Complemental Use of Multiple Cameras for Stable Tracking of Multiple Markers

Yuki Arai and Hideo Saito

Department of Information and Computer Science,
Keio University, Yokohama, Japan
{araiguma,saito}@ozawa.ics.keio.ac.jp

Abstract. In many applications of Augmented Reality (AR), rectangular markers are tracked in real time by capturing with cameras. In this paper, we consider the AR application in which virtual objects are displayed onto markers while the markers and the cameras are freely moving. In this situation, the marker cannot be tracked when the marker is occluded by some objects. In this paper, we propose a method for tracking the projection matrix between the image and the marker even when the maker is occluded, by using cameras. In this method, we transfer the projection matrix for the marker that is detected by the cameras in order to estimate the relative projection matrix for the occluded marker. After computing the relative projection matrices using multiple cameras, we compute a more accurate projection matrix by using particle filter. As a result, we can continuously track the markers even when the marker is occluded.

Keywords: augmented reality, marker tracking, particle filter, multiple cameras.

1 Introduction

In recent years, Augmented Reality (AR) is well studied to present information for education or entertainment. AR is the combination of real-world and virtual reality, where computer generated virtual objects are overlaid onto video of the real world in real-time. “Magic Book” [1] is the example of AR. As marker is printed on the book, we can see virtual object on the book when we open the book. In order to achieve AR, we need to determine the camera pose accurately. Sensor based methods and vision based methods are well known for tracking the camera pose. Using sensors such as a magnetic sensor or a gyroscope are a robust way to rapid by track camera motion and change of lighting condition [2,3]. However sensor can not get enough information to calibrate accurate camera pose and has area that can be used. To compute camera pose, vision based method uses landmark in real space such as markers [4], interest points or edges [5,6], models that are pre-calibrated before using system [7]. Vision based methods do not need any particular device such as sensor, it is low cost and we can construct the system easily. In this paper, we propose marker based method since we can easily build the environment.

We use ARToolkit [8] to compute camera pose and position using rectangular markers. A unique pattern is drawn on each marker and the system can identify each

R. Shumaker (Ed.): Virtual and Mixed Reality, LNCS 5622, pp. 413–420, 2009.

marker using this pattern. Therefore the marker can easily be detected from input image so that we can achieve AR easily. However the marker can not be detected when the marker is occluded or illumination condition is not sufficient. There are many researches to solve such marker occlusion problem. Tateno et al. [9] have proposed nested marker to solve partial occlusion. The nested marker consists of several small markers. They employ small markers in the nested marker when the marker is occluded. They can display virtual object on the marker even when marker occlusion occurs. David et al. [10] have applied particle filter with tracking. They use markers to compute the projection matrix when the marker is not occluded. While marker occlusion occurs, they use marker corners as features for tracking using particle filter. David et al. use only one camera to solve partial occlusion of markers, while Pilet et al. [11] have used fixed multiple cameras to solve full occlusion of markers. As a preprocessing step, Pilet et al. calibrate all the cameras. Then they use another camera's information when the marker is occluded. This method can display a virtual object even if the camera can not see the marker perfectly. However this method requires to fix the cameras.

In this paper, we consider the AR application in which virtual objects are displayed onto markers while all the markers and the cameras are freely moving by users' interaction. In this situation, the marker cannot be tracked when the marker is occluded by some objects. For solving this problem, we propose a method for tracking the projection matrix between the image and the marker even when the maker is occluded, by complemental use of multiple cameras. In this method, all cameras share the projection matrices between the markers and the cameras for computing the relative projection matrix for occluded markers, so that virtual objects can be displayed onto the occluded markers.

2 System Description

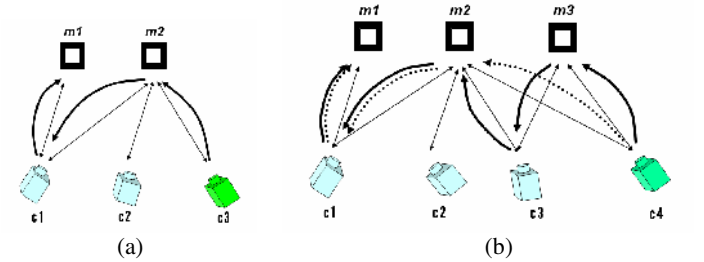
In this paper, we consider the AR application in which virtual objects are displayed onto markers while all the markers and the cameras are freely moving by users' interaction. In the proposed method, we compute the projection matrix using ARToolkit when we can detect the marker from the image. While marker occlusion occurs, we compute projection matrix by sharing projection matrices between cameras. However the projection matrix computed from other cameras is not accurate, we optimize the projection matrix using particle filter.

