Vision-based System for Head Pose Tracking in Indoor Immersive Environments

Srinivasa G. Rao
University of North Carolina, Charlotte
srao3@uncc.edu

Stephen J. Schmugge
University of North Carolina, Charlotte
sjschmug@uncc.edu

Larry F. Hodges
University of North Carolina, Charlotte
lfhodges@uncc.edu

Abstract

We describe an inexpensive, accurate, fast and scalable vision based indoor head pose tracking system. This system can be used for indoor tracking in VR and AR environments. Our approach uses video projectors to project a *display grid pattern* on the floor. The pattern consists of circular binary code color markers on a black and white checkerboard. A camera is attached to a user's back, looking down. The camera looks at the projected pattern, and its position is calculated in *tracking space based on the correspondence* between the global position of the display grid markers and their image coordinates. We calibrate for the offset between the user's head and the camera; hence head position can be calculated. The system has a mean position error of 4 millimeters and a mean jitter of less than 0.3 millimeters. We augment this position information with an inertial sensor to compute head rotation to achieve full 6DOF tracking.

Keywords

Human Motion Tracking, Virtual Reality, Computer Vision.

1. Introduction

Computer Vision algorithms have been extensively used in head pose tracking for Virtual and Augmented Reality (VR/AR) environments. Most of the systems are hybrid – they may use cameras, inertial devices and/or GPS in a combined form to do tracking. One approach to vision based tracking involves a camera, rigidly attached to some part of a participant's body (usually the head). The camera looks at artificial or natural markers in known positions, and using computer vision techniques, calculates its (and hence the user's head's) pose in a global coordinate system. This approach is known as *inside looking out*. It has been shown by a researchers that, *inside looking out* is better than *outside looking in* (where cameras are mounted in the environment at known positions looking at the user and/or markers attached to him) [9]. This is because, with *inside looking out* even a small motion causes a large change in the image captured by the camera, whereas with *outside looking in* small motions become undetectable as the user moves further away from the sensors. The steady increase in CPU speeds now allows many vision algorithms to run at interactive rates (20 - 30 Hz), making vision based tracking systems viable for use in VR. The best vision based systems are fast and accurate, but relatively expensive. For example, the HiBall tracker uses multiple infrared cameras and infrared LEDs mounted on ceiling for tracking [3,22]. Our main goal in this work is to develop an accurate, scalable and inexpensive system for head pose tracking in VR. In this paper we describe a vision based tracking system that is accurate, affordable, scalable and produces readings at interactive rates.

2. Previous Work

Koller et al. [4] have used inexpensive cameras to track fiducial markers mounted on walls to compute the pose of a camera with respect to a ground-based coordinate system. Thomas et al. [5] mounted an auxiliary camera perpendicular to a primary camera (used for movie video capture and production) to look at the circular fiducial markers mounted on the ceiling to figure out the pose of the primary camera. Lee et. al. [23] used an omni-directional camera to estimate head pose in outdoor and indoor environments. In these cases, the update rate is limited by the frame rate of the camera used and further reduced by the computational time required by the vision algorithms to estimate pose.

In movie production systems, offline processing is performed to add special effects to the captured video, in which case pose computation need not be real time. But for all AR and VR systems, the head pose has to be computed at interactive rates. Hirota et al., Yokokohji et al., Nuemann et al. and Foxlin et al. [1,2,7,8,24] have used either inertial or magnetic tracking systems, in conjunction with camera based systems, to get higher update rates. Usually, the camera provides the pose information to initialize inertial trackers every frame – and before the next frame gives the pose – the readings from the inertial trackers are used. Foxlin et al. [8,24] demonstrate more complex sensor integration. Though these techniques work for indoor environments, they are not very useful outdoors where there are few walls on which to mount fiducials. Hence, Feiner et al. [10] and Piekarski et al. [11] have used GPS along with a camera and inertial trackers to track movement through outdoor environments. A drawback to these systems is that the registration relative to actual position is usually no better than ten centimeters. You and Neumann [6, 20], Ribo et. al. [13], Azuma et. al. [21] and Simon et. al. [12]

have used natural scenery as markers to estimate pose of a camera. The idea is to have a database of outdoor scenery whenever possible and use this information to compute camera pose. Implementing vision algorithms outdoors is difficult due to uncontrolled lighting conditions.

