IPT-EGVE Symposium (2007)
B. Fröhlich, R. Blach, and R. van Liere (Editors)

Monocular head tracking for desktop virtual environments

O. Korkalo and T. Takala

Telecommunications Software and Multimedia Laboratory
Helsinki University of Technology, Finland

Abstract

We present an approach to head tracking to be applied in desktop virtual environments. The system is able to estimate both the position and orientation of the user's head using only monocular view. In our approach, the edge of the front face of shutter or polarization glasses is detected, and the pose is estimated using algorithm based on planar homographies. The solution is based on marker configuration made up from lines, whose intersection points are used to estimate the pose. Instead of using planar square markers common in augmented reality applications, we take the advantage of the natural shape of the virtual reality glasses. In this paper, we describe our system set-up and detail the steps to implement the algorithm. In addition, we compare the proposed approach to well-known solution. The system performs real-time in a standard laptop computer.

Categories and Subject Descriptors (according to ACM CCS): H.5.1 [Multimedia Information Systems]: Artificial, augmented, and virtual realities

1. Introduction

Pose estimation and tracking are essential parts of virtual reality (VR) and augmented reality (AR) installations. In VR, the pose (i.e. position and orientation) of the user's head has to be estimated at every time step to render the scene into stereoscopic displays correctly. In addition, the user has to be offered different types of input devices to control the system.

Several approaches to tracking have been presented including electromagnetic, acoustic, inertia-based and optical systems. Optical trackers are attractive since they are wireless and accurate. Using off-the-shelf hardware like webcams, it is possible to develop affordable trackers that are competitive with commercial trackers [MJvR03].

Optical trackers can rely on fiducial markers or natural features (markerless tracking). In marker-based approaches, the tracked object is equipped with some easily detectable targets. For example, retro-reflective balls are commonly used in commercial motion capture systems. Another example of marker based tracking is well-known ARToolKit [ART07], which employs square-shaped planar markers for pose calculations needed in AR applications or desktop input devices [WMB03], [BGGH05].

One example of marker-based head tracking for desktop-VRs was contributed by Mulder [MJvR03] et al. They developed a system where a triangular planar marker was attached to the VR-glasses. They used calibrated stereo pair and standard epipolar geometry to estimate the pose of the marker. The other marker-based construction was introduced by Mathieu [Mat05]. In his system, the tracking of the head and a wand-like input device was acquired by estimating the pose of a rigid, three dimensional set of retro reflective balls using monocular video. Duca et al. [DFF07] applied square markers to track hands using a single webcam. They also added lines to user's fingers to get additional information about fingertip positions.

Markerless (head) tracking systems do not require any artificial targets to be attached to the object of interest. Instead, they rely only on other features such as strong corners or lines occurring on the images of the target. One of the early examples of these type of head trackers was introduced by Rekimoto [Rek95]. His system performed using only one camera, but it did not offer all the six rotation and translation parameters. Gorodnichy et al. [GMR02] have more recently presented a system that uses uncalibrated stereo pair and projective vision to solve the problem. These kind of solutions are attractive, since they are more imperceptible compared to marker-based systems. On the other hand, they do not perform as well as marker-based solutions at least in cases, where the goal is an accurate six-degrees-of-freedom tracking with minimal jitter.

In this paper, we present our first steps and approach to the head tracking problem. Our aim is to develop a monocular tracking framework to be used in desktop VR systems. Ideally, it would be accurate, robust and especially jitter-free in order to support a stable view for the user. Our solution employs fiducial markers, but not in a sense of common square shaped AR markers.

2. Tracking framework

2.1. General outline

Our tracking method is based on planar homographies. This general approach to pose estimation was presented to multimedia community by Prince et al. [PXC02], who applied homography-based markerless tracking to AR and mixed reality applications. Based on the work by Zhang [Zha00], Malik [Mal02] applied this tracking method to marker-based uncalibrated AR. We follow partly the path of Malik, but instead of detecting separate corners from the marker, we detect a polyline, and find the best-fit polygon for it. As the equations of the lines of the polygon are known, their intersection points can be calculated in sub-pixel precision. Furthermore, these points are used in pose estimation algorithm.

In general level, our method is as follows:

- Convert acquired color image to monochromatic
- Binarize gray-scale image by thresholding
- Find connected components (blobs) from the binary image
- Filter blobs to detect the marker
- Traverse the edge of the marker and extract the corners
- Fit lines to the edge segments and find the intersection points of the lines in sub-pixel accuracy
- Use intersection points to estimate the homography between the marker and the image plane
- Use homography to estimate the rigid transformation matrix between the marker and the camera
- Refine the solution by non-linear optimization

Next, we discuss the approach in greater detail.

