2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

A Moment-based 3D Object Tracking Algorithm for High-speed Vision

Takashi Komuro and Masatoshi Ishikawa

Abstract— **In this paper we propose a method of realizing continuous tracking of a three-dimensional object by calculating moments of a translating and rotating object whose shape is known, either analytically or by using a table, and matching them with those of the input image. In simulation, the position and orientation is accurately recognized. Using a noise model and particle filter, we show that the position and orientation can be recognized even with noisy images.**

I. INTRODUCTION

In order for a robot hand to catch a rapidly moving object and dexterously manipulate a tool, it is effective to acquire visual information at a frame rate as high as 1000 fps[1], [2]. Manipulation of an object requires obtaining the position and orientation of the object. However, it is generally difficult to calculate the position and orientation of an object in real time from visual information obtained at such rates. To calculate the position and orientation of an object which has a known shape, the following methods are commonly used:

- Calculating from features such as vertices and lines
- Matching with model figures
- Pattern recognition

The method of calculating the position and orientation from features such as vertices and lines[3] has the drawback that it does not work well for noisy images. Images captured at high frame rates are generally dark due to short exposure time, and the effect of noise therefore becomes larger.

Matching with an image of a model object that is virtually translated and rotated[4] is relatively robust to noise; however, it takes much time to generate model images and high-speed execution is thus difficult.

In the pattern recognition approach, it is easy to match with model images prepared in advance by expressing an image as, for example, a vector in which each pixel value is an element, and mapping the vector to a point in eigenspace[5]. This method, however, is not robust to geometric changes such as rotation and scaling of the target because it involves statistical processing on pixel values.

On the other hand, by using the moments of a silhouette image as feature values, it is possible to achieve recognition reflecting the geometric properties of the target, and this method is expected to be robust to noise. Fig. 1 shows continuous tracking of a target by generating a window around the centroid of the target in the previous frame and making the centroid of the image in the window the centoid of the target in the current frame. The recognized centroid of the target is marked with a bright point. You can see from the figure that tracking is realized even with very noisy images.

T. Komuro is with the Department of Information Physics and Computing, Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan Takashi_Komuro@ipc.i.u-tokyo.ac.jp

M. Ishikawa is with the Department of Creative Informatics, Graduate School of Information Science and Technology, The University of Tokyo

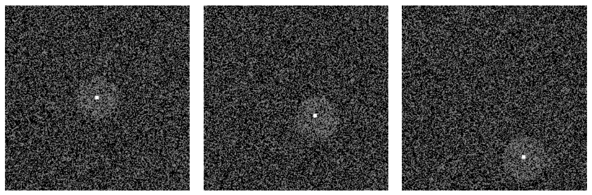

Fig. 1. Example of centroid-based tracking.

The centroid of the image $I(x, y)$ is calculated from the moments, as shown below; here, $I(x, y)$ is a silhouette image having a value of 1 in the target region and 0 elsewhere:

$$(C_x, C_y) = (m_{1,0}/m_{0,0}, m_{0,1}/m_{0,0}) \tag{1}$$

The general form of the moment is expressed as:

$$m_{p,q} = \int\int x^p y^q I(x, y)dxdy \tag{2}$$

Because the calculation of moments is regular for the x and y coordinates and it is easy to parallelize it in hardware, it is suited for high-speed processing.

By using not only the centroid but also 2nd order moments and above, it is possible to detect the planar angle of the target and to use invariants to rotation and scaling for tracking. However, this is valid only for movement in a plane and cannot be applied to three-dimensional movement. Though there are some methods that can deal with three-dimensional movement by performing coordinate transformation in the moment calculation[6], there is a restriction that the target has to be a figure in the same plane.

In this paper we propose a method of realizing continuous tracking of a three-dimensional object by calculating, either analytically or by using a table, moments of a translating and rotating object whose shape is known and matching the moments with those of the input image.

