

Calibration Requirements and Procedures for a Monitor-Based Augmented Reality System

Mihran Tuceryan Douglas S. Greer Ross T. Whitaker David E. Breen
Chris Crampton Eric Rose Klaus H. Ahlers

European Computer Industry Research Centre
Arabellastraße 17
81925 Munich, Germany

July 6, 1995

Abstract

Augmented reality entails the use of models and their associated renderings to supplement information in a real scene. In order for this information to be relevant or meaningful, the models must be positioned and displayed in such a way that they blend into the real world in terms of alignments, perspectives, illuminations, etc. For practical reasons the information necessary to obtain this realistic blending cannot be known a priori, and cannot be hard-wired into a system. Instead a number of *calibration* procedures are necessary so that the location and parameters of each of the system components are known. In this paper we identify the calibration steps necessary to build a computer model of the real world and then, using the monitor-based augmented reality system developed at ECRC (GRASP) as an example, we describe each of the calibration processes. These processes determine the internal parameters of our imaging devices (scan converter, frame grabber, and video camera), as well as the geometric transformations that relate all of the physical objects of the system to a known world coordinate system.

1 Introduction

Augmented reality (AR) is a technology in which a user's view (or vision) 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. In contrast to virtual reality, augmented reality brings the computer into the "world" of the user rather than immersing the user in the world of the computer. Computer-aided surgery, repair and maintenance of complex engines, facilities modification, and interior design are some of the target application domains for AR. For example, using AR a surgeon may have images of MRI-derived models overlaid on her view of a patient during surgery to help identify malignant tissue to be removed, or sensitive healthy areas to avoid. A mechanic may observe diagnostic or maintenance data while repairing a complicated automobile, locomotive, or aircraft engine. In this second scenario, AR could provide a monitored pointing device which allows the mechanic to identify engine components. Once identified, on-line data for the component, such as schematics, manufacturer's specifications, and repair procedures, may be retrieved and displayed on top of or next to the real part.

In our approach to augmented reality we combine computer-generated graphics with a live video signal to produce an enhanced view of a real scene, which is then displayed on a standard video monitor. In order for monitor-based augmented reality to be effective, the real and computer-generated (virtual) objects of the environment must be accurately positioned relative to each other and the properties of the system's devices must be accurately modeled. Indeed, a key goal of AR is to blend together real-world objects and representations of virtual entities, making them practically indistinguishable to the user. The success of this illusion depends on a number of factors including:

1. the quality of the computer graphics,

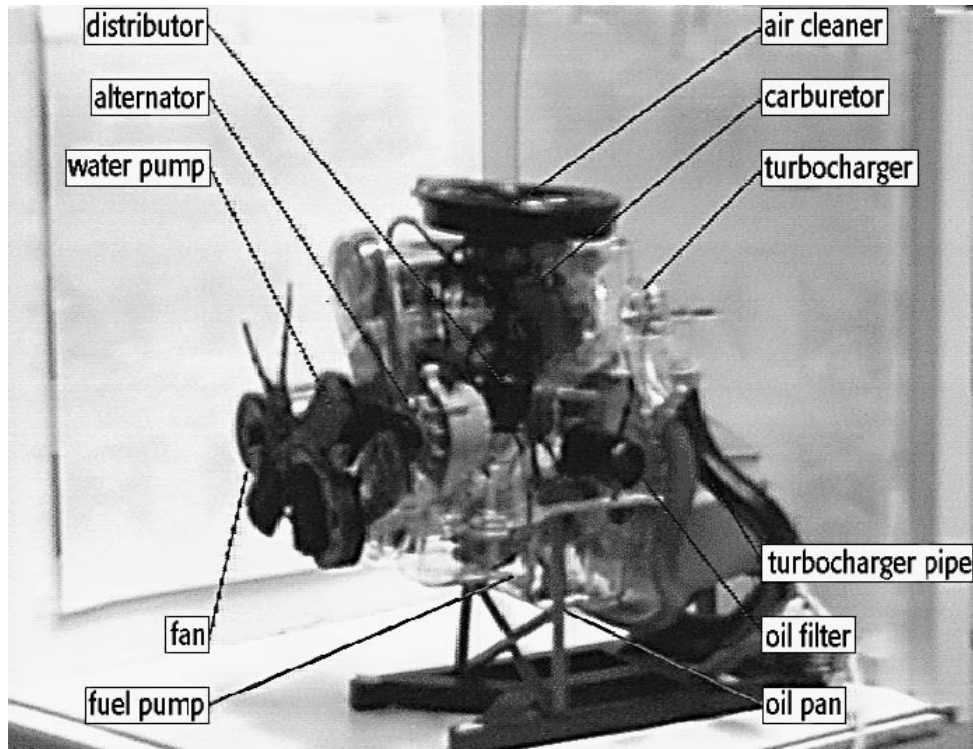


Figure 1: A real model engine annotated in augmented reality.

2. the modeling of the camera motion (tracking) so that the changes in the viewpoint can be properly reflected in the rendered graphics,
3. the modeling of the physical video camera through which the real world is viewed,
4. the modeling of the geometry of the real-life scene, including the *alignment* or *registration* of that geometry with the scene it represents, and
5. the modeling of the scene's lighting and other environmental attributes.

In this paper, we look at requirements 2, 3, and 4 in the above list and their implications in our monitor-based augmented reality system, GRASP [2,3]. These requirements fall into two general categories, the generation of pose information for real and virtual objects, and the determination of device parameters. The complete set of rigorous procedures needed to estimate the parameters of the components comprising an augmented reality system is called *calibration*.

A fundamental feature of augmented reality is the combination of real-world objects and graphical data. The graphical data is generated from geometric models of both virtual and real objects in the environment, as well as other types of information stored in databases. The real-world objects are captured within a video signal from a camera. 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 their real counterparts. 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, those having 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. An example of this is presented in Figure 1, where the visible components of an automobile engine are labeled in augmented reality [33]. As the camera or the engine is moved, the visibility of the various engine components are determined. Lines with annotation tags are drawn to the visible components in order to identify the component. The lines follow the proper component as the engine or the camera moves.

An augmented reality system can be complex, consisting of a variety of computer graphics, computer vision, tracking, and video components. There are usually a number of physical devices which make up each of these components. Each of these devices has parameters that affect the final result of the system. These parameters include distortion characteristics of frame grabbers and scan converters, the optical properties of cameras, the location of tracking equipment, and the full definition of models rendered on graphics hardware. Some of these parameters come in the form of manufacturer specifications (e.g., camera focal length); others can conceivably be measured physically. In most cases, however, these specifications do not correspond directly to the mathematical models of the devices and sometimes physical measurements are not feasible for producing accurate estimates.

Many of these parameters cannot be hard-wired into an augmented reality system. For example, we know from manufacturer specifications that the coordinate system of our magnetic tracking system (Ascension Technologies' Flock of Birds) is roughly centered in the tracker transmitter box, however, we do not know exactly where the coordinate system is located in the box. Hence, even though the location of the transmitter box can be physically measured in relation to our world coordinate system, we still do not know where the tracker coordinate system is located in the world with great accuracy. Therefore, the location of the tracker coordinate system must be estimated using some other procedure which models the uncertainty of the position of the box as well as the uncertainty in the location of the tracker coordinate system with respect to the transmitter box.

The equipment used in typical monitor-based augmented reality applications has all the above mentioned shortcomings of using manufacturer specifications and of using physical measurements. Manufacturer's specifications can be inaccurate and imprecise, and often provided without tolerance information. These nominal parameter values may also change over time with exposure to varying temperatures and vibrations. Physical measurement of the environment and the objects contained within it is also problematic. Manually measuring the real world is time-consuming, unreliable, and even sometimes impossible. In order to address these issues and to develop a flexible and robust approach to calibration, we have decided that all of the device and positioning parameters required to produce a working augmented reality system should be estimated using computational calibration methods.

This paper presents a systematic view of the calibration issues arising in a monitor-based augmented reality system, and describes methods, both previously published and new, for solving each one. We have brought these techniques together into a single, integrated, working system, where they have been tested in a variety of applications. We address the problems of calibrating the camera, the 6 degrees-of-freedom (6-DOF) magnetic tracking system, and the various objects in the real scene. Some of the calibration issues have been studied in other disciplines. For example, camera calibration has been studied extensively in computer vision. We also present several new calibration procedures for augmented reality, namely image, tracker transmitter, pointer, and pointer-based object calibration.

Since the GRASP system is intended for use in a variety of industrial applications, our calibration techniques are designed to be general and robust. More advanced techniques exist for calibration, but we have chosen to investigate methods suitable for a general-purpose augmented reality system, that can be applied to a variety of AR applications. The result of these calibrations is the accurate display and registration of the virtual entities, that only exist in the computer's world, with their real-world counterparts. It is important to the quality of the keying of the real and virtual data—and therefore the seamlessness of the image presented to the user—that this registration be precise and that it be maintainable as the various objects, real and virtual, are moved or otherwise updated.

This paper proceeds by reviewing some of the previous work which highlights the importance of calibration in augmented reality. We then provide an overview of the GRASP system, along with the

calibration steps required to produce augmented reality in Section 3. Section 4 details the procedures and calculations that accomplish the various calibration steps. The paper concludes with examples that demonstrate the results of successful calibrations.

2 Previous Work

Research in augmented reality is a recent but expanding area of research. We briefly summarize the research conducted to date in this area. Baudel and Beaudouin-Lafon [8] 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. [17] 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 [35] has demonstrated an augmented reality system for office work in the form of a virtual desktop on a physical desk. He interacts on this physical desk both with real and virtual documents.

Bajura et al. [5] have used augmented reality in medical applications in which the ultrasound imagery of a patient is superimposed on the patient's video image. Once more, various registration issues, realism, etc. are presented as open research questions which need to be studied and improved. Lorensen et al. [28] use an augmented reality system in surgical planning applications. Milgram, Drascic et al. [14, 32] use augmented reality with computer generated stereo graphics to perform telerobotics tasks. Caudell and Mizell [10] describe the application of augmented reality to manual manufacturing processes. Fournier [18] has posed the problems associated with illumination in combining synthetic images with images of real scenes.

Calibration is 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. The earliest camera calibration methods were formulated as the estimation of the 3×4 perspective transformation¹ matrix (e.g., see [7, pages 481–482]). Some methods also proposed computing the camera parameters from the perspective transformation matrix [19]. The major disadvantage of these methods is that they do not model nonlinear lens distortion effects. Tsai [27, 34] proposed a method in which the camera calibration could proceed in two stages. The first stage computes the 3D position and orientation of the camera (extrinsic parameters). In the second stage he estimates the camera intrinsic parameters, such as the focal length and distortion coefficients. Weng et al. [36] proposed a camera calibration method to estimate intrinsic and extrinsic camera parameters. Weng models tangential lens distortions, as well as radial lens distortions, and estimates the distortion parameters through a separate nonlinear optimization method.

More recent approaches to camera calibration have been formulated in the context of robotics and have concentrated on so called “on-line” or “self-calibration” methods. The representative example of this approach is described by Faugeras [15, 30]. In this approach camera calibration is continuously performed while the camera is in operation (as opposed to off-line before it is put to use). This technique does not utilize a known 3D shape to perform the calibration; instead it is based on being able to track a small number of points from frame to frame. Kumar and Hanson [26] have also studied the effect of errors in camera calibration on other computer vision tasks such as object pose determination. They conclude that for small field of view imaging systems, the errors in image center estimation do not effect the position estimation significantly. The orientation estimation, on the other hand, is significantly affected. Additionally, the incorrect estimation of effective focal length affects only the estimates of the translation along the optical axis in pose computations. Other pose parameters are not affected.

The determination of object pose from known landmark points in camera images is another topic of ongoing investigation in computer vision. Several authors [13, 29] propose iterative solutions that model the projective transformation explicitly. Approximations to the full perspective transformation can make the problem more tractable under limited circumstances. Dementhon and Davis [12] use a weak perspective approximation in order to determine object pose, but the method is not adequate in situations where perspective cues are strong. Horaud et al. [24] propose a paraperspective approximation to improve on the weak perspective approach. Despite these advancements, the determination of fast, general, and robust algorithms for determining the pose of a 3D object is topic of continued study.

The utilization of calibration in computer graphics has depended upon the requirements of particular applications. Deering [11] has explored the methods required to produce accurate high resolution head-

¹We use the term *transformation* to mean any mapping which transforms one kind of data into another in our system. This includes the distortion from the scan converter, the projection of the 3D world onto a 2D image plane thus forming the camera image, as well as the more traditional rigid transformations that map various coordinate systems into each other.

tracked stereo display in order to achieve sub-centimeter virtual to physical registration. Azuma and Bishop [4], and Janin et al. [25] describe techniques for calibrating a see-through head-mounted display. Gottschalk and Hughes [21] present a method for auto-calibrating tracking equipment used in AR and VR. Gleicher and Witkin [20] state that their through-the-lens controls may be used to register 3D models with objects in images. More recently, Bajura and Neumann [6] have addressed the issue of dynamic calibration and registration in augmented reality systems. They use a closed-loop system which measures the registration error in the combined images and tries to correct the 3D pose errors. Grimson et al. [22] have explored vision techniques to automate the process of registering medical data to a patient's head. In a related project, Mellor [31] recently developed a real-time object and camera calibration algorithm that calculates the relationship between the coordinate systems of an object, a geometric model, and the image plane of a camera.