In this paper we focus completely on indoor tracking. All the indoor camera based tracking systems described above suffer from common problems. First, the positions of the markers mounted in the environment need to be manually measured. For large spaces manual measurement is error prone and burdensome. Hence, scalability of such tracking systems is a problem. Another fundamental problem is, as the camera's distance from the markers it can see increases (especially markers stuck in high ceilings as in [5,8]), the accuracy of the pose calculated decreases.

3. Tracking using a Floor-based Fiducial Grid

We address the above problems in the following way. Instead of mounting fiducial markers in the environment, we use projectors to form a *display grid* on the floor of the tracked area. When the camera is mounted on the user, the plane of the floor is always approximately at a small and fixed distance from the camera. In our system, we mount a single inexpensive camera on the user just below the middle of his back so that it looks down at the pattern on the floor. Mounting the camera this way means that the camera is only, on average, about three feet away from the fiducials on the display grid and can still see the grid if the user bends at the waist. See figure 1.

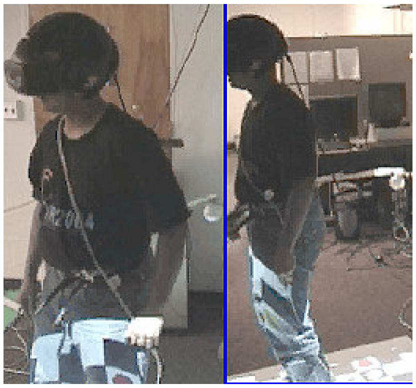

Figure 1: Near frontal view of user and his side view.

3.1 Pre-warped projection and shadow cancellation

To create the *display grid* on the floor, we mount a projector on the wall of the tracked area. Projection is at an oblique angle to the ground. It is necessary to pre-warp the projected pattern such that it registers orthogonally to the ground. The user's body can get in the way of the projected pattern, creating a shadow. To compensate, we use a second projector mounted on the opposite wall to project the same pattern and effectively cancel the shadow as shown in figure 2. Registration of the two patterns is accomplished using the techniques described in Raskar et al. [14,15]. This is done by using a camera mounted near the ceiling such that the image plane of the camera is parallel to and facing the floor on which the patterns are projected. The relationship between a projected image point, p_i, and the corresponding camera image point, c_i , is given up to a scale factor by

$$p_i = \mathbf{H}c_i \qquad (1)$$

where **H** is the homography between the camera and the projector induced by the floor. We pre-warp the projected pattern using **H** to register it to the floor. We do the same to the second projector and hence the projected patterns are registered to each other. See [14] for more details. The projected pattern itself defines a Global Cartesian coordinate system in the tracking space, in which the user can be tracked.

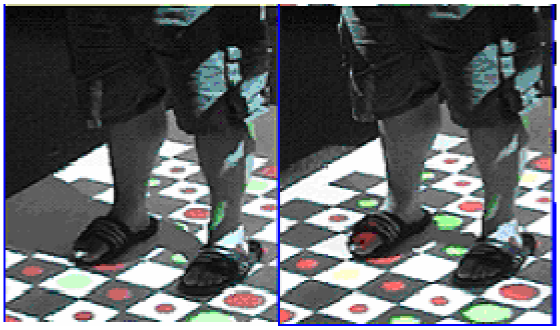

Figure 2: Shadow present (one projector); Shadow cancellation using the second projector on opposite side.

Though we have built our system using only two projectors, Raskar et. al. [14, 15] have shown that many projectors can be registered to create a seamless display on any flat surface. The idea is to fill up the indoor environment with small projectors that can create a *display grid* in which a user can be tracked. Shadows created by projectors on one side of the user are cancelled by the projectors on his other side.

3.2 Tracking Algorithm

3.2.1 ***Calibration***. Red and green colors are projected on the floor and we build a histogram for both colors. This is because the actual colors seen by the camera may not be pure depending on the surface properties of the floor on which the pattern is projected. These (r, g, b) histograms will have peaks near but not exactly at (255,0,0) and (0,255,0). The histograms are then used to locate circular red and green markers in the image captured by the camera using cvCalcBackProject function in the OpenCV library [18].

We also compute the offset between the user's eyes and the camera mounted on his back by placing a second camera very close to the user's eye position looking at a known pattern in a known position, while the camera on the user's back is also looking at another pattern in another known position as shown in figure 5.