II. 3D TRACKING USING NEWTON'S METHOD

A. Algorithm

We introduce $\mathbf{x} = [x\ y\ z\ \theta_x\ \theta_y\ \theta_z]^{\mathbf{T}}$ as the position and orientation of the target object and $\mathbf{m} = [m_{0,0}\ m_{1,0}\ m_{0,1}\ m_{2,0}\ \cdots\ ]^{\mathbf{T}}$ as the observed moments. We define $\tilde{\mathbf{x}}$ as the estimation of $\mathbf{x}$, and the value that the moment takes is $\tilde{\mathbf{m}}(\tilde{\mathbf{x}})$. The difference between $\mathbf{x}$ and $\tilde{\mathbf{x}}$ is $\Delta\tilde{\mathbf{x}}$, and the difference between $\mathbf{m}$ and $\tilde{\mathbf{m}}(\tilde{\mathbf{x}})$ is $\Delta\tilde{\mathbf{m}}(\tilde{\mathbf{x}})$. We obtain

$$\begin{aligned} \mathbf{m} &= \tilde{\mathbf{m}}(\tilde{\mathbf{x}}) + \Delta\tilde{\mathbf{m}}(\tilde{\mathbf{x}}) & (3) \\ &\approx \tilde{\mathbf{m}}(\tilde{\mathbf{x}}) + \frac{\partial\tilde{\mathbf{m}}(\tilde{\mathbf{x}})}{\partial\tilde{\mathbf{x}}}\Delta\tilde{\mathbf{x}} & (4) \end{aligned}$$

$$\Delta\tilde{\mathbf{x}} \approx \left(\frac{\partial\tilde{\mathbf{m}}(\tilde{\mathbf{x}})}{\partial\tilde{\mathbf{x}}}\right)^{+} (\mathbf{m} - \tilde{\mathbf{m}}(\tilde{\mathbf{x}}))\,. \quad (5)$$

Here A^{+} is a pseudo inverse matrix of A.

Therefore, x is calculated iteratively as follows:

$$\mathbf{x_k} = \tilde{\mathbf{x}}_{\mathbf{k-1}} + \Delta\tilde{\mathbf{x}}_{\mathbf{k}} \quad (6)$$

$$\Delta\tilde{\mathbf{x}}_{\mathbf{k}} = \left(\frac{\partial\tilde{\mathbf{m}}(\tilde{\mathbf{x}}_{\mathbf{k-1}})}{\partial\tilde{\mathbf{x}}_{\mathbf{k-1}}}\right)^{+} (\mathbf{m} - \tilde{\mathbf{m}}(\tilde{\mathbf{x}}_{\mathbf{k-1}}))\,. \quad (7)$$

Here, $\tilde{\mathbf{m}}(\tilde{\mathbf{x}})$ is calculated in either way below according to the category the target object belongs to.

1) Sphere and Spheroid: Under weak perspective projection, the projection image of a ball is always a circle, regardless of the object's rotation. When the three-dimensional coordinates of the center of the ball in the camera coordinate system are (X, Y, Z), the equation of the projected circle is

$$\left(x - f\frac{X}{Z}\right)^2 + \left(y - f\frac{Y}{Z}\right)^2 = f^2\frac{R^2}{Z^2}, \quad (8)$$

where R is the radius of the ball and f is the focal length of the camera lens. This is identical to a circle with its center at the origin and a radius of 1, scaled by fR/Z vertically and horizontally and shifted by $(fX/Z, fY/Z)$ in the image plane. Therefore, the moments of the circle image are calculated in advance and the effects of scaling and shifting are corrected for.

The effect of scaling on the moments is corrected for as follows. When $x' = ax, y' = by$ and $I'(x', y') = I(x, y)$, the moments of $I'(x', y')$ are

$$\begin{aligned} m'_{p,q} &= \int\int x'^p y'^q I'(x', y')dx'dy' & (9) \\ &= \int\int (ax)^p (by)^q I(x, y) ab dx dy & (10) \\ &= a^{p+1}b^{q+1}\int\int x^p y^q I(x, y)dxdy & (11) \\ &= a^{p+1}b^{q+1}m_{p,q}. & (12) \end{aligned}$$

The effect of shifting on the moments is corrected as follows. When $x' = x - x_s, y' = y - y_s$ and $I'(x', y') = I(x, y)$, the moments of $I'(x', y')$ are

$$m'_{p,q} = \int\int (x - x_s)^p (y - y_s)^q I(x, y)dxdy \quad (13)$$

For example, when $p = 1$ and $q = 0$, the moment is calculated by:

$$\begin{aligned} m'_{1,0} &= \int\int (x - x_s) I(x, y)dxdy & (14) \\ &= m_{1,0} - x_s m_{0,0}. & (15) \end{aligned}$$

In the general case, it is calculated using Equ. (31) shown later.

