The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Online 3-D Trajectory Estimation of a Flying Object from a Monocular Image Sequence

R. Herrejon, S. Kagami and K. Hashimoto

Abstract— **The problem considered here involves the design and application of a recursive algorithm to extract and predict the position of an object in a 3D environment from one feature correspondence from a monocular image sequence. Translational model involves an object moving in a parabolic path using projectile physics. A state-space model is constructed incorporating kinematic states, and recursive techniques are used to estimate the state vector as a function of time.**

The measured data are the noisy image plane coordinates of object match taken from image in the sequence. Image plane noise levels are allowed and investigated. The problem is formulated as a tracking problem, which can use an arbitrary large number of images in a sequence. The recursive estimation is done using Recursive Least Squares (RLS). Results on both synthetic and real imagery illustrate the performance of the estimator.

I. INTRODUCTION

Visual ability to locate and track rigid objects undergoing motion from monocular or binocular image sequences is an important issue in computer vision, because of its potential application in object recognition, robot vision and autonomous navigation.

The problem of determining location and motion from a sequence of images has been studied extensively [1]-[7]. A major difference exists between motion and structure estimation from binocular image sequences and that from monocular image sequences. With binocular image sequences, once the baseline is calibrated, the 3-D position of the object with reference with the cameras can be obtained. Motion recovery techniques can be divided generally into two categories; flow based and correspondence based. This paper uses the later.

The research in [1] considers the problem of reconstructing the 3D coordinates of a moving point seen from a monocular camera under motion using trajectory triangulation. Using a single moving camera with known projection matrices is the same as using a number of cameras, but the fact that the object is moving impides the use of position triangulation to obtain the position in 3D. When the object follows a lineal trajectory, the algorithm was able to predict the object's movement, but results when the object moves following a conic trajectory, the reconstructe points in 3D of a radious 5 percent off from the ground truth and around 4 degrees in orientation.

Manuscript received March 1, 2009

R. Herrejon is a candidate of PhD in the Intelligent Control Systems Laboratory at the Department of System Information Sciences of Tohoku University, Sendai, Japan rafael@ic.is.tohoku.ac.jp

S. Kagami is the associate professor of the Intelligent Control Systems Laboratory at the Department of System Information Sciences of Tohoku University, Sendai, Japan swk@ic.is.tohoku.ac.jp

K. Hashimoto is the professor of the Intelligent Control Systems Laboratory at the Department of System Information Sciences of Tohoku University, Sendai, Japan koichi@ic.is.tohoku.ac.jp

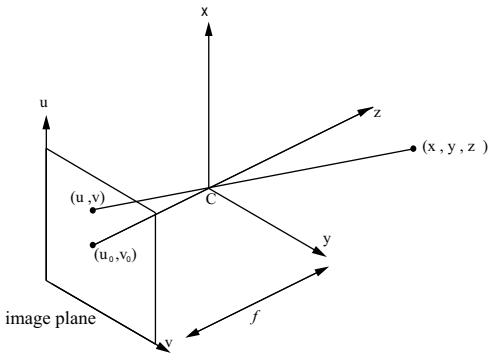


Fig. 1. Coordinate frame from the camera.

Extended Kalman Filter has been by several researchers to solve pose estimation and motion [5]-[7]. In these researches, previous knowledge of the structure or a model of the object is considered. Recursive and batch estimation approaches to extract the object motion parameters of an object under 2D constant translation and rotation from a sequence of monocular images of known structures are presented in [7].

II. GENERAL PROBLEM AND NOTATION

Taking 3D points to a 2D plane is the objective of projective geometry. Due to its importance in artificial vision, work on this area has been used and developed thoroughly. The approach to determine motion consists of two steps: 1) Extract, match and determine the location of corresponding features, 2) Determine motion parameters from the feature correspondences. In this paper, only the second step is discussed.

