \le C \\ 0 & |x| \ge C \end{cases} \qquad x = \frac{e}{\hat{\sigma}} \tag{7}$$

where e is the error (ϕ_l for w_l) and C is a constant. To determine w_l from eq.7, it is necessary to estimate the standard deviation of errors. According to MAD, standard deviation is estimated as;

$$\hat{\sigma}_{MAD} = 1.4826 \text{ median } \{|\phi_1|, ..., |\phi_L|\} \tag{8}$$

If we use MAD, however, 2D-3D correspondences necessary to estimate the pose uniquely are sometimes regarded as outliers. We therefore set $\hat{\sigma}$ from maximum absolute correspondence error of those necessary to estimate the pose uniquely if the plenty number of correspondences are not obtained. By writing this maximum absolute error as $|\phi'|$, $\hat{\sigma}$ is determined as follows.

$$\hat{\sigma} = \max\{1.4826|\phi'|, \hat{\sigma}_{MAD}\} \tag{9}$$

To decrease the number of variables, we represent rotation component of the pose by a quaternion $\mathbf{r}_n$ instead of a rotation matrix R_n. The number of variables representing the camera pose is reduced from twelve (nine for R_n and three for $\boldsymbol{t}_n$) to seven (four for $\mathbf{r}_n$ and three for $\boldsymbol{t}_n$). The objective function is rewritten as follows;

$$f(\mathbf{r}, \boldsymbol{t}) = \sum_{l=1}^{L} w_l\phi_l^2 + g(\mathbf{r}) \tag{10}$$

On the above equation, $g(\mathbf{r})$ becomes the constraint term for the rotation quaternion. The minimization of the objective function is done by repeating weighting coefficients determination and pose parameter estimation. Pose parameter estimation is done by using nonlinear minimization techniques such as Gauss-Newton approach. The estimation procedure is described as follows.

1. Set coefficients as $w_1 = w_2 = ... = w_L = 1$ and set $R_{n-1}, \boldsymbol{t}_{n-1}$ as initial guess of the pose parameters $R_n, \boldsymbol{t}_n$.
2. Update the pose parameter by the following procedure.
   (a) Compute the scale parameters $\mu_{lk} = 1/Z'_{lk}$, $(k = 1, 2)$ and ν_{m1}, ν_{m2} from currently estimated pose $R_n, \boldsymbol{t}_n$.
   (b) Convert R_n to a quaternion $\mathbf{r}_n$, and update the pose to the one which decreases $f(\mathbf{r}_n, \boldsymbol{t}_n)$. Then, convert updated $\mathbf{r}_n$ to a rotation matrix and substitute it to R_n.
   (c) Repeat above procedure until the objective function converges.
3. Calculate ϕ_l and ψ_m, and update the coefficients w_l.
4. Repeat 2 and 3 until the objective function becomes sufficiently small.

4.2 Pose from Known 2D-3D Correspondences and Motion Constraints

In this method, we add the motion constraint errors to the objective function of eq.2. The objective function becomes as follows:

$$f(R_n, \boldsymbol{t}_n) = \sum_{l=1}^{L} w_l\phi_l^2 + \sum_{m=1}^{M} w'_m\psi_m^2 + g(R_n) \tag{11}$$

where $\psi_m(R_n, \boldsymbol{t}_n)$ is the motion constraint error. ψ is defined from epipolar constraints for the motion of feature points on the consecutive frames.

When 2D coordinates of a feature point on $n-1$ and n-th frame are $\boldsymbol{x}_{n-1}$ and $\boldsymbol{x}_n$ respectively, epipolar lines for $\boldsymbol{x}_{n-1}, \boldsymbol{x}_n$ are represented by the following vectors.

$$\begin{cases} \boldsymbol{n}_1 = \nu_1 E^T \tilde{\boldsymbol{x}}_n & = [a_1, b_1, c_1]^T \\ \boldsymbol{n}_2 = \nu_2 E \tilde{\boldsymbol{x}}_{n-1} & = [a_2, b_2, c_2]^T \end{cases} \tag{12}$$

where E is the essential matrix composed of camera motion parameters on the consecutive frames $(R', \boldsymbol{t}')$, ν_1, ν_2 are the scale parameters set to satisfy $a_k^2 + b_k^2 = 1 (k = 1, 2)$, and $\tilde{\boldsymbol{x}}$ is the homogeneous representation of 2D point $\boldsymbol{x}$, i.e., $\tilde{\boldsymbol{x}} = [x, y, 1]^T$. Squared error ψ^2 is defined as follows.