3 Calibration Issues Within The GRASP System

The GRASP system is a platform, developed at ECRC, for research in augmented reality [1,2]. It has been used to develop several prototype AR applications, including a mechanic's assistant [33] and a collaborative tool for interior design [3]. The system provides functionalities in the areas of 3D graphics, image processing, 3D magnetic tracking, and real-time 3D interaction. GRASP's most important feature for augmented reality, the one which differentiates it from an ordinary 3D graphics or virtual reality system, is its collection of calibration techniques.

3.1 System Overview

A schematic of the GRASP hardware configuration is shown in Figure 2. The workstation hardware generates the graphical image and displays it on the workstation's high resolution monitor. A scan converter transforms the graphics displayed on the high resolution monitor into a standard video resolution and format. The scan converter also mixes this generated video signal with the video signal input from the camera via luminance keying. A 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, including the video camera and the pointing device. A frame grabber digitizes video images for processing within the computer during certain operations. The system is currently based on Sun workstations with Flock of Birds magnetic trackers from Ascension Technologies, an Otto scan converter from Folsom Research, and Sun VideoPix frame grabber hardware. The software has been implemented using the C++ programming language. A schematic diagram of the software architecture is shown in Figure 3.

3.2 Calibration Categories

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 "models" which map the physical environment to internal representations, so that the computer's internal model matches the physical world. These models 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. In order to understand which calibration steps are required to establish a complete mapping from the real world to the virtual world, it is first useful to list the various devices and coordinate systems present in a typical application of the GRASP system.

Figure 4 shows the main components of GRASP and the coordinate systems defined by each one. In addition, there are devices such as the camera and the scan converter that have certain intrinsic parameters which affect the resulting augmented image. 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 GRASP is the world coordinate system. Therefore, all the coordinate systems shown in Figure 4 must be tied together, and the transformations relating each coordinate system as well as the intrinsic parameters of any special devices need to be determined. A subset of these transformations are known by direct measurements from devices (e.g., those provided by the magnetic tracking system). Other transformations are known as a result of a calibration procedure. Finally, the remaining transformations are inferred from the set of known transformations. Some of the intrinsic device properties are also known as a result of calibration

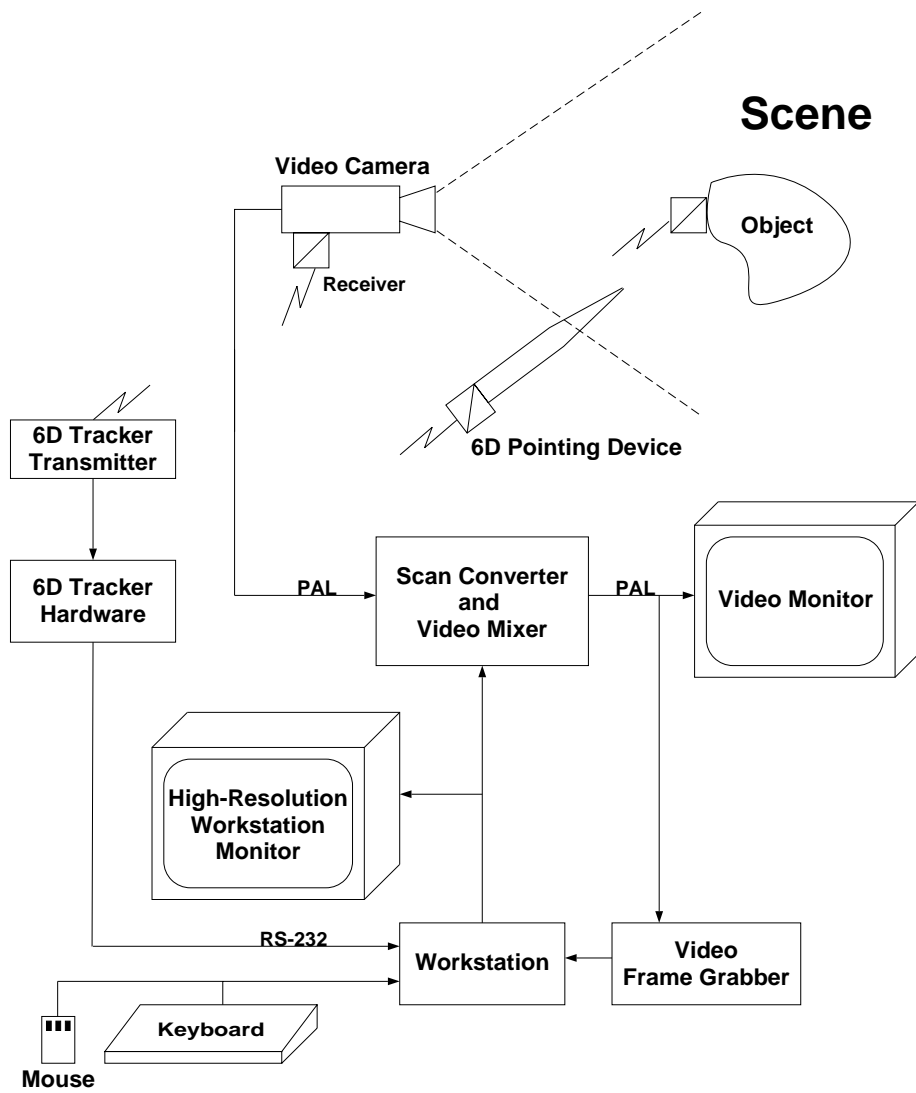


Figure 2: The GRASP system hardware configuration.

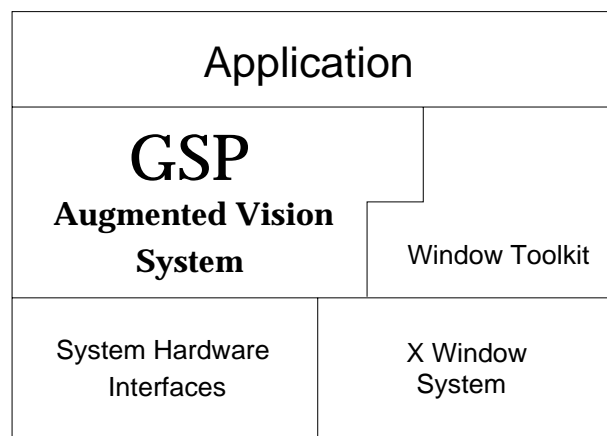


Figure 3: The GRASP system software configuration.

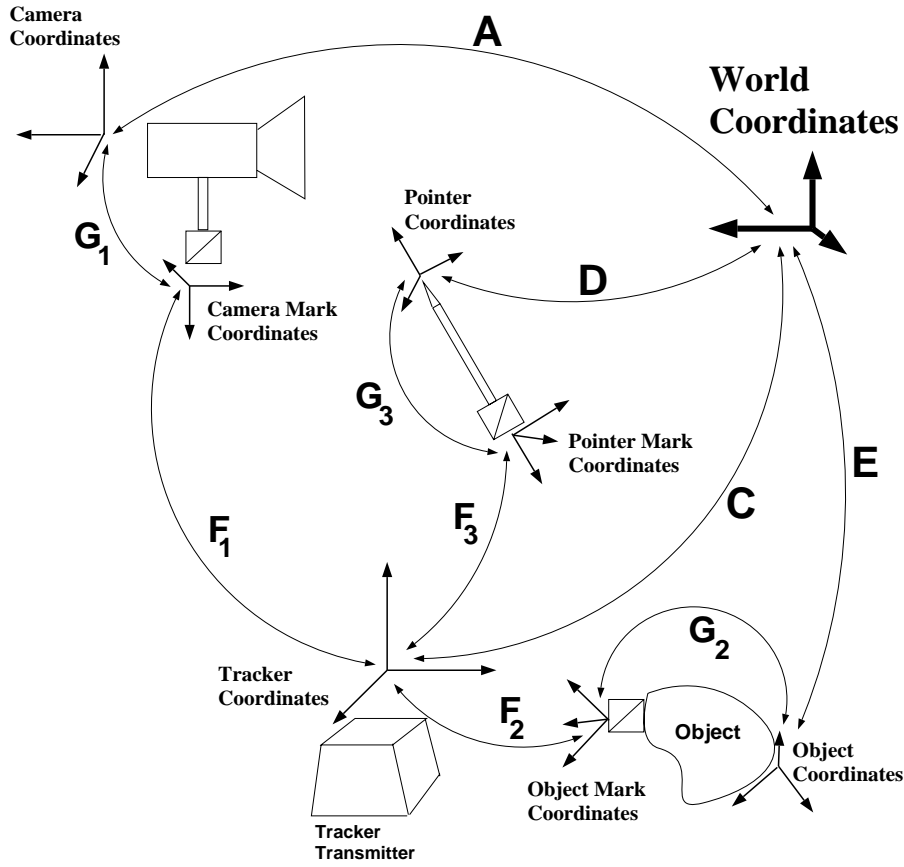


Figure 4: GRASP coordinate systems and their relationships.

procedures (e.g., distortion transformation from the scan converter). An initial goal in the AR system is to fill in all the gaps that might exist in Figure 4 in terms of unknown transformations (in the general sense of the term used in this paper). Any transformation which needs to be known and cannot be directly measured by a sensor device or cannot be inferred is estimated by a calibration procedure. This framework then defines the calibration requirements in an AR system, which is the focus of our paper.

In addition, we can divide the set of transformations into three categories, which affect when a calibration is to be performed. Some of the transformations shown in Figure 4 are fixed once and rarely change again (e.g., G_3); some are fixed at the beginning of each AR session, but one cannot be sure that they will remain unchanged from one session to the next (e.g., C); and some vary during a session. The first category of transformations can be estimated once and need not be calibrated again as long as the rigid attachment does not change. The second category must be estimated at the beginning of each session. Finally, the third category of transformations is estimated at the beginning of each session and also tracked while the program is running.

3.3 Calibration Requirements

Considering this framework, we now give an overview of all the calibration requirements in the GRASP system. Each of these calibration procedures will be presented in detail in the section that follows. Table 1 lists the important transformations, their nature, and how they are obtained.

The location of the camera's origin lies somewhere inside the camera and, in order that to match the virtual camera with the real camera, the location of this coordinate system within the world coordinate system must be calculated. The world-to-camera transformation relating the two coordinate systems is labeled A in the diagram.

The *tracker coordinate system* refers to the reference coordinate system used by the 6-DOF tracker. The world-to-tracker transformation, which is labeled C in Figure 4, defines its relationship to world coordinates. The tracker coordinate system is centered in the transmitter box. During each session, the transmitter box remains at a fixed location with respect to the world coordinate system. However, there

Transformation	Description	Nature	Calculation Process
scan converter parameters	distortion parameters due to scan converter	fixed	image calibration
camera intrinsics	camera optical properties	fixed	camera calibration
\mathbf{A}	world-to-camera	varying	camera calibration
\mathbf{G}_3	pointer-mark-to-pointer	fixed	pointer calibration
\mathbf{C}	world-to-tracker	fixed	tracker transmitter calibration
\mathbf{D}	world-to-pointer	varying	from \mathbf{G}_3 , \mathbf{C} and \mathbf{F}_3
\mathbf{E}	world-to-object	varying	object calibration
\mathbf{F}_i	tracker-to-mark	varying	from tracker system
$\mathbf{G}_1, \mathbf{G}_2$	object-to-mark	fixed	mark calibration (from \mathbf{C} , \mathbf{E} (or \mathbf{A}), and \mathbf{F}_i)

Table 1: Key transformations between GRASP coordinate systems. The receivers of the magnetic tracking system whose positions and orientations are actually read are called *marks*.



Figure 5: The pointer object consists of a wooden wand with a tracker receiver attached to the base and a tapered tip. The tip is considered the “hot-spot” during user interaction.

is no guarantee that the box has not been moved between sessions. The tracker transmitter parameters have been tuned to provide the most accurate readings within the operating range of our applications. Modifying these internal parameters of the transmitter may also significantly alter the coordinate frame of the tracker system. Therefore, transformation \mathbf{C} must be recalibrated at the beginning of each session. The receivers of the magnetic tracking system, whose positions and orientations are actually read, are called *marks*. Tracked objects have a mark rigidly attached to them and the tracking system generates data describing the tracker-to-marker transformations, represented by the arcs labeled \mathbf{F}_1 , \mathbf{F}_2 , and \mathbf{F}_3 in the diagram.

