A Fast, Affordable System for Augmented Reality

David LaRose

CMU-RI-TR-98-21

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania

Submitted in partial fulfillment of the requirements for the degree of Master of Science in the field of Electrical and Computer Engineering

April 30, 1998

Advisor:
Dr. Takeo Kanade

Abstract:

This paper presents a framework for augmented reality. Head tracking is accomplished using a head-mounted camera, which uses an optical flow based algorithm to track patterns on the ceiling. The position and orientation of the ceiling need not be known a priori, and the necessary calibration, except for radial lens distortion, can be performed on the fly using simple procedures.

1.0 Introduction

In recent years, there has been an explosion of interest in *virtual reality* systems. Virtual reality is relevant to many applications involving data visualization, communications, and immersive entertainment. A similar area, with perhaps even more commercial applications than virtual reality, is *augmented reality.*

Augmented reality systems differ from virtual reality systems in that the user is not completely immersed in the virtual environment. In augmented reality systems, a heads-up display is used to superimpose computer generated graphics on the user's view of the real world. The superimposed images supplement the information available to the user in the natural scene. For example, an augmented reality system could be used to help a repair technician find the appropriate adjustment points in a complicated piece of machinery, or to help a surgeon by superimposing CT data on a patient's body, essentially giving a surgeon x-ray vision.

Despite its potential, the development of functional AR systems faces several technical challenges. In many AR applications, it is crucial that the synthetic images be registered precisely with the real world. The degree of accuracy required depends on the task, but in many cases the requirements are quite stringent. Furthermore, many tasks require large motions of the user's head, which place demands on the sensors which track head motion. Finally, in nearly all cases display updates must occur with latency of only a fraction of a second. These technical challenges have prevented the development of viable, inexpensive AR systems for precise applications.

We present a fast, inexpensive augmented reality system designed for use with an optical see-through head mounted display. The system is designed to track a user's head position using a CCD camera as a sensor. The camera is rigidly mounted on the user's heads-up display, and head position is tracked by observing a reference pattern printed on a nearby wall or ceiling. By extending the reference pattern, the working volume of the sensor can be made arbitrarily large. The system software runs on readily available computer hardware,

and we anticipate that an inexpensive back or waist mounted version could be easily developed.

2.0 Previous Work

Current augmented reality systems differ from each other primarily in three ways: the display technology used to overlay synthesized graphics on the user's field of view, the sensing technology used to track the user's head, and the calibration method used to determine system parameters.

Many research projects in augmented reality have used optical see-through head mounted displays [Azuma and Bishop, 1994] [Caudell and Mizell, 1992] [Janin et al., 1994]. These displays work by optically combining light from the environment with the overlay images. The combination is done using lenses, half-silvered mirrors, or other optical components. The principal advantage of this type of display is that the user's view of the real world is substantially unobstructed. Consequently, the user has a high resolution, high contrast view of the workspace. One disadvantage of optical see-through head mounted displays is that the optics used to combine the images typically have a narrow field of view, and also somewhat decrease the light intensity reaching the user's eyes. Another disadvantage is that the software in the augmented reality system has no access to the combined image (natural scene plus overlay), so correcting registration errors and establishing system calibration are difficult [Azuma, 1995].

A second category of display system for augmented reality is video based display, which is typically used in medical augmented reality applications [Taylor et al., 1994] [Taylor et al., 1992]. In this type of display, the user views the workspace through one or two video cameras, which may be head mounted. The real and synthesized images are composited as video signals, and presented to the user through a video display which occludes the user's natural view of the environment. While this type of display provides flexibility in video compositing strategies, and gives the underlying software access to the combined images, the workspace view lacks the fidelity of natural vision, and the user's view of the workspace

is from the perspective of the system video camera(s), which generally doesn't match that of the user's eye(s) [Azuma, 1995].

Four types of sensors have traditionally been used for head tracking in augmented reality applications. Mechanical sensors measure the position of the user's head using an attached linkage. This type of sensor is typically very accurate, and can meet the bandwidth requirements of augmented reality, but is often somewhat cumbersome, and restricts the user's range of motion. Magnetic position sensors (polhemus, etc.) have seen wide use in virtual reality applications, and limited use in augmented reality [Janin et al., 1993] [Caudell and Mizell, 1992]. These sensors are inexpensive and readily available, but data rates are typically slow, and their accuracy suffers in applications where a large working volume is required, or where there are nearby ferromagnetic objects such as steel wall studs. Acoustic position sensors are inexpensive, fast, and accurate, but latency increases with distance between the acoustic transmitter and receiver [Meyer et al., 1992]. Optical sensors using video cameras have the potiential to be inexpensive, fast, accurate, and offer large working volume. Unfortunately, systems to date either require large arrays of markers, such as LEDs, to be installed at precise locations in the workspace, or use custom camera hardware [Azuma and Bishop, 1994], or have limited working volume [Janin et al., 1994]. One disadvantage of optical position sensors is that there must be an unobstructed line of sight between the sensor and target [Meyer et al., 1992].

Calibration strategies typically fall into two categories. In the first strategy, measurements are taken, often by asking the user to move his/her head until specified points or crosshairs in the real and synthesized images are aligned, and the measurements are used to calculate the system parameters directly [Azuma and Bishop, 1994] [Caudell and Mizell, 1992]. These calibration strategies can be made fairly quick, at the expense of vulnerability to measurement errors. Adding redundant measurements, or repeating the measurements, makes the procedure more robust, but also more time consuming. The biggest disadvantage of this type of strategy is that the alignment steps are typically cumbersome, requiring the user to place his head in a calibration jig, or return to a specified point in the workspace. This makes it hard to recalibrate if the headset is accidentally moved during system operation.

In the second type of calibration strategy, recovery of system parameters is cast as an optimization problem [Janin et al., 1993]. Calibration constraints, usually point feature matches in the real and synthesized images are collected, and an optimization routine is used to find the set of system parameters which most nearly satisfies the constraints.

3.0 Background Information

3.1 Camera Model

We model the sensing camera using the pinhole camera model described in [Faugeras, 1993]. According to this model, light rays emitted from any point x in the real world are recorded by the imaging system only when they strike the sensing array, or retinal plane, of the camera. The imaging optics allow the sensing array to be hit only by rays which pass through an imaginary point, in 3D space, called the optical center of the camera (see figure 1). The projected light rays form an image on the retinal plane.

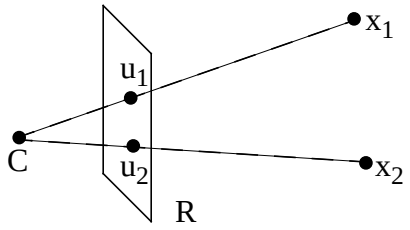

Figure 1: Pinhole Camera Model. The optical center of the camera is denoted by C. Light rays from objects at 3D coordinates x_1 and x_2 intersect the retinal plane, R, at points u_1 and u_2

The geometry of this camera model is most conveniently expressed using 2- and 3-dimensional projective spaces. Projective geometry is a rich area, and we present only a few details here. For a more complete description, please refer to [Faugeras, 1993].

