Registration Error Analysis for Augmented Reality

Richard L. Holloway[†]

Department of Computer Science
University of North Carolina
Chapel Hill, NC

Abstract

Augmented reality (AR) systems typically use see-through head-mounted displays (STHMDs) to superimpose images of computer-generated objects onto the user's view of the real environment in order to augment it with additional information. The main failing of current AR systems is that the virtual objects displayed in the STHMD appear in the wrong position relative to the real environment. This *registration error* has many causes: system delay, tracker error, calibration error, optical distortion, and misalignment of the model, to name only a few. Although some work has been done in the area of system calibration and error correction, very little work has been done on characterizing the nature and sensitivity of the errors that cause misregistration in AR systems.

This paper presents the main results of an end-to-end error analysis of an optical STHMD-based tool for surgery planning. The analysis was done with a mathematical model of the system and the main results were checked by taking measurements on a real system under controlled circumstances. The model makes it possible to analyze the sensitivity of the system registration error to errors in each part of the system. The major results of the analysis are (1) even for moderate head velocities, system delay causes more registration error than all other sources combined, (2) eye tracking is probably not necessary, (3) tracker error is a significant problem both in head tracking *and* in system calibration, (4) the World (or reference) coordinate system adds error and should be omitted when possible, and (5) computational correction of optical distortion may introduce more delay-induced registration error than the distortion error it corrects, and (6) there are many small error sources which will make sub-millimeter registration almost impossible in an optical STHMD system without feedback. Although this model was developed for optical STHMDs for surgical planning, many of the results apply to other HMDs as well.

1. Introduction

See-through head-mounted displays (STHMDs)[1] combine three-dimensional computer-generated imagery with the view of the real environment in order to augment the view of the real world with additional information. The promise of these systems is that allowing users to view the data *in situ* will increase their understanding and improve their interactions with it. A critical problem, however, is that the computer-generated objects do not currently remain correctly aligned, or *registered*, with the real environment—objects aligned from one viewpoint appear misaligned from

---

[1] The model presented here is for *optical* STHMDs, which use beam-splitters to optically merge the real and virtual views. *Video* STHMDs, such as the one described in [Bajura et al. 92], acquire the real-world view via miniature video cameras and merge the real and synthetic views electronically.

another viewpoint and appear to swim about as the user moves her head. This is clearly unacceptable to a user seeking understanding of the relationship between the virtual and real objects, since this registration error causes the relationship to vary as the viewpoint changes. This swimming effect is also a problem with the more common, opaque HMDs: even though the real-world reference is gone, it is painfully obvious with most systems that supposedly stationary virtual objects roam around as the user changes his viewpoint.

The reason that no one has solved the problem yet is that good registration demands accuracy and speed from nearly every component of the system and near-perfect system calibration. The number of error sources is large and the interactions and sensitivities of the system have not been explored in detail until now. While some progress has been made at correcting for some of the most egregious error sources (such as system delay), no previous work has completely enumerated the registration error sources and modeled their effect on the net registration error.

A mathematical error model for augmented reality systems enables the system architect to determine
1. what the registration error sources are and which ones are the most significant contributors to the total error,
2. the sensitivity of the net registration error to input errors in each part of the system,
3. the nature of the distortions caused by each type of input error, and
4. the level of registration accuracy one can expect as a function of the input errors,

and also provides insights on how to best calibrate the system.

In other words, the model tells the system architect where to spend his time and money in order to improve the system's registration, and also gives some idea of what level of registration he can expect for a given set of hardware and software.

The main results of the analysis conducted using the model are
1. Even for moderate head velocities, system delay causes more registration error than all other sources combined. A rule of thumb for medical applications is that each millisecond of delay introduces a millimeter of registration error in the worst case, and $\frac{1}{3}$ mm/s in the average case. The only hope for good dynamic registration with optical see-through systems will be to use predictive head tracking.
2. Eye tracking is probably not necessary, since error due to eye rotation can be minimized by using the eye's center of rotation as the center of projection.
3. Tracker error is a significant problem both in head tracking *and* in system calibration, even when the tracker is calibrated for field distortion and other static errors.
4. The World (or reference) coordinate system adds error and should be omitted when possible.
5. Computational correction of optical distortion may introduce more delay-induced registration error than the distortion error it corrects.
6. There are many small error sources which will make sub-millimeter registration almost impossible in an optical STHMD system.

The table below gives an approximate ranking of error sources with estimated error ranges and assumptions.

Rank	Error source	Registration error (mm)	Assumptions
1	Delay	20-60+	Max head velocities of 500 mm/s, 50°/s
2	Optical distortion	0-20 (in image plane)	11% distortion at corner of image, 4% at top of image, magnification = 6.0, image distance = 500 mm
3	World-Tracker calibration error	4-10+	Assumes head is ~500 mm from transmitter and viewed point is at arm's length (500 mm)
4	Tracker measurement error (static, dynamic, jitter)	1-7+	Assumes magnetic tracker w/ source-sensor distance ≤ 500 mm
5	Acquisition/ alignment error	1-3	Typical medical dataset w/ voxel sizes of 1x1x3 mm
6	Viewing error	0-2+	Virtual image at 500 mm, 5 mm of eye movement or calibration error, viewed point is ±200 mm from virtual image plane.
7	Display non-linearity	1-2	1" CRT with non-linearity ≈ 1%, magnification = 6.0
8	Image misalignment, lateral color, aliasing	< 1	Good calibration procedures Perceived image point is average of RGB images NTSC resolution

Table 1. Error sources and associated registration-error magnitudes

[Holloway 95] describes the model and the test system in full detail; the rest of this paper gives a brief overview of the model and summarizes the most important results.

2. Prior Work

While there are many descriptions of STHMD systems in the literature ([Sutherland 68], [Furness 86], [Bajura et al. 92], and [Feiner et al. 93]), only a few papers have dealt with errors or models for HMD systems. [Robinett & Rolland 91] presents a computational model for HMDs which identifies key parameters for characterizing a system, and [Grinberg et al. 94], [Robinett & Holloway 95] and [Southard 94] present models for correctly computing the complete viewing transformation once the system parameters are known. [Janin et al 93] and [Oishi & Tachi 96] describe methods for calibration of STHMDs and discuss a number of registration error sources and their effects. [Deering 92] describes a number of error sources encountered in creating a CRT-based AR tool. [Azuma & Bishop 94] gives a brief listing of error sources and presents methods for correcting some of the worst error sources via calibration procedures and predictive head tracking. [Hodges & Davis 93] discusses the geometry of stereoscopic viewing and lists a number of error sources and their effects, but stops short of a complete system analysis. [Min & Jense 94] also lists several error sources and describes a user study to determine the optimal system parameters for each subject. [State et al. 94] describes problems encountered in attempting to register ultrasound data displayed with a video STHMD with a real patient. [Bajura & Neumann 95] presents a model for video-based AR systems which dynamically corrects the registration error by forcing alignment of the real and virtual images.

3. Error Model

3.1. Registration Error Metrics

In the course of the analysis of registration error sources, it became clear that there are several metrics for registration error and that each is useful for describing some aspect of the problem. This section describes the metrics used and when they are useful.

If two points that are supposed to be coincident are separated by some distance, one can describe the degree of separation or misregistration with a 3D error vector from one point to the other. *Linear registration error* is defined here to be the length of this error vector. While this is often a useful metric for registration error, generating the 3D error vector for a stereoscopic display requires knowledge of the projectors from both eyes in order to specify the location of the perceived point, and there are many cases in which we would like to examine the registration error for a single eye. Moreover, in cases where the projectors are nearly parallel, even tiny errors can cause the projectors to become parallel, inducing theoretically infinite linear registration error. To characterize such a situation as having infinite registration error seems overly pessimistic and not very useful, since the projectors do pass near the point and may appear to converge at the point when coupled with other depth cues, such as head-motion parallax. All of this leads to the conclusion that registration errors in depth are somehow different from registration errors that cause a clear visual separation between the real and virtual points, particularly when a stereoscopic display is involved. For this reason, I will also describe the registration error using the three related metrics pictured below.
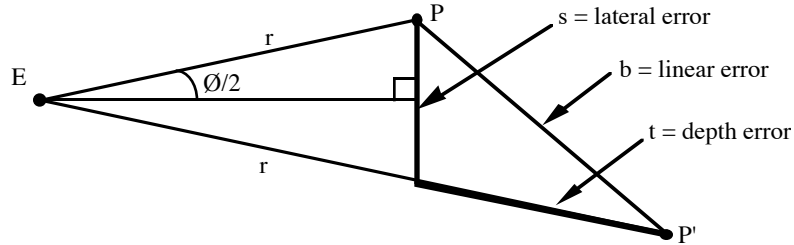
Figure 1. Registration error metrics

In the figure, E is the eyepoint, P' is the displayed point, and P is the real point (where we want P' to appear). The *angular error* Ø is the visual angle subtended at the eyepoint by the line segment PP'. The *lateral error s* answers the question, "If P' were at the same distance as P from the eye, how far apart would they be?" Mathematically, this is the length of a line from P to EP' which is perpendicular to the bisector of the angle PEP', and is given by

$$s \;=\; 2\,r\,\sin\tfrac{\emptyset}{2} \qquad\qquad (1)$$

Finally, the *depth error t* tells us how much closer or further P' is relative to P, and is given by

$$t \;=\; \|\mathbf{v}_{E\_P'}\| - \|\mathbf{v}_{E\_P}\| \qquad\qquad (2)$$

where $\|\mathbf{v}_{E\_P'}\|$ and $\|\mathbf{v}_{E\_P}\|$ are the magnitudes of the vectors from E to P' and P, respectively.

