Interaction-Free Calibration for Optical See-Through Head-Mounted Displays based on 3D Eye Localization

Yuta Itoh*
Technische Universität München

Gudrun Klinker †
Technische Universität München

Abstract

It is a common problem of AR applications that optical see-through head-mounted displays (OST-HMD) move on users' heads or are even temporarily taken off, thus requiring frequent (re)calibrations. If such calibrations involve user interactions, they are time consuming and distract users from their applications. Furthermore, they inject user-dependent errors into the system setup and reduce users' acceptance of OST-HMDs.

To overcome these problems, we present a method that utilizes dynamic 3D eye position measurements from an eye tracker in combination with pre-computed, static display calibration parameters. Our experiments provide a comparison of our calibration with SPAAM (Single Point Active Alignment Method) for several head-display conditions: in the first condition, repeated calibrations are conducted while keeping the display position on the user's head fixed. In the second condition, users take the HMD off and put it back on in between calibrations. The result shows that our new calibration with eye tracking is more stable than repeated SPAAM calibrations. We close with a discussion on potential error sources which should be removed to achieve higher calibration quality.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities

1 Introduction

A crucial issue in AR applications using optical see-through head-mounted displays (OST-HMDs) is to display 3D information from the current viewpoint of the user – and, more particularly, according to the user's eye position, relative to a not quite stable HMD pose on the user's head.

Head-mounted displays (and particularly optical see-through displays) are a critical component of AR systems. They were part of the settings in the early days [30, 4, 5, 28, 3, 27], but subsequently, they have been superseded by video-based AR solutions (using opaque HMDs, smartphones or tablets). There are many reasons for this; especially the limited field of view, contrast issues, as well as the requirement for user-dependent display calibration.

Several display calibration algorithms exist. However, they are cumbersome to use and disrupt the user's AR-experience. In practice, the calibration process is frequently skipped (e.g. when quickly showing an AR-demonstration to a visitor), or performed sub-optimally in order to enhance user convenience.

The issue becomes even more critical during extended use (i.e., in the context of an application lasting more than a few minutes [14]). In principle, the calibration has to be (re)done whenever the position of the HMD changes on the user's head. Such changes are likely to occur frequently. Reasons can be abrupt user motions, as well as situations when the user temporarily removes the glasses, e.g. to rub the eyes, to move through spatially complex terrain or to perform some tasks without the glasses. Quite often, subsequent (re)calibration is skipped since it is too much trouble.

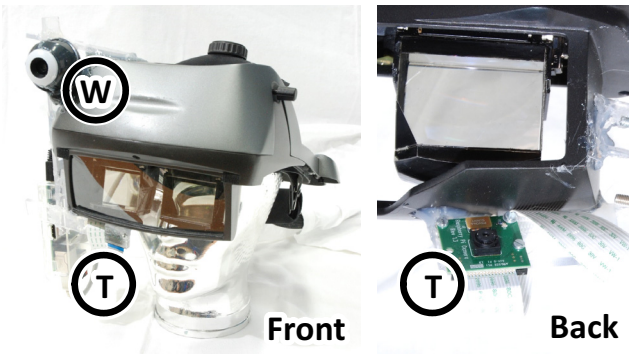

Figure 1: Our technical setup: a world camera, W, and an eye tracking camera, T, are connected to an OST-HMD.

Recently, gaze tracking cameras have become commercially viable. They are small enough to be combined with an OST-HMD. In combination with recent commercial activities to productize AR-glasses for a sizeable market, as well as the emerging trend to build mobile "intelligent" devices that include an increasing number of built-in sensors, we expect that future HMDs will include such cameras. Since such a camera has direct view of (one of) the user's eyes, it generates additional information that can be used to simplify and improve the display calibration process. Yet, the question arises, whether the eye position can be determined precisely and robustly enough to be usable for stable HMD calibrations. After all, small estimation errors of the eye position can have a significant impact on user-perceived offsets between real and virtual objects.

This paper reports on an approach towards combining camera-based eye tracking with HMD calibration (see Fig. 1). An eye tracking camera (T) is rigidly attached to the bottom rim of an HMD, oriented towards one of the user's eyes. A second camera (W) determines the HMD pose within the surrounding world environment. The rigid setup of the two cameras and the HMD is predetermined in an offline calibration process. Combining this static HMD calibration with dynamic eye tracking, we are able to generate world-related augmentations in the HMD even when the HMD is moved on the user's head. We have first, encouraging results that the setup generates a registration quality that is comparable to the state of the art – with potential for further improvements by employing more rigorous offline calibration procedures.

In Section 2, we present and discuss the current state of the art of calibrating viewing setups that include an eye tracker. In Section 3, we present the conceptual foundations of our approach. Section 4 describes a technical implementation of the approach. Section 5 presents first results, evaluating the setup in a real setting and comparing it with already existing display calibration approaches that do not use an eye tracker. The section also discusses the current status, its shortcomings and future steps. The major lessons learned thus far are summarized in the concluding Section 6.

*e-mail: yuta.itoh@in.tum.de
†e-mail: klinker@in.tum.de

2 Related Work

Our work combines research from several different areas that are about to merge.

2.1 Display calibration

It is the goal of OST-HMD calibrations to identify the virtual camera system consisting of a screen of an OST-HMD and a user's eye by estimating a projection between them. Since the eye is a part of the system, the projection contains geometric information of the eye relative to the screen. To obtain such projections, many of the existing methods require user interactions. A very good, detailed discussion is given in Zhou's dissertation [35, 26]. We here provide a brief overview of the most relevant approaches.

First OST-HMD calibration algorithms required users to submit themselves to very complex, pre-arranged physical processes and settings. They were, for instance, requested to physically align their own view with a given world-based reference frame using a boresight approach [3] or to position their heads on a head rest [4, 15].

More conveniently, Tuceryan et al. allowed (or even requested) users to move their heads freely within the physical environment in their Single Point Active Alignment Method (SPAAM) [34]. In their approach, users have to repeatedly align a given physical point of interest (at a known world position) with a cross hair drawn at random locations on their HMD screen. Each such 2D-3D correspondence describes a projection equation, with the head motion being compensated by world-based head tracking. In principle, 6 correspondences suffice to fully determine the 11 parameters of the projection matrix. Yet, to enhance robustness against user errors (imprecise alignments), Tuceryan et al. recommend using at least 12 correspondences. This has become a widely used method for display calibration in AR applications. Genc et al. [7] extended SPAAM for stereo OST-HMDs. To calibrate the left and right eye screens simultaneously, users need to align virtual 3D points to real 3D points by relying on their stereo perceptions.

Yet, even though SPAAM is more convenient than the early methods, it suffers severely from the required large number of user interactions. They not only add physical burden on users, they also degenerate the final calibration quality. Axholt et al. [2, 1] analyzed estimation variance of SPAAM methods and reported that there is a significant effect on parameter estimation variance depending on how the 3D points are distributed in space – primarily in depth. In addition, the authors found that when users are occupied by the alignment task they tend to forget to change their postures and thus collect points within a limited depth range. The paper concluded that users need to be carefully instructed when employing SPAAM. A further source of error stems from the confirmation process for users to indicate when they have achieved a good alignment. Originally, users had to click a button. Maier et al. recently showed that more robust alignments can be achieved, if users are asked to merely hold still for a short amount of time – thereby signaling that they are done [19].