Points in an n-dimensional projective space are represented by vectors with n+1 elements. A point in a three dimensional projective space is represented by a four element vector, and a point in a two dimensional projective space is represented by a 3 element vector. The mapping between points and vectors is not one to one; each point in a projective space can be represented by many different vectors. Vectors in a projective space are considered to represent the same point if they are scalar multiples of each other. That is to say, the 3 dimensional projective vectors $[1, 2, 4, 1]^T$ and $[3, 6, 12, 3]^T$ are considered equivalent.

One way to visualize the relationship between points and vectors in projective space is to imagine that the points are mapped onto an n dimensional hypersphere centered at the origin. Collinear vectors are considered equivalent, and represent the point which is mapped to their intersection with the hypersphere.

In this paper, we adopt the convention of representing the 3D point (x, y, z) by the 3 dimensional projective vector $[x, y, z, 1]^T$ (and its scalar multiples). In this case we say that the point is expressed in *homogeneous coordinates*.

We represent linear transformations in projective spaces using matrices, as we do in linear spaces. We transform a vector in the usual way: by left multiplying it using the appropriate matrix. A linear transformation from n dimensional projective space to n dimensional projective space is represented with an n+1 by n+1 matrix. As is the case with projective vectors, two such transformation matrices are equivalent if they are scalar multiples of each other.

Following figure 2, we define a two dimensional coordinate system ($\mathbf{u_c}$,$\mathbf{v_c}$) in the retinal plane, R. This coordinate system is chosen so that light striking R at the ($\mathbf{u_c}$,$\mathbf{v_c}$) coordinate (i, j) contributes to the activation of the $(i, j)^{th}$ pixel in the image returned by the camera*. We also define a 3 dimensional coordinate system ($\mathbf{x_c}$,$\mathbf{y_c}$,$\mathbf{z_c}$) with its origin at the camera's optical center, C, and with the $\mathbf{z_c}$ axis perpendicular to R. We orient the 3 dimensional coordinate system so that $\mathbf{x_c}$ lies along the same direction as $\mathbf{u_c}$, and $\mathbf{y_c}$ lies along the same direc-

* Implicit in this choice of coordinate system is the assumption that the axes of the camera imaging array are orthogonal. This assumption is generally valid for modern CCD cameras.

tion as $\mathbf{v_c}$. We permit the units of $\mathbf{u_c}$ and $\mathbf{v_c}$ to differ from those of $\mathbf{x_c}$, $\mathbf{y_c}$, and $\mathbf{z_c}$ by two different scale factors k_u and k_v, and denote the point where $\mathbf{z_c}$ intersects R as (u_0,v_0). We can express the projection of a 3D point (x_c,y_c,z_c) into the image using homogeneous coordinates [Faugeras, 1993]

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} \approx \begin{bmatrix} f_c k_u & 0 & u_0 & 0 \\ 0 & f_c k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{1}$$

where we use the congruence operator to denote projective equality (equality up to a scale factor), and f is the distance along $\mathbf{z_c}$ from C to the plane R.

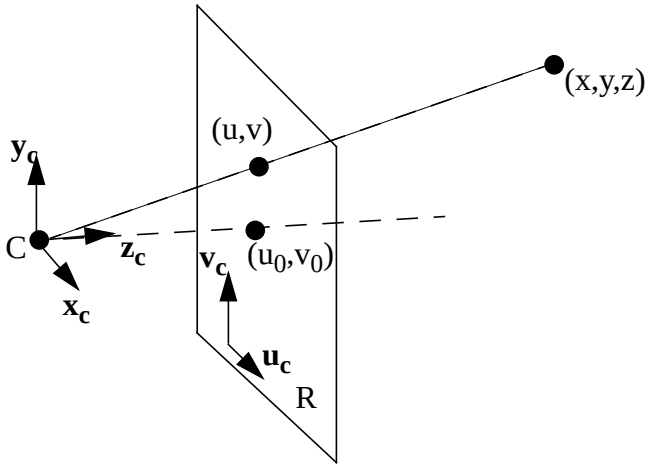

Figure 2: Placement of Coordinate Axes in Camera Projection Model

3.2 Planar Projection

The previous section discusses the projection of points in 3D space onto the 2D image plane. The 3D to 2D projection model is conveniently represented using a 3x4 projection matrix. In this section we consider the implications of our projection model for the projection of a planar pattern onto the image.

Recall that in our sensing problem, we recover camera position by observing just such a projection: a known reference pattern is printed on a planar surface, and then imaged using the camera. We can represent this planar projection process in homogeneous coordinates using a 3x3 matrix as described in the following paragraphs.

Following figure 3, we begin by associating a 3D coordinate system ($\mathbf{x_p}$,$\mathbf{y_p}$,$\mathbf{z_p}$) with the pattern so that the x and y axes lie in the plane of the pattern, and so that the z axis is perpendicular to the plane of the pattern. We construct a 2D coordinate system ($\mathbf{u_p}$,$\mathbf{v_p}$) with its u and v axes aligned with $\mathbf{x_p}$ and $\mathbf{y_p}$. We choose the units of the 3D coordinate system to be the same as the units of ($\mathbf{x_c}$,$\mathbf{y_c}$,$\mathbf{z_c}$). The units of the 2D coordinate system are permitted to differ from those of the 3D coordinate system by a scale factor, k_p. We will refer to coordinates expressed in these coordinate systems as 3D pattern coordinates and 2D pattern coordinates.

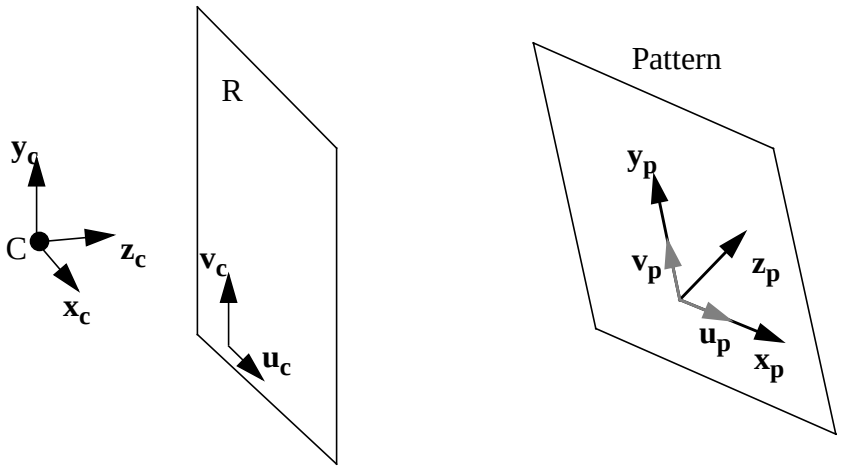

Figure 3: Camera and Pattern Coordinate Systems.

Since ($\mathbf{x_p}$,$\mathbf{y_p}$,$\mathbf{z_p}$) and ($\mathbf{u_p}$,$\mathbf{v_p}$) are aligned, points expressed in 2D pattern coordinates can be written in 3D pattern coordinates using the homogeneous equation.

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} \approx \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \tag{2}$$

The 3D coordinate frame associated with the pattern differs from the 3D camera coordinate system by a rigid body transformation, which can be expressed as a 3D rotation about the origin composed with a translation. We can imagine that the rotation aligns the directions of the coordinate axes in the two systems, and the translation moves the origins to be coincident. We can express this relationship using the projective equation.

