STABLE POSE ESTIMATION WITH A MOTION MODEL IN REAL-TIME APPLICATIONS

Po-Chen Wu[1], *Jui-Hsin Lai*[2], *Ja-Ling Wu*[2], *Shao-Yi Chien*[1]

[1]Media IC and System Lab
Graduate Institute of Electronics Engineering and Dept. of Electrical Engineering
[2]Communications and Multimedia Lab
Graduate Institute of Networking and Multimedia and Dept. of Computer Science and Information Eng.
National Taiwan University
pcwu@media.ee.ntu.edu.tw, {larrylai, wjl}@cmlab.csie.ntu.edu.tw, sychien@cc.ee.ntu.edu.tw

ABSTRACT

Object pose estimation from a camera is a well-developing subject in computer vision. In theory, the pose can be uniquely determined from a calibrated camera. However, in practice, most of the real-time pose estimation algorithms suffer from pose ambiguity due to low accuracy of the target object. We think that pose ambiguity—two distinct local minima of the according error function—exist because of the phenomenon of geometric illusion. Both of the ambiguous poses are plausible. After obtaining the solution of two minima (pose candidates), we develop a real-time algorithm for stable pose estimation of target objects using a motion model. In the experimental results, the proposed algorithm effectively diminishes the effect of pose jumping and pose jittering. To the best of our knowledge, this is the first work conducted for solving the pose ambiguity problem with a motion model in real-time applications.

Index Terms— Pose estimation, pose ambiguity, pose stabilization.

1. INTRODUCTION

The objective for determining pose estimation is to calculate the position and orientation of a target object from a calibrated camera. Augmented reality (AR) [1], wherein synthetic objects are inserted into a real scene in real-time, is a prime candidate system for pose estimation. After obtaining the pose computed with some geometric information, the system could render computer-generated images (CGI) according to the pose on the display. ARToolkit [2], for example, is a widely used system for use with AR applications. The target object in AR systems is usually the planar fiducial marker, which is frequently used for navigation and localization.

The information available for solving the pose estimation problem is usually a set of point correspondences. They are composed of a 3D reference point expressed in object coordinates and its 2D projection expressed in image coordinates. Using the object-space collinearity error, Lu et al. [3] derived an iterative algorithm that directly computes orthogonal rotation matrices. Instead of using the iterative algorithm, Ansar et al. [4] developed a framework that provided a set of linear solutions to the pose estimation problem; the algorithm is applicable for both points and lines. These online pose estimation works determined a unique pose for each frame without taking the pose ambiguity problem into consideration.

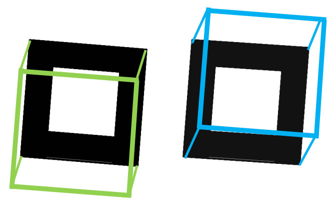


Fig. 1. Illustration of pose ambiguity. It is a geometric illusion: There appears to be more than one 3D geometrical explanations obtained from the same perspective projected marker on the image plane.

Pose ambiguity, as shown in Fig. 1, is the main cause of pose jumping. The obtained pose would be one of the random ambiguous poses in each frame, and this causes pose jumping. From our experiences, several state-of-the-art pose algorithms suffer from pose jumping. These pose ambiguity problems have been discussed by previous works [5], [6]. Oberkampf et al. [5] give a straightforward interpretation for the case of orthographic projection. They developed their algorithm for planar targets, which uses scaled orthographic projection at each iteration step. Schweighofer et al. [6] extended to tackle the general case of perspective projection and developed an algorithm for obtaining a unique solution to pose estimation. However, even with these algorithms, the problem of pose jumping still persists occasionally. Besides, the problem of pose jittering also bothers users due to the noisy images.

In order to reduce the effects of pose jumping, we propose an algorithm with a motion model to obtain the pose of the target object. The motion model will get updated through the Kalman filter [7]. The Kalman filter provides an efficient computational means for estimating the true poses by computing a weighted average of the measured pose and predicted pose from the motion model. From our observation, one of the two ambiguous poses with distinct local minima of error function is the correct pose. Therefore, every time, after obtaining the two ambiguous poses, the pose that is more similar to the predicted pose is chosen. If the predicted pose is realistic, then it is almost ensured that the chosen pose is the correct one.