2.1 Compute Projection Matrix Using Other Cameras

For computing a projection matrix between the camera and occluded marker, we compute extrinsic parameters between them using other cameras. Extrinsic parameters represent the rotation matrix and translation vector between two coordinates. For computing the extrinsic parameters between the camera and the occluded marker, we use common markers that are detected from more than two cameras. By multiplying extrinsic parameters computed for the common markers, we can compute extrinsic parameters between the camera and the occluded marker that are not computed directory using ARToolkit. After computing these parameters, we compute projection

matrix using them and the intrinsic parameters of the camera that were previously estimated. Fig.1 shows the example for computing extrinsic parameters using other cameras. In Fig.1, straight arrows represent that the camera can detect the marker, so that the projection matrix between the camera and the marker can be computed using ARtoolkit. Then the extrinsic parameters between them can also be computed. We denote the extrinsic parameters between the camera c and the marker m by $\mathbf{M}[m \leftarrow c]$. The reverse extrinsic parameters are computed by inverting the matrix which we denote by $\mathbf{M}[c \leftarrow m]$. In Fig.1(a), occlusion occurs on the marker $m1$ from the camera $c3$, we can not compute $\mathbf{M}[m1 \leftarrow c3]$. However we can compute $\mathbf{M}[m1 \leftarrow c1]$, $\mathbf{M}[c1 \leftarrow m2]$ and $\mathbf{M}[m2 \leftarrow c3]$, then we can compute $\mathbf{M}[m1 \leftarrow c3]$ from multiplying them.

In some case, there are many routes from the camera to the marker, so we have to select the best route. Fig.1 (b) shows the case using four cameras and three markers. In this case, camera $c4$ can not detect marker $m1$, so we can not compute $\mathbf{M}[m1 \leftarrow c4]$ directory. However using common markers that can be detected from more than two cameras like marker $m2$, we can compute $\mathbf{M}[m1 \leftarrow c4]$. Fig.1 (b) shows two routes to link camera $c4$ and marker $m1$. The $\mathbf{M}[m \leftarrow c]$ computed with a marker generally includes some errors, the errors are multiplied while passing other cameras. Therefore we have to select the route that is the most accurate. To select such route, we consider the least number of common markers for computing the objective extrinsic parameters. The reason to select the route that passes the least number of common markers is that the error can be increased by the number of common markers. We use the Dijkstra algorithm to select the shortest route. Dijkstra algorithm is famous to solve shortest path problem. When the number of common markers to pass is same, we select the path that includes more accurate extrinsic parameters. For evaluating the accuracy, we use the area of the common markers captured in the image.



(a) (b)

Fig. 1. Computing projection matrix from sharing projection matrices. Straight arrows represent the fact that the camera can detect the marker. (a) Computing extrinsic parameter using common marker $m2$. (b) There are some routes between camera $c4$ and marker $m1$. We have to select the shortest route.

2.2 Particle Filter Optimization

We denote the relative projection matrix using multiple cameras between the camera c and the marker m at time t by $\mathbf{P}_{tmp}(c,m,t)$, which was discussed in Section 2.1. The projection matrix computed with markers directly includes some error. The projection matrix $\mathbf{P}_{tmp}(c,m,t)$ still includes error as illustrated in Fig.2. For computing the correct

projection matrix, we detect the occluded marker in the image using $\mathbf{P}_{tmp}(c,m,t)$. The shape of the marker, projected by using $\mathbf{P}_{tmp}(c,m,t)$, is almost correct even if the $\mathbf{P}_{tmp}(c,m,t)$ includes error. So we can detect the occluded marker in the image to scale and translate the projected marker. In order to detect the occluded marker, we compute the matrix $\mathbf{H}_t$. $\mathbf{H}_t$ is affine transformation matrix at time t which transforms the marker projected by using $\mathbf{P}_{tmp}(c,m,t)$ onto the occluded marker in the image. $\mathbf{H}_t$ is composed of parameters λ, u_x and u_y. λ indicates scale and u_x, u_y indicate the distance of translation. After detecting the marker in image using $\mathbf{P}_{tmp}(c,m,t)$ and $\mathbf{H}_t$, we compute correct projection matrix using it.