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \approx {}_cT_p \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} \qquad (3)$$

$${}_cT_p = \begin{bmatrix} R & t \\ 0\ 0\ 0 & 1 \end{bmatrix}$$

where R is a 3x3 (non-homogeneous) rotation matrix, and t is the 3x1 vector of translations which aligns the origins of the two coordinate systems.

Substituting equations (2) and (3) into equation (1), we can write the projective equation which relates points in pattern coordinates to points in image coordinates.

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} \approx \begin{bmatrix} f_c k_u & 0 & u_0 & 0 \\ 0 & f_c k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}_cT_p \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \qquad (4)$$

or

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} \approx {}_cH_p \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \qquad {}_cH_p \approx \begin{bmatrix} f_c k_u & 0 & u_0 & 0 \\ 0 & f_c k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}_cT_p \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (5)$$

Where ${}_cH_p$ is a 3x3 matrix.

3.3 Planar Projection in Non-Homogeneous Coordinates

In section 4.1, it will be useful to express planar projection as a rational function in non-homogeneous coordinates. We will be specifically interested in the inverse of the projection of equation (5). This inverse exists for non-degenerate cases, so we can write

$$\begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \approx {}_cH_p^{-1} \begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} \tag{6}$$

Recall that our convention is to represent the image point $[u1, u2]^T$ using the homogeneous vector $[u_1, u_2, 1]^T$ and its scalar multiples, $[\alpha u_1, \alpha u_2, \alpha]^T$. By explicitly including the free parameter α, we can rewrite equation (6) as a strict equality

$$\begin{bmatrix} \alpha u_c \\ \alpha v_c \\ \alpha \end{bmatrix} = {}_cH_p^{-1} \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \tag{7}$$

Since linear transformations in homogeneous coordinates are themselves only defined up to a scale factor, we can scale the matrix ${}_cH_p^{-1}$ so that its lower right element is equal to 1. Consequently, we can write

$$\begin{bmatrix} \alpha u_c \\ \alpha v_c \\ \alpha \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix} \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \tag{8}$$

Solving for u_c and v_c, we can express the projection as a rational function of 8 parameters, as desired.

$$\begin{aligned} u_c &= \frac{h_1 u_p + h_2 v_p + h_3}{h_7 u_p + h_8 v_p + 1} \\ v_c &= \frac{h_4 u_p + h_5 v_p + h_6}{h_7 u_p + h_8 v_p + 1} \end{aligned} \tag{9}$$

3.4 Heads-up Display

In an augmented reality application, the purpose of a heads-up display is to superimpose graphics on the user's view of the workspace. In order to draw the graphics in the right places on the display, it is important to know which pixels of the display overlap with each 3D feature in the workspace. Consequently we need a projection model for the heads-up display as well as for the sensing camera.

We consider only a monocular display, and model it as a planar surface suspended in front of the user's eye. Turning on a pixel in the display illuminates a point on the suspended surface. The correspondence between points in the workspace and pixels in the display is determined by drawing an imaginary line from the workspace point to the center of the user's eye. The corresponding pixel is the one which lies at the intersection of the imaginary line and the plane of the display.

This geometric model corresponds exactly to the camera model of section 3.1. Following that discussion, we define a 2D display coordinate system ($\mathbf{u_d}$,$\mathbf{v_d}$), representing pixel locations in the display, and a 3D coordinate system ($\mathbf{x_d}$,$\mathbf{y_d}$,$\mathbf{z_d}$) centered at the user's eye. These correspond to ($\mathbf{u_c}$,$\mathbf{v_c}$) and ($\mathbf{x_c}$,$\mathbf{y_c}$,$\mathbf{z_c}$) in the camera model, and the units of ($\mathbf{x_d}$,$\mathbf{y_d}$,$\mathbf{z_d}$) are chosen to match those of ($\mathbf{x_c}$,$\mathbf{y_c}$,$\mathbf{z_c}$). Similarly, we define f_d, l_u and l_v corresponding to f_c, k_u and k_v, and (u_1,v_1) corresponding to (u_0,v_0). We write the projection model for the heads-up display in homogeneous coordinates.

$$\begin{bmatrix} u_d \\ v_d \\ 1 \end{bmatrix} \approx \begin{bmatrix} f_d l_u & 0 & u_1 & 0 \\ 0 & f_d l_v & v_1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_d \\ y_d \\ z_d \\ 1 \end{bmatrix} \tag{10}$$

By further defining a 3D coordinate ($\mathbf{x_w}$,$\mathbf{y_w}$,$\mathbf{z_w}$) system associated with the workspace, we can express the rigid body transformation between 3D work coordinates and 3D display coordinates as a 4x4 matrix ${}_{d}T_{w}$

$$\begin{bmatrix} x_d \\ y_d \\ z_d \\ 1 \end{bmatrix} \approx {}_dT_w \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{11}$$

And substituting equation (11) into equation (10) allows us to write the projection from 3D work coordinates to 2D display coordinates.

$$\begin{bmatrix} u_d \\ v_d \\ 1 \end{bmatrix} \approx \begin{bmatrix} f_d l_u & 0 & u_1 & 0 \\ 0 & f_d l_v & v_1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}_dT_w \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{12}$$

Equation (12) specifies the correspondence between 3D features in the workspace and pixels on the heads-up display.

4.0 Approach

In our system a user wears a heads-up display* which is used to overlay graphics on the user's field of view during the completion of a task. A reference pattern is printed on or attached to a stationary planar surface such as the ceiling or a nearby wall, and a camera is rigidly attached to a heads-up display in such a way that it images part of the pattern whenever the user's head is in position to complete the task. The position of the camera, and of the heads-up display, are recovered by observing the projection of the pattern onto the camera image. Schematically, the physical system can be viewed as shown in figure 4.

* At the time of publication, an optical see-through heads-up display was not available, so the system was implemented and tested using a video camera and digital image composition,

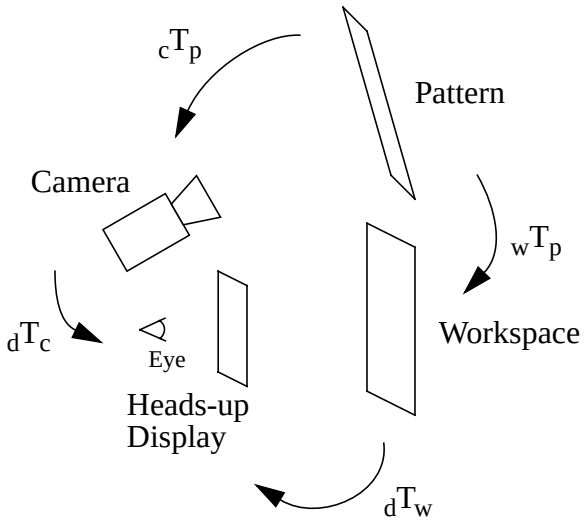

Figure 4: Augmented Reality System Schematic Diagram.

In the figure, ${}_{d}T_{c}$ represents a fixed rigid body transformation in 3 dimensional homogeneous coordinates which maps 3D camera coordinates to 3D display coordinates, and ${}_{w}T_{p}$ is a fixed rigid body transformation in 3 dimensional homogeneous coordinates which maps 3D pattern coordinates to 3D work coordinates (See section 3). Similarly, the transformation ${}_{d}T_{w}$ takes 3D work coordinates to 3D display coordinates, and ${}_{c}T_{p}$ takes 3D pattern coordinates to 3D camera coordinates. Note that ${}_{d}T_{w}$ and ${}_{c}T_{p}$ are not fixed, and depend on the position of the user's head.