If the object is a spheroid, the projection image is always an ellipse under weak perspective projection. Consider a spheroid whose center is at the origin and whose rotation axis is the x axis. When it is rotated by θ_y around the y axis and θ_z around the z axis and is translated by (X, Y, Z), the projection image is expressed by the following equations[7], where R_a and R_b are the long axis and the short axis, respectively.

$$\begin{aligned} \frac{(x' - fX/Z)^2}{(fR'_a/Z)^2} + \frac{(y' - fY/Z)^2}{(fR_b/Z)^2} &= 1 & (16) \\ x' &= x\cos\theta_z - y\sin\theta_z & (17) \\ y' &= x\sin\theta_z + y\cos\theta_z & (18) \\ R'^2_a &= R'^2_a\cos^2\theta_y + R'^2_b\sin^2\theta_y & (19) \end{aligned}$$

This is identical to an ellipse formed from a circle with its center at the origin and a radius of 1, scaled by fRa'/Z vertically and fRb/Z horizontally, rotated by θ_z, and shifted by $(fX/Z, fY/Z)$ in the image plane.

The effect of rotation on the moments is corrected for as shown below. When $x' = x\cos\theta - y\sin\theta$, $y' = x\sin\theta + y\cos\theta$, and $I'(x', y') = I(x, y)$, the moments of $I'(x', y')$ are

$$\begin{aligned} m'_{p,q} = \int\int &(x\cos\theta - y\sin\theta)^p \\ &(x\sin\theta + y\cos\theta)^q I(x, y)dxdy \quad (20) \end{aligned}$$

For example, when $p = 1$ and $q = 0$, the moment is calculated by:

$$\begin{aligned} m'_{1,0} &= \int\int (x\cos\theta - y\sin\theta) I(x, y)dxdy & (21) \\ &= m_{1,0}\cos\theta - m_{0,1}\sin\theta & (22) \end{aligned}$$

In the general case, it is calculated using Equ. (31) shown later.

2) Coplanar Figure and Convex Polyhedron: When a point on a three dimensional object $\mathbf{X} = (X, Y, Z)$ is moved to $\mathbf{X}' = (X', Y', Z')$ along with the object's rotation and translation, $\mathbf{X}'$ is described as:

$$\mathbf{X}' = R\mathbf{X} + \mathbf{t} \quad (23)$$

Here, R is the rotation matrix and $\mathbf{t}$ is the translation vector. In the camera coordinate system, the object is on the plane $Z = Z_c$.

Assuming weak perspective projection, the projection image of the object after rotation and translation (x', y') is

$$x' = f\frac{X'}{Z_c'} = \frac{r_{11}X + r_{12}Y + r_{13}Zc + t_x}{Z_c'} \quad (24)$$

$$y' = f\frac{Y'}{Z_c'} = \frac{r_{21}X + r_{22}Y + r_{23}Zc + t_y}{Z_c'}, \quad (25)$$

where r_{ij} are the elements of the rotation matrix, and t_x, t_y, t_z are the elements of the translation vector. Here, Zc' is the z coordinate of the centroid of the object after rotation and translation.