3.2.2 ***Tracking.*** Figure 4 presents a flowchart of the algorithm. A camera with known intrinsic parameters (calibrated using the functions provided by the OpenCV library) is mounted on the user's back looking down at a projected pattern. The main idea is to determine what part of the global pattern the camera is looking at based on binary location codes and orientation markers within the camera's field of view. See yellow box in figure 3.

We use two types of markers on top of a black and white checker board – large orientation markers and small binary code markers. The large markers play a dual role of "remembering" the global position and representing the global orientation. The relative positions of a pair of red and green large markers in the camera's field of view give us the user's azimuth orientation in the global coordinate system defined by the projected pattern. Binary code markers give the user's position in the global coordinate system. Detecting the global position every frame using the binary codes in that frame is not efficient. Instead, once the global position is established using binary code markers, the bigger orientation markers are placed in their global position. Once this is done, as long as a pair of the bigger markers (one red orientation marker and the green orientation marker nearest to it) are visible to the camera and can be tracked using the CAMSHIFT [17] algorithm, they are used to place the detected corners in their global positions. For the small binary code markers, a small red circle represents 0 and a small green circle represents a 1. Once we have both global position and orientation, we use the internal chess board corners in the camera's field of view to compute the exact camera pose information.

For example, at a particular frame, suppose the camera sees what is inside the yellow box in figure 3. The large red and green marker pair (pointed to by the yellow arrow) provides orientation. The four small circles (inside the circle) are binary code markers (0100 in this case – going in raster scan order). We could have used the position of the binary markers by themselves to compute camera pose, but, their position is not sub pixel accurate. Instead, we use the position of the detected internal chessboard corners to compute the pose. A simple connected component algorithm is used to find the markers. The resulting contours from this algorithm are then classified into binary markers and code markers based on thresholds set on their areas. The binary markers are searched to see if they contain a binary code and the orientation markers are searched for a pair of orientation markers. If a binary code is found and a pair of orientation markers is found, the image is warped (rotated) so that the red orientation marker is to the left of a green orientation marker, and the line connecting them is horizontal.

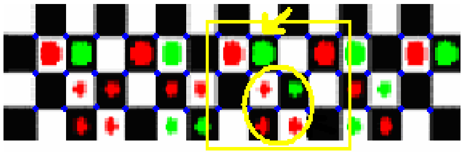

Figure 3: Part of the projected global pattern - blue dots represents detected corners. Yellow box represents what the camera sees. Yellow arrow is pointing to orientation markers. Yellow circle contains binary code markers.

These orientation markers – the pair currently used – are tracked using the CAMSHIFT algorithm. If the algorithm loses track of the orientation markers, then the image is searched again for both orientation markers and code markers. Global position of orientation markers in the coordinate system is established using binary code markers. Global position of the binary code markers is known apriori and is stored in a look up table.

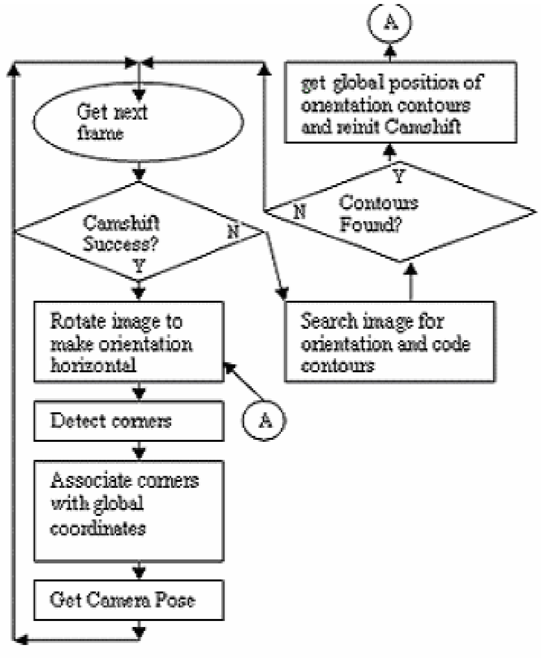
A
get global position of orientation contours and reinit Camshift
Get next frame
Y
N
Contours Found?
Camshift Success?
N
Y
Rotate image to make orientation horizontal
Search image for orientation and code contours
Detect corners
A
Associate corners with global coordinates
Get Camera Pose