For each real object that is to interact with entities in the virtual world, we need a geometric model to act as a virtual counterpart for the real object. Each model is defined within its own coordinate system. A calibration process calculates the position and orientation of an object within the world coordinate system (transformation \mathbf{E}). Some real objects will be tracked, in which case we need to know the fixed object-to-mark (\mathbf{G}_i) transformation. Once we have the object-to-mark transformation, we can continually update the object-to-world transformation as the tracker-to-mark transformation varies. Note that in some applications, not all world-to-object transformations are varying, because some objects may be static within the scene.

The pointer object takes the form of a wand with a tracking mark (receiver) attached at the base (see Figure 5). The pointer is a particular case of a tracked object with its coordinate system centered on the pointer tip and transformation \mathbf{G}_3 representing the transformation between the coordinate system of the pointer and the tracker mark. Because the pointer is used as part of the process of locating the tracker transmitter within the world coordinate system, the procedure for calibrating the pointer is somewhat

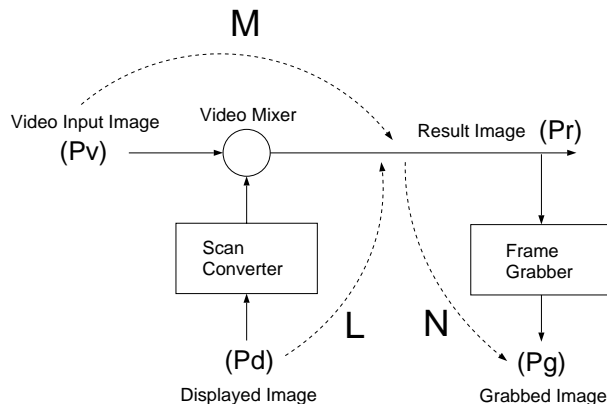


Figure 6: 2D image transformations associated with the image formation process.

different from the other tracked objects.

During the execution of applications, transformations **A**, **D**, and **E** are needed to update the graphics, but the application receives changes in the transformations **F_i** from the magnetic tracker. However, in conjunction with the other transformations calculated during the calibration procedures (**G_i**), it is always possible to calculate updated values for **A**, **D** and **E**.

4 Calibration Procedures and Calculations

The calibration process by which each of the above transformations can be calculated consists of the following steps:

1. image calibration (distortion parameters for scan converter and frame grabber),
2. camera calibration (transformation **A** and intrinsic camera parameters),
3. pointer calibration (transformation **G₃**),
4. tracker transmitter calibration (transformation **C**),
5. object calibration (transformation **E**),
6. tracker mark calibration, one per tracked entity (transformations **G_i**).

We present the details of each of the calibration steps in the following subsections.

4.1 Image Calibration

Image calibration is the process which estimates the 2D image distortion introduced into the system by the scan converter and frame grabber. This information is required because subsequent calibration procedures use grabbed images taken from the scan converter output which include this distortion. These images can be camera images or computer generated images, but in either case, they go through the scan converter (Figure 2).

Figure 6 depicts the path along which the computer generated images are mixed with video images from a physical camera and the final images to be displayed are formed. This resulting image is also the one captured for use in the various calibration procedures described in this paper. More precisely, the scan converter converts the computer image, P_d , to a standard video format and mixes it with the camera image, P_v , to produce the resultant image, P_r . The frame grabber converts the video image, P_r , into an internal format for the computer to produce the grabbed image, P_g .

During this process, two main types of image misalignments may be introduced. The first is a simple misalignment of the two images caused by unequal delays in the horizontal and vertical *sync* processing of the analog video signals. The second source of alignment errors results from the fact that the scan converter and frame grabber do not necessarily preserve the pixel aspect ratios of the images. Because the scan converter can be set to any rectangular region of the workstation screen, the pixel resolution

and image aspect ratio of the computer graphics image may be entirely different from the resulting video and/or the grabbed image. Fortunately, both types of alignment errors can be accurately modeled using a linear transformation without rotation, i.e., a simple scaling and translation in the horizontal and vertical directions. This transformation only captures the distortion from the scan converter and the frame grabber, and none of the distortions introduced into the image by the camera system.

As shown in Figure 6 the distortion in the video input signal caused by the video mixing process is labeled \mathbf{M} . Thus a point p_v in the input image is mapped to the resulting image as $p_r = \mathbf{M}p_v$ (where $\mathbf{M}p_v$ is to be read as a transformation \mathbf{M} applied to the point p_v). Similarly the mappings \mathbf{L} and \mathbf{N} correspond to the distortions caused by the scan converter and frame grabber, respectively. In our system configuration the video mixer and the scan converter are combined in a single device. If p_d is the location of a point on the computer graphics display then its location in the resulting image and the grabbed image is given by:

$$\begin{aligned} p_r &= \mathbf{L}p_d, \\ p_g &= \mathbf{N}\mathbf{L}p_d. \end{aligned} \quad (1)$$

If a point p_v in the input image should correspond with a point p_d in the computer generated graphics then the composition process must insure that

$$\mathbf{M}p_v = \mathbf{L}p_d. \quad (2)$$

During the image calibration process a test image is produced on the display and corresponding points are identified in the grabbed image. This provides a means to empirically measure the distortion function ($\mathbf{N}\mathbf{L}$).

During the camera calibration process a physical calibration pattern is placed in front of the camera and the positions of known points in the calibration pattern are measured in the grabbed image. These data points are then mapped back into the displayed image using the inverse of the function determined during image calibration, i.e., $(\mathbf{N}\mathbf{L})^{-1}$. Consequently, after the camera calibration has been completed and the computer graphics model set up correctly, the following relationship will hold:

$$\begin{aligned} p_d &= (\mathbf{N}\mathbf{L})^{-1}\mathbf{N}\mathbf{M}p_v \\ &= \mathbf{L}^{-1}\mathbf{N}^{-1}\mathbf{N}\mathbf{M}p_v \\ &= \mathbf{L}^{-1}\mathbf{M}p_v. \end{aligned} \quad (3)$$

Applying transformation \mathbf{L} to both sides of Equation 3, we see that after the camera calibration, Equation 2, will be satisfied and therefore the resulting composite image will be correctly aligned.

4.1.1 Image Model

A combination of a two dimensional scaling and translation transformations models the distortion ($\mathbf{N}\mathbf{L}$ in Figure 6) resulting from combining the camera video signal and the computer generated graphics. Let the grabbed video image be defined by a $p_g = (c, r)$ coordinate system and the computer-generated graphical image be defined by a $p_d = (x, y)$ coordinate system as shown in Figure 7. The two are related by:

$$\begin{aligned} c &= k_x x + t_x, \\ r &= k_y y + t_y. \end{aligned} \quad (4)$$

4.1.2 Calibration Procedure

An image calibration procedure estimates the image distortion parameters described above, (k_x, k_y, t_x, t_y) . The calibration steps are:

- send a specially created image containing a marked pattern through the scan converter,
- grab the resulting video image using the frame grabber,
- locate two *known* points within the grabbed image (Figure 7),

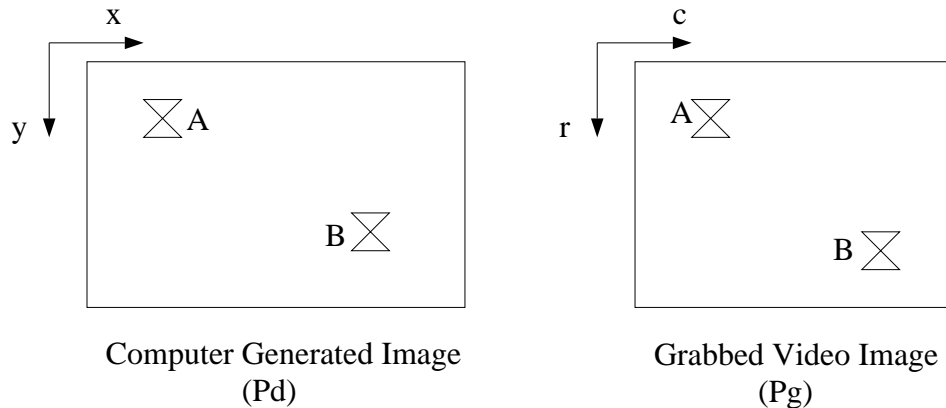


Figure 7: The relationship between the computer-generated and the displayed video images. The distortion is modeled as a two-dimensional scaling and translation transformation.

- calculate the translational and scale factor relationship between the locations of the points in the original image and their location in the resulting image.

we have implemented a manual procedure requiring the user to pick the points in the resulting image, and a robust automatic procedure, which locates known landmarks in the grabbed image using a geometric-fitting process.

The four quantities modeling the image distortion are then estimated as follows. Suppose the known positions of the two points, A and B, are (x_A, y_A) and (x_B, y_B) and their positions displayed on the screen are (c_A, r_A) and (c_B, r_B) . Then the estimates of the parameters are given by (estimations are indicated by a tilde):

$$\tilde{k}_x = (c_B - c_A)/(x_B - x_A), \quad (5)$$

$$\tilde{k}_y = (r_B - r_A)/(y_B - y_A), \quad (6)$$

$$\tilde{t}_x = c_A - \tilde{k}_x x_A, \quad (7)$$

$$\tilde{t}_y = r_A - \tilde{k}_y y_A. \quad (8)$$

4.2 Camera Calibration

Camera calibration is the process which calculates the extrinsic (pose) and intrinsic parameters (focal length, image center, and aspect ratio) of the camera. We assume that the intrinsic parameters of the camera remain fixed during an augmented reality session. The camera's extrinsic parameters may be tracked and updated. We first give the mathematical model of the camera and then describe the calibration procedure.

4.2.1 Camera Model

A simple pinhole model (Figure 8) is used to represent the camera, and 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 nonlinear 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 image of a point in 3D is formed by passing a ray through the point and the center of projection, O_C , and then by computing the intersection of this ray with the image plane. The equations for this case are obtained by using similar triangles,

$$u = fx/z \quad \text{and} \quad v = fy/z. \quad (9)$$

In addition, the camera model used for the generation of the graphical images and for the calibration of the camera has to work with the pixel coordinate system on the display device, which must be accounted

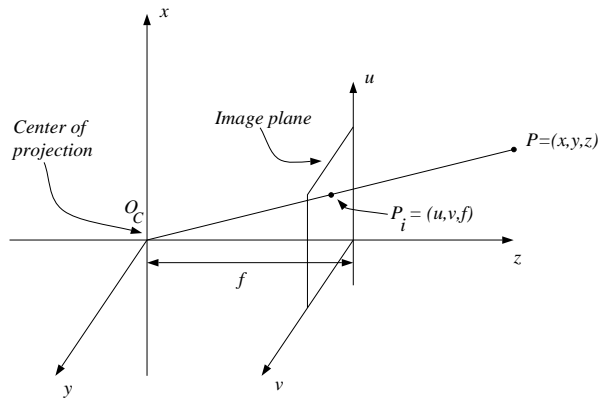


Figure 8: The geometry of the simple pinhole camera model for perspective transformation.

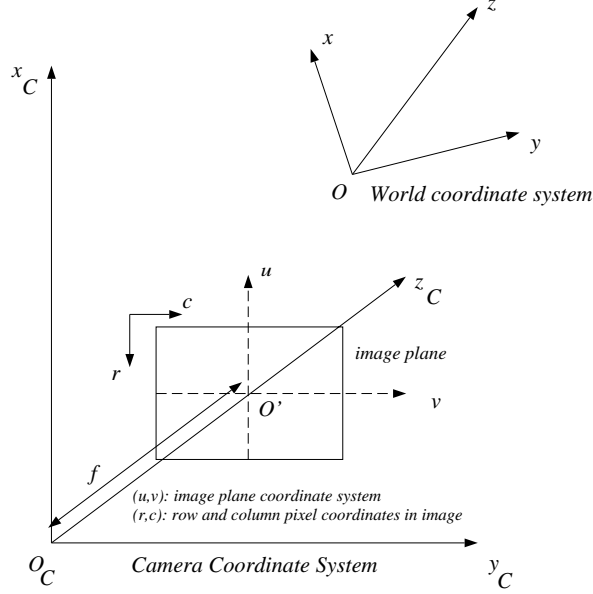


Figure 9: The various coordinate systems with respect to each other.

for in the mathematical model. The relationships between the screen coordinates, the pinhole model, and the world coordinates are shown in Figure 9, which is the model used for the camera calibration procedure described below.

The position and orientation (pose) of the camera with respect to the world coordinate system (O, x, y, z) is defined by a rigid transformation

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = \mathbf{R} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \mathbf{T}, \quad (10)$$

where \mathbf{R} is a 3×3 rotation matrix $[r_{ij}]$, and \mathbf{T} is a translation vector, $[t_1, t_2, t_3]^T$.

