Inertial Head-Tracking

by

Eric Fuchs

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1993

Author .....
Department of Electrical Engineering and Computer Science
August 31, 1993

Certified by .....
Nathanial I. Durlach
Senior Research Scientist
Thesis Supervisor

Accepted by .....
Frederic R. Morgenthaler
Chairman, Committee on Graduate Students

Inertial Head-Tracking

by

Eric Fuchs

Submitted to the Department of Electrical Engineering and Computer Science
on August 31, 1993, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

Abstract

The purpose of this thesis is to evaluate the suitability of inertial sensors for use in Virtual Environment and Teleoperator head-tracking applications. Many other technologies, including optical, magnetic, acoustic and mechanical, have been used to track head position and orientation, but they all suffer from limitations such as delay, inaccuracy, limited working volume, vulnerability to electromagnetic interference, line-of-sight requirements, and high cost. Inertial components such as accelerometers, angular rate sensors and gyroscopes have been largely neglected, perhaps because of the severity of the drift problem when integrating linear accelerometers to get position. However, this author believes that an aided inertial system or a hybrid system may be able to bypass most of the limitations that afflict current head-tracking technologies. As a first step in evaluating this possibility, an inertial orientation tracker is designed, implemented, and evaluated. The system achieves 0.1 ms lag, 0.008° resolution and 0.5° accuracy over virtually any size working volume.

Thesis Supervisor: Nathanial I. Durlach
Title: Senior Research Scientist

Acknowledgments

This work was made possible by NASA grant NCC 2-771 and AFOSR grant AFOSR-90-0020, for which I am of course grateful. It also would not have been possible without the wisdom and gentle guidance of a remarkable advisor. Through his passion for VEs and his sense of humor, Nat has shaped a relaxed but exciting environment in which to work. He