Some research effort has gone into reducing user interaction during the calibration procedure. Easy SPAAM [8, 21] reuses an old projection matrix from a previous SPAAM calibration and adjusts it for a new user eye position. In this method, the change of eye position is approximated by 2D warping of the screen image including scaling – requiring fewer parameters than the full eye pose estimation. Thus, users need to establish only (at least) 2 2D-3D correspondences for online calibration. After Easy SPAAM, Owen et al. [26] proposed DRC (Display-Relative Calibration). The projection can be decomposed into display parameters and an eye position. The former depend only on an OST-HMD and can be determined offline while the latter needs to be determined online. In DRC, the authors formalized this as a two-step calibration process. They described an offline calibration for the display parameters using mechanical jigs and proposed 5 different options for the second, online step, involving varying degrees of simplifying assumptions (ranging from not performing any online calibration over performing a simple warping such as Easy SPAAM to a full 6 DoF eye pose estimation). Hua et al. [10] developed an approach similar to Easy SPAAM for projection-based HMPDs using also this two-step calibration concept. Their difference is that they recalibrate full display parameters in the online step.

2.2 Head-mounted eye tracking

As discussed in the introduction, there are already a number of efforts on wearable head-mounted eye tracking systems including [32, 33, 13, 29, 25]. Those systems are developed mostly to collect and analyze a user's viewing direction for the purpose of gaze analysis. Among them, Tsukada et al. [32, 33] present first-person vision systems that formulate 3D model-based eye tracking using weak perspective projection. Nitschke et al. [24, 25] have built an eye tracking system which reconstructs a user's view from a reflected image on his/her eye. They also employ a 3D eye model and use perspective projection for the eye pose estimation from the image.

2.3 Combinations of HMDs and eye trackers

Some research efforts combine eye trackers with HMDs. Nilsson et al. [23] have developed a video see-through HMD with an eye tracker, and present hands-free interaction based on users' gaze. Also, Lee et al. [16] employ a monocular OST-HMD with a scene camera and an eye tracker for interaction purposes. Due to an HMD positioning issue, their eye tracker tracks the left eye while the graphics are shown on the right screen of the HMD.

Unlike such HMD systems where trackers are additionally attached to the display frames, Hua and Gao [11] have designed a compact eye-tracked HMD (ET-HMD) to which an eye tracker is integrated as a part of the optics system using a free-form prism.

Recently, Makibuchi et al. [20] have developed a calibration method for OST-HMDs. They have attached a camera to an OST-HMD and estimated users' eye positions with a fiducial marker. However, they needed user interaction to calibrate the system.

3 Method

This section describes our calibration algorithm in two subsections: Section 3.2 formulates the overall calibration procedure using a 3D eye position, and Section 3.3 provides more detail on the 3D eye position estimation.

3.1 Notations

Throughout the paper, coordinate systems are treated as right-handed. Bold lower/upper-case letters denote vectors/matrices such as a translation vector $\mathbf{t}$ and a rotation matrix $\mathbf{R}$. $(\cdot)^{\mathrm{T}}$denotes transpose vectors and matrices. If a matrix is explicitly written with its elements, zero elements are left blank for clarity. Lower-case letters represent scalars. Upper-case letters denote coordinate systems such as the world coordinate system W. Given a coordinate system A, a 3D point in A is denoted by using vectors with the coordinate symbol as the lower index: $\mathbf{p}_A$. Given coordinate systems A and B, the relative transformation from A to B is described by $(\mathbf{R}_{AB}, \mathbf{t}_{AB})$ where $\mathbf{R}_{AB}$ and $\mathbf{t}_{AB}$ stand for rotation and translation respectively. Furthermore, explicit transformation of a 3D point $\mathbf{p}_A$ in A to $\mathbf{p}_B$ in B can be written as $\mathbf{p}_B = \mathbf{R}_{AB}\mathbf{p}_A + \mathbf{t}_{AB}$.

3.2 Calibration formulation

This section describes our method in four steps. Fig. 2 illustrates the spatial relationships of our calibration formulation.

In the first step, consider a virtual camera defined by an eye and the virtual screen of an OST-HMD, assuming that the screen with its coordinate system S is planar and located at $\mathbf{t}_{SE_0} := \left[s_x, s_y, s_z\right]^{\mathrm{T}}$ in the camera coordinate system E_0. The camera can be considered as

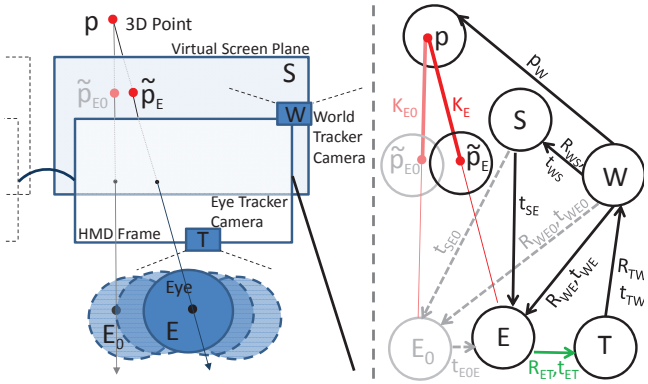

Figure 2: (left) Schematic drawing, illustrating the relevant internal coordinate systems of the right screen S with an eye tracking camera T, a world camera W, and the user's eye E (or E_0). (right) Spatial relationships necessary to determine the projection matrix from 3D object points p onto the screen point $\widetilde{\mathbf{p}}_E$ or $\widetilde{\mathbf{p}}_{E_0}$, respectively.

an off-axis pinhole camera. Now, without loss of generality, assume E_0's z-axis is perpendicular to the virtual screen. The camera is then expressed by the intrinsic camera matrix $\mathbf{K}_{E_0}$ as:

$$\mathbf{K}_{E_0} := \underbrace{\begin{bmatrix} \alpha_x & & \\ & \alpha_y & \\ & & 1 \end{bmatrix}}_{=:\mathbf{A}} \underbrace{\begin{bmatrix} s_z & & -s_x \\ & s_z & -s_y \\ & & 1 \end{bmatrix}}_{=:\mathbf{S}(\mathbf{t}_{SE_0})=\mathbf{S}(s_x,\, s_y,\, s_z)}. \tag{1}$$

$\mathbf{S}(\mathbf{t}_{SE_0})$ transforms 3D points in E_0 to the virtual image screen in real scale, and $\mathbf{A}$ is a diagonal matrix which transforms projected screen points into image pixel points by scaling factors $\{\alpha_x, \alpha_y\}$. Note that $\mathbf{S}(\mathbf{t}_{SE_0})$ is chosen so that $\mathbf{t}_{SE_0}$ is projected to the origin of the image pixel plane. Furthermore, the scaling factors $\{\alpha_x, \alpha_y\}$ are independent of the eye position, whereas $\mathbf{t}_{SE_0}$ is dependent.

