Machine Vision and Applications (2009) 21:1–15
DOI 10.1007/s00138-008-0137-z

ORIGINAL PAPER

Global hand pose estimation by multiple camera ellipse tracking

Jorge Usabiaga · Ali Erol · George Bebis · Richard Boyle · Xander Twombly

Received: 30 August 2006 / Accepted: 11 February 2008 / Published online: 24 May 2008
© Springer-Verlag 2008

Abstract Immersive virtual environments with life-like interaction capabilities have very demanding requirements including high-precision motion capture and high-processing speed. These issues raise many challenges for computer vision-based motion estimation algorithms. In this study, we consider the problem of hand tracking using multiple cameras and estimating its 3D global pose (i.e., position and orientation of the palm). Our interest is in developing an accurate and robust algorithm to be employed in an immersive virtual training environment, called "Virtual GloveboX" (VGX) (Twombly et al. in J Syst Cybern Inf 2:30–34, 2005), which is currently under development at NASA Ames. In this context, we present a marker-based, hand tracking and 3D global pose estimation algorithm that operates in a controlled, multi-camera, environment built to track the user's hand inside VGX. The key idea of the proposed algorithm is tracking the 3D position and orientation of an elliptical marker placed on the dorsal part of the hand using model-based tracking approaches and active camera selection. It should be noted that, the use of markers is well justified in the context of our application since VGX naturally allows for the use of gloves without disrupting the fidelity of the interaction. Our experimental results and comparisons illustrate that the proposed approach is more accurate and robust than related approaches. A byproduct of our multi-camera ellipse tracking algorithm is that, with only minor modifications, the same algorithm can be used to automatically re-calibrate (i.e., fine-tune) the extrinsic parameters of a multi-camera system leading to more accurate pose estimates.

Keywords Virtual environments · Ellipse tracking · Model-based tracking · Hand pose estimation

J. Usabiaga · A. Erol · G. Bebis (✉)
Computer Vision Laboratory, University of Nevada,
Reno, NV 89557, USA
e-mail: bebis@cse.unr.edu

J. Usabiaga
e-mail: usabiaga@cse.unr.edu

A. Erol
e-mail: aerol@cse.unr.edu

R. Boyle · X. Twombly
BioVis Laboratory, NASA Ames Research Center,
Moffett Field, CA 94035, USA
e-mail: rboyle@mail.arc.nasa.gov

X. Twombly
e-mail: xtwombly@mail.arc.nasa.gov

1 Introduction

Virtual environments (VEs) should provide effective human computer interaction (HCI) for deployment in applications involving complex interaction tasks. In these applications, users should be supplied with sophisticated interfaces allowing them to navigate in the VE, select objects, and manipulate them. Implementing such interfaces raises challenging research issues including the issue of providing effective input/output. At the input level, new modalities are necessary to allow natural interaction based on direct sensing of the hands, eye-gaze, head or even the whole body.

Computer vision (CV) has a distinctive role as a direct sensing method because of its non-intrusive, non-contact nature; on the other hand, it is also facing various challenges in terms of precision, robustness and processing speed requirements. Various solutions have been proposed to support simple applications (i.e., no intricate object manipulation) based on gesture classification and rough estimates of almost rigid hand motion. However, systems that can support advanced VE applications with life-like interaction requirements have yet to come. Applications such as immersive

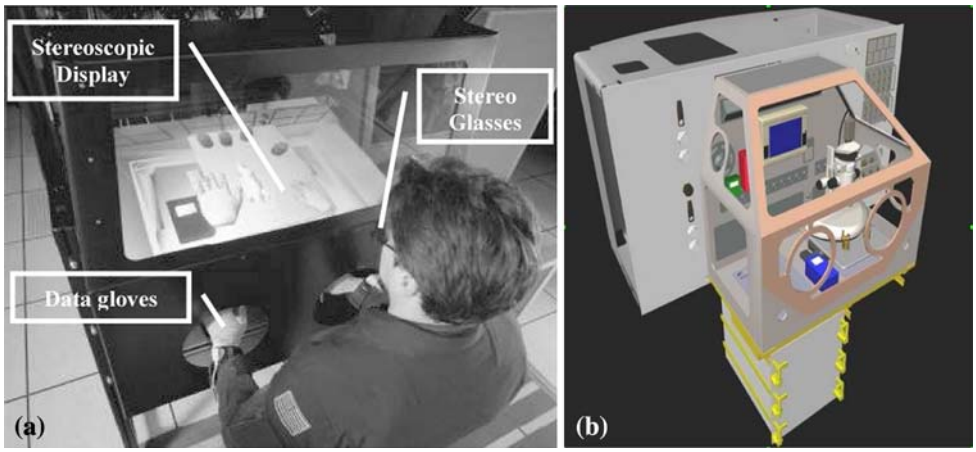

Fig. 1 **a** Virtual Glove Box: a stereoscopic display station provides a high-resolution immersive environment corresponding to a glovebox facility. The users interact with virtual objects wearing datagloves, **b** graphical model of the Glovebox which is available at the International Space Station

training or surgical simulations require very accurate and high frequency estimates of the 3D pose of the hand in a view independent fashion (i.e., the user need not even know where the cameras are located). Recovering the full degrees of freedom (DOF) hand motion from images with unavoidable self-occlusions is a very challenging and computationally intensive problem [2,3].

This study is part of an effort to improve the fidelity of interaction in an immersive virtual environment, called "Virtual GloveboX" (VGX) [1], which is currently under development at NASA Ames (see Fig. 1a). Our objective is to employ computer vision-based hand motion capture. VGX is being designed to assist in training astronauts to conduct technically challenging life-science experiments in a glovebox aboard the International Space Station (see Fig. 1b). It integrates high-fidelity graphics, force-feedback devices, and real-time computer simulation engines to achieve an immersive training environment.

The effectiveness of VGX as a training tool depends both on precision of the sensed motion and ease of use. The current interface of VGX uses off-the-shelf tracking and haptic feedback devices which contain cumbersome elements such as wired gloves, tethered magnetic trackers, and haptic armatures inside the workspace. All of these hinder the ease and naturalness with which the user can interact with the computer controlled environment and calibration of each measured degree of freedom is time consuming and imprecise. Further research is thus required to reduce the need for encumbered interface devices and increase the value of VGX as a training tool.

A fully generic unconstrained and precise solution to the hand pose estimation is not available yet. Existing unadorned hand tracking systems are mostly limited to a single camera and implicitly or explicitly accompanied with viewing constraints to minimize self-occlusions [2,3]. Obviously, such approaches are not acceptable in this and similar applications. Although some marker-based approaches are available, precision issues are often not addressed in these studies.

In this paper, we present a 3D global hand pose (i.e., position and orientation of the palm) estimation system using an elliptical marker placed on the dorsal surface of the palm. The system operates in a multi-camera environment built to track the user's hand inside the VGX. The use of markers is well justified in the context of our application since VGX naturally allows for the use of gloves without disrupting the fidelity of the interaction. Moreover, users are not looking at their hands during the simulation but at graphical hand models displayed in the virtual environment (see Fig. 1).

Estimating the global pose of the hand has several advantages. First, it reduces the dimensionality of hand pose estimation by 6 DOF. Second, it is a requirement for inverse kinematics-based methods. Finally, for some interfaces (e.g. navigation in VE), estimating the rigid motion of the hand is sufficient to generate control signals for the application. Our experimental results illustrate that the proposed approach is more accurate and robust than related approaches. A byproduct of our multi-camera ellipse tracking algorithm is that, with only minor modifications, the same algorithm can be used to automatically re-calibrate (i.e., fine-tune) the extrinsic parameters of the multi-camera system. In our case, camera re-calibration leads to improved hand pose estimates.

The rest of the paper is organized as follows: in the next Section, we present a brief review of previous work on marker-based hand pose estimation approaches. In Sect. 3, we describe the multiple camera environment used track the hand in the context of our application. In Sects. 5 and 6, we provide detailed descriptions of the multiple camera ellipse tracking algorithm and its application to camera re-calibration. Section 7 presents our experimental results and comparisons. Finally, Sect. 9 concludes this study.

2 Previous work

Marker-based hand tracking could be considered intrusive but on the other hand it has considerable technical advantages

in terms of processing speed and robustness. Therefore, there have been many attempts mostly using gloves painted with point markers [4–10].

