Appeared in the Proceedings of the IEEE and ACM International Symposium on Augmented Reality, pp. 149-158, Munich, Germany, October, 2000.

# Single point active alignment method (SPAAM) for optical see-through HMD calibration for AR

Mihran Tuceryan
Department of Computer and Information Science
Indiana University Purdue University Indianapolis (IUPUI)
Indianapolis, IN 46202-5132
tuceryan@acm.org

Nassir Navab
Siemens Corporate Research, Inc.
755 College Road East
Princeton, NJ 08540
navab@scr.siemens.com

## Abstract

*Augmented reality (AR) is a technology in which a user's view of the* real *world is enhanced or augmented with additional information generated from a computer model. In order to have a working AR system, the see-through display system must be calibrated so that the graphics is properly rendered. The optical see-through systems present an additional challenge because we do not have access to the image data directly as in video see-through systems.*

*This paper reports on a method we developed for optical see-through head-mounted displays. The method integrates the measurements for the camera and the magnetic tracker which is attached to the camera in order to do the calibration. The calibration is based on the alignment of image points with a single 3D point in the world coordinate system from various viewpoints. The user interaction to do the calibration is extremely easy compared to prior methods, and there is no requirement for keeping the head static while doing the calibration.*

## 1. Introduction

Augmented reality (AR) is a technology in which a user's view of the *real* world is enhanced or augmented with additional information generated from a computer model. The enhancement may take the form of labels, 3D rendered models, or shading modifications. AR allows a user to work with and examine real 3D objects, while receiving additional information about those objects. Computer-aided surgery, repair and maintenance of complex engines, facilities modification, and interior design are some of the target application domains for AR.

In a typical AR system, the view of a real scene is augmented by superposing computer generated graphics on this view such that the generated graphics are properly aligned with real world objects as needed by the application. The graphics are generated from geometric models of both nonexistent (virtual) objects and real objects in the environment. In order for the graphics and the video to align properly, the pose and optical properties of the real and virtual cameras must be the same. The position and orientation of the real and virtual objects in some world coordinate system must also be known. The locations of the geometric models and virtual cameras within the augmented environment may be modified by moving its real counterpart. This is accomplished by tracking the location of the real objects and using this information to update the corresponding transformations within the virtual world. This tracking capability may also be used to manipulate purely virtual objects, ones with no real counterpart, and to locate real objects in the environment. Once these capabilities have been brought together, real objects and computer-generated graphics may be blended together, thus augmenting a dynamic real scene with information stored and processed on a computer.

In order for augmented reality to be effective the real and computer-generated objects must be accurately positioned relative to each other and properties of certain devices must be accurately specified. This implies that certain measurements or *calibrations* need to be made at the start of the system. These calibrations involve measuring the pose of various components such as the trackers, pointers, cameras, etc. What needs to be calibrated in an AR system and how easy or difficult it is to accomplish this depends on the architecture of the particular system and what types of components are used.

There are two major modes of display which determine what types of technical problems arise in augmented reality systems, what the system architecture is, and how these problems are to be solved: (i) video-see-through systems and (ii) optical see-through systems. The calibration issues in a video-see-through system is described in detail elsewhere [29]. We define an optical see-through system as the combination of a see-through head-mounted display and a human eye.

In this paper, we look at the calibration issues in an AR system of the second type, namely, an optical see-through system. In particular, we concentrate on the camera calibration in a *monocular* optical see-through system and describe a method of calibration in such a system.

## 2. Previous Work

Research in augmented reality is a recent but expanding activity. We briefly summarize the research conducted to date in this area. Baudel and Beaudouin-Lafon [3] have looked at the problem of controlling certain objects (e.g., cursors on a presentation screen) through the use of free hand gestures. Feiner et al. [8] have used augmented reality in a laser printer maintenance task. In this example, the augmented reality system aids the user in the steps required to open the printer and replace various parts. Wellner [31] has demonstrated an augmented reality system for office work in the form of virtual desktop on a physical desk. He interacts on this physical desk both with real and virtual documents.

Bajura et al. [2] have used augmented reality in medical applications in which the ultrasound imagery of a patient is superposed on the patient's video image. Once more, the various registration issues, realism, etc. are open research questions which need to be studied and improved. Lorensen et al. [21] use augmented reality system in surgical planning applications. Milgram, Drascic et al. [7, 25] use augmented reality with computer generated stereo graphics to perform telerobotics tasks. Alain Fournier [9] has posed the problems associated with illumination in combining synthetic images with images of real scenes.