The pixel coordinates are related to the image plane coordinates by the following equations (r stands for row and c for column):

$$\begin{aligned} r - r_0 &= s_u u, \\ c - c_0 &= s_v v, \end{aligned} \quad (11)$$

where (r_0, c_0) are the pixel coordinates of the image plane coordinate system O' . s_u and s_v are needed in order to (i) account for the proper sign (note that the r -axis and u -axis are pointing in opposite directions), and (ii) the non-isotropic nature of some cameras (i.e., cameras with non-square pixels) and

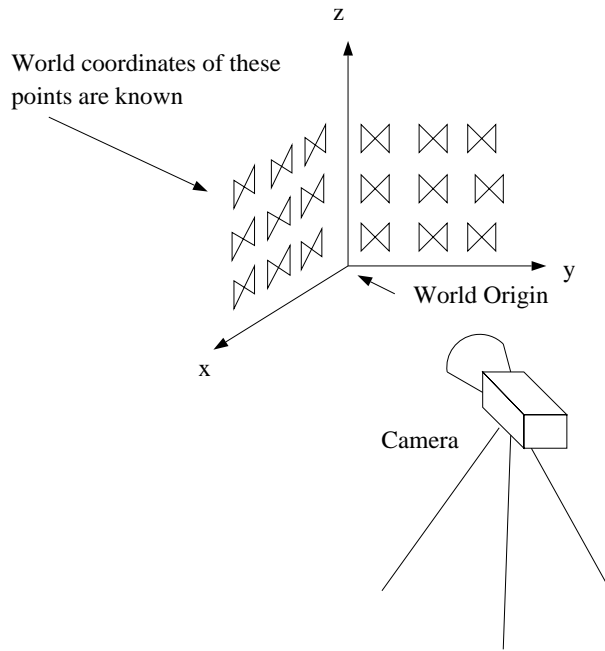


Figure 10: The set-up with which the camera is calibrated.

thus also capture the aspect ratio of the camera. Let $f_u = s_u f$ and $f_v = s_v f$. Then the four quantities f_u , f_v , r_0 and c_0 are called the *intrinsic camera parameters*. The transformations \mathbf{R} and \mathbf{T} are known as the *extrinsic camera parameters*.

Substituting all the expressions in Equations 10 and 11 into Equation 9, we get

$$\begin{aligned} \frac{u}{f} &= \frac{r - r_0}{s_u f} = \frac{r - r_0}{f_u} = \frac{r_{11}x + r_{12}y + r_{13}z + t_1}{r_{31}x + r_{32}y + r_{33}z + t_3}, \\ \frac{v}{f} &= \frac{c - c_0}{s_v f} = \frac{c - c_0}{f_v} = \frac{r_{21}x + r_{22}y + r_{23}z + t_2}{r_{31}x + r_{32}y + r_{33}z + t_3}. \end{aligned} \quad (12)$$

4.2.2 Calibration Procedure

In the GRASP system, a single initial calibration is performed during the start-up phase of an application. It is assumed that the intrinsic parameters of the camera do not subsequently change; changes in the extrinsic parameters are tracked using the tracker mark attached to the camera. It is important that the intrinsic camera parameters are calculated accurately. The virtual, digital camera used during the generation of the computer graphics must match the real camera if the resulting images are to appear realistic when merged with the video signal.

Figure 10 depicts the set-up used for the camera calibration process. The actual procedure consists of the following steps.

- The camera is pointed at the calibration grid.
- A copy of the camera image is read into the computer via the frame grabber.
- The centers of the butterfly patterns are located within the grabbed image thereby obtaining 2D image coordinates corresponding to the *known* 3D locations of the actual butterflies. As with the image calibration procedure, the process of locating the points of interest in the grabbed image may be performed manually by the user. An automatic procedure may also be used, which locates the corners of dark squares on a light background. This second procedure requires no user input.
- The camera parameters are computed using the tuples of $(x_i, y_i, z_i, r_i, c_i)$, where (x_i, y_i, z_i) are the world coordinates of the center of the i th butterfly and (r_i, c_i) are the corresponding 2D image coordinates.

The values calculated during the image calibration stage are used during the camera calibration

process and therefore become integrated into the values of the intrinsic camera parameters. This ensures that the scan-converted image is correctly aligned with the video image.

4.2.3 Numerical Estimation of Camera Parameters

If we substitute the known world coordinates (x_i, y_i, z_i) and the image coordinates (r_i, c_i) for each calibration point P_i into Equation 12, then for each point we get a pair of equations:

$$\begin{aligned}
(r_i - r_0)x_i r_{31} &+ (r_i - r_0)y_i r_{32} + (r_i - r_0)z_i r_{33} + (r_i - r_0)t_3 \\
&- f_u x_i r_{11} - f_u y_i r_{12} - f_u z_i r_{13} - f_u t_1 = 0, \\
(c_i - c_0)x_i r_{31} &+ (c_i - c_0)y_i r_{32} + (c_i - c_0)z_i r_{33} + (c_i - c_0)t_3 \\
&- f_v x_i r_{21} - f_v y_i r_{22} - f_v z_i r_{23} - f_v t_2 = 0.
\end{aligned} \tag{13}$$

Rearranging and redefining some terms (so that we are left with a linear system to solve) for n calibration points we get a homogeneous linear system to solve,

$$\mathbf{A}\mathbf{W} = 0. \tag{14}$$

Here \mathbf{A} is a $2n \times 12$ matrix of the form:

$$\mathbf{A} = \begin{bmatrix} -x_1 & -y_1 & -z_1 & 0 & 0 & 0 & r_1 x_1 & r_1 y_1 & r_1 z_1 & -1 & 0 & r_1 \\ 0 & 0 & 0 & -x_1 & -y_1 & -z_1 & c_1 x_1 & c_1 y_1 & c_1 z_1 & 0 & -1 & c_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -x_n & -y_n & -z_n & 0 & 0 & 0 & r_n x_n & r_n y_n & r_n z_n & -1 & 0 & r_n \\ 0 & 0 & 0 & -x_n & -y_n & -z_n & c_n x_n & c_n y_n & c_n z_n & 0 & -1 & c_n \end{bmatrix}. \tag{15}$$

The \mathbf{W} is a standard change of variables found in the computer vision literature which linearizes the above equations [16, 19, 36]:

$$\begin{aligned}
\mathbf{W}_1 &= f_u \mathbf{R}_1 + r_0 \mathbf{R}_3, \\
\mathbf{W}_2 &= f_v \mathbf{R}_2 + c_0 \mathbf{R}_3, \\
\mathbf{W}_3 &= \mathbf{R}_3, \\
w_4 &= f_u t_1 + r_0 t_3, \\
w_5 &= f_v t_2 + c_0 t_3, \\
w_6 &= t_3.
\end{aligned} \tag{16}$$

where $\mathbf{R}_1 = [r_{11}, r_{12}, r_{13}]^T$, $\mathbf{R}_2 = [r_{21}, r_{22}, r_{23}]^T$, and $\mathbf{R}_3 = [r_{31}, r_{32}, r_{33}]^T$ for notational shorthand. In other words, the \mathbf{W}_i 's are actually column matrices $\mathbf{W}_1 = [w_{11}, w_{12}, w_{13}]^T, \dots$, and

$$\mathbf{W} = [w_{11} \ w_{12} \ w_{13} \ w_{21} \ w_{22} \ w_{23} \ w_{31} \ w_{32} \ w_{33} \ w_4 \ w_5 \ w_6]^T.$$

The solution to Equation 14 is the first step to finding the camera parameters. To solve for \mathbf{W} , we temporarily set $t_3 = 1$ in order to get a non-homogeneous system. We ensure that the world coordinate system is defined so that $t_3 \neq 0$, by placing the world origin in front of the camera. Setting the $t_3 = 1$ is equivalent to dividing Equation 13 by t_3 and renaming the variables. That is, we now deal with the equations of the form:

$$\begin{aligned}
(r_i - r_0)x_i r_{31}/t_3 &+ (r_i - r_0)y_i r_{32}/t_3 \\
&+ (r_i - r_0)z_i r_{33}/t_3 + (r_i - r_0) \\
&- f_u x_i r_{11}/t_3 - f_u y_i r_{12}/t_3 \\
&- f_u z_i r_{13}/t_3 - f_u t_1/t_3 = 0, \\
(c_i - c_0)x_i r_{31}/t_3 &+ (c_i - c_0)y_i r_{32}/t_3 \\
&+ (c_i - c_0)z_i r_{33}/t_3 + (c_i - c_0) \\
&- f_v x_i r_{21}/t_3 - f_v y_i r_{22}/t_3 \\
&- f_v z_i r_{23}/t_3 - f_v t_2/t_3 = 0.
\end{aligned} \tag{17}$$

The result is a new equation:

$$\mathbf{A}'\mathbf{W}' + \mathbf{B}' = \mathbf{0}, \quad (18)$$

where \mathbf{A}' is the first 11 columns of \mathbf{A} , \mathbf{B}' is the last column of \mathbf{A} , and \mathbf{W}' is the corresponding reduced unknown vector. Since the system is, in general, over-determined, we have to use a least-squares method and find the \mathbf{W}' that satisfies

$$\min_{\mathbf{W}'} \|\mathbf{A}'\mathbf{W}' + \mathbf{B}'\|. \quad (19)$$

We employ one of the standard methods utilizing Singular Value Decomposition to solve this system of linear equations. In the solution \mathbf{W}' that we compute, everything is scaled because of setting $t_3 = 1$. In addition to this, there are constraints on \mathbf{W} .

- The norm $\|\mathbf{W}_3\| = 1$, because \mathbf{W}_3 is the last row of the rotation matrix \mathbf{R} .
- The sign of w_6 must be chosen to be compatible with the camera coordinate system since w_6 is the z component of the translation vector. That is, w_6 is positive if the camera is in front of the (x, y) plane.

Once \mathbf{W}' is computed, we get a new set of parameters \mathbf{W} which satisfy the above constraints,

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ \mathbf{W}_3 \\ w_4 \\ w_5 \\ w_6 \end{bmatrix} = \pm \frac{1}{\|\mathbf{W}'_3\|} \begin{bmatrix} \mathbf{W}'_1 \\ \mathbf{W}'_2 \\ \mathbf{W}'_3 \\ w'_4 \\ w'_5 \\ 1 \end{bmatrix}. \quad (20)$$

Now, based on Equations 17, the camera parameters are obtained by the following expressions (where $\bar{r}_0, \bar{c}_0, \dots$ are the estimated values of their corresponding parameters r_0, c_0, \dots):

$$\begin{aligned} \bar{r}_0 &= \mathbf{W}_1^T \mathbf{W}_3, \\ \bar{c}_0 &= \mathbf{W}_2^T \mathbf{W}_3, \\ \bar{f}_u &= -\|\mathbf{W}_1 - \bar{r}_0 \mathbf{W}_3\|, \\ \bar{f}_v &= \|\mathbf{W}_2 - \bar{c}_0 \mathbf{W}_3\|, \\ \bar{t}_1 &= (w_4 - \bar{r}_0 w_6) / \bar{f}_u, \\ \bar{t}_2 &= (w_5 - \bar{c}_0 w_6) / \bar{f}_v, \\ \bar{t}_3 &= w_6, \\ \bar{\mathbf{R}}_1 &= (\mathbf{W}_1 - \bar{r}_0 \mathbf{W}_3) / \bar{f}_u, \\ \bar{\mathbf{R}}_2 &= (\mathbf{W}_2 - \bar{c}_0 \mathbf{W}_3) / \bar{f}_v, \\ \bar{\mathbf{R}}_3 &= \mathbf{W}_3. \end{aligned} \quad (21)$$

The expressions in Equation 21 rely on the fact that the rotation matrix \mathbf{R} is orthonormal. That is, $\mathbf{R}^T \mathbf{R} = \mathbf{I}$. This may not be true for the solution $\bar{\mathbf{R}}$ estimated above. Therefore, we must take care that the estimated rotation matrix is orthonormal. There are a number of ways of doing this. One is to perform the above estimation using a constrained optimization in which the orthonormality of \mathbf{R} is enforced at the same time as the solution to Equation 19 is computed. That is, minimize the following:

$$\min_{\mathbf{W}'} \|\mathbf{A}'\mathbf{W}' + \mathbf{B}'\|^2 + \alpha \|\mathbf{R}^T \mathbf{R} - \mathbf{I}\|^2. \quad (22)$$