2.2. Feature detection

We added retro reflective tape to the edge of the front-face of the polarization glasses to make it easier to be extracted from the image. We used visible light to illuminate the camera view, but of course, the same results could have been achieved by infrared. As the geometry of the edge of the glasses is known, it is possible to determine the homography between the glasses and the image plane. If we have the information about camera's internal parameterization, it is furthermore possible to calculate the relative orientation and translation of the object and the camera.

We applied simple thresholding to detect the retro reflective tape from the video. The resulting binary image was labeled using fast connected components algorithm [dSB99] in 4-connected sense. All the blobs being smaller than predefined limit were discarded as well as the blobs that had pixels in the image border. Finally, the rest of the blobs were classified by extracting their 4-connected (outer) border, and calculating the number of corners of the edge using k-cosine angle measure [RJ73]. The outer border was selected to represent the edge of the glasses, since the material typically used in polarization glasses reflects the incoming directional light just as the retro reflective material does, and it disturbs the image of the edge from the inside.

To acquire the corner points in sub-pixel precision, we applied line fitting algorithm to the edge pixels that were divided into groups by previously extracted corners. The line fitting procedure followed standard principal component analysis (PCA), where the mean and directional unit vector of the pixels belonging to a certain line are determined. After the line equations were acquired, all the intersection points were calculated. These sub-pixel intersection points are known forehand in marker coordinate frame, and are used to determine the pose. Figure 1 shows the user wearing polarization glasses with their front face detected. The thresholded image and detected blobs are presented in figure 2.

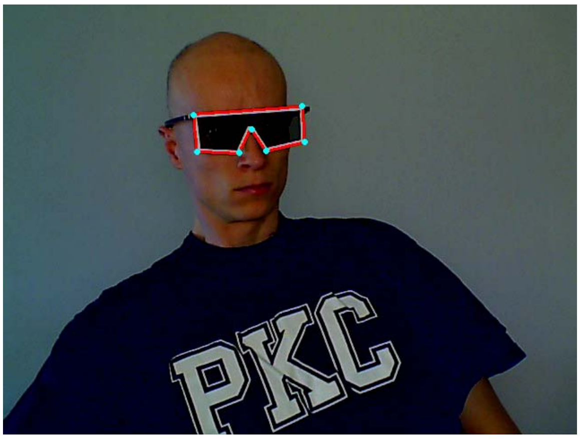

Figure 1: *The edge of the front face of the polarization glasses is detected and its outer border is traversed (highlighted in red). The corner points of the polygon are detected to divide the edge into parts (cyan dots).*

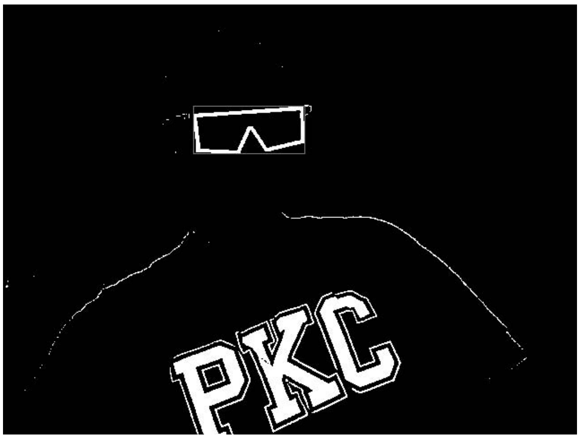

Figure 2: *Binary image after thresholding. The glasses are detected and identified from the other blobs. Corresponding bounding box is overlaid on the image.*

2.3. Camera geometry

We applied simple pinhole camera model [HZ00], where the world coordinate frame points $\mathbf{x}_w$ are transformed and mapped to the image plane of the camera as follows:

$$\mathbf{x}_i = \mathtt{M}\mathbf{x}_w \tag{1}$$

where $\mathtt{M}$ is called the camera calibration matrix and $\mathbf{x}_i$ are the points of the image plane. $\mathtt{M}$ is a combination of the intrinsic and extrinsic parameterizations of the camera: $\mathtt{M} = \mathtt{K}\left[\mathtt{R}|\mathbf{t}\right]$. The extrinsic parameterization describes the orientation and position of the camera relative to the marker. That is, the inverse of the marker pose we are about to estimate if the pose of the camera is fixed. The extrinsic parameterization matrix consist of two components $\mathtt{R}$ and $\mathbf{t}$, which define the rotational and translational parts of the extrinsic parameterization, respectively.

