

Calibration Requirements and Procedures for Augmented Reality

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

ECRC-95-14

(A revised version appears in the journal *IEEE Transactions on Visualization and Computer Graphics*, Sept 1995)

Calibration Requirements and Procedures for Augmented Reality

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



**European Computer-Industry
Research Centre GmbH
(Forschungszentrum)**

Arabellastraße 17

D-81925 Munich

Germany

Tel. +49 89 9 26 99-0

Fax. +49 89 9 26 99-170

Although every effort has been taken to ensure the accuracy of this report, neither the authors nor the European Computer-Industry Research Centre GmbH make any warranty, express or implied, or assume any legal liability for either the contents or use to which the contents may be put, including any derived works. Permission to copy this report in whole or in part is freely given for non-profit educational and research purposes on condition that such copies include the following:

1. a statement that the contents are the intellectual property of the European Computer-Industry Research Centre GmbH
2. this notice
3. an acknowledgement of the authors and individual contributors to this work

Copying, reproducing or republishing this report by any means, whether electronic or mechanical, for any other purposes requires the express written permission of the European Computer-Industry Research Centre GmbH. Any registered trademarks used in this work are the property of their respective owners.

**For more
information
please**

contact : Mihran Tuceryan
mihran@ecrc.de
Tel: +(49)(89) 92699-125

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 complete computer model of the real world and then, using the augmented reality system developed at ECRC (GRASP) as an example, we describe each of the calibration processes.

Contents

1	Introduction	2
2	Previous Work	4
3	The GRASP system	5
4	Overview of Calibration Requirements	7
	4.1 Necessary Calibrations	7
5	Calibration Procedures and Calculations	12
	5.1 Image Calibration	12
	5.2 Camera Calibration	15
	5.3 Pointer Calibration	21
	5.4 Tracker Transmitter Calibration	24
	5.5 Object Calibration	25
	5.6 Mark Calibration	30
6	Results	32
7	Conclusion	35
8	Acknowledgments	37

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 be provided diagnostic or maintenance data while repairing a complicated automobile, locomotive, or aircraft engine. In this scenario, AR provides 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. The graphics is generated from geometric models of both non-existent (virtual) objects and real objects in the environment. In order for the graphics and the video to align properly, the pose and optical properties of the real and virtual cameras must be the same. The position and orientation of the real and virtual objects in some world coordinate system must also be known. The locations of the geometric models and virtual cameras within the augmented environment may be modified by moving its real counterpart. This is accomplished by tracking the location of the real objects and using this information to update the corresponding transformations within the virtual world. This tracking capability may also be used to manipulate purely virtual objects, ones with no real counterpart, and to locate real objects in the environment. Once these capabilities have been brought together, real objects and computer-generated graphics may be blended together, thus augmenting a dynamic real scene with information stored and processed on a computer. An example of this is presented in Figure 6.1, where the visible components of an automobile engine are labeled in augmented reality [24]. 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). Some 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 the magnetic tracking system is roughly centered on the tracker transmitter box, however, we do not know exactly where the coordinate system is located on the box. Hence, even though the location of the transmitter box can be physically measured in relation to our world coordinate system, we still would not know where the tracker coordinate system is located in the world with great accuracy. Therefore, the location of the tracker coordinate system needs to 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. In this paper, the set of rigorous procedures needed to estimate the parameters of the components comprising the augmented reality system is called *calibration*.

In order for augmented reality to be effective the real and computer-generated objects must be accurately positioned relative to each other and properties of certain devices must be accurately specified. Indeed, a key goal of AR is that the real-world objects and the representations of the virtual entities should blend together and be practically indistinguishable to the user. The success of this illusion depends on a number of factors including:

1. the quality of the computer graphics,
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 for the various calibration steps required. 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. Certain calibration methods have also been utilized in the computer graphics literature, but, we find, not very often and not very rigorously. Because the equipment used in typical augmented reality applications has all the above mentioned shortcomings of using manufacturer specifications and of using physical measurements, we have decided that all of the device and positioning parameters required to get a working augmented reality system should be estimated using mathematical calibration methods. Our aim and contribution in this paper is to systematically look at all the calibration issues arising in augmented reality applications and present methods for solving each one.

The result of these calibrations is that the virtual entities that exist only in the computer's world are accurately displayed and/or that they are registered 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. To this end, we have implemented a set of calibration procedures and techniques which we describe in the context of the GRASP system [2, 3]. Since GRASP is intended for use in a variety of industrial applications, our calibration techniques are designed to be general and robust.

In the paper, we use the term *transformation* in a more general sense than is usually understood in computer graphics. Our use of the word transformation covers 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.

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 in Section 3. Section 4 describes the calibration steps required to produce augmented reality. The procedures and calculations to accomplish the various calibration steps are detailed in Section 5. The paper concludes with examples that demonstrate the results of successful calibration.

2 Previous Work

Research in augmented reality is a recent but expanding activity. We briefly summarize the research conducted to date in this area. Baudel and Beaudouin-Lafon [6] 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. [11] 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 [25] has demonstrated an augmented reality system for office work in the form of virtual desktop on a physical desk. He interacts on this physical desk both with real and virtual documents.

Bajura et al. [5] have used augmented reality in medical applications in which the ultrasound imagery of a patient is superposed on the patient's video image. Once more, the various registration issues, realism, etc. are open research questions which need to be studied and improved. Lorensen et al. [20] use augmented reality system in surgical planning applications. Milgram, Drascic et al. [9, 23] use augmented reality using computer generated stereo graphics to perform telerobotics tasks. Alain Fournier [12] has posed the problems associated with illumination in combining synthetic images with images of real scenes.

Calibration has been an important aspect of research in augmented reality, as well as in other fields, including robotics and computer vision. Camera calibration, in particular, has been studied extensively in the computer vision community (e.g., [19, 22, 26]). Its use in computer graphics, however, has been limited. Deering [8] has explored the methods required to produce accurate high resolution head-tracked stereo display in order to achieve sub-centimeter virtual to physical registration. Azuma and Bishop [4], and Janin et al. [18] describe techniques for calibrating a see-through head-mounted display. Gottschalk and Hughes [15] present a method for auto-calibrating tracking equipment used in AR and VR. Gleicher and Witkin [14] state that their through-the-lens controls may be used to register 3D models with objects in images. Grimson et al. [16] have explored vision techniques to automate the process of registering medical data to a patient's head.

3 The GRASP system

The GRASP system has been developed at ECRC as a platform for research in augmented reality [1, 2]. It has been used to develop several prototype AR applications, including a mechanic's assistant [24] 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.