In order to virtually mark a known point in the workspace, we need to know its position in 3D display coordinates, and we need to know the 4 intrinsic parameters of the heads-up display projection system. If these are known, we can use equation (10) to recover the corresponding point in 3D display coordinates, and draw the appropriate graphics on the heads-up display.

Since the position of the point in 3D workspace coordinates is known, its position in 3D display coordinates can be found using the transformation ${}_{d}T_{w}$. This transformation is dependent on the position of the user's head, and can be written in terms of ${}_{c}T_{p}$

$$ {}_{d}T_{w} \approx ({}_{d}T_{c})({}_{c}T_{p})({}_{w}T_{p})^{-1} \tag{13} $$

Using equation (13) and equation (12), we have

$$ \begin{bmatrix} u_d \\ v_d \\ 1 \end{bmatrix} \approx \begin{bmatrix} f_d l_u & 0 & u_1 & 0 \\ 0 & f_d l_v & v_1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} ({}_{d}T_{c})({}_{c}T_{p})({}_{w}T_{p})^{-1} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{14} $$

In section 4.1 we describe a technique for recovering the planar projection from 2D pattern coordinates to 2D camera coordinates, and in section 4.2 we show how to use this projection to recover ${}_{c}T_{p}$. In section 5 we describe a calibration procedure for recovering the fixed transformations ${}_{d}T_{c}$ and ${}_{w}T_{p}$, and the intrinsic parameters of the heads-up display projection system.

4.1 Image Registration

We represent an image as a function with a two dimensional, discrete valued, vector argument. The discrete values of the argument represent pixel positions within the image and correspond to points in (non-homogeneous) 2D camera coordinates. The function returns the value of the pixel at the specified position. We use a similar notation to represent the reference pattern in our augmented reality system. In this case the argument is a two dimensional continuous valued position in 2D pattern coordinates. In this section, we describe a technique for recovering the planar projection which maps 2D pattern coordinates to 2D camera coordinates. In other words, we recover the function m(u) so that the equation

$$ I_c[u] \;=\; P(m(u)) \qquad u \in R \tag{15} $$

is most nearly satisfied. The function $I_c[]$ represents an image from the sensing camera, and P() is the function which describes the planar reference pattern. The discrete valued 2 element vector u ranges over R, which is the set of valid pixel indices for $I_c[]$.

Following equation (9), we parameterize the planar projection m() by an 8 element vector h

$$m(u, h) = \begin{bmatrix} \dfrac{h_1 u_1 + h_2 u_2 + h_3}{h_7 u_1 + h_8 u_2 + 1} \\ \dfrac{h_4 u_1 + h_5 u_2 + h_6}{h_7 u_1 + h_8 u_2 + 1} \end{bmatrix} \tag{16}$$

Our method for recovering this projective mapping is based on an iterative approach described in [Szeliski, 1994]. This approach involves defining an error function over the space of transformation parameters, and using a nonlinear optimization routine to find a transformation which minimizes this error function.

The error function is defined by computing, for a given set of transformation parameters, a predicted camera image. The error is the sum of squared pixel differences between the predicted and observed images.

$$e_r(h) = \sum_{u \in R} \left(I_c[u] - P(m(u, h))\right)^2 \tag{17}$$

The derivatives of this function with respect to the elements of h can be easily computed.

$$\frac{\partial e_r}{\partial h_i} = (-2) \sum_{u \in C} \left(I_c[u] - P(m(u, h))\right) \left[\frac{d}{dr} P(r)\right]\Bigg|_{r = m(u, h)} \frac{\partial m}{\partial h_i}(u, h) \tag{18}$$

Where the 2 element row vector dP(r)/dr is simply the spacial gradient of the reference pattern, and $(\delta m/\delta h_i)(u,h)$ is a 2 element column vector found by differentiating the right hand side of equation (16).

As is conventional in nonlinear least squares optimization problems, we make the approximation

$$\frac{\partial^2 e_r}{\partial h_i \partial h_j} = 2 \sum_{u \in C} \left(\frac{\partial}{\partial h_i} P(m(u, h))\right)\left(\frac{\partial}{\partial h_j} P(m(u, h))\right) \tag{19}$$

This approximation is justified under the assumption of uncorrelated zero mean additive noise in the observed image [Press et al., 1988]. We use these quantities to minimize e_r over

h using the Levenberg-Marquardt nonlinear least squares minimization technique. For a description of Levenberg-Marquardt, see [Press et al., 1988]. After the minimization procedure, we find the planar projective transformations ${}_cH_p$ and ${}_pH_c$ according to

$$ {}_pH_c = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix} \qquad {}_cH_p = {}_pH_c^{-1} \tag{20} $$

As in [Szeliski, 1994], $I_c[]$ is repeatedly low pass filtered and subsampled before registration to create an image pyramid. With each subsampling, the number of rows and the number of columns in the image are decreased by a factor of two. The image registration procedure is first performed on the smallest image in the pyramid, and then the recovered parameters are refined by iterating on successively higher resolution images. The use of an image pyramid has two significant advantages. The first advantage is that computation time is greatly reduced; initial registration is performed on very small images, so the summations in equations (17) and (18) can be performed very quickly. The second advantage is that low pass filtering increases the robustness of the initial registration. High frequency components of the image have short wavelengths, and contribute local detail to the error function. While this local detail is important for the final registration, it can be detrimental during initial registration if the initial estimate of the transformation parameters is bad, since it can lead to local minima in the error surface, and incorrect convergence of the minimization procedure.

In practice, the registration technique presented above takes much too long to execute. Fine registration, which must be done at full resolution in order to achieve maximum precision, involves repeatedly summing over the entire image, and is still prohibitively expensive. Therefore we modify the registration procedure as described below.

Fine registration of the images can be sped up by exploiting knowledge about the reference pattern. When the images are nearly aligned, regions of the pattern which have zero spacial gradient make no contribution to the sum in equation (18), and provided the change in transformation parameters introduced by the fine registration is sufficiently small, summation

over the interior of these regions in equation (17) adds only a constant offset to the value of the error function. This suggests that fine registration can be speeded up if we assume that the images are already approximately aligned by previous pyramid levels, and restrict the region R in equations (17) and (18) so that it includes only pixels in the neighborhood of edges, corners and other features with nonzero spatial gradient. Our current system considers only a 4x4 pixel window around each corner feature in the reference pattern. For a description of the reference pattern, see section 6.

Any gradient based optimization technique is vulnerable to local minima in the error function. The registration technique of this section is prone to spectacular failure when the initial estimate of the camera position is very different from the actual position.

This type of failure is generally easy to detect, since it results in a large residual error e_r after the registration procedure. When incorrect convergence is suspected, the registration procedure is restarted from a new starting point. In the current system this starting point is chosen randomly from a set of 4 different candidates which were selected a priori to be typical of 4 different regions of the parameter space. The algorithm is allowed to converge, and the new solution is checked for correctness. If the solution is still suspect, a new initialization point is chosen, and the cycle repeats until an acceptable minimum is reached.