The intrinsic parameterization describes how the points are projected into image plane from the camera coordinate frame:

$$\mathtt{K} = \begin{bmatrix} f_u & s & u_0 \\ 0 & f_b & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where f_u and f_v are the focal lengths in both the horizontal and vertical directions, u_0 and v_0 tells the principal point of the camera, and s defines how skewed the elements of the imaging sensor are.

Typically the pinhole model is augmented with lens distortion model, where non-linear radial and tangential components of the lens distortion occurring on the image plane are approximated with polynomial functions. Although it is possible to extract f_u anf f_v from the estimated homography [Zha00], we calibrated our camera internally using the method implemented in OpenCV [Ope07] to obtain more accurate results.

2.4. Pose from homographies

Homographies are linear mappings between two planes located in three dimensional space. They describe how the points of the first plane are mapped to the second one, and can be expressed in matrix form as follows:

$$\mathbf{x}' = \mathtt{H}\mathbf{x} \tag{3}$$

where $\mathbf{x} = \left[x\ y\ 1\right]^T$ and $\mathbf{x}' = \left[x'\ y'\ 1\right]^T$ are the the homogeneous representations of the points in the first and the second plane, respectively.

The $\mathtt{H}$ can be solved using DLT-method as described in [HZ00], where the minimum number of data points in each image is 4. The more we have corresponding points, the more accurate the results are. In practice, the data has to be normalized before homography estimation as advised in [HZ00]. The implementation of the OpenCV [Ope07] refines the solution by minimizing the reprojection error of the data points, too.

If we know the homography between the planar marker and its image, *and* the internal parameterization of the camera, we can obtain the transformation between the marker and the camera. The extrinsic calibration matrix of the camera can be written as

$$\left[\mathtt{R}|\mathbf{t}\right] = \left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3 \quad \mathbf{t}\right] \tag{4}$$

where $\mathbf{r}_i$ define the rotational part of the transformation matrix and $\mathbf{t}$ describes the translation. In case of a planar marker, the z-values are zero, and we can write

$$\mathbf{x}_i = \mathtt{K}\left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3 \quad \mathbf{t}\right] \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} \tag{5}$$

and thus

$$\mathbf{x}_i = \mathtt{K}\left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}\right] \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{6}$$

and we see, that the points locating on a plane in the world coordinate frame are mapped by 3×3 homography to the image plane:

$$\mathbf{x}_i = \mathtt{H}\mathbf{x}_w \tag{7}$$

Now, if the internal parameterization of the camera ($\mathtt{K}$) is known, we can determine $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{t}$:

$$\left[\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}\right] = \mathtt{K}^{-1}\mathtt{H} \tag{8}$$

The third rotational vector is obviously the cross product of $\mathbf{r}_1$ and $\mathbf{r}_2$: $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$.

The homography-based pose calculation may suffer a significant amount of jitter, and thus, non-linear optimization procedure is needed to achieve more robust tracking. We applied the method implemented in OpenCV, which refines the pose parameterization by minimizing the back-projection error of the image points.

3. Results

We implemented a head tracking system following the principles described in this paper using C++ and Intel's OpenCV [Ope07] libraries. To verify our approach, we compared its static and dynamic accuracy to well-known algorithm of the ARToolKit [ART07]. To carry out the experiments, we printed two markers, one for both methods, on the same sheet of paper (see figure 3). As the transformation matrix between the markers was known, we were able to determine the transformation between the camera and the markers. In ARToolKit case, we used the standard marker coming with the open source software package. For the method proposed here, we used the same planar shape that was used to track the polarization glasses. As the methods apply different lens distortion models, the camera was calibrated twice; once for the ARToolKit tests as advised in [ART07] and once using OpenCV procedures. The camera that was used in the experiments was a standard USB-webcam (Phillips SPC 900 NC) running 30fps@320×240 resolution using standard laptop computer (1.7 GHz Intel Pentium).