Placing a number of point markers on the palm, fingertips and/or joints allows for simplified feature extraction methods and provides valuable information that can be used to estimate joint angles by solving an inverse kinematics problem. Lien at al. [7,10] used multiple cameras to extract the 3D locations of seven colored markers, two of which marks the wrist and the palm. Using colored markers greatly simplifies the task of finger and palm identification. Genetic algorithms (GAs) were employed for estimating the palm orientation. Once the global hand pose is estimated, it becomes possible to employ inverse kinematics solutions based on fingertip locations to estimate the joint angles. In a typical kinematic chain this is usually an ill-posed problem with multiple solutions. However, in case of the hand, the dependencies between neighboring joints helps to regularize the problem. Lien et al. [7,10] derived closed form solutions using regression techniques to map fingertip positions to joint angles. In a much older study, Lee et al. [8] presented a model fitting algorithm that gradually updates the joint angles based on the finger motion constraints to reach the extracted fingertip locations. Holden el al. [6] employed joint markers together with fingertip markers to fit the model directly to the point features in the images through a modified version of Gauss-Newton method [11]. Chua et al. [4] presented some closed form solutions to calculate the angles directly from 2D marker positions assuming orthographic projection.

The main problem with point markers is their susceptibility to occlusions and localization difficulties. Because of the proximity and flexibility of fingers, loosing some of the markers completely or collision of the markers on the image plane are very likely events that increase the complexity of tracking [6]. Using the predicted marker positions in place of the missing markers [6,10] is one partial remedy to deal with occlusions. However, localization problems may be more difficult to alleviate. When the hand is allowed to move in a relatively large area, it is not feasible to use point markers due to localization errors which affect the precision of pose estimates.

In the case of fingers, it is not possible to use other than point or line markers, which do not guarantee good precision and robustness due to practical resolution constraints and abundance of these features in images. The palm, however, is large enough allowing the use of more robust markers. Among them, conics have often proved to be good candidates due to several following reasons [12]. First, like points or straight lines, they are preserved under projective transforms. Second, conics are more compact primitives which contain global information of an object's pose. Finally, a conic can be represented by a symmetric matrix, which is easy to manipulate. In some cases, a closed-form solution [12,13] can be obtained, avoiding more expensive non-linear iterative techniques. To the best of our knowledge, Maggioni et al. [14] is the only study using conics, (i.e., two coplanar elliptical markers) for estimating global hand pose in 3D. Viewing the two ellipses from a single camera is sufficient to obtain the orientation and position of the palm.

3 Operational environment

The glovebox environment has some features that can be easily exploited by vision-based algorithms for hand tracking. First, the users are expected to wear gloves, which enables the use of markers naturally. Second, hand motion is restricted to a relatively small area inside the glovebox. This justifies the use of multiple cameras to deal with occlusions and controlled lighting along with uniform background to enable segmentation of the hands. Taking these facts into consideration, we have built a mock-up of VGX to perform our experiments as shown in Fig. 2.

Specifically, the VGX mock-up contains eight hardware-synchronized cameras located at the corners of the box, several fluorescent lights, and a white background to help segment the hands The intrinsic parameters of the cameras and radial distortion parameters were calibrated using Matlab's Calibration Toolbox [15]. To estimate the extrinsic camera parameters, Svoboda's [16] multiple camera self-calibration procedure that only needs a point light source as the calibration object was used.

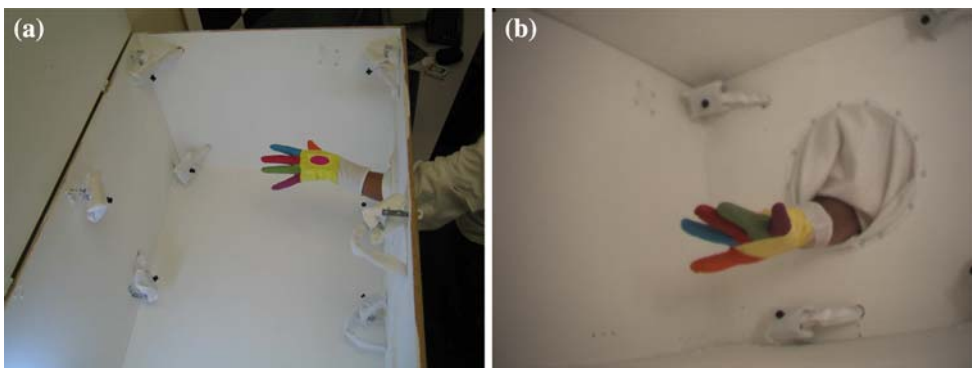

Fig. 2 The VGX mockup: **a** upside view showing eight cameras dedicated to capturing the motion of one hand **b** an image captured using one of the cameras

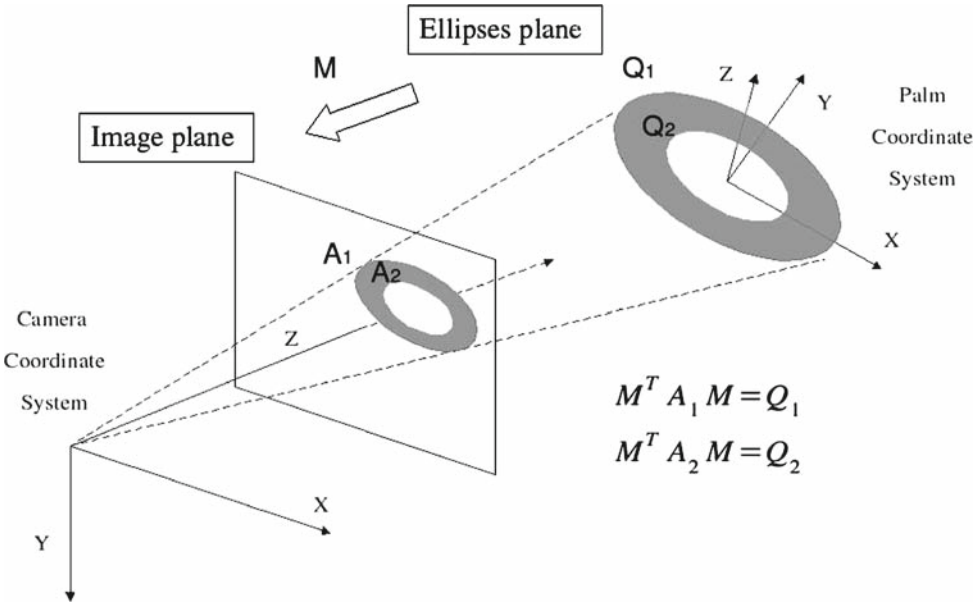

Fig. 3 Pose can be recovered from one image of two coplanar ellipses. A_1 and A_2 are the projections of the two coplanar ellipses Q_1 and Q_2. The transformation M between the image plane and the common plane of Q_1 and Q_2 determines the position and orientation of the common plane

During simulation, users wear a colored glove (see Fig. 2) with an elliptical marker placed on the dorsal part of the palm. To avoid perspective distortions due to bending, the elliptical marker was placed on a piece of cardboard which was then attached to the hand. In the future, we expect the colored fingers to help us divide the finger pose estimation problem into smaller pieces corresponding to each finger. The ellipse serves for estimating the global pose of the hand. In principle, it is possible to estimate the ellipse pose using two coplanar ellipses and single camera [12], however, resolution limitations combined with un-constrained hand motion makes it difficult to locate each ellipse separately. Therefore, we decided to use a single ellipse, which would need to be visible from at least two cameras for estimating its pose [13]. Camera placement in the VGX mock-up satisfies this visibility constraint.

4 Ellipse tracking and pose estimation

Ellipse tracking and pose estimation is a well studied topic. In this section, we provide a brief review of two well-known algorithms, the first based on a single camera and two co-planar ellipses [12] and the second based on a pair of cameras and a single ellipse [13]. Both algorithms reviewed in this section were used in our experimental comparisons while Quan's algorithm [13] was also used to initialize our multi-camera ellipse tracking and pose estimation system.

4.1 Pose estimation from two ellipses and a single camera

This algorithm, proposed by Ma [12], employs a single camera and two coplanar ellipses . In particular, Ma has shown that when only one camera is available, then two co-planar ellipses provide sufficient information for recovering the position and orientation of the common plane containing the ellipses (i.e., the problem becomes ill-posed when considering one ellipse only) . Using an invariant property between the two ellipses, Ma's solution is given in closed-form, making this approach very fast.

Given two coplanar conics $\boldsymbol{Q}_1$, and $\boldsymbol{Q}_2$, and their respective projections $\boldsymbol{A}_1$, and $\boldsymbol{A}_2$, the following relationship is satisfied:

$$\begin{aligned} \boldsymbol{M}^{\mathrm{T}} \boldsymbol{A}_1 \boldsymbol{M} &= \boldsymbol{Q}_1 \\ \boldsymbol{M}^{\mathrm{T}} \boldsymbol{A}_2 \boldsymbol{M} &= \boldsymbol{Q}_2 \end{aligned} \tag{1}$$

where $\boldsymbol{M}$ is the transformation between the camera and the conic plane that contains the pose information we are interested in recovering (see Fig. 3).

