Trajectory Triangulation over Conic Sections

Amnon Shashua Shai Avidan Michael Werman
Institute of Computer Science,
The Hebrew University,
Jerusalem 91904, Israel
e-mail: {shashua,avidan,werman}@cs.huji.ac.il

Abstract

We consider the problem of reconstructing the 3D coordinates of a moving point seen from a monocular moving camera, i.e., to reconstruct moving objects from line-of-sight measurements only. The task is feasible only when some constraints are placed on the shape of the trajectory of the moving point. We coin the family of such tasks as "trajectory triangulation". In this paper we focus on trajectories whose shape is a conic-section and show that generally 9 views are sufficient for a unique reconstruction of the moving point and fewer views when the conic is a known type (like a circle in 3D Euclidean space for which 7 views are sufficient). Experiments demonstrate that our solutions are practical.

The paradigm of Trajectory Triangulation in general pushes the envelope of processing dynamic scenes forward. Thus static scenes become a particular case of a more general task of reconstructing scenes rich with moving objects (where an object could be a single point).

1 Introduction

We wish to remove the static scene assumption in 3D-from-2D reconstruction. This paper introduces another stage in a new paradigm we call "trajectory triangulation" that pushes the envelope of processing "dynamic scenes" from "segmentation" to 3D reconstruction.

Consider the situation in which a 3D scene containing a mix of static and moving objects is viewed from a moving monocular camera. The typical question addressed in this context is that of "segmentation": can one separate the static from dynamic in order to calculate the camera ego-motion (and 3D structure of the static portion)? this question is basically a robust estimation issue and has been extensively (and successfully) treated as such in the literature (cf. [6, 5]). A byproduct of the robust estimation is the segmentation of the scene to the static and dynamic portions, or to the portions corresponding to multiply moving objects.

However, consider the next (natural) question in this context: can one reconstruct the 3D coordinates of a (single) point on a moving object? Unlike the segmentation problem, the reconstruction problem *is not feasible*, unless further constraints are imposed. In order to reconstruct the coordinates of a 3D point, the point must be static in at least two views (to enable triangulation) — if the point is moving generally then the task of triangulation is not feasible. Note that the feasibility issue arises regardless of whether we assume the ego-motion of the camera to be known or not. Knowledge of camera ego-motion does not change the feasibility of the problem.

The feasibility status changes when we constrain the trajectory of the moving point to belong to some (parametric) family of trajectories. We call the topic of reconstructing moving points, whose motion (in 3D) is constrained parametrically, from general multiple 2D projections as "trajectory triangulation". In the sequel we assume that the camera ego-motion (projection matrices) is known. We acknowledge the difficulty of recovering the camera ego-motion in general, and under dynamic scene conditions in particular, but believe it to be reasonable in view of the large body of theoretical and applied literature on the subject. Thus, we treat the problem of ego-motion as a "black-box" and a first layer in a hierarchy of tasks that are possible in a "3D-from-2D" family of problems.

In this paper we extend the notion of "linear trajectory triangulation" (see section below) to second-order trajectories (See Figure 1). In other words, we investigate the problem of a point moving along some 3D conic trajectory and show that the reconstruction can be done in a practical manner. Extensions and future work are discussed in Section 5.

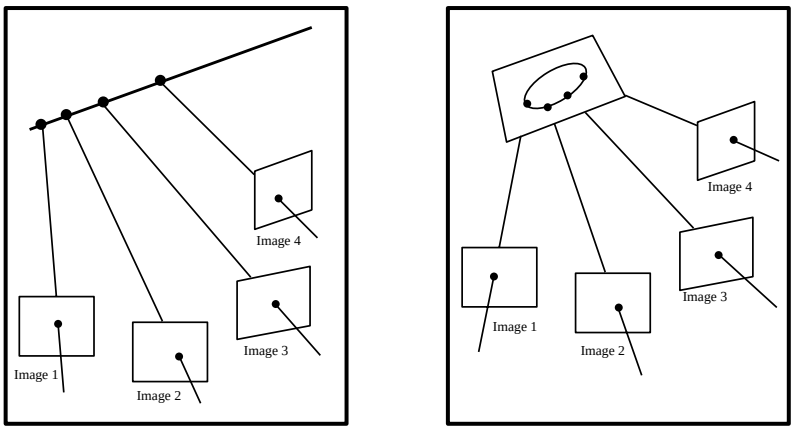

Figure 1: (a) "Trajectory Triangulation" along a line [1]. A point is moving along a line while the camera is moving. (b) "Trajectory triangulation" along a planar conic. A point is moving along a planar conic while the camera is moving.