A schematic of the GRASP hardware configuration is shown in Figure 3.1. The graphical image is generated by the workstation hardware and displayed on the workstation's high resolution monitor. A scan converter is used to convert the graphics displayed on the high resolution monitor to a standard video resolution and format. The scan converter also mixes this generated video signal with the video signal input from the camera. 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

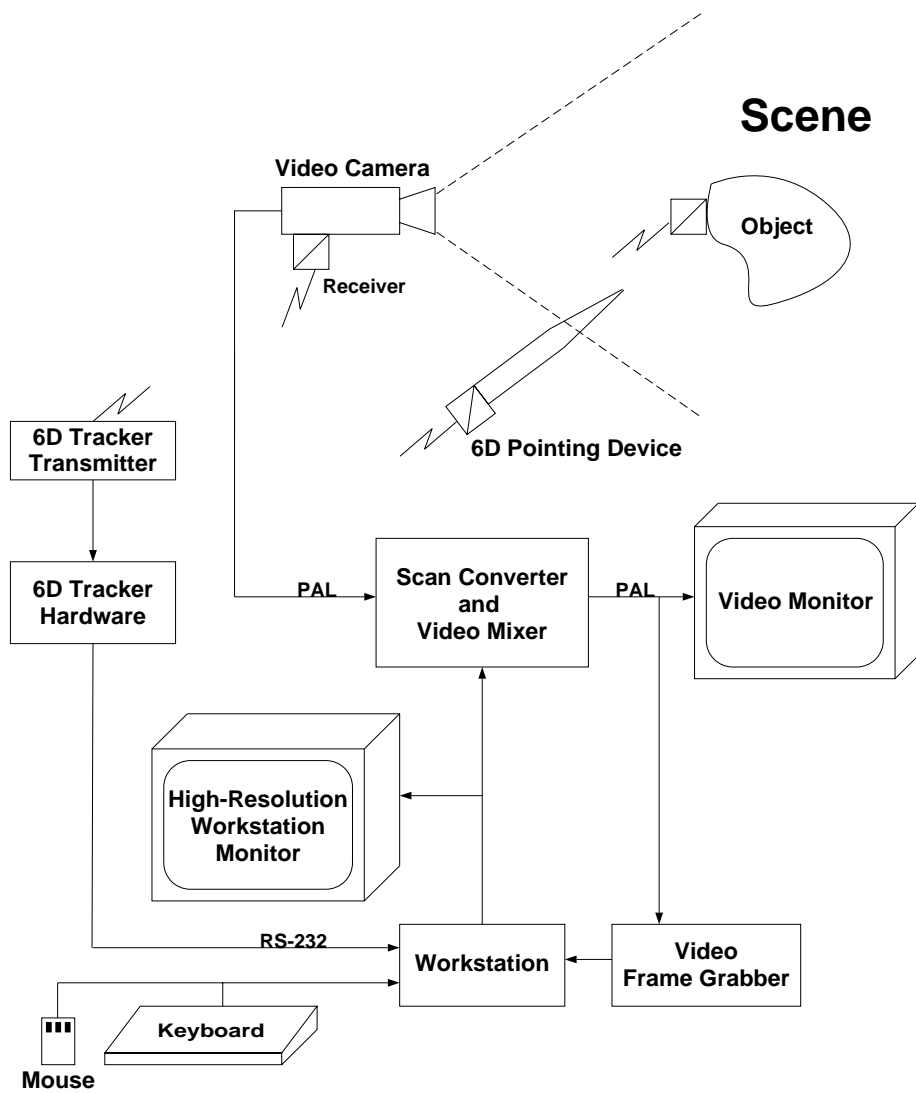


Figure 3.1: The GRASP system hardware configuration.

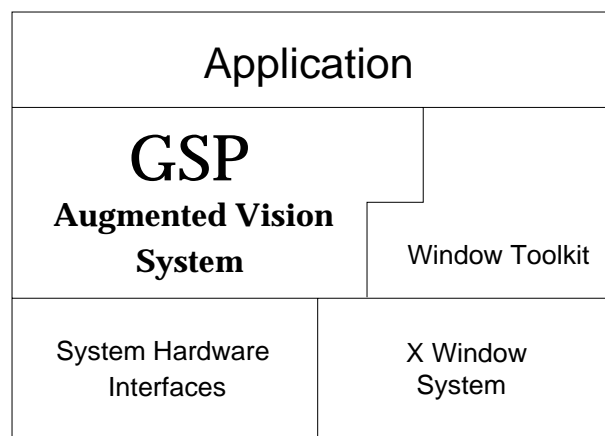


Figure 3.2: The GRASP system software configuration.

camera and the pointing device. A frame grabber is used to grab (digitize) 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 3.2.

4 Overview of Calibration Requirements

In an AR system there are both “real” entities in the user’s environment and virtual entities. Calibration is the process of instantiating parameter values for “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.

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

4.1 Necessary Calibrations

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 4.1 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 effect 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 the GRASP system is the world coordinate system. Therefore, all the coordinate systems shown in Figure 4.1 need to be tied together, the transformations relating each coordinate system as well as the intrinsic parameters of any special devices need to be known. A subset of these transformations are known by direct measurements from

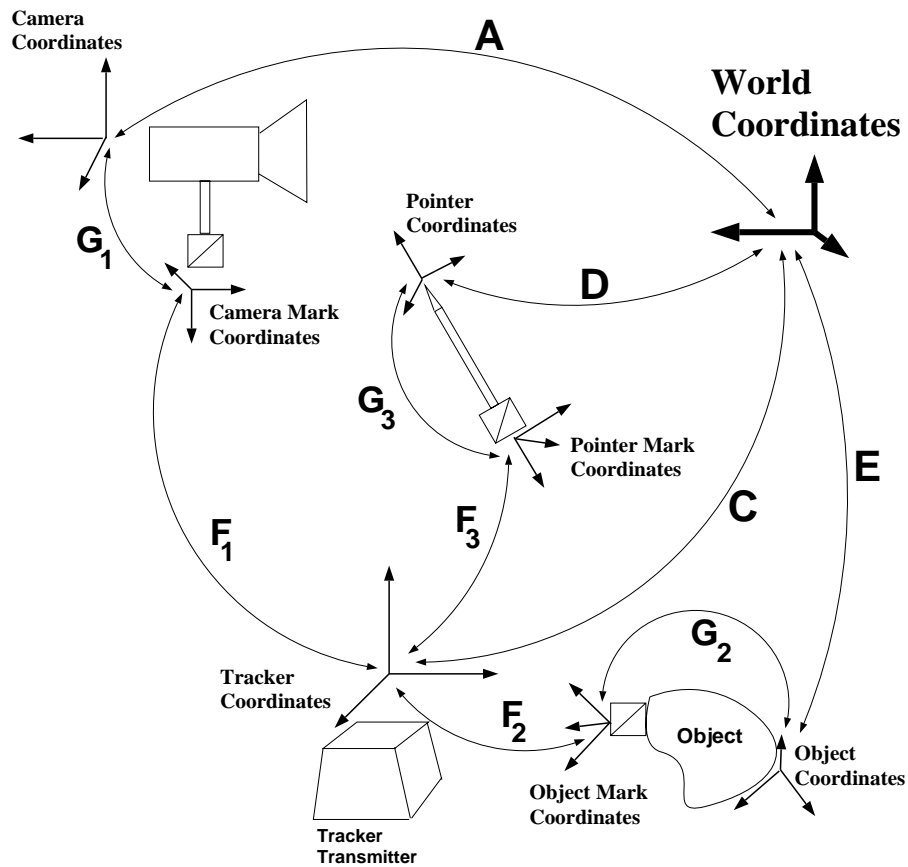


Figure 4.1: GRASP coordinate systems and their relationships.

devices (e.g., those provided by the magnetic tracking system). Some of the transformations are known as a result of an elaborate calibration procedure. And finally, some of the transformations are inferred from the remaining set of known transformations. Some of the intrinsic device properties are also known as a result of calibration procedures (e.g., distortion transformation from the scan converter). The goal in the AR system is to fill in all the gaps that might exist in Figure 4.1 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 effect when a calibration is to be performed. Some of the transformations shown in Figure 4.1 are fixed once and never change again (e.g., G₃); some are fixed at the beginning of each 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 needs to be estimated at the beginning of each session. Finally, the third category of transformations needs

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
A	world-to-camera	varying	camera calibration
G_3	pointer-mark-to-pointer	fixed	pointer calibration
C	world-to-tracker	fixed	tracker transmitter calibration
D	world-to-pointer	varying	from G_3 , C and F_3
E	world-to-object	varying	object calibration
F_i	tracker-to-mark	varying	from tracker system
G_1, G_2	object-to-mark	fixed	mark calibration (from C , E (or A), and F_i)

Table 4.1: Key transformations between GRASP coordinate systems.

to be estimated at the beginning of each session and also be tracked while the program is running.