Secondly, consider another eye position with its coordinate system E. We can define a new virtual camera consisting of the virtual screen and the new eye position. Following the same concept as for E_0, set the pose of E so that its z-axis is perpendicular to the virtual screen. Then, the transformation from E_0 to E is defined only by the translation $\mathbf{t}_{E_0E} := [t_x, t_y, t_z]^{\mathrm{T}}$. Thus the screen position in E can be written as $\mathbf{t}_{SE} = \mathbf{t}_{E_0E} + \mathbf{t}_{SE_0}$.

From eq. (1), the intrinsic matrix $\mathbf{K}_E$ of the new virtual camera is obtained as

$$\mathbf{K}_E = \mathbf{A}\mathbf{S}(\mathbf{t}_{SE}) = \mathbf{K}_{E_0}\mathbf{S}(t_x/s_z,\, t_y/s_z,\, 1+t_z/s_z). \tag{2}$$

The above shows that we can convert a virtual camera to another given by a new eye position and some display-specific parameters.

Thirdly, consider relocating the above coordinate systems together into a world coordinate system W defined somewhere on the HMD. By recalling the perspective projection of a pinhole camera, we obtain

$$\widetilde{\mathbf{p}}_E = \underbrace{\mathbf{K}_E \begin{bmatrix} \mathbf{R}_{WE} & \mathbf{t}_{WE} \end{bmatrix}}_{=:\mathbf{P}_{WE}(\mathbf{t}_{WE})} \begin{bmatrix} \mathbf{p}_W \\ 1 \end{bmatrix} \tag{3}$$

where $\mathbf{P}_{WE}$ is the projection matrix that projects world points onto the new virtual camera. Note that the rotations from the world to any eye coordinate systems including $\mathbf{R}_{WE}$ are actually identical since they are defined by the rotation of the screen $\mathbf{R}_{WS}$. Thus, it follows that $\mathbf{R}_{WS} = \mathbf{R}_{WE}(= \mathbf{R}_{WE_0})$. Then from eq. (2), eq. (3), and $\mathbf{t}_{SE} = \mathbf{t}_{WE} - \mathbf{t}_{WS}$, we obtain the following,

$$\mathbf{P}_{WE}(\mathbf{t}_{WE}) = \mathbf{A}\mathbf{S}(\mathbf{t}_{WE} - \mathbf{t}_{WS}) \begin{bmatrix} \mathbf{R}_{WS} & \mathbf{t}_{WE} \end{bmatrix} \tag{4}$$

$$= \mathbf{K}_{E_0}\mathbf{S}(t_x/s_z,\, t_y/s_z,\, 1+t_z/s_z) \begin{bmatrix} \mathbf{R}_{WS} & \mathbf{t}_{WE} \end{bmatrix}. \tag{5}$$

Acq.	Condition	Param.	Relationship(From/To)
Online	Required	$\mathbf{t}_{ET}$	Eye(current) / Tracker
Offline	Required	$\mathbf{R}_{WS}$	World / Screen
		$\mathbf{R}_{WT}$	World / Tracker
		$\mathbf{t}_{WT}$	World / Tracker
	Option 1	$\mathbf{t}_{WS}$	World / Screen
		$\alpha_{\{x,y\}}$	Real scale / Img. pixel
	Option 2	$\mathbf{K}_{E_0}$	Eye(previous)
		$\mathbf{t}_{WE_0}$	World / Eye(prev.)
		$[\mathbf{t}_{WS}]_z$	World / Screen

Table 1: A summary of calibration parameters. Option 1 and 2 can be selected depending on available calibration environments.

Eq. (4) does not rely on the old eye position $\mathbf{t}_{WE_0}$. Instead it requires a complete set of display parameters: $\mathbf{t}_{WS}$ and the pixel scaling factors $\{\alpha_x, \alpha_y\}$. On the other hand, eq. (5) does not rely on these parameters, except for $[\mathbf{t}_{WS}]_z$ (since $s_z + t_z = [\mathbf{t}_{WE} - \mathbf{t}_{WS}]_z$), – and it reuses the intrinsic matrix $\mathbf{K}_{E_0}$ from the old eye position. Both cases also require $(\mathbf{R}_{WE}, \mathbf{t}_{WE})$, the pose between the world and the eye.

Finally, consider an eye tracker rigidly mounted on the OST-HMD. Let T be the eye tracker's coordinate system. The tracker then provides the position of the eye E in T as $\mathbf{t}_{ET}$. Then the relationship between the eye and the world can then be written as

$$\mathbf{t}_{WE} = \mathbf{R}_{TE}(\mathbf{t}_{WT} - \mathbf{t}_{ET}) = \mathbf{R}_{WE}\mathbf{R}_{WT}^{\mathrm{T}}(\mathbf{t}_{WT} - \mathbf{t}_{ET}) = \mathbf{R}_{WS}\mathbf{R}_{WT}^{\mathrm{T}}(\mathbf{t}_{WT} - \mathbf{t}_{ET}). \tag{6}$$

Since the eye tracker and the OST-HMD are rigidly connected, $(\mathbf{R}_{WT}, \mathbf{t}_{WT})$ is constant and needs to be calibrated only once. All parameters except for the eye position $\mathbf{t}_{ET}$ relative to $\mathbf{P}_{WE}$ can be determined offline. Therefore, if such offline calibration is conducted beforehand, the system can reconstruct a projection matrix online for a given eye position $\mathbf{t}_{ET}$. Since the position is estimated by the tracker automatically, the system does not require user interaction at run time.

Table 1 summarizes the calibration parameters necessary to compute the projection matrix $\mathbf{P}_{WE}$. Two calibration *Options* exist by choosing either eq. (4) or (5) for the derivation of $\mathbf{P}_{WE}$. Section 4 will present methods to obtain the calibration parameters for real settings.

3.3 Eye position acquisition

This section describes an eye tracking algorithm which estimates $\mathbf{t}_{ET}$, the 3D eye position relative to a tracker. In principle, any eye tracking method that determines the optical center of an eyeball can be used for our calibration system. In the current implementation, we employ a 3D eye position estimation method by Nitschke et al. [24]. For the 2D ellipse extraction, we developed an ellipse fitting method based on work by Swirski et al. [31] together with their open-source iris detector [9, 6].

3.3.1 3D Eye Position Estimation for Perspective Projection

We briefly describe the 3D eye position estimation method of Nitschke et al. [24]. A more elaborated derivation is found in section 2.2.1 of their paper.

Nitschke et al. model the eyeball as two overlapping spheres with different radii and separate centers of curvature. Their method reconstructs the 3D position of the eye and its gaze direction through inverse projection of a 2D ellipse (typically used to describe the eye's limbus of the cornea) from an eye image. They assume that a 3D limbus circle with known constant radius r_L is observed as a 2D ellipse $\mathbf{Q}$ in an image captured by a camera with a known intrinsic matrix $\mathbf{K}$. $\mathbf{Q}$ is a matrix representation of the 2D ellipse with $\tilde{\mathbf{p}}^{\mathrm{T}}\mathbf{Q}\tilde{\mathbf{p}} = 0$ for all homogeneous 2D points $\tilde{\mathbf{p}}$ on the ellipse. Convert the ellipse $\mathbf{Q}$ to the physical scale as $\mathbf{Q}_e := \mathbf{K}^{\mathrm{T}}\mathbf{Q}\mathbf{K}$.

