Global Hand Pose Estimation by Multiple Camera Ellipse Tracking

Jorge Usabiaga[1], Ali Erol[1], George Bebis[1]
Richard Boyle[2], and Xander Twombly[2]

[1]Computer Vision Laboratory, University of Nevada, Reno, NV 89557
(usabiaga,aerol,bebis)@cse.unr.edu
[2]BioVis Laboratory, NASA Ames Research Center, Moffett Field, CA 94035
Richard.Boyle@nasa.gov, xtwombly@mail.arc.nasa.gov

Abstract. Immersive virtual environments with life-like interaction capabilities have very demanding requirements including high precision and 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) [1], 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.

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 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. 1). 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. It integrates high-fidelity graphics, force-feedback devices, and real-time computer simulation engines to achieve an immersive training environment.

Fig. 1. Virtual Glove Box: A stereoscopic display station provides a high-resolution immersive environment corresponding to a glovebox facility. The users interact with virtual objects using datagloves.

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 a number of 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 an marker-based 3D global hand pose (i.e., position and orientation of the palm) estimation system that 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 a 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 Section 3, we describe the multiple camera environment used track the hand in the context of our application. In Sections 4 and 5, we provide detailed descriptions of the multiple camera ellipse tracking algorithm and its application to camera re-calibration. Section 6 presents our experimental results and comparisons. Finally, Section 7 concludes this study.

2 Previous Work

Marker-based hand tracking is not a very common approach due its intrusive nature. Nevertheless, there have been many attempts using point markers [4–9]. Placing a number of markers on the dorsal surface of the hand, fingertips and/or joints can provide valuable information that can be used to estimate joint angles by solving an inverse kinematics problem. In [6], Holden applied model-based tracking using fingertip and joint markers for ASL recognition. Lien et al. [7] and Lee [8] used stereo cameras to extract the 3D locations of a number of markers on the palm and fingertips and then applied Genetic Algorithms (GAs) to estimate the orientation of the palm. The state of the fingers was estimated using inverse kinematics and regression techniques. In [4], closed form solutions were derived to calculate the angles from 2D marker positions under orthographic

projection. In a more recent study, Kim et al. [9] used white fingertip markers under black-light and stereo cameras to extract 3D fingertip locations for gesture classification.

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]. Moreover, 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 quite 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 [10]. First, like points or straight lines, they are preserved under perspective and projective transformations. 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 [10, 11] can be obtained, avoiding more expensive non-linear iterative techniques.

To the best of our knowledge, Maggioni et al. [12] is the only study using conics, (i.e., two concentric circular markers) for estimating global hand pose in 3D. Viewing the markers 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 Figure 3.

Specifically, the VGX mock-up contains 8 hardware-synchronized cameras located at the corners of the box, several fluorescent lights, and a white background to help segmenting the hands. The intrinsic parameters of the cameras and radial distortion parameters were calibrated using Matlab's Calibration Toolbox [13]. To estimate the extrinsic camera parameters, Svoboda's [14] multiple camera self-calibration procedure was used.

During simulation, users wear a glove with an elliptical marker placed on the dorsal part of the palm. In principle, it is possible to estimate the pose of the hand using two coplanar ellipses [10], however, resolution limitations combined with un-constrained hand motion can make 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 [11]. Camera placement in the VGX mock-up satisfies this visibility constraint.

4 Multiple-Camera Ellipse Tracking

Ellipse pose estimation is a well studied topic and there exist several efficient algorithms for estimating pose information using one or two cameras under certain conditions. In our initial experiments, we found Quan's algorithm [11] using two views of a single ellipse to be very efficient, fast, and accurate. This algorithm 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 two ellipses in each of the two images are described in [11].

The use of multiple cameras was deemed necessary in our application to allow hand tracking independent of viewpoint. In our system, the elliptical 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 4.1), 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 hand tracking approach that integrates information from any number of cameras.

In model-based tracking, at each frame of an image sequence, a search in the parameter space is performed to find the best parameters that minimize a matching error between groups of model features and groups of features extracted from the input images. In the case of multiple cameras, the errors over all the available views are accumulated. The search is often initiated by a prediction based on the dynamics of the system. In the first frame, however, a prediction is not available and a separate initialization procedure is required.

In our system, we have used Quan's algorithm [11] for initialization purposes. There are many different ways to conducting the search or equivalently minimize the matching error. Here, we present an algorithm based on Martin and Horaud's [15] extension of Lowe's model-based pose estimation algorithm [16]. Specifically, there are three main processing steps in our algorithm: (1) active camera selection, where the best cameras for pose estimation are determined, (2) matching error computation, where the similarity between the projected model ellipse and the image features is calculated, and (3) pose estimation, where the matching error is minimized.

4.1 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 provide an image where the contour of the ellipse is too close to the silhouette of the hand. The criterion used for this is 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. Finally, we do not consider cameras that provide images where the ellipse is completely or partially occluded.