2 Related Work

The problem of trajectory triangulation of a point moving along some straight line path was discussed in [1]. Also related to the problem addressed in this paper, conic trajectories, is the problem of "orbit determination" in astro-dynamics (cf. [3]). We briefly discuss below these two related sources.

The case of a linear trajectory (straight-line path) has a simple geometric intuition: the projection of a moving point gives rise to a collection of 3D rays which are the lines of sight to the moving point. Because the trajectory of the moving point is a straight line the collection of rays form a "linear line complex", i.e., they have a common intersecting line (the trajectory of the moving point) as their kernel. Thus, the problem formulation is to figure out the conditions (number of views) for a unique kernel and to follow it with an algebraic solution.

The algebraic method for recovering the kernel is based on representing the kernel (the trajectory of the moving point) with its Plucker coordinates. One can then show that each projection provides a linear equation for the kernel, and thus 5 equations are necessary for a unique solution (with 4 views one can obtain two solutions using the quadratic constraint of plucker representation). So, 5 views are sufficient to linearly recover for the trajectory of a point moving on a line. Further details and implementation can be found in [1].

The problem of *orbit determination* in astrodynamics is about determining the orbit (conic section, typically elliptic) of body A around body B under a gravitational field. A branch of this problem includes the determination of an orbit from directional measurements only (lines of sight). However, the assumption of motion under a gravitational field constrains not only the shape of the trajectory (conic section) but also the law of motion *along* the trajectory — in this case the motion is Keplerian, which is to say that equal areas are swept during equal times (See Figure 2).

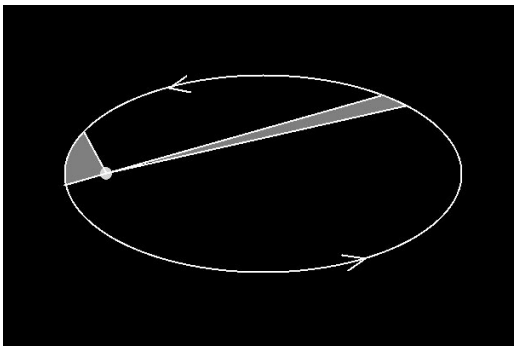

Figure 2: In keplerian motion a body sweeps equal areas in equal time.

Our work on determining a conic trajectory from line of sight measurements (trajectory triangulation over conics) differs from the classic work on orbit determination by that *the motion of the point along the trajectory is arbitrary.* In other words, the only assumption we make is about the *shape* of the trajectory (conic section) while the motion of the point along the trajectory is unconstrained. Therefore, what we wish to recover are the following parameters: the position

of the plane on which the conic resides (3 parameters) and the position, shape and type of the conic (5 parameters). Once these parameters are recovered it becomes a simple matter to determine the 3D coordinates of the moving point at each frame of the image sequence.

3 Trajectory Triangulation over Conics

As mentioned above we wish to recover — from measurements of line-of-sight only (2D projections of the moving point) — 8 parameters in general: 3 for the position of the plane on which the conic resides on, and 5 for the conic itself. Once these parameters are recovered the 3D coordinates of the moving point can be recovered by intersecting the line-of-sight with the conic section.

We propose two methods for recovering the parameters. The first method performs a 2D optimization (based on conic fitting) on some arbitrary virtual common plane. The method is very simple, but can only deal with general conics — a-priori constraints on the shape of the 3D conic cannot be enforced due to the projective distortion from the conic plane to the virtual common plane.

The second method is slightly more complex as the optimization is performed in 3D (projective or Euclidean) but enables the enforcement of a-priori constraints on the shape of the conic when the cameras are calibrated. Numerical stability is greatly enhanced when a-priori information is integrated into the estimation process. We will derive the second method for the case of calibrated cameras and when the conic in 3D is a circle. The extension to general conics follows in a straightforward manner but will not be derived here.