Calibration has been an important aspect of research in augmented reality, as well as in other fields, including robotics and computer vision. Camera calibration, in particular, has been studied extensively in the computer vision community (e.g., [20, 22, 32]). Its use in computer graphics, however, has been limited. Deering [6] has explored the methods required to produce accurate high resolution head-tracked stereo display in order to achieve sub-centimeter virtual to physical registration. Azuma and Bishop [1], and Janin et al. [16] describe techniques for calibrating a see-through head-mounted display. Janin's method comes closest to our approach in terms of its context and intent. In this paper, they do consider the tracker in the loop so that the user is free to move during calibration. There are differences between our and their method. The first difference is that we use only a single point in the world for calibration. They use a calibration object with multiple points so that the user has to make an extra decision about picking the calibration point and its image. The use of a single calibration point simplifies the user interaction process which is very important. The second difference is that they use the traditional intrinsic and extrinsic camera parameterization. This results in the non-linear equations to be solved. We use a projection matrix representation of the camera which results in the equations to be solved to be linear. We don't need to extract anything more than the projection matrix because ultimately what we want to do is to project the 3D objects onto the image plane. The projection matrix has also been found to be more accurate and less sensitive to data collection errors [26]. Recently, Kato and Billinghurst describe an interactive camera calibration method [18] that uses multiple points on a grid. Gottschalk and Hughes [11] present a method for auto-calibrating tracking equipment used in AR and VR. Gleicher and Witkin [10] state that their through-the-lens controls may be used to register 3D models with objects in images. Grimson et al. [12] have explored vision techniques to automate the process of registering medical data to a patient's head.

Some researchers have studied the calibration issues relevant to head mounted displays [1, 2, 5, 15, 17]. Others have focused on monitor based approaches [4, 13, 14, 24, 27, 29, 30]. Both approaches can be suitable depending on the demands of the particular application.

Kutulakos et al. have taken a different approach and demonstrated a calibration-free AR system [19]. These uncalibrated systems work in contexts in which using metric information is not necessary and the results are valid only up to a scale factor.

## 3. Overview of the hardware and software

The typical optical see-through AR system hardware is illustrated in Figure 1. In this configuration, the display consists of a pair of *i-glasses*™ head-mounted display (HMD) which can be used both as immersive displays as well as see-through displays by removing a piece of opaque plastic from the front of the display screens. Since our research involves augmented reality systems, we have been using these HMD's as see-through displays permanently. The graphical image is generated by the workstation hardware and displayed on the workstation's monitor which is fed at the same time to the *i-glasses*™ over a VGA port. A 6-degrees-of-freedom (6-DOF) magnetic tracker, which is capable of sensing the three translational and the three rotational degrees of freedom, provides the workstation with continually updated values for the position and orientation of the tracked objects which includes the *i-glasses*™ and a 3D mouse pointing device.

The software is based on the Grasp system that was developed at ECRC for the purposes of writing AR applications. We have added the calibration capabilities to the Grasp software and tested our methods in this environment.

The Grasp software was implemented using the C++ programming language.

## 4. Overview of calibration requirements

In an AR system there are both “real” entities in the user’s environment and virtual entities. Calibration is the process of instantiating parameter values for mathematical models which map the physical environment to internal representations, so that the computer’s internal model matches the physical world. These parameters may be the optical characteristics of a physical camera as well as position and orientation (pose) information of various entities such as the camera, the magnetic trackers, and the various objects.

For an AR system to be successful it is crucial that this calibration process be both complete and accurate. Otherwise, the scene rendered by the computer using the internal model of the world will look unrealistic. For example, objects rendered by the computer using a virtual camera whose intrinsic parameters did not match those of the real camera would result in unrealistic and distorted images which looked out of place compared to the physical world.

The calibration requirements of a video-see-through augmented reality system have been described elsewhere [29]. We briefly summarize the highlights of this system as modified for an optical see-through system to determine its calibration requirements. Figure 2 shows the local coordinate systems relevant for calibration and their relationships in a typical optical-see-through AR system. All the calibration requirements for such a system originate from the fact that all the transformations shown must be known during the operation of the AR system. Some of these transformations are directly read from sensors such as the magnetic trackers. However, some of them need to be estimated through a calibration process and some of them inferred and computed from the rest of the transformations.

The coordinate systems are related to each other by a set of rigid transformations. The central reference is the **World Coordinate System** which is at a fixed and known location relative to the operating environment. During the operation of an AR system, all of the components need to operate in a unified framework which in the case of the Grasp system is the world coordinate system.

The main calibration requirements are the following:

1. Camera calibration (transformation **A** and intrinsic camera parameters).
2. Tracker transmitter calibration (transformation **C**).
3. Tracker mark calibration (transformation **G**). A *mark* is a tracker receiver that is attached to an object being tracked, in this case the *i-glasses*™.

*Camera Calibration* is the process by which the extrinsic camera parameters (location and orientation) as well as the intrinsic camera parameters (focal length, image center, and aspect ratio) are calculated. This process calculates the transformation labeled **A** in Figure 2 as well as the camera intrinsic parameters. In the case of a video-see-through camera calibration system, this would be the estimation of the parameters for the physical camera. In the case of optical see-through AR system, we would estimate the parameters of the virtual camera which models the combined display system formed by the *i-glasses*™ display and the human visual system.

*Tracker Transmitter Calibration* calculates the position and orientation of the tracker’s coordinate system within the world coordinate system (the transformation represented by the arc labeled **C** in Figure 2).

The details of tracker and marker calibrations as well as other calibrations such as object calibration are described in [29]. *In this paper we focus on the camera calibration for the monocular optical-see-through display system*.