Considering this framework, we now give an overview of all the calibration requirements in the GRASP system and a brief explanation of each procedure. Each one of these calibration procedures will be presented in detail in Section 5. Table 4.1 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 the virtual camera can be matched 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 reference coordinate system used by the 6-DOF tracker is referred to as the *tracker coordinate system*. Its relationship to world coordinates is defined by the world-to-tracker transformation which is labeled C in Figure 4.1. The tracker coordinate system is centered on the transmitter box. During each session, the transmitter box remains at a fixed location with respect to the world coordinate system. However, there is no guarantee that the box has not been moved between sessions. Therefore, transformation C needs to 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 F_1 , F_2 , and F_3 in the diagram.

For each real object that is to interact with entities in the virtual world, we need

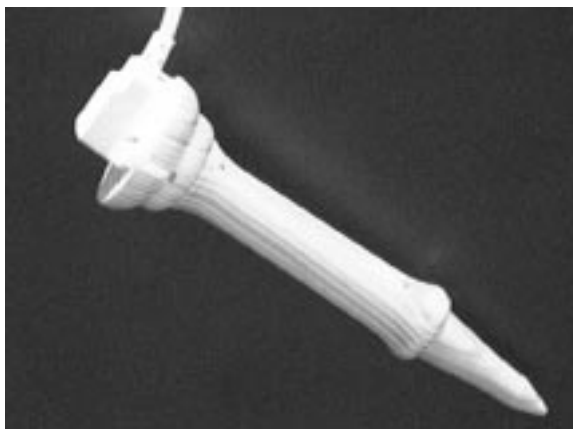


Figure 4.2: The pointer object consists of a wooden wand with a tracker receiver attached to the base and a tapered tip. The tip is considered the “hot-spot” during user interaction.

a model to act as a virtual counterpart for the real object. Each model is defined within its own coordinate system. The position and orientation of an object within the world coordinate system (transformation E) is calculated by a calibration process. Some real objects will be tracked in which case we need to know the fixed, object-to-mark (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 as 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 4.2). The pointer is a particular case of a tracked object with its coordinate system centered on the pointer tip and transformation G_3 representing the transformation between the coordinate system of the pointer and that of 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 different from the other tracked objects.

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 G_3),
4. tracker transmitter calibration (transformations D and C),
5. object calibration (transformation E),

6. tracker mark calibration, one per tracked entity (transformations G_i).

Image Calibration is the process by which the 2D image distortion introduced into the system by the scan converter and frame grabber is estimated. This information is required because subsequent calibration procedures use grabbed images which are taken from the scan converter output in which this distortion is embedded. These images can be camera images or computer generated images, but in either case, they go through the scan converter (Figure 3.1).

Camera Calibration is the process by which the camera location and orientation as well as the intrinsic camera parameters (focal length, image center, and aspect ratio) are calculated. This process calculates the transformation labeled A in Figure 4.1.

Pointer Calibration is used to calculate 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 (the transformation labeled G_3). This step is necessary before many of the subsequent calibration steps can be completed as they require the use of the pointer by the user to pick points on real objects. Transformation G_3 can be estimated once and stored and would be valid as long as the rigid attachment of the receiver does not change.

Tracker Transmitter Calibration calculates the position and orientation of the tracker's coordinate system within the world coordinate system (the transformation represented by the arc labeled C in Figure 4.1). This transformation is calculated indirectly after calculating the transformations D and G_3 and measuring transformation F_3 .

Object Calibration (object registration) is the process by which the position and orientation of a real object is calculated such that a virtual counterpart can be aligned with it by being placed at the corresponding location, with the appropriate orientation, within the virtual world (transformation E in the diagram). Some real objects will subsequently have their movements tracked by their virtual counterparts, in which case the corresponding tracker marks must also be calibrated.

The *Mark Calibration* calculates the transformation between the coordinate system of the tracker mark and the coordinate system of the object itself (transformations G_i) using the measured transformations F_i , along with the previously calculated world-to-object transformations (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 will be 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 [7].

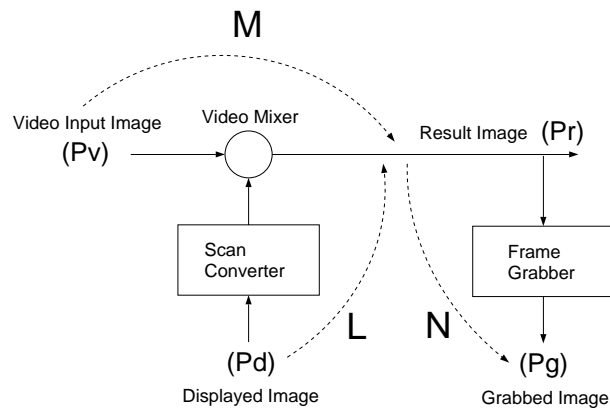


Figure 5.1: 2D image transformations associated with the image formation process.

During the execution of applications, it is transformations **A**, **D**, and **E** that actually interest us, but it is changes in the transformations F_i that the application receives 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**.

5 Calibration Procedures and Calculations

This section details the calibration steps presented in the previous section.

5.1 Image Calibration

Figure 5.1 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 which is used in various calibration procedures to be described below.

More precisely, the computer image, P_d , is converted to a standard video format by the scan converter and mixed 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. In fact, since 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 that

of 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.

As shown in Figure 5.1 the distortion on the video input signal caused by the video mixing process is labeled 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. 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:

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

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

$$\mathbf{M}p_v = \mathbf{L}p_d. \tag{2}$$

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{NL}).

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., $(\mathbf{NL})^{-1}$. Consequently, after the camera calibration has been completed and the computer graphics model set up correctly, the following relationship will hold:

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

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.