Figure 4: Tracking Algorithm

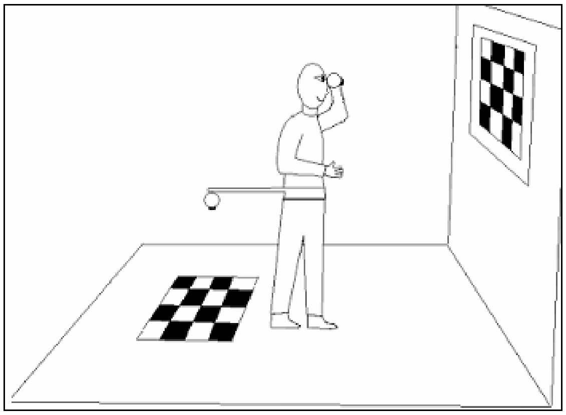

Figure 5: Offset Calibration

The checkerboard corners (see Figure 3) are detected using functionalities provided in the OpenCV library. The corners detected are then associated with their global positions using their distance from the orientation markers in image space. Hence, we have many points (about 20) whose image coordinates and global coordinates are known. This allows us to establish the Homography **H** (**H**= [$\mathbf{h_1}$ $\mathbf{h_2}$ $\mathbf{h_3}$] where each $\mathbf{h_i}$ is a (3,1) column vector) between camera coordinates and global coordinates defined by the projected pattern in which the user is tracked. Using **H** and knowing the camera intrinsic matrix **A**, the pose of the camera can be calculated by

$$K = 1/\|\mathbf{A}^{-1}\,\mathbf{h_1}\| \tag{2}$$

$$\mathbf{r_1} = K\mathbf{A}^{-1}\,\mathbf{h_1}; \tag{3}$$

$$\mathbf{r_2} = K\mathbf{A}^{-1}\,\mathbf{h_2}; \tag{4}$$

$$\mathbf{r_3} = \mathbf{r_1} \times \mathbf{r_2}; \tag{5}$$

$$\mathbf{t} = K\mathbf{A}^{-1}\,\mathbf{h_3} \tag{6}$$

Where **R = [**$\mathbf{r_1}$ $\mathbf{r_2}$ $\mathbf{r_3}$**]** is the rotation, and **t** is the translation of the camera attached to the users back in the global coordinate system. (See [16] for more details.)

R and **t** give us the position of the camera in the global coordinate system. What we actually want is the position of the mid point between the user's eyes. We know the offset between the user's eye and the camera. Once the position and offset of the camera on the user's back is known, we can calculate head position in our global coordinate system.

We use an inertial tracker built into a VFX3D Head-Mounted Display for orientation of the user's head in conjunction with our vision based tracking to get full six degree of freedom tracking information. We could have mounted the camera directly on the back of the user's head looking down to get both the position and orientation of the head. But, the user usually moves his head much faster than his body in a VR environment. Fast movement of the camera causes motion blur which hinders the performance of the vision algorithm. In the future we intend to experiment with a high speed camera attached to the user's head and experimentally find out which method is better. The current system was implemented using an inexpensive IBOT firewire camera running at 320X240 at 30 fps. The camera was connected to a Pentium 4, 3.06 GHz machine with 1GB RAM.

4. Comparison with HiBall

As a relative measure of how well this system works, we compared our vision based position tracking system (total cost less than $3000 USD for an inexpensive camera and 2 projectors) to our 3rdTech HiBall tracker (total cost $30,000 USD) with respect to jitter (sensor noise) and registration (accuracy). Our experimental setup consisted of a HiBall rigidly attached to an IBot camera mounted three feet above the ground and looking down as shown in figure 6. We also made sure that all the following coordinate systems were parallel to each other while conducting the experiments and while we moved the setup to various locations on the pattern – the camera's local coordinate system, the HiBall's local coordinate system, the projected pattern's global coordinate system and the HiBall tracker's ceiling global coordinate system. This way we ensured that both the HiBall and the camera moved by the same distance when the setup was moved during both of the experiments described below.

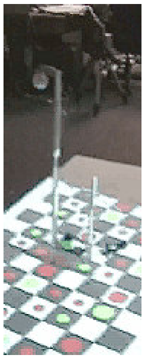

Figure 6: Setup for comparison with HiBall tracker.