Clearly, all of these measures depend on the geometry defining the segment PP', so a complete specification will depend on the situation being discussed. I will use these metrics for discussing viewing and display errors, and the linear registration error for the analysis of head-tracking error and digitization/alignment error.
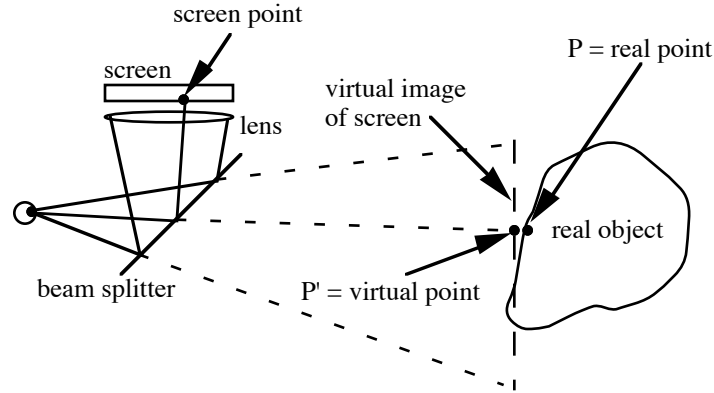
3.2.   System Operation

In a typical AR system, the virtual objects which are to be registered with their real counterparts are first acquired by some form of imaging equipment (*e.g.,* a CT scanner) or modeled with some design tool (*e.g.,* a computer-aided design package). This typically produces a virtual object defined in its own coordinate system (CS), which must then be aligned with the real object(s) in the laboratory or World CS. The next figure shows this process.
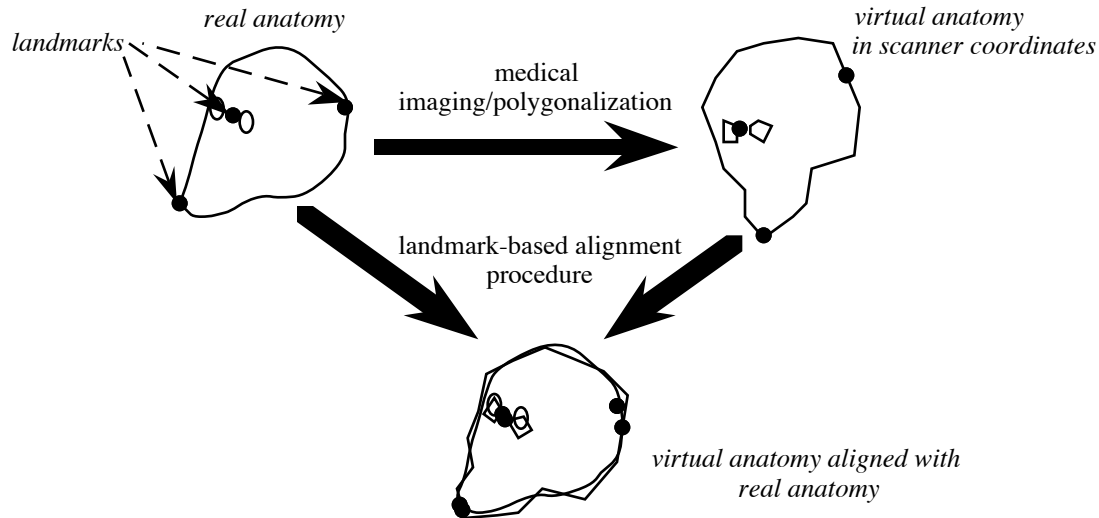
**Figure 2.**   Acquisition/alignment process

In the top part of the figure, the virtual object is created via a scanning or modeling process (indicated by the right-facing arrow). Because of errors in this process (such as scanning artifacts and approximation errors), the virtual object is only an approximation of the object it is intended to represent. In the next part of the figure, the real and virtual objects are aligned in World space via some alignment procedure. A typical method is to digitize landmarks on both objects and use an algorithm such as that described in [Besl & McKay 92] to rotate, translate, and scale the virtual object to be in a least-squares alignment with the real object. This is shown in the bottom part of the figure. Note that there is already some registration error at various points on the real and virtual objects, and that this error is independent of the process of viewing the objects (which is discussed next).

In order to view the real and virtual objects, the system employs a see-through head-mounted display (STHMD), which superimposes the view of the virtual object onto the real object, as shown in the figure below.

screen point

screen

lens

beam splitter

virtual image
of screen

P' = virtual point

P = real point

real object

Figure 3. STHMD operation for one eye

The user sees the virtual image of the screen (created by the lens) reflected off of the beam splitter and can see the real environment as well. The next figure gives a more abstract view from the top showing the situation for both eyes.[2]

STHMD

LE

RE

Image CS

LQ

RQ

T_{S_I}

P'

Sensor CS
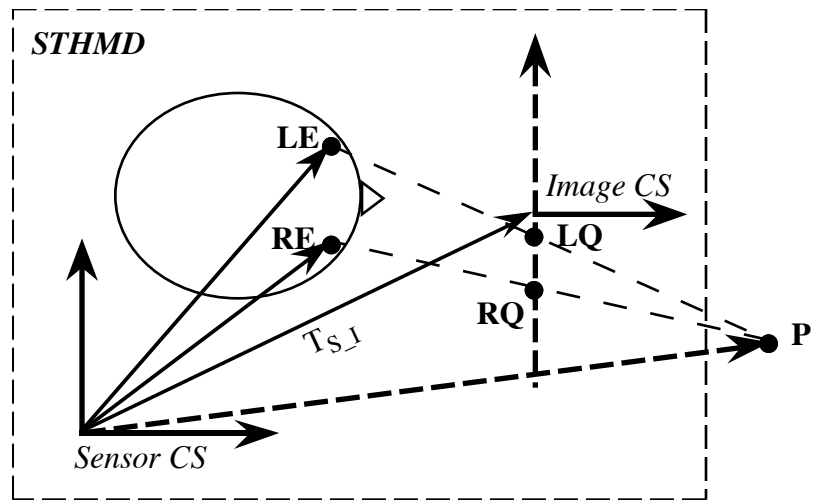
Figure 4. Top view of binocular case showing perceived point

As in the previous figure, P' is the point displayed by the STHMD, and is defined as the intersection of the projectors from the eyepoints (LE and RE) through the projected points (LQ and RQ). The reference coordinate system shown is that of the Sensor, which is the part of the tracker which is attached to the STHMD. The arrow labeled T_{S_I} in the figure represents the transformation between the Sensor and Image coordinate systems. The sensor's position and orientation is reported relative to the Tracker CS, as shown in the next figure.

---

[2] For clarity, errors are not shown in the figures in this section. See [Holloway 95] for a more thorough explanation.
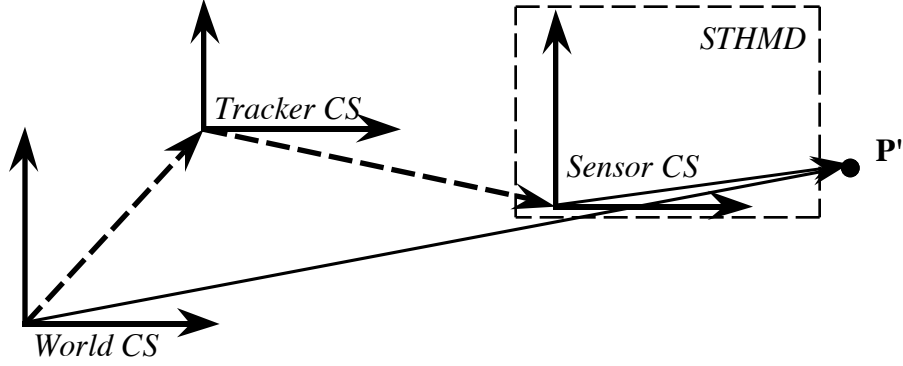
Figure 5. System overview showing STHMD as a black box and all coordinate systems

In this figure, the Tracker CS is the coordinate system defined by the tracker's base or transmitter, which is mounted somewhere in the World CS. The Sensor coordinate system is the reference CS for the STHMD, which displays the point P'. The World-Tracker transformation (represented by the heavy dashed line between these two CSs) is derived via a calibration procedure, and Tracker-Sensor transform is measured and reported by the tracker each frame.