$$\mathbf{H} = \begin{bmatrix} \lambda & 0 & u_x \\ 0 & \lambda & u_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

In this section, we describe our method for estimating $\mathbf{H}_t$ by using a particle filter for each marker. We denote a particle by hypothesis $\mathbf{H}_t^i$ and weight w_t^i at time t. (i=1…N). N is the number of particles. The estimation of error from using a particle filter starts from the time that marker occlusion occurs (t=t_0). We assume that the marker in image is known at time t_0-1. We compute affine transformation matrix between the marker that is projected using $\mathbf{P}_{tmp}(c,m,t_0)$ and the marker in image at time t_0-1. We set the initial parameter of $\mathbf{H}_t^i$ with the affine transformation matrix. When the route that is used to compute projection matrix from sharing projection matrices changes, we also initialize the particles.

After initialization, we estimate H_t^i from $\mathbf{H}_{t-1}^i$ sequentially. We have no prior knowledge of cause of error in each frame, we estimate $\mathbf{H}_t^i$ as:

$$\mathbf{H}_t^i = \mathbf{H}_{t-1}^i + \varepsilon = \begin{bmatrix} \lambda & 0 & u_x \\ 0 & \lambda & u_y \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} \varepsilon_{scale} & 0 & \varepsilon_x \\ 0 & \varepsilon_{scale} & \varepsilon_y \\ 0 & 0 & 0 \end{bmatrix} \tag{2}$$

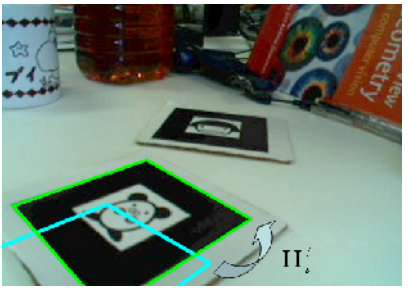


Fig. 2. Projecting marker using $\mathbf{P}_{tmp}(c,m,t)$ computed by Section 2.1. $\mathbf{P}_{tmp}(c,m,t)$ has error, projected marker does not match marker in image.

ε composed of $\varepsilon_{\text{scale}}$, ε_{x} and ε_{y}. Each parameter represents the random noise which has a normal distribution.

$$\varepsilon_{scale} = N(0, \sigma_{scale}^{2}) \tag{3}$$

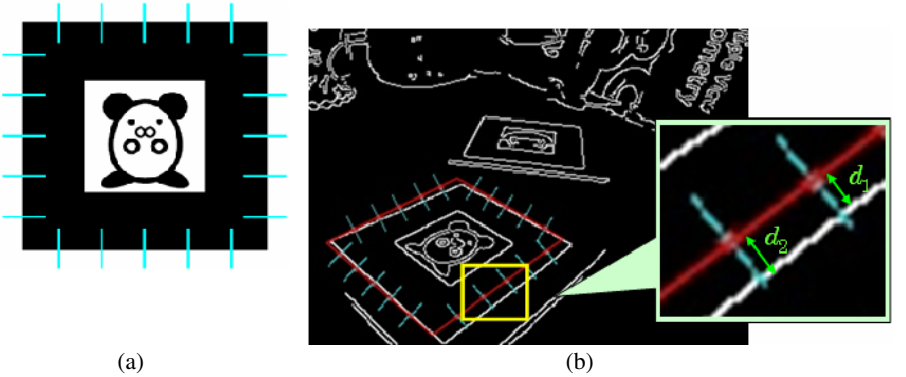
$$\varepsilon_{x}, \varepsilon_{y} = N(0, \sigma_{trans}^{2}) \tag{4}$$

After estimating hypothesis at time t, we compute weight from input image for each particle. Since the weight indicates the confidence of the estimation, the particles that have a heavy weight indicate a correct hypothesis. We weight particles using the distance between the edge in input image and the projected marker that was computed with $\mathbf{H}_t^i$ and $\mathbf{P}_{\text{tmp}}(c,m,t)$. First we extract the edge from input image using a Canny edge detector, and then project the marker with $\mathbf{P}_{\text{tmp}}(c,m,t)$ and $\mathbf{H}_t^i$. Next we put sample points on the projected marker contour. Fig.3 shows the example of sample points (K=20). We scan the pixels along the marker contour normal from the sample points. We measure the distance d between the sample point and edge that is scanned. We compute distance d from each sample point and compute sum of distance. We represent the sum of distance as expectation c_t^i. After computing c_t^i for each particle, we weight the particles. We weight particle heavier when c_t^i is small by using a Gaussian function to weight the particles.