## 5. Camera calibration for optical see-through displays

This section details the necessary camera calibration steps for the optical see-through head-mounted display. The camera calibration method described in our previous work on video see-through systems was based on using the relationship between the projected image positions of known 3D points and their 3D positions. From this well known mathematical relationship, the camera parameters were estimated [29]. This assumes that we have access to the picture points (pixel) which we can select and whose image coordinates we can obtain. This can be done in a video-see-through display system because we can always access the image digitized by the video camera and use it to analyze the input images. With an optical see-through system, the images of the scene are formed on the retina of the human user’s eye and we do not have direct access to the image pixels. Therefore, we need to have a different approach to the calibration.

In an earlier paper we described an interactive approach for calibrating an optical see-through AR system [23]. The approach let the user adjust camera parameters interactively until he was satisfied that a 3D model of a calibration jig was aligned properly with the physical calibration jig itself. This method worked but the user interface was cumbersome. In addition, the number of parameters being estimated was too large, and therefore, the interaction did not provide a very intuitive feedback to the user.

The approach described in this paper simplifies both the mathematical model of the camera and the user interaction. First, the user interaction needed to collect the data for the

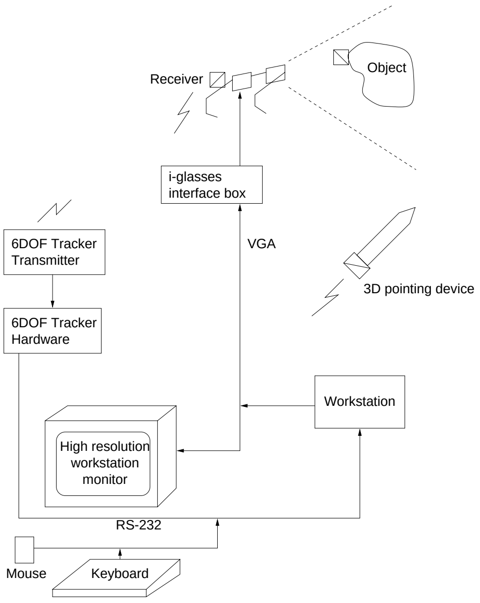

**Figure 1. The hardware diagram of a typical see-through augmented reality system. The see-through displays we use are from *i-glasses*™, and have a limited resolution ($640 \times 480$).**

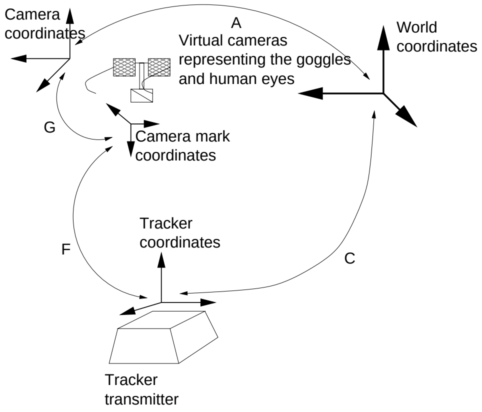

**Figure 2. A simplified version of the coordinate systems that are relevant for the camera calibration of optical see-through systems.**

calibration is a streamlined process and does not impose a great burden on the user. The user's collection of the necessary data to calibrate the display is a very quick and easy process. During this process, the user is not required to have his head fixed and is allowed to move. Therefore, the approach is truly dynamic. Second, the camera model is simplified by not insisting on recovering the intrinsic and extrinsic camera parameters separately.

In the following sections, we first briefly describe the camera model we are using which defines the parameters to be estimated. We then describe the calibration procedure.

### 5.1. Camera model

A simple pinhole model is used for the camera, which defines the basic projective imaging geometry with which the 3D objects are projected onto the 2D image surface. This is an ideal model commonly used in computer graphics and computer vision to capture the imaging geometry. It does not account for certain optical effects (such as non-linear distortions) that are often properties of real cameras. There are different ways of setting up the coordinate systems, and in our model we use a right-handed coordinate system in which the center of projection is at the origin and the image plane is at a distance $f$ (focal length) away from it.

The camera can be modeled by a set of intrinsic and extrinsic parameters. The intrinsic parameters are those that define the optical properties of the camera such as the focal length, the aspect ratio of the pixels, and the location of the image center where the optical axis intersects the image plane. One last intrinsic parameter is the skew of the image plane axes. The extrinsic parameters define the position and orientation (pose) of the camera with respect to some external world coordinate system. The 3D points in the world coordinate system get projected onto the image plane of the camera to form the image points.

The camera transformation that maps 3D world points into 2D image coordinates can be characterized by writing the transformation matrices for the rigid transform defining the camera pose and the projection matrix defining the image formation process.

Let $\mathbf{R}$ and $\mathbf{T}$ represent the camera pose in the world coordinate system, in which $\mathbf{R}$ is a $3 \times 3$ rotation matrix and $\mathbf{T}$ is a $3 \times 1$ translation vector. This rigid transformation can also be written as a $4 \times 4$ homogeneous matrix

$$\mathbf{T}_{pose} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The perspective projection is modeled by a $3 \times 4$ projection matrix given by

$$\mathbf{T}_{proj} = \begin{bmatrix} f_u & \tau & r_0 & 0 \\ 0 & f_v & c_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2}$$

In this matrix $\tau$ is a parameter that models the skew in the image plane coordinate axes.