A second way is to first find a solution to Equation 19 using a simple least squares estimation, then find an improved estimate of the rotation matrix. Let the initial estimate of the rotation matrix be $\bar{\mathbf{R}}$ and the improved rotation matrix be $\tilde{\mathbf{R}}$. One such technique is given in [36, 37] in which a closed form solution is provided for computing $\tilde{\mathbf{R}}$. We have used this second approach to enforce orthonormality of the rotation matrix. The $\tilde{\mathbf{R}}$ then is used to estimate the rest of the parameters as shown in Equation 23:

$$\begin{aligned} \bar{r}_0 &= \mathbf{W}_1^T \tilde{\mathbf{R}}_3, \\ \bar{c}_0 &= \mathbf{W}_2^T \tilde{\mathbf{R}}_3, \\ \bar{f}_u &= -\|\mathbf{W}_1 - \bar{r}_0 \tilde{\mathbf{R}}_3\|, \end{aligned}$$

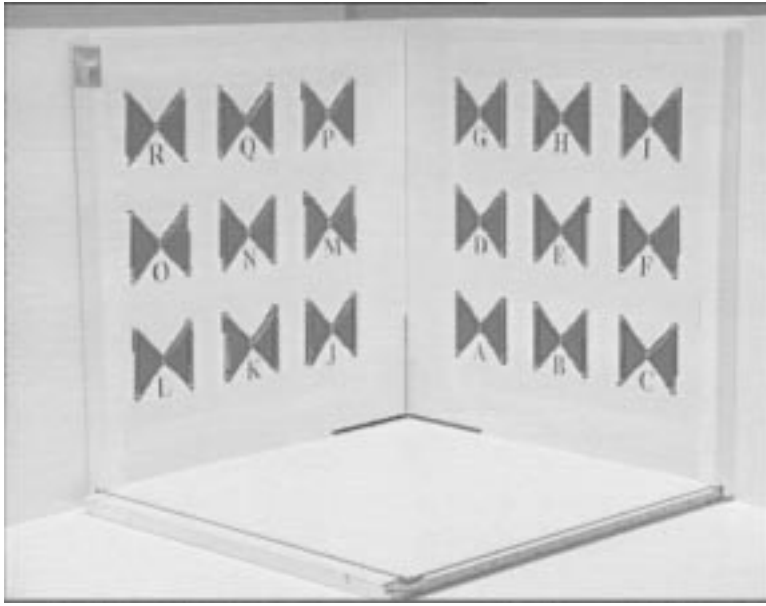


Figure 11: Result of camera calibration.

$$\begin{aligned}
 \bar{f}_v &= \|\mathbf{W}_2 - \bar{c}_0 \tilde{\mathbf{R}}_3\|, \\
 \bar{t}_1 &= (w_4 - \bar{r}_0 w_6) / \bar{f}_u, \\
 \bar{t}_2 &= (w_5 - \bar{c}_0 w_6) / \bar{f}_v, \\
 \bar{t}_3 &= w_6.
 \end{aligned} \tag{23}$$

Figure 11 shows the result of the camera calibration process. In this figure, the 3D model of the calibration pattern is rendered using the camera parameters estimated from the calibration process and superimposed on the video image of the physical calibration pattern. As can be seen, both the world coordinate axes and the butterfly patterns are properly aligned with the image of the physical calibration grid.

One of the factors that affects the accuracy of the camera calibration process, and consequently many of the following steps, is the nonlinear lens distortions of the physical camera. Nonlinear distortions of the camera can cause problems because they are inconsistent with the pinhole model assumed when modeling and calibrating the camera. Therefore, the linear parameters estimated from such a camera are not valid. We have, in fact, seen this effect very clearly in some of our experiments. One solution to this problem is to model the nonlinear camera distortions and estimate those parameters, too. Such methods have been developed in computer vision and they are usable [36]. This is attractive if one is only interested in tasks in which the camera model plays an important role only in *analyzing* the contents of the image. On the other hand, if one wants to use the estimated camera parameters to render in common computer graphics environments, this proposition loses its feasibility. In particular, most real-time 3D graphics rendering systems assume the pinhole model, thus making the inclusion of non-standard camera models into the system very difficult.

In order to address the issue of accuracy, we have conducted tests to estimate the error between the calibration points in the image and the backprojection of the model points using estimated camera parameters (i.e., we estimated the 2D error). We ran the calibration procedure described above on 19 images of the calibration grid from different viewpoints. The image points that are input to the calibration algorithm are detected with sub-pixel accuracy. Table 2 contains the calibration errors in terms of number of pixels. Each entry is the average error over all the calibration points in the specific image. The average error over all the tests is only 1.32 pixels. The standard deviation of 0.41 indicates that the variation in the error is small and view-independent. This confirms the validity of our camera calibration procedure, and also supports the use of the pinhole camera model in the GRASP() system. The lenses of our cameras introduce insignificant distortions that may be ignored in most cases. This may not be true for other augmented reality systems that use head-mounted displays or fiber optic lenses.

Image number	Error (pixels)
1	1.38654
2	1.63748
3	1.52379
4	0.953627
5	0.845233
6	1.72065
7	2.08185
8	1.54767
9	0.819965
10	1.21385
11	0.933343
12	1.60646
13	0.700969
14	0.741939
15	1.13696
16	1.44271
17	1.36449
18	1.31004
19	2.13014
Average Error	1.32
Std. Deviation	0.411

Table 2: The image error (in pixels) using the results of the calibration results.

4.3 Pointer Calibration

Pointer calibration determines the geometry of the pointer object used by the GRASP system applications. In particular, this calibration step calculates the position of the tip of the pointer relative to the tracker mark, i.e., the transformation between the coordinate system of the pointer object and the coordinate system of the tracker mark. This step is necessary before several other calibration steps can be completed, because they require the user to pick points on real objects with the pointer. The transformation can be estimated once and stored, and is valid as long as the rigid attachment of the receiver does not change.

The pointer used in our applications is a wooden wand with a tracker receiver attached to the base and a tapered tip which is used to locate 3D points on real objects during user interaction (Figure 5). The geometry of the pointer object is not pre-defined but calculated during the calibration procedure.

The mechanism to calibrate the pointer requires the user to pick the same point in 3D space several times, using a different orientation for the pointer each time (see Figure 12). For each pick, the position and the orientation of the tracker mark within the tracker coordinate system are recorded. The result of this procedure is a set of points and directions with the common property that the points are all the same distance from the single, picked point in 3D space and all of the directions associated with the points are oriented toward the picked point.

4.3.1 Pointer Model

The detailed geometry of the relationship between the tracker coordinate system and the world coordinate system is shown in Figure 13. The mark is attached to the pointer device, and provides readings to the magnetic tracking system. The geometry of the pointer, since it will be used to calibrate the tracker, can be defined as the following. The tip of the pointer is related to the tracker measurements by a rigid transformation given by the formula:

$$p_w = p_m + \mathbf{R}_m p_t, \tag{24}$$

where p_w is the position vector representing the tip of the pointer in world coordinates, p_t is the receiver attachment offset, \mathbf{R}_m is the rotation matrix defining the orientation of the receiver as measured by the tracker, and p_m is the measured position of the receiver. The tracker and world coordinate systems are related to each other by a tracker-to-world transformation.

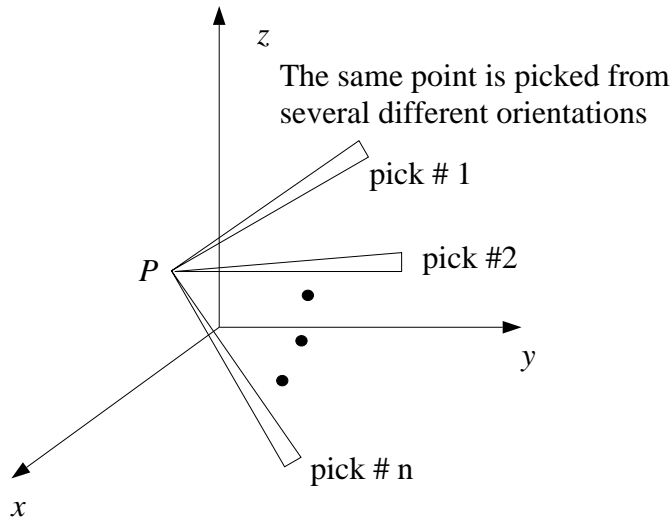


Figure 12: Schematic of the procedure for pointer calibration.

4.3.2 Calibration Procedure

Pointer calibration produces the quantities p_w and p_t . If we pick n points (for us n is a number between 3 and 6) whose positions are the same, but whose orientations are different, then we have the following equation to solve:

$$\begin{pmatrix} \mathbf{I} & -\mathbf{R}_{m1} \\ \mathbf{I} & -\mathbf{R}_{m2} \\ \vdots & \vdots \\ \mathbf{I} & -\mathbf{R}_{mn} \end{pmatrix} \begin{pmatrix} p_w \\ p_t \end{pmatrix} = \begin{pmatrix} p_{m1} \\ p_{m2} \\ \vdots \\ p_{mn} \end{pmatrix}, \quad (25)$$

where \mathbf{I} is a 3×3 identity matrix. Initially, p_t and p_w are unknown and must be estimated as a result of the calibration procedure. p_{mi} and \mathbf{R}_{mi} , on the other hand, are the position and orientation readings coming from the tracker measurements. Equation 25 is constructed using the measurements made from reading a point at n different orientations. Altogether, there are six unknowns (three for p_t and three for p_w) and there are $3n$ rows. Therefore, the system is over-determined and is solved using a least squares method.

4.4 Tracker Transmitter Calibration

The calibration procedures described above provide us with most of the information needed to calibrate our system as a whole, but we still need to locate the tracker coordinate 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 \mathbf{C} in Figure 4). This transformation is calculated indirectly after calculating the transformations \mathbf{D} and \mathbf{G}_3 and measuring transformation \mathbf{F}_3 .

This information is calculated as an extension to the pointer calibration procedure. In addition to the numerous 3D picks of the same point from different orientations, we also pick three points on the same plane, but only once each (see Figure 14). By using the same calibration grid as before, for which the locations of the points in the world coordinate system are known, we can determine the location of the points used for the tracker calibration (J, P, L).

Once p_t is estimated as described above, the tracker-to-world transformation is computed by using the p_w values for the three points, J, P , and L . The line segment JP is defined to be parallel to the world z -axis, and the line segment JL is defined to be parallel to the world x -axis. The p_w values for the three points are computed using the mark readings p_{mi} (for $i \in \{J, P, L\}$) and the computed p_t according to Equation 26,

$$\begin{aligned} p_{wJ} &= p_{mJ} + \mathbf{R}_{mJ} p_{tJ}, \\ p_{wL} &= p_{mL} + \mathbf{R}_{mL} p_{tL}, \end{aligned} \quad (26)$$

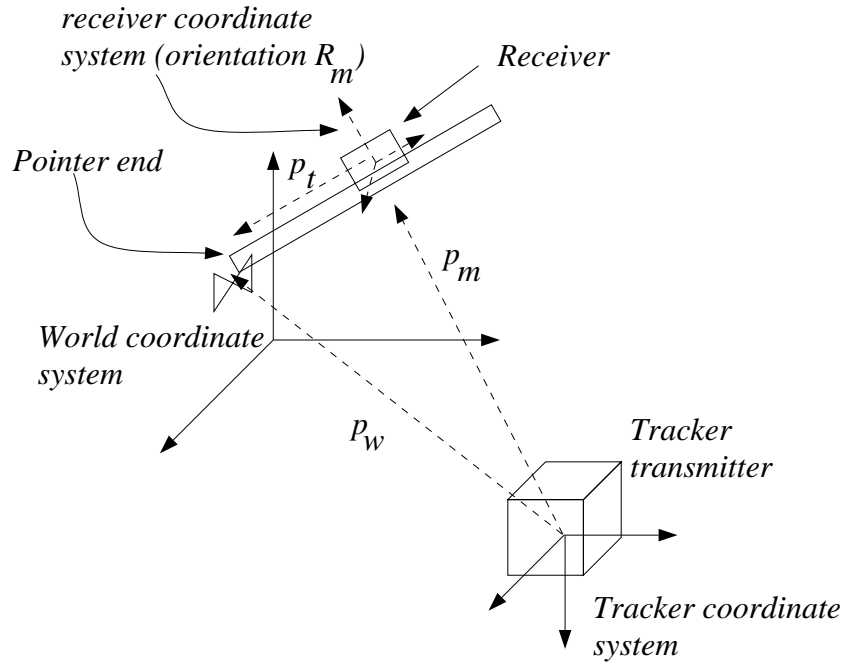


Figure 13: The relation between the world coordinate system and the tracker coordinate system.

$$p_w P = p_m P + \mathbf{R}_m P p_t P.$$

The x , y , and z -axes are then computed from these p_{wi} 's. The x -direction vector is

$$\mathbf{x} = p_w L - p_w J,$$

and the z -direction vector is computed using

$$\mathbf{z} = p_w P - p_w J.$$

The y -direction vector is obtained by $\mathbf{y} = \mathbf{z} \times \mathbf{x}$. These vectors are normalized so that $\|\mathbf{x}\| = 1$, $\|\mathbf{y}\| = 1$, and $\|\mathbf{z}\| = 1$. The rotation matrix then is given by

$$\mathbf{R} = [\mathbf{x} \quad \mathbf{y} \quad \mathbf{z}]^T,$$

and the translation vector p_0 , the origin of the world coordinate system with respect to the tracker coordinate system, is computed from

$$p_w J = p_m J + \mathbf{R}_m J p_0.$$

Figure 15 shows the result of pointer calibration and tracker calibration. This figure shows a cone which is aligned with the physical wand using the estimated parameters. The cone is then rendered and superimposed on the image of the physical pointer.