$$w_t^i \propto e^{-\frac{c_t^{i\,2}}{2\sigma^2}} \tag{5}$$

We normalize the weight so that:

$$\sum_{n=1}^{N} w_t^n = 1 \tag{6}$$



(a) (b)

Fig. 3. (a) K sample points on the marker (K = 20). (b) Computing distance between projected marker using particle and edge in input image. We scan the pixels along the marker contour normal from all sample points.

After computing all weights for each particle, we evaluate the weights and select the particle that has the heaviest weight. We indicate the selected particle as the error at time t and compute projection matrix using the particle and $\mathbf{P}_{\mathrm{tmp}}(c,m,t)$.

After computing the projection matrix, we remove particles that have small weight. After a few iterations, all but one particle will have negligible weight and reduce accuracy. To remove this particle, we eliminate particles which have small weight and concentrate on particles with large weights.

3 Experiments

In order to evaluate the performance of the method that we proposed, we track markers using multiple cameras. We move the markers on the desk and record the markers using three web cameras. The parameters that we use in the experiment are as follows: number of particle N=100, sample points on the marker K=20.

First, we changed the parameters σ_{scale} and σ_{trans}, and evaluated the accuracy of marker tracking. When σ_{scale} or σ_{trans} were too big, the marker position was not fixed.

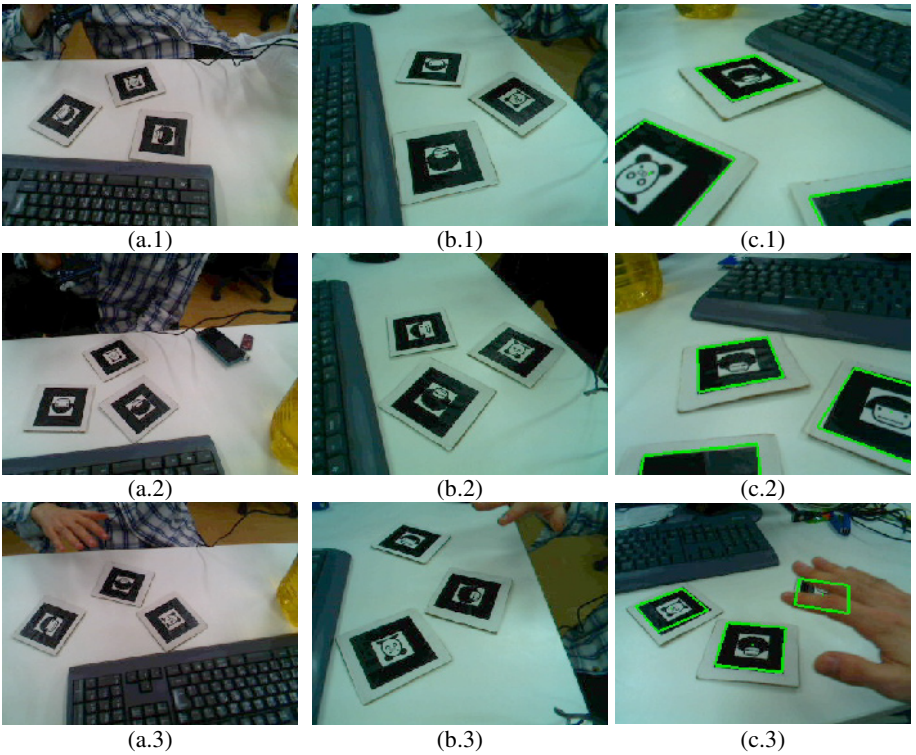


Fig. 4. The result of tracking markers. We track markers using the proposed method. (a)(b) Input image from cameras $c1$, $c2$. (c) Input image from camera $c3$ and projecting marker computed by proposed method on the image.