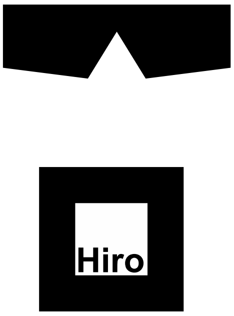

Figure 3: *The experiments were made by printing the ARToolKit marker and the model of the polarization glasses to the same sheet of paper. As the transformation matrix between the markers is known, we were able to compare the results obtained by the two different tracking methods. The width of the polarization glasses was 12.6 cm and height 4.1 cm. The edge length of the ARToolKit marker was 8.0 cm. The origo was set to the center corner of the glasses.*

We conducted three experiments ('Pose 1', 'Pose 2' and 'Pose 3'), where the camera was moved to different positions and orientations. In the experiments, the world origo was set to the middle corner of the glasses, and the camera center distances from the origo (as estimated by ARToolKit) were 63.78 cm, 64.59 cm and 52.64 cm in 'Pose 1', 'Pose 2' and 'Pose 3', respectively. To compare the static accuracies, we calculated the average of the pose estimates over 100 frames. This test was done in good lighting conditions to avoid noise. To measure the jitter, we collected data in low-light using noisy video. The homography-based algorithm was tested with and without non-linear optimization procedure. Figure 4 shows snapshots from all the three views as seen from the camera. The detected lines and corresponding intersection points are highlighted, too.

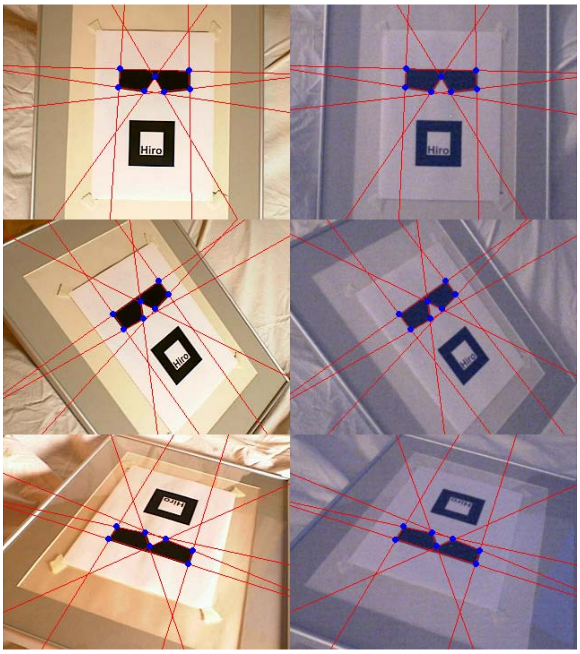

Figure 4: *Snapshots from the video sequences used in the experiments. The experiments are (from top to bottom) 'Pose 1', 'Pose 2' and 'Pose 3'. Right images are taken in low-light.*

The results on the table 1 show the difference between the static accuracies as described above. Table 2 shows the normalized RMS errors in translation vector components and in the third column of the rotation matrix. In addition, figures 5, 6 and 7 show the jitter in every experiment, as the normalized z-value of the translation vector is plotted over time.

There are only negligible differences between the averaged estimates of the pose matrices as can be seen from the table 1. Rotational matrices are almost identical, and the translational parts differ at most 2.3 cm (in 'Pose 2'). However, table 2 shows, that in our experiments the method implemented in ARToolKit suffer more from jitter than the approach we applied. Especially, the first two experiments,

Table 1: *Static accuracy comparisons between the homography-based method (with non-linear optimization, $\mathtt{M}_{hom}$) and the results given by ARToolKit ($\mathtt{M}_{art}$). Table shows $\mathtt{M}_{art}$ in every experiment and the difference between the matrix components of the pose ($\Delta\mathtt{M} = \mathtt{M}_{art} - \mathtt{M}_{hom}$). d_i refer to camera center distances from the origo estimated by ARToolKit. Translations are measured in centimeters.*