$\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$ are invertible in general, so Eq. 1 can be transformed into

$$\boldsymbol{M}^{\mathrm{T}} \boldsymbol{A}_1 \boldsymbol{M} \boldsymbol{Q}_1^{-1} = \boldsymbol{M}^{\mathrm{T}} \boldsymbol{A}_2 \boldsymbol{M} \boldsymbol{Q}_2^{-1} \tag{2}$$

and by multiplying by $\boldsymbol{M}^{-\mathrm{T}}$

$$\boldsymbol{A}_1 \boldsymbol{M} \boldsymbol{Q}_1^{-1} = \boldsymbol{A}_2 \boldsymbol{M} \boldsymbol{Q}_2^{-1} \tag{3}$$

Let us denote:

$$\boldsymbol{P}_a = \boldsymbol{A}_2^{-1} \boldsymbol{A}_1 \tag{4}$$

$$\boldsymbol{P}_q = \boldsymbol{Q}_2^{-1} \boldsymbol{Q}_1 \tag{5}$$

Combining together the last three equations, we have:

$$\boldsymbol{M}^{-1} \boldsymbol{P}_a \boldsymbol{M} = \boldsymbol{P}_q \tag{6}$$

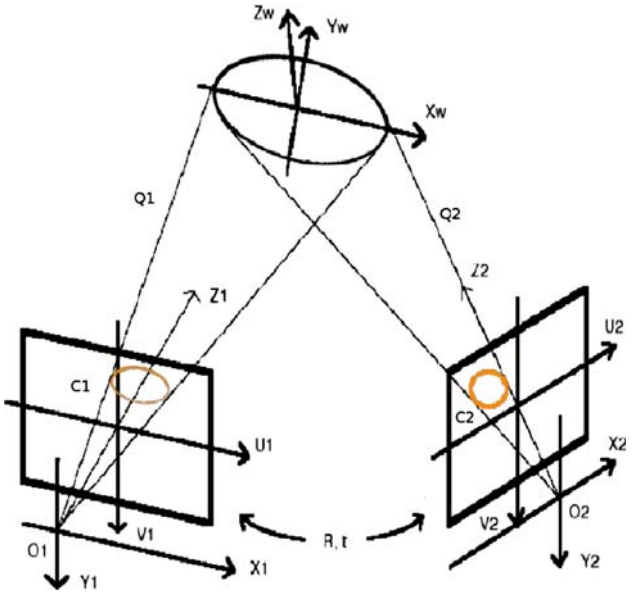

Fig. 4 One ellipse from two viewpoints problem. The original 3D ellipse projects onto C_1 and C_2. Q_1 and Q_2 are the cones that go from the center of projection of the cameras, O_1 and O_2, through C_1 and C_2 and the original ellipse

The trace of $\boldsymbol{P}_a$ is one of the invariants of a pair of conics if all the matrices are normalized such that their determinants are equal to 1. If $\boldsymbol{P}_a$ and $\boldsymbol{P}_q$ correspond to the same pair of conics, then they will have the same value. Therefore, Eqs. 4 and 5 can be employed to establish the correspondence between a pair of coplanar conics and their projections. From Eq. 6 we can recover $\boldsymbol{M}$ as shown in [12].

4.2 Pose estimation from a single ellipse and two cameras

This algorithm, proposed by Quan [13], deals with the problem of conic correspondences and reconstruction in 3D from two views using projective properties of quadric surfaces. A closed-form solution for both projective and Euclidean reconstruction of conics as well as a mechanism to select the correct corresponding ellipses in two views are described.

Given a corresponding pair of conics $\mathcal{C}_i$ from two views $i = 1, 2$ (see Fig. 4) the equation of the ellipses on the image coordinate systems $\boldsymbol{u}_i$ are given by

$$\mathcal{C}_i \equiv \boldsymbol{u}_i^{\mathrm{T}} \boldsymbol{C}_i \boldsymbol{u}_i = 0 \quad \text{for } i = 1, 2 \tag{7}$$

A conic in space is represented by the intersection of a quadric surface and a plane, so finding this plane is equivalent to reconstructing the conic in 3D because the intersection of one of the two cones associated with the 2D conics and the plane gives the reconstruction. Given the projection matrices $\boldsymbol{P}_i$ the cameras, which satisfies $\lambda \boldsymbol{u}_i = \boldsymbol{P}_i \boldsymbol{x}$, then the cones $\boldsymbol{Q}_i$ corresponding to the pair of 2D conics $\boldsymbol{C}_i$ in the two views are given by

$$\mathcal{Q}_i \equiv \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A}_i \boldsymbol{x} = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{P}_i^{\mathrm{T}} \boldsymbol{C}_i \boldsymbol{P}_i \boldsymbol{x} \quad \text{for } i = 1, 2 \tag{8}$$

in $\mathcal{P}^3$.

Quan considers the pencil of quadric surfaces $\mathcal{Q}_1 + \lambda \mathcal{Q}_2 = 0$, that represents a quadric surface which passes through all the common points of $\mathcal{Q}_1$ and $\mathcal{Q}_2$. He shows that the reconstruction of a conic in space from two views is equivalent to finding a λ such that the matrix $\boldsymbol{B}(\lambda) = \boldsymbol{A}_1 + \lambda \boldsymbol{A}_2$ has rank 2. Then $\boldsymbol{A}_i$ are proper cones and the plane we need can be recovered from $\boldsymbol{B}$.

5 Multiple-camera ellipse tracking and pose estimation

In our initial experiments, we found Quan's algorithm [13] to be very efficient, fast, and accurate. However, the use of multiple cameras was deemed necessary in our application to allow hand tracking independent of viewpoint. In our system, the marker could be visible from up to four cameras. Although not all of the cameras would contain reliable information for pose estimation (i.e., see Section 5.2), it would be possible in general to use information from more than two cameras to improve pose estimation and robustness. Therefore, we have developed a model-based ellipse tracking approach that integrates information from any number of cameras.

A block diagram of a generic model-based tracking system, which is common in many studies for tracking various types of objects in 2D or 3D [11,17–19], is shown in Fig. 5. The approach requires a parametric model of the object to be constructed first. The main function of the model is to define a mapping from the parameter space (i.e., the pose of the object) to the feature space. At each frame of the input image sequence(s), a search initiated by a prediction is executed to find the pose of the model that best matches the features extracted from the input. Basically, a matching error, which is a measure of similarity between groups of model features and groups of features extracted from the input images, is minimized. The synthesis of features using the model on the image plane also enables a selective analysis that focuses on regions or a subset of features instead of the whole input image. When multiple cameras are used, the matching error on all the cameras can be combined without solving any correspondence problem between the images [18]. In the first frame, a prediction is not available, therefore, a separate initialization procedure is needed.

In our case, the model to track is an ellipse, which is represented as a set of uniformly sampled points on its boundary. We use Quan's ellipse pose estimation algorithm [13] for initialization purposes. The prediction is taken to be the pose estimate at the previous frame. Employing higher order

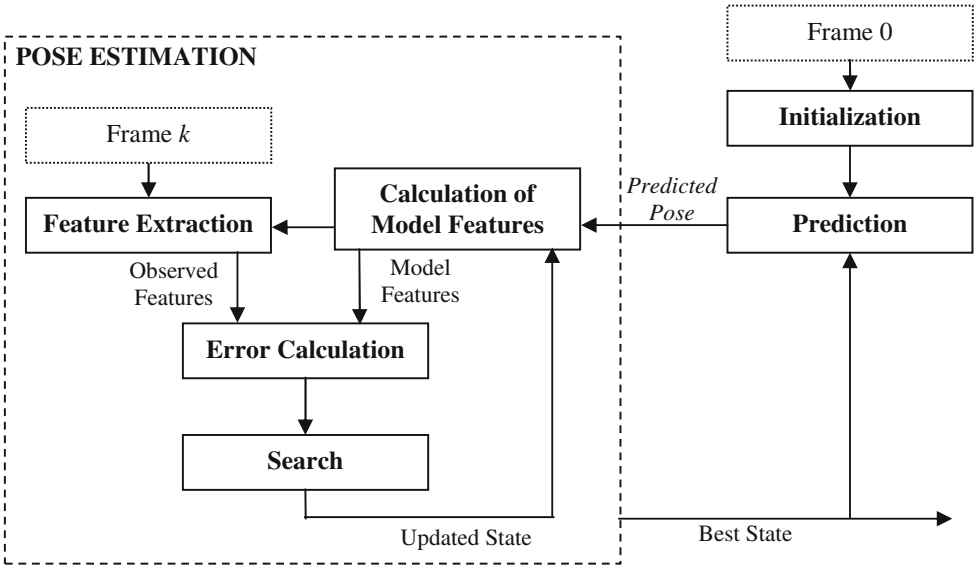

Fig. 5 Model-based tracking

dynamic models did not provide any improvement on the performance of the system. There are many different ways to conduct the search or equivalently minimize the matching error. Here, we present an algorithm based on Lowe's model-based pose estimation algorithm [11]. Specifically, our system includes the following four processing steps:

(1) Initialization, where the pose of the ellipse in the first frame is estimated.
(2) Active camera selection, where the best cameras for pose estimation are determined.
(3) Matching error computation, where the similarity between the images and the projection of the ellipse at a given pose is computed.
(4) Pose estimation, where the pose parameters that minimizes the matching error between the projected model ellipse and the image features are computed.