3.1 Method I: 2D Optimization on a Common Plane

We denote the 3D position of the moving point and the camera matrix (projection matrix) at time $i, i = 1..k$ by $P_i = [X_i, Y_i, Z_i, 1]^T$ and $M_i = [H_i \; ; \; t_i]$, respectively. The image measurements are thus $p_i \cong M_i P_i$. Our goal is to recover the 3D points P_i, given the uncalibrated camera matrices M_i and the image measurements p_i. This can be formulated as a non-linear optimization problem in which 8 parameters are to be estimated. The 3 parameters of the normal to the plane $\mathbf{n}$ and the 5 parameters of the conic as defined (up to scale) by a symmetric 3×3 matrix C.

Let the sought-after plane on which the conic resides on be denoted by π. Let A_i be the 2D homography from image i to some common arbitrary plane (image plane $i = 1$ if $M_1 = [I; 0]$) through the plane π,

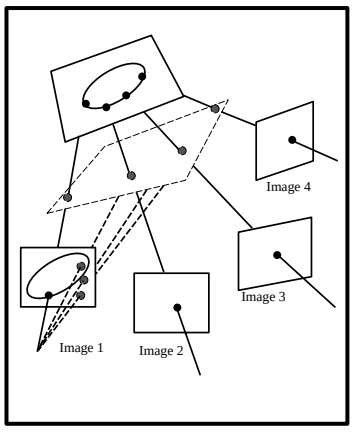

Figure 3: Sketch of method I. The true plane is shown in bold, the guessed plane is shown with dashed lines. Choosing a different plane affects the projection of the points on the first image (or common plane in general). If the plane is not the correct one, then the points on the first image will not form a conic.

i.e., $A_i p_i$, $i = 1, ..., k$ must be a conic on the common plane (see Fig. 3). The following relation must hold:

$$A_i = (\|\mathbf{n}\| H_i + t_i \mathbf{n}^\top)^{-1}.$$

Therefore, each view provides one (non-linear) constraint:

$$p_i^\top A_i^\top C A_i p_i = 0, \qquad i = 1, ..., k.$$

Since the total number of parameters are 8, and each view contributes one (non-linear) equation, then 8 views are necessary for a solution (up to a finite-fold ambiguity) and 9 views for a unique solution. It is possible to solve for $\mathbf{n}$ and C by means of numerical optimization, or to use an interleaving approach described below:

1. Start with an initial estimate of $\mathbf{n}$.
2. Compute $\hat{p}_i = A_i p_i$, where $A_i = (\|\mathbf{n}\| H_i + t_i \mathbf{n})^{-1}$.
3. Fit a conic C to the points $\hat{p}_i$.
4. Search over the space of all possible $\mathbf{n}$ to minimize the error term:

$$min_{\mathbf{n}}, err(C, \hat{p}_i))$$

There are a number of points worth mentioning. The minimization is over 3 parameters only due to step 3 of conic fitting. A large body of literature is

devoted to conic fitting and the numerical biases associated with this problem (cf. [5, 2, 4]). The error term in step 4 is also an important choice: the algebraic error $p^\top Cp$ between a point p and a conic C is least recommended because of numerical biases. In our implementation, for example, we have chosen to minimize the distance to the polar line Cp, i.e., $err(C,p) = dist(Cp,p)$. Finally, the search in step 4 is achieved (in our implementation) by Levenberg-Marquardt optimization using numerical differentiation. Using Matlab, the optimization step consists of simply calling the *leastsq* function.

To summarize, this approach has the two advantages. It is simple and is carried over the 2D plane only. The disadvantages are, first, that the method does not facilitate a-priori constraints on the shape of the conic, and second, the method involves a conic fitting (and evaluation) stage which could be challenging on the numerical front.