On the other hand, a point in the projection image (x, y) before rotation and translation is expressed as:

$$x = f\frac{X}{Z_c} \quad (26)$$

$$y = f\frac{Y}{Z_c} \quad (27)$$

Therefore, movement of a point in the image after rotation and translation is expressed by the affine transform:

$$x' = \frac{Z_c}{Z_c'}(r_{11}x + r_{12}y) + \frac{f}{Z_c'}(r_{13}Zc + t_x) \quad (28)$$

$$y' = \frac{Z_c}{Z_c'}(r_{21}x + r_{22}y) + \frac{f}{Z_c'}(r_{23}Zc + t_y). \quad (29)$$

The effect of this affine transform on the moments is corrected for as follows. When $x' = ax + by + c$, $y' = dx + ey + f$, and $I'(x', y') = I(x, y)$, the moments of $I'(x', y')$ are

$$m'_{p,q} = \int\int (ax + by + c)^p (dx + ey + f)^q (ae - bd)dxdy \quad (30)$$

$$= (ae - bd)\sum_{i_1}\sum_{j_1}\sum_{i_2}\sum_{j_2} \frac{p!}{i_1!j_1!k_1!}\frac{q!}{i_2!j_2!k_2!} a^{i_1}b^{j_1}c^{k_1}d^{i_2}e^{j_2}f^{k_2}m_{i,j} \quad (31)$$

where $i = i_1 + i_2, j = j_1 + j_2, p = i_1 + j_1 + k_1, q = i_2 + j_2 + k_2$,
$i_1 \geq 0, j_1 \geq 0, k_1 \geq 0, i_2 \geq 0, j_2 \geq 0, k_2 \geq 0$.

Consequently, the moments of a planar object after rotation and translation are calculated from the moments before rotation and translation, which are calculated in advance.

Because it is difficult to calculate the Jacobi matrix $\partial\tilde{\mathbf{m}}(\tilde{\mathbf{x}})/\partial\tilde{\mathbf{x}}$ directly, it is calculated via parameters of the affine transform $\tilde{\mathbf{p}}(\tilde{\mathbf{x}}) = (\tilde{a}(\tilde{\mathbf{x}})\ \tilde{b}(\tilde{\mathbf{x}})\ \tilde{c}(\tilde{\mathbf{x}})\ \tilde{d}(\tilde{\mathbf{x}})\ \tilde{e}(\tilde{\mathbf{x}})\ \tilde{f}(\tilde{\mathbf{x}}))$ as follows:

$$\frac{\partial\tilde{\mathbf{m}}(\tilde{\mathbf{x}})}{\partial\tilde{\mathbf{x}}} = \frac{\partial\tilde{\mathbf{m}}(\tilde{\mathbf{x}})}{\partial\tilde{\mathbf{p}}(\tilde{\mathbf{x}})}\frac{\partial\tilde{\mathbf{p}}(\tilde{\mathbf{x}})}{\partial\tilde{\mathbf{x}}} \quad (32)$$

In addition, the moments of a convex polyhedron without self-occlusion can be calculated from the sum of the moments of all visible surfaces. Whether each surface is visible or not can be determined using a basic hidden surface elimination algorithm, or more easily, the moments can be obtained by dividing the sum of the moments of all surfaces by two.

Because the direction of the each surface before rotation and translation is different, the moment of each surface is calculated by moving the line of sight as to be orthogonal to the surface. In rotation and translation, the rotation matrix is calculated for each surface by considering the movement of the line of sight:

$$R_i = R\hat{R}_i^{-1}, \quad (33)$$

where R is the rotation matrix of the object, R_i is the rotation matrix of each surface, considering the movement of the line of sight, and $\hat{R}_i$ is the line of sight for each surface in the moment calculation before rotation and translation.

3) Arbitrary Shape: In case of an object with an arbitrary shape, because it is difficult to calculate the moments analytically from its position and orientation, it is necessary to generate a table in advance. Under weak perspective projection, because the change of the projection image by x, y, z, θ_z can be described by the affine transform, the table needs to have only θ_x, θ_y as inputs.

B. Simulation

We implemented the algorithm above with MathWorks' Matlab software and conducted a simulation. The target object was a box which followed the movement model of uniform translational and angular velocity, and a variation according to a normal probability distribution was added to translational velocity and angular velocity components in every frame. Its central projection image was given as an input image, and tracking was performed by the above algorithm, using moments up to 4th order with weights to make the moment values uniform because they differ depending on their order.