$$\begin{aligned} \psi^2 &= \left(\boldsymbol{n}_1^T \tilde{\boldsymbol{x}}_{n-1}\right)^2 + \left(\boldsymbol{n}_2^T \tilde{\boldsymbol{x}}_n\right)^2 \\ &= \left(\nu_1^2 + \nu_2^2\right) \left(\tilde{\boldsymbol{x}}_n^T E \tilde{\boldsymbol{x}}_{n-1}\right)^2 \\ &= \left(\nu_1^2 + \nu_2^2\right) \left[\tilde{\boldsymbol{x}}_n^T \left\{\boldsymbol{t}' \times (R' \tilde{\boldsymbol{x}}_{n-1})\right\}\right]^2 \end{aligned} \tag{13}$$

Then, we express the right side of eq.13 by camera pose parameters $R_{n-1}, \boldsymbol{t}_{n-1}, R_n, \boldsymbol{t}_n$ using the following equation which represents the relationship between the motion and the pose on each frame.

$$\begin{cases} R_n &= R' R_{n-1} \\ \boldsymbol{t}_n &= R' \boldsymbol{t}_{n-1} + \boldsymbol{t}' \end{cases} \tag{14}$$

As we know the pose parameter on $(n-1)$-th frame $R_{n-1}, \boldsymbol{t}_{n-1}$, ψ^2 becomes the function of the pose parameter on n-th frame $R_n, \boldsymbol{t}_n$.

Same as the method of 4.1, we represent the rotation component by a quaternion, and estimate the pose by repeating weight coefficients determination and pose parameter estimation.

4.3 Pose from Decomposition of Essential Matrix and 2D-3D Line Correspondences

Essential matrix is composed of rotation and translation components of the camera motion R' and $\boldsymbol{t}'$, and those parameters are obtainable by decomposing essential matrix by using SVD(Hartley and Zisserman, 2000). However, motion parameters cannot be determined uniquely from SVD method. We need to choose rotation components from two rotation matrices R'_1, R'_2 obtained by the decomposition, and also need to determine the scale of translation vector $\boldsymbol{t}'$. We therefore estimate the pose by the following procedure:

1. Decompose the essential matrix estimated from feature point motions and calculate $R'_1, R'_2, \bar{\boldsymbol{t}}'$.
2. Choose R' from R'_1 and R'_2, and determine the scale of $\boldsymbol{t}'$ from 2D-3D line correspondences which are easily obtained from existing 2D-3D line segment correspondences.

In the following discussion, we represent 3D line L by point $\boldsymbol{P}$ on the line and direction $\boldsymbol{D}$. Any point $\boldsymbol{X}$ on line L is represented using coefficient κ as $\boldsymbol{X} = \boldsymbol{P} + \kappa \boldsymbol{D}$. When a 3D line L is corresponding to the 2D line l, there exists the following equation:

$$\begin{cases} \boldsymbol{n}^T R \boldsymbol{D} & = 0 \\ \boldsymbol{n}^T (R\boldsymbol{P} + \boldsymbol{t}) & = 0 \end{cases} \tag{15}$$

From eqs.14 and 15, and representing the translation vector $\boldsymbol{t}'$ as the product of scale parameter α and normalized vector $\bar{\boldsymbol{t}}'$ $(\boldsymbol{t}' = \alpha \bar{\boldsymbol{t}}')$, we obtain the following equations.

$$\boldsymbol{n}^T \left(R' R_{n-1} \boldsymbol{D}\right) = 0 \tag{16}$$

$$\boldsymbol{n}^T \left\{R_n \boldsymbol{P} + \left(R' \boldsymbol{t}_{n-1} + \alpha \bar{\boldsymbol{t}}'\right)\right\} = 0 \tag{17}$$

At first, we choose rotation matrix R' from R'_1 and R'_2. We can regard the value of left side of eq.16 as 2D-3D line correspondence error, and we choose R' which gives less median of absolute error. If 2D-3D line correspondences (l_i, L_i), $(i = 1, 2, ..., M)$ are obtained on the n-th frame and two rotation matrices R'_1 and R'_2 appear, R' is chosen from the following equation.