4.5 Object Calibration

The object calibration process calculates the location and orientation of a real-world object, so a virtual counterpart can be aligned with it by being placed at the corresponding location, with the appropriate orientation, within the virtual world [38]. This process of alignment of a 3D model with the physical object is also called object registration. There are two particular cases where this is necessary:

- to support *direct* interaction between the user and real-world objects by using a computer model aligned with the real object, and
- to support interaction between real and virtual objects using computer models of the real objects that can interact with purely virtual objects, i.e., objects that do not have any real-world counter-

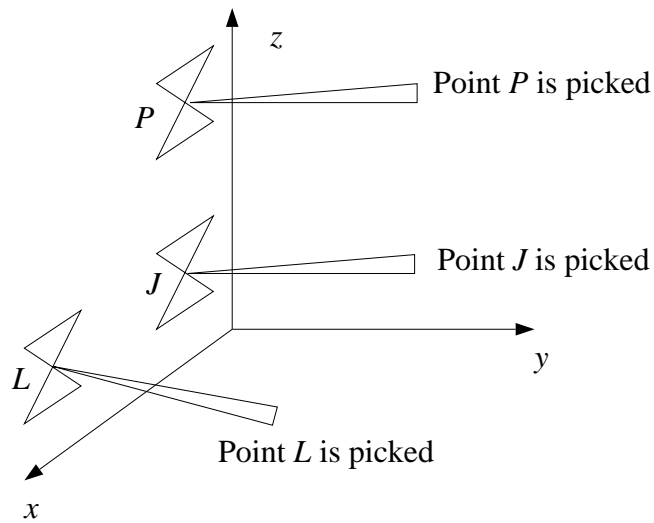


Figure 14: Schematic of the procedure for the tracker transmitter calibration.



Figure 15: The model of a pointer is aligned with the physical pointing device by using the results of pointer and tracker calibration.

parts.

The calibration procedure requires the model to include a number of *landmark points*, points whose positions are known in the coordinate system of the object model. Geometric models of objects might be created piece-by-piece from a set of geometric primitives or they might come from a CAD system. These models are generally stored as files on disk for subsequent use. In addition to the geometric and attribute data, these files must also contain the 3D positions and labels of the landmark points. These points should correspond to natural features on the object, such as corners or creases, that can be easily identified by a user. The registration procedure consists of locating the corresponding points on the real object and the model, and then calculating the object-to-world transformation from these point pairs.

There are two ways for the user to identify these landmarks. The first is an image-based approach in which the image location of the landmark points are identified and the object pose is estimated using the framework of the camera model described in section 4.2.1. The second is by the user picking the 3D coordinates of the landmark points directly on the physical object using a 3D pointing device. The object pose is then computed as a rigid transformation that maps the 3D coordinates of the landmark points in object coordinates into the 3D coordinates of the same points in world coordinates. The next two sections present the details of the two approaches to object calibration.

4.5.1 Image-based Object Calibration

The goal of an image-based object calibration/registration is to start with a calibrated camera and compute the object-to-camera transformation of a single object for which there is a known geometric model and landmark points. The position of an object is determined “through the lens” of the camera. The camera is usually the same camera used to capture the video signal that is combined with the graphics. The calibration begins by capturing an image of the real-world object and locating the set of landmark points in the image. There is a great deal of work in the area of automatic pose determination in the computer vision literature [23, 29], but in general these techniques apply to only limited classes of models and scenes. In our work, the locations of landmark points in the image are identified interactively by the user. We assume that the points are mapped from known locations in 3-space to the image via a rigid 3D transformation and a projection.

Because the camera is already calibrated, its coordinate system is known. Finding the position of the object in camera coordinates is identical to the camera calibration problem (which has been described above), except that we assume the internal parameters, f_u , f_v , r_0 , and c_0 , are known. The result is a linear system which follows from Equations 10 and 11,

$$\begin{aligned} & (r_i - r_0)x_i r_{31} + (r_i - r_0)y_i r_{32} \\ & + (r_i - r_0)z_i r_{33} + (r_i - r_0)t_3 \\ & - f_u x_i r_{11} - f_u y_i r_{12} \\ & - f_u z_i r_{13} - f_u t_1 = 0, \text{ and} \end{aligned} \quad (27)$$

$$\begin{aligned} & (c_i - c_0)x_i r_{31} + (c_i - c_0)y_i r_{32} \\ & + (c_i - c_0)z_i r_{33} + (c_i - c_0)t_3 \\ & - f_v x_i r_{21} - f_v y_i r_{22} \\ & - f_v z_i r_{23} - f_v t_2 = 0, \end{aligned} \quad (28)$$

where $(r_i, c_i, x_i, y_i, z_i)$ are the pixel and 3D coordinates of the landmark points. This can be written in matrix form as:

$$\mathbf{A} \mathbf{x} = 0, \quad (29)$$

where \mathbf{x} is the vector of unknowns, r_{ij} , and t_i for $i, j = 1, 2, 3$ and \mathbf{A} is the coefficient matrix.

This is a homogeneous linear system. In order to solve it, we convert it to a non-homogeneous system similar to the camera calibration problem discussed in Section 4.2.3. The equations are divided by t_3 and the resulting non-homogeneous linear system is solved for the 11 unknowns, r_{11} , r_{12} , r_{13} , r_{21} , r_{22} , r_{23} , r_{31} , r_{32} , r_{33} , t_1 , and t_2 , to within a scale factor t_3 . We can assume $t_3 > 0$ (the object is in front of the camera), and use the fact that \mathbf{R} is a rotation matrix (i.e., $\mathbf{R}^T \mathbf{R} = \mathbf{I}$ and $r_{31}^2 + r_{32}^2 + r_{33}^2 = 1$) to find t_3 after we have solved for the other eleven parameters. The linear system can be over-constrained to account for error by choosing more than six landmarks, in which case a least-squares solution is sought.

We have found in practice that this approach does not work well enough because even with as many as 15 points, the error introduced by the mouse clicks in the image, the physical measurements of the model, and the camera calibration give solutions that are not rigid transformations. Thus, Equations 27 and 28 must be solved while accounting for the fact that \mathbf{R} is a rotation matrix. One approach is to express \mathbf{R} in terms of three degrees of freedom which span the space of rotation matrices (Euler angles for instance). However, this produces a nonlinear system which contains some singularities that make finding a solution difficult. Instead we treat Equations 27 and 28 as a homogeneous linear system with twelve unknowns (obtained by multiplying by t_3) and simultaneously enforce rigidity by solving the nonlinear optimization problem

$$\|\mathbf{A} \mathbf{x}\|^2 + \alpha \|\mathbf{R}^T \mathbf{R} - \mathbf{I}\|^2, \quad (30)$$

where \mathbf{x} is the set of twelve unknowns in the rigid transformation, and α is a term which controls the amount of rigidity. The $2n \times 12$ matrix \mathbf{A} comes directly from equations 27 and 28, and n is the number of landmark points.

Since this is a nonlinear optimization problem, it is important to select the starting point properly. This is done by using the solution of the Equation 29 as the initial guess. This initial guess is often sufficiently close to the correct solution that the optimization method converges to the final solution within a small number of iterations. In practice the precise value of α is not important ($\alpha = 1$ is chosen

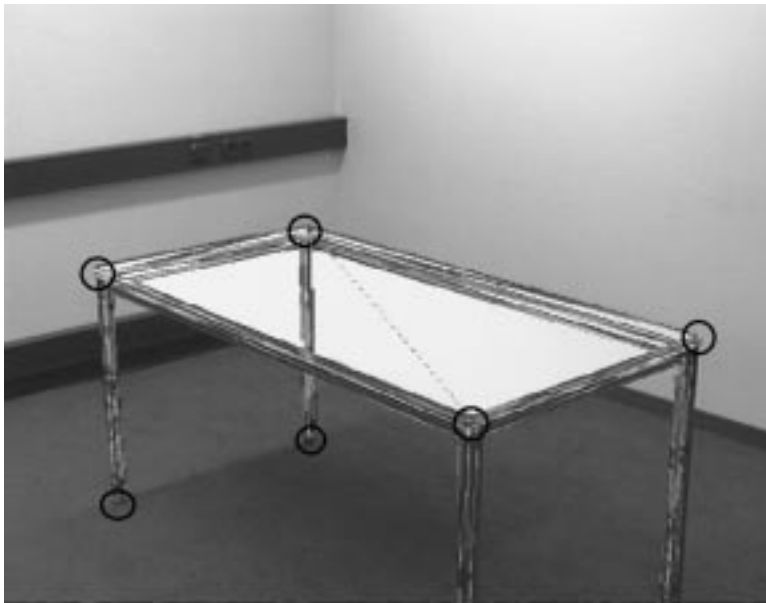


Figure 16: The wire-frame of image of a 3D model of a table is aligned with its physical counterpart. The centers of the circles in the image indicate the locations of the landmarks on the 3D object, the corners of the table in this case. This alignment utilizes the result of camera calibration, as well as object calibration.

for our work), and this optimization problem can be solved in a reasonable amount of time by gradient descent. We have found this method produces rigid transformations (to within 0.1%) and point matches that are within 2 pixels when points are re-projected onto the image plane. However, there can be significant error in the t_3 term which we attribute primarily to error in the camera calibration.

Figure 16 shows the result of a calibration procedure described above. In this example, the 3D wire-frame model of a table is rendered using position and orientation information obtained by the calibration process. The rendered image is superimposed on the image of the physical table. The circles indicate the landmarks utilized in this particular object calibration.

4.5.2 Pointer-Based Object Calibration

While the image-based object calibration described above is an area of on-going research, it currently has some limitations which make it inadequate for certain applications. For instance, in the case of the engine model shown in Figure 17, the image-based approach can produce a rigid transformation which matches landmark points in the image to within about 2 pixels. Yet the error in the z -direction (distance from the camera) can be unacceptably large. This error becomes evident as the object is turned as in Figure 18.

Another source of error not addressed in this paper is the error in the underlying CAD model used for the augmenting graphics. In the engine case, the associated geometric model was interactively created using a commercial CAD program. Errors in the model are evident around the fan and air pump at the front of the engine. The automatic generation of accurate geometric models for augmented reality is an on-going research topic at ECRC.

We have developed an object calibration procedure using the 3D pointing device to overcome the errors produced in the image-based procedure. It uses the same pointing device which is shown in Figure 5 and which is already calibrated according to the procedure described in Section 4.3. This device has an accuracy of about $\pm 1\text{cm}$. The pointer-based calibration works similar to the image-based calibration in that a set of landmark points are picked whose positions in the object coordinate system are known. The difference from the image-based calibration is that the picking is done in 3D using the pointer. Thus, we have a set of points whose 3D coordinates are known in the object's local coordinate system (p^l) as well as in the world coordinate system (p^w). The two sets of coordinates are related by a rigid transformation, specified by a rotation matrix \mathbf{R} and a translation vector \mathbf{T} .

The goal of the calibration in this case is to estimate the rigid transformation, \mathbf{R} and \mathbf{T} . The calibration proceeds by having the user pick n landmark points using the 3D pointer. The object coordinates



Figure 17: A shaded engine model registered to a real model engine using an image-based calibration. The geometric model is shaded in red, blue and yellow, before being overlaid on the plastic model.