In the measurement of the moments from an input image, a rectangular measurement window a little larger than the target region calculated from the estimated position and orientation was generated, and the moments in the window were calculated. The measured moments and the moments calculated from the estimated position and orientation were both calculated with the projected coordinates of the centroid of the target calculated from the estimated position and orientation defined as the origin.

The tracking result is shown in Fig. 2. The vertices of the recognized box are indicated by bright points. A graph of the actual and estimated positions and orientations is shown in Fig. 3. The position and orientation were accurately recognized.

III. NOISE MODEL

The algorithm shown in the previous section does not work properly when applied to noisy images like Fig. 1. In calculation of the moments $m_{p,q}$ with either p or q being odd, like $m_{1,0}$ and $m_{0,1}$ which are used to obtain the centroid, changes in the moments due to noise in the measurement

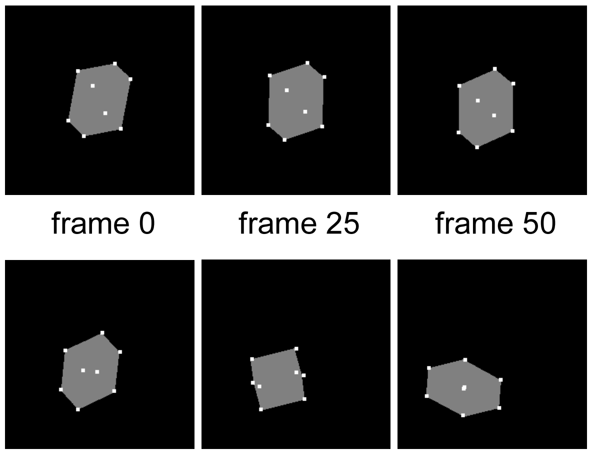

Fig. 2. Simulation Result: Box tracking using Newton's method.

window are averaged out to almost zero; however, with both of p and q being even, the changes do not averaged to zero.

It is thus necessary to calculate the expected values of the changes in the moments due to noise in the measurement window. It is necessary to model the noise for this calculation.

Consider the noise model below, for example. Assume that the ratio of pixels having a value of 1 in the original input silhouette image becoming 0 is α, and the ratio of pixels having a value of 0 becoming 1 is β.

Defining the original input image, i.e. the target image, as $I^{(T)}(x,y)$ and an image in which the pixel value in the window region is 1 as $I^{(W)}(x,y)$, the expected measured moments are

$$\begin{aligned}\bar{m}_{p,q} &= \int\int x^p y^q \Big(I^{(T)}(x,y)(1-\alpha) \\ &\quad + \Big(I^{(W)}(x,y) - I(x,y)\Big)\beta\Big)\,dxdy \quad (34)\\ &= (1-\alpha-\beta)m^{(T)}_{p,q} + \beta m^{(W)}_{p,q}, \quad (35)\end{aligned}$$

where $m^{(T)}_{p,q}$ are the moments of the target and $m^{(W)}_{p,q}$ are the moments of the image with the pixel value in the window region being 1. The moments of the target image can be estimated from the measured moments using the equation above.

Similarly, the variances of the measured moments can be obtained from:

$$\begin{aligned}\sigma^2_{p,q} &= \int\int \Big(\big(x^p y^q I^{(T)}(x,y)\big)^2(1-\alpha) \\ &\quad -\big(x^p y^q I^{(T)}(x,y)(1-\alpha)\big)^2 \\ &\quad \Big(x^p y^q\big(I^{(W)}(x,y) - I^{(T)}(x,y)\big)\Big)^2\beta \\ &\quad -\Big(x^p y^q\big(I^{(W)}(x,y) - I^{(T)}(x,y)\big)\beta\Big)^2 \\ &\quad \Big)dxdy \quad (36)\end{aligned}$$