3.2 Method II: Conic fitting in 3D

In this method the objective function is minimized in 3D space and is designed such that it can express a-priori shape constraints, when available and when cameras are calibrated. The general idea is that a conic in 3D is represented by the intersection of the plane π and a quadric surface. By defining a suitable coordinate system of the quadric surface one can obtain an 8 parameter objective function. In case of calibrated projection matrices and if a-priori information about the type of conic is given, say a circle, then the quadric surface representation can be simplified further. We will derive here a special case in which the sought-after conic is a *circle* in 3D.

In the case of a circle, we wish to represent the arrangement of a sphere and a cutting plane. We expect the total number of parameters to be 6 (three for $\mathbf{n}$ and 3 for representing a circle in the plane), yet a sphere is defined by 4 parameters. Therefore an additional constraint is necessary and this is obtained by constraining the plane π to coincide with the center of the sphere. The details are below.

Let p_i and M_i be the projection and camera matrices of frame $i = 1, ..., k$ as defined previously. In case the cameras are calibrated, then the projection matrices represent the mapping from an Euclidean coordinate system to the image plane, i.e., $M_i = K_i[R_i \ ; \ u_i]$ where R_i, u_i are the rotational and translational components of the mapping, and K_i is an upper-diagonal matrix containing the internal parameters of the camera (focal length, aspect ratio, principle point). For our needs, since we assume M_i to be known, we can still denote M_i by the composition $M_i = [H_i \ ; \ t_i]$ as was done previously (thus, at this juncture it doesn't really matter whether the camera are calibrated or not). Let the 3D coordinates of the moving point P be denoted (as before) by $P_i = [X_i, Y_i, Z_i]^T$ at time $i = 1, ..., k$. We first represent P_i as a function of $\mathbf{n}$ as follows:

$$\lambda_i p_i = M_i P_i. \tag{1}$$

Which after substitution becomes:

$$\lambda_i p_i = [H_i \ t_i] \begin{bmatrix} P_i \\ 1 \end{bmatrix} \tag{2}$$

$$\lambda_i H_i^{-1} p_i = P_i + H_i^{-1} t_i \tag{3}$$

thus, P_i as a function of M_i and p_i becomes:

$$P_i = \lambda_i H_i^{-1} p_i - H_i^{-1} t_i. \tag{4}$$

Next, we know that the moving point resides on the plane π, thus

$$P_i \mathbf{n} + 1 = 0. \tag{5}$$

After substitution we obtain

$$\lambda_i = \frac{(H_i^{-1} t_i)^T \mathbf{n} - 1}{(H_i^{-1} p_i)^T \mathbf{n}}. \tag{6}$$

Taken together, eqn. 4 and λ_i above, give rise to:

$$P_i = \begin{bmatrix} X_i \\ Y_i \\ \frac{n_1 X_i + n_2 Y_i + 1}{-n_3} \end{bmatrix}$$

in which X_i, Y_i are functions of $\mathbf{n}$ (and Z_i is eliminated by being expressed as a function of $X_i, Y_i, \mathbf{n}$).

Let the center of sphere be at the coordinates $P_c = [X_c, Y_c, Z_c]$ and its radius R, thus the points P_i satisfy the constraint:

$$(X_i - X_c)^2 + (Y_i - Y_c)^2 + (Z_i - Z_c)^2 - R^2 = 0 \tag{7}$$

which can be written as

$$[P_i^T \ 1] Q \begin{bmatrix} P_i \\ 1 \end{bmatrix} \tag{8}$$

The 4×4 symmetric matrix Q is given by:

$$Q = \begin{pmatrix} 1 & 0 & 0 & q_1 \\ 0 & 1 & 0 & q_2 \\ 0 & 0 & 1 & q_3 \\ q_1 & q_2 & q_3 & q_4 \end{pmatrix}$$

where

$$\begin{aligned} q_1 &= X_c, \\ q_2 &= Y_c, \\ q_3 &= Z_c, \\ q_4 &= X_c^2 + Y_c^2 + Z_c^2 - R^2 \end{aligned} \tag{9}$$

Since the center of the circle P_c is on the plane π we have:

$$Z_c = \frac{n_1 X_c + n_2 Yc + 1}{-n_3} \qquad (10)$$

so we need to solve for the three parameters q_1, q_2, q_4. Taken together, each view provides one (non-linear) constraint (Eq. 7) over 6 parameters $\mathbf{n}$ and q_1, q_2, q_4. Thus, 7 views are necessary for a unique solution. As with Method I, it is possible to solve for the system over 6 parameters or to adopt an interleaving approach:

1. Start with an initial estimate of $\mathbf{n}$.
2. Compute the point $\hat{P}_i$ from eqn. 4 and 6.
3. Solve for q_1, q_2, q_4 (linear least-squares). P_c, R follow by substitution.
4. Search over the space of all possible $\mathbf{n}$ to minimize the error term:

$$min_{\mathbf{n}}, (dist(\hat{P}_i, P_c) - R)^2, \qquad i = 1, ..., k$$

where the search is done using numerical optimization (*leastsq* function of Matlab).

4 Experiments

We have conducted a number of experiments on both synthetic and real image sequences. We report here a typical example of a real image sequence experiment.

A sequence of 16 images was taken with a handheld moving camera viewing a small Lego piece on a turntable. The Lego piece is therefore moving along a circular path. The first, middle and last images of the sequence are shown in Fig. 4. The projection matrices were recovered from matching points on the static calibration object (the folded chess-board in the background). The corners of the chess-board were the control points for a linear system for solving for M_i for each image. The linear solution is not optimal but was good enough for acheiving reasonbale results for the trajectory triangulation experiments. A point on the Lego cube was then (manually) tracked over the sequence and its image positions p_i was recorded.

We tested both methods I,II. In general, the 3D-based optimization (method II) always converged from any initial guess of $\mathbf{n}$ (the position of the plane π). Fig. 6a shows the conic due to the intial guess that was used for this experiment, for example. The 2D-based optimization (method I) was more sensitive to the intial guess of $\mathbf{n}$, and Fig. 5a shows a typical initial guess. The remaining displays in Figs. 5 and 6

(a)

(b)

(c)

Figure 4: The original image sequence. (a),(b) and (c) are the first, middle and last images, respectively in a sequence of 16 images. The camera is moving mainly to the left while the Lego cube traces a circle on the turntable.

show the projection of the final conic (following convergence of the numerical optimization) on the first, middle and last images of the sequence. In method II, the reconstructed points in 3D define a circle (as it was constrained to begin with) of a radius 5% off from the ground truth, and around 4^o in orientation. In method I, the resulting conic had an aspect ratio of 0.9 (recall that we solved for a general conic), radius roughly 8% off, and orientation of the plane was 6^o off.

To summarize, both methods generally behave well in terms of convergence from reasonable intial guesses. Method II was much less sensitive to the initial guess (converged in all our experiments) and generally produced more accurate results.

5 Summary and Future Research

We have introduced a new approach for handling scenes with dynamically moving objects viewed by a monocular moving camera. In a general situation, when both the camera and the target are moving without any constrains, the problem is not solvable, i.e., one cannot recover the 3D position of the target even when the camera ego-motion is known. In previous work we have shown that by assuming that the target is moving along a straight 3D line the problem of recovering the target's trajectory is uniquely solved given at least five views of the moving target. In this paper we have extended the family of trajectories to include conic sections as well. In this context we have introduced two methods. The first method performs the optimization on some arbitrary virtual plane and is very simple. However, it can only deal with general conics only — a-priori constraints on the shape of the 3D conic cannot be enforced due to the projective distortion from the conic plane to the virtual common plane. The second method performs the optimization in 3D. The advantage of the second method is that under calibrated cameras it is possible to enforce a-priori constraints on the shape of the conic. For example, we have derived the equations necessary for recovering a 3D circular path.

We believe that future work on the family of trajectory triangulation tasks may include the following directions:

- Sliding-window linear or conic trajectory fitting. A reconstruction of a generally moving point can be decomposed onto smaller sub-problems in case many (dense) samples of the moving point are avilable (like in continous motion).

- A unification of static and dynamic reconstruction. It is possible to estimate whether a point is static or moving simply by the size of the kernel in the case of linear trajectory triangulation. A one-dimensional kernel corresponds to a striaght-line path, whereas higher dimensional kernels correspond to a single point (static situation).