The overall camera transformation is then given by the product

$$\begin{aligned} \mathbf{T}_{camera} &= \mathbf{T}_{proj}\mathbf{T}_{pose} \\ &= \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \end{aligned}$$

where

$$\begin{aligned} a_{11} &= f_u r_{11} + \tau r_{21} + r_0 r_{31} \\ a_{12} &= f_u r_{12} + \tau r_{22} + r_0 r_{32} \\ a_{13} &= f_u r_{13} + \tau r_{23} + r_0 r_{33} \\ a_{14} &= f_u t_1 + \tau t_2 + r_0 t_3 \\ a_{21} &= f_v r_{21} + c_0 r_{31} \\ a_{22} &= f_v r_{22} c_0 r_{32} \\ a_{23} &= f_v r_{23} + c_0 r_{33} \\ a_{24} &= f_v t_2 + c_0 t_3 \\ a_{31} &= r_{31} \\ a_{32} &= r_{32} \\ a_{33} &= r_{33} \\ a_{34} &= t_3 \end{aligned}$$

## 5.2. Calibration Formulation

The overall projective transformation defined by the camera can be written by a $3 \times 4$ matrix $\mathbf{T}_{camera}$ and the entries $a_{ij}$ of this projection matrix can be estimated directly instead of the actual extrinsic and intrinsic camera parameters.

The estimation of the $3 \times 4$ projection matrix is a standard technique often used in computer vision. The calibration proceeds by collecting a number of 2D image coordinates of known 3D calibration points, and the correspondence between the 3D and 2D coordinates defines a linear system to be solved in terms of the projection matrix entries $a_{ij}$.

There are some modifications to the traditional camera calibration method in our approach which allows the user to move his head during the calibration procedure. In our method, we have a magnetic tracking system that consists of a transmitter and a receiver. The transmitter can be positioned anywhere within the world coordinate system. The receiver is attached to the object being tracked, the head-mounted-display in this case. The tracker system can read (sense) the position and orientation of the receiver in the tracker coordinate system. For convenience we call the receiver the *mark*. Because the receiver is attached rigidly to the HMD, the camera can be defined and calibrated with respect to the mark coordinate system. Therefore, taking this approach, we have the camera transformation fixed and unaffected by the head motion. *This is the reason that the head is allowed to move freely during the calibration procedure.*

The entire setup is summarized in Figure 2 which shows the coordinate systems that are relevant for see-through calibration. In this figure, we see four transforms (**A, C, F**, and **G**) that need to be estimated. The transformation **A** is the traditional $3 \times 4$ projective camera transformation with respect to the world coordinate system that is estimated. **C** is the $4 \times 4$ homogeneous transform matrix that defines the world to tracker rigid transform. That is, **C** is the pose of the tracker transmitted with respect to the world coordinate system. Similarly, **F** is a $4 \times 4$ homogeneous transformation matrix that defines that tracker to mark rigid transform. That is, **F** is the pose of the mark with respect to the tracker transmitter coordinate system. Finally, **G** is the $3 \times 4$ projection matrix that defines the camera transformation with respect to the mark coordinates. The figure can be summarized by the equation

$$\mathbf{A} = \mathbf{GFC} \tag{3}$$

The transformation **A** varies as the HMD moves about, and this movement is captured by the transformation **F** in Equation 3. The other transformations, **G** and **C** are fixed and do not change. They are also estimated by calibration procedures, whereas the transformation **F** is read (measured) directly from the tracker system.

So, to summarize, in order to calibrate the camera (i.e., estimate the transformation **A**) we need to get the image coordinates of known 3D points in the world coordinate system. But **A** is not fixed and varies as the user moves his head. Therefore, unless we want to force the user to keep his head static (an unrealistic assumption) we need to estimate the camera transformation another way. This we do by calibrating the camera in the mark coordinate system (i.e., estimate the transformation **G**) which does not change. In order to accomplish this we take the known 3D calibration point and transform it into the mark coordinate system, then perform the standard camera calibration procedure on the new point. Let $\mathbf{P_W} = [x_W, y_W, z_W, 1]^T$ be the homogeneous coordinates of the known 3D point in the world coordinate system. Let $\mathbf{P_I} = [u, v, s]^T$ be the homogeneous coordinates of its image point. First, we transform the world coordinates to mark coordinates by

$$\mathbf{P_M} = \mathbf{FCP_W} \tag{4}$$

Then we use the $\mathbf{P_M}$ and its image $\mathbf{P_I}$ to estimate the transformation $\mathbf{G}$. The standard projective camera calibration is set up as follows. Let there be $n$ calibration points whose image coordinates we measure. There are 12 parameters of the 3 × 4 projection matrix we need to estimate. But the projection matrix is defined up to a scale factor, therefore, there are really 11 independent parameters that need to be estimated. So, $n$, the number of calibration points to be measured, should be at least 6. Let the $i^{th}$ measurement point have mark coordinates $P_{M,i} = [x_{M,i}, y_{M,i}, z_{M,i}]^T$ and its image point have coordinates $P_{I,i} = [x_i, y_i]^T$. The basic camera equation is given by

$$\begin{pmatrix} u_i \\ v_i \\ w_i \end{pmatrix} = G_{3\times 4} \begin{pmatrix} x_{M,i} \\ y_{M,i} \\ z_{M,i} \\ 1 \end{pmatrix} \quad \text{for } i = 1, \cdots, n \tag{5}$$