The main contributions of this work are as follows:

1. We can solve the problem of pose jumping effectively because we obtain the proper pose from two ambiguous poses using the motion model.

2. The effects of pose jittering will be reduced because of the use of the Kalman filter. We can estimate the pose that tends to be closer to the true pose than the measured pose. The sequences of the estimated poses would be also much smoother because the poses are much more consistent with their previous ones.

3. This is the first work on pose estimation combined with a motion model. Even if the target object is undetected in some frames of long sequences, we can just use the predicted pose using the motion model as the final pose to prevent discontinuity in the sequence of poses.

The remainder of this article is organized as follows. First, we describe the formulation of the pose estimation problem in detail in Sec. 2. Then, we interpret the pose ambibuity and show how to develop the two poses with local minima of the according error function in Sec. 3. In Sec. 4, we describe the details of our stable pose estimation algorithm. In Sec. 5, we show the results of the proposed pose estimation algorithm and compare its performance with other competitive pose algorithms. Conclusions are drawn in Sec. 6.

2. PROBLEM FORMULATION

The main problem of camera pose estimation is to find out the six degrees of freedom, which are parameterized by the orientation and position of the target object with respect to a calibrated camera (with known interior parameters), as shown in Fig. 2. Given a set of noncollinear 3D coordinates of reference points $\mathbf{p}_i = (x_i, y_i, z_i)^t, i = 1, ..., n, n \geq 3$ expressed in object-space coordinates and a set of camera-space coordinates $\mathbf{q}_i = (x'_i, y'_i, z'_i)^t$, the transformation between them can be formulated as

$$\mathbf{q}_i = R\mathbf{p}_i + \mathbf{t}, \tag{1}$$

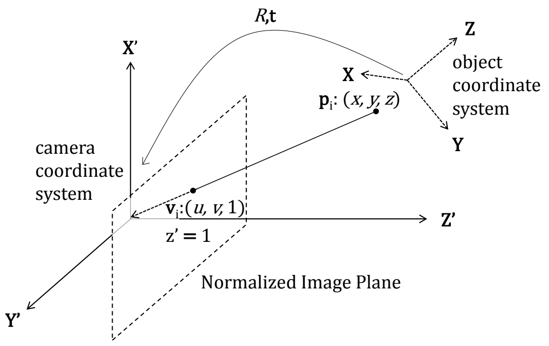


Fig. 2. Coordinate systems between camera and target objects in the pose estimation problem.

where

$$R = \begin{pmatrix} \mathbf{r}_1^t \\ \mathbf{r}_2^t \\ \mathbf{r}_3^t \end{pmatrix} \in SO(3) \quad \text{and} \quad \mathbf{t} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \in R(3) \tag{2}$$

are a rotation matrix and a translation vector, respectively.

We introduce the *normalized image plane located* at $z' = 1$ as the camera reference frame. In such a normalized image plane, we define the image point $\mathbf{v}_i = (u_i, v_i, 1)^t$ to be the projection of $\mathbf{p}_i$ on it. In the idealized pinhole camera model, $\mathbf{v}_i$, $\mathbf{q}_i$, and the center of projection are collinear. We can express this relationship by the following equation:

$$u_i = \frac{\mathbf{r}_1^t \mathbf{p}_i + t_x}{\mathbf{r}_3^t \mathbf{p}_i + t_z}, \quad v_i = \frac{\mathbf{r}_2^t \mathbf{p}_i + t_y}{\mathbf{r}_3^t \mathbf{p}_i + t_z}. \tag{3}$$

Given the observed image points $\hat{\mathbf{v}}_i = (\hat{u}_i, \hat{v}_i, 1)^t$, the pose estimation algorithm has to determine values for R and $\mathbf{t}$ that minimize an according error function. In our work, we use the *object-space error*, as used by [3] and [6],

$$E_{os}(R, \mathbf{t}) = \sum_{i=1}^{n} \left\| (I - \hat{V}_i)(R\mathbf{p}_i + \mathbf{t}) \right\|^2, \quad \hat{V}_i = \frac{\hat{\mathbf{v}}_i \hat{\mathbf{v}}_i^t}{\hat{\mathbf{v}}_i^t \hat{\mathbf{v}}_i}. \tag{4}$$

3. POSE AMBIGUITY INTERPRETATION

Pose ambiguity denotes situations where the error function have several local minima for a given configuration. The cause of pose ambiguity is the low accuracy of the reference points extraction; low accuracy is almost inevitable in general cases. Fig. 1. shows the illustration of pose ambiguity.