5.1.1 Image Model

The distortion (\mathbf{NL} in Figure 5.1) resulting from combining the camera video signal and the computer generated graphics is modeled as a combination of a

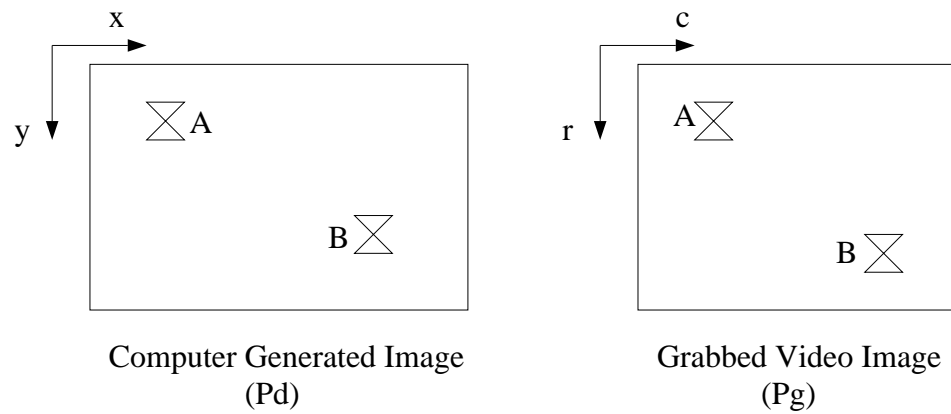


Figure 5.2: The relationship between the computer-generated and the displayed video images. The distortion is modeled as a two-dimensional scaling and translation transformation.

two dimensional scaling and translation transformations. 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 5.2. The two are related by:

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

5.1.2 Calibration Procedure

The image distortion parameters described above, (k_x, k_y, t_x, t_y) , are estimated using an image calibration procedure. 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 5.2),
- calculate the translational and scale factor relationship between the locations of the points in the original image and their location in the resulting image.

This process is currently implemented as a manual procedure requiring the user to pick the points in the resulting image.

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):

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

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

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

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

5.2 Camera Calibration

Camera calibration is the process by which the extrinsic and intrinsic parameters of the camera are calculated. We first give the mathematical model of the camera and then describe the calibration procedure.

5.2.1 Camera Model

A simple pinhole model (Figure 5.3) is used for the camera, which defines the basic projective imaging geometry with which the 3D objects are projected onto the 2D image surface. This is an ideal model commonly used in computer graphics and computer vision to capture the imaging geometry. It does not account for certain optical effects (such as non-linear distortions) that are often properties of real cameras. There are different ways of setting up the coordinate systems, and in our model we use a right-handed coordinate system in which the center of projection is at the origin and the image plane is at a distance f (focal length) away from it.

The 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 has to be accounted for in the mathematical model. The relationships between the screen coordinates, the pinhole model, and the world coordinates is shown in Figure 5.4 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

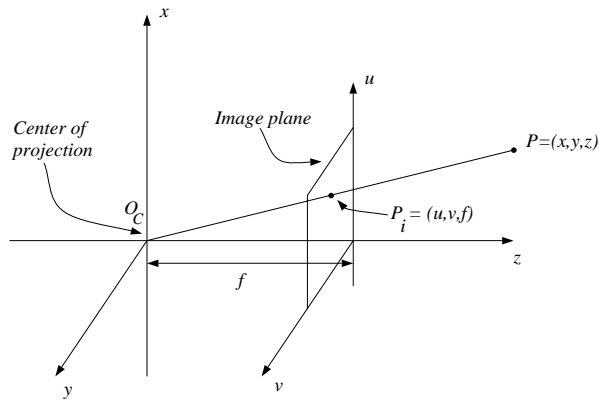


Figure 5.3: The geometry of the simple pinhole camera model for perspective transformation.

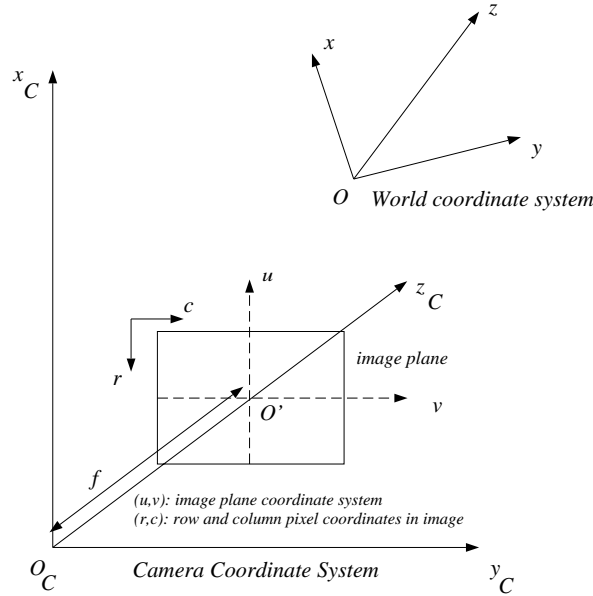


Figure 5.4: The various coordinate systems with respect to each other.

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

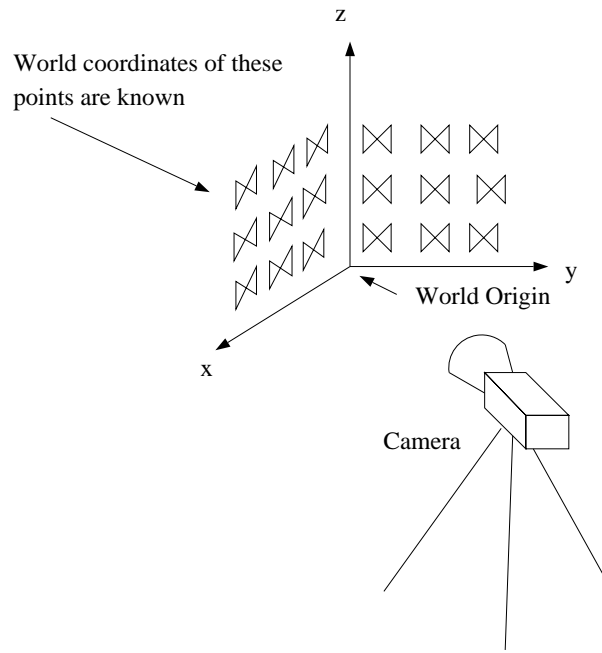


Figure 5.5: The set-up with which the camera is calibrated.

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 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

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

5.2.2 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 as 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 5.5 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. Again, as with the image calibration procedure, the process of locating the points of interest in the grabbed image is currently performed manually by the user.
- 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.

5.2.3 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 the 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,

$$\mathbf{AW} = 0. \tag{14}$$

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 [10, 13, 26]:

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

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

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{21} & w_{22} & w_{23} & w_{31} & w_{32} & w_{33} & w_4 & w_5 & w_6 \end{bmatrix}^T.$$

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 make sure that the world coordinate system is defined so that $t_3 \neq 0$). 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:

$$\mathbf{A}'\mathbf{W}' + \mathbf{B}' = \mathbf{0}, \tag{18}$$

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

$$\min_{\mathbf{W}'} \|\mathbf{A}'\mathbf{W}' + \mathbf{B}'\|. \tag{19}$$

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 [26, 27] in which a closed form solution is provided for computing $\tilde{\mathbf{R}}$. We have used this second approach to enforce