A. Camera model

The standard pinhole model is used throughout this article. Consider an isolated rigid body viewed by a camera as shown in Fig. 1. The camera coordinate system is assigned so as the x and y axis form the basis for the image plane, the z-axis is perpendicular to the image plane and goes through its optical center (c_u, c_v). Its origin is located at a distance f from the image plane. Using a perspective projection model, every 3-D point $\mathbf{P} = [X, Y, Z]^T$ on the surface of an object is deflated to a 2D point $\mathbf{p} = [u, v]^T$ in the image plane via a linear transformation known as the projection or intrinsic matrix $\mathbf{A}$.

$$\mathbf{A} = \begin{bmatrix} -f_u & 0 & c_u \\ 0 & -f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where f_u and f_v are conversion factors transforming distance units in the retinal plane into horizontal and vertical image pixels.

The projection of a 3D point on the retinal plane is given by

$$s\tilde{\mathbf{p}} = \mathbf{A}\mathrm{P} \quad (2)$$

where $\tilde{\mathbf{p}}$ is an augmented vector and s is an arbitrary scale factor. From this model, it is clear that any point in the line defined by the projected and original point produces the same projection on the retinal plane.

B. 3D Rigid-Body Motion

In this coordinate system, the camera is stationary and the scene is moving. For simplicity, assume that the camera takes images at regular intervals. As the rigid object move with respect to the camera, a sequence of images is obtained.

The motion of a rigid body in a 3D space has six degree of freedom. These are the three translation components of an arbitrary point within the object and the three rotation variables about that point. Matematically, it can be represented by

$${}^c\mathbf{P}(t_i) = {}^c\mathbf{R}(t_i)_o{}^o\mathbf{P}(t_i) + {}^c\mathbf{t}(t_i)_o \quad (3)$$

where the vector ${}^o\mathbf{P}(t_i)$ consists of the coordinates $x(t_i), y(t_i), z(t_i)$ relative to a selected object axis, the vector ${}^c\mathbf{P}(t_i)$ consists of the coordinates $x(t_i), y(t_i), z(t_i)$ relative to a camera axis, the matrix ${}^c\mathbf{R}(t_i)_o$ represents the rotation matrix around the focal point and the translation vector ${}^c\mathbf{t}(t_i)_o$ describes the displacement of the center of the coordinate system.

The position of a point at time t_i can be calculated with

$$x_i = C_1 + C_2 t_i + C_3 t_i^2 \quad (4)$$

$$y_i = C_4 + C_5 t_i + C_6 t_i^2 \quad (5)$$

$$z_i = C_7 + C_8 t_i + C_9 t_i^2 \quad (6)$$

where C_1, C_4, C_7 are initial positions, C_2, C_5, C_8 are velocities and C_3, C_6, C_9 are accelerations in x, y, z axis respectively.

C. Observation Vector

From (2), let the perspective of $\mathbf{P}_i$ be $\mathbf{p}_i = (u_i', v_i', 1)^T$. Its first two components u_i', v_i' represent the position of the point in image coordinates, and are given by

$$u_i' = -f_u \frac{x_i}{z_i} + c_u \quad (7)$$

$$v_i' = -f_v \frac{y_i}{z_i} + c_v. \quad (8)$$

If $u_i = u_i' - c_u$ and $vi = v_i' - c_v$, (7, 8) can be expressed as

$$u_i = -f_u \frac{x_i}{z_i} \quad (9)$$

$$v_i = -f_v \frac{y_i}{z_i}. \quad (10)$$

Substituting (4, 5, 6) into (9) and (10) to obtain

$$u_i = -f_u \frac{C_1 + C_2 t_i + C_3 t_i^2}{C_7 + C_8 t_i + C_9 t_i^2} \quad (11)$$

$$v_i = -f_v \frac{C_4 + C_5 t_i + C_6 t_i^2}{C_7 + C_8 t_i + C_9 t_i^2}. \quad (12)$$