$$R' = \arg\min_{R'_k} e_k \qquad (k = 1, 2) \tag{18}$$

$$e_k = \operatorname*{median}_i \left|\boldsymbol{n}_i^T \left(R'_k R_{n-1} \boldsymbol{D}_i\right)\right| \quad (i = 1, 2, ..., M)$$

Next, the scale parameter α is obtainable from eq.17 as follows:

$$\alpha = -\frac{\boldsymbol{n}^T \left(R_n \boldsymbol{P} + R' \boldsymbol{t}_{n-1}\right)}{\boldsymbol{n}^T \bar{\boldsymbol{t}}'} \tag{19}$$

This scale parameter is computed from every line correspondences (l_i, L_i) and denoted as α_i, then, we set the scale parameter α as the median of α_i:

$$\alpha = \operatorname*{median}_i \alpha_i \tag{20}$$

By using the median of parameters in the estimation process, we can eliminate the influence of some incorrect correspondences.

4.4 Switching the Two Pose Estimation Methods

On every frame, our tracker checks whether it has sufficient number of 2D-3D correspondences to estimate the pose from 2D-3D correspondences alone. If it has, it estimates the pose by the method of section 4.2. Otherwise, the method of section 4.3 is used for estimation.

If there are the sufficient number of 2D-3D correspondences, the tracker also estimates the pose from 2D-3D correspondence alone using the method of section 4.1. Then, the tracker computes the maximum absolute errors $|\phi'|$ of those two estimated poses, and chooses the pose whose $|\phi'|$ is smaller. By this choice of the pose from two, the tracker can estimate correct pose even when one of the estimated pose becomes quite wrong for various reasons. Especially, on the next frame after the incorrect pose is estimated, the feature motion constraint becomes incorrect because R_{n-1} and $\boldsymbol{t}_{n-1}$ are incorrect, and only 2D-3D correspondences are correct. On such frames, the tracker can obtain the correct pose estimated from 2D-3D correspondences alone.

5 POSE AND CORRESPONDENCES ESTIMATION ON THE INITIAL FRAME

On the initial frame of the tracking, it is necessary to know the camera pose and the 2D-3D line segment correspondence. However, it is difficult to estimate the pose and the correspondences automatically if the tracker doesn't have any prior knowledge about the pose. In this section, we propose a method that estimates camera pose and 2D-3D line segment correspondences simultaneously provided that we know approximate camera pose (initial guess).

5.1 Automatic Camera Pose and 2D-3D Line Correspondence Estimation

This method is based on ICP-algorithm(Besl and McKay, 1992) and integrates robust statistics techniques, and can estimate camera pose and correspondences simultaneously.

Our method is composed of two steps, namely correspondence step and pose step, just like EM algorithm. On the correspondence step, correspondences between 2D line segments $\boldsymbol{l} = \{l_1, l_2, ..., l_i, ..., l_M\}$, and 3D line segments $\boldsymbol{L} = \{L_1, L_2, ..., L_j, ..., L_N\}$ are fixed from the currently estimated camera pose. On the pose step, the pose is reestimated from those correspondences. These two steps repeat until the pose parameter converges.

On the correspondence step, 3D line segments are projected using the currently estimated pose. Then, the 2D line segment corresponding to a 3D line segment is determined as the one whose absolute line segment correspondence error $|\phi|$ defined on eq.6 is the smallest. 2D line segment l'_j corresponding to the 3D line segment L_j is determined as the following equation.

$$l'_j = \arg\min_{l_i} |\phi(l_i, L_j, R, \boldsymbol{t})| \tag{21}$$

On the pose step, camera pose is updated from the correspondences fixed above. We introduce the weighting coefficients again to decrease the bad effects of incorrect correspondences. The objective function used for updating the pose parameter is described as follows.

$$f(\mathbf{r}, \boldsymbol{t}) = \sum_{j=1}^{N} w_j \phi^2(l'_j, L_j, \mathbf{r}, \boldsymbol{t}) + g(\mathbf{r}) \tag{22}$$

Coefficient w_j is determined from the correspondence error $\phi(l'_j, L_j, \mathbf{r}, \boldsymbol{t})$ and eq.7, and it is necessary to estimate standard deviation $\hat{\sigma}$ of line segment correspondence errors. If half or more correspondences are incorrect, we cannot correctly estimate $\hat{\sigma}$ using MAD. We therefore determine $\hat{\sigma}$ not from the distribution of ϕ but from the number of iteration. That is, we use large value of $\hat{\sigma}$ so that every coefficient w_j has almost the same value on the initial few iterations and gradually decrease $\hat{\sigma}$ as the number of iteration increases, and gradually regard the correspondences that have large errors as incorrect and give less effect to pose update process. Overall procedure of our method is described as follows.