Most of recent pose estimation algorithms working in real time suffer from pose ambiguity. Schweighofer et al. [6] found that in the case where coplanar points $\mathbf{p}_i = (p_{i_x}, p_{i_y}, 0)$, viewed by a perspective camera, the algorithm typically delivers two distinct minima according to E_{is} and E_{os}. We derive the two poses with the minima of E_{os} by the method mentioned in [6].

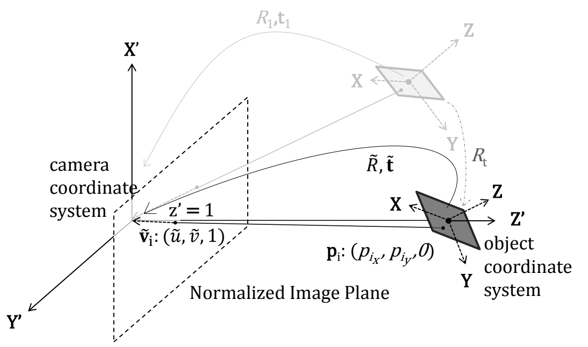

R_1,$\mathbf{t}_1$
X'
Z
X
Y
R_t
camera coordinate system
$\tilde{R}, \tilde{\mathbf{t}}$
Z
X
Z'
z' = 1
$\tilde{\mathbf{v}}_i$: $(\tilde{u}, \tilde{v}, 1)$
$\mathbf{p}_i$: $(p_{ix}, p_{iy}, 0)$
object coordinate system
Y
Normalized Image Plane
Y'

Fig. 3. Transformed coordinate system.

3.1. Derivation of Poses With Local Minima

We begin with a known pose $(R_1, \mathbf{t}_1)$ obtained from any pose estimation algorithm; the iterative algorithm proposed by [3] had been used in our work. Then, use this first guess of the pose to estimate a second pose, which also has a minimum of E_{os}.

Assume reference points $\mathbf{p}_i$, which are measured in the image as $\hat{\mathbf{v}}_i$ such that

$$\hat{\mathbf{v}}_i \approx \mathbf{v}_i \propto R_1\mathbf{p}_i + \mathbf{t}_1. \tag{5}$$

Multiply both sides of (5) with R_t to get a transformed system such that $R_t\mathbf{t}_1 = [0\ 0\ \|\mathbf{t}_1\|]^t$ (see Fig. 3). Let

$$\tilde{\mathbf{v}}_i = R_t\hat{\mathbf{v}}_i,\ \ \tilde{R} = R_tR_1,\ \ \tilde{\mathbf{t}} = R_t\mathbf{t}_1, \tag{6}$$

and the pose $(\tilde{R}, \tilde{\mathbf{t}})$ minimizes

$$E_{os}(\tilde{R}, \tilde{\mathbf{t}}) = \sum_{i=1}^{n} \left\|(I - \tilde{V}_i)(\tilde{R}\mathbf{p}_i + \tilde{\mathbf{t}})\right\|^2. \tag{7}$$