4.2 Computation of Matching Error

The ellipse model is represented by a set of uniformly sampled points on its boundary. For each active camera i, a signed error vector e^i is computed by (1) projecting the m points onto the camera's image plane using the current prediction of the pose of the ellipse, and (2) searching for the maximum gradient along the normal to the projected contour at the sampled points. The errors of all the points are concatenated to form a vector:

$$\boldsymbol{e^i} = \left[\boldsymbol{e^i_1}, ..., \boldsymbol{e^i_m}\right]^{\boldsymbol{T}} \tag{1}$$

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.

4.3 Pose Estimation

Pose estimation corresponds to finding the pose parameters T (i.e., position and orientation of the ellipse) that minimize the matching error. Many studies [16] [15] employ Newton's method which subtracts a vector of corrections, x from the current estimate for T at each iteration. If T^k is the parameter vector corresponding to iteration k, then,

$$\boldsymbol{T^{k+1}} = \boldsymbol{T^k} - \boldsymbol{x} \tag{2}$$

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

$$\boldsymbol{e} = \boldsymbol{Jx} \tag{3}$$

where J is the Jacobian. The total error vector e is is obtained by weighting and concatenating the error vectors (see Eq. 1) of the active cameras given by:

$$\boldsymbol{e} = \left[\boldsymbol{w^1 e^1}, ..., \boldsymbol{w^n e^n}\right]^{\boldsymbol{T}} \tag{4}$$

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 algorithm to converge an it does not really improve results.

5 Extrinsic Parameters Re-calibration

Multiple camera calibration assuming an arbitrary camera configuration is a difficult problem. Svoboda's [14] 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.

Re-calibration of a multi-camera system could be necessary for many reasons, for example, when the cameras move. In this case, even a slight variation in the position or orientation of the cameras could affect pose estimation. To update and further optimize the extrinsic camera parameters, we have employed our ellipse tracking algorithm. Specifically, re-calibration 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**, load the poses, images, and frames p where this camera was active; we will be referring to these frames as **active** frames (see Table 1 bottom).
3. Using this information, compute the errors as explained in 4.2, however, instead of putting them in a vector, add their absolute values (m is the number of samples per frame):

$$\boldsymbol{e^i = \sum_{k=1}^{p} \sum_{j=1}^{m} \mid e^i_{kj}(x^i) \mid} \tag{5}$$

   where k indicates the frame number, j indicates the sample number, and i indicates the camera number. e^i_{kj} is a function of the extrinsic camera parameters x^i. The Nelder-Mead's Simplex algorithm [17] was used to find the Δx^i that minimizes the error e^i.
4. Go to step 1 until the error is smaller than a threshold or a maximum number of iterations has been reached.

6 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. 2).

Table 1. Example for a sequence with 999 frames. **Top** Recover the ellipse pose for all frames. **Bottom** Run re-calibration for all cameras (A = Active camera/frame; I = Inactive camera/frame)

1- Run ellipse tracking for whole training sequence

Frame	Camera					Ellipse pose	
⇓	0	1	...	7		Orientation	Position
1	I	I	...	A	→	θ_1	p_1
2	I	I	...	A	→	θ_2	p_2
⋮	⋮				→	⋮	⋮
997	A	A	...	I	→	θ_{997}	p_{997}
998	A	A	...	I	→	θ_{998}	p_{998}
999	A	A	...	I	→	θ_{999}	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	→	R_0	t_0
1	I	I		A	→	R_1	t_1
2	I	I		I	→	R_2	t_2
3	I	I	...	I	→	R_3	t_3
4	A	A		I	→	R_4	t_4
5	I	I		A	→	R_5	t_5
6	I	I		A	→	R_6	t_6
7	A	A		I	→	R_7	t_7

6.1 Accuracy of Ellipse Pose Estimation

To evaluate the accuracy of pose estimation, we compared our algorithm with Quan's ellipse pose estimation algorithm, which, in our opinion, is the best available algorithm for a two camera system. To be able to use Quan's algorithm in our multi-camera environment, we used the same criteria given in section 4.1 for selecting the best two cameras.

Fig. 2 shows the re-projection error (i.e., matching error given in section 4.2) for both algorithms. The square wave shaped curve on the top 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.

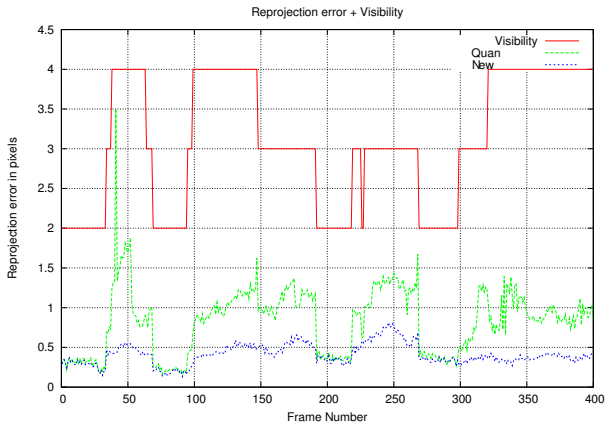

Fig. 2. The re-projection error of Quan's algorithm and our algorithm.