Here $[u_i, v_i, w_i]^T$ are the homogeneous image coordinates of the projected point and are related to the image coordinates by

$$\begin{aligned} x_i &= u_i/w_i \\ y_i &= v_i/w_i \end{aligned} \tag{6}$$

Let

$$\mathbf{G} = \begin{bmatrix} g_{11} & g_{12} & g_{13} & g_{14} \\ g_{21} & g_{22} & g_{23} & g_{24} \\ g_{31} & g_{32} & g_{33} & g_{34} \end{bmatrix} \tag{7}$$

Then from Equations 5 and 7, we get

$$\begin{aligned} u_i &= g_{11}x_{M,i} + g_{12}y_{M,i} + g_{13}z_{M,i} + g_{14} \\ v_i &= g_{21}x_{M,i} + g_{22}y_{M,i} + g_{23}z_{M,i} + g_{24} \\ w_i &= g_{31}x_{M,i} + g_{32}y_{M,i} + g_{33}z_{M,i} + g_{34} \end{aligned}$$

Then using equation 6 we get

$$\begin{aligned} & x_i(g_{31}x_{M,i} + g_{32}y_{M,i} + g_{33}z_{M,i} + g_{34}) \\ & \quad = g_{11}x_{M,i} + g_{12}y_{M,i} + g_{13}z_{M,i} + g_{14} \\ & y_i(g_{31}x_{M,i} + g_{32}y_{M,i} + g_{33}z_{M,i} + g_{34}) \\ & \quad = g_{21}x_{M,i} + g_{22}y_{M,i} + g_{23}z_{M,i} + g_{24} \end{aligned}$$

This can be rearranged in terms of the unknown parameter vector $\mathbf{p} = [g_{ij}]^T$ ($\mathbf{p}$ is all the entries of $\mathbf{G}$ put into a column vector) to be estimated into a homogeneous equation to be solved

$$\mathbf{Bp} = \mathbf{0} \tag{8}$$

in which the coefficient matrix $\mathbf{B}$ is given by

$$\mathbf{B} = \begin{bmatrix} \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{M,i} & y_{M,i} & z_{M,i} & 1 & 0 & 0 & 0 & 0 & -x_i x_{M,i} & -x_i y_{M,i} & -x_i z_{M,i} & -x_i \\ 0 & 0 & 0 & 0 & x_{M,i} & y_{M,i} & z_{M,i} & 1 & -y_i x_{M,i} & -y_i y_{M,i} & -y_i z_{M,i} & -y_i \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{9}$$

The matrix $\mathbf{B}$ has $2n$ rows, two rows for each data point, and 12 columns.

Solving this equation gives us the camera matrix $\mathbf{G}$. As we mentioned above there are only 11 independent parameters and the camera equation is valid up to a scale factor. Therefore, to solve the camera equation 8, we estimate the unknown parameter vector $\mathbf{p}$ by minimizing $||\mathbf{Bp}||^2$ such that $||p|| = 1$. This puts a constraint on the scale and reduces the number of parameters to 11. The solution to this constraint minimization is found by finding the eigenvector associated with the smallest eigenvalue [28, Appendix A]. In practice this is done by finding the singular value decomposition (SVD) of the matrix $\mathbf{B}$ given by $\mathbf{B} = \mathbf{UDV}^T$, and the solution is the column of the matrix $\mathbf{V}$ corresponding to the smallest singular value.

### 5.3. Calibration Procedure

In order to get a practical calibration procedure for the see-through displays, the above formulation needs to be converted to a user-friendly procedure. This means that the design of the way the calibration data is collected by the user has to be thought out carefully in order to minimize the burden on the user and the chances of making errors. We have implemented the calibration procedure as follows:

1. The world coordinate system is fixed with respect to the tracker coordinate system by defining the world coordinate system on the tracker transmitter box (see Figure 3). The tracker transmitter calibration is performed as described in [29]. This calibration is then stored and unless the decal put on the transmitter box is replaced or is somehow moved, there is no need to redo this calibration again. Fixing the world coordinate system with respect to the transmitter box has the added advantage that the tracker can be moved at will to any position and the calibration will still stay valid. The world coordinate system could also have been assumed to correspond to the tracker coordinate system by definition, however, this would have been harder to use because we do not know exactly where the tracker coordinate system is on the transmitter box. Therefore, it is better to define the world coordinate system whose location we know and estimate its relation to the unknown tracker coordinate system by a calibration procedure.