Here, we introduce a rotation matrix $\tilde{R}_z$ to let (7) be

$$E_{os}(\tilde{R}, \tilde{\mathbf{t}}) = \sum_{i=1}^{n} \left\|(I - \tilde{V}_i)(\tilde{R}\underbrace{\tilde{R}_z\tilde{R}_z^{-1}}_{I}\mathbf{p}_i + \tilde{\mathbf{t}})\right\|^2, \tag{8}$$

where the rotation matrix $\tilde{R}_z^{-1}$ rotates the planar model $\mathbf{p}_i$ only about its z-axis. The rotation matrix $\tilde{R}\tilde{R}_z$ can be broken down into the product of three rotation matrices $\tilde{R}\tilde{R}_z = R_z(\tilde{\gamma}_1)R_y(\tilde{\beta}_1)R_x(\tilde{\alpha}_1)$, where $R_i(\phi)$ describes a rotation of ϕ degrees about axis i. By selecting $\tilde{R}_z$ such that $\tilde{\alpha}_1 = 0$, we obtain another transformed system

$$\tilde{\mathbf{v}}_i \approx R_z(\tilde{\gamma})R_y(\tilde{\beta})\tilde{\mathbf{p}}_i + \tilde{\mathbf{t}} \tag{9}$$

with the corresponding error function

$$E_{os}(\tilde{\beta}, \tilde{\gamma}, \tilde{\mathbf{t}}) = \sum_{i=1}^{n} \left\|(I - \tilde{V}_i)(R_z(\tilde{\gamma})R_y(\tilde{\beta})\tilde{\mathbf{p}}_i + \tilde{\mathbf{t}})\right\|^2. \tag{10}$$

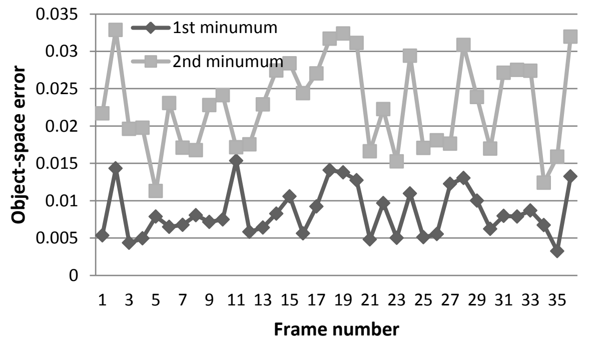


Fig. 4. Object-space errors E_{os} from a sequence video with a planar target. The pose that has the lower error E_{os} (the dark plot) is the final pose in each frame.

Because $\tilde{\mathbf{t}} = [0\ 0\ \|\mathbf{t}_1\|]^t$, known from (6), we can rewrite (9) as

$$\tilde{\mathbf{v}}_i \approx R_z(\tilde{\gamma})(R_y(\tilde{\beta})\tilde{\mathbf{p}}_i + \tilde{\mathbf{t}}) \tag{11}$$

because $R_z(\tilde{\gamma})\tilde{\mathbf{t}} = \tilde{\mathbf{t}}_1$. Thus, $R_z(\tilde{\gamma})$ is a rotation just around the optical axis (z-axis) of the camera. This rotation leaves the geometric relation between the image plane and model plane invariant and just affects the image coordinates. Thus, we can just fix $\tilde{\gamma} = \tilde{\gamma}_1$ and search for the local minima of E_{os} with respect to $\tilde{\beta}$ [3], [6].

4. STABLE POSE ESTIMATION ALGORITHM

After obtaining the poses with local minima, some previous work determined the final pose with the lowest error E_{os} [6], as shown in Fig. 4. Unfortunately, the previous work still suffers from pose ambiguity even when choosing the optimal solution for E_{os}. In fact, the correct pose $\hat{P}$ is not consistent with the pose with the lowest error. From our experimental evidence, we deemed the second pose to be the correct one when pose jumping occurs. The results shown in Fig. 5 is consistent with our assumption: The two poses with local minima sometimes interchange and one of the them is correct. Based on these observations, we develop our *Stable Pose Estimation Algorithm*: In each time step, the system generates a predicted pose $\tilde{P}$ according to a motion model. This motion model simulates the orientation of the pose in real conditions and is updated through the Kalman filter in each time step. From two candidates, the pose that is more similar to $\tilde{P}$ is chosen as the correct pose $\hat{P}$. The final pose is the weighted average of the predicted pose $\tilde{P}$ and measured pose $\hat{P}$.

4.1. Motion Model

Assuming that the motion model of the pose rotation about three axes—X, Y, and Z—is identical, we just discuss the case of rotation about the X-axis in the remainder of this paper. The cases of rotation about the Y-axis and Z-axis are all the same.