- The possibility of recovering both the camera ego-motion and the trajectory (linear or conic) of the point. The task is of a multi-linear nature (for the linear trajectory triangulation) and thus there may be an elegant way of decoupling the system as is done in the static case.

- Handling more complex trajectories by tracking multiple points. If a sufficient number of points are tracked on a rigid body than the full motion of the object (relative to the camera ego-motion) can be recovered. It may be interesting to investigate the possible trajectory shapes when fewer points are available — such as two points.

References

[1] S. Avidan and A. Shashua. Trajectory triangulation of lines: Reconstruction of a 3D point moving along a line from a monocular image sequence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 1999.

[2] F.L. Bookstein. Fitting conic sections to scattered data. In *Computer Graphics and Image Processing*, pages (9):56–71, 1979.

[3] P.R. Escobal. *Methods of Orbit Determination.* Krieger Publishing Co., 1976.

[4] K. Kanatani. Statistical bias of conic fitting and renormalization. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 16(3):320–326, 1994.

[5] P. Meer and Y. Leedan. Estimation with bilinear constraints in computer vision. In *Proceedings of the International Conference on Computer Vision*, pages 733–738, Bombay, India, January 1998.

[6] Torr P.H.S., Zisserman A., and Murray D. Motion clustering using the trilinear constraint over three views. In *Workshop on Geometrical Modeling and Invariants for Computer Vision.* Xidian University Press., 1995.