2. A *single* point in the world coordinate system is used to collect the calibration data. This single point in the world coordinate system is mapped to many distinct points in the marker coordinate system as the user's head is moved about. This is given by the formula $\mathbf{P_M} = \mathbf{FCP_W}$. Since $\mathbf{F}$ is changing as the head moves, so is, therefore, the coordinates of the point,

$\mathbf{P_M}$ in the marker coordinate system even though $\mathbf{P_W}$ is fixed.

3. The user is presented with cross-hairs on the display and is asked to move about his head until the cross-hair is aligned with the image of the single calibration point as seen by the user (see Figure 4). The user then clicks a button on the 3D mouse and the data is collected for calibration that consists of the image coordinates of the cross-hair $(x_i, y_i)$ and the 3D coordinates of the calibration point in marker coordinates $\mathbf{P_M} = (x_M, y_M, z_M)$. These collected points are then fed into the Equation 8 which is then used to estimate the camera parameters. Since we are trying to estimate 11 parameters and each calibration point gives us two equations, we need at least 6 points for the calibration. However, in order account for the errors and obtain a more robust result, we collect 12 points and use a least squares estimation as stated in Equation 8. Notice here that the more of the tracker volume the user's head covers, the more of possible systematic errors in the tracker measurements will be taken into account in the optimization process. The user is encouraged to move his head around the tracker transmitter as much as possible while collecting the calibration data.

### 5.4. Integrating with OpenGL

Since our camera model now consists of a $3 \times 4$ projection matrix, we have to implement the renderer to use a camera defined by a $3 \times 4$ projection matrix. Unfortunately, OpenGL does not provide an easy interface to do this, so, we had to write a camera class in C++ that is defined by a projection matrix, but uses a number of OpenGL calls to implement the camera. The decision to write a C++ camera class is a result of the fact that all our implementation is done using the GRASP platform developed at ECRC which was written in C++. In fact, the new camera class is implemented as a subclass of the GRASP camera class. In implementing this camera class, we have to be careful that (i) the renderer does not take a performance hit, and (ii) we do not want to extract explicit intrinsic camera parameters for doing this. So, in our implementation we set up the viewing transformation as a Orthographic projection, but push our own constructed viewing matrix onto the transformation stack.

In order to accomplish this, we need to create a $4 \times 4$ matrix that has the clipping plane information from OpenGL as well as our estimated camera projection matrix entries. So, here are the steps to convert it into an OpenGL viewing matrix. First, we make our $3 \times 4$ camera matrix $\mathbf{G}$ into a $4 \times 4$ matrix which has the depth entries in the third row. This is accomplished by multiplying the camera matrix with the transform

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -(f+n) \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

Here, $f$ and $n$ are the *far* and *near* clipping planes used by OpenGL. In addition to the far and near clipping planes, there are the *top* (t), *bottom* (b), *left* (l), and *right* (r) clipping planes, which will be used in the equations below.

Next, we add in the entry that is used for Z-buffer quantization as defined by the matrix:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & f \cdot n \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{11}$$

Next, we define the form of the orthographic projection matrix in OpenGL as defined by the function call *glOrtho(l,r,b,t,n,f)*. This is given by the matrix

$$\begin{bmatrix} 2\ (r-l)^{-1} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & 2\ (t-b)^{-1} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -2\ (f-n)^{-1} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Finally, we obtain the OpenGL viewing matrix by putting all these together as follows:

$$CAM = \left( \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -f-n \\ 0 & 0 & 1 \end{bmatrix} \mathbf{G} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & fn \\ 0 & 0 & 0 & 0 \end{bmatrix} \right)$$
$$\cdot \begin{bmatrix} 2\ (r-l)^{-1} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & 2\ (t-b)^{-1} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -2\ (f-n)^{-1} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

## 6. Experimental verification for calibration

A serious problem with the verification of an optical see-through display calibration is that it is not possible to show how well the model corresponds with the object for a human viewer. We have built a setup in which a camera is put in a mannequin's head behind the *i-glasses*™ displays and the display is recorded. One sample result of the calibration

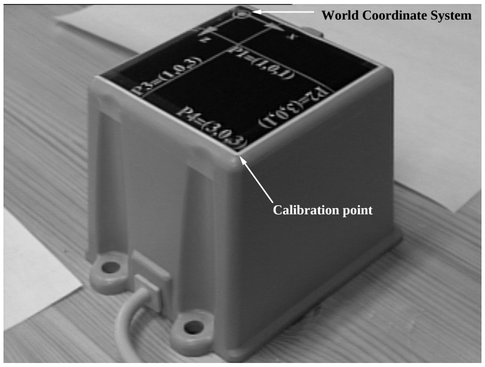

**Figure 3. The world coordinate system is fixed on the tracker transmitter box as shown in this image.**

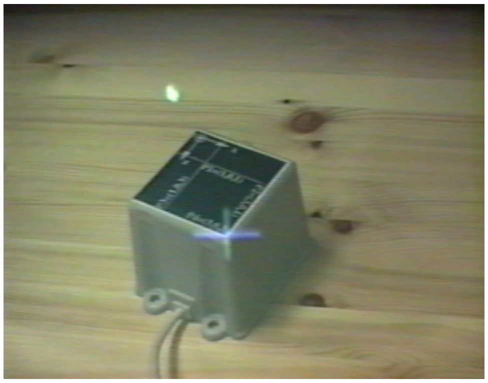

**Figure 4. The calibration procedure requires the user to align a cursor as shown here with a fixed point in the world.**