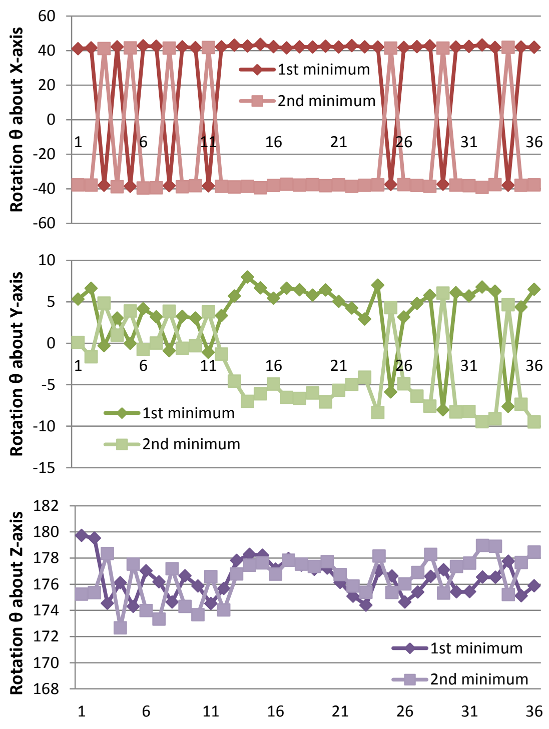


Fig. 5. Rotation angle about X-axis, Y-axis, and Z-axis of the poses with minimum error E_{os}. The value will drastically change during some frames.

To estimate the following rotation angle with a motion model, the motion model should maintain the current angle and angular velocity. The angle and angular velocity are described by the linear state space $\mathbf{x}_k = [x\ \dot{x}]^t$, where $\dot{x}$ is the angular velocity. Assume that between the $(k-1)$ and k time step the system undergoes a constant angular acceleration of a_k, which is normally distributed with mean 0 and deviation σ_a, for Δt seconds. From Newton's laws of motion we conclude that

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{G}a_k, \tag{12}$$

where

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{G} = \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}. \tag{13}$$

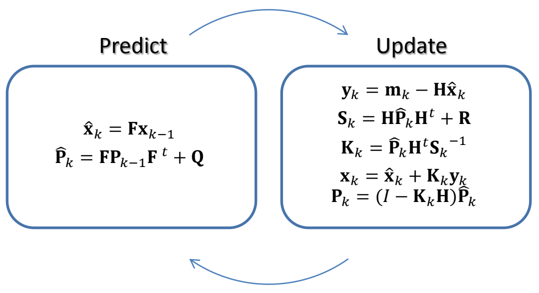
We rewrite (12) in another form

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{w}_k, \tag{14}$$

where

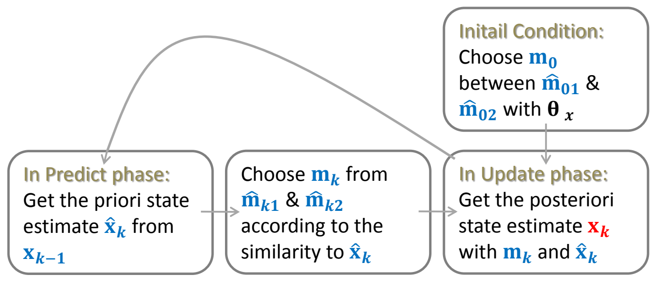
$$\mathbf{w}_k \sim N(0, \mathbf{Q}) \quad \text{and} \quad \mathbf{Q} = \mathbf{G}\mathbf{G}^t\sigma_a^2 = \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \sigma_a^2. \tag{15}$$

For each time step, we obtain measurements of the rotation angle about the X-axis. Let us assume that the measurement noise $\mathbf{v}_k$ is also normally distributed with mean 0 and standard deviation σ_z:



Predict
Update
$\hat{\mathbf{x}}_k = \mathbf{F}\mathbf{x}_{k-1}$
$\hat{\mathbf{P}}_k = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^t + \mathbf{Q}$
$\mathbf{y}_k = \mathbf{m}_k - \mathbf{H}\hat{\mathbf{x}}_k$
$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_k\mathbf{H}^t + \mathbf{R}$
$\mathbf{K}_k = \hat{\mathbf{P}}_k\mathbf{H}^t\mathbf{S}_k^{-1}$
$\mathbf{x}_k = \hat{\mathbf{x}}_k + \mathbf{K}_k\mathbf{y}_k$
$\mathbf{P}_k = (I - \mathbf{K}_k\mathbf{H})\hat{\mathbf{P}}_k$

Fig. 6. Detail operation of two phases, "Predict" and "Update," of the Kalman filter.