4.2 Recovering Heads-up Display Position

Once the 8 parameter projective transformations from the previous sections are known, the 3 dimensional homogeneous transformation ${}_cT_p$, which describes the position and orientation of the 3D camera coordinate frame with respect to the 3D pattern coordinate frame (see section 4.0), can be found. We have from equation (5) that

$$ {}_cH_p \approx \begin{bmatrix} f_c k_u & 0 & u_0 & 0 \\ 0 & f_c k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}_cT_p \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{21} $$

Substituting from equation (3) we can write

$$ {}_cH_p \approx \begin{bmatrix} f_c k_u & 0 & u_0 & 0 \\ 0 & f_c k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0\ 0\ 0 & 1 \end{bmatrix} \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{22} $$

where R is a 3x3 (non-homogeneous) rotation matrix, and t is the 3x1 vector of translations. Eliminating null rows and columns gives

$$ {}_cH_p \approx \begin{bmatrix} f_c k_u & 0 & u_0 \\ 0 & f_c k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_1 & r_2 & t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{23} $$

where r_1 and r_2 are the first and second columns of the rotation component of ${}_cT_p$. This equation can be solved linearly up to a scale factor, and then the scale factor chosen which most nearly makes r_1 and r_2 be of unit magnitude.

$$ \begin{bmatrix} r_1 & r_2 & t \\ 0 & 0 & 1 \end{bmatrix} = k \begin{bmatrix} f_c k_u & 0 & u_0 \\ 0 & f_c k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} {}_cH_p \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \tag{24} $$

The vector r_3 is chosen to be of unit length, and orthogonal to r_1 and r_2. At present, no attempt is made to enforce orthogonality between r_1 and r_2. We take ${}_cT_p$ to be

$$ {}_cT_p = \begin{bmatrix} r_1 & r_2 & r_3 & t \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{25} $$

4.3 Image Overlay

Once the pattern to camera transformation is known, the positions of features in the work-space can be projected into homogeneous 2D display coordinates as shown in equations (10) and (12). In order to escape the projective notation, we rewrite equation (12) as a strict equality with an explicit scale factor.

$$\begin{bmatrix} \alpha u_d \\ \alpha v_d \\ \alpha \end{bmatrix} = \begin{bmatrix} f_d l_u & 0 & u_1 & 0 \\ 0 & f_d l_v & v_1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}_d T_w \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{26}$$

This equation can be solved for u_d and v_d,

$$\cdot \begin{bmatrix} u_d \\ v_d \end{bmatrix} = \begin{bmatrix} \left(\begin{bmatrix} f_d l_u & 0 & u_{0e} & 0 \end{bmatrix} {}_d T_w \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \right) \Big/ \left(\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} {}_e T_w \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \right) \\ \left(\begin{bmatrix} 0 & f_d l_u & u_{0e} & 0 \end{bmatrix} {}_d T_w \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \right) \Big/ \left(\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} {}_e T_w \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \right) \end{bmatrix} \tag{27}$$

and appropriate marks can be made on the heads-up display at these locations.

5.0 Calibration

In order to perform the image overlay described above, we need to have an estimate of several static system parameters. Recovering these parameters is difficult because, with an optical see-through display, there is no direct way of measuring the accuracy the image overlay. A synthesized mark is aligned well if it appears in the “right” place in the user’s field of view, and the only way to know if this is true is to query the user. Consequently, any calibration procedure necessarily involves user feedback. We choose to get user feedback by having the user align marks on the display with known landmarks in the workspace. This section describes the calibration procedure.

There are sixteen system parameters to be calibrated: six parameters describe the transformation between 3D camera coordinates and 3D display coordinates, six parameters describe

the transformation between 3D pattern coordinates and 3D work coordinates, and four parameters describe the heads-up display/eye projection matrix. The four intrinsic parameters of the sensing camera are assumed to be known (for a method of calibrating these four parameters using planar projection, see Appendix A). Initially, all sixteen parameters must be calibrated, but since the camera-to-display transformation is fixed, and the pattern-to-workspace transformation may be also, subsequent calibrations may involve as few as four parameters. These recalibrations are necessary whenever the user removes and replaces the heads-up display, changing the four parameters of the eye/display projection model.

To understand how the 4x4 matrices ${}_cT_p$ and ${}_dT_w$ are fully described by 6 parameters, we consider in more detail the relationship between the 3D camera coordinate frame and the 3D pattern coordinate frame. The camera frame has arbitrary position and orientation with respect to the reference pattern. By inspection, we see that the 3D translation which aligns the origins of the two coordinate systems can be written in homogeneous coordinates

$$T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{28}$$

We represent the 3D rotation which aligns the axes of the two coordinate frames as the composition of 3 consecutive rotations θ_x, θ_y and θ_z around the x, y, and z axes. The matrices corresponding to these simple rotations can again be determined by inspection.

$$\begin{aligned} R &= \begin{bmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_y) & 0 & \cos(\theta_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) & 0 \\ 0 & \sin(\theta_x) & \cos(\theta_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos(\theta_y)\cos(\theta_z) & \sin(\theta_x)\sin(\theta_y)\cos(\theta_z) - \cos(\theta_x)\sin(\theta_z) & \cos(\theta_x)\sin(\theta_y)\cos(\theta_z) + \sin(\theta_x)\sin(\theta_z) & 0 \\ \cos(\theta_y)\sin(\theta_z) & \sin(\theta_x)\sin(\theta_y)\sin(\theta_z) + \cos(\theta_x)\cos(\theta_z) & \cos(\theta_x)\sin(\theta_y)\sin(\theta_z) - \sin(\theta_x)\cos(\theta_z) & 0 \\ -\sin(\theta_y) & \sin(\theta_x)\cos(\theta_y) & \cos(\theta_x)\cos(\theta_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{29}$$

Composing these two transformations gives the following matrix

$$M(\theta) = TR$$

$$= \begin{bmatrix} \cos(\theta_y)\cos(\theta_z) & \sin(\theta_x)\sin(\theta_y)\cos(\theta_z) - \cos(\theta_x)\sin(\theta_z) & \cos(\theta_x)\sin(\theta_y)\cos(\theta_z) + \sin(\theta_x)\sin(\theta_z) & t_x \\ \cos(\theta_y)\sin(\theta_z) & \sin(\theta_x)\sin(\theta_y)\sin(\theta_z) + \cos(\theta_x)\cos(\theta_z) & \cos(\theta_x)\sin(\theta_y)\sin(\theta_z) - \sin(\theta_x)\cos(\theta_z) & t_y \\ -\sin(\theta_y) & \sin(\theta_x)\cos(\theta_y) & \cos(\theta_x)\cos(\theta_y) & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (30)$$

where we define the parameter vector $\theta = [\theta_x, \theta_y, \theta_z, t_x, t_y, t_z]$.

For the purposes of calibration, we define a 16 element parameter vector λ. We parameterize ${}_cT_p$ by its first six elements and ${}_dT_w$ by its second six elements. The remaining four elements represent the four unknown parameters of the eye/heads-up display projection matrix.