is shown in Figure 5 in which a model of the calibration pattern defining the world coordinate axes is shown superimposed on the image of the real tracker with the world coordinate system on it. We have tried this calibration method in numerous trials and in all instances the calibration results are very good. The quality of the alignment shown in Figure 5 is representative of the calibration results in these trials. The quality of the calibration results does not change greatly as the head moves around in the world. The only problem is due to the lag in the readings from the magnetic tracker which tends to settle down to the correct position after a certain delay after the head stops moving.

Some of the factors that affect the calibration include the distance of the user's head from the tracker transmitter and how quickly the user clicks the mouse to collect the calibration data. The magnetic tracker we use has a range of about 3 feet and the quality of the sensor readings are not very reliable when the receivers operate near the boundaries of this range. The problems arising from this can be alleviated if an extended range tracker is used which has a larger operational volume (about 10 feet). The second factor that affects the calibration is the lag in the tracker data at the point of collection (i.e., when the mouse is clicked). If the button is clicked too quickly, the tracker data read may not correspond to where the user's head is. We have found that if the user is careful during the calibration, both of these factors can be put under control and the calibration results are good.

Finally, as a demonstration of possible applications, we have implemented an system in which the user interactively places a 3D object in the scene using the 3D pointer. Figure 6 shows an example of such an application in which a virtual lamp is being placed in the scene where the tip of the pointer is placed.

## 7. Conclusion

In this paper, we presented a camera calibration procedure for optical see-through head-mounted displays for augmented reality systems. Because in augmented reality systems we do not have direct access to the image produced on the retina, the procedure needs to use indirect methods to do the calibration. The method presented in this paper uses an interactive method to collect calibration data and it does not require that the user keep his head still.

The resulting calibrations using this method are acceptable within the calibration volume, but the errors increase as the camera moves outside the calibration volume. The quality of the calibrations seem to be better when done on a human head as they are intended, instead of the artificial setting we have for the purposes of collecting quantitative data.

## Acknowledgements

We thank the European Computer-Industry Research Centre (ECRC) for allowing us to use their Grasp system in our research.

## References

[1] R. Azuma and G. Bishop. Improving static and dynamic registration in an optical see-through display. *Computer Graphics*, pages 194–204, July 1994.

[2] M. Bajura, H. Fuchs, and Ohbuchi. Merging virtual objects with the real world: Seeing ultrasound imagery within the patient. *Computer Graphics*, pages 203–210, July 1992.

[3] M. Baudel and M. Beaudouin-Lafon. Charade: Remote control of objects using freehand gestures. *Communications of the ACM*, 37(7):28–35, 1993.

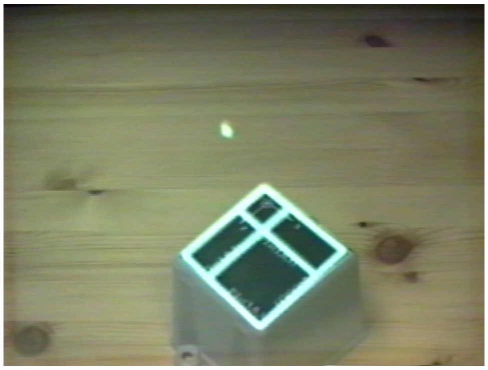

Figure 5. An image captured from the camera in the mannequin's head showing the aligned model of the world coordinate axes with their physical locations.

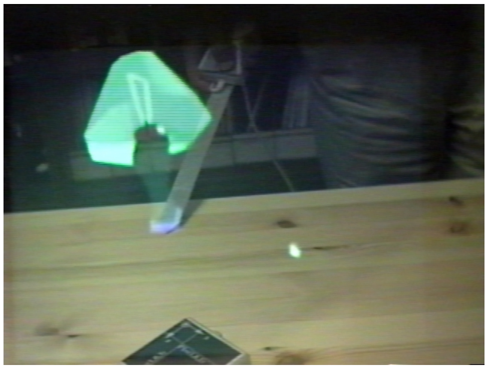

Figure 6. A lamp is being placed in the scene by using the tip of the pointer to indicate the location. This type of interaction works properly, because both the display and the pointer are properly calibrated.

[4] F. Betting, J. Feldmar, N. Ayache, and F. Devernay. A framework for fusing stereo images with volumetric medical images. In *Proc. of the IEEE Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed '95)*, pages 30–39, 1995.

[5] T. Caudell and D. Mizell. Augmented reality: An application of heads-up display technology to manual manufacturing processes. In *Proceedings of the Hawaii International Conference on System Sciences*, pages 659–669, 1992.