5.1 Initialization

We employed Quan's algorithm [13], which deals with the problem of conic correspondences and reconstruction of conics in space from two views. In the initialization module, we assume that the ellipse is visible from two designated cameras. These images are first processed to extract edges using Canny's edge detector [20] (See Fig. 6a), then contours are extracted and finally an ellipse is fit to each contour using *Direct Least Squares Ellipse Fitting* algorithm provided in [21] (see Fig. 6b,c). As its is demonstrated in Fig. 6, various imperfections in edge extraction process do not allow to extract a single ellipse contour from the image. However the correspondence constraints allow us to detect the correct pair of ellipses by evaluating all possible pairs of correspondence hypotheses. Once the correct pair is found, closed form solutions provided by Quan are applied to determine the pose of the ellipse in 3D.

For comparison purposes, we extended this initialization algorithm for processing an image sequence through active camera selection algorithm presented in Section 5.2. At each frame, two best views seeing the ellipse are first selected based on the predicted pose of the ellipse. Then the processing steps described above are applied to estimate the ellipse pose. Because of prediction it becomes possible to reduce the amount of image processing computations and number of correspondence hypotheses by determining a region of interest in the images based on the expected size of the ellipse (see Fig. 9).

5.2 Active camera selection

We use a number of criteria to select the "best" cameras for pose estimation. First, we select only those cameras that provide us with images of the ellipse at a satisfactory resolution. If the ellipse is too far away, large changes in its pose will only cause small image displacements. The criterion used to test this constraint is the area covered by the ellipse in the image. Second, we try to avoid selecting cameras that that does not see the ellipse at a vertical angle. The angle between the normal to the ellipse and the vector that goes from the center of the camera to the center of the ellipse is used to determine such cases. Finally, we do not consider cameras that provide images where the ellipse is completely or partially occluded. The cameras that passes the above tests contributes to the tracking.

5.3 Matching error computation

The matching error on the images is computed using point to contour distance measure (See Fig. 7). For each active

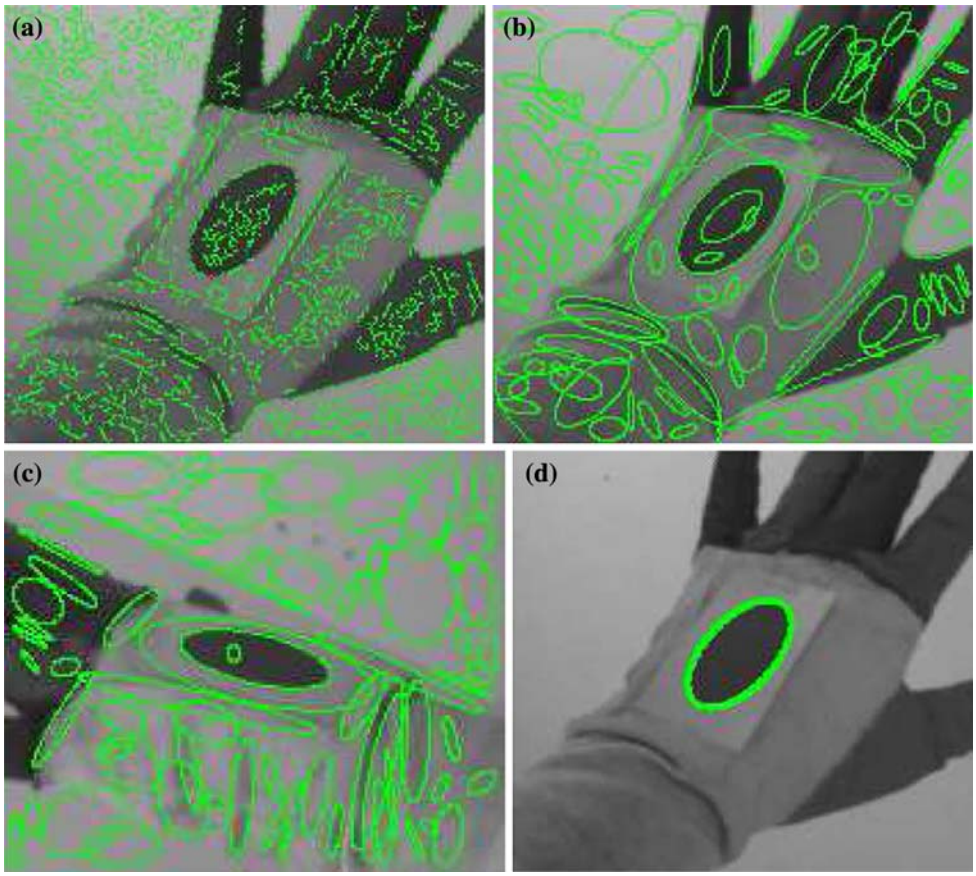

Fig. 6 **a** Canny edge detector output on one view, **b** detected ellipses on the same view, **c** detected ellipses on a different view, **d** final detected ellipse using the true correspondence

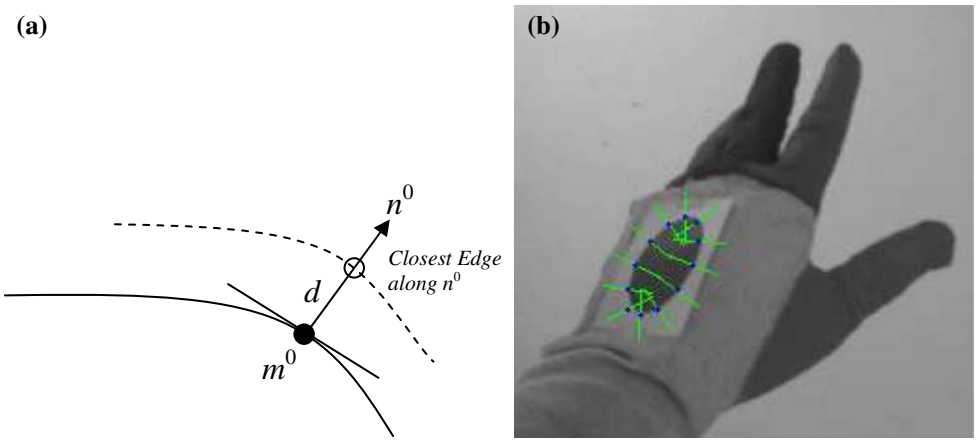

Fig. 7 Point-to-contour distance calculation: **a** m^0 is a sample point on the projection of the model, n^0 is the normal to the contour for that point, and d is the error distance, **b** the normals of the projected contour (12 sample points are used) overlaid on an input image

camera i, an error vector $\boldsymbol{e}^i$ is computed by first projecting the uniformly sampled m points onto the camera's image plane using the current pose of the ellipse, and then searching for the closest edge (i.e., the closest gradient local maximum) along the normal to the projected contour at the sampled points.

The errors of all the points are concatenated to form a vector:

$$\boldsymbol{e}^i = \left[e_1^i, \ldots e_m^i\right]^{\mathrm{T}} \tag{9}$$

where i denotes camera i. It should be noted that, a large number of sample points m would provide a better estimate; however, it would also slow down the system significantly.

Finally a total error vector $\boldsymbol{e}$ is obtained by weighting and concatenating the error vectors (see Eq. 9) of the active cameras given by:

$$\boldsymbol{e} = \left[w^1\boldsymbol{e}^1, \ldots, w^n\boldsymbol{e}^n\right]^{\mathrm{T}} \tag{10}$$

where the weights w^i are calculated as a combination of (1) calibration error (i.e., the larger the calibration error the smaller the weight), and (2) area (i.e., the larger the area covered by the ellipse on this camera's image the larger the weight). Weighting mainly helps to reduce the number of iterations required by the optimization algorithm to converge. We did not observe any improvement in estimation accuracy compared to the non-weighted version of the total error vector.

5.4 Pose estimation

Pose estimation corresponds to finding the pose parameters $\boldsymbol{T}$ (i.e., position and orientation of the ellipse) that minimize the matching error given by the magnitude of the total error vector in Eq. 10. Many studies [11,19] employ Newton's method, which subtracts a vector of corrections, $\boldsymbol{x}$ from the current estimate pose $\boldsymbol{T}$ at each iteration. If $\boldsymbol{T}^k$ is the parameter vector corresponding to iteration k, then,

$$\boldsymbol{T}^{k+1} = \boldsymbol{T}^k - \boldsymbol{x} \tag{11}$$

By linearizing the system at the current estimate, the correction vector is calculated by solving an over-determined system of equations

$$\boldsymbol{e} = \boldsymbol{J}\boldsymbol{x} \tag{12}$$

where $\boldsymbol{J} = [J_{ij}] = [\frac{\partial e_i}{\partial x_j}]$ is the Jacobian.