We estimate the 16 unknown parameters using an iterative nonlinear optimization. We begin the initial calibration process by hand-coding “ballpark” estimates for the sixteen parameters, and running the system to place a synthetic mark on top of a known landmark in the workspace. The initial estimates may not be very accurate, and consequently the synthesized mark will generally appear in the wrong place. The user indicates this by pressing a button, which freezes the display and records the display coordinates of the synthesized mark. After pressing the button, the user moves his/her head until the mark and feature are aligned, and presses the button again. The position of the camera with respect to the pattern is tracked during the motion, and is recorded at the time of the second button press. This procedure is repeated several times at varying positions and with several landmarks. Each screen coordinate-camera position pair provides an equation in the form of equation (27),

$$u_i = \begin{bmatrix} \left(\begin{bmatrix} f_d l_u & 0 & u_{0e} & 0 \end{bmatrix} {}_dT_w x_i \right) / \left(\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} {}_eT_w x_i \right) \\ \left(\begin{bmatrix} 0 & f_d l_u & u_{0e} & 0 \end{bmatrix} {}_dT_w x_i \right) / \left(\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} {}_eT_w x_i \right) \end{bmatrix} \quad (31)$$

where u_i is a 2 element column vector, the position of the synthesized mark on the heads-up display, and x is a four element column vector, the position of the landmark in (homogeneous) 3D workspace coordinates.

Once a sufficiently large set of point correspondences has been generated, at least 8 for the sixteen parameter calibration, a squared error function is defined.

$$e_c(\lambda) = \sum_{i=1}^{N} \left\| \left(u_i - \begin{bmatrix} \left(\begin{bmatrix} \lambda_{13} & 0 & \lambda_{15} & 0 \end{bmatrix} [{}^{i}_{d}T_w(\lambda)] x_i \right) / \left(\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} [{}^{i}_{d}T_w(\lambda)] x_w \right) \\ \left(\begin{bmatrix} 0 & \lambda_{14} & \lambda_{16} & 0 \end{bmatrix} [{}^{i}_{d}T_w(\lambda)] x_i \right) / \left(\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} [{}^{i}_{d}T_w(\lambda)] x_w \right) \end{bmatrix} \right) \right\|^2 \quad (32)$$

Where we define

$${}^{i}_{d}T_w(\lambda) = {}_{d}T_c(\lambda)[{}^{i}_{c}T_p]_w T_p(\lambda)^{-1} \quad (33)$$

Where ${}^{i}_{c}T_p$ is the i^{th} pattern to camera transformation recorded by the user during the manual alignment, and corresponds to u_i, the i^{th} synthetic mark position.

The first and second derivatives of this function are messy, but calculable, and a local minimum can be found using the Levenberg-Marquardt method.

This method of estimating λ may fail to converge to the correct result for three distinct reasons: the set of specified points may not be sufficient to completely determine the parameter set; the minimization may converge to an incorrect (local) minimum; and alignment errors during the calibration process may lead to the recovery of incorrect parameters.

If the procedure converges to an incorrect minimum because the calibration parameters are underdeterimined by the specified points, or because of a local minimum in the error space, the user can select an image-feature pair which exhibits especially bad registration and repeat the calibration process with this pair added to the calibration set. The misregistered pair will increase the value of the error function at the local minimum, or add additional constraints to the parameter set.

Alignment errors during the calibration process are a more difficult problem. If the user fails to accurately line up the synthesized and real features, or if the tracking procedure of

section 4.1 returns a bad estimate of the transformation ${}_cT_p$, an incorrect term will be added to the error function. We currently make no attempt to detect these errors.

6.0 Implementation

All code was written in C++ to run on a Silicon Graphics Indy R5000 Workstation. Since an optical see-through head mounted display was not available, the heads-up display/eye combination was modeled using a Panasonic VPC-920 CCD camera, and the real and synthetic images were composed digitally as is done when a video based heads-up display is used. The SGI Indycam (supplied with the workstation) was used as the tracking sensor, and was rigidly attached to the display camera using a custom aluminum bracket. Digitization hardware was configured to return 320x240 24 bit color images.

The reference pattern was chosen to look roughly like a pattern of small ceiling tiles. Specifically, the pattern consists of black and white squares arranged in a 7x7 checkerboard pattern (Figure 5). The pattern is 34”x34”, and the squares are roughly 5 inches on a side.

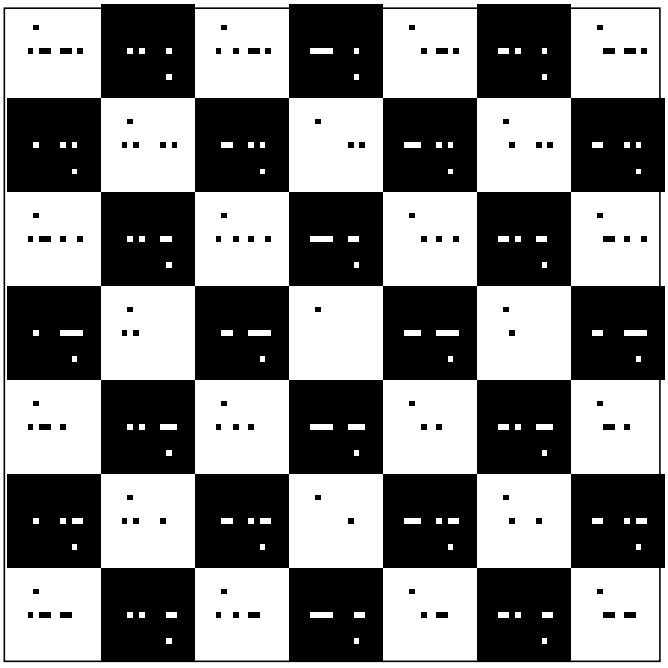

Figure 5: Reference Pattern.

Since a plain checkerboard is a symmetrical repeating pattern, there is a phase ambiguity in the interpretation of the camera images. Images shifted in x or y by an even number of blocks look identical, as do images in which the pattern is rotated around the center of a square by an angle which is an integer multiple of 90 degrees. We addressed this ambiguity by printing a unique identification code on each square. This identification encodes the position of the square within the pattern, as well as the orientation of the pattern. The code consists of two 4 bit binary words and two 1 bit orientation indicators (see figure 6). By reading this code after registering the image, it is possible to eliminate the ambiguity in the position and orientation of sensing camera relative to the reference pattern.

.

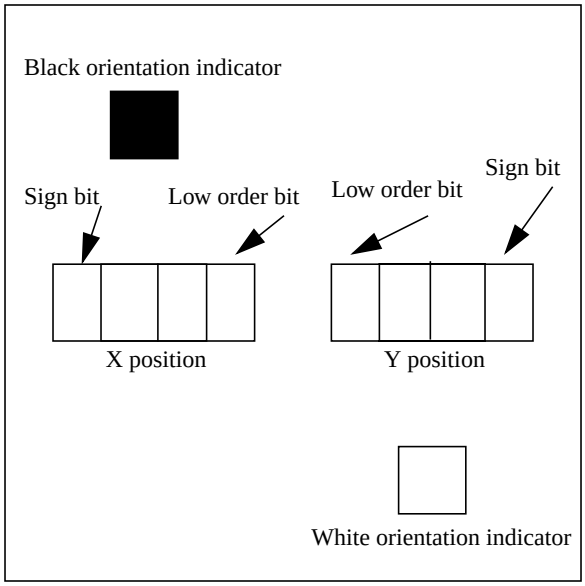

Figure 6: Organization of identification code on reference pattern squares.

The reference pattern was affixed to a wall near the author's desk. Two workspace coordinate systems were defined and used in two separate runs of the system. The first workspace coordinate system was attached to the wall, with its x and y axes in the plane of the pattern. The second workspace coordinate system was attached to a nearby desk (see figure 7). Workspace features with known coordinates were determined in each of the two coordinate

systems. All of the features in the first workspace coordinate system (the one attached to the wall) were coplanar with the pattern, and all of the features in the second workspace coordinate system lay in the plane of the desktop.