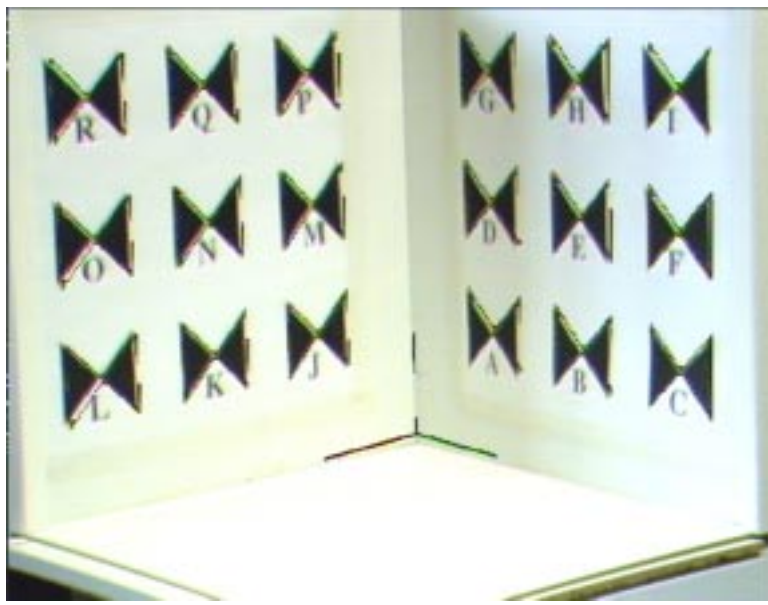


Figure 5.6: Result of camera calibration.

orthonormality of the rotation matrix. The $\tilde{\mathbf{R}}$ then is used to estimate the rest of the parameters as shown in Equation 23:

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

Figure 5.6 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 superposed 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.

5.3 Pointer Calibration

As has been mentioned earlier, the GRASP system provides for a number of tracked objects in addition to the camera. The *pointer* object is a special one of these tracked objects in that it is used in some of the other calibrations. Therefore, it has to be calibrated before tracker calibration or object calibration can be performed.

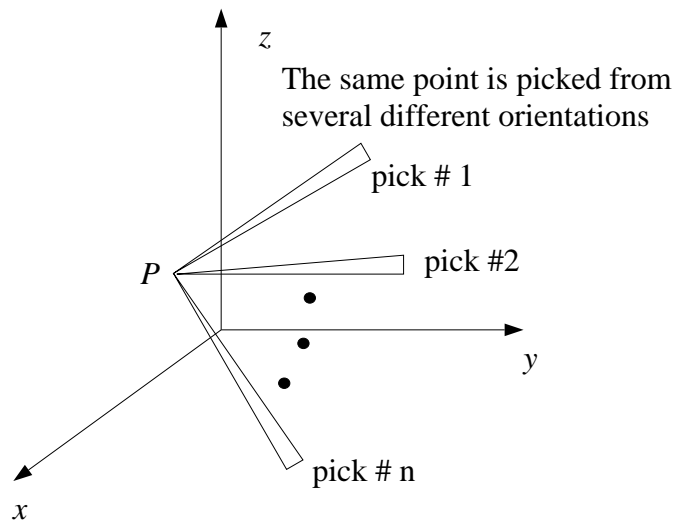


Figure 5.7: Schematic of the procedure for pointer calibration.

The pointer currently takes the form of 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 4.2). 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 5.7). 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.

5.3.1 Pointer Model

The detailed geometry of the relation between the tracker coordinate system and the world coordinate system is shown in Figure 5.8. The mark is attached to the pointer device for which the magnetic tracking system is providing readings.

The geometry of the pointer, as it will be used to calibrate the tracker, can be defined as follows, where the tip of the pointer is related to the tracker measurements by a rigid transformation given by the formula:

$$p_w = p_m + \mathbf{R}_m p_t, \quad (24)$$

where p_w is the position vector representing the tip of the pointer in world coordinates, p_t is the offset on the pointer where the receiver is attached, \mathbf{R}_m is

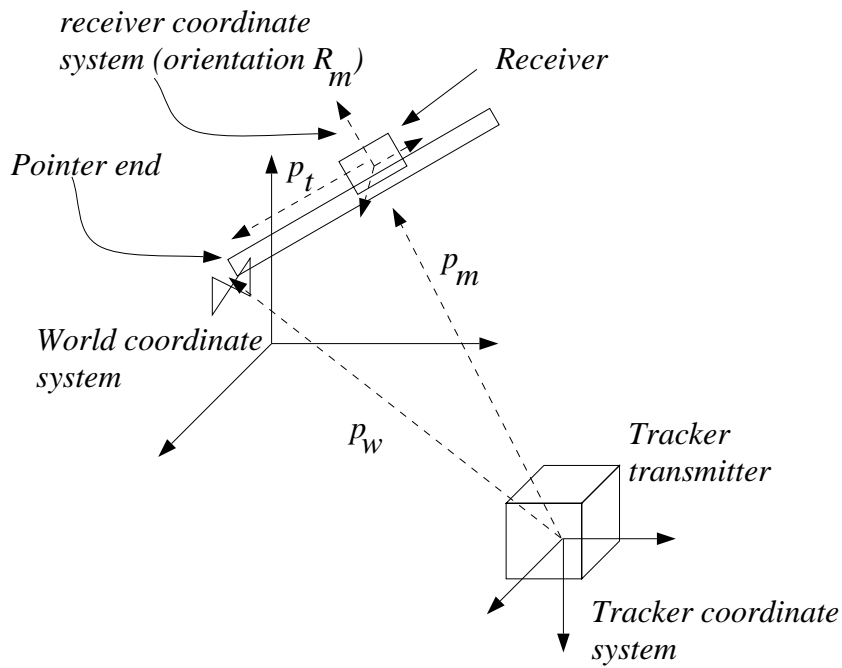


Figure 5.8: The relation between the world coordinate system and the tracker coordinate system.

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.

5.3.2 Calibration Procedure

The result of the pointer calibration is the estimation of 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 the orientations are different, then we have the following equation which must be solved:

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

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.

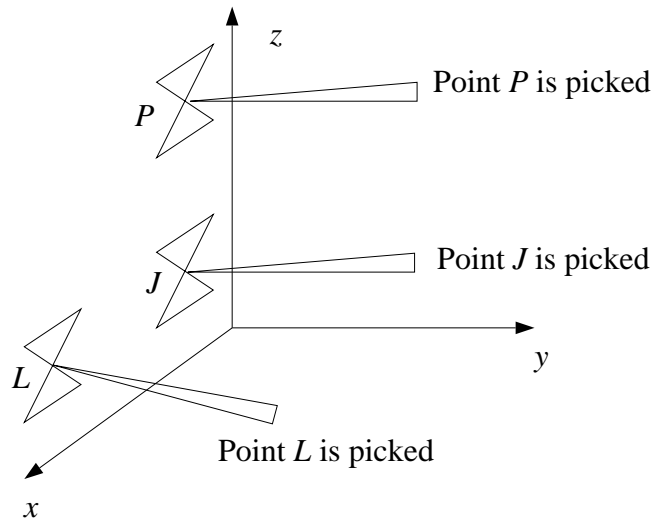


Figure 5.9: Schematic of the procedure for the tracker transmitter calibration.

5.4 Tracker Transmitter Calibration

The calibration procedures described above will have provided us with most of the information we need to calibrate our system as a whole, but we still need to locate the tracker coordinate system within the world coordinates (i.e., calculate transformation C in Figure 4.1).

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 5.9). By using the calibration grid we used for camera calibration, for which the locations of the points in the world coordinate system are known, we can arrange that we also know the location of the points used for the tracker calibration (J , P , L).