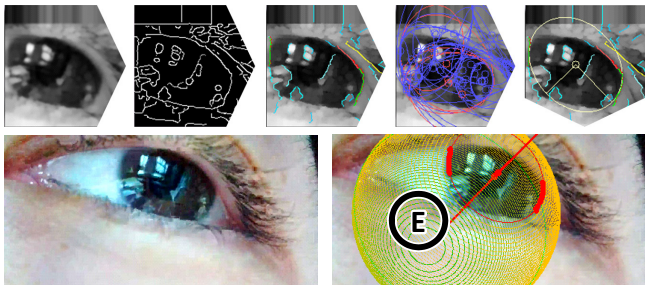

Figure 3: Eye position estimation overview. The images in the first row are output of Algorithm 1. From left to right: smoothing (line 4), Canny edge detection (line 5), edge segmentation (line 7), RANSAC ellipses $\{\mathbf{Q}_{\text{local\_best}}\}$, and the final $\mathbf{Q}$. The second row shows an original image and the visualization of an estimated 3D eyeball with an annotation of the eye coordinate system E.

Then factorize $\mathbf{Q}_e$ by the eigenvalue decomposition as $\mathbf{Q}_e = \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}$ to obtain the eigenvector matrix $\mathbf{U}$ and the eigenvalue matrix Λ. Assume that the three eigenvalues, α, β and γ are ordered such that $\alpha\beta > 0$, $\alpha\gamma < 0$, and $|\alpha| > |\beta|$.

With $g := \sqrt{\frac{\beta-\gamma}{\alpha-\gamma}}$, $h := \sqrt{\frac{\alpha-\beta}{\alpha-\gamma}}$, and undetermined signs $\{s_k\}_{k=1}^{3}$, the 3D circle can be represented by the 3D limbus center position $\mathbf{t}_{LT}$ and the gaze normal vector $\mathbf{n}_T$ in T as:

$$\mathbf{t}_{LT} := \frac{s_3 r_L}{\sqrt{-\alpha\gamma}}\mathbf{U}\begin{bmatrix} s_2 h\gamma \\ 0 \\ -s_1 g\alpha \end{bmatrix}, \mathbf{n}_T := \mathbf{U}\begin{bmatrix} s_2 h \\ 0 \\ -s_1 g \end{bmatrix}. \quad (7)$$

Under the two-sphere eye model, the eye position $\mathbf{t}_{ET}$ can then be expressed for a given constant eye radius r_E as:

$$\mathbf{t}_{ET} = \mathbf{t}_{LT} - \sqrt{r_E^2 - r_L^2}\mathbf{n}_T. \quad (8)$$

Due to $\{s_k\}_{k=1}^{3}$, there are up to $2^3 = 8$ mathematically valid solutions for $\mathbf{t}_{ET}$. Applying the assumptions that the eyeball is in front of the camera and that the gaze vector is oriented toward the camera, the ambiguity can be reduced down to 2. In general, resolving this remaining ambiguity requires additional prior knowledge such as anthropometric properties of the eyeball or constraints from relationships between both eyes. The next section explains our disambiguation approach based on the temporal consistency of a single eyeball position. Fig. 3 top right and bottom right visualize $\mathbf{Q}$ and $\mathbf{t}_{ET}$ respectively. In our implementation, r_L is set to 5.5 [mm] and r_E to 12.6 [mm] according to Nitschke et al. [24].

3.3.2 Eye Position Disambiguation

The 3D eye position estimation method gives true and false estimates as described in the above section. Our basic idea is to disambiguate between them by sampling multiple eye images taken in a row, and finding a consistent combination of the estimates across time. The underlying assumption is that eye positions during a calibration stay the same or at least similar to one another.

Thus the correct estimates from several images should yield eye center positions that are very close to one another. Once a combination is obtained, the final estimate $\mathbf{t}_{ET}$ can be generated by taking their mean or median, or by estimating a time series of them. In the current system, k-means clustering [18] with cluster size $k = 2$ is employed to find the combination. After the clustering, the median of the cluster with the smallest within-class variance is used as the final eye position estimate.

Although the above method is applied in the experiment section, a simple median across all true and false estimates (namely, $k = 1$) also gave a result with a similar tendency.

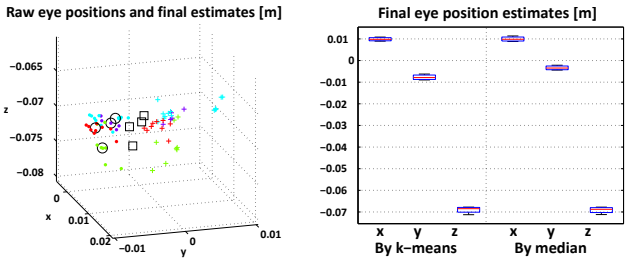

Figure 4: Disambiguation of raw 3D eye positions for 4 eye-image sets collected in a row: 3D visualization of the raw eye positions and final estimates $\mathbf{t}_{ET}$ by two different approaches, and a boxplot of the final estimates.

Fig. 4 shows a set of eye position estimates and final estimates $\mathbf{t}_{ET}$. The sets of estimates were computed from each of 4 sets of eye images taken during 4 SPAAM calibrations (data sequence 1, as explained in Section 5). Each image set consisted of between 10 and 28 images. The k-means-based method and the simple median had a similar estimation variance of $\sigma^2 \approx 1.6e^{-6}$[m].

3.3.3 2D Limbus Ellipse Extraction

Our method to extract 2D limbus ellipses from eye images consists of 2 steps: detection of the limbus image region and extraction of limbus edges. Algorithm 1 describes our procedure.

The first phase (line 2) employs an open-source Haar-like iris detector[1] made by Swirski et al. [31]. Given an eye image, the detector returns an estimated 2D iris center and the surrounding region of interest (ROI) information. We expand the output ROI region from the central iris area to also cover the surrounding limbus area because the limbus radius is static while the iris's is not. The 3D eye position estimation algorithm needs a constant radius.

The second phase (lines 4-28) extracts the limbus ellipse from the ROI. We have adapted the algorithm by Swirski et al. [31] to our purpose of detecting the limbus rather than the iris. The image is smoothed by a morphological opening operation and Gaussian blurring to obtain smoother edges around a limbus. Canny edge detection then computes a binary edge image (lines 4-5).

Next, isolated edge segments E are extracted from the edge image (line 7). In Swirski's code, the star burst algorithm was applied to obtain the iris edge pixels. However, this cannot be easily applied to our limbus case since eyelids often hide the top and bottom edges of the limbus. Thus, we erase edge pixels that are connected in horizontal chains and we subdivide the edge segments recursively (line 7). Due to this heuristic, the algorithm does not create long edges that contain false edge pixels stemming from eyelid-limbus borders. The obtained edges are then sorted by length. Only the top N edges are selected for further processing and the edges are re-fitted at subpixel precision (lines 9-12).