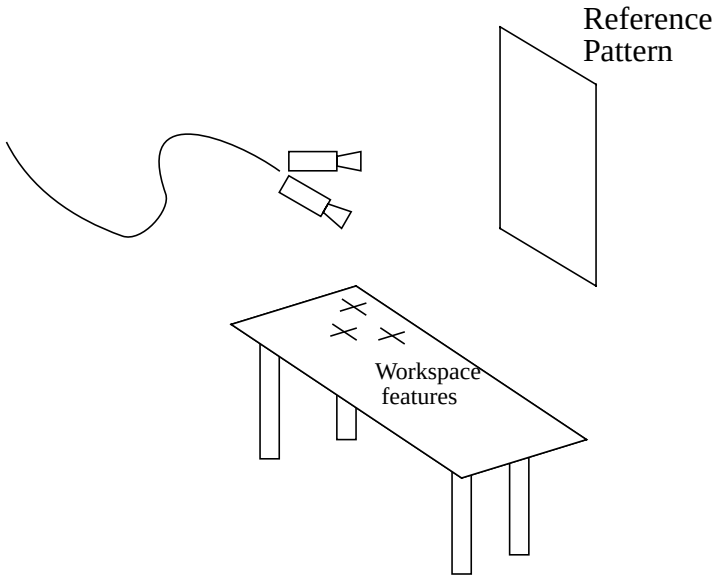

Figure 7: Experimental setup

7.0 Results and Discussion

7.1 Results

The setup described in section 6 was timed to determine system speed. Registration takes about 50 ms, although the time required depends on the amount of motion between frames and the distance from the pattern. For small motions, times as short as 25 ms are common. Larger motions can take as much as 80ms. Actual system speed was a little slower because of the time required to capture and display video from the display camera. If an optical see-through heads-up display were used, this additional delay would not be an issue.

Overlay accuracy was measured by locating workspace features manually in the display camera image. The distance between the actual feature and the synthesized mark was mea-

sured in pixels for each of three workspace features, with the sensing camera/display camera assembly at six different positions and orientations. The camera assembly was held still as each image was acquired.

The mean error for features in the plane of the reference pattern was 3.4 pixels, which projected to a misregistration of about 0.4 inches in workspace coordinates when the camera was held at a range of 3 feet. The mean error for features in the plane of the desk surface was higher, at 7.4 pixels, which projected to a misregistration of about an inch in workspace coordinates at a range of 3 feet, and about 0.5 inches when the distance between the camera assembly and desk surface was 18 inches (about arms length). Error for the desk features also tended to vary more, with a standard deviation of 4.8 pixels, compared to 2.2 pixels for the wall features.

7.2 Discussion

The system speed reported above is fast enough that the display moves fairly smoothly. However, it is sufficiently slow that the synthesized marks lag perceptibly behind the actual image features when the camera assembly is in motion. This lag is distracting, and detracts from the illusion that the synthesized marks are actually part of the workspace.

We propose to address this dynamic error by predicting sensor position and by speeding up the registration process of section 4.1. There is evidence that predictive filtering of sensor position can reduce dynamic errors by as much as 50% [Azuma and Bishop, 1994], and we anticipate that by exploiting the binary nature of the reference pattern, the registration step can be speeded up considerably.

The static errors reported above are consistent with those of other sensing technologies currently in use [Azuma, 1995], but we believe that they could be significantly reduced by compensating for radial lens distortion in the sensing camera. The SGI Indycam has inexpensive imaging optics, and deviates significantly from the pinhole camera model of section 3.1. If this optical distortion were characterized and mapped, the camera image could be resampled using a lookup table without incurring significant computational cost.

Appendix A: Calibration of Camera Intrinsic Parameters

In equation (1), we described the projection model for the sensing camera using a three by four matrix in homogeneous coordinates, called the projection matrix of the camera

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} \approx \begin{bmatrix} f_c k_u & 0 & u_0 & 0 \\ 0 & f_c k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{34}$$

As before, we use the congruence operator to represent projective equality (equality up to a scale factor). This projection matrix is dependent upon four quantities, $[f_c k_u, f_c k_v, u_0, v_0]$, the intrinsic parameters of the camera. In the body of this paper, we have assumed these parameters to be known.

If the intrinsic parameters of the sensing camera are not known, they can be estimated as follows. Recall from equation (5) that the planar projection from 2D pattern coordinates to 2D image coordinates can be written as a three by three matrix in homogeneous coordinates.

$$\begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix} \approx {}_c H_p \begin{bmatrix} u_p \\ v_p \\ 1 \end{bmatrix} \tag{35}$$

Recall also that this three by three matrix can be expressed as the product of three matrices in homogeneous coordinates.

$${}_c H_p \approx \begin{bmatrix} f_c k_u & 0 & u_0 & 0 \\ 0 & f_c k_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0\ 0\ 0 & 1 \end{bmatrix} \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{36}$$

where R is a three by three rotation matrix in non-homogeneous coordinates, and t is a three element column vector. Eliminating null rows from equation (28), we have

$$H \approx \begin{bmatrix} f_c k_u & 0 & u_0 \\ 0 & f_c k_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{1,1} & r_{1,2} & t_1 \\ r_{2,1} & r_{2,1} & t_2 \\ r_{3,1} & r_{3,2} & t_3 \end{bmatrix} \begin{bmatrix} p & 0 & 0 \\ 0 & p & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{37}$$

where we define $r_{i,j}$ to be the $(i,j)^{th}$ element of R, and t_i to be the i^{th} element of the column vector t. The two outside matrices on the right hand side of equation (37) are trivially invertible, so we can write

$$\begin{bmatrix} 1/(f_c k_u) & 0 & (-u_0)/(f_c k_u) \\ 0 & 1/(f_c k_v) & (-v_0)/(f_c k_v) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} \\ h_{2,1} & h_{2,2} & h_{2,3} \\ h_{3,1} & h_{3,2} & h_{3,3} \end{bmatrix} \begin{bmatrix} 1/p & 0 & 0 \\ 0 & 1/p & 0 \\ 0 & 0 & 1 \end{bmatrix} \approx \begin{bmatrix} r_{1,1} & r_{1,2} & t_1 \\ r_{2,1} & r_{2,1} & t_2 \\ r_{3,1} & r_{3,2} & t_3 \end{bmatrix} \tag{38}$$

or equivalently,

$$\begin{bmatrix} \frac{1}{f_c k_u}(h_{1,1} - u_0 h_{3,1}) & \frac{1}{f_c k_u}(h_{1,2} - u_0 h_{3,2}) & \frac{1}{f_c k_u}(h_{1,3} - u_0)p \\ \frac{1}{f_c k_v}(h_{2,1} - v_0 h_{3,1}) & \frac{1}{f_c k_v}(h_{2,2} - v_0 h_{3,2}) & \frac{1}{f_c k_v}(h_{2,3} - v_0)p \\ h_{3,1} & h_{3,2} & p \end{bmatrix} \approx \begin{bmatrix} r_{1,1} & r_{1,2} & t_1 \\ r_{2,1} & r_{2,1} & t_2 \\ r_{3,1} & r_{3,2} & t_3 \end{bmatrix} \tag{39}$$

where we have multiplied the left hand side by a constant factor of p without affecting the projective equality of the two matrices. Since the first two columns of the right side of this equation are columns of a rotation matrix, we know that they are orthogonal and of equal length. Under projective equality, these properties hold for the left hand side of the equation.