Reordering and multiplying (11) and (12) by a constant d yields

$$d(C_7 u_i + C_8 u_i t_i + f_u C_1 + f_u C_2 t_i + f_u C_3 t_i^2) = -dC_9 t_i^2 u_i, \quad (13)$$

and

$$d(C_7 v_i + C_8 v_i t_i + f_v C_4 + f_v C_5 t_i + f_v C_6 t_i^2) = -dC_9 t_i^2 v_i. \quad (14)$$

By considering

$$dC_9 = 1 \quad (15)$$

we have the equation describing the state observation as follows

$$\mathbf{H}_i \mathbf{a}_i + \boldsymbol{\mu}_i = \mathbf{q}_i, \quad (16)$$

where $\boldsymbol{\mu}_i$ is a vector representing the noise in observation, $\mathbf{H}_i$ is the state observation matrix given by

$$\mathbf{H}_i = \begin{bmatrix} f_u & f_u t_i & f_u t_i^2 & 0 & 0 & 0 & u_i & u_i t_i \\ 0 & 0 & 0 & f_v & f_v t_i & f_v t_i^2 & v_i & v_i t_i \end{bmatrix}, \quad (17)$$

$\mathbf{a}_i$ is the state vector

$$\mathbf{a}_i = \begin{bmatrix} dC_1 & dC_2 & dC_3 & dC_4 & dC_5 & dC_6 & dC_7 & dC_8 \end{bmatrix}^T \quad (18)$$

and

$$\mathbf{q}_i = [-u_i t_i^2, -v_i t_i^2]^T \quad (19)$$

is the observation vector.

Considering one point in the space as the only feature to be tracked (the center of mass of an object), the issue of acquiring feature correspondences is dramatically simplified, but it is impossible to determine uniquely the solution. If the rigid object was n times farther away from the image plane but translated at n times the speed, the projected image would be exactly the same.

In order to be able to calculate the motion, one constraint in motion has to be added. We consider the case of a not-self propelled projectile, in this case, the vector of acceleration is gravity.

$$C_3^2 + C_6^2 + C_9^2 = \frac{g^2}{4} \quad (20)$$

Equation 20 is a constraint given by the addition of the decomposition of the vector of gravity in its different components on each axis for a free falling object.

Substituting C_3, C_6 and C_9 from (15) and (18) into (20) yields

$$\frac{1}{d^2}a_3^2 + \frac{1}{d^2}a_6^2 + \frac{1}{d^2} = \frac{g^2}{4}. \tag{21}$$

From (21) the constant d can be calculated as

$$d = 2\sqrt{\frac{a_3^2 + a_6^2 + 1}{g^2}}. \tag{22}$$

III. Estimation Method

Recursive least squares is used to find the best estimate of the state from the previous state. The best estimate for time i is computed as

$$\hat{\mathbf{a}}_i = \hat{\mathbf{a}}_{i-1} + \mathbf{K}_i(\mathbf{q}_i - \mathbf{H}_i\hat{\mathbf{a}}_{i-1}). \tag{23}$$

where $\mathbf{K}_i$ is the gain matrix, $\mathbf{q}_i$ is the measurement vector for one point, and $\mathbf{H}_i$ is the projection matrix and given by the camera model and time.

The equation that describes the computation of the gain matrix is

$$\mathbf{K}_i = \mathbf{P}_i\mathbf{H}_i^T. \tag{24}$$

$\mathbf{P}_i$ is the covariance matrix for the estimation of the state i, and can be expressed mathematically as

$$\mathbf{P}_i = (\mathbf{P}_{i-1}^{-1} + \mathbf{H}_i^T\mathbf{H}_i)^{-1}. \tag{25}$$

The accuracy of the estimation depends of the number of points projected in the camera plane. This number is directly related to the position, orientation and focus of the camera with respect to the path of the object. Assuming we can observe enough points, the error from the calculated path and the projected path tends to zero.