3.3. Error Model Overview

Following the system overview given above, we can simplify the error analysis by dividing the registration error sources into four categories:

1. *Acquisition/alignment error:* Error in acquiring the data for the virtual anatomy and aligning it with the real patient in the laboratory. For this application, the error sources are CT scanning artifacts, approximations made in polygonalizing the resulting CT volume, and errors in the landmark-based alignment procedure.

2. *Head-tracking error:* Error in the World-Tracker and Tracker-Sensor transformations, which define where the STHMD is in World space. Error sources are tracker delay, static and dynamic tracker measurement error, and calibration error.

3. *Display error:* Error made in displaying the computed image. This includes optical distortion, miscalibration of the virtual images with respect to the tracker's sensor, aliasing, nonlinearity in the display devices (*e.g.,* CRTs), and lateral color aberration.

4. *Viewing error:* Error in the modeled location of the user's eyepoints in the computer graphics model. Error sources are calibration error, rotation of the user's eyes and slippage of the STHMD on the user's head.

The error model was derived by examining each of the four types of error and (where possible) deriving separate[3] analytical expressions for the registration error as a function of the system parameters (*e.g.,* viewing distance, sensor-to-transmitter separation, etc.) and the size of the input error (*e.g.,* the magnitude of the translational error in the tracker measurement). Although the classic approach to error analysis is to use partial derivatives to determine a function's sensitivity to errors in its inputs, this approach yielded expressions so large they were useless. Instead, I derived error expressions by modeling the input errors explicitly in the geometry for each situation, which generally yielded smaller, more intuitive expressions. Moreover, while the partial-derivative approach is valid only for small errors, the geometric error model is valid for both small and large

---

[3] To first order, these error sources can be treated as independent; the issue of interaction between the error sources is treated more thoroughly in [Holloway 95].

errors, which is important for examining the behavior of large error sources. Finally, the use of the error metrics discussed in Section 3.1 allowed the model to give finite error bounds whenever possible.

3.4.   Description of System for Testing the Error Model

To test the error model, I conducted a set of experiments to verify that the model was both complete and accurate. That is, I wanted to verify that each error source contributed to the net registration error in the expected fashion *and* that I had not left out any significant error sources. I only checked the equations describing the major sources of error (discussed in the next section); that is, head-tracking error (due to delay, tracker error, and World-Tracker calibration error), optical distortion, and viewing error[4]. The behavior of the smaller error sources (image calibration error, aliasing, display nonlinearity, lateral color, and acquisition/alignment errors) was not tested (except to note the absence of any major effects due to these sources). The experimental results are reported in [Holloway 95] and will not be repeated here, since they add nothing to the discussion of the error expressions themselves. However, the test system itself may be of some interest, since it turned out to be a rather accurate AR system.

The system used for the error model test experiments was the UNC 30° STHMD connected to Pixel-Planes 5 [Fuchs et al. 89] and a Faro Industrial Metrecom mechanical tracker. The Faro arm is a very accurate tracker/digitizer[5] and was used so that the system could be calibrated accurately; however, due to its limited range and unwieldiness, it is not an ideal solution for a real surgical planning system. The figure below shows the experimental setup.

Figure 6.   System overview

The general approach was to calibrate the system as well as possible and then deliberately introduce errors of each type and record their effect on the overall registration error. The setup used for most of these experiments was as follows: A small video camera (a Panasonic model GP-KS102 color CCD camera) was inserted into a Styrofoam head with the entrance-pupil center positioned roughly

---

[4] This was tested not because it was a large error source, but rather because its behavior seemed complex enough to warrant at least a simple check.

[5] According to [Faro 93], the 2σ value for single-point repeatability is .3 mm, and the 2σ value for linear displacement is 0.5 mm. My experience is that it meets these specifications in real use.

at the head's eyepoint. The system was calibrated and then errors from the list above were introduced and their corresponding registration errors measured. The test point in World space was a point on a sheet of graph paper surrounded by a ruled grid, which was used to measure the size of the error in World space. Because the system is calibrated and the errors are artificially introduced, the system has full knowledge of the errors in the transformations and can calculate and display both the correct location for the displayed point *and* the erroneous location, as shown in the picture below.
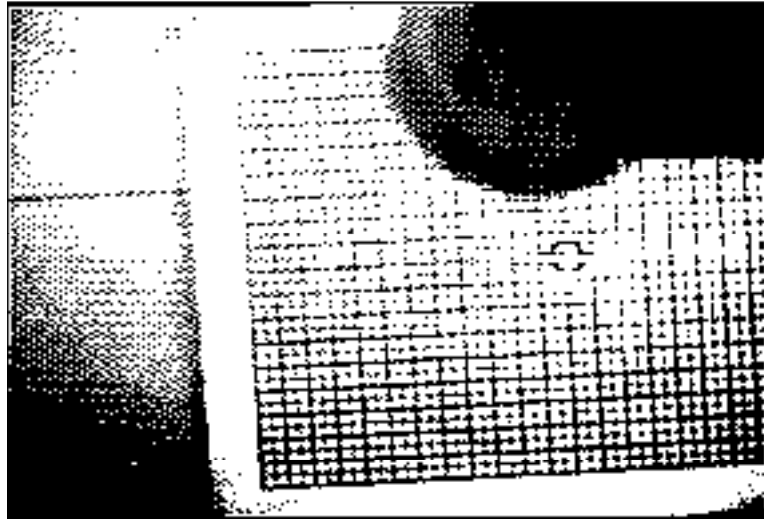
Figure 7. Camera image of virtual and real scene.

In the figure, the large crosshair on the right is the point drawn using transformations containing error (in this case, error in the World-Tracker transform which moves it 20 mm from its modeled location). The small crosshair aligned with the circle is the point drawn using the correct transformations. The box and crosshair to left of the other two crosshairs is just a reference marker fixed in the center of the virtual image. At any instant, the distance between the small and large crosshair gives the registration error due to the deliberately introduced error source, and can be measured on the paper grid in World space or on a distortion-corrected grid in Image space (not shown). Since the location of the camera's "eyepoint" is also known via measurement, we can compute the angular error, the lateral error, and the depth error.

A side benefit to the model verification experiments is that the test system turned out to be a very accurate (albeit unwieldy) augmented reality system. After calibration, the system achieves static registration of 1-2 mm, which is the best of which I am aware. The main reason for the good registration with this system is that it uses an accurate mechanical tracker both for head-tracking *and* for digitizing calibration points, which (as we shall see) avoids some of the largest error sources. The disadvantages to this configuration are its unwieldiness and poor dynamic performance (the system latency is on the order of 300 ms).

The error model was quite useful during the calibration process, since it can be used to determine which parts of the system need to be calibrated carefully and which ones can be approximated. In particular, I found that the parameters whose values are difficult to measure are often those for which precise calibration is not necessary. For example, the z coordinate (*i.e.,* the depth) for the Image coordinate system is difficult to measure with precision, but the net registration error is fairly insensitive to error in this parameter. In contrast, error in the x or y coordinate induces registration error directly and is therefore easy to detect and correct.

The next section discusses the most interesting results of the analysis and related experiments.

4.   Main Results

The results in this section follow the organization just presented, except that small error sources (including acquisition/alignment error) are treated together at the end. The section begins with delay and other head-tracking errors, then treats optical distortion, then viewing error, and finishes with a brief discussion of smaller error sources.

4.1.   Delay Swamps Other Error Sources

It should come as no surprise to anyone who has used an AR system that the largest source of registration error is due to system delay. Even relatively slow head motions can induce large registration errors, which quickly kills the illusion that the virtual objects are fixed in the real environment. System delay is the sum of all the delays from the time the measurement of head position/orientation is made until the time that the image generated using that information is finally visible to the user and is discussed in [Adelstein et al. 92], [Mine 93], [Olano et al. 95] and [Wloka 95]. Although many of the delays that contribute to this tracker-to-display latency are not specifically associated with the tracker, they each contribute to the discrepancy between the real and reported head position and orientation at display time. [Mine 93] gives a complete listing and analysis of delay sources for the UNC system; a similar list follows.

Delay Sources
- *Tracker delay:* This is the time required for gathering data, making calculations in order to derive position and orientation from the sensed data, and transmitting the result to the host. The Polhemus Fastrak is quoted at 4 ms of delay; Mine measured 11 ms at UNC, but this included transmission time and some host processing.
- *Host-computer delay:* This delay includes tasks such as fetching and massaging the tracker data, running host-based application code, and any operating-system tasks.
- *Image-generation delay:* This is the time to render the image corresponding to the current tracker report into the frame buffer. For UNC's Pixel-Planes□5 graphics engine, typical delay values for a small hardware configuration (13 graphics processors, 5 renderers) range from 75 ms for 4,000 primitives to 135 ms for 60,000 primitives. An experimental low-latency rendering system developed by Cohen and Olano [Cohen & Olano 94] reduced the delay to 17 ms, but only for very small datasets (100-200 triangles).
- *Video sync delay:* This is the delay while waiting for the next video frame to begin. The worst case for a 60 Hz refresh rate is 16.7 ms, and the best case is a synchronized system for which the delay is zero.
- *Frame delay:* Most raster devices paint the image sequentially from top to bottom. For a 60 Hz non-interlaced display, the delay is roughly 17 ms between the display of the upper left pixel and the lower right pixel.
- *Internal display delay:* Some display devices add additional delays due to processing within the display device itself. For example, [Mine 93] reports that the LCDs in an HMD in use at UNC added an additional field time (about 17 ms) of delay. This could be due to the display having a different resolution from the input signal and having to resample the input before it can display the current frame.