Once p_t is estimated as described above, then the tracker-to-camera transformation is computed by using the p_w values for the three points, J , P , and L . 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,

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

$$p_{wP} = p_{mP} + \mathbf{R}_{mP} p_{tP}. \tag{27}$$

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

$$\mathbf{x} = p_{wL} - p_{wJ},$$

and the z -direction vector is computed using

$$\mathbf{z} = p_{wP} - p_{wJ}.$$

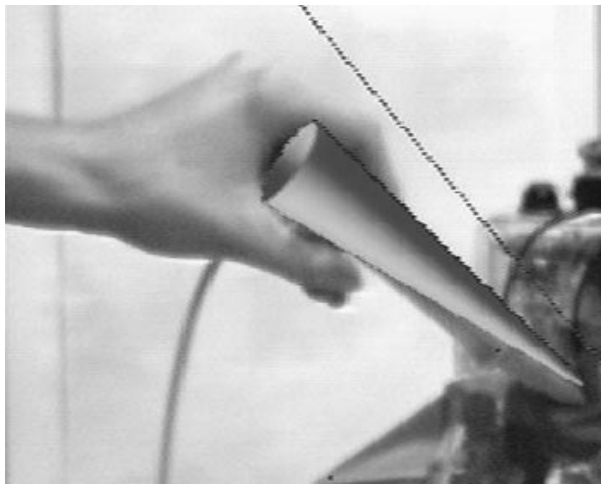


Figure 5.10: The model of a pointer is aligned with the physical pointing device by using the results of pointer and tracker calibration.

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

$$\mathbf{R} = \begin{bmatrix} \mathbf{x} & \mathbf{y} & \mathbf{z} \end{bmatrix}^T,$$

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

$$p_{wJ} = p_{mJ} + \mathbf{R}_{mJ}p_0.$$

Figure 5.10 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 which shows it as superposed on the image of the physical pointer.

5.5 Object Calibration

Object calibration is the process by which the location and orientation of a real-world object is calculated so that a virtual counterpart can be aligned with it by being placed at the corresponding location, with the appropriate orientation, within the virtual world. 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 counterparts.

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 by which these landmarks can be identified by the user and object calibration accomplished. The first is an image based approach in which the images of the landmark points are identified and the object pose is estimated using the framework of the camera model described in section 5.2.1. The second is by the user picking the 3D coordinates of the landmark points directly on the physical object. 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.

5.5.1 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 [17, 21], 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,

$$(r_i - r_0)x_i r_{31} + (r_i - r_0)y_i r_{32}$$

$$\begin{aligned}
&+(r_i - r_0)z_i r_{33} + (r_i - r_0)t_3 \\
&\quad - f_u x_i r_{11} - f_u y_i r_{12} \\
&\quad - f_u z_i r_{13} - f_u t_1 = 0, \text{ and}
\end{aligned} \tag{28}$$

$$\begin{aligned}
&(c_i - c_0)x_i r_{31} + (c_i - c_0)y_i r_{32} \\
&\quad + (c_i - c_0)z_i r_{33} + (c_i - c_0)t_3 \\
&\quad - f_v x_i r_{21} - f_v y_i r_{22} \\
&\quad - f_v z_i r_{23} - f_v t_2 = 0,
\end{aligned} \tag{29}$$

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:

$$\mathbf{A}\mathbf{x} = 0, \tag{30}$$

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 5.2.3. 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 measurements of the model, and the camera calibration give solutions that are not rigid transformations.

Thus, Equations 28 and 29 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 28 and 29 as a homogeneous linear system with twelve unknowns (obtained by multiplying by t_3) and simultaneously enforce rigidity by solving the nonlinear optimization problem

$$\|\mathbf{A}\mathbf{x}\|^2 + \alpha \|\mathbf{R}^T \mathbf{R} - \mathbf{I}\|^2, \tag{31}$$

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 28 and 29, and n is the number of landmark points.

Since this is a non-linear optimization problem, it is important to select the starting point properly. This is done by using the solution of the Equation 30 as

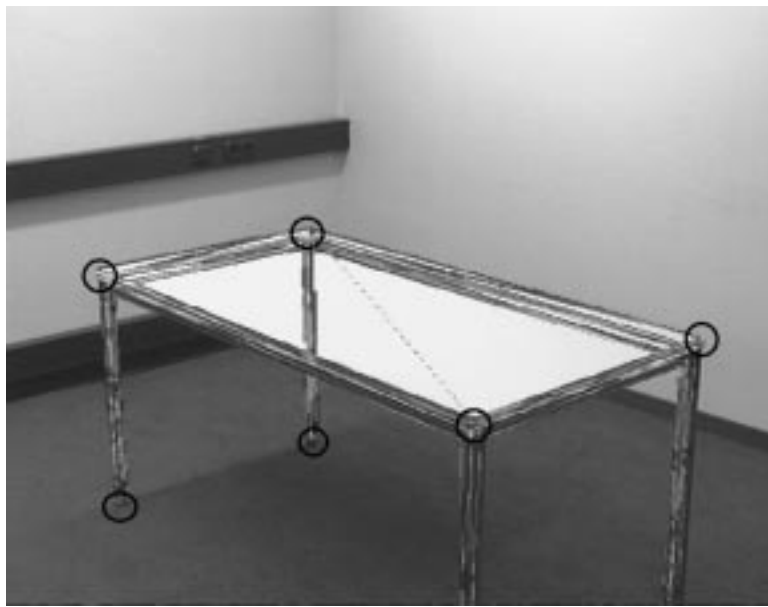


Figure 5.11: The wire-frame of image of a 3D model of a table is aligned with its physical counterpart. This alignment is using the result of camera calibration as well as object calibration.

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 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 2-3 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 5.11 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 superposed on the image of the physical table.

5.5.2 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 5.12, 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.

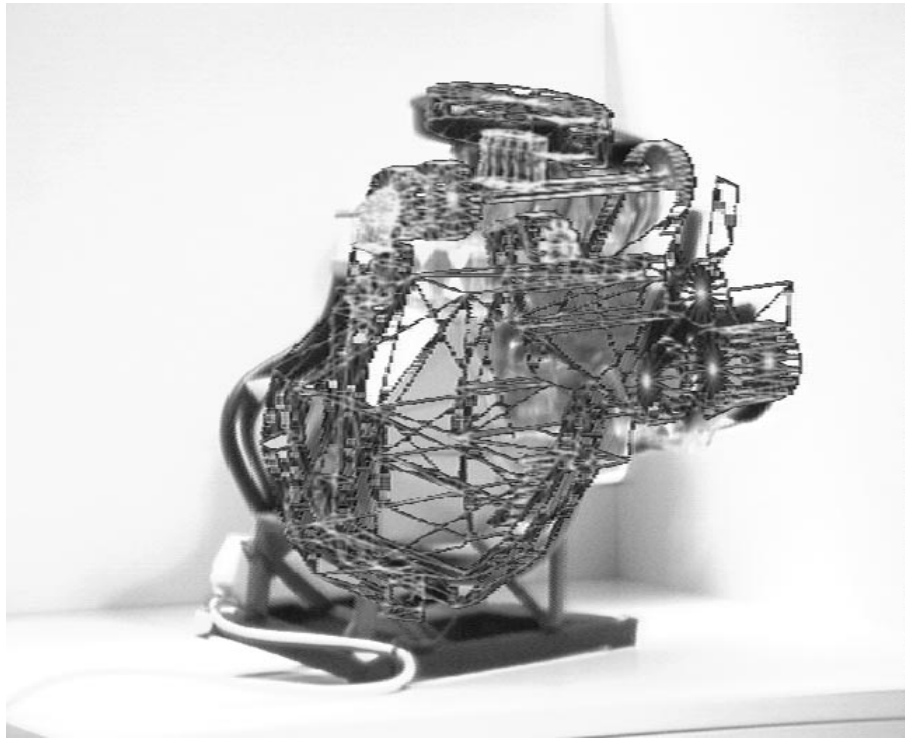


Figure 5.12: A wire-frame engine model registered to a real model engine using an image-based calibration.