	Pose 1, $\mathtt{M}_{art}$				$d_1 = 63.78$ cm		
$\mathbf{r}_1$	0.998	0.016	-0.068	$\Delta\mathbf{r}_1$	-0.002	-0.010	-0.051
$\mathbf{r}_2$	0.028	-0.984	0.176	$\Delta\mathbf{r}_2$	-0.002	-0.014	-0.065
$\mathbf{r}_3$	-0.064	-0.177	-0.982	$\Delta\mathbf{r}_3$	-0.053	0.063	-0.012
$\mathbf{t}$	1.702	-5.436	63.53	$\Delta\mathbf{t}$	-0.154	0.095	1.791
	Pose 2, $\mathtt{M}_{art}$				$d_2 = 64.59$ cm		
$\mathbf{r}_1$	0.671	-0.522	0.526	$\Delta\mathbf{r}_1$	-0.024	0.026	0.062
$\mathbf{r}_2$	-0.591	-0.806	-0.047	$\Delta\mathbf{r}_2$	0.034	-0.058	0.034
$\mathbf{r}_3$	0.448	-0.280	-0.849	$\Delta\mathbf{r}_3$	-0.003	0.000	0.026
$\mathbf{t}$	-0.564	-3.892	64.47	$\Delta\mathbf{t}$	-0.021	0.136	-2.320
	Pose 3, $\mathtt{M}_{art}$				$d_3 = 52.64$ cm		
$\mathbf{r}_1$	-0.945	-0.293	0.147	$\Delta\mathbf{r}_1$	-0.000	-0.003	-0.006
$\mathbf{r}_2$	-0.300	0.597	-0.744	$\Delta\mathbf{r}_2$	-0.006	0.024	-0.029
$\mathbf{r}_3$	0.130	-0.747	-0.652	$\Delta\mathbf{r}_3$	-0.002	0.030	0.024
$\mathbf{t}$	0.981	0.582	52.63	$\Delta\mathbf{t}$	0.001	-0.514	0.099

Table 2: *Normalized RMS error of the pose elements in all the three experiments. Upper values correspond to ARToolKit method, and the values below represent approach applied in this paper. Only the third column of the rotation matrix is shown to save space. Translations are measured in centimeters.*

	r_{31}	r_{32}	r_{33}	t_1	t_2	t_3
Pose 1	0.0179 0.0071	0.0351 0.0083	0.0026 0.0018	0.0183 0.0041	0.0262 0.0063	0.4672 0.0469
Pose 2	0.0129 0.0056	0.0205 0.0064	0.0041 0.0015	0.0719 0.0054	0.0453 0.0073	0.4835 0.1229
Pose 3	0.0027 0.0034	0.0015 0.0017	0.0016 0.0017	0.0276 0.0083	0.0141 0.0042	0.1533 0.0685

where the markers are near facing the camera, were advantageous for homography-based method. The differences decrease in the third experiment as the markers are tilted.

Figures 5, 6 and 7 show also, that the homography-based method has to be refined. For example, if the pose is estimated using homography without non-linear optimization, the jitter of the z-value between two consecutive frames can be over four centimeters (in 'Pose 3'), which is not acceptable result for head-tracking purposes.

4. Discussion and conclusions

We presented our approach to head tracking problem to be used in desktop-like virtual environments. We detected the planar-shaped polarization glasses from the monocular video, and applied general homography-based method for pose estimation. The camera was calibrated internally forehand and the pose estimation results were refined by a non-linear optimization procedure minimizing the back-projection error of the image points.

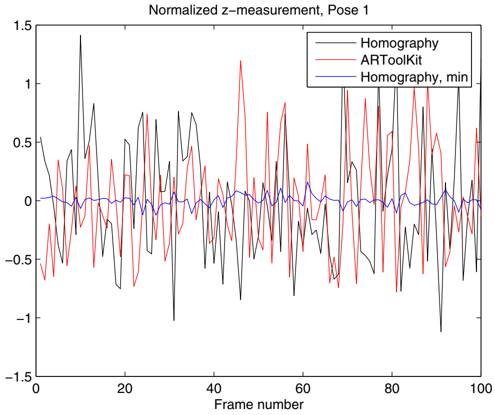

Figure 5: *Comparison of the jitter between the methods in the experiment 'Pose 1'. Lines represent the normalized z-values in different cases. The black line shows the homography-based calculations without any optimization procedures. The red line presents the results given by ARToolKit, and the blue line represents the homography-based algorithm, that is refined by minimizing the re-projection error of the image points.*

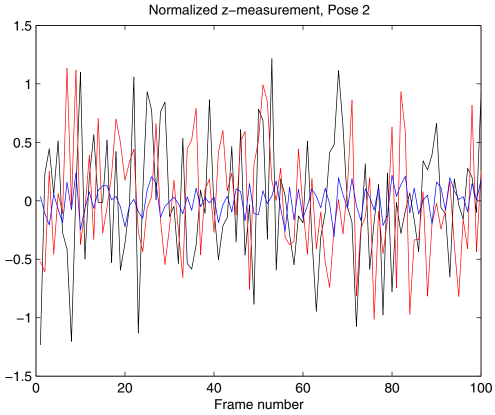

Figure 6: *Comparison of the jitter of the z-values in the second experiment ('Pose 2'). Lines are as in figure 5.*

We compared our approach to well-known algorithm of the ARToolKit. The experiments show, that the proposed method gives at least as good static accuracy, and it does not suffer as much jitter as the ARToolKit does. However, the homography-based method has to be refined by a non-linear optimization to get plausible results.