For immobile objects, the amount of registration error due to delay is determined by the amount the head moves from the time the tracker makes its measurement until the image is scanned out. If we use a simple first-order model for head motion, the general expression for bounding the delay-induced error is

$$b_{delay} \; = \|\dot{\mathbf{v}}_{\;head}\Delta t\| + 2{\cdot}\sin\frac{|\ddot{\varnothing}_{head}\Delta t|}{2} \; \cdot \; \|\mathbf{v}_{S\_P}\| \tag{3}$$

where $\dot{\emptyset}_{head}$ and $\dot{\mathbf{v}}_{head}$ are the angular and linear velocity of the user's head, and Δt is the net delay (which in the worst case is the sum of all the delays listed above). The values for $\dot{\emptyset}_{head}$ and $\dot{\mathbf{v}}_{head}$ will clearly be application-dependent: one would expect fighter pilots to have higher velocity values than surgeons, for example.

To get an idea of representative head velocities for surgery planning, I measured the angular and linear velocities of a physician's head (with a Fastrak magnetic tracker) while he conducted a simulated planning session. Most of the head movements were slower than about 50 deg/s and 500 mm/s, and the average velocities 164 mm/s and 20 deg/s. This is consistent with data collected by Azuma [Azuma 95] for naive users in a demo application: the linear velocities peaked at around 500 mm/s, and most of the angular velocities were below 50 deg/s (although the peak velocities did get as high as 120 deg/s in some cases).

If we take 500 mm/s and 50 deg/s as fairly conservative upper bounds for head movement and plug them into the expression for b_{delay} for the minimum delay number for the normal Pixel-Planes rendering system (65 ms), we get

$$b_{T\_S} \;=\; 500 \text{ mm/s·.065s} + \; 2 \cdot \sin\frac{50\ \text{deg/s·.065s}}{2} \cdot 500 \text{ mm} \;=\; 28.4 + 32.5 = 60.9 \text{ mm}$$

which is clearly a very large error. If we plug in the mean velocities, we get 22 mm. This is still quite a large error and gives an indication of just how serious a problem delay-induced registration error is. Note that, at least for this application, the linear and angular terms contribute equally to the net registration error. Note also that these delay values have not included Δt_{frame}, the time to draw a full NTSC field, which adds up to 17 ms for the last pixel scanned out.

Using these numbers, a simple rule of thumb for this application is that *we can expect about 1 mm of registration error for every millisecond of delay in the worst case and $\frac{1}{3}$ mm/ms in the average case*. Note the significance of this result: if our goal is registration to within 1 mm, unless we do predictive head tracking, the system will only have (in the worst case) 1 millisecond to read the tracker, do its calculations, and update the displays! Even the most aggressive strategies for reducing system delay cannot hope to achieve this level of performance. *The only hope for good dynamic registration will be to use predictive head tracking*.

In summary, delay is clearly the largest error source in our current system, and is likely to be a problem for the foreseeable future. For the maximum head velocities and typical system delays, delay-induced registration error is greater than all other registration errors combined. Angular velocity seems to dominate for applications in which the user is surrounded by the data (e.g., a building walkthrough), while translational velocity is more of a factor in applications where a single object is being studied (as in surgery planning). Azuma and Bishop [Azuma & Bishop 94] have achieved good results for moderate head velocities using predictive filtering with rate gyroscopes and linear accelerometers, but the problem is far from solved. One of their results is that prediction errors increase at greater than linear rates with respect to increasing prediction intervals, which means that prediction may not be effective for long delays (which for their data was > 80 ms). Thus, systems must be optimized for low latency [Olano et al. 95], which is in direct conflict with the need for high throughput. In addition, techniques to synchronize the rendering process with the display scanout (such as beginning-of-frame synchronization, frameless rendering [Bishop et al. 94], and just-in-time pixels [Mine & Bishop 93]) will also be essential for reducing delay-induced error.

4.2. Beware the World Coordinate System

It is common practice to have a user- or system-defined World CS as a reference. An example of when we need a World CS is when we have a special digitizer (such as a camera measurement system) for precisely locating points in W, but which is not suitable for head tracking. We then

have W as the reference CS and T is expressed relative to it. To understand the error this causes, let us examine the process of aligning virtual points within the real environment. If we want the virtual point P' to coincide with the real point P, we must express the location of P in some coordinate system known to the system. In one method, we measure P relative to some World CS which we have defined for convenience, and then transform the vector from W to P into Sensor space for viewing in the STHMD via the transformation:

$$\mathbf{v}_{S\_P} \;=\; T_{S\_T} \cdot T_{T\_W} \cdot \mathbf{v}_{W\_P} \tag{4}$$

where T_{S_T} is the inverse of the Tracker-Sensor transform (reported by the head tracker), T_{T_W} is the inverse of the World-Tracker transform (which expresses the Tracker CS in W), and $\mathbf{v}_{W\_P}$ and $\mathbf{v}_{S\_P}$ are the vectors to P from W and S, respectively.

A second method is to measure P with respect to the Tracker CS directly by digitizing the point via a number of measurements in Tracker space. This reduces the previous equation to

$$\mathbf{v}_{S\_P} \;=\; T_{S\_T} \cdot \mathbf{v}_{T\_P} \tag{5}$$

This method has better error properties (as we shall see), but is not always an option, since accurate digitizing trackers are often ill-suited for head tracking (as mentioned in Section 3.3).

The problem with the first approach lies in its error propagation behavior. In the absence of other errors, the linear registration error due to error in T_{W_T} is given by

$$b_{W\_T} \;=\; \|\delta\mathbf{v}_{W\_T}\| + 2\cdot\sin\frac{|\delta\varnothing_{W\_T}|}{2}\cdot(\|\mathbf{v}_{T\_S}\| + \|\mathbf{v}_{S\_P}\|) \tag{6}$$

Here, $\delta\mathbf{v}_{W\_T}$ is the error in positioning the tracker's origin in W, $\delta\varnothing_{W\_T}$ is the orientation error of T in W, $\mathbf{v}_{T\_S}$ is the vector from the origin of T to the origin of S, and $\mathbf{v}_{S\_P}$ is the vector from the Sensor CS to P. Note that while the translational error in T_{W_T} just adds to the net error, the rotational error term is scaled by the magnitudes of two vectors which may be rather large.

As reported in [Janin et al 93], the origin and orientation of magnetic trackers (such as the Polhemus Fastrak) are difficult to measure directly with any accuracy, since the origin is inside of a transmitter. Therefore, a common approach for orienting and positioning the tracker source in World space involves taking tracker readings and deducing the T_{W_T} transform from them. If we assume that the determination of T_{W_T} is limited by the static accuracy of the tracker, we can use the specifications for the tracker to get another estimate of the net error. The quoted specifications for the Polhemus Fastrak for distances up to 760 mm are 0.15° RMS for static angular accuracy and 1.3 mm static translational accuracy [Polhemus 93]. If we assume that the user's head is 500 mm from the tracker origin and that P is 500 mm from the sensor, we get a linear registration error of 4 mm *just from error in locating the tracker in World space*. If the tracker-head distance and sensor-to-point distances go up to 1000 mm, the error reaches 6.6 mm. Again, this is independent of error in measuring the head position and orientation, which will add even more error.

The heart of the problem is that angular errors in orienting the tracker precisely in W are magnified by the "moment arm" of the tracker-to-point distance, which can be quite large. If we can eliminate this transform from the system by measuring point locations relative to the tracker, the net error should go down. For systems that require a separate digitizer for aligning the virtual objects in the real environment (and therefore a World CS), it may be possible to use the scaling behavior of this error source in order to calibrate it out of the system; that is, by using large head-to-tracker and head-to-point distances, we might be able to use this moment arm to our advantage to reduce $\delta\mathbf{v}_{W\_T}$ to a negligible level.

4.3. Tracker Measurement Error

Apart from the delay error and World-Tracker error already discussed, there is the problem of error in the measurement of head position as reported by the tracker. We can break the tracker error into three categories:

1. *Static field distortion:* This is any systematic, repeatable distortions of the measurement volume, such as the warping seen in the presence of metal for magnetic trackers. This distortion can be corrected via calibration to the extent that it is repeatable and systematic.

2. *Non-repeatable tracker error (or jitter):* This is error that cannot be calibrated out of the system and includes both short-term variations due to noise and long-term variations that cause readings to change from one day to the next.

3. *Dynamic tracker error:* This is any error which is a function of the sensor's motion. For example, systems that assume the sensor's motion is negligible with respect to their measurement interval will have some amount of error for moving objects.