Figure 5.13: When the model is turned and its movements tracked, the graphics show the misalignment in the camera's z -direction.

This error becomes evident as the object is turned as in Figure 5.13.

It is because of such errors that we have developed a procedure for using the 3D pointing device to calibrate objects. This is the same pointing device which is shown in Figure 4.2 and which is already calibrated according to the procedure described in Section 5.3. 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 and the world coordinates of the landmark points are related by:

$$p_i^w = \mathbf{R}p_i^l + \mathbf{T} \quad \text{for } i = 1, \dots, n \quad (32)$$

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 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 non-linear optimization problem:

$$\|p_i^w - \mathbf{R}p_i^l - \mathbf{T}\|^2 + \alpha\|\mathbf{R}^T\mathbf{R} - \mathbf{I}\|^2. \quad (33)$$

This procedure provides the object-to-world transformation that is needed for object registration. Figure 5.14 shows a model engine which has been calibrated in such a manner. Rotations of the engine (Figure 5.15) show that this calibration does not suffer from the *depth problem* of the image-based approach.

5.6 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

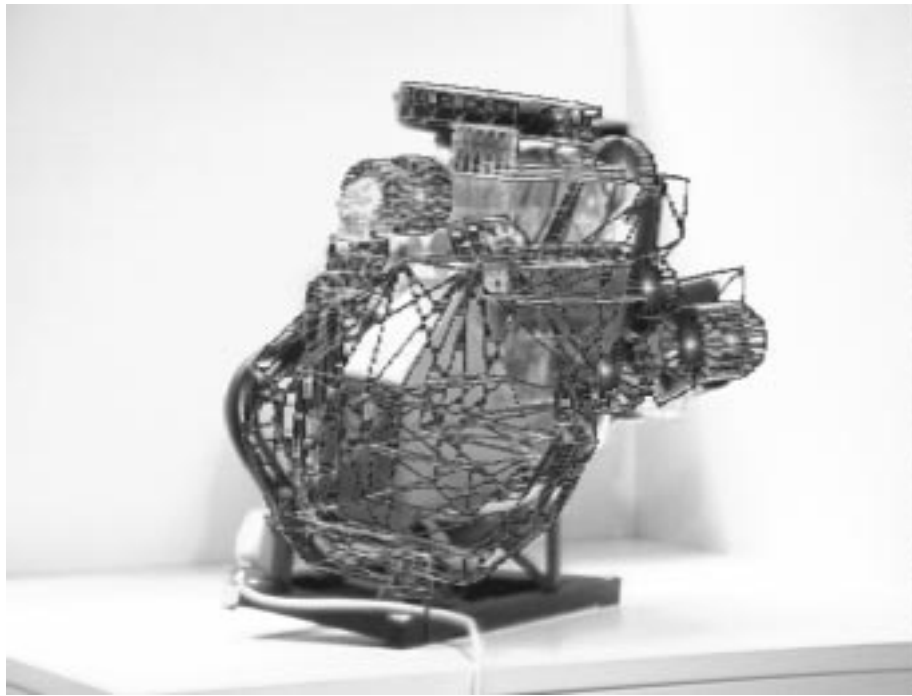


Figure 5.14: A wire-frame engine model registered to a real model engine using an pointer-based calibration.

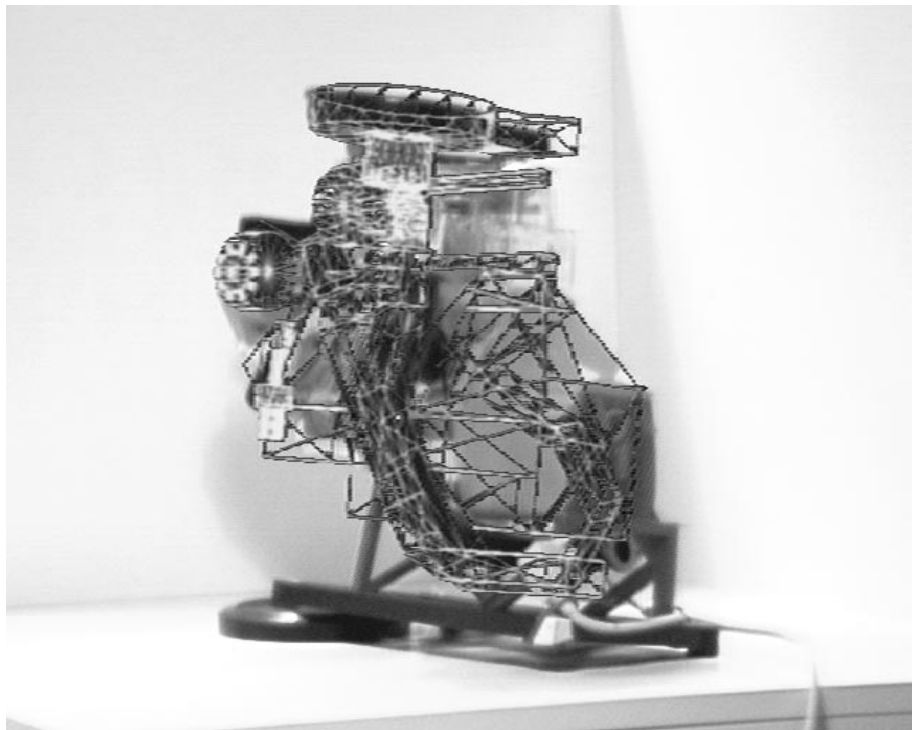


Figure 5.15: When the model is turned and its movements tracked, the graphics show the correct alignment in the camera's z -direction.

physical objects in a rigid manner, but this rigid relationship must be determined in order to utilize the tracker readings.

This estimation of the mark position with respect to the object to which it is attached is accomplished by using the various transformations already computed. Referring to Figure 4.1, the quantities to be computed by the mark calibration step are G_1 (mark position with respect to the camera coordinate system) and G_2 (mark position with respect to the object coordinate system).

To compute G_1 , we use the previously estimated transformations A (from camera calibration), C (from tracker transmitter calibration), and F_1 (measurement read by the tracker). Then G_1 is computed by

$$G_1 = AC^{-1}F_1. \quad (34)$$

Similarly, G_2 is estimated using transformations E (from object calibration), transformation C , and transformation F_2 . The expression for computing G_2 is:

$$G_2 = EC^{-1}F_2. \quad (35)$$

6 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 [24] and an interior design tool [3].