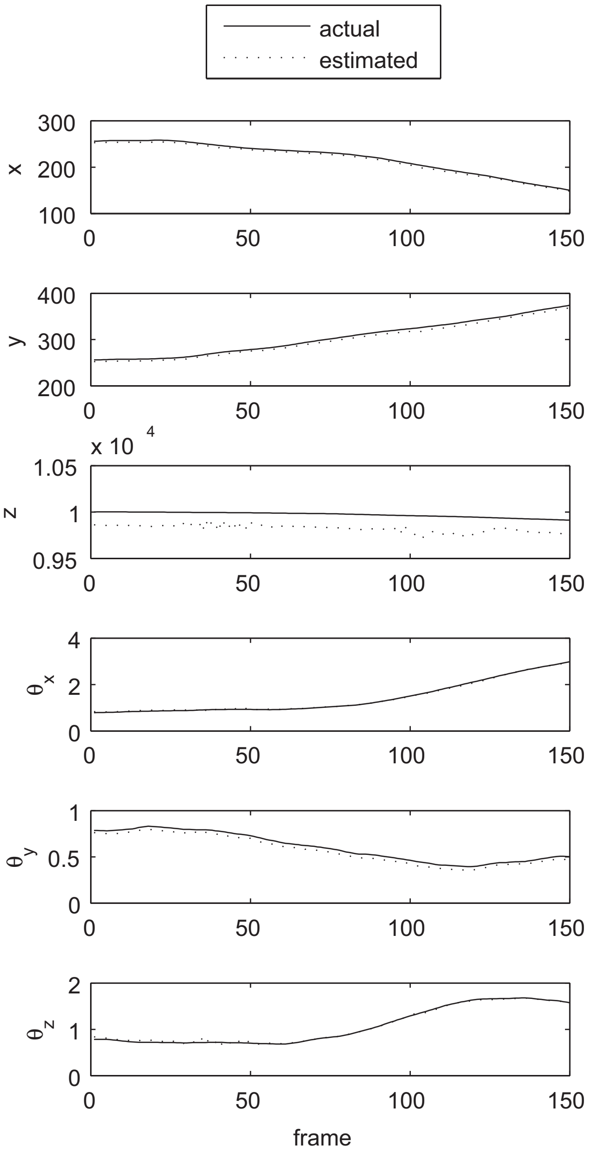

Fig. 3. Simulation Result: Box tracking using Newton's method.

$$\begin{aligned}&= \alpha(1-\alpha)m^{(T)}_{2p,2q} \\ &\quad +\beta(1-\beta)(m^{(W)}_{2p,2q} - m^{(T)}_{2p,2q}) \quad (37)\end{aligned}$$

These variances can be used for normalization to flatten the noise effect, depending on the order of the moment.

IV. 3D TRACKING USING PARTICLE FILTER

Even if the model above is introduced, Newton's method is not stable in the presence of a large amount of noise. Furthermore, the algorithm does not work with images having large motion between frames, though it is assumed that the motion between frames is small at high frame rates. An example of tracking failure is shown in Fig. 4

Newton's method sometimes does not properly converge if the difference between the initial value and the solution is large. Also, the effect of noise becomes larger as it converges in every frame.

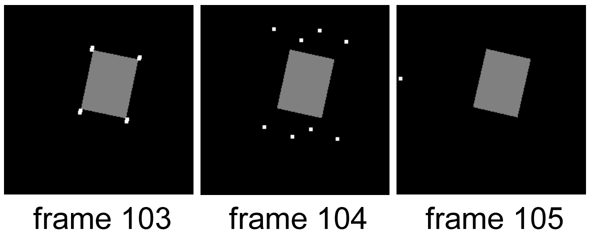

Fig. 4. Simulation Result: Example of tracking failure.

By using a time-series filter such as a Kalman filter or a particle filter [8], the convergence problem can be solved and the effect of noise can be averaged temporally .

We describe an algorithm using a particle filter below.