The problem with quantifying tracker error is that it is very dependent on the tracker technology and the environment in which the tracker is used. For example, magnetic trackers are sensitive to metal and electromagnetic fields in their operating environment, yet most AR setups are in labs chock full of electronic equipment, and have significant amounts of metal in walls, floors, etc. In such hostile environments, the error in the tracker measurements may exceed the manufacturer's specifications by an order of magnitude or more.

As indicated above, there are two questions for static tracker accuracy: 1) How much noise (or jitter) is there in the tracker readings over time, and 2) Can we calibrate the tracker so that the average accuracy is roughly equal to the average jitter?

The work of [Bryson 92] addresses both questions for a Polhemus Isotrak. They measured the accuracy of a Polhemus Isotrak and reported that for source-sensor distances of up to 760 mm, readings taken on two different days varied as much as 1"-2" (25-51 mm), even though the standard deviation of the readings in any one-second period was less 3 mm (all errors increased with the source-sensor distance). They tried several calibration methods and were able to calibrate the tracker to within 1-2" for separations of around 30". In short, they were able to calibrate the tracker to within the long-term jitter, but the net error was still about ten times the standard deviation of the short-term jitter (25-50 mm).

I measured the jitter of the Polhemus Fastrak and the Ascension Flock of Birds in the UNC laboratory; summary plots for both are given below.
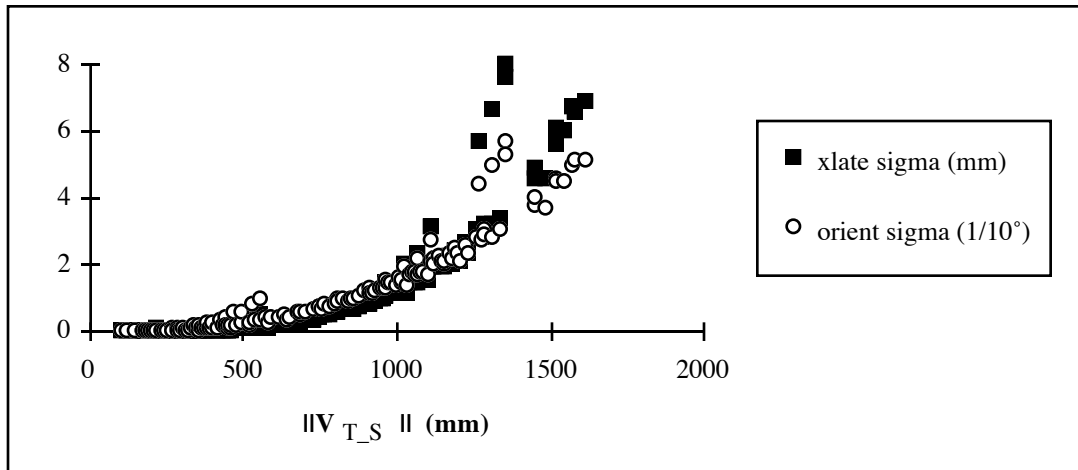
Figure 8. Translation (in mm) and orientation (in tenths of degrees) jitter sigma values for Fastrak vs. transmitter-to-sensor distance (in mm)

For the Fastrak for transmitter-sensor separations of less than 500 mm,[6] the translation sigma values are 0.25 mm or lower, and the orientation sigmas are all below 0.05°. At 1000 mm, the translation sigma rises to 1.3 mm and the orientation is 0.15°. The readings were taken over intervals of a few seconds (although intervals of a few minutes showed no significant difference); longer intervals were not tested.
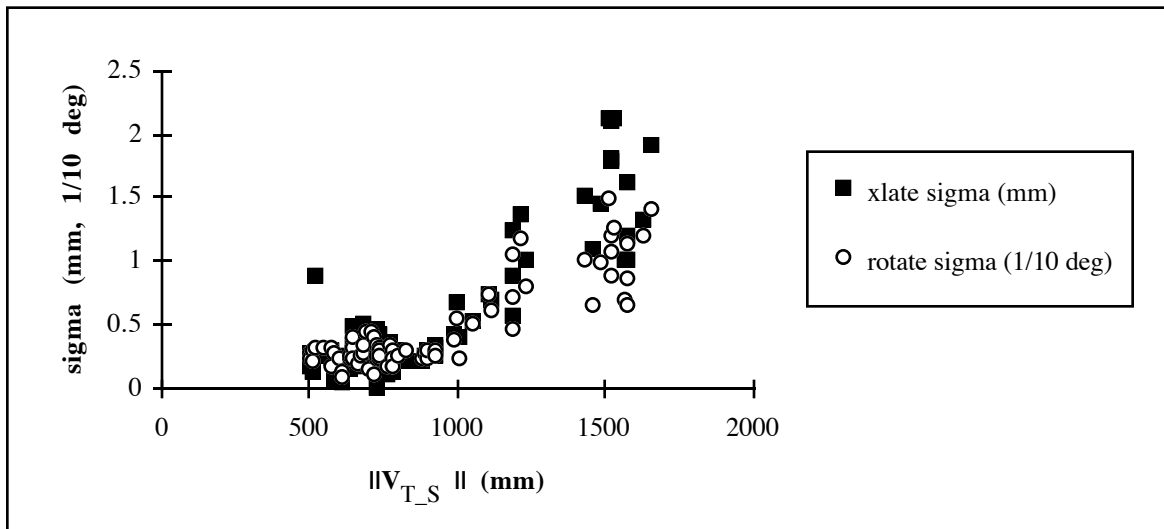
Figure 9. Translation (in mm) and orientation (in tenths of degrees) jitter sigma values for Flock of Birds vs. transmitter-to-sensor distance (in mm)

For the Flock of Birds with the extended range transmitter, transmitter-to-sensor separations of less than 500 mm led to saturation of the sensor inputs and therefore readings were not taken in this region. The jitter was generally less than 0.5 mm and 0.05° for separations of 500 mm - 1000

---

[6] In the head-motion study cited earlier, I also measured the range of head motion for the surgeon and found that most of the time his head was within 500 mm of the patient; therefore a centrally located transmitter could keep the transmitter-sensor separation to 500 mm or less most of the time.

mm, and then rose to about 2 mm and 0.15° at 1500 mm. As with the Fastrak, the jitter appears to be a function of the square of the source-sensor separation (because of the falloff of the magnetic field with distance).

Efforts at calibrating the Flock at UNC have not come close to the measured short-term jitter values. Recent work by [Livingston & State 95] reports that they were able to calibrate the Flock to an average error of 5 mm (in a volume roughly equal to one half a cubic meter) for translation error (down from an average error of 42 mm before calibration). This is similar to what Bryson reported: his calibration reduced the error to about 10 times the short-term jitter standard deviation. Thus, while calibration can reduce tracker error significantly, calibrating trackers to the one-sigma level looks like a non-trivial task.

Turning now to the error model, the sensitivity of overall registration error to tracker measurement error is given by

$$b_{tracker} \;=\; \|\delta \mathbf{v}_{T\_S}\| + 2 \cdot \sin\frac{|\delta \varnothing_{T\_S}|}{2} \cdot \|\mathbf{v}_{S\_P}\| \tag{7}$$

This shows that translation error just adds to the registration error, but rotational error is magnified by the distance to the point. For $\|\mathbf{v}_{S\_P}\| = 500$ mm, each tenth of a degree of angular error yields about a millimeter of registration error.

If we use the specified static accuracy for the Polhemus Fastrak (1.3 mm, 0.15°), we get 2.6 mm of error for $\|\mathbf{v}_{S\_P}\| = 500$ mm. If the error is as bad as 10 times the measured jitter value at 500 mm (*i.e.,* 2.5 mm and 0.5°), we get about 7 mm of registration error.

As for dynamic error, recent work by [Adelstein et al. 95] indicates that for the Fastrak and the Flock, there does not seem to be any additional error for a moving sensor for normal volitional head motion.

The upshot is that tracker calibration is a difficult but important task and that systems using magnetic trackers should do as much as possible to reduce the source-sensor distance in order to reduce the non-repeatable tracker errors. Equation 7 gives an idea of quality of registration one can expect for a given amount of non-repeatable tracker error.