The algorithm subsequently (lines 19-25) uses a RANSAC approach fitting a tentative ellipse to each pair of edges (i.e. $N(N-1)/2$ pairs) and computing a score defined by the number of inliers among all pixels in the edge pair. Suitable heuristic criteria (ellipse properties such as its size, angle, center position etc.) are used to discard inappropriate ellipse candidates $\mathbf{Q}_{\text{local\_best}}$ (line 27). For instance, the eye center should stay inside the image. The top row of the pictures in Fig. 3 shows an example of the series of intermediate outputs of the second step.

4 Technical Setup

For the practical use of calibration methods, it is critical to establish a setup procedure. Following the mathematical formulation of our

[1] http://www.cl.cam.ac.uk/research/rainbow/projects/pupiltracking/

Algorithm 1: Pseudo code for the limbus ellipse estimation.

input : Eye Image $\mathbf{I}$
output: 2D limbus ellipse $\mathbf{Q}$

// PHASE 1
$\mathbf{I}_L \leftarrow$ LIMBUS-DETECTION($\mathbf{I}$) // From [31]
// PHASE 2
$\mathbf{I}_L \leftarrow$ IMAGE-SMOOTHING($\mathbf{I}_L$)
$\mathbf{I}_E \leftarrow$ CANNY-EDGE-DECETION($\mathbf{I}_L$)
// Collect non-horizontal edge segments
$E \leftarrow$ EDGE-SEGMENTATION($\mathbf{I}_E$)
// Chose top N edges by their length
$E \leftarrow$ TOP-LENGTH-EDGES(E)
// Refine points on each edge
foreach $E_i \in E$ **do**
    $E_i \leftarrow$ SUBPIXEL-FITTING(E_i)
$S_{\text{global\_best}} \leftarrow 0$, $\mathbf{Q} \leftarrow null$
// Find Ellipse for each edge pair
foreach $\{E_i, E_j\} \subset E$ *s.t.* $i \neq j$ **do**
    $edge \leftarrow E_i \cup E_j$
    // Ellipse fitting by RANSAC
    $S_{\text{local\_best}} \leftarrow 0$, $\mathbf{Q}_{\text{local\_best}} \leftarrow null$
    repeat $N_{\max}$ **times**
        $points \leftarrow$ RANDOM-SAMPLE($edge$, 5)
        $\mathbf{Q}_k \leftarrow$ ELLIPSE-FITTING($points$)
        // Count the # of inlier points
        $S_k \leftarrow$ INLIER-COUNT($\mathbf{Q}_k, edge$)
        if $S_k < S_{\text{local\_best}}$ **then**
            $S_{\text{local\_best}} \leftarrow S_k$, $\mathbf{Q}_{\text{local\_best}} \leftarrow \mathbf{Q}_k$
    // Update the best ellipse
    if $S_{\text{local\_best}} < S_{\text{global\_best}}$ **then**
        $S_{\text{global\_best}} \leftarrow S_{\text{local\_best}}$, $\mathbf{Q} \leftarrow \mathbf{Q}_{\text{local\_best}}$

calibration algorithm in section 3, this section describes procedures to obtain the required display parameters for the algorithm.

4.1 Hardware setup

We have built an OST-HMD system equipped with an eye tracker, as described below and in Fig. 1. We use nVisor ST60 from NVIS – an OST-HMD with 1280x1080 resolution. Although the display is stereo capable, only the right eye display is used for the current setup. An outward looking Logitech Webcam C200 serves as the world coordinate system W. It provides 640x480-pixel video and is attached to the OST-HMD. The display and the camera are both connected to a commodity laptop.

For the eye tracker T, a Raspberry Pi CSI Camera Module (or RaspiCam) is used. It is connected to a Raspberry Pi board and they are both attached to the OST-HMD (see Fig. 1). The position of the module was chosen to be at the bottom of the right display lens of the OST-HMD so that the module can capture the right eye of an operator easily. Although the module can provide 5MP (2592x1944 pixel) static images maximum, its hardware-encoded H.264 1080p (1920x1280) video stream is sent and resized to HVGA(480x320).

The video stream was transferred by the board to the laptop through a wired local network using gstreamer 1.0. It is received by the laptop using the VLC player. The default focal length of the module is too far for capturing eye images from near distance. Thus its lens component is carefully unsealed and reconfigured for suitable focal length.

4.2 System calibration

To apply our method to an OST-HMD system, such as the one described above, we have to precalibrate the system such that the calibration parameters listed in Table 1 become known.

Figure 5: A multi-marker setup used for calibrating $(\mathbf{R}_{WT}, \mathbf{t}_{WT})$: multi markers only (left), with the OST-HMD (right). The distances between markers and the cameras are less than 10 cm.

In our system, both cameras are calibrated beforehand by using printed checkerboard patterns of different sizes. An open-source MATLAB toolbox[2] is used for the calibration.

Calibration of $\{\mathbf{R}_{WT}, \mathbf{t}_{WT}\}$: The parameters describe the relationship between the eye tracker and the world camera. Since both are optical sensors, visual tracking using fiducial markers can determine their 6 DoF poses relative to the markers. Therefore, by letting them observe several markers that are jointly registered to a common coordinate system, we can compute the required parameters. We employed a multi-marker setup as depicted in Figure 5. Our setup uses printed square markers which are installed on planar world objects and rigidly fixed with respect to one another. The marker positions were measured beforehand to identify their spatial relationship by using the Ubitrack library [22, 12]. Then our OST-HMD system was placed in the environment such that both cameras can see at least one of the markers of the scene. Finally, the relative pose between the tracker and the world camera is derived via the multi-marker coordinate system using the library.

Calibration of $\mathbf{R}_{WS}$: This parameter is the rotation from the world to the virtual screen coordinate system. The easier of two ways is to apply SPAAM for the OST-HMD once - independently of further use of SPAAM. In that case, $\mathbf{R}_{WS}$ can be obtained as one of the calibration parameters.

A second way towards recovering this rotation from the world to the screen coordinate system is to use the display calibration procedure of DRC [26]. The method uses a camera to capture the virtual screen from different view points. Then, it computes the parameters by reconstructing the 3D position of a pattern displayed on the virtual screen. The method is a bit more complicated, but it does not require any user interaction and thus might be more robust.

Calibration of $\{\mathbf{t}_{WS}, \alpha_x, \alpha_y\}$ (Option 1): These parameters are necessary when one uses the eq. (4) for the method. These parameters define the position of OST-HMD's virtual screen relative to the world, and scaling factors that convert points on the virtual screen in real scale into the image as pixels. The DRC display calibration [26] can also provide this information.

Calibration of $\{\mathbf{K}_{E_0}, \mathbf{t}_{WE_0}, [\mathbf{t}_{WS}]_z\}$ (Option 2): These parameters are necessary when one uses the eq. (5) for the method. This option was used instead of the other one for the simplicity of its calibration requirement. A SPAAM calibration can give the $\{\mathbf{K}_{E_0}, \mathbf{t}_{WE_0}\}$. The distance, $[\mathbf{t}_{WS}]_z$, from the world coordinate system to the virtual screen plane is ideally estimated by a method like DRC. Instead, we performed a calibration relying on a manual focus camera. We placed the camera so that it focuses on the virtual screen, then the obtained focal length was further subtracted by manually-measured distance between this camera and the world camera on the display. In our case, the distance was about 78 cm.