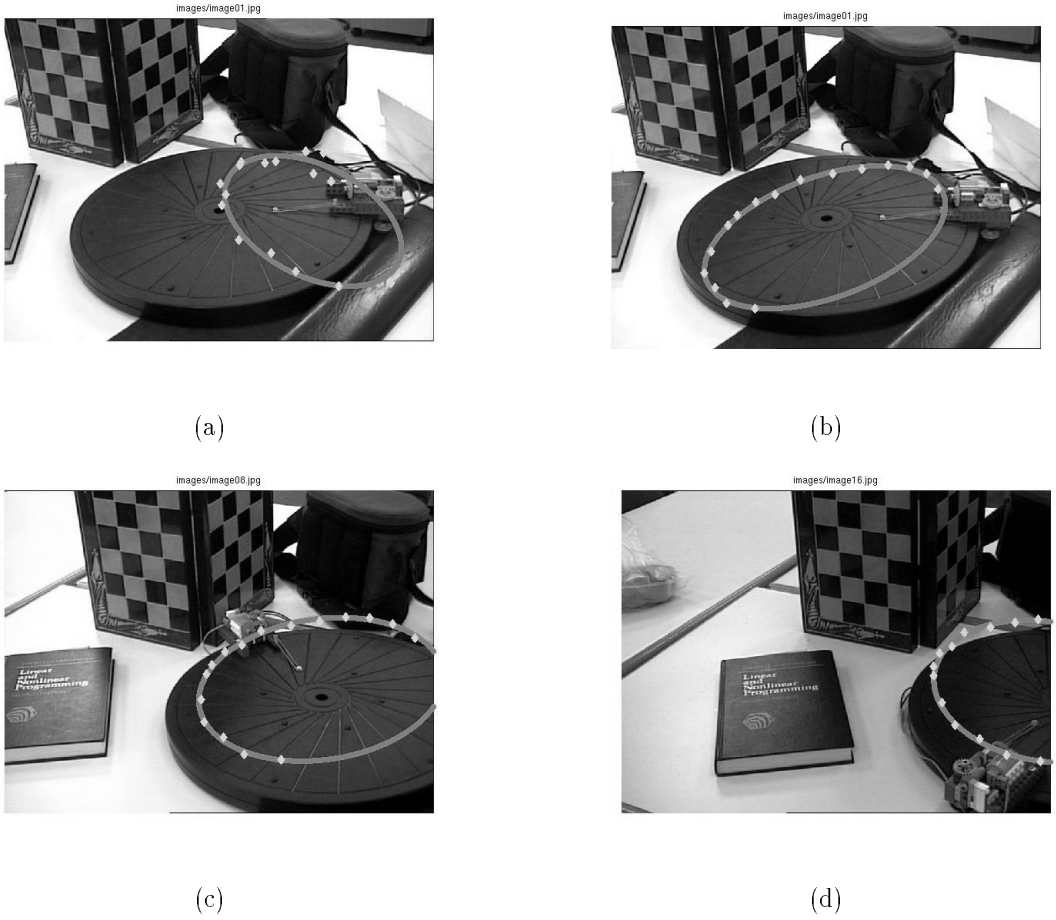

Figure 5: Using 2D conic fitting (method I) to recover the planar conic section. The results are shown by projecting the recovered planar conic (and the 3D points traced along the conic) on several reference images from the sequence. (a) shows the initial guess with the first image as the reference image. (b),(c), (d) shows the the results of the 2D conic fitting when the reference image is the first, middle and the last images of the sequence, respectively. The resulting conic had an aspect ratio of 0.9, radius roughly 8% off, and orientation of the plane was 6^{o} off.

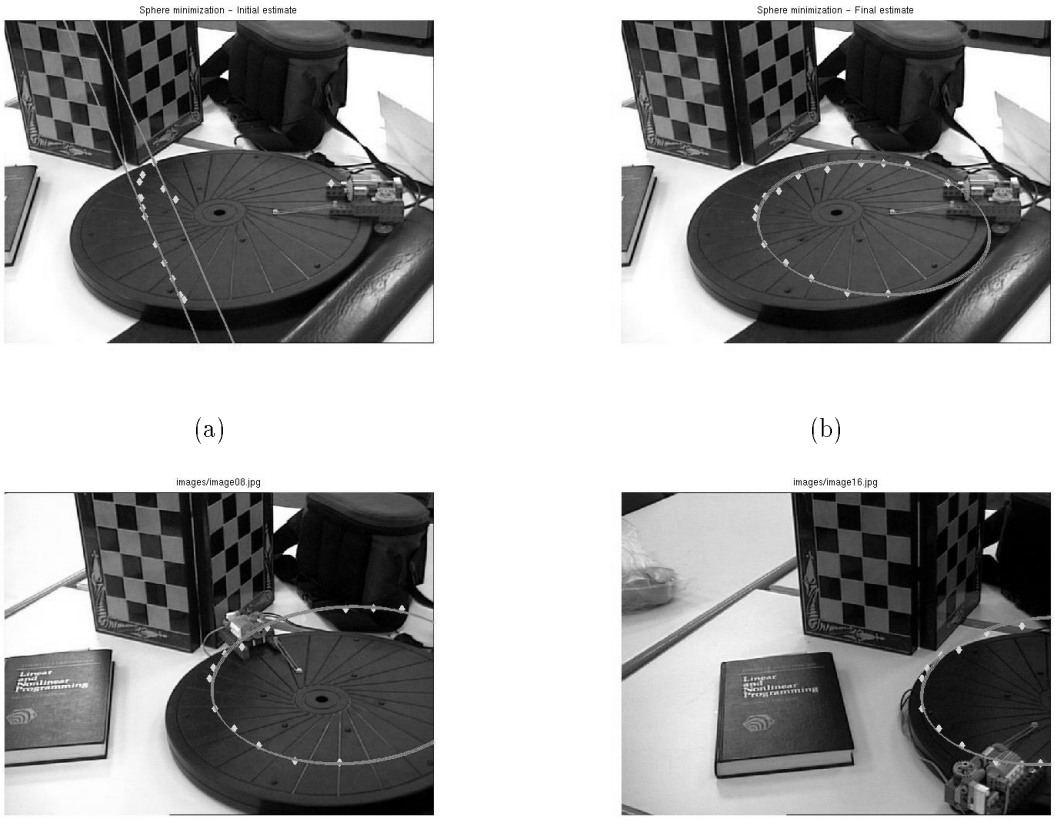

Figure 6: Using 3D sphere fitting (method II) to recover a planar conic section. The results are shown by projecting the recovered planar conic (and the 3D points traced along the conic) on several reference images from the sequence. (a) shows an extereme initial guess with the first image as the reference image. (b),(c), (d) shows the the results of the 3D sphere fitting when the reference image is the first, the middle and the last images of the sequence, respectively. The resulting radius of the circular path was 5% off from the ground truth, and around 4^o off in orientation.