IV. Experiments

To assess the performance of the presented algorithm, simulations as well as experiments with real monocular image sequences have been conducted.

The performance of the presented algorithm was tested on simulated as well as real image sequences. The algorithm is the same in both cases, but the performance analysis is done differently because the correct values are only known for the experiments using simulated data.

A. Trajectory prediction using synthetic data

Camera internal parameters and operation parameters (rotation angles) are specified. The trajectory of the object is modeled using projectile physics in which the horizontal and vertical motion are independent of each other. Due to absence of acceleration in the horizontal direction, the velocity in horizontal direction remains unchanged. The velocity in vertical direction changes due to free fall acceleration.

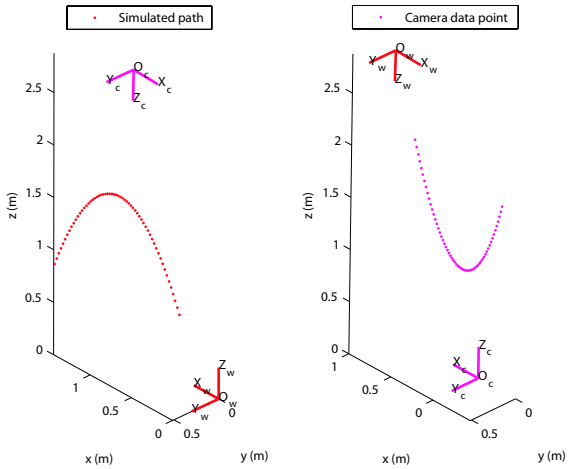


Fig. 2. Path of the object in a)world and b) camera coordinates

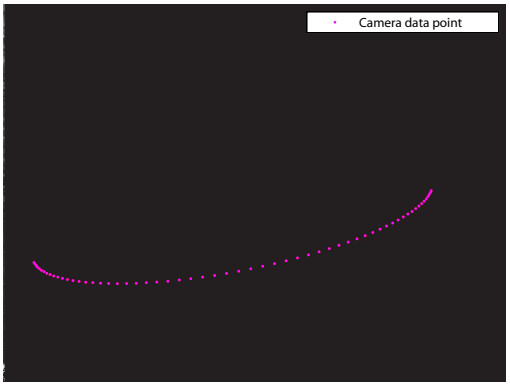


Fig. 3. Projection of points in the image plane

The object motion in world coordinates considered for this simulation (Fig. 2.a) is given by

$$^w x_{sim}(t) = 1.465 - 1.5t \tag{26}$$

$$^w y_{sim}(t) = 0.509 - 0.25t \tag{27}$$

$$^w z_{sim}(t) = 0.8 + 4.318t + \frac{1}{2}gt^2 \tag{28}$$

The coordinates of the object with respect to the camera can be calculated by

$$^c\mathbf{X}(t) = {}^c\mathbf{R}_w\,{}^w\mathbf{X}(t) + {}^c\mathbf{t}_w \tag{29}$$

where ${}^c\mathbf{R}_w$ is the rotation matrix from world to camera coordinates. First, a rotation about the x-axis, then about the y-axis, and finally the z-axis is considered. This sequence of rotations can be represented as the matrix product $\mathbf{R} = R_z(\phi)R_y(\theta)R_x(\psi)$.

The image of the simulated camera is a rectangle with a pixel array of 480 rows and 640 columns. The number of frames used is 60 at a sampling rate of 69 MHz, which accounts for a flying time of 0.87 seconds. Experimental results for four different noise levels in two different camera configurations are conducted.