[2] http://www.vision.caltech.edu/bouguetj/calib_doc/

5 Experiment

5.1 Design of the test process

As argued in section 1, OST-HMDs are not stable on users' heads during use in real AR applications. Currently, SPAAM-like methods are common practice, requiring users to align 3D targets to 2D points on the display screen. But, for the sake of time, users may compromise, staying on old calibration parameters rather than reperforming a tedious calibration routine (Degraded SPAAM). In sections 3 and 4, we have presented an eye tracking-based approach towards interaction-free display calibration.

We have evaluated the performance of our method (*proposed* condition) compared to SPAAM (*training-error* condition) and to Degraded SPAAM (*test-error* condition). Fig. 6 shows an overview of the process.

5.1.1 Data acquisition

Prior to the evaluation, we acquired a series of data sets. Each set consists of 20 2D-3D point correspondences, with each 3D world point having been manually aligned to a 2D point on the screen (aka SPAAM). The 3D points were distributed across an area of about 90x50x60 cm^3 (width, height, depth) centered around position (-4, 3, -100) [cm] relative to the operator. During this process, we also recorded 15 eye images per 2D-3D point correspondence (i.e., 300 images in total). The top row of Fig. 6 illustrates the step in form of a pink and a green box. The 2D-3D correspondences formed the basis for a SPAAM-based estimation of the display projection matrix (blue box). The eye images were used to compute a series of 3D eye positions using our proposed algorithm (orange box). We call such a data collection session a *block*.

A total of 4 data collection sessions were performed while the HMD was kept as stably as possible on the user's head (top row of Fig. 6). After the first sequence, the OST-HMD was taken off from the head and put back on to simulate a degraded calibration situation. Then, the second set of blocks was collected in the same manner as the first one (indicated in Fig. 6 by variable $N = 1,2$). These two sequences form the ground-truth (GT) data which are the basis for subsequent evaluations of the three evaluation conditions.

Further details of the eye image acquisition process: The eye images were acquired manually and then processed automatically to obtain eye positions. Images for which the algorithm failed to extract eye positions were identified manually. This way, outlier images stemming, e.g., from blinking eyes, motion-blurred eyes, strong inward-light reflection etc. were eliminated manually. Yet, both true and false position estimates were passed to the calibration algorithm and disambiguated automatically, as described in section 3.3.2. An extra visible light source was used due to the limited brightness caused by the OST-HMD.

5.1.2 Data evaluation process

Training-error evaluation: We selected one of the four blocks of a sequence N to conduct a SPAAM calibration. The other three blocks of the same sequence N were subsequently used to evaluate the quality of the SPAAM calibration, using the evaluation procedure of section 5.1.3. Switching the block for the calibration and redoing the same, a total of 24 ($4 \times 3 \times 2$) error measurement sets were obtained. The second row of Fig. 6 shows the procedure of this evaluation.

Test-error evaluation: We used a block from one sequence, N, for the SPAAM calibration and tested the results against the four blocks from the other sequence N' – simulating the Degraded SPAAM condition in which a user continues using the same initial display calibration after the display was moved. This yields 32 ($4 \times 4 \times 2$) sets of error measurements. The third row of Fig. 6 shows this evaluation procedure.

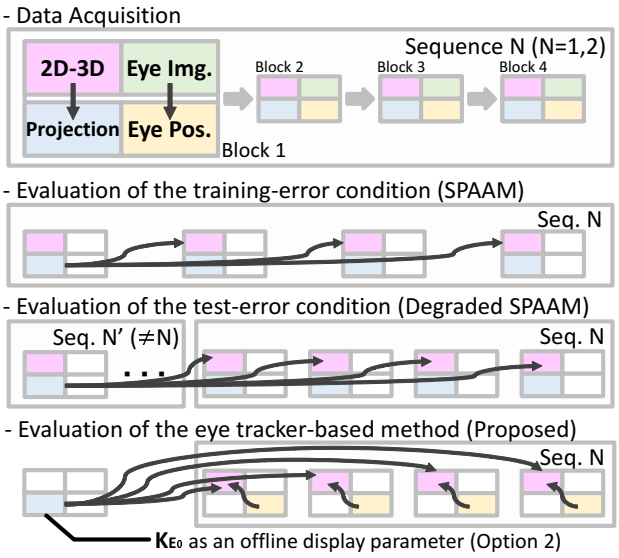

Figure 6: Overview of the experiment: the data acquisition (top row), the training-error condition (second row), the test-error condition (third row), and our proposed condition (bottom). Arrows in the evaluation diagrams indicate which data source from which block and sequence is to be projected and compared w.r.t. which GT data of which other block and sequence. For clarity, only one *Projection* cell is chosen to visualize the arrows.

Evaluation of our proposed method: For all eye image sets in each block, we applied the eye pose estimation method described in Section 3.3. The required precalibration parameters were estimated once, as described in session 4. Option 2 in Table 1 was used to compute projection matrices for our method. Remember that the dataset used for this precalibration was collected in the same manner as for the acquisition of the blocks, yet the dataset was not included in the 8 blocks for a fair comparison. The bottom row of Fig. 6 shows the evaluation procedure.

5.1.3 Evaluation algorithm

Our evaluation aims to determine how well an estimated eye position approximates the true one that existed during the ground-truth data acquisition process. The following indirect and direct error measurements are employed.

2D projection error: This indirect error is considered as an image-based indicator of the estimation quality of the eye position. Firstly, 3D points of the GT data set are reprojected by the estimated projection matrices. Then the error is computed as the average distance between the reprojected points and the GT 2D points. This error is computed for each pair of estimated projections and the GT data set in the three evaluations of the previous section.

3D eye positions: 3D eye positions can be decomposed from the projection matrices by SPAAM. Thus, for each block, we compare the positions with the ones given by our 3D eye position estimation.

5.2 Results

Comparison of 2D projection error: Fig. 7 shows the analysis of the 2D projection error of each algorithm. In the following analysis, we have excluded a single outlier of the SPAAM calibration for fair comparison regarding mean values.

The mean 2D projection error of the proposed method is larger than that of SPAAM, and their difference is statistically significant

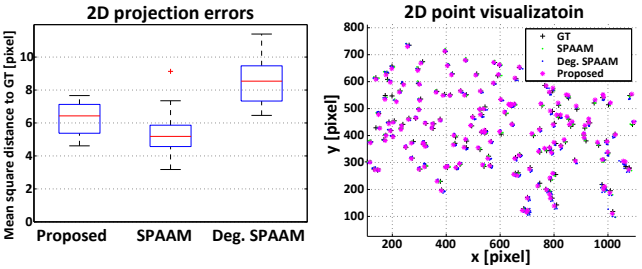

Figure 7: (left) A boxplot of the 2D projection analysis with the y axis showing the mean squared error distance. (right) Plot of both the projected points and the GT points .

in a two-sample t test $\left(p \approx 5.01e^{-5} < 0.05\right)$. However, the variance of the error for SPAAM $\left(\sigma^2 \approx 1.56\right)$ is larger than that for the eye tracker-based method $\left(\sigma^2 \approx 1.09\right)$ even though an outlier was excluded. Thus, in the display-head fixed situation, SPAAM achieves higher quality than our method, yet it might be unstable when users need to recalibrate the system often. This negative effect in SPAAM calibrations is further amplified in Degraded SPAAM. In turn, it can be expected that our proposed method is better than Degraded SPAAM since it is independent of the change of the display position relative to user's head.