[6] M. Deering. High resolution virtual reality. *Computer Graphics*, 26(2):195–202, 1992.

[7] D. Drascic, J. J. Grodski, P. Milgram, K. Ruffo, P. Wong, and S. Zhai. Argos: A display system for augmenting reality. In *Formal video program and proc. of the Conference on Human Factors in Computing Systems (INTERCHI'93)*, page 521, 1993.

[8] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(2):53–62, 1993.

[9] A. Fournier. Illumination problems in computer augmented reality. In *Journée INRIA, Analyse/Synthèse D'Images*, pages 1–21, January 1994.

[10] M. Gleicher and A. Witkin. Through-the-lens camera control. *Computer Graphics*, pages 331–340, July 1992.

[11] S. Gottschalk and J. Hughes. Autocalibration for virtual environments tracking hardware. *Computer Graphics*, pages 65–72, August 1993.

[12] E. Grimson, T. Lozano-Perez, W. M. Wells, G. J. Ettinger, S. J. White, and R. Kikinis. An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 430–436, Seattle, WA, June 1994.

[13] W. Grimson, G. J. Ettinger, S. White, P. Gleason, T. Lozano-Perez, I. W. M. Wells, and R. Kikinis. Evaluating and validating an automated registration system for enhanced reality visualization in surgery. In *Proc. of the IEEE Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed'95)*, pages 3–12, 1995.

[14] C. J. Henri, A. Colchester, J. Zhao, D. Hawkes, D. Hill, and R. L. Evans. Registration of 3D surface data for intra-operative guidance and visualization in frameless stereotactic neurosurgery. In *Proc. of the IEEE Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed'95)*, pages 47–58, 1995.

[15] R. Holloway. *An Analysis of Registration Errors in a See-Through Head-Mounted Display System for Craniofacial Surgery Planning*. PhD thesis, University of North Carolina at Chapel Hill, 1994.

[16] A. Janin, D. Mizell, and T. Caudell. Calibration of head-mounted displays for augmented reality applications. In *Proc. of the Virtual Reality Annual International Symposium (VRAIS'93)*, pages 246–255, 1993.

[17] A. R. Kancherla, J. P. Rolland, D. L. Wright, and G. Burdea. A novel virtual reality tool for teaching dynamic 3D anatomy. In *Proc. of the IEEE Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed'95)*, pages 163–169, 1995.

[18] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality '99*, pages 85–94, San Francisco, CA, October 20–21, 1999.

[19] K. N. Kutulakos and J. R. Vallino. Affine object representations for calibration-free augmented reality. In *Proceedings of the IEEE Virtual Reality Annual International Symposium*, 1996.

[20] R. K. Lenz and R. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-10:713–720, 1988.

[21] W. Lorensen, H. Cline, C. Nafis, R. Kikinis, D. Altobelli, and L. Gleason. Enhancing reality in the operating room. In *Proceedings of the IEEE Conference on Visualization*, pages 410–415, 1993.

[22] S. J. Maybank and O. D. Faugeras. A theory of self calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[23] E. McGarrity and M. Tuceryan. A method for calibrating see-through head-mounted displays for AR. In *2nd International Workshop on Augmented Reality (IWAR '99)*, pages 75–84, San Francisco, CA, October 1999.

[24] J. P. Mellor. Real-time camera calibration for enhanced reality visualizations. In *Proc. of the IEEE Conference on Computer Vision Virtual Reality and Robotics in Medicine (CVRMed'95)*, pages 471–475, 1995.

[25] P. Milgram, S. Zhai, D. Drascic, and J. J. Grodski. Applications of augmented reality for human-robot communication. In *Proc. of the International Conference on Intelligent Robots and Systems (IROS'93)*, pages 1467–1472, 1993.

[26] N. Navab, A. Bani-Hashemi, M. S. Nadar, K. Wiesent, P. Durlak, T. Brunner, K. Barth, and R. Graumann. 3D reconstruction from projection matrices in a C-arm based 3D-angiography system. In *First International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pages 119–129, Cambridge, MA, USA, 1998.

[27] O. Peria, L. Chevalier, A. François-Joubert, J. P. Caravel, S. Dalsoglio, S. Lavallee, and P. Cinquin. Using a 3D position sensor for registration of spect and us images of the kidney. In *Proc. of the IEEE Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed'95)*, pages 23–29, 1995.

[28] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall, 1998.

[29] M. Tuceryan, D. Greer, R. Whitaker, D. Breen, C. Crampton, E. Rose, and K. Ahlers. Calibration requirements and procedures for a monitor-based augmented reality system. *IEEE Transactions on Visualization and Computer Graphics*, 1(3):255–273, September 1995.

[30] M. Uenohara and T. Kanade. Vision-based object registration for real-time image overlay. In *Proc. of the IEEE Conference on Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed'95)*, pages 13–22, 1995.

[31] P. Wellner. Interacting with paper on the digital desk. *Communications of the ACM*, 36(7):87–96, 1993.

[32] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-14(10):965–980, 1992.