4.4.   Optical Distortion

Optical distortion (hereafter referred to as *distortion*) is the most significant optical aberration for most HMD systems and has been analyzed in [Robinett & Rolland 91][Rolland & Hopkins 93]. Distortion is a lateral shift in the position of the imaged points which can be approximated to third order by the following equation:

$$r_i \;=\; m \cdot r_s + k(m \; r_s)^3 \tag{8}$$

where r_i is the radial distance from the optical axis to the point in image space, m is the linear magnification, k is the third-order coefficient of optical distortion, and r_s is the radial distance to the point in screen space. If k is positive, the magnification increases for off-axis points, and the aberration is called *pincushion distortion;* if k is negative, the magnification decreases, and it is called *barrel distortion*. Pincushion distortion is more common in HMD systems and is pictured below.
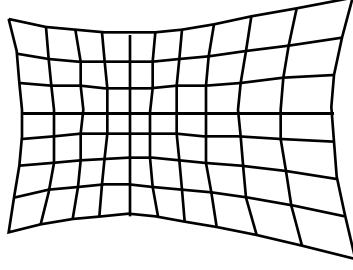
Figure 10.   Pincushion distortion

Since this error does not vary with time, it can be corrected by pre-warping the image prior to display so that it appears undistorted when viewed through the optics (several approaches for this are given in the above references). While optical and electronic methods for predistortion exist, they are not always feasible for various reasons and many systems can only predistort in the rendering process. Currently, though, predistortion is so computationally intensive that it may induce more system-latency error than the warping error it corrects. For example, on Pixel-Planes□5, Lastra [Lastra 94] reports that he was able to achieve 20 frames per second with predistortion, but only by adding it as a stage in the rendering pipeline. This added a frame of delay, or about 50 ms, which corresponds to about 50 mm of error for the head velocities observed in this application. This is a much larger registration error than that introduced by the distortion itself, and leads to the conclusion that predistortion in the rendering process is not a quick and simple fix for all systems. It is therefore useful to examine the effect of uncorrected distortion in order to compare it with other error sources.

The general term for lateral display error is

$$s_{display} \approx (d{-}z)\,\frac{q}{d} \tag{9}$$

where *q* is the magnitude of the display error in the projection plane. Note that lateral display error is zero at the eyepoint (where z = d) and increases linearly as z → -∞. Assuming that we have correctly modeled the linear magnification, the *q* value for distortion (in the absence of other errors) is

$$q_{dist} = k(m\,r_s)^3 \tag{10}$$

The plot below shows the distortion error for the current UNC STHMD, for which m = 6.0, k = 2.66 x 10^{-6} mm^{-2}, and the screens are 54.7x41 mm. *x_n* is the normalized *x* coordinate in screen space (similarly for *y_n*) and we have used the previously calculated values for *k* and *m*. Because of the 4:3 aspect ratio, the normalized screen-space coordinates have maxima in *x* and *y* at 0.8 and 0.6, respectively, and the corner is at unit screen-space radius. It is clear from the plot that the error in image space becomes quite significant in the corners of the image, where the distortion error is 23 mm (corresponding to 11.2% distortion). At the center of right edge, the distortion error gets up to 11.8 mm (7.2%), and at the center of the top edge it reaches 5 mm (4%). For points 200 mm beyond the screen image, the lateral error is 32 mm at the corner, 17 mm at the left/right edge, and 7 mm at the top/bottom.
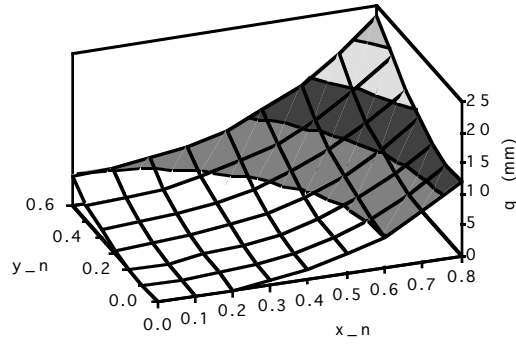
Figure 11.   Distortion error

In general, the distortion error scales linearly with k and as the cube of $m \cdot r_G$. Thus, for a given value of k, using a larger display device to increase the system FOV will also increase the distortion error significantly at the edges of the larger virtual image. Because distortion is systematic, we can look at the binocular case to see what sort of warpings in depth it is likely to cause. The following figure shows a top view of the warping of a square caused by distortion:
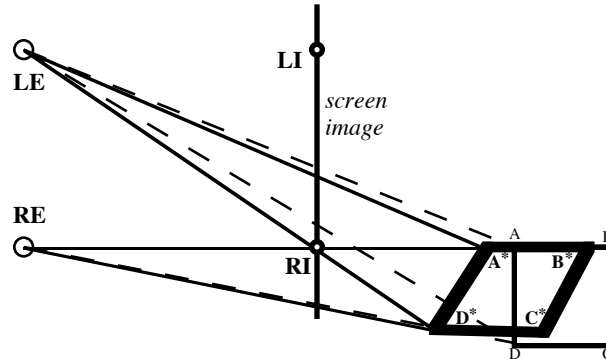
**Figure 12.**   Warping due to optical distortion

The points LI and RI denote the centers of the two coplanar images and therefore the centers for distortion in each image. Thus, the projectors for the points A and B for RE pass through RI and are not distorted (since $r_s = 0$), whereas all the other displayed points have non-zero r_s values and are moved accordingly. The distorted projectors are shown for A* and D*; note how much more the projectors from LE are moved than those from RE, since the projected points for LE are much further from LI. Distortion tends to cause peripheral objects to "wrap around" the user; that is, it tends to move points further into the periphery but shifted inward toward the user.

In general, distortion is a small error source in the center of the images but increases rapidly in the periphery and can become quite large. For objects that fill the field of view, the misregistration may well be unacceptable. Moreover, because distortion is an image-space error, the amount of warping will be a function of where the object is drawn within the field of view, which means that the object will seem to change shape as the user's head moves. Finally, because the eyes converge to look at an object, one or both eyes will typically be looking at image points that are not at the exact center of the image, which means that some amount of distortion error will be present even in the best of cases.

4.5. Is Eye Tracking Unnecessary?

Another source of registration error is *viewing error*, which is the error in the modeled eyepoint locations. This is not usually a large source of registration error, but one of the byproducts of the analysis was the realization that there is a method for calibrating systems such that eye tracking may not be necessary in order to eliminate the small error that eye rotation causes. This section begins with a discussion of viewing error in general and then moves on to the issues of eye tracking and system calibration.

For this discussion, I use the center of the entrance pupil[7] *E* as the eyepoint, following [Rolland et al 95] rather than the first nodal point[8] *N* as in [Deering 92]. *E* is approximately 11 millimeters forward of the center of rotation for the eye[9] (vs. 6 mm for the first nodal point), as shown in the following figure.
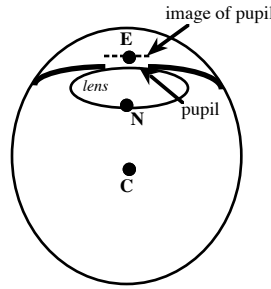
Figure 13. Simple schematic of eye

The eyepoints deviate from their modeled locations for two reasons: calibration error and eye movement. That is, a calibration procedure is used to derive the eyepoint locations, but the actual eyepoint E will deviate from the modeled eyepoint E' because of eye movement[10] and error in the procedure. The approximate bound for the resulting lateral error is

$$s_{view} \approx e \cdot \frac{|z|}{d} \tag{11}$$

where *e* is the magnitude of the viewing error, *d* is the distance from E' to the screen image, and *z* is the distance *from the screen image (or projection plane)* to P'. The first observation is that the registration error due to viewing error goes to zero for points in the projection plane[11] (like P_4 in Figure 14) since P = Q and z = 0, which suggests that we should position the screen's virtual image in the center of the working volume (preferably dynamically) in order to minimize the effects of viewing error. Also, this property should also be of use in system calibration, since it helps distinguish viewing error from other error sources. While it might seem that letting d → ∞ (i.e., using collimated images) would reduce viewing error to zero, the situation is somewhat more complex. *z* is measured relative to the virtual image (not the eye) so that when *d* becomes large, *z*

---

[7] The *entrance pupil* of the eye is the image of the pupil seen through the cornea.

[8] A ray passing through the first nodal point will emerge at the same angle from second nodal point.

[9] This value was calculated using data from [Longhurst 57]. The entrance pupil is about 0.5 mm forward of the pupil itself.

[10] If an eye tracker is used, there will still be residual error which we can attribute to eye movement.

[11] [Oishi & Tachi 96] also noted this property and use it to make their calibration method more accurate.

also becomes large (for relatively close points). Thus, moving the projection plane to infinity will make z/d□→□1 for near points, inducing lateral errors approaching *e*. For close work with shallow depths of field, this is probably not desirable. However, for applications requiring a large range of depths, putting the virtual image at infinity has the advantage of capping the lateral error at a value equal to *e*, while a smaller value for *d* can induce large lateral errors for z/d >> 1.