A. Algorithm

The candidates of a set of positions and orientations and velocity components of the target object in frame t are defined by $\mathbf{x}^{(i)}(t) = [x^{(i)}(t)\ y^{(i)}(t)\ z^{(i)}(t)\ \theta_x^{(i)}(t)\ \theta_y^{(i)}(t)\ \theta_z^{(i)}(t)\ \dot{x}^{(i)}(t)\ \dot{y}^{(i)}(t)\ \dot{z}^{(i)}(t)\ \dot{\theta_x}^{(i)}(t)\ \dot{\theta_y}^{(i)}(t)\ \dot{\theta_z}^{(i)}(t)]^{\mathbf{T}}$ $(i = 1, ..., P)$. Here, P is the number of candidates.

Based on the movement model, each candidate is updated as follows:

$$\tilde{\mathbf{x}}^{(i)}(t) = D\mathbf{x}^{(i)}(t-1) + \mathbf{n}^{(i)}(t-1) \tag{38}$$

where $\mathbf{n}^{(i)}(t)$ is Gaussian noise with an average of $\mathbf{0}$ and a covariance matrix of Σ.

From the squared error of the moments measured from the input image $\mathbf{m}(t)$ and the moments calculated from the position and orientation candidates, the likelihood of each candidate $w^{(i)}(t)$ is calculated:

$$\begin{aligned} w^{(i)}(t) &= \alpha\big(ae^{b} + 1\big) \quad (b < 0) && (39)\\ e &= ||\mathbf{m}(t) - \tilde{\mathbf{m}}(\tilde{\mathbf{x}}^{(i)}(t))||^2 && (40)\\ \alpha &= 1/\sum_i w^{(i)}(t) && (41) \end{aligned}$$

By creating and deleting candidates based on the obtained likelihood, $\{\mathbf{x}^{(i)}(t)|i = 1, ..., P\}$ is generated.

By executing this in every frame, the candidates near the position and orientation of the input image survive, and continuous tracking is thus realized.

B. Simulation

We executed the above algorithm in our simulation. Noise based on the noise model described in Section III, with $\alpha = \beta = 0.4$, was added to the same input image as that used in Section II. The number of candidates was 1000, and the average was regarded as the recognition result. Moments up to 4th order were normalized to flatten the noise effect by dividing each moment by its standard deviation, the square root of the variance described in Section III.

The tracking result is shown in Fig. 5. The vertices of the recognized box are indicated by bright points. Using the particle filter, we demonstrated that the position and orientation could be recognized even with noisy images. A graph of the actual and estimated positions and orientations is shown in Fig. 6. Though there are some errors in the recognized position and orientation, continuous tracking was realized.

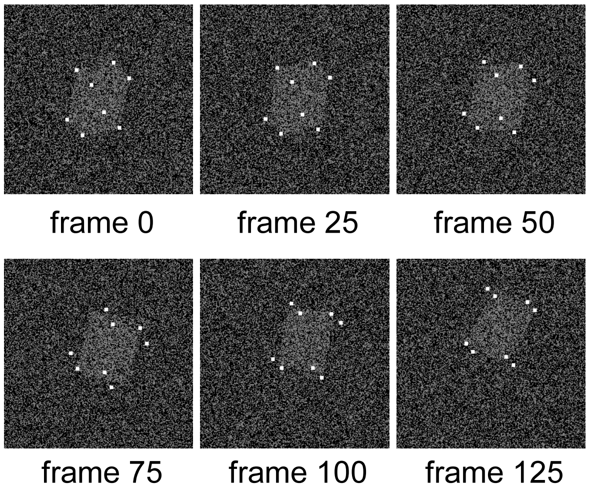

Fig. 5. Simulation Result: Box tracking using particle filter.

V. CONCLUSIONS AND FUTURE WORK

In this paper we propose a method for three-dimensional object tracking with rotation and translation using moments. By using a time-series filter, we showed that tracking with noisy images was possible. When using a particle filter with our proposed algorithm, the number of moments to be obtained from the input image is equal to the number of candidates. Therefore, we expect that the real-time processing at high frame rates will be difficult. Some techniques are required to overcome this problem, such as reusing the previous results for candidates having similar position and size of the measurement window. Future work on this technique will include simulation of arbitrary shapes, experiments using real images, and real-time implementation of the proposed algorithm.