Figures 6.1 and 6.2 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 as a result of 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

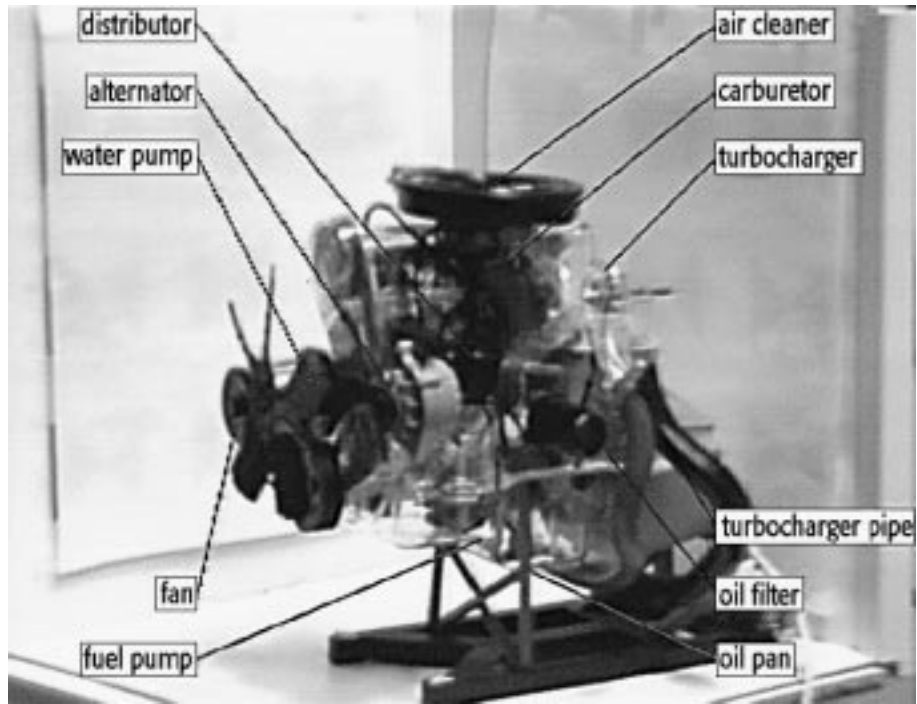


Figure 6.1: A real model engine annotated in augmented reality.

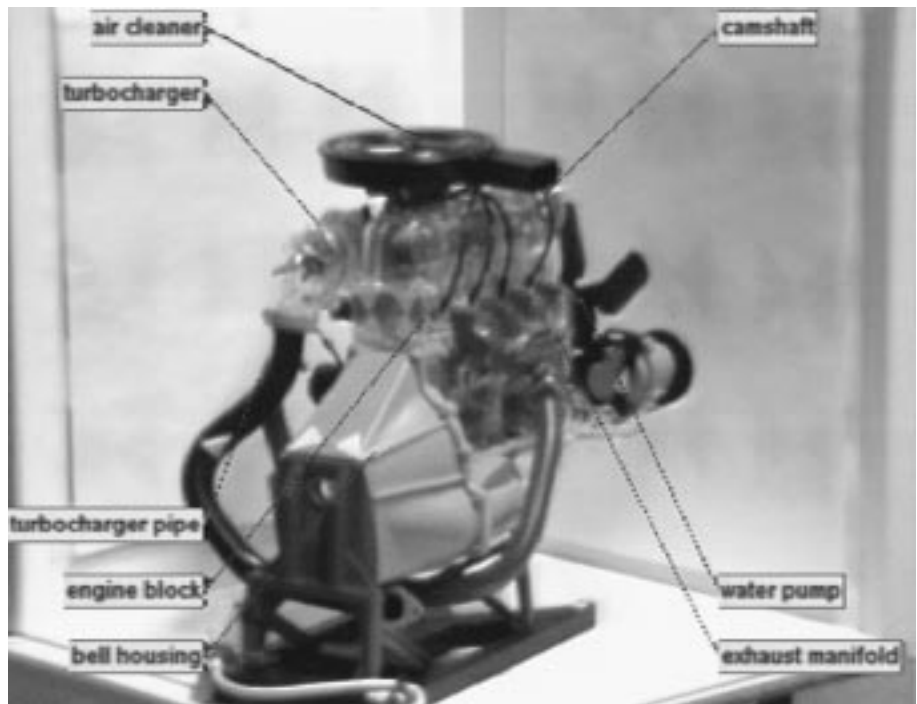


Figure 6.2: As the model is turned only visible components are annotated.



Figure 6.3: A real table occluding two virtual chairs.

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.

Figure 6.3 shows the result of a successful calibration process in the interior design tool. This example shows the interaction of real and virtual objects where a real table is shown as occluding two virtual chairs [7]. 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. In this example, 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 5.11.

7 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 realistic and accurate augmented reality systems.

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.
- 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 measurement problems which often arise in computer graphics and in augmented reality. A

major goal is to make these measurements 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.

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 that has been made in our approach is to keep the user in the loop 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, using the rigor of computer vision, and involve the user only when it is necessary.

One of the factors that affect the accuracy of the camera calibration process, and consequently many of the following steps, is the nonlinear lens distortions of the physical camera. The nonlinear distortions of the camera mean that the pinhole model assumed in modeling and calibrating the camera does not hold any longer and, therefore, the parameters estimated from such a camera are not correct. We have, in fact, seen this effect very clearly in 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 [26]. 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 do rendering 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.

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 future is to minimize the involvement by 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.

8 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.

Bibliography

- [1] K. Ahlers, D. Breen, C. Crampton, E. Rose, M. Tuceryan, R. Whitaker, and D. Greer. An augmented vision system for industrial applications. In *Telemanipulators 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. Baudel and M. Beaudouin-Lafon. Charade: Remote control of objects using freehand gestures. *Communications of the ACM*, 36(7):28–35, July 1993.
- [7] 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.
- [8] M. Deering. High resolution virtual reality. *Computer Graphics (Proceedings of the SIGGRAPH Conference)*, 26(2):195–202, July 1992.
- [9] 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.
- [10] 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.
- [11] S. Feiner, B. MacIntyre, and D. Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(7):53–62, July 1993.

- [12] A. Fournier. Illumination problems in computer augmented reality. In *Journée INRIA, Analyse/Synthèse D'Images*, pages 1–21, January 1994.
- [13] 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.
- [14] 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.
- [15] S. Gottschalk and J. Hughes. Autocalibration for virtual environments tracking hardware. In *Computer Graphics (Proceedings of the SIGGRAPH Conference)*, pages 65–72, August 1993.
- [16] 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.
- [17] W.E. Grimson. *Object Recognition by Computer*. The MIT Press, Cambridge, MA, 1990.
- [18] 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.
- [19] 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.
- [20] 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.
- [21] D. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, Norwell, MA, 1985.
- [22] S. J. Maybank and O. D. Faugeras. A theory of self calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.
- [23] 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.

- [24] 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.
- [25] P. Wellner. Interacting with paper on the digital desk. *Communications of the ACM*, 36(7):87–96, July 1993.
- [26] 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.
- [27] 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.

The technical reports [2, 3, 7] are available via anonymous FTP from the site `ftp.ecrc.de`.