Registration (accuracy) – We compared distance moved by our vision based system to distance moved by the HiBall over a number of positions. At each position on the projected grid we took an average of ten readings from both the camera based system and the HiBall, and then computed the distanced moved from the previous position. We repeated this procedure at 100 locations and plotted the distances moved. Figure 7 shows the plot of distance moved (in mm) along the Y-axis. We found the mean error of our system relative to the HiBall to be 4.0405mm with a standard deviation of error of 4.665mm.

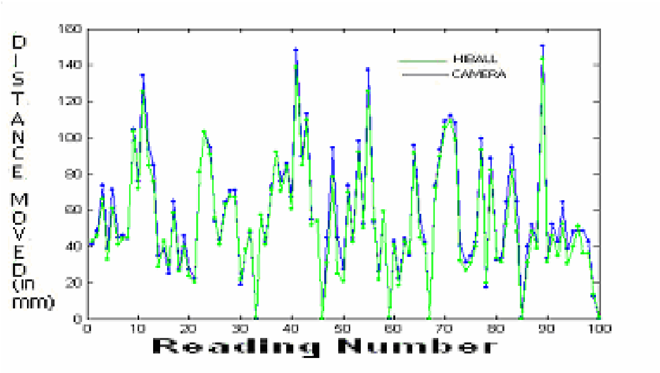

Figure 7: Registration comparison with HiBall

Jitter (sensor noise) - To calculate the jitter of our tracking system, we placed the setup in a fixed position and without moving it, took 100 readings of the displacement reported by the camera and the HiBall. We found the mean jitter of the camera to be 0.2843mm and the mean of HiBall jitter to be 0.0678mm (see Figure 8).

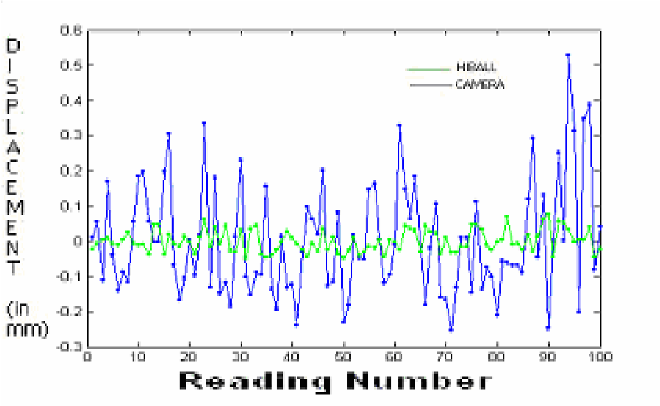

Figure 8: Jitter comparison with HiBall

5. Conclusion and Future Work

We have presented a hybrid 6_DOF tracking system that is scalable, has a registration error of approximately 4 mm, jitter of less than 1 mm and an update rate of 15Hz. In the future, we would like to improve the accuracy by using higher resolution and higher frame rate cameras. Since the projected patterns are static, we can further decrease the overall cost of the system by replacing our video projectors with slide or overhead projectors. Raskar et. al [19] have demonstrated use of small, Lycos projectors. Even less expensive transparency based slide projectors can be built using printed slide patterns, Fresnel lenses and common bulbs. We intend to combine many such *intelligent lamps* and create a *display grid* in which a user in a VR environment can be tracked. We will further optimize the vision algorithms to make them faster. We are also working on computation of end-to-end delay in the system and introducing Kalman Filtering techniques for prediction and interpolation of the pose reported by the tracker. Further, as Foxlin et. al. [8] have demonstrated, inertial sensors can be integrated into our system to increase the update rate.

6. References

[1] State A., Hirota G., Chen D. T., Garrett W. F, and Livingston M. A, “Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking”, ACM SIGGRAPH ’96. pp 429–438.

[2] Y. Yokokohji, Y. Sugawara, and T. Yoshikawa, “Accurate Image Overlay on Video See-through HMDs Using Vision and Accelerometers”, Proc. IEEE VR 2000, Los Alamitos, CA.

[3] Welch, Greg, Gary Bishop, Leandra Vicci, Stephen Brumback, Kurtis Keller, and D’nardo Colucci, “The HiBall Tracker: High-Performance Wide-Area Tracking for Virtual and Augmented Environments”, ACM VRST 1999, London.

[4] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, M. Tuceryan , “Real-time Vision-Based Camera Tracking for Augmented Reality Applications”, ACM VRST, Switzerland, Sept. 15, 1997, pp. 87-94.