As the method we used in our work is very flexible and suitable for different tracking tasks, we will continue to work

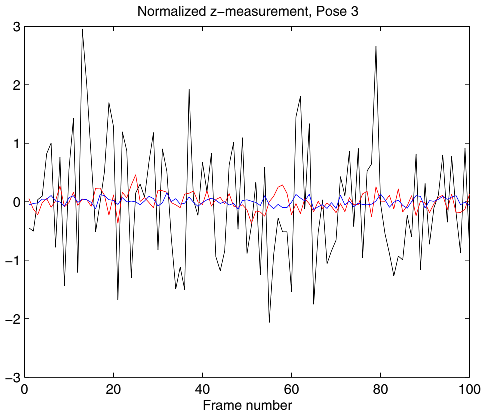

Figure 7: *Comparison of the jitter of the z-values in the third experiment ('Pose 3'). Lines are as in figure 5.*

with other planar-shaped input devices, too. In addition, planar markers can be seen as calibration rigs, and instead of applying separate calibration procedure, we will use our markers to extract the internal parameterization of the cameras on-line. Additional features to our system that will be considered are Kalman filtering to reduce noise and remaining jitter, and other methods to handle partial or total occlusions making tracking more robust.

References

[ART07] ARToolKit project homepage http://www.hitl.washington.edu/artoolkit/

[BGGH05] Billinghurst M., Grasset R., Green R., Haller M.: Inventing the future down under: the human interface technology laboratory new zealand [hit lab nz]. *SIGGRAPH Comput. Graph. 39*, 2 (2005), 18–23.

[DFF07] Duca F., Fredriksson J., Fjeld M.: Real-time 3d hand interaction: Single webcam low-cost approach. In *Workshop at the IEEE Virtual Reality 2007 Confrence: Trends and Issues in Tracking for Virtual Environments* (2007), pp. 1–5.

[dSB99] di Stefano L., Bulgarelli A.: A simple and efficient connected components labeling algorithm. In *ICIAP '99: Proceedings of the 10th International Conference on Image Analysis and Processing* (Washington, DC, USA, 1999), IEEE Computer Society, p. 322.

[GMR02] Gorodnichy D., Malik S., Roth. G.: Affordable 3d face tracking using projective vision. In *Proceedings of International Conference on Vision Interface (VI'2002)* (2002), pp. 383–390.

[HZ00] Hartley R. I., Zisserman A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[Mal02] Malik S.: *Robust Registration of Virtual Objects for Real-time Augmented Reality*. Master's thesis, Carleton University, 2002.

[Mat05] Mathieu H.: The cyclope : A 6 dof optical tracker based on a single camera. In *2nd INTUITION International Workshop, Paris, France, 24-25 Nov 2005* (nov 2005).

[MJvR03] Mulder J. D., Jansen J., van Rhijn A.: An affordable optical head tracking system for desktop vr/ar systems. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003* (New York, NY, USA, 2003), ACM Press, pp. 215–223.

[Ope07] Open source computer vision library from Intel Corporation, http://www.intel.com/technology /computing/opencv/

[PXC02] Prince S. J. D., Xu K., Cheok A. D.: Augmented reality camera tracking with homographies. *IEEE Comput. Graph. Appl. 22*, 6 (2002), 39–45.

[Rek95] Rekimoto J.: A vision-based head tracker for fish tank virtual reality-vr without head gear. In *Virtual Reality Annual International Symposium, 1995. Proceedings.* (1995), pp. 94–100.

[RJ73] Rosenfeld A., Johnston E.: Angle detection on digital curves. *IEEE Trans. Computers 22* (1973), 875–878.

[WMB03] Woods E., Mason P., Billinghurst M.: Magicmouse: an inexpensive 6-degree-of-freedom mouse. In *GRAPHITE '03: Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia* (New York, NY, USA, 2003), ACM Press, pp. 285–286.

[Zha00] Zhang Z.: A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 22*, 11 (2000), 1330–1334.