The performance of the estimation algorithm in each frame is evaluated by comparing the simulated position with the fitted position for the whole trajectory. The error e_i is given by

$$e_i = \sum_{t=0}^{T}((^c x_{sim}(t) - {}^c x_{fit}(t))^2 + (^c y_{sim}(t) - {}^c y_{fit}(t))^2 + (^c z_{sim}(t) - {}^c z_{fit}(t))^2) \tag{30}$$

1) Camera position 1: In this simulation, the camera pose is given by rotating $\psi = 3.1806135$, $\theta = -0.0123876$ and $\phi = .0084783$ radians in the order previously stated. The translation vector is given by $\mathbf{t} = [0.889; -0.209; -2.853]$ meters. Using the previous parameters in (29), the object's motion in camera coordinates is given by

$$^c x_{sim}(t) = -0.599 + 1.374t + 0.137t^2 \tag{31}$$

$$^c y_{sim}(t) = 0.317 - 0.299t + 0.030t^2 \tag{32}$$

$$^c z_{sim}(t) = 2.091 - 4.356t + 4.898t^2 \tag{33}$$

as shown in Fig. 2.b.

The coordinates (u, v) projected in the image plane obtained when focal lengths $f_u = 799$, $f_v = 799$ and centers of image $c_u = 267.2$, $c_v = 205.7$ were utilized in (7,8) can be seen in Fig. 3.

Considering only this projected points and time known, the algorithm calculates the path in camera coordinates (31, 32, 33)

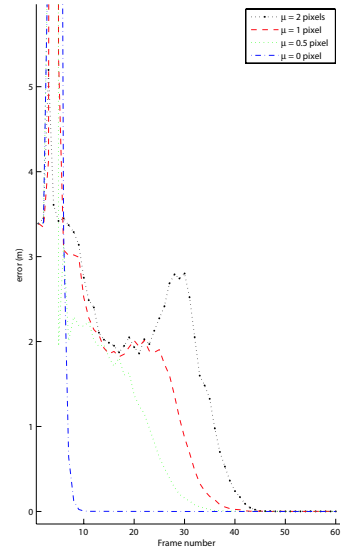
Case 1 No measurement error was considered in this case. The trajectory error converges quite well, after 10 frames, the error is smaller than the desired threshold of 0.005 meters. When all the frames have been calculated, the error has converged to 0 m. The obtained vector $C = [-0.599, 1.374, 0.137, 0.317, -0.299, 0.030, 2.091, -4.357, 4.898]^T$

Case 2 A moderately measurement error of 0.5 pixels was considered in this case. The trajectory error converges to less than 0.005 meters in frame 38. When the data of all the frames have been calculated, the error converges to 0.0002 m. The obtained vector $C = [-0.6000, 1.3768, 0.1377, 0.3175, -0.2978, 0.027, 2.093, -4.356, 4.898]^T$

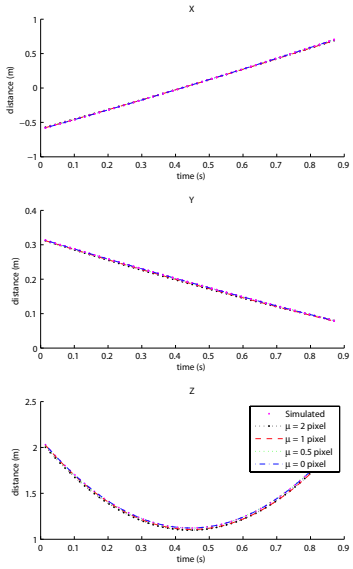
Case 3 In this case, a measurement error of 1 pixels was considered. The trajectory error converges to less than 0.005 meters in frame 43, and the error for the last frame is 0.0009 m. The obtained vector $C = [-0.608, 1.413, 0.096, 0.321, -0.309, 0.034, 2.118, -4.408, 4.898]^t$

Case 4 The measurement error in this case was fairly high (2 pixels). The trajectory error converges in frame 45, with an error for the last frame of 0.0005 m. and vector $C = [-0.607, 1.424, 0.064, 0.320, -0.306, 0.026, 2.117, -4.446, 4.899]^T$

Fig. 4.a shows the error of the fitted trajectory with different measurement errors considered. Fig. 4.b shows the fitted trajectory on xyz with different measurement errors considered.