The mean error of the proposed method is smaller than that of Degraded SPAAM at a statistically significant level $\left(p \approx 6.07e^{-8} < 0.05\right)$. Besides, the error variance of Degraded SPAAM $\left(\sigma^2 \approx 2.01\right)$ is worse than that of our proposed method. Thus, once users start compromising speed for precision by reusing old calibration parameters, our proposed method can provide more accurate and precise projection results.

A potential reason for the difference between our proposed method and the SPAAM methods can be illustrated in the following analysis of the 3D eye positions.

Comparison of 3D eye positions: Fig. 8 shows an analysis of the estimated eye positions in the world coordinate system, using eye positions from datasets $N = 1$ and 2 (left and right subfigures).

The first row illustrates the distribution of estimated eye positions as boxplots separately for x, y and z. It shows that the SPAAM method causes a large variance along the z-axis – the viewing direction of eye, with about ± 3 cm in each sequence. This error tendency coincides with the SPAAM analysis results by Axholt et al. [2, 1]. On the other hand, eye positions estimated by the proposed method have smaller variance. This feature is preferable since it makes the calibration system more consistent.

The second row shows distances between eye positions and their mean position. The subfigures illustrate a similar trend. The unstable characteristic propagates to the projection matrix at the end, causing 2D projection errors with larger variance in SPAAM than in eye tracker-based calibration. The last row of Fig. 8 visualizes the 3D eye positions showing a much larger variance for SPAAM (green) than for our proposed method (red).

5.3 Discussion

Throughout the experiment, the eye tracker-based method achieved better calibration quality compared to Degraded SPAAM. Yet, it did not work better than SPAAM in terms of the 2D projection error. There are several types of possible reasons for this.

The first is the estimation quality of the offline calibration parameters. For example, in our implementation, the distance from the world camera to the virtual screen was crudely measured by hand with a manual focus camera (see section 4.2). Furthermore, our formulation to compute projection matrices reuses a projection matrix obtained by another calibration method (here: SPAAM). Thus

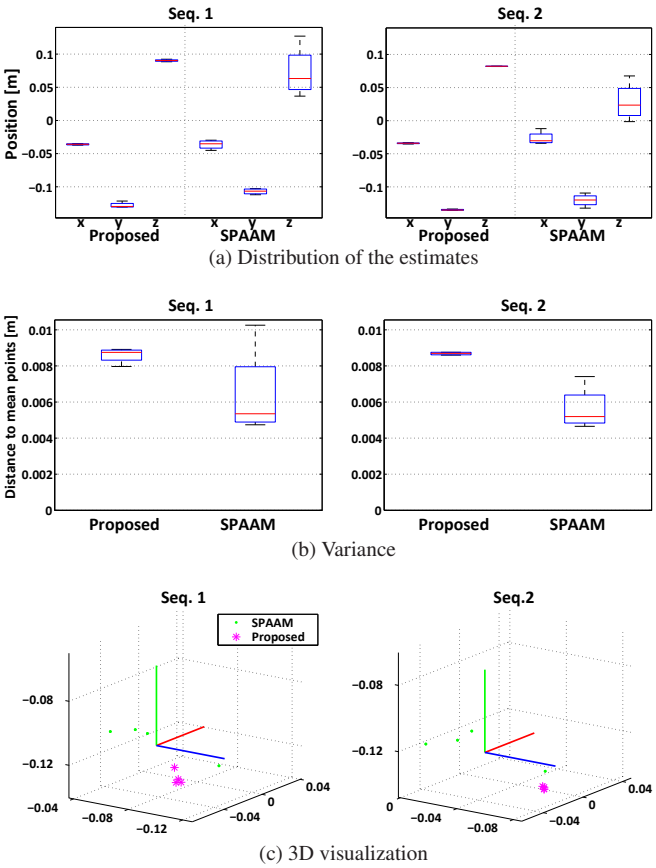

Figure 8: Analysis of 3D eye positions $\mathbf{t}_{WE}$: (a) Boxplots of the positions, (b) Variance of their distance from their mean positions, and (c) a 3D visualization of the points.

the maximal calibration accuracy that our method could achieve for the static head-display setup might be upper bounded by that of SPAAM. This hypothesis can be tested by conducting a complete virtual display calibration. A method such as the DRC [26] can be an option for such an investigation. Also, as mentioned in their work, a virtual display possibly has distortion due to its complicated optics, and thus certain "undistortion" might be required. For example, recent work by Lee and Hua [17] tackles this problem using a nonlinear distortion model from computer vision.

A second possible type of error is related to the eye tracking part. It requires several anatomical parameters related to the human eyeball, such as its radius and the limbus radius. These values should be different for each individual. Furthermore, the simplified 3D eye model used in the algorithm might be insufficient for the application. For example, the assumption that the visual axis of the eye is aligned with the optical axis does not hold in general. This might add a systematic error bias to the eye pose estimation. However, our informal examinations that analyze the noise tolerance of our method by perturbing parameters of the eyeball indicate that they had a lower impact on the projection error than adding noise to the virtual screen parameters.

Thorough investigations of potential error sources need to be conducted to further improve the performance of the interaction-free calibration method.

6 Conclusion and Future Work

This paper presented an interaction-free calibration method for OST-HMDs utilizing 3D eye localization. The method estimates

the projection matrix by using static calibration parameters of an OST-HMD and online eye position measurements. The experiment shows that our calibration is more stable than the SPAAM calibration in terms of the 2D reprojection error and the estimated 3D eye position. Furthermore, our method performs better than a degraded SPAAM setup where users stay on an old set of calibration parameters – which is often the case in AR applications.

Future work directions involve the integration of a precalibration procedure to obtain complete virtual display parameters, the analysis of error sources, and sophistication of the eye tracking system with consideration to real-time capability.

Furthermore, many user-oriented issues arise – how can a system detect that the current calibration has collapsed and needs to be redone? Are there *good* gaze directions that produce better calibration accuracy? And if so, how can the system benefit from that at recalibration time without putting burden on the users? Are even *frame-wise* calibrations possible? These questions would be crucial for making OST-HMDs practically usable.

Acknowledgements

The authors wish to thank Frieder Pankratz, Manuel Huber, Marcus Tönnis, Manfred Röslmair, and all members of the Fachgebiet Augmented Reality (FAR) of TU Munich and the EDUSAFE project. This research has been supported by a Marie Curie Initial Training Network Fellowship of the European Community FP7 Programme under contract number PITN-GA-2012-316919-EDUSAFE.

References

[1] M. Axholt, M. Skoglund, S. O'Connell, M. Cooper, S. Ellis, and A. Ynnerman. Parameter Estimation Variance of the Single Point Active Alignment Method in Optical See-Through Head Mounted Display Calibration. In *VR*, pages 27–34, 2011.