Let us now examine the case where all the error is due to eye rotation. That is, we assume a perfect calibration procedure which identifies E when the eye is looking straight ahead, and then examine the error as the eye rotates. If we designate the measured, straight-ahead eyepoint by E' and rotate E by an angle ε about C, the viewing error magnitude is given by

$$ e = \; \|\mathbf{v}_{C\_E} - \mathbf{v}_{C\_E'}\| \; = \; 2 \cdot \sin\tfrac{|\epsilon|}{2} \, \cdot \, \|\mathbf{v}_{C\_E}\| \tag{12} $$

For a 60° monocular FOV, we would expect ε to range from -30° to +30°, corresponding to an eyepoint movement of ±5.7 mm in the worst case. If we use d = 500 mm and e = 5 mm, we find that the lateral errors for points in the range 0□≤ |z| ≤ 500 mm vary linearly from 0 to 5 mm (i.e., the lateral error increases by 1 mm every 100 mm). Depending on the precision required by the application, it would seem that eye tracking would be the only way to reduce this error. Fortunately, there is reason to hope that eye tracking will not be necessary if the point C can be located with precision. That is, it turns out that using C as the modeled eyepoint may reduce the viewing error in this case to a negligible level, even without eye tracking. This is because C is always aligned with the true eyepoint for a point in the center of the eye's field of view, as shown in Figure 14.
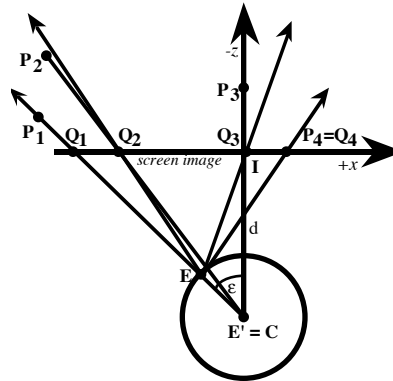
Figure 14. Viewing error for modeled eyepoint at center of rotation

The figure shows four points, P_1-P_4, projected using E' = C as the center of projection. As the eye rotates about C to fixate on P_1, E comes into alignment with E' (= C), Q_1, and P_1, and thus there is no registration error for P_1. While the eye is fixated on P_1 (and Q_1), there is a slight registration error for P_2, a larger error for P_3, but none for P_4 since it is in the projection plane. Similarly, if the eye rotates to fixate on P_3 for example, E, E', Q_3 and P_3 will all fall on the same line, and the registration error for P_3 will then be zero. Thus, when the eye is not looking at a point, it will have some amount of registration error, but when it turns to look at the point, its error goes to zero.

The next question is: how much viewing error is induced in the non-fixated points? The answer depends on the viewing parameters, but appears to be fairly small in general. For d = 500 mm, the angular errors for a 60° FOV HMD are less than 1.5° for points from 200 mm from the eye on out to infinity (points closer than 200 mm can have much larger angular errors, but these points are closer than the *near point*, or the closest point of comfortable focus for an adult [Longhurst 57]). Points lying along or near the gaze direction and points within or near the screen image will have zero or very small angular registration error. Although the angular errors calculated here

correspond to large lateral errors for distant points, because the angles corresponding to the errors are small, it is unlikely that the human eye would detect such errors due to the falloff in acuity for non-foveal vision. Thus, although it remains to be confirmed by user studies, there is reason to hope that eye tracking will not be necessary for systems that can accurately locate the center of rotation of the user's eyes. Another benefit of this result is that in some cases it is easier to find C than E, since calibration procedures often require the eye to swivel in order to align itself with two or more World-space vectors (as in [Azuma & Bishop 94] and one method used in [Janin et al 93]); such procedures identify the center of rotation rather than the eyepoint[12].

As for eye calibration error, if the center of rotation of the eye is used as the center of projection, the lateral error is bounded by

$$s_{view} \approx c \cdot \frac{|z|}{d} \tag{13}$$

where c is the magnitude of the calibration error. For $d = 500$ mm, 5 mm of calibration error will induce lateral errors of about 5 mm for points 500 mm from the screen image (and 0 mm at the screen image). Clearly, the error is linear in c, so halving the calibration error will halve the lateral error, etc.

Finally, if we examine the binocular case, we can characterize and quantify some of the distortions induced by viewing error. Two cases of particular interest are rigid-pair motion, where the eyes translate together relative to the screen image, and inter-pupillary distance (IPD) error, in which the modeled IPD is different from the actual IPD. As noted in [Hodges & Davis 93] and [Rolland et al. 95], rigid-pair motion induces a shear distortion for horizontal/vertical motion, and a compression/elongation distortion for in/out movements. According to the error model, the deviation of each point is equal to $\frac{|z|}{d}\,e$ for a motion of size e.

IPD error can induce gross registration errors in depth[13], as shown by the exaggerated case depicted in the next figure.
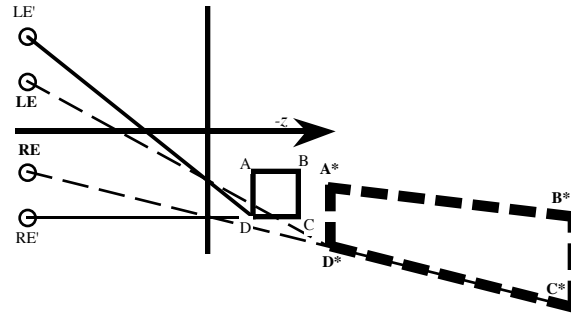
Figure 15. IPD-error distortion

In this top view, LE' and RE' are the modeled eyepoints and LE and RE are the actual eyepoints. The heavy dashed line indicates the theoretically perceived version of the square ABCD. In

---

[12] Note that if the user is wearing eyeglasses, the point to use is the image of C as seen through the glasses, which is exactly the point located by these calibration procedures.

[13] That is, the mathematical model shows gross errors in depth based on the intersection of the projectors from each eye; since the process of visual perception is based on multiple depth cues, it is somewhat unlikely that the depth error predicted by the model will be a reliable predictor of *perceived* depth error.

general, the error vector between the theoretically perceived point P' and the real point P = (x, y, z) is

$$\mathbf{v}_{P'\_P} \; = \; \frac{1}{1\square + \square \frac{i'}{i}\frac{z}{(z-d)}} \; \cdot \begin{bmatrix} x \\ y \\ (z-d) \end{bmatrix} \tag{14}$$

where i' is the modeled IPD and i is the actual IPD. Note that for suitable values of the parameters, the denominator can go to zero, leading to an infinite-length error vector, corresponding to the case where the projectors for a point become parallel for the perceived point.

In summary, viewing error is not likely to be a large registration error source, but its effects can be minimized by using the center of the eye's rotation as the center of projection, calibrating the system carefully for each user, and setting the screen image in the center of the working volume (since the registration error due to viewing error is zero there). Systems displaying objects at different depths may need to have an automatic adjustment for screen-image distance, which would also make the accommodation distance for the virtual and real objects the same.

4.6.   Other Error Sources

This section will briefly discuss each of the remaining, small error sources. While these are small in comparison to the other error sources, it is worth noting that they would have to be dealt with in a system requiring sub-millimeter accuracy, and might prove difficult to correct as well.

- *Virtual object alignment/scanning:* These are errors accrued in modeling or scanning the virtual objects and aligning them in the real environment. The amount of error is clearly a function of the accuracy of the scanning/modeling method and the accuracy of the alignment process. Since both of these are very application-dependent, it is very hard to generalize about this error source. For 3D registration of medical datasets, the literature seems to indicate that mean-squared error values of less than 1 mm can be achieved, although larger errors (2-8 mm) are not uncommon depending on the alignment method and scan accuracy [Udupa & Herman 91]. A general registration algorithm by Besl and McKay [Besl & McKay 92] reports good results at matching 3D shapes, often to within 0.1% of the shape size, which for a head-sized dataset would correspond to less than a millimeter. [Holloway 95] examines the behavior of this alignment algorithm in the presence of errors in picking the landmarks.

- *Virtual-image calibration errors:* In order to properly render the stereo projections of the virtual objects, the transformation between the Sensor CS and the Image CSs must be determined precisely. The approximate image-space error q for image alignment error is

$$q_{im} \; \approx \; \|\delta\mathbf{v}_{S\_I}\| \; + 2\sin\frac{|\delta\emptyset_{S\_I}|}{2}\,\|\mathbf{v}_{I\_P}\| \tag{15}$$

  which is just rigid-body transformation error of the projected points. Various calibration methods and their results have been reported [Janin et al 93][Azuma & Bishop 94].

  The critical observation regarding virtual-image misalignment is that the resulting error is fixed in image space and is therefore independent of viewing direction, etc. Moreover, certain misalignments are readily detectable (and therefore correctable). For example, if the screen image is shifted in *x* or *y* relative to its modeled location by 2 mm, *all* of the points in the scene will be shifted by that amount. Rotation about the *z* axis will displace points as a function of their distance from the rotation axis and can easily be detected with the use of a crosshair or grid. Errors in the remaining three degrees of freedom (*z* translation, *x* and *y* axis rotation) are less easily detected, *but that is precisely because they do not induce much registration error unless the error is severe*. That is, these errors for the most part move points along their projectors, which, because of the projection operation, has very little effect

on q_{im}. If the errors are severe, they will be systematic and can therefore be distinguished from other error sources and corrected. In summary, errors in this transformation which induce noticeable registration error can be corrected, and those which do not induce noticeable registration error can be ignored. Based on these observations and the experience with the prototype system, I estimate that this error can be reduced to 1 mm or less.

- *Aliasing:* If antialiasing is not done (*e.g.*, for performance reasons), the worst-case error is for all of the edge pixels for a primitive to be shifted by half a pixel in x and y. In this case, the primitive's center of mass will shift by $\sqrt{2}/2$ times the pixel spacing. Assuming the display does not resample the signal from the frame buffer, the net error is just this shift magnified by the optics, or

$$ q_{al} \; = \; m \cdot \frac{\sqrt{2}}{2} \, p \qquad\qquad (16) $$

where p is just the screen width divided by the horizontal resolution (for square pixels), and m is the linear magnification. For a 52 mm-wide LCD screen with 640 pixels/line and a magnification of 6.0, this amounts to 0.4 mm of error. Aliasing should not be a major error source for most systems.