Since the first two columns of the left side of equation (39) are orthogonal, their dot product must be equal to zero. This gives

$$\left(\frac{1}{f_c k_u}\right)^2 (h_{1,1} - u_0 h_{3,1})(h_{1,2} - u_0 h_{3,2}) + \left(\frac{1}{f_c k_v}\right)^2 (h_{2,1} - v_0 h_{3,1})(h_{2,2} - v_0 h_{3,2}) + h_{3,1} h_{3,2} = 0 \tag{40}$$

Since the first two columns of the left side of equation (39) are of equal length, we can write that the difference of their magnitudes is equal to zero

$$\left(\frac{1}{f_c k_u}\right)^2 ((h_{1,1} - u_0 h_{3,1})^2 - (h_{1,2} - u_0 h_{3,2})^2) + \left(\frac{1}{f_c k_v}\right)^2 ((h_{2,1} - v_0 h_{3,1})^2 - (h_{2,2} - v_0 h_{3,2})^2) + h_{3,1}^2 - h_{3,2}^2 = 0 \tag{41}$$

We define a parameter vector $\theta = [\theta_1\ \theta_2\ \theta_3\ \theta_4]^T$, and parameterize the left sides of equations (40) and (41) by θ and H

$$g_1(\theta, H) = \left(\frac{1}{\theta_1}\right)^2 (h_{1,1} - \theta_3 h_{3,1})(h_{1,2} - \theta_3 h_{3,2}) + \left(\frac{1}{\theta_2}\right)^2 (h_{2,1} - \theta_4 h_{3,1})(h_{2,2} - \theta_4 h_{3,2}) + h_{3,1} h_{3,2} \tag{42}$$

$$g_2(\theta, H) = \left(\frac{1}{\theta_1}\right)^2 ((h_{1,1} - \theta_3 h_{3,1})^2 - (h_{1,2} - \theta_3 h_{3,2})^2) + \left(\frac{1}{\theta_2}\right)^2 ((h_{2,1} - \theta_4 h_{3,1})^2 - (h_{2,2} - \theta_4 h_{3,2})^2) + h_{3,1}^2 - h_{3,2}^2 \tag{43}$$

Note that each possible position and orientation of the camera defines a projection from pattern coordinates to camera coordinates, and a corresponding planar projection matrix H. For each H, $g_1(\theta,H)$ and $g_2(\theta,H)$ will be equal to zero when $\theta_1 = f_c k_u$, $\theta_2 = f_c k_v$, $\theta_3 = u_0$, and $\theta_4 = v_0$.

Our calibration procedure, is to collect a large number of different planar projection matrices (corresponding to widely varied camera positions) using the procedure described in section 4.1. We then define the error function

$$e_c(\theta) = \sum_{i=1}^{N} (g_1(H_i, \theta)^2 + g_2(H_i, \theta)^2) \tag{44}$$

where the H_i are the individual planar projection matrices, and a total of N have been collected. We minimize this function over θ using an iterative nonlinear least squares algorithm,

$$\hat{\theta} = \arg min(e_c(\theta)) \tag{45}$$

and estimate the camera intrinsic parameters based on this result

$$\begin{aligned} f_c k_u &= \hat{\theta}_1 \\ f_c k_v &= \hat{\theta}_2 \\ u_0 &= \hat{\theta}_3 \\ v_0 &= \hat{\theta}4 \end{aligned} \tag{46}$$

Bibliography

Azuma, R. and Bishop, G. (1994). Improving static and dynamic registration in an optical see-through hmd. In *Proceedings of ACM SIGGRAPH '94*, pages 197–204, Orlando, FL.

Azuma, R. and Ward, M. (1991). Space resection by collinearity: Mathematics behind the optical ceiling head-tracker. Technical Report TR91-048, The University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC.

Azuma, R. T. (1995). A survey of augmented reality. In *Proceedings of ACM SIGGRAPH '95*, Los Angeles, CA.

Bergen, J. R., Anandan, P., Hanna, K. J., and Hingorani, R. (1992). Heirarchical model-based motion estimation. In *Proceedings of the Second European Conference on Computer Vision*, Santa Margherita Ligure, Italy.

Caudell, T. P. and Mizell, D. W. (1992). Augmented reality: An application of heads-up display technology to manual manufacturing processes. In *Proceedings of the 25th Hawaii International Conference on Systems Sciences*, pages 7–10, Kauai, HI.

Faugeras, O. (1993). *Three-Dimensional Computer Vision: a Geometric Viewpoint*. The MIT Press, Cambridge, Massachusetts.

Foxlin, E. and Durlach, N. (1994). An inertial head-orientation tracker with automatic drift compensation for use with hmd's. In Singh, G., Feiner, S. K., and D.Thalmann, editors, *Proceedings of VRST94: Virtual Reality Software and Technology*, pages 159–173, Singapore.

Irani, M., Anandan, P., and Hsu, S. (1995). Mosaic based representations of video sequences and their applications. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 605–611, Cambridge, MA. IEEE Computer Society Press.

Irani, M., Rousso, B., and Peleg, S. (1994). Computing occluding and transparent motions. *The International Journal of Computer Vision*, 12(1):5–16.

Janin, A. L., Mizell, D. W., and Caudell, T. P. (1993). Calibration of head-mounted displays for augmented reality applications. In *Proceedings of IEEE Virtual Reality Annual Intrinsic Symposium*, pages 246–255, New York, NY.

Meyer, K., Applewhite, H. L., and Biocca, F. A. (1992). A survey of position trackers. *Presence*, 1(2):173–200.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1988). *Numerical Recipes in C: the Art of Scientific Computing*. Cambridge University Press, Cambridge, UK.

Szeliski, R. (1994). Image mosaicing for tele-reality applications. Technical Report CRL 94/2,

Digital Equipment Corporation Cambridge Research Lab, Cambridge, MA.

Szeliski, R. and Coughlan, J. (1994). Spline based image registration. Technical Report CRL 94/1, Digital Equipment Corporation Cambridge Research Lab, Cambridge, MA.

Taylor, R., Funda, J., LaRose, D., and Treat, M. (1992). A telerobotic system for augmentation of endoscopic surgery. In *IEEE EMBS*, Paris, France.

Taylor, R., Gruben, K., LaRose, D., and Gomory, S. (1994). Image guided command and control of a surgical robot. In *Proceedings of Medicine Meets Virtual Reality II*, San Diego, CA.

Zikan, K., Curtis, W. D., Sowizral, H. A., and Janin, A. L. (1994a). Fusion of absolute and incremental position and orientation sensors. In *SPIE Proceedings volume 2351: Telemanipulator and Telepresence Technologies*, pages 316–327, Boston, MA. SPIE.

Zikan, K., Curtis, W. D., Sowizral, H. A., and Janin, A. L. (1994b). A note on dynamics of human head motions and on predictive filtering of head-set orientations. In *SPIE Proceedings volume 2351: Telemanipulator and Telepresence Technologies*, pages 328–336, Boston, MA. SPIE.