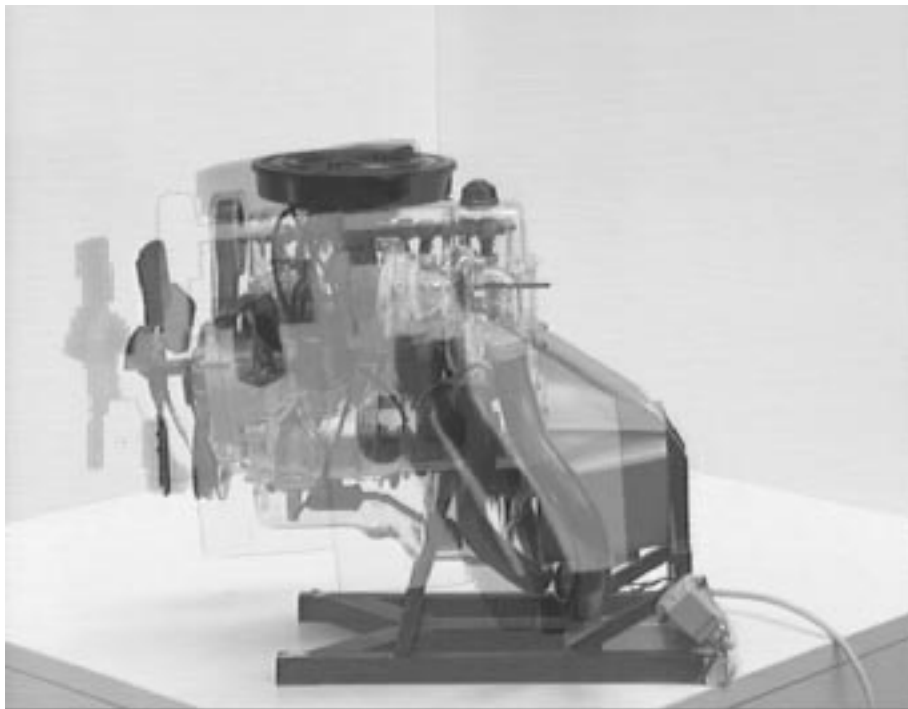


Figure 18: When the model is turned and its movements tracked, the graphics show the misalignment in the camera's z -direction.



Figure 19: A shaded engine model registered to a real model engine using a pointer-based calibration.

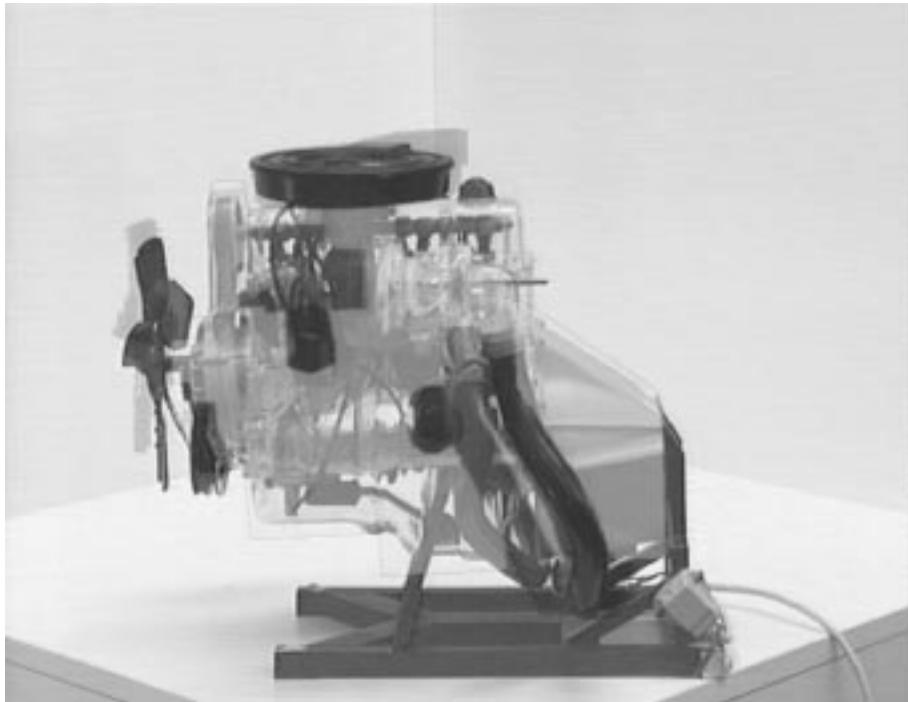


Figure 20: When the model is turned and its movements tracked, the graphics show the correct alignment in the camera's z-direction.

and the world coordinates of the landmark points are related by:

$$p_i^w = \mathbf{R}p_i^l + \mathbf{T} \quad \text{for } i = 1, \dots, n \quad (31)$$

This is a linear system with 12 unknowns. For a unique solution, a minimum of 4 points are needed, but in most cases we use more than 4 points (approximately 10) and solve for the least-squares error.

As with the image-based object calibration, error in the measurements can produce solutions that represent non-rigid transformations. Such non-rigidities can produce undesirable artifacts when this transformation is combined with others in the graphics system. As in the case of image-based calibration, we force \mathbf{R} to be a rotation matrix by solving a nonlinear optimization problem:

$$\|p_i^w - \mathbf{R}p_i^l - \mathbf{T}\|^2 + \alpha\|\mathbf{R}^T\mathbf{R} - \mathbf{I}\|^2. \quad (32)$$

This procedure provides the object-to-world transformation that is needed for object registration. Figure 19 shows a model engine which has been calibrated in such a manner. The calibration here appears to be similar to the one presented in Figure 17, including the CAD model errors near the fan. Rotations of the engine (Figure 20) show that the pointer-based calibration does not suffer from the *depth problem* of the image-based approach.

4.5.3 Object Calibration Error Analysis

We applied both object calibration methods to the camera calibration grid. We know the grid’s true pose parameters, because it defines the world coordinate system. That is, the object-to-world transformation for the camera calibration grid is identity. We performed ten separate pose computations on the camera calibration grid. Tables 3 and 4 present the results of the computed pose parameters using the image-based and the pointer-based methods. The average translational error for the image-based method is approximately 0.65 cm, and the rotational error is slightly over 1 degree. The average translational error for the pointer-based method is approximately 0.5 cm, and the rotational error is less than 1 degree. The small standard deviations associated with these averages shows that these errors change only slightly with varying views and object orientations. As observed in practice the pointer-based method provides better results. The image-based object calibration procedure produces greater errors spread over a wider range of values. For our current applications, the errors from the pointer-based method are acceptable, and are within the error range of the magnetic tracking system used to locate our 3D pointer.

4.6 Mark Calibration

The marks can be attached to any physical entity (e.g., the camera, pointer, or various objects) which needs to be tracked. The marks are attached to these physical objects in a rigid manner, but this rigid relationship must be determined in order to utilize the tracker readings. Mark calibration calculates the transformation between the coordinate system of the tracker mark and the coordinate system of the object itself (transformations \mathbf{G}_i) using the measured transformations \mathbf{F}_i , along with the previously calculated world-to-object transformations (\mathbf{E} , for example). This transformation is fixed for as long as the object and mark are rigidly attached to one another. Non-tracked real objects are expected to remain static so that their virtual counterparts can function as part of the scenery in the virtual world. This is used, for example, when real-world objects appear to occlude virtual objects in the augmented video image [9].

Various pre-computed transformations may be used to estimate the mark position with respect to the object to which it is attached. Referring to Figure 4, the quantities to be computed by the mark calibration step are \mathbf{G}_1 (mark position with respect to the camera coordinate system) and \mathbf{G}_2 (mark position with respect to the object coordinate system).

We use the previously estimated transformations \mathbf{A} (from camera calibration), \mathbf{C} (from tracker transmitter calibration), and \mathbf{F}_1 (measurement read by the tracker) to compute \mathbf{G}_1 . \mathbf{G}_1 is computed by

$$\mathbf{G}_1 = \mathbf{A}\mathbf{C}^{-1}\mathbf{F}_1. \quad (33)$$

Similarly, \mathbf{G}_2 is estimated using transformations \mathbf{E} (from object calibration), \mathbf{C} , and \mathbf{F}_2 . The expression for computing \mathbf{G}_2 is:

$$\mathbf{G}_2 = \mathbf{E}\mathbf{C}^{-1}\mathbf{F}_2. \quad (34)$$

no.	Position				Orientation			
	x (cm)	y (cm)	z (cm)	d (cm)	θ_z (deg)	θ_y (deg)	θ_x (deg)	Θ (deg)
1	0.32278	1.4406	0.11128	1.4805	-1.2125	0.015508	0.36904	1.2657
2	-0.18398	0.067902	-0.32748	0.38171	-0.71164	-0.51442	0.58884	1.0503
3	-0.34908	-0.088844	0.13148	0.38346	-1.1302	0.37981	-0.48577	1.2863
4	-0.3448	0.26537	0.12379	0.45236	-1.6523	0.38806	-0.44248	1.7529
5	0.071564	0.15015	-0.20113	0.261	-0.13397	-0.52943	0.48979	0.72474
6	0.17125	0.32938	-0.21011	0.42657	-0.262	-0.58563	0.4597	0.79392
7	0.066979	0.39086	-0.13217	0.41801	-0.7308	-0.18575	0.32566	0.82633
8	0.42134	0.27922	-0.0063667	0.5055	0.40926	0.11829	0.0	0.42876
9	1.7111	0.46733	-0.16515	1.7814	2.1022	-1.4316	0.31404	2.5675
10	-0.1862	0.3404	0.0086823	0.38807	-1.189	0.12378	0.065611	1.2019
	Average			0.648	Average			1.19
	Standard Deviation			0.500	Standard Deviation			0.577

Table 3: The registration errors as a result of computing the pose of the calibration grid from ten different camera viewpoints using the image-based object calibration method. x , y , and z is the translation in centimeters necessary to bring the object origin to the world origin. d is the distance to the world origin. θ_x , θ_y , and θ_z are the Euler rotation angles needed to align the model with the physical object. Θ is the magnitude of the single rotation around an arbitrary axis needed to align the model.

no.	Position				Orientation			
	x (cm)	y (cm)	z (cm)	d (cm)	θ_z (deg)	θ_y (deg)	θ_x (deg)	Θ (deg)
1	-0.19623	0.18351	0.16807	0.31691	0.1327	0.42423	0.33043	0.56133
2	-0.40714	0.1785	0.57244	0.7248	-0.044879	1.1964	-0.54304	1.3166
3	-0.5295	0.057373	0.69751	0.8776	0.094873	1.4231	-0.94401	1.7074
4	-0.33319	-0.03144	0.44515	0.55693	0.15195	1.1219	0.019782	1.1344
5	-0.27894	0.15702	0.12128	0.34231	0.079129	0.70108	-0.18081	0.72474
6	0.31868	0.19687	-0.024905	0.37542	0.25525	-0.25621	-0.44315	0.58431
7	0.48705	0.30314	0.30291	0.64874	-0.20294	-0.04053	0.47847	0.51247
8	0.16739	0.22347	0.041672	0.28231	-0.297	-0.18015	0.58678	0.68755
9	0.58507	0.59087	-0.10316	0.8379	-0.31294	-0.3475	0.69631	0.84208
10	0.047195	0.187	0.21884	0.2917	-0.28522	0.079583	0.48009	0.56138
	Average			0.525	Average			0.863
	Standard Deviation			0.222	Standard Deviation			0.378

Table 4: The registration errors as a result of computing the pose of the calibration grid from ten different camera viewpoints using the pointer-based object calibration method. x , y , and z is the translation in centimeters necessary to bring the object origin to the world origin. d is the distance to the world origin. θ_x , θ_y , and θ_z are the Euler rotation angles needed to align the model with the physical object. Θ is the magnitude of the single rotation around an arbitrary axis needed to align the model.

5 Results

The calibration procedures outlined above have been implemented as an integral part of the GRASP system. Several AR applications have been implemented using the GRASP system that utilize the suite of calibration procedures described in this paper. These applications include a mechanic’s assistant [33] and an interior design tool [3].

Figures 1 and 21 present the results of a successful calibration process in the mechanic’s assistant application. In this application, a real engine is viewed through a video camera. The image of the engine is augmented by providing part identification information in the form of labels pointing to the proper parts of the real engine. This effect is achieved by aligning a 3D model of the engine with the real engine, rendering the model from the same viewpoint as the physical camera (but not displaying the rendered model), and using the locations of the parts to determine which parts are visible and where the tags are to be attached. The rendered model is not displayed so that only the real engine is visible, but the z-buffer information produced during rendering is used for obtaining depth and visibility information about the engine. The engine is also tracked so that when it is physically moved, the location of the model is updated and the part identification labels are changed accordingly.

This example illustrates the success of the following set of calibration procedures.

- The engine model is displayed from the same viewpoint as the physical camera. The proper viewpoint effect and the registration of the resulting image is accomplished by the camera calibration and image calibration processes.
- The registration of the 3D engine model to the real engine is accomplished as a result of the object calibration process.
- The object calibration is performed using the pointer-based calibration method. This, in turn, is made possible as a result of the pointer calibration procedure.
- The proper updating of the labels and graphics when the physical engine is moved is accomplished as a result of successful tracker transmitter and mark calibration procedures.