1. Set $\hat{\sigma} = \hat{\sigma}_0$.
2. Repeat the following procedure until the pose parameter converges.
   (a) For every 3D line segmentL_j, determine corresponding 2D line segment l'_j from eq.21.
   (b) Set coefficient w_j from eq.7.
   (c) Update pose parameter to the one which decreases the value of objective function $f(\mathbf{r}, \boldsymbol{t})$.
   (d) for next iteration, set $\hat{\sigma} := \gamma\hat{\sigma}$ $(\gamma < 1)$

5.2 Initial Pose and Correspondences Estimation

We applied this method to the initial pose and correspondence estimation problem. As we know the approximate initial guess of the camera pose, we can restrict candidates for the true correspondences on the first. To exclude the correspondences whose projected model line segment is not overlapping to the corresponding straight edge at all, we calculate the distances and the overlapping ratios defined on eq.1 for all possible 2D-3D correspondences and adopt only the correspondences as candidate whose distance is below d_{init} and overlapping ratio is above γ_{init}.

As we restrict the correspondences at first, it is necessary to use criteria different from the one described in eq.21. We use the following criteria on the correspondence step of estimation.

$$l'_j = \arg\min_{l_i \in \boldsymbol{l}'_j} |\phi(l_i, L_j, R, \boldsymbol{t})| \tag{23}$$

Where l'_j is the 2D line segments that are candidates for the one corresponding to the 3D line segment L_j.

After the estimation, it is necessary to choose the true correspondences from ones obtained on the correspondence step. As we have already obtained accurate camera pose, we determine the 2D-3D line segment correspondences same as the correspondence addition process described in section 2.

We applied this method to the pose and correspondences estimation on the initial frame of tracking. The top of fig.3 shows initial guess of the camera pose (bold line) and the straight edges which are the candidate edges of those corresponding to 3D-Model line segments (thin line), and the bottom shows accurately estimated camera pose using our method.

Figure 3: Initial pose and correspondence estimation (Top : Initial guess of the pose and candidates for the corresponding edges, Bottom : Accurately estimated pose and edges corresponding to the line segments of CAD model.

6 EXPERIMENTAL RESULTS

To evaluate our method, we took the motion image sequences of an object in a room (a CRT display) and a corridor scene. We used a conventional USB camera (Creative Webcam Pro eX, image size:320×240 pixels) which was calibrated with a conventional calibration software. We prepared CAD models of CRT display and corridor scene by measuring its 3D contours by hand. Because of the measurement error in the CAD model and intrinsic parameters of the camera, displacements sometimes appear between the target objects on the image and the projected model although we carefully estimated the correct pose by hand.

We have tested on two image sequences which are 1000 frames sequence for the CRT display and 300 frames sequence for the corridor scene, and our tracker could track in those sequences well. Snapshots from tracking result are shown in fig.4.

Our tracker can track even when the camera moves rapidly. An example of such cases is shown in fig.5. Our tracker can also recover to correct pose even if once it estimates a incorrect pose. Fig.6 shows an example of such scene. The pose on the left side is apparently wrong and large displacements appear. However, we could get the pose with less displacement on the next frame as shown in right side.

Figure 4: Projected CAD model using tracking results (Top:CRT display[1000frames sequence], Bottom:corridor scene[300frames sequence]).

One of the reasons for the incorrect pose estimation is the failure of numerical minimization in the pose estimation. As we use nonlinear minimization techniques for the estimation, the parameter sometimes falls into a local minimum and the tracker misestimates the pose. We have tried several nonlinear minimization methods such as proposed by Phong et al.(Phong et al., 1995) and Powell's Dogleg method(Powell, 1970). However, the results were not good enough.

We implemented the tracker with C++ language and the program is not fully optimized. Even with this program, our tracker could track on 15-20 fps with a conventional PC (CPU : Intel Pentium4 Processor 2.2GHz, 1GB memory). We believe that our method can easily track over 30fps by using optimized program and a faster PC.