[5] G.A. Thomas (BBC) J. Jin, T. Niblett, C. Urquhart, “A Versatile Camera position system for Virtual Reality TV Production”, IBC 1997, Amsterdam, 12-16, September, pp 284-289.

[6] B. Jiang, S.You, U. Neumann, “A Robust Tracking System for Outdoor Augmented Reality”, IEEE VR 2004, Chicago, Mar. 27, 2004.

[7] S. You, U. Neumann, and R. Azuma, “Hybrid Inertial and Vision Tracking for Augmented Reality Registration”, IEEE VR, Los Alamitos, Calif., 1999, pp. 260-267.

[8] Eric Foxlin and Leonid Naimark, “VIS-Tracker: A Wearable Vision-Inertial Self-Tracker”, IEEE VR 2003 Mar. 22, 2003, LA, CA.

[9] Bhatnagar D. K., “Position Trackers for Head Mounted Display Systems: A Survey”, Dept. of CS at the UNC Chapel Hill, Mar. 29, 1993

[10] S. Feiner et al., “A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment”, Proc.1st Int’l Symposium on Wearable Computers, CA, 1997.

[11] P.W. Piekarski, B. Gunther, and B. Thomas, “Integrating Virtual and Augmented Realities in an Outdoor Application”, 2nd Int’l Workshop Augmented Reality, Los

Alamitos, C.A, 1999, pp. 45-54.

[12] G. Simon, A.W. Fitzgibbon, and A. Zisserman, "Markerless Tracking Using Planar Structures in the Scene", Proc. Int'l Symposium Augmented Reality 2000, IEEE CS Press, Los Alamitos, Calif., 2000, pp. 120-128.

[13] M. Ribo, P. Lang, H. Ganster, M. Brandner, Ch. Stock, A. Pinz, "Hybrid Tracking for Outdoor Augmented Reality Applications", IEEE Computer Graphics and Applications, 22(6):54-63, 2002.

[14] Raskar R, Van Baar J., Chai J.X, "A Low-Cost Projector Mosaic with Fast Registration", ACCV, January 2002.

[15] Raskar R., Van Baar J., Beardsley P., Willwacher T., Rao S., Forlines C., "iLamps: Geometrically Aware and Self-Configuring Projectors", SIGGRAPH**,** 27-31 July, 2003, San Diego.

[16] Z. Zhang, "A flexible new technique for camera calibration", IEEE Transactions on PAMI 22(11):1330-1334, 2000.

[17] Gary R. Bradski, "Computer Vision Face Tracking For Use in a Perceptual User Interface", Intel Tech. Journal 2nd quarter 1998.

[18] http://www.intel.com/research/mrl/research/opencv/

[19] Ramesh Raskar, Paul Beardsley, Jeroen van Baar, Yao Wang, Paul Dietz, Johnny Lee, Darren Leigh, Thomas Willwacher, "RFIG Lamps: Interacting with a Self-Describing World via Photo sensing Wireless Tags and Projectors", Siggraph 2004.

[20] You, Suya, Ulrich Neumann, and Ronald Azuma, "Orientation Tracking for Outdoor Augmented Reality Registration", IEEE Computer Graphics and Applications *19*, 6 (Nov/Dec 1999), 36-42.

[21] Azuma Ronald, Jong Weon Lee, Bolan Jiang, Jun Park, Suya You, and Ulrich Neumann, "Tracking in unprepared environments for augmented reality systems", Computers & Graphics 23, 6 (December 1999), 787-793

[22] Welch, Greg, Gary Bishop, Leandra Vicci, Stephen Brumback, Kurtis Keller, and D'nardo Colucci (2001). "High-Performance Wide-Area Optical Tracking -The HiBall Tracking System," Presence: Teleoperators and Virtual Environments 10(1).

[23] J. W. Lee, S. You, and U. Neumann, "Tracking with Omni-Directional Vision for Outdoor AR Systems," IEEE ACM International Symposium on Mixed and Augmented Reality (ISMAR 2002), pp. 47-56, Darmstadt, Germany, Oct, 2002.

[24] Eric Foxlin and Leonid Naimark, "Miniaturization, Calibration & Accuracy Evaluation of a Hybrid Self-Tracker", IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2003), October 7-10,2003,Tokyo.