Initail Condition:
Choose $\mathbf{m}_0$ between $\hat{\mathbf{m}}_{01}$ & $\hat{\mathbf{m}}_{02}$ with θ_x
In Predict phase:
Get the priori state estimate $\hat{\mathbf{x}}_k$ from $\mathbf{x}_{k-1}$
Choose $\mathbf{m}_k$ from $\hat{\mathbf{m}}_{k1}$ & $\hat{\mathbf{m}}_{k2}$ according to the similarity to $\hat{\mathbf{x}}_k$
In Update phase:
Get the posteriori state estimate $\mathbf{x}_k$ with $\mathbf{m}_k$ and $\hat{\mathbf{x}}_k$

Fig. 7. System flow of stable pose estimation algorithm.

$$\mathbf{m}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k, \tag{16}$$

where

$$\mathbf{H} = [1\ 0] \quad \text{and} \quad \mathbf{v}_k \sim N(0, \mathbf{R}),\ \mathbf{R} = \underline{\mathbf{v}_k\mathbf{v}_k^t} = \sigma_z^2. \tag{17}$$

4.2. Predict and Update through Kalman Filter

The operations of two phases of the Kalman filter, "Predict" and "Update," are shown in Fig. 6. Because of pose ambiguity, two measurements, $\hat{\mathbf{m}}_{k1}$ and $\hat{\mathbf{m}}_{k2}$, of the pose are obtained in a real condition at each time step. Assuming that the priori state estimate $\hat{\mathbf{x}}_k$ is very authentic, the measurement that is more consistent with $\hat{\mathbf{x}}_k$ is regarded as the only measurement $\mathbf{m}_k$. After the operations of the Kalman filter, a new posteriori state estimate $\mathbf{x}_k$ is obtained, which can be used in the next recursion.

To guarantee that the state estimate is reliable at each time step, we have to ensure that the state estimate is authentic at the beginning. From our experiences, the planar target almost faces upward under any initial condition, implying that the rotation angle of the X-axis θ_x is larger than 0, as shown in Fig. 5. Based on this assumption, we choose the first measurement $\mathbf{m}_0$ with a larger θ_x from two candidates. If the state estimate $\mathbf{x}_k$ at every time step is similar to its previous one (this is one distinguishing feature of the Kalman filter), then all of them are reliable to some degree.