a) Square error of the trajectory



b) Result of the estimation of fitted path in xyz

Fig. 4. Results of trajectory estimation with camera position 1

2) Camera position 2: In this other example, the camera pose is given by rotating $\psi = -0.1299$, $\theta = 0.0218$ and $\phi = -1.6545$ radians, with translations $t_x = -0.2697$ meters, $t_y = 1.2121$ meters , $t_z = 0$ meters, focal lengths $f_u = 550$, $f_v = 550$ and centers of image $c_u = 267$, $c_v = 205$. The same object trajectory was used for comparison effects.

For this camera pose, the trajectory of the object in camera coordinates is given by

$$^c x_{sim}(t) = 0.212 + 0.427t - 0.623t^2 \tag{34}$$

$$^c y_{sim}(t) = -0.314 + 1.374t + 0.158t^2 \tag{35}$$

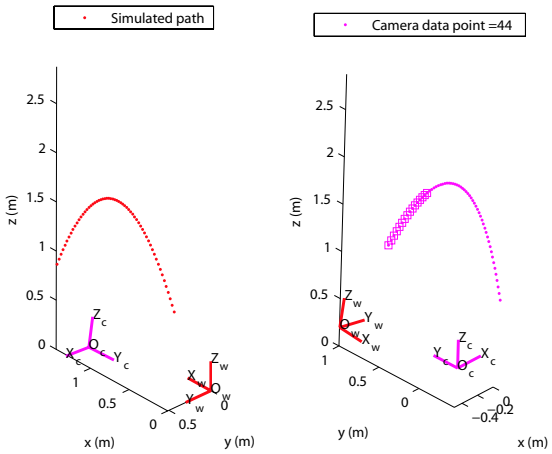


Fig. 5. Path of the object in a)world and b) camera coordinates

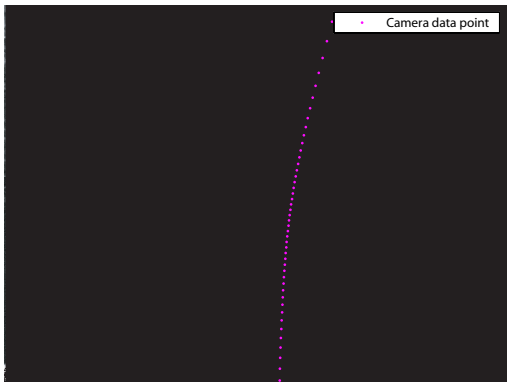


Fig. 6. Projection of points in the image plane

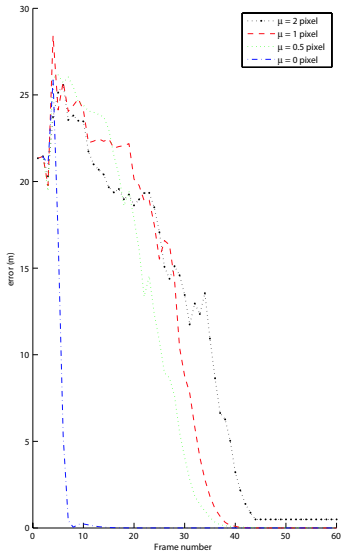
$$^{c}z_{sim}(t) = 0.695 + 4.345t - 4.857t^2 \quad (36)$$

The path in world coordinates and camera coordinates for this camera pose can be seen in Fig. 5. Because of the parameters chosen for this camera, not all the points of the trajectory are projected into the image. Positions that are not projected in the image plane are shown in the figure with a □. The image observed by the camera is shown in Fig. 6.