One important issue in a general problem is the singularities of the Jacobian, where the change in some pose parameters do not produce a change in object's appearance. Lowe [11] introduces a technique to provide stability in the presence of singularities. Another method to eliminate singularities is the use of multiple views [19]. In our case, due to the availability of multiple views, we do not employ any stabilization mechanism.

5.5 Algorithm overview

The main steps of the multi-camera ellipse tracking algorithm can be summarized as follows:

(1) Predict 3D pose of the ellipse for current frame.
(2) Choose the *active cameras*.
(3) Compute error distance vectors and Jacobian matrices for all n active cameras.
(4) Run a nonlinear minimization algorithm to recover the ellipse's pose that best minimizes the residual error of Eq. 12.
(5) Update current pose estimation.

6 Extrinsic parameters re-calibration

Multiple camera calibration assuming an arbitrary camera configuration is a difficult problem. Svoboda's [16] approach provides a relatively practical solution. Instead of a complex calibration pattern, it uses a colored light source (e.g., a small LED in our case), which is visible by many cameras simultaneously. Calibration is performed by moving the light source arbitrarily inside the area covered by the cameras. The trajectory of the light source as perceived from different cameras provides the necessary information for calibration purposes. However, this process is rather slow, it requires some user interaction, and it does not always guarantee good results since it depends on how well the trajectory of the light source covers the area enclosed by the cameras. Another practical reason for the re-calibation was to compensate for the accidental motion of the cameras, which is quite likely due to the moving hand inside the box.

To update and further optimize the extrinsic camera parameters, we have employed our ellipse tracking algorithm. Given a training sequence of ellipse motion, we first estimate the pose of the ellipse at each frame using our tracking algorithm,which basically minimizes the matching error on all the active cameras. In the second step, we use the pose estimates as the ground truth and find the best extrinsic parameters that minimizes matching error. Repetitive application of these two steps results in a more extrinsic parameters that yields smaller matching error and more accurate estimates. More specifically, the re-calibration algorithm works as follows:

(1) **For all frames**, run the tracking algorithm and record (i) the pose of the ellipse and (ii) which cameras are active for each frame (see Table 1, top)
(2) **For each camera**,
   (a) Load the poses, images, and frames where this camera was active; we will be referring to these frames as **active** frames (see Table 1 bottom).
   (b) Minimize the total projection error on all active frames corresponding to the camera. The total projection error is given by given by

$$e^i = \sum_{k=1}^{p} \sum_{j=1}^{m} \mid e^i_{kj}(\boldsymbol{x}^i) \mid \tag{13}$$

where p denotes the number of active frames and m denotes number of points in the ellipse model. e^i_{kj} is the point to contour distance at frame k for point j as explained in Sect. 5.3 and is a function of extrinsic camera parameters $\boldsymbol{x}^i$. We employ Nelder–Mead's Simplex algorithm [22] to find the extrinsic parameters that minimizes e^i.

Table 1 Sample re-calibration for a sequence with 999 frames

1- Run ellipse tracking for whole training sequence

Frame	Camera					Ellipse pose	
⇓	0	1	…	7		Orientation	Position
1	I	I	…	A	→	$\boldsymbol{\theta}_1$	$\boldsymbol{p}_1$
2	I	I	…	A	→	$\boldsymbol{\theta}_2$	$\boldsymbol{p}_2$
⋮	⋮				→	⋮	⋮
997	A	A	…	I	→	$\boldsymbol{\theta}_{997}$	$\boldsymbol{p}_{997}$
998	A	A	…	I	→	$\boldsymbol{\theta}_{998}$	$\boldsymbol{p}_{998}$
999	A	A	…	I	→	$\boldsymbol{\theta}_{999}$	$\boldsymbol{p}_{999}$
Total	300	351	…	415			

2- Run re-calibration for all cameras

Camera	Frame					Extrinsic Parameters	
⇓	0	1	.. .	999		Rotation	Translation
0	I	I		A	→	$\boldsymbol{R}_0$	$\boldsymbol{t}_0$
1	I	I		A	→	$\boldsymbol{R}_1$	$\boldsymbol{t}_1$
2	I	I		I	→	$\boldsymbol{R}_2$	$\boldsymbol{t}_2$
3	I	I	.. .	I	→	$\boldsymbol{R}_3$	$\boldsymbol{t}_3$
4	A	A		I	→	$\boldsymbol{R}_4$	$\boldsymbol{t}_4$
5	I	I		A	→	$\boldsymbol{R}_5$	$\boldsymbol{t}_5$
6	I	I		A	→	$\boldsymbol{R}_6$	$\boldsymbol{t}_6$
7	A	A		I	→	$\boldsymbol{R}_7$	$\boldsymbol{t}_7$

Top: Recover the ellipse pose for all frames. Bottom: Run re-calibration for all cameras A = Active camera/frame, I = Inactive camera/frame

(3) If the error for all the cameras is less than a threshold or a maximum number of iterations has been reached exit, otherwise go to step 1.

7 Experimental results

In this section, we present quantitative and visual experimental results to evaluate the pose estimation and re-calibration algorithms. In all the experiments, we assumed that the ellipse was placed flat on the dorsal part of the hand (see Fig. 11).

7.1 Accuracy

To evaluate the accuracy of pose estimation, first we compared our algorithm with Ma's algorithm [12]. Specifically, using two co-planar ellipses, we recovered the pose of the largest ellipse using one camera. To make the comparison fair, we used our active camera selection algorithm to choose the best camera to reconstruct the pose of the largest ellipse. To evaluate the accuracy of this method, we re-projected the ellipse not only on the camera used to recover its pose but also several other cameras, each having a different viewpoint. Figure 8 shows the re-projection results where the low-right image corresponds to the camera used to recover the pose of the ellipse. Obviously, Ma's method does not produce very accurate pose estimation results, therefore, it was excluded from further comparisons.

Then we compared our algorithm with Quan's algorithm, which is among the best available algorithms for a two camera system. We used the extension of the initialization module with camera selection described in Section 5.1 to employ Quan's algorithm over an image sequence. We ran both of the algorithms over the same sequence.

(1) *Re-projection error*: The re-projection error is a measure of the image displacement between the projection of the estimated pose and the actual ellipse in the image. We use the matching error vector given in Eq. 10 without the weights as the re-projection error.

The two camera algorithm always uses the best two cameras to recover the pose (see Fig. 9), while the multiple-camera algorithm uses all *active cameras*. We compute the re-projection for both methods on all active cameras.

Figure 10 shows the re-projection error (i.e., matching error given in Sect. 5.3) for both algorithms. The square wave shaped curve on the top portion of the graph indicates the number of active cameras at each frame. Two interesting observations can be made:

(1) When only two cameras are active in the case of the multiple-camera algorithm, both algorithms give very close results. However, when more than two cameras are active, the performance of the multiple-camera algorithm is significantly better.
(2) Although the re-projection error is smaller in the multiple-camera case, it increases with the number of cameras. The reason is that there are more calibration errors involved as the number of cameras increases.

(2) *Difference in position estimates*: Figure 11a shows the differences in the position estimates of the two algorithms. Interestingly enough, these differences resemble the differences in the re-projection error shown in Fig. 10. Overall, we can conclude that when both algorithms use the same two cameras, the results are very similar, however, when more cameras are available, the multiple-camera approach yields more accurate position estimates which implies lower re-projection error. Similar observations can be made for the orientation estimates of the ellipse. Figure 11b shows some examples to demonstrate the multiple-camera algorithm in the case of three active cameras. The re-projection on the inactive bottom left camera demonstrates the accuracy of the estimates.