REFERENCES

[1] N. Furukawa, A. Namiki, T. Senoo, M. Ishikawa, "Dynamic Regrasping Using a High-speed Multifingered Hand and a High-speed Vision System", Proc. IEEE Int. Conf. on Robotics and Automation, pp. 181-187 (2006).

[2] A. Namiki and M. Ishikawa, "The Analysis of High-speed Catching with a Multifingered Robot Hand", Proc. IEEE Int. Conf. on Robotics and Automation , pp.2666-2671 (2005).

[3] D. Lowe, "Fitting Parameterized Three-Dimensional Models to Images", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 13, No. 5, pp. 5-24 (1991).

[4] T. Moritani, S. Hiura, S. Inokuchi, "Object Tracking by Comparing CG Images with Multiple Viewpoint Images", Proc. IEEE conf. on Multisensor Fusion and Integration for Intelligent Systems, pp. 241-246 (2003)

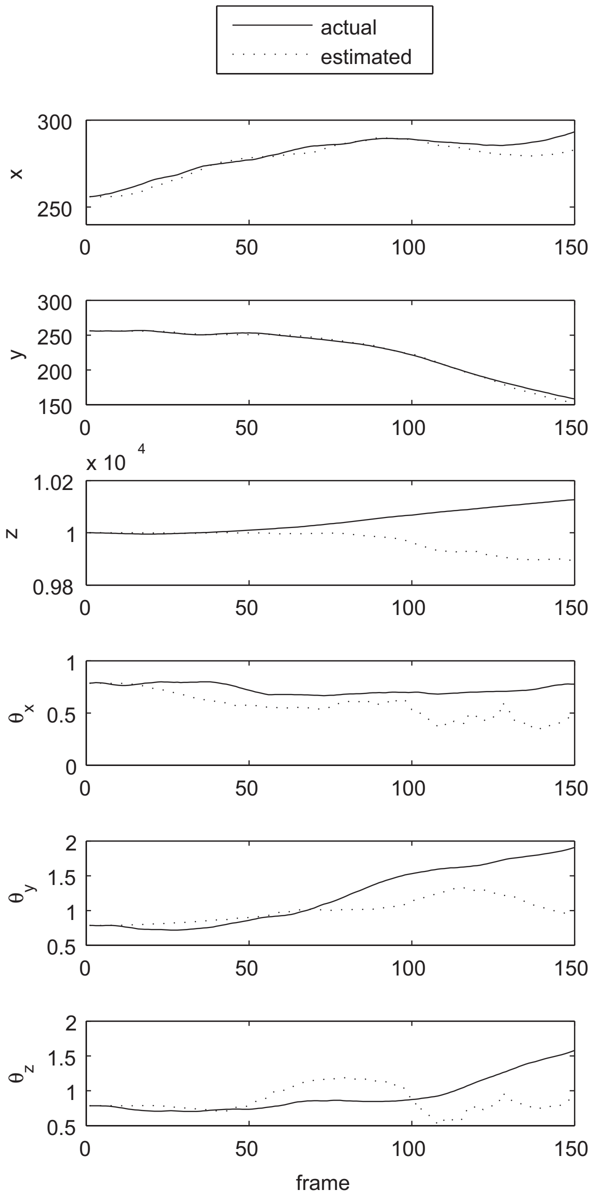

Fig. 6. Simulation Result: Box tracking using particle filter.

[5] H. Murase and S. Nayar, ”Visual learning and recognition of 3-D objects from appearance”, International Journal of Computer Vision, Vol. 14, pp. 5-24 (1995).
[6] O. Tahri and F. Chaumette, ”Complex Objects Pose Estimation Based on Image Moment Invariants”, Proc. Int. Conf. on Robotics and Automation, pp. 438-443 (2005).
[7] Y. Nakabo, I. Ishii, and M. Ishikawa, ”3D Tracking Using Two High-Speed Vision System”, Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 360-365 (2002).
[8] A. Blake and M. Isard, ”The Condensation Algorithm - conditional density propagation and applications to visual tracking”, Int. J. Computer Vision, Vol. 29, No.1, pp. 5-28 (1998).