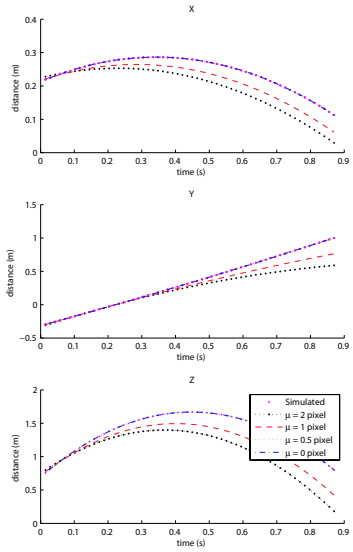
Case 1 No measurement error was considered in this case. The trajectory error converges quite well, after 10 frames, the error is smaller than the desired threshold of 0.005 meters. When all the frames have been calculated, the error has converged to 0 m. The obtained vector $C = [0.212, 0.428, -0.623, -0.314, 1.374, 0.159, 0.695, 4.346, -4.857]^T$

Case 2 A moderately measurement error of 0.5 pixels was considered in this case. The trajectory error converges to less than 0.005 meters in frame 38. When the data of all the frames have been calculated, the error converges to 0.0002 m. The obtained vector $C = [0.212, 0.430, -0.624, -0.313, 1.368, 0.175, 0.694, 4.360, -4.856]^T$

Case 3 In this case, a measurement error of 1 pixels was considered. The trajectory error converges to less than 0.005 meters in frame 43, and the error for the last frame is 0.0009 m. The obtained vector $C = [0.2120.399 - 0.614 - 0.3121.383\ 0.0500.6904.204 - 4.861]^t$



a) Square error of the trajectory



b) Result of the estimation of fitted path in xyz

Fig. 7. Results of trajectory estimation with camera position 2

Case 4 The measurement error in this case was fairly high (2 pixels). The trajectory error converges in frame 45, with an error for the last frame of 0.0005 m. and vector $C = [0.206, 0.341, -0.599, -0.308, 1.472, -0.341, 0.683, 3.831, -4.8511]^T$

Fig. 7.a shows the error of the fitted trajectory with different measurement errors considered. Fig. 7.b shows the fitted trajectory on X-Y-Z axes with different measurement errors considered.

B. Trajectory prediction using data from images

For this experiment, 58 images were taken with a Dragonfly Express Camera at 70 fps, the center of gravity of the object (a flipping coin) in the image plane (u, v) is used to calculate the trajectory. Camera calibration to obtain the intrinsic parameters was realized. Because the coin is turning, the calculated center of gravity varies accordingly to the image obtained, blur in the image might cause errors in the calculation of the center of the coin. With real data, the algorithm takes longer time to fit the path than with simulated data, this maybe due the noise in the images. The center of mass as observed by the camera is shown in Fig. 8. After the object was tossed, it landed on a table parallel to the ground and approximately 2.1 meters away from the center of the camera on its Z-axis. As observed in Fig. 9, the position in Z closely resembles that of our setup. It is difficult to corroborate the validity of the found path in 3D, but the results obtained for the path resembles the values obtained with simulated data. The fitted path in the image plane converges after 38 frames.

Mean absolute error was used to measure the performance of estimator. After all the positions were calculated, the error in v was 0.396 pixels, and in u was 1.0862 pixels. This difference is due to the error in the calculated center of gravities of the coin. In other words, the path calculated and projected into the image is more correct than the values obtained directly from the camera.