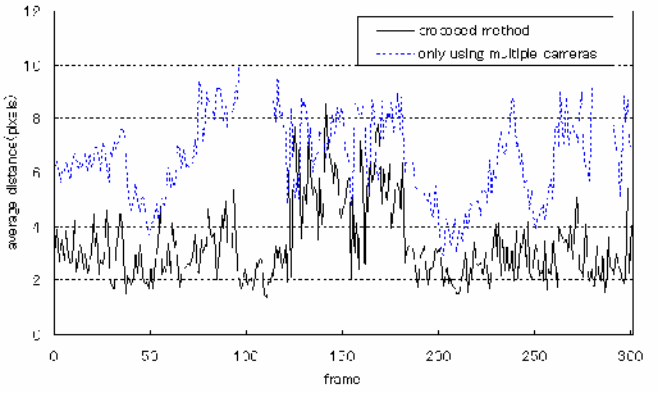


Fig. 5. A comparison of the accuracy of the proposed method and a method which does not use a particle filter

When σ_{scale} or σ_{trans} were too small, we could not adjust cause of error. After several tests, we have decided the parameter at σ_{scale}=0.05 and σ_{trans}=10.

After experimentally determining σ, we evaluated the accuracy of our method using a video sequence captured with three USB cameras. To evaluate the accuracy, we used the average distance between sample points on the marker that is projected using projection matrix and the edge in input image. The distance written in Section 2.2 is used for computing weight of each particle. In the video sequence, the cameras and markers move freely, and target marker is partially occluded. Target marker means the marker for which we estimate the projection matrix using our method. At least one marker can be seen from camera $c3$. The other cameras $c1$ and $c2$ see more than two markers in the sequence.

Fig.4 shows the input images from three cameras and marker tracking result when marker occlusion occurs or marker is partially out of the image. As we track markers using other cameras when marker occlusion occurs, we can track the marker which has full occlusion such as illustrated in Fig.4 (c.3). However, because we use the edge to optimize the marker position, we fail to track marker if the marker is completely occluded for a long time or the marker is near an edge that is parallel to the marker contour.

Fig.5 shows the average distance of the proposed method and the method which does not use a particle filter. Accuracy of the projection matrix computed by the proposed method is higher than the method which uses only multiple cameras.

4 Conclusion

In this paper, we presented a method that is robust enough to handle occlusion of markers by using multiple handheld cameras. When marker occlusion occurs, we share projection matrices between cameras and we can detect existence of marker even if marker is occluded.

Future work will focus on how to best select the route to pass to compute projection matrix using other cameras. Accuracy of the projection matrix depends not only on the result of marker area in input image, but also camera pose. We need to consider selecting the route using the different in camera poses as well.

References

1. Mark, B., Hirokazu, K., Ivan, P.: The Magic Book: a transitional AR interface. Computers & Graphics 25, 745–753 (2001)
2. Andrei, S., Gentaro, H., David, T.C., William, F.G., Mark, A.L.: Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking. In: Proceedings of ACM SIGGRAPH 1996, pp. 429–438 (1996)
3. Livingston, M.A., Andrei, S.: Magnetic tracker calibration for improved augmented reality registration. Presence: Teleoperators and Virtual Environments 6(5), 532–546 (1997)
4. Xiang, Z., Stephan, F., Nassir, N.: Visual Marker Detection and Decoding in AR Systems: A Comparative Study. In: International Symposium on Mixed and Augmented Reality, p. 97 (2002)
5. Iryna, S., David, G.L.: Scene Modeling, Recognition and Tracking with Invariant Image Features. In: Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 110–119 (2004)
6. Luca, V., Vincent, L., Pascal, F.: Combining edge and Texture Information for Real-Time accurate 3D Camera Tracking. In: Proceedings of the third IEEE and ACM International Symposium on mixed and Augmented Reality, pp. 18–56 (2004)
7. Harald, W., Florent, V., Didier, S.: Adaptive Line Tracking with Multiple Hypotheses for Augmented Reality. In: Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 62–69 (2005)
8. Kato, H., Billinghurst, M.: Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. In: Proceedings of the 2nd International Workshop on Augmented Reality, San Francisco, USA (1999)
9. Tateno, K., Kitahara, I., Ohta, Y.: A Nested Marker for Augmented Reality. In: IEEE Virtual Reality Conference VR 2007, pp. 259–262 (2007)
10. David, M., Yannich, M., Yousri, A., Touradj, E.: Particle filter-based camera tracker fusing marker and feature point cues. In: Proc. of the IS and SPIE Conf. on Visual Communications and Image Processing (2007)
11. Julien, P., Andreas, G., Pascal, L., Vincent, L., Pascal, F.: An All-In-One Solution to Geometric and Photometric Calibration. In: Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 69–78 (2006)