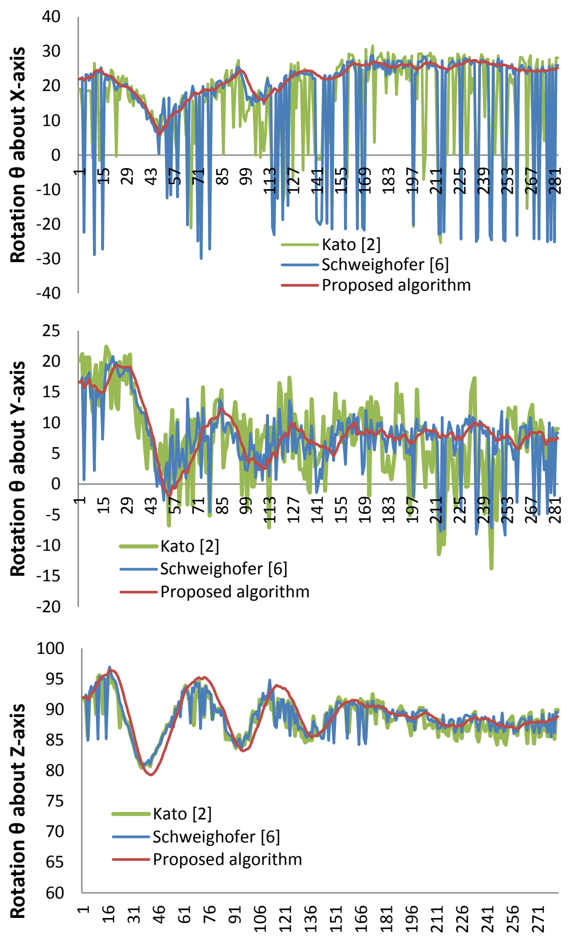


Fig. 9. Comparison of the rotation angle about the X-axis, Y-axis, and Z-axis of the poses.

Fig. 7 shows the processing flow of the proposed stable pose estimation algorithm. Finally, we use the first element of $\mathbf{x}_k$ as the output value of the pose estimation instead.

5. EXPERIMENTAL RESULTS

In this section, we will discuss the setting of parameters and show the results of pose estimation. Some video sequences of markers with random rotation angles from the camera are used as the test data. According to the marker pattern in the database provided by [8], we determined the set of point correspondences between the object space and image plane as $\mathbf{p}_i$ and $\hat{\mathbf{v}}_i$ in Sec.2. Then, we calculated the pose of the planar marker from the camera with the set of point correspondences by the proposed algorithm and other state-of-the-art algorithms.

5.1. Parameter Settings

The state estimate and estimate covariance matrix, $\mathbf{x}_0$ and $\mathbf{P}_0$, respectively, were initialized as follows:

$$\mathbf{x}_0 = \begin{bmatrix} \mathbf{m}_0 \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{P}_0 \begin{bmatrix} L & 0 \\ 0 & L \end{bmatrix}, \tag{18}$$

where L is a value determined by the variance of the initial state. A larger L means that the initial state estimate is very unreliable and the true value tends to be closer to the measurement values. Here, we set $L = 10$ in our initial condition.

The other parameters of the motion model in Sec.4.1 to be determined are the deviation in the motion acceleration σ_a and deviation in the measurement noise σ_z. A large σ_a means that the model has a drastic acceleration motion, and a small σ_z means that the measurements are very trustworthy. These two parameters are chosen empirically, with $\sigma_a^2 = 1000$ and $\sigma_z^2 = 2$. The experimental results below are generated by these parameters.

5.2. Pose Estimation Result Comparison

We recorded some video sequences of a marker that is at a random rotation angle from the camera. Fig. 8 shows the pose estimation results, which are compared with the results obtained using state-of-the-art algorithms. In every condition and every time step, our algorithm provides a solution for real-time pose estimation with high stability. The first row in Fig. 8 is the original continuous video sequence with a fiducial marker. The second and third raws are the pose estimation results with CGIs of Kato et al. [2] and Schweighofer et al. [6]. The last row shows the results obtained by our proposed algorithm. Even with low resolution and noisy images, pose sequences without pose jumping was derived, and it shows a marked difference from the other sequences obtained from other algorithms.

Fig. 9 shows the rotation angle of the marker with respect to a camera. When the pose jumps during the video sequences, the rotation angle would vary drastically. The most obvious example is the first chart in Fig. 9. Using the proposed algorithm, the situation of pose jumping can almost be avoided.

Furthermore, the pose would be much more stable by using a motion model. People would feel more comfortable if the difference in values of the rotation angles between two continuous frames about each axis are as small as possible. Moreover, pose jittering implies that the difference in values during video sequences are unsettled and sway around 0. Fig. 9 depicts the pose sequences derived by our algorithm are much more stable with a smaller difference between frames. We have also applied some temporal filters to the other two methods trying to diminish the effects of pose jittering, but the final pose would be badly affected by the ambiguous poses nearby.