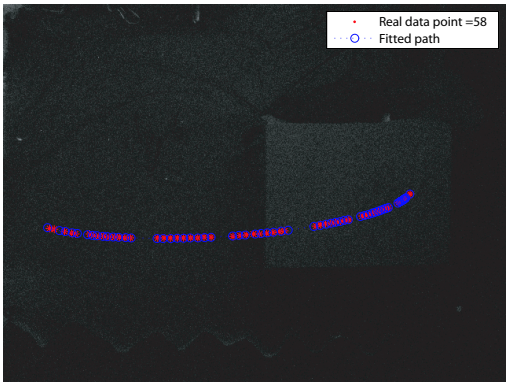


Fig. 8. The center of mass as observed by the camera

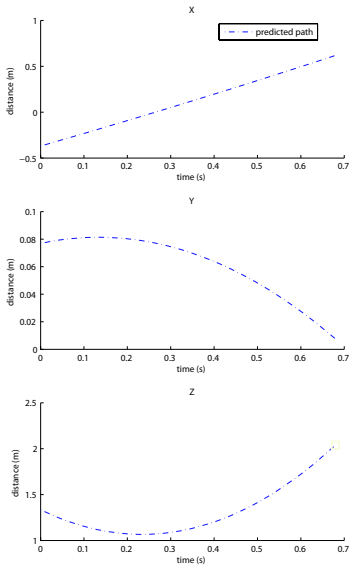


Fig. 9. Calculated path in 3D

V. CONCLUSIONS AND FUTURE WORK

A. Conclusions

This paper presented a recursive least squares (RLS) algorithm to extract and predict the position of a flying object in a 3D environment from one feature correspondence from a sequence of noisy monocular images. The trajectory path was obtained successfully even under high noise images. The recursive estimation technique presented in this paper has numerous advantages over other methods currently in use. First, using only one feature point, the issue of feature points correspondence is simplified. Another advantage is the recursive nature of the computations makes it suitable for real-time applications. Results on simulation and real imagery illustrate the performance of the estimator. The results obtained in both simulation and experiments were good. Current research is directed towards the obtainment of the rotation parameters of the camera from the relation that exists between the accelerations in the camera coordinates and gravity.

REFERENCES

[1] S. Avidan and A. Shashua. Trajectory Triangulation: 3D Reconstruction of Moving Points from a Monocular Image Sequence, *IEEE. Trans of Pat, An. and Mac. Int*, Vol. 22, pp. 348-357, 2000.

[2] N. Cui, J. J. Weng and P. Cohen Recursive-Batch Estimation of Motion and Structure from Monocular Image Sequences, *CVGIP: Image Understanding*, Vol. 59, pp. 154-170, 1994.

[3] C. Chan, A. Guesalaga and V. Obac Robust Estimation of 3D Trajectories from a Monocular Image Sequence *International journal of imaging systems and technology* , Vol. 12, pp. 128-137, 2002.

[4] N. P. Papanikolopoulos, P. K. Khosla and T. Kanade Visual tracking of a moving target by a camera mounted on a robot: combination of control and vision *Trans. Robotics Automat.* Vol. 9, pp. 14-35, 1993.

[5] S. Lee and Y. Kay A Kalman Filter Approach for Accurate 3-D Motion Estimation from a Sequence of Stereo Images. *CVGIP; Image Understanding*, Vol. 54. pp. 244-258, 1991.

[6] J. Wang and W. J. Wilson. 3D Relative Position and Orientation Estimation Using Kalman Filter for Robot Control," *Proc. IEEE Int. Conf. on Rob. and Aut.*, pp. 2638-2645, 1992.

[7] T. J. Broida and R. Chellapa. Estimation of object motion parameters from noise images, *IEEE. Trans of Pat, An. and Mac. Int*, Vol. 8, pp. 90-99, 1986.

[8] O. Faugeras. *Three-Dimensional Computer Vision, A Geometric Viewpoint*, MIT Press, Cambridge, Massachusetts, 2001.

[9] Z. Zhang. A flexible new technique for camera calibration *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22 (11), pp. 1330-1334, 2000.

[10] K. Hashimoto. A review on vision-based control of robot manipulators. *IEEE Int. Conf. Robotics and Automation*, pp. 2254-2260, Sacramento, CA, 1991.

[11] R. L. Anderson. *A robot ping-pong player: Experimental in Real-Time Intelligent Control*, ATT Bell Laboratories, MIT Press, 1989.

[12] W. Hong and J.J.E. Slotine. Experiments in Hand-Eye Coordination Using Active Vision *Proc. 4th Int. Symposium on Experimental Robotics*, Stanford, CA, 1995.