Figures 22 and 23 show the results of a successful calibration process [9]. The first example shows the interaction of real and virtual objects where a real table occludes two virtual chairs. This is accomplished by registering a 3D model of the table with the physical table and displaying the graphics from the vantage point of the video camera looking at the room. The occlusion effect is achieved by rendering the 3D model of the registered table, not showing the rendered image of the model but using the z-buffer information from the rendering to hide the proper parts of the virtual chairs. The second example presents a simulated “gravity” capability that allows us to automatically place virtual objects on top of real ones. Once the room and table models are calibrated in the augmented scene, they may be used to provide collision detection. The virtual objects are incrementally moved in the direction of “gravity” until a collision is detected. They are then rotated until three points on each object come in contact with the real environment. In these examples, the results of camera calibration, image calibration, and image-based object calibration are used to obtain the final effect. The registration of the 3D model of the table with the physical table is shown in Figure 16.

6 Conclusion

Augmented reality is a powerful technology that provides new paradigms for human-computer interaction and communication. An essential component of an augmented reality system is the calibration of its devices and objects. Without proper and accurate calibration, convincing augmented reality effects are not possible. In this paper we have focused on the whole set of calibration procedures necessary for producing a realistic and accurate monitor-based augmented reality system.

In particular, the set of calibrations we have addressed in this paper are the following.

- Image calibration estimates the parameters of the 2D distortion model due to scan conversion.
- Camera calibration estimates the intrinsic and extrinsic parameters of a physical camera. This method utilizes computer vision techniques to compute the camera parameters. A calibration grid is used with points whose positions in the world are known.

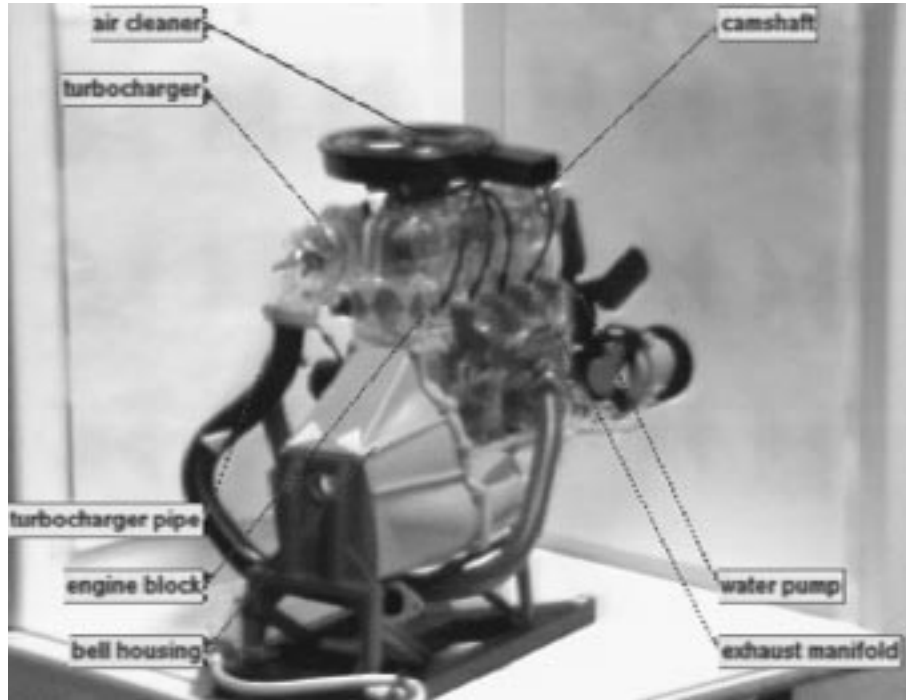


Figure 21: A real model engine annotated in augmented reality with a different set of visible components. (Also see Figure 1.)



Figure 22: A real table occluding two virtual chairs.



Figure 23: Three virtual objects automatically placed on a real table.

- Pointer calibration estimates the configuration of a pointing device attached to a magnetic tracker. This is accomplished by picking a single point with the pointer at many different orientations.
- Tracker transmitter calibration estimates the position and orientation of the tracker transmitter in the world coordinate system. This is accomplished by picking three known points from the camera calibration grid.
- Object calibration estimates the position and orientation (pose) of arbitrary objects in the world coordinate system. Two different approaches have been presented to perform this calibration. The first technique utilizes computer vision algorithms to determine the pose of the object. An image of the object is grabbed and known landmarks are manually identified. A transformation that aligns the object and the model from that particular view is then automatically calculated. In the second technique, a calibrated pointing device is used to locate the object landmarks in the world coordinate system. Estimating the rigid transformation between the two sets of 3D points produces the object pose.
- Mark calibration estimates the position and orientation of the magnetic tracker receivers with respect to the objects to which they are attached.

The overall goal of this work has been to attack the various calibration problems which often arise in computer graphics and in augmented reality. A major goal is to make these calibrations in a mathematically rigorous way in order to increase the accuracy and precision of the final result. We have tried to avoid the usual practice of using physical measuring tapes, or using manufacturer's specifications (e.g., for camera focal lengths) to estimate various parameters. Some of the individual calibration issues (e.g., camera calibration) have been studied and addressed in other research fields. One of the goals of this paper has been to bring many of these techniques into computer graphics. Another goal is to detail new calibration procedures which have not been previously published (e.g. image, tracker transmitter, pointer, and pointer-based object calibration.) The final goal is to bring together a variety of calibration techniques within a coherent framework for use in a new domain, augmented reality.

For some of the calibration steps, our work relies heavily on computer vision techniques. It is well known that many computer vision techniques, when used in a completely automated fashion, are not very robust. Therefore, a decision has been made in our approach to keep the user in the loop in order to increase robustness, but still utilize the mathematical framework of the computer vision techniques. This does not mean that we do not aim to automate the processes as much as possible. Our ultimate goal is to have robust methods of calibration, based on rigorous mathematical analysis, which only involve the user when necessary.

The current method of image-based object registration involves a great amount of user interaction. One of the goals we would like to achieve in the near future is to minimize the involvement of the user (but still keep him in the loop). Model-based vision techniques that increase the automation of the object registration process is one of the topics we are currently investigating. Minimizing user involvement in other calibration steps is also part of our future research plans. Our goals include developing more automated methods for doing the pointer and tracker transmitter calibration using image-based approaches.

7 Acknowledgments

The AutoCAD model of the automobile engine was initially produced by Anne Bachy. The plastic automobile engine model was generously provided by Revell, Inc. This work is financially supported by Bull SA, ICL PLC, and Siemens AG. We would also like thank Dieter Koller for providing extensions and error statistics for camera calibration.

References

- [1] K. Ahlers, D. Breen, C. Crampton, E. Rose, M. Tuceryan, R. Whitaker, and D. Greer. An augmented vision system for industrial applications. In *Telem manipulators and Telepresence Technologies*, volume 2351, pages 345–359. SPIE Proceedings, October 1994.
- [2] K. Ahlers, C. Crampton, D. Greer, E. Rose, and M. Tuceryan. Augmented Vision: A technical introduction to the Grasp 1.2 system. Technical Report ECRC-94-14, ECRC, Munich, Germany, 1994.

- [3] K. Ahlers, A. Kramer, D. Breen, P.-Y. Chevalier, C. Crampton, E. Rose, M. Tuceryan, R. Whitaker, and D. Greer. Distributed augmented reality for collaborative design applications. In *Proceedings of Eurographics '95 Conference*, Maastricht, NL, August 1995.
- [4] R. Azuma and G. Bishop. Improving static and dynamic registration in an optical see-through display. In *Computer Graphics (Proceedings of the SIGGRAPH Conference)*, pages 194–204, July 1994.
- [5] M. Bajura, H. Fuchs, and R. Ohbuchi. Merging virtual objects with the real world: Seeing ultrasound imagery within the patient. In *Computer Graphics (Proceedings of the SIGGRAPH Conference)*, pages 203–210, Chicago, IL, July 1992.
- [6] M. Bajura and U. Neumann. Dynamic registration correction in augmented-reality systems. In *Proceedings of the Virtual Reality Annual International Symposium (VRAIS '95)*, pages 189–196, RTP, North Carolina, March 1995.
- [7] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, New Jersey 07632, 1982.
- [8] M. Baudel and M. Beaudouin-Lafon. Charade: Remote control of objects using freehand gestures. *Communications of the ACM*, 36(7):28–35, July 1993.
- [9] D. Breen, E. Rose, and R. Whitaker. Interactive occlusion and collision of real and virtual objects in augmented reality. Technical Report ECRC-95-02, ECRC, Munich, Germany, 1995.
- [10] T. Caudell and D. Mizell. Augmented reality: An application of heads-up display technology to manual manufacturing processes. In *Proceedings of Hawaii International Conference on System Sciences*, pages 659–669, January 1992.
- [11] M. Deering. High resolution virtual reality. *Computer Graphics (Proceedings of the SIGGRAPH Conference)*, 26(2):195–202, July 1992.
- [12] D. DeMenthon and L. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1/2):123–141, June 1995.
- [13] M. Dhome, M. Richetin, J.T. Lapreste, and G. Rives. Determination of the attitude of 3d objects from a single perspective view. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(11):1265–1278, 1989.
- [14] D. Drascic, J.J. Grodski, P. Milgram, K. Ruffo, P. Wong, and S. Zhai. Argos: A display system for augmenting reality. In *Formal Video Programme and Proceedings of Conference on Human Factors in Computing Systems (INTERCHI'93)*, page 521, Amsterdam, Netherlands, 1993.
- [15] O. D. Faugeras, Q.-T. Luong, and S.J. Maybank. Camera self-calibration: theory and experiments. In *Proceedings of European Conference on Computer Vision*, pages 321–334, Santa-Margherita, Italy, 1992.
- [16] O. D. Faugeras and G. Toscani. Calibration problem for stereo. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 15–20, Miami Beach, FL, 1986.
- [17] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(7):53–62, July 1993.
- [18] A. Fournier. Illumination problems in computer augmented reality. In *Journée INRIA, Analyse/Synthèse D'Images*, pages 1–21, January 1994.
- [19] S. Ganapathy. Decomposition of transformation matrices for robot vision. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 130–139, Atlanta, Georgia, 1984.
- [20] M. Gleicher and A. Witkin. Through-the-lens camera control. In *Computer Graphics (Proceedings of the SIGGRAPH Conference)*, pages 331–340, Chicago, IL, July 1992.
- [21] S. Gottschalk and J. Hughes. Autocalibration for virtual environments tracking hardware. In *Computer Graphics (Proceedings of the SIGGRAPH Conference)*, pages 65–72, August 1993.

- [22] 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.
- [23] W.E. Grimson. *Object Recognition by Computer*. The MIT Press, Cambridge, MA, 1990.
- [24] R. Horaud, Stéphane Christy, Fadi Dornaika, and Bart Lamiroy. Object pose: Links between paraperspective and perspective. In *Fifth International Conference on Computer Vision*, pages 426–433, June 1995.
- [25] A. Janin, D. Mizell, and T. Caudell. Calibration of head-mounted displays for augmented reality applications. In *Proceedings of the Virtual Reality Annual International Symposium (VRAIS '93)*, pages 246–255, September 1993.
- [26] R. Kumar and A. R. Hanson. Robust methods for estimating pose and sensitivity analysis. *CVGIP: Image Understanding*, 60(3):313–342, November 1994.
- [27] R. K. Lenz and R.Y. 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*, 10:713–720, 1988.
- [28] W. Lorensen, H. Cline, C. Nafis, R. Kikinis, D. Altobelli, and L. Gleason. Enhancing reality in the operating room. In *Proceedings of Visualization '93 Conference*, pages 410–415, Los Alamitos, CA, October 1993. IEEE Computer Society Press.
- [29] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Norwell, MA, 1985.
- [30] 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.
- [31] J.P. Mellor. Realtime camera calibration for enhanced reality visualizations. In *Proceedings of Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed '95) Conference*, pages 471–475, Nice, France, April 1995.
- [32] P. Milgram, S. Zhai, D. Drascic, and J.J. Grodski. Applications of augmented reality for human-robot communication. In *Proceedings of IROS '93: International Conference on Intelligent Robots and Systems*, pages 1467–1472, Yokohama, Japan, July 1993.
- [33] E. Rose, D. Breen, K. Ahlers, C. Crampton, M. Tuceryan, R. Whitaker, and D. Greer. Annotating real-world objects using augmented reality. In *Computer Graphics: Developments in Virtual Environments (Proceedings of CG International '95 Conference)*, pages 357–370, Leeds, UK, June 1995.
- [34] R. Y. Tsai. A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. Research Report RC 11413 (#51342), IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, September 1985.
- [35] P. Wellner. Interacting with paper on the digital desk. *Communications of the ACM*, 36(7):87–96, July 1993.
- [36] 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.
- [37] J. Weng, T. S. Huang, and N. Ahuja. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(5):451–476, May 1989.
- [38] R. Whitaker, C. Crampton, D. Breen, M. Tuceryan, and E. Rose. Object calibration for augmented reality. In *Proceedings of Eurographics '95 Conference*, Maastricht, NL, August 1995.

The technical reports [2,9] are available via anonymous FTP from the site <ftp.ecrc.de>.