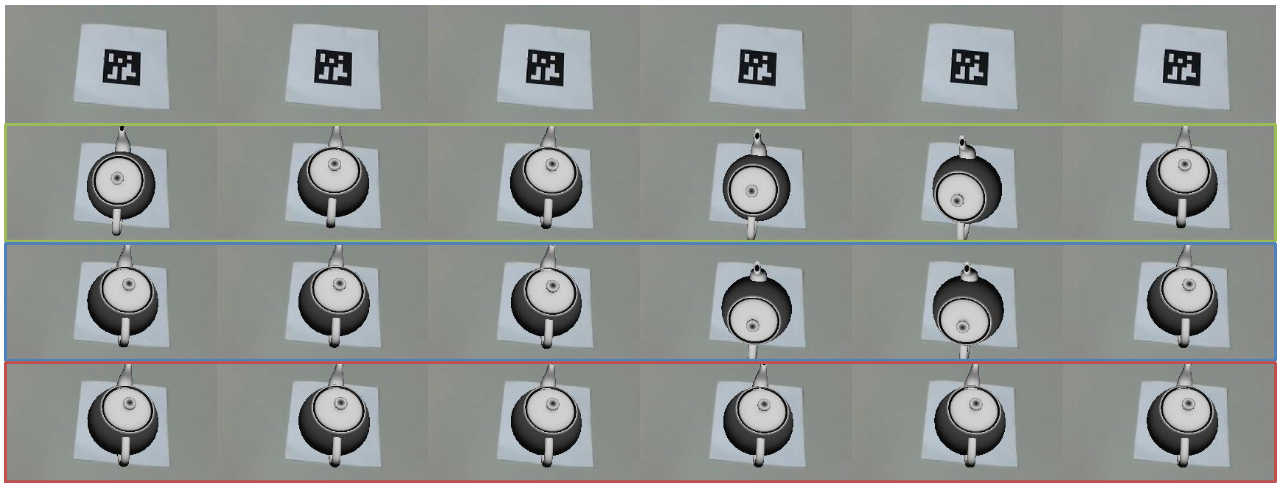


Fig. 8. Pose estimation result comparison. The first row is the continuous raw image sequence with a marker. The second and third rows are the results obtained by other algorithms, and the forth row are the results obtained by our proposed algorithm.

6. CONCLUSION

In this work, we propose a stable pose estimation algorithm for real-time applications. The proposed concept of motion modeling cannot be only used with the proposed algorithm, but with other pose estimation algorithms as well. Hence, the effect of pose jittering can be diminished drastically. Even the correct pose from two candidate ambiguous poses can be predicted with the motion model, so the problem of pose jumping can be solved effectively.

To the best of our knowledge, this is the first work combining a pose estimation algorithm with a motion model. Because several applications of pose estimation are processed in video format, we cannot estimate the pose by considering information from just one frame. With the implementation of the Kalman filter, the derived pose in each time step would be more consistent with the previous one. In addition, users of these applications will feel more comfortable with the much smoother pose sequences.

Acknowledgment

This work was supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC99-2911-I-002-201, 99R70600, and 10R80800.

7. REFERENCES

[1] Ronald Azuma, "A survey of augmented reality," *Presence: Teleoperators and Virtual Environments*, vol. 7, pp. 355–385, 1997.

[2] H. Kato and M. Billinghurst, "Marker tracking and hmd calibration for a video-based augmented reality conferencing system," in *Proceedings of 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR '99)*, 1999, pp. 85–94.

[3] C.-P. Lu, G.D. Hager, and E. Mjolsness, "Fast and globally convergent pose estimation from video images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 610–622, Jun. 2000.

[4] A. Ansar and K. Daniilidis, "Linear pose estimation from points or lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 578 – 589, May 2003.

[5] Denis Oberkampf, Daniel F. DeMenthon, and Larry S. Davis, "Iterative pose estimation using coplanar feature points," *Computer Vision and Image Understanding*, vol. 63, no. 3, pp. 495 – 511, 1996.

[6] G. Schweighofer and A. Pinz, "Robust pose estimation from a planar target," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2024–2030, Dec. 2006.

[7] Greg Welch and Gary Bishop, "An introduction to the kalman filter," 1995, Technical Report TR 95-041, University of North Carolina, Department of Computer Science.

[8] Schmalstieg D. Wagner, D., "Artoolkitplus for pose tracking on mobile devices," in *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*, 2007, pp. 139–146.