Fig. 3(Left) 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. 2. 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.

Finally, Fig. 3(Right) shows several examples to demonstrate our multiple-camera algorithm in the case of three active cameras.

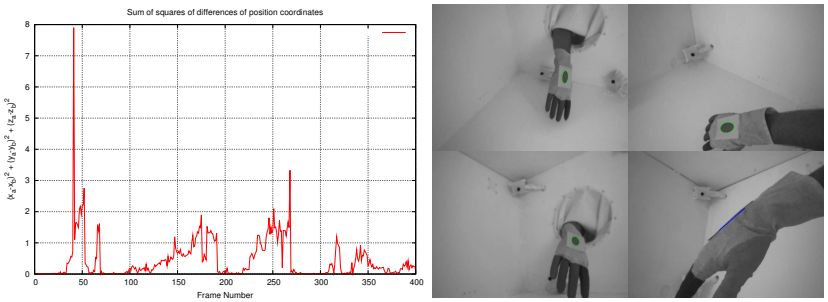

Fig. 3. (Left) Sum of squares differences of position coordinates between two and multiple-camera algorithms. (Right) Re-projection of the ellipse on the input images for frame 42 of a sequence. The images where the re-projected ellipse is drawn in green correspond to the active cameras.

6.2 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.

6.3 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 different test sequence. Fig. 4 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, to the point where it is almost constant.

7 Conclusion and Further 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 illustrate the

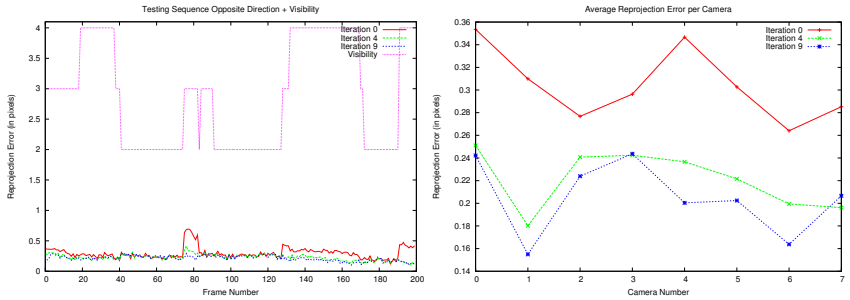

Fig. 4. The re-projection error on a testing sequence with and without re-calibration. (Left) Average re-projection error per frame for the testing sequence computed with the extrinsic parameters iterations 0, 4 and 9. (Right) Average error of the whole sequence per camera

effectiveness of the proposed approach both in terms of pose estimation and re-calibration. For future work, we plan to consider the problem of estimating the full DOF of the hand. Estimating the global pose of the hand is an important step in this process.

Acknowledgments

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

References

1. Twombly, X., Smith, J., Montgomery, K., Boyle, R.: The virtual glovebox (vgx): a semi-immersive virtual environment for training astronauts in life science experiments. Journal of Systemics and Informatics **6** (2006)
2. Pavlovic, V.I., Sharma, R.S., Huang, T.S.: Visual interpretation of hand gestures for human-computer interaction: A review. PAMI (1997)
3. Erol, A., Bebis, G., Nicolescu, M., Boyle, R., Twombly, X.: A review on vision based full dof hand motion estimation. IEEE Workshop on Vision for Human Computer Interaction (2005)
4. Chua, C.S., Guan, H.Y., Ho, Y.K.: Model-based finger posture estimation. In: ACCV2000. (2000)
5. Chua, C., Guan, H., Ho, Y.: Model-based 3d hand posture estimation from a single 2d image. Image and Vision Computing **20** (2002) 191–202
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 and Vision Computing **16** (1998) 121–134
8. Lee, J., Kunii, T.: Constraint-based hand animation. In: Models and Techniques in Computer Animation. Springer, Tokyo (1993) 110–127
9. Kim, H., Fellner, D.W.: Interaction with hand gesture for a back-projection wall. In: CGI '04: Proceedings of the Computer Graphics International (CGI'04), Washington, DC, USA, IEEE Computer Society (2004) 395–402

10. Ma, S.: Conics-based stereo, motion estimation and pose determination. IJCV **10** (1993) 7–25
11. Quan, L.: Conic reconstruction and correspondence from two views. IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1996) 151–160
12. Maggioni, C., B., K.: Gesturecomputer - history, design and applications. In Cipolla, R., Pentland, A., eds.: Computer Vision for Human-Machine Interaction. Cambridge (1998) pp. 312–325
13. Bouguet, J.Y.: (Camera calibration toolbox for matlab)
14. Svoboda, T., Martinec, D., Pajdla, T.: (A convenient multi-camera self-calibration for virtual environments)
15. Martin, F., Horaud, R.: Multiple-camera tracking of rigid objects. In: INRIA. (2001)
16. Lowe, D.: Fitting parameterized three-dimensional models to images. PAMI **13** (1991) 441–450
17. : (Gnu scientific library) http://www.gnu.org/software/gsl/.