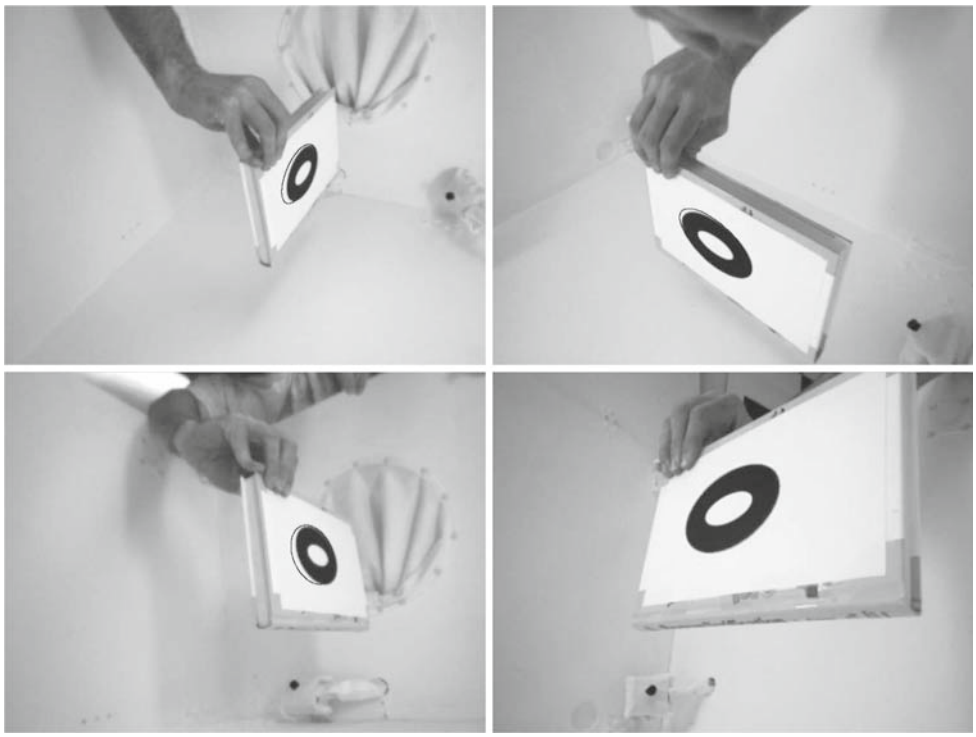

Fig. 8 Illustration of Ma's algorithm. The position and orientation of the big ellipse was recovered using the *lower-right image*. The figures on the *top* and *lower-left* show the re-projection of the ellipse in three other cameras. Obviously, Ma's algorithm produces large re-projection errors

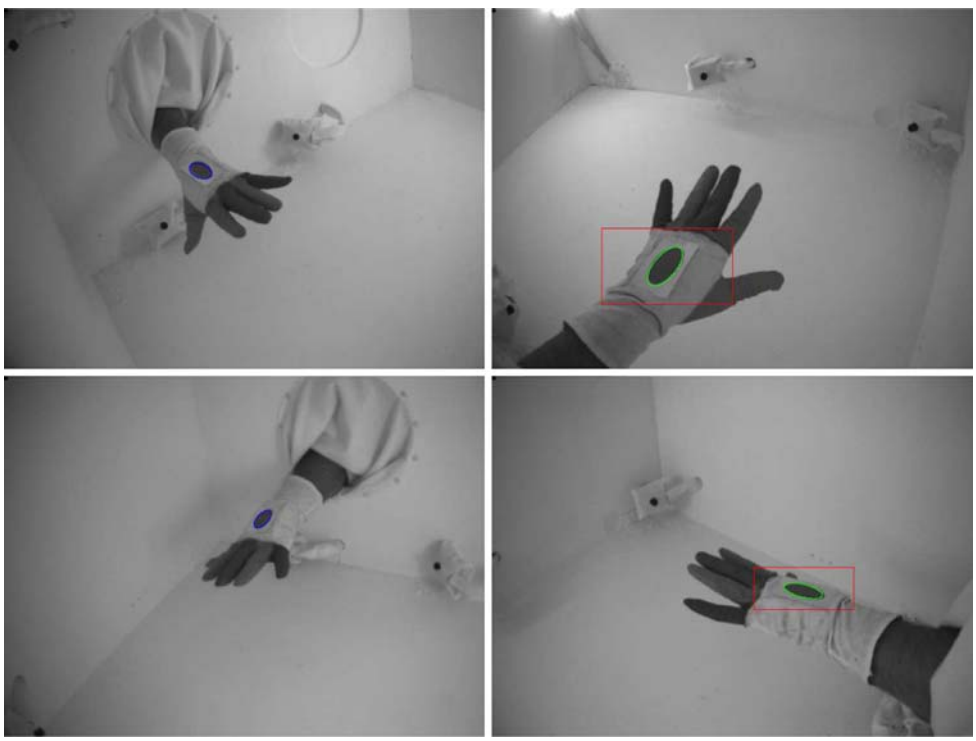

Fig. 9 Illustration of Quan's algorithm. All four cameras are active. Cameras on the right hand side containing green ellipse are the best two views. The red rectangle shows the region of interest, in which we search the ellipse. Blue ellipses on the left two images are re-projections of the final result

(3) *Difference in orientation estimates*: As with the position (Fig. 12), we plot the difference in orientation estimates. Unlike the position, the differences in the orientation do not exhibit an obvious relationship with the re-projection error or the number of active cameras.

8 Robustness

The multiple-camera algorithm has the upper-hand over the two camera algorithm when it comes to robustness. This is due to the fact that, as explained before, the two camera

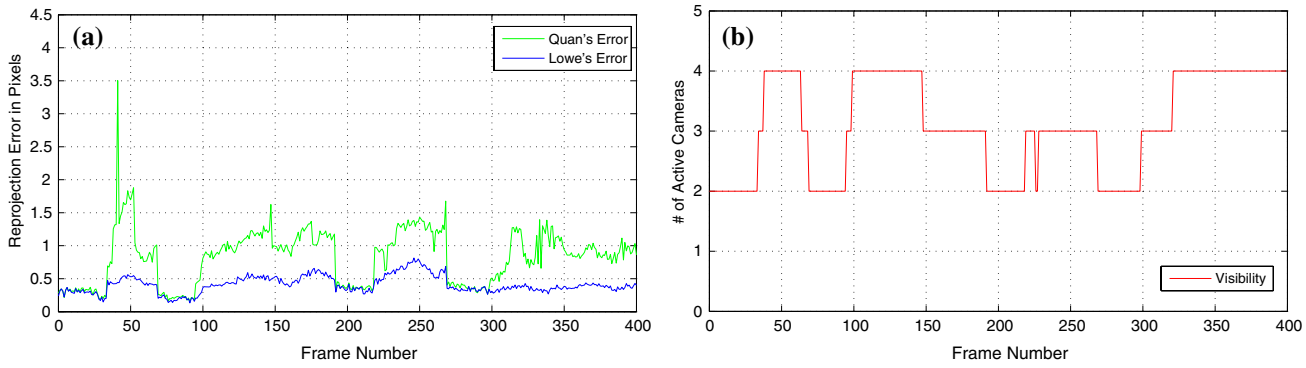

Fig. 10 **a** Re-projection error of Quan's algorithm and our algorithm, **b** number of active cameras for our multiple-camera approach

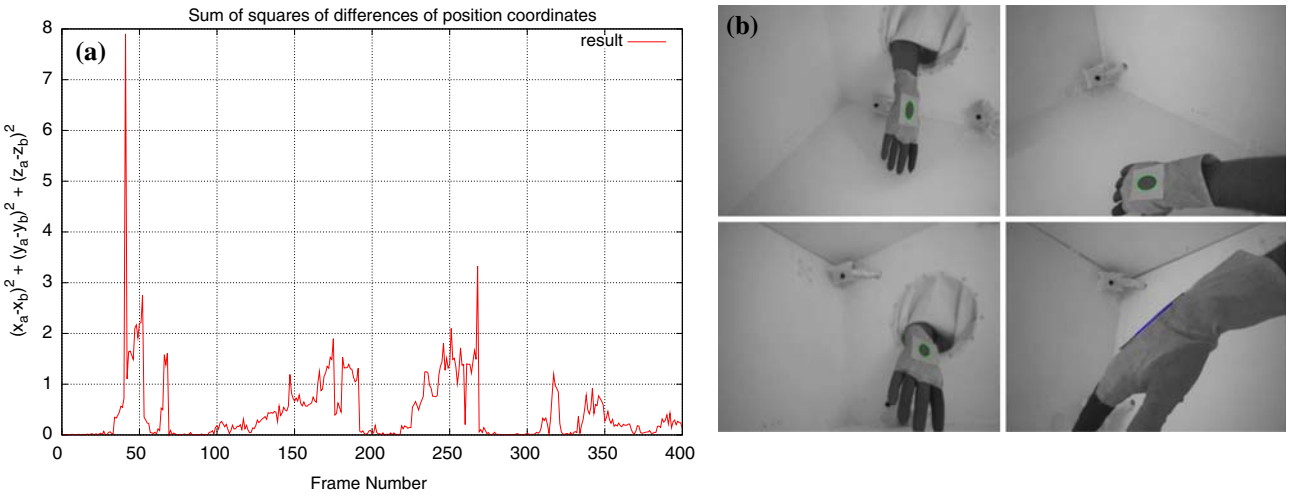

Fig. 11 **a** Sum of squares differences of position coordinates between two and multiple-camera algorithms, **b** re-projection of the ellipse on the input images for frame 42 of a sequence. The images where the re-projected ellipse is drawn in green correspond to the active cameras

algorithm takes as input the 2D image equation of the best two ellipses, which requires going through edge detection and ellipse fitting procedures. Thus, in the cases when we do not have two good input images at our disposal, it is very likely that the system will break down in one of these steps.

Figure 13 shows an example frame when occlusion prevents the two cameras closest to the ellipse from being used. The two camera algorithm has to resort to the other two cameras shown, which do not provide enough resolution of the ellipse to provide a correct 2D ellipse fit. The multiple-camera algorithm, on the other hand, only needs an image error computed from the gradient in the image, being consequently more robust in these cases and thus able to endure poor quality input until the ellipse moves back into a better region of the box.