[2] M. Axholt, M. Skoglund, S. Peterson, M. Cooper, T. Schön, F. Gustafsson, A. Ynnerman, and S. Ellis. Optical see-through head mounted display direct linear transformation calibration robustness in the presence of user alignment noise. In *Human Factors and Ergonomics Society 54th Annual Meeting*, pages 2427–2431, 2010.

[3] R. Azuma. *Predictive Tracking for Augmented Reality*. PhD thesis, University of North Carolina, Chapel Hill, NC, 1995.

[4] T. Caudell and D. Mizell. Augmented Reality: An Application of Heads-Up Display Technology to Manual Manufacturing Processes. In *25th Hawaii Int. Conf. on Systems Sciences*, pages 659–669, 1992.

[5] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-Based Augmented Reality. *Commun. ACM*, 36(7):53–62, 1993.

[6] A. Fitzgibbon, M. Pilu, and R. Fisher. Direct Least Square Fitting of Ellipses. *IEEE Trans. PAMI*, 21(5):476–480, 1999.

[7] Y. Genc, F. Sauer, F. Wenzel, M. Tuceryan, and N. Navab. Optical See-Through HMD Calibration: A Stereo Method Validated with a Video See-Through System. In *ISAR*, pages 165–174, 2000.

[8] Y. Genc, M. Tuceryan, and N. Navab. Practical Solutions for Calibration of Optical See-Through Devices. In *ISMAR*, pages 169–175, 2002.

[9] R. Halır and J. Flusser. Numerically Stable Direct Least Squares Fitting of Ellipses. In *Int. Conf. in Central Europe on Computer Graphics and Visualization WSCG*, pages 125–132, 1998.

[10] H. Hua and C. Gao. A Systematic Framework for On-line Calibration of a Head-Mounted Projection Display for Augmented-Reality Systems. *Journal of the Society for Information Display*, 15(11):1–9, 2007.

[11] H. Hua and C. Gao. A Compact, Eye-Tracked Optical See-Through Head-Mounted Display. In *Proc. SPIE*, volume 8288, page 82881F, 2012.

[12] M. Huber, D. Pustka, P. Keitler, F. Echtler, and G. Klinker. A System Architecture for Ubiquitous Tracking Environments. In *ISMAR*, pages 211–214, 2007.

[13] Y. Ishiguro, A. Mujibiya, T. Miyaki, and J. Rekimoto. Aided eyes: Eye Activity Sensing for Daily Life. In *Augmented Human Int. Conf*, page 25. ACM, 2010.

[14] Y. Itoh, F. Pankratz, C. Waechter, and G. Klinker. Calibration of Head-Mounted Finger Tracking to Optical See-Through Head Mounted Display. Demonstraton at ISMAR, 2013.

[15] G. Klinker, D. Stricker, and D. Reiners. Augmented Reality: A Balancing Act Between High Quality and Real-Time Constraints. In *ISMR'99*, pages 325–346, 1999.

[16] J.-Y. Lee, H.-M. Park, S.-H. Lee, T.-E. Kim, and J.-S. Choi. Design and Implementation of an Augmented Reality System Using Gaze Interaction. In *Int. Conf. on Information Science and Applications (ICISA)*, pages 1–8. IEEE, 2011.

[17] S. Lee and H. Hua. A Robust Camera-based Method for Optical Distortion Calibration of Head-Mounted Displays. In *VR*, pages 27–30, 2013.

[18] J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[19] P. Maier, A. Dey, C. Waechter, C. Sandor, M. Tönnis, and G. Klinker. An Empiric Evaluation of Confirmation Methods for Optical See-Through Head-Mounted Display Calibration. In *Joint Virtual Reality Conference of ICAT - EGVE-EuroVR*, pages 73–80, 2012.

[20] N. Makibuchi, H. Kato, and A. Yoneyama. Vision-Based Robust Calibration for Optical See-Through Head-mounted Displays. In *Int. Conf. on Image Processing (ICIP)*, pages 2177–2181. IEEE, 2013.

[21] N. Navab, S. Zokai, Y. Genc, and E. Coelho. An On-line Evaluation System for Optical See-through Augmented Reality. In *VR*, pages 245–246, 2004.

[22] J. Newman, M. Wagner, M. Bauer, A. MacWilliams, T. Pintaric, D. Beyer, D. Pustka, F. Strasser, D. Schmalstieg, and G. Klinker. Ubiquitous Tracking for Augmented Reality. In *ISMAR*, pages 192–201, 2004.

[23] S. Nilsson, T. Gustafsson, and P. Carleberg. Hands Free Interaction with Virtual Information in a Real Environment. In *COGAIN: COmmunication by GAze INteraction*, pages 53–57, 2007.

[24] C. Nitschke, A. Nakazawa, and H. Takemura. Image-based Eye Pose and Reflection Analysis for Advanced Interaction Techniques and Scene Understanding. *Computer Vision and Image Media (CVIM) (Doctoral Theses Session)*, pages 1–16, 2011.

[25] C. Nitschke, A. Nakazawa, and H. Takemura. Corneal Imaging Revisited: An Overview of Corneal Reflection Analysis and Applications. *IPSJ Trans. Computer Vision and Applications*, 5:1–18, 2013.

[26] C. Owen, J. Zhou, A. Tang, and F. Xiao. Display-Relative Calibration for Optical See-Through Head-Mounted Displays. In *ISMAR*, pages 70–78, 2004.

[27] D. Reiners, D. Stricker, G. Klinker, and S. Müller. Augmented Reality for Construction Tasks: Doorlock Assembly. *IWAR*, 1998.

[28] J. Rolland, R. Holloway, and H. Fuchs. A Comparison of Optical and Video See-Through Head-Mounted Displays. *SPIE Telemanipulator and Telepresence Technologies*, 2351:293–307, 1994.

[29] E. Schneider, T. Villgrattner, J. Vockeroth, K. Bartl, S. Kohlbecher, S. Bardins, H. Ulbrich, and T. Brandt. EyeSeeCam: An Eye Movement-Driven Head Camera for the Examination of Natural Visual Exploration. *Annals of the New York Academy of Sciences*, 1164(1):461–467, 2009.

[30] I. Sutherland. The Ultimate Display. In *Congress of the Int. Fed. of Information Processing (IFIP) 65*, volume 2, pages 506–508, 1965.

[31] L. Swirski, A. Bulling, and N. Dodgson. Robust Real-Rime Pupil Tracking in Highly Off-Axis Images. In *Eye Tracking Research and Applications (ETRA)*, pages 173–176, Mar. 2012.

[32] A. Tsukada and T. Kanade. Automatic Acquisition of a 3D Eye Model For a Wearable First-Person Vision Device. In *Eye Tracking Research and Applications (ETRA)*, pages 213–216, 2012.

[33] A. Tsukada, M. Shino, M. Devyver, and T. Kanade. Illumination-free gaze estimation method for first-person vision wearable device. In *ICCV Workshops*, pages 2084–2091, June 2011.

[34] M. Tuceryan and N. Navab. Single Point Active Alignment Method (SPAAM) for Optical See-Through HMD Calibration for AR. In *ISAR*, pages 149–158, 2000.

[35] J. Zhou. *Calibration of Optical See Through Head Mounted Displays for Augmented Reality*. PhD thesis, Michigan State University, 2007.