- *Display device non-linearity:* Certain displays (CRTs in particular) have non-linearities that cause the final screen display to deviate from a regular rectangular grid. For CRTs, there non-linearities in the beam deflection process that can distort the final image. This non-linearity is quoted as a percentage of the screen size; values range from 0.1% to 3%. While values of 3% can induce enough error to be troublesome (on the order of 5 mm), values of 1% (corresponding to 1-2 mm of error) are more common. Moreover, any prewarping implemented for distortion correction could be modified to compensate for this problem as well. Finally, one can always use more elaborate (and expensive) drive electronics to further reduce this error source.

5.   Conclusions

Most of the major error sources are associated with the tracker in some way. The tracker is a critical component for making augmented reality work, and any error or delay in its data causes serious flaws in the illusion. The errors associated with the tracker are due to delay in displaying the tracker data (due to delay in the entire system), error in the tracker measurement, error in locating the Tracker CS in the World CS, and errors in tracker readings used for system calibration.

Another result of this analysis is that eye tracking is probably unnecessary if the eye's center of rotation is used as the center of projection, since it gives the correct projection for fixated points and small errors for non-fixated points. The last of the major error sources is optical distortion, which can cause large errors in the image periphery; unfortunately, inverting the distortion in the rendering process may cause more delay error than the warping error it corrects. Finally, there are several other small error sources, each of which may add a small amount of registration error.

The analysis shows that sub-millimeter registration is not likely any time soon since there are many error sources on the order of a millimeter; but it does seem probable that we will achieve 5-10 millimeter dynamic accuracy (and 1-2 mm for static accuracy) with the use of predictive tracking, synchronized display methods, and careful calibration. The progress toward sub-millimeter registration error will probably be asymptotic, with increasing effort and expense required to gain each small increase in precision.

6.   Future Work

The error model discussed here was tailored to a particular application (surgery planning) and as such was not thoroughly explored for different systems and different applications. In particular,

this model could easily be expanded in order to analyze video STHMD systems, CAVE systems[14], and opaque HMDs. For video STHMDs, the model for viewing error would have to be changed, but much of the rest of the model would work as is. In CAVE systems, many of the problems of head tracking disappear (since the images are fixed in the environment), but the analysis of viewing error would be rather useful, especially since multiple users often view a scene which is only correct for the user wearing the head-tracker. Finally opaque HMDs do not have the strict requirements for registering real and virtual objects (since the real objects are not visible), but nevertheless suffer from the apparent swimming of virtual objects due to delay and tracker error, as well as the visual distortions from viewing error and optical distortion.

For the most part, however, the future work suggested by this research is not in the area of extending the work presented here, but rather in addressing the problems that it describes. In particular, more work needs to be done in the following areas: tracker and system calibration methods, low-latency rendering, synchronized rendering with just-in-time incorporation of tracker measurements, predictive and hybrid tracking methods, and feedback methods such as those used in video STHMDs.

7. Acknowledgments

8. References

Adelstein, B, E Johnston, S Ellis. 1992. A testbed for characterizing dynamic response of virtual environment spatial sensors. *Proceedings, UIST '92, Fifth Annual ACM Symposium on User Interface Software and Technology*. Monterey, CA. pp. 15-22.

Adelstein, B, E Johnston, S Ellis. 1995. Dynamic response of electromagnetic spatial displacement trackers. Submitted for publication.

Azuma, R, and G Bishop. Improving static and dynamic registration in an optical see-through HMD. *Proceedings of SIGGRAPH '94*. Orlando, July 24-29, pp. 197-204.

Bajura, Michael, and Ulrich Neumann. 1995. Dynamic registration correction in AR systems. *Proc IEEE Virtual reality annual international symposium* (VRAIS).

Bajura, M., H. Fuchs, and R. Ohbuchi. 1992. Merging Virtual Objects with the Real World. *Computer Graphics*. Number 26. pp. 203 -10.

Besl, P, and N McKay. 1992. A method for registration of 3-D shapes. *IEEE Trans. on Pat. Anal and Mach. Int*. 14:2.

Bishop, G, H Fuchs, L McMillan, E Zagier. 1994. Frameless rendering: Double buffering considered harmful. *Proceedings of SIGGRAPH '94*.

---

[14] CAVE systems use several rear-projection screens and head-tracked stereo glasses to immerse the user in a 10'x10' virtual environment [Ghazisaedy et al. 95].

Bryson, S. 1992. Measurement and calibration of static distortion of position data from 3D trackers. *Proc. SPIE Vol. 1669: Stereoscopic displays and applications III*.

Cohen, J, and M Olano. 1994. Low latency rendering on Pixel-Planes□5. UNC technical report TR 94-028.

Deering, M. 1992. High resolution virtual reality. *Computer graphics*. 26:2:195.

Feiner, S., B. Macintyre, D. Seligmann. 1993. Knowledge-based augmented reality. *Communications of the ACM*. 36 (7), 53-62.

Fuchs, H, J Poulton, J Eyles, T Greer, J Goldfeather, D Ellsworth, S Molnar, G Turk, B Tebbs, L Israel. 1989. Pixel-Planes 5: A heterogeneous multiprocessor graphics system using processor-enhanced memories. *Computer Graphics: Proceedings of SIGGRAPH '89*. 23:3.

Furness, T. 1986. The super cockpit and its human factors challenges. *Proceedings of the Human Factors Society*. 30, 48-52.

Ghazisaedy, M, D Adamczyk, D Sandin, R Kenyon, T DeFanti. 1995. Ultrasonic calibration of a magnetcic tracker in a virtual reality space. *Proceedings of Virtual reality annual international symposium*. pp 179-88.

Grinberg, V., G Podnar, M Siegel. 1994. Geometry of binocular imaging. *Proc. SPIE Stereoscopic displays and VR systems*. February. Vol. 2177.

Hodges, L, and E Davis. 1993. Geometric considerations for stereoscopic virtual environments. *Presence* 2:1.

Holloway, Richard. 1995. Registration errors in augmented reality systems. PhD dissertation. University of North Carolina at Chapel Hill. Also UNC technical report TR95-016.

Janin, A, D Mizell, T Caudell. 1993. Calibration of head-mounted displays for augmented reality applications. *Proc. IEEE Virtual reality annual international symposium*.

Lastra, A. 1994. University of North Carolina. Personal communication.

Livingston, M, and A State. 1995. Improved registration for augmented reality systems via magnetic tracker calibration. UNC technical report TR95-037.

Longhurst, R. 1957. *Geometrical and physical optics*. Longmans. New York.

Min, P. and H Jense. 1994. Interactive stereoscopy optimization for head-mounted displays. *Proc. SPIE Stereoscopic displays and VR systems*. February.

Mine, M. 1993. Characterization of end-to-end delays in head-mounted display systems. UNC technical report TR93-001.

Mine, M, & G. Bishop. 1993. Just-in-time pixels. UNC technical report 93-005.

Oishi, Takashi, and Susumu Tachi. 1996. Methods to calibrate projection transformation parameters for see-through head-mounted displays. Presence 5:1.

Olano, Marc, J Cohen, M Mine, G Bishop. 1995. Combatting rendering latency. *Proceedings of 1995 Symposium on Interactive 3D Graphics*. Monterey, CA. April 9-12.

Polhemus. 1993. Fastrak specifications sheet and personal communication. Colchester, VT.

Robinett, W, and R Holloway. 1995. The visual display transformation for virtual reality. *Presence* 4:1.

Robinett, W & J Rolland. 1991. A computational model for the stereoscopic optics of a head-mounted display. *Presence* 1:1.

Rolland, J. P., D. Ariely & W. Gibson. 1995. Towards quantifying depth and size perception in 3D virtual environments. *Presence* 4:1.

Rolland, J. P. & T. Hopkins. 1993. A method of computational correction for optical distortion in head-mounted displays. UNC Technical Report #TR93-045.

Southard, D. 1994. Viewing model for stereoscopic head-mounted displays. *Proc. SPIE Stereoscopic displays and VR systems*. February.

State, Andrei, David T. Chen, Chris Tector, Andrew Brandt, Hong Chen, Ryutarou Ohbuchi, Mike Bajura, and Henry Fuchs. 1994. Case Study: Observing a Volume-Rendered Fetus within a Pregnant Patient. *Proceedings*

of IEEE Visualization '94. edited by R. Daniel Bergeron and Arie E. Kaufman. IEEE Computer Society Press, Los Alamitos, Calif. pp 364-368.

Sutherland, I. 1968. A head-mounted three dimensional display. *Proc. Fall Joint Computer Conference*.

Udupa, J and G Herman, Eds. 1991. *3D imaging in medicine*. CRC Press. Boca Raton, FL.