The need for low-level image processing steps also brings another weakness. Edge detection and contour extraction need several thresholds, which are highly dependent on the lighting. Unfortunately, this requires the fine-tuning of several parameters depending on a given setup. The multiple-camera algorithm is free of these restrictions, which makes it more portable.

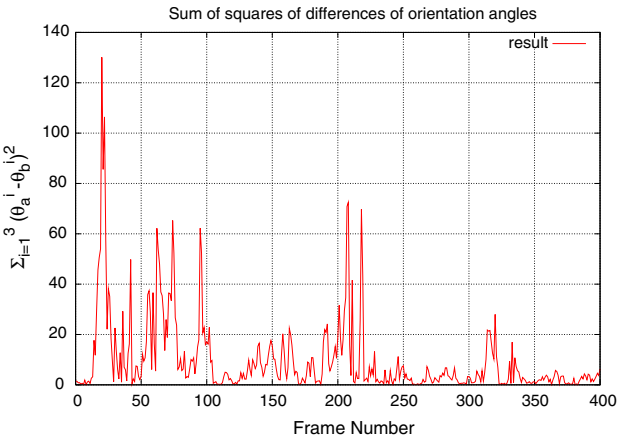

Fig. 12 Sum of squares differences of orientation coordinates between two and multiple-camera algorithms

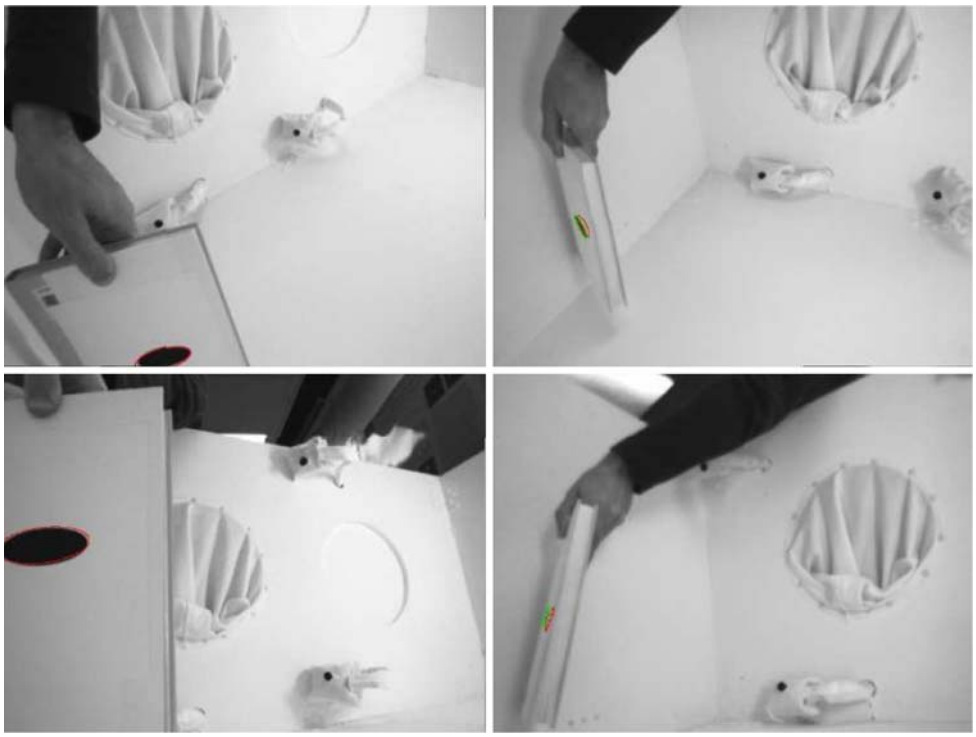

Fig. 13 An example frame where occlusion prevents from using the two views with largest projected ellipses (i.e., highest resolution). The multiple-camera algorithm is able to continue tracking by using the two images on the right. The *red ellipse* shows the prediction for that frame; the *green* one shows the recovered pose from the cameras used

8.1 Processing speed

A disadvantage of the multiple camera tracking system is the higher computational requirements due to its iterative nature. Quan's algorithm processes each frame in about 4 ms. The multiple camera algorithm deals with more cameras and computational cost depends linearly on the number of sampled points of the ellipse used for re-projection. Using a rather conservative number of samples (100) and un-optimized code, the processing speed was about 150 ms per frame. The most expensive part of the algorithm is the matching error calculation step, which is repeated a few times at each frame.

8.2 Effects of re-calibration

The results of the multiple-camera algorithm on a sequence taken in the VGX were used to re-calibrate the extrinsic camera parameters. To asses the effects of re-calibration, the modified extrinsic parameters were used to estimate the pose parameters on a test sequence. Figure 14 shows the re-projection errors obtained with and without re-calibration assuming different number of iterations. As it can be observed, lack of re-calibration increases the re-projection error as number of active cameras increases. However, re-calibration reduces the dependency of the re-projection error on the number of active cameras with each iteration, to the point where it is almost constant.

9 Conclusion and future work

We have presented a multiple camera, model-based ellipse tracking algorithm for global hand-pose estimation in an immersive training environment. We have also shown how to employ the proposed algorithm for re-calibrating the extrinsic parameters of a multi-camera system. Our experimental results and comparisons illustrate the effectiveness of the proposed approach both in terms of pose estimation and re-calibration. Our algorithm was compared with two well know algorithms, Ma's algorithm which is based on tracking a pair of co-planar ellipses using one camera [12], and Quan's algorithm which is based on tracking a single ellipse using a pair of cameras [13]. Both algorithms were successfully ported in our, eight camera, environment by employing an efficient camera selection scheme.

Ma's algorithm has several desirable properties since it only requires that the ellipse is visible from one camera, it offers a way to solve the correspondence problem and it provides a closed-form solution. However, the recovered pose is quite biased towards the camera used, and it is not accurate in a global sense. This makes transitions from one camera to another less robust, and the accuracy is not high enough for our application.

Quan's algorithm is is much more accurate than Ma's algorithm while having the same advantages as Ma's algorithm (i.e., it offers a way to solve the correspondence problem and it provides a closed-form solution). However, this algorithm

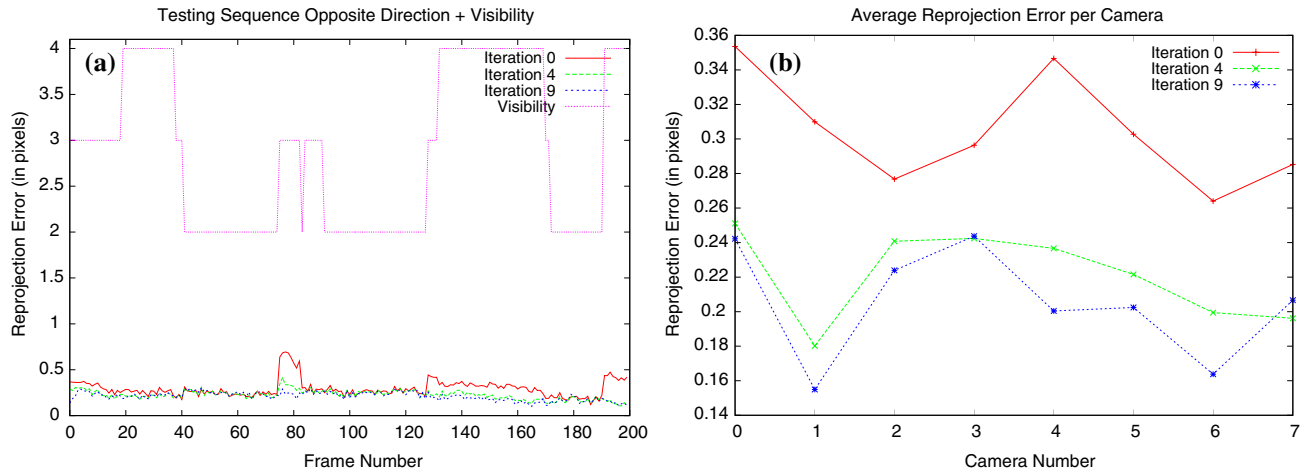

Fig. 14 The re-projection error on a testing sequence with and without re-calibration: **a** average re-projection error per frame for the testing sequence computed with the extrinsic parameters iterations 0, 4 and 9 (note that the number of active cameras is plotted in the same graph with the re-projection error to better illustrate how re-calibration improves results); **b** average error of the whole sequence per camera

is highly dependent on the image processing steps required to recover the 2D equation of the projected ellipse (i.e., edge detection, contour extraction and ellipse fitting). This makes it less robust in cases where there is not a good quality pair images (e.g., due to partial occlusions). Moreover, it requires fine-tuning several thresholds which dependent on the given environment.