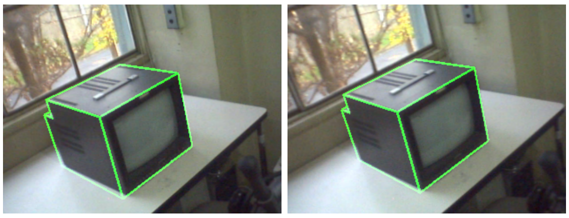

Figure 5: Projected model on the consecutive frames with large camera motion (Distance between the target on each frame are approximately 15-20pixels).

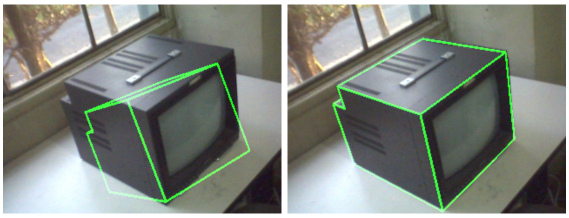

Figure 6: Recovery from incorrect pose (Left : Incorrectly estimated pose, Right : Correctly estimated pose on three frames later from the left image).

7 CONCLUSIONS

In this paper, we proposed a 3D tracking method which integrates the 2D feature tracking. By tracking edges and holding 2D-3D correspondences, our tracker can handle large camera motions and can recover to the correct pose even once the pose estimation fails. Moreover, our tracker estimates the pose from both 2D-3D line segment correspondences and motions of feature points. By fusing those two kinds of information, the tracker can suppress the influence of the incorrect correspondence and can track even when the sufficient number of 2D-3D correspondences are not obtained. We also proposed automatic camera pose and 2D-3D correspondences estimation method and succeeded to estimate the pose and correspondences on the initial frame automatically. From the experiments, we confirmed our tracker can track in real-time with noizy low resolution images taken by a cheap USB camera.

As the future work, we intend to measure the 3D position of feature points appeared during the tracking from their 2D positions and estimated poses on a few frame, then, continue 2D tracking for them and use their 2D-3D correspondences on the latter frame of the 3D tracking.

REFERENCES

Besl, P. J. and McKay, N. D. (1992). A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256.

Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698.

Chiba, N. and Kanade, T. (1998). A Tracker for Broken and Closely Spaced Lines. In *ISPRS '98*, volume XXXII, pages 676 – 683.

Christy, S. and Horaud, R. (1999). Iterative Pose Computation from Line Correspondences. *CVIU*, 73(1):137–144.

Comport, A., Marchand, E., and Chaumette, F. (2003). A real-time tracker for markerless augmented reality. In *ISMAR'03*, pages 36–45.

Drummond, T. and Cipolla, R. (2002). Real-Time Visual Tracking of Complex Structures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):932–946.

G.A.F. Seber, C. (1981). *Nonlinear Regression*, chapter 14. Wiley.

Hartley, R. I. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

L. Vacchetti, V. L. and Fua, P. (2004a). Combining Edge and Texture Information for Real-Time Accurate 3D Camera Tracking. In *ISMAR'04*, pages 48–57.

L. Vacchetti, V. L. and Fua, P. (2004b). Stable Real-Time 3D Tracking Using Online and Offline Information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(10):1385–1391.

Liu, Y., Huang, T. S., and Faugeras, O. D. (1990). Determination of Camera Location from 2-D to 3-D Line and Point Correspondences. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(1):28–37.

Lowe, D. G. (1987). Three-Dimensional Object Recognition from Single Two-Dimensional Images. *Artificial Intelligence*, 31(3):355–395.

Lowe, D. G. (1992). Robust model-based motion tracking through the integration of search and estimation. *IJCV*, 8(2):113–122.

Lucas, B. and Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. In *IJCAI'81*, pages 674–679.

Phong, T., Horaud, R., Yassine, A., and Tao, P. (1995). Object Pose from 2-D to 3-D Point and Line Correspondences. *IJCV*, 15(3):225–243.

Powell, M. J. D. (1970). A Hybrid Method for Non-linear Equations. In Rabinowitz, P., editor, *Numerical Methods for Non-linear Equations*, pages 87–114. Gordon and Breach.

Shi, J. and Tomasi, C. (1994). Good Features to Track. In *CVPR'94*.