Our multiple-camera algorithm was shown to be the most accurate of all. It can cope better with calibration errors since it utilizes more cameras. It is also more robust to illumination changes and lower quality images due to partial occlusions. It does not require using as many thresholds as in Quan's algorithm, since there are no image processing steps involved (i.e., the error is computed directly from the intensity gradient). However, like with Quan's, our algorithm requires at least two images. Although it is slower than Quan's algorithm, with appropriate optimizations it would be able to run in real-time.

For future work, we plan to improve the processing speed of our algorithm. As indicated in the previous section, using a rather conservative number of ellipse point samples (100) and un-optimized code, the processing speed achieved was about 150 ms per frame. However, higher speeds would be necessary in order to successfully deploy the proposed approach in the VGX environment which is aimed for real-time use. There are several ways to improve speed. First, optimizing our code would certainly improve processing speed. Second, computing an approximation of the matching error, which is the most expensive part of our algorithm, instead of computing it exactly would offer significant savings. Finally, running the algorithm on a faster machine or even on a multi-processor machine (e.g., dual or quad processor) would offer significant speed-ups. In addition to improving the speed of the system, we plan to extend the it by estimating the full DOF of the hand. Recovering the global pose of the hand is an important step in this process.

Acknowledgements This work was supported by NASA under grant # NCC5-583

References

1. Twombly, A., Smith, J., Montgomery, K., Boyle, R.D.: The virtual glovebox (vgx): a semi-immersive virtual environment for training astronauts in life science experiments. J. Syst. Cybern. Info. **2**, 30–34 (2005)
2. Pavlovic, V.I., Sharma, R., Huang, T.S.: Visual interpretation of hand gestures for human-computer interaction: A review. IEEE Trans. Pattern Anal. Mach. Intell. **19**(7), 677–695 (1997)
3. Erol, A., Bebis, G., Nicolescu, M., Boyle, R.D., Twombly, A.: Vision-based hand pose estimation: a review. Comp. Vis. Image Understanding, **108**(1-2), 52–73 (2007)
4. Chua, C.S., Guan, H.Y., Ho, Y.K.: Model-based finger posture estimation. IEEE Asian Comp. Vis. Conf. pp 43–48 (2000)
5. Chua, C.S., Guan, H., Ho, Y.K.: Model-based 3d hand posture estimation from a single 2d image. Image Vis. Comput. **20**(3), 191–202 (2002)
6. Holden, E.: Visual Recognition of Hand Motion. PhD Thesis, Department of Computer Science, University of Western Australia (1997)
7. Lien, C.C., Huang, C.L.: Model based articulated hand motion tracking for gesture recognition. Image Vis. Comput. **16**, 121–134 (1998)
8. Lee, J., Kunii, T.: Constraint-based hand animation. In: Models and Techniques in Computer Animation pp. 110–127. Springer, Tokyo, 1993
9. Kim, H., Fellner, D.W.: Interaction with hand gesture for a back-projection wall. Comp. Graph. Int. pp. 395–402, 2004.
10. Lien, C.C.: A scalable model-based hand posture analysis system. Mach. Vis. Appl. **16**(3), 157–169 (2005)
11. Lowe, D.G.: Fitting parameterized three-dimensional models to images. IEEE Trans. Pattern Anal. Mach. Intell. **13**(5), 441–450 (1991)
12. Ma, S.D.: Conics-based stereo, motion estimation and pose determination. Int. J. Comp. Vis. **10**(1), 7–25 (1993)

13. Quan, L.: Conic reconstruction and correspondence from two views. IEEE Trans. Pattern Anal. Mach. Intell. **18**(2), 151–160 (1996)
14. Maggioni, C., Kammerer, B.: Gesturecomputer - history, design and applications. In: Computer Vision for Human-Machine Interaction, pp. 312–325. Cambridge University, Cambridge (1998)
15. Bouguet, J.-Y.: Camera calibration toolbox for matlab.
16. Svoboda, T., Martinec, D., Pajdla, T.: A convenient multi-camera self-calibration for virtual environments. PRESENCE: Teleoperators Virt. Environ. **14**(4), 407–422 (2005)
17. O'Rourke, J., Badler, N.I.: Model-based image analysis of human motion using constraint propagation. IEEE Trans. Pattern Anal. Mach. Intell. **2**(6), 522–536 (1980)
18. Gavrila, D.M., Davis, L.S.: 3D Model-based tracking of humans in action: a multi-view approach. Comp. Vis. Pattern Recogn. Conf. pp. 73–80 (1996)
19. Martin, F., Horaud, R.: Multiple-camera tracking of rigid objects. Research Report 4268 INRIA Montbonnot, France, September (2001)
20. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679–698 (1986)
21. Fitzgibbon, A.W., Pilu, M., Fisher, R.B.: Direct least square fitting of ellipses. IEEE Trans. Pattern Analy. Mach. Intell. **21**(5), 476–480 (1999)
22. Gnu scientific library. http://www.gnu.org/software/gsl/

Author biographies

Jorge Usabiaga received the B.S degree in Physics from The University of the Basque Country, Spain in 1999 and the M.Sc. in Computer Science from the University of Nevada, Reno in 2005, where he worked as a research assistant at the Computer Vision Laboratory (CVL) of the Department of Computer Science and Engineering. Currently, he is working as a software engineer at the International Game Technology (IGT) in Reno, Nevada. His research interests include computer vision, artificial intelligence and pattern recognition.

Ali Erol received the B.S degree in electrical and electronics Engineering from Bilkent University, Turkey in 1991 and the M.Sc. and Ph.D. degrees in electrical and electronics engineering from Middle East Technical University, Turkey in 1995 and 2001 respectively. He worked as a postdoctoral fellow in the Computer Vision Laboratory (CVL) of the Department of Computer Science and Engineering at the University of Nevada, Reno (UNR). Currently, he is working as a research scientist in Ocali Software, Turkey for the development of an image-based 3D reconstruction software. His research interests include computer vision, image processing and pattern recognition.

George Bebis received the B.S. degree in mathematics and M.S. degree in computer science from the University of Crete, Greece in 1987 and 1991, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Central Florida, Orlando, in 1996. Currently, he is an Associate Professor with the Department of Computer Science and Engineering at the University of Nevada, Reno (UNR) and Director of the UNR Computer Vision Laboratory (CVL). His research interests include computer vision, image processing, pattern recognition, machine learning, and evolutionary computing. His research is currently funded by NSF, NASA, ONR, and Ford Motor Company. Dr. Bebis is an associate editor of the Machine Vision and Applications Journal, and serves on the Editorial Board of the Pattern Recognition Journal and the International Journal on Artificial Intelligence Tools. He has served on the program committees of various national and international conferences, and has organized and chaired several conference sessions. In 2002, he received the Lemelson Award for Innovation and Entrepreneurship. He is a member of the IEEE and the IAPR Educational Committee.

Richard Boyle received a B.A. in (bio) psychology from the University of Colorado, Boulder, an M.Sc. in physiology from McGill University, Montreal, Canada, and a Ph.D. in biological sciences from the Scuola Normale Superiore, Pisa, Italy. Currently, he is Director of the BioVIS Technology Center at NASA Ames. BioVis does biology research and develops advanced visualization, imaging, simulation and computer know-how to support the goals of NASA's life sciences and space biosciences programs. In the past, he has been on the faculty of the Department of Otolaryngology/Head-Neck Surgery at the Oregon Health Sciences University (OHSU) in Portland, the Departments of Physiology and Pharmacology and Neurosciences Graduate Program at OHSU, and adjunct scientist at the R. S. Dow Neurological Sciences Institute (formally of the Good Samaritan Hospital, Legacy Health System, and now OHSU). Since 1984, he has regularly conducted hair cell studies at the Marine Biological Laboratory, Woods Hole, Massachusetts. Dr. Boyle has presented and published more than 140 papers and abstracts on neural mechanisms of cardiovascular regulation; spinal cord physiology; cerebellar mechanisms controlling head posture, optokinetic and smooth pursuit ocular pursuit in behaving animals; peripheral and central vestibular morphophysiology in fish and primates; neural regeneration; and biophysical mechanisms of hair cell function. He has been an invited speaker at over 50 institutions in the United States, Europe, Russia, and Japan.

Xander Twombly received the B.A. degree in Physics from Reed College, Portland, in 1987 and the Ph.D. degree in biophysics and computational neuroscience from The Johns Hopkins University, Baltimore, in 1997. Currently, he is a Staff Scientist with the Research Institute for Advanced Computer Science (RIACS) at the NASA Ames Research Center, Moffett Field, CA and technical area lead in Discovery and Systems Health (DaSH). His research interests multi-source data fusion, Bayesian modeling, pattern recognition, image processing and adaptive control systems. Recent research has focused on physics-based modeling of wiring insulation failure in aging aircraft and Bayesian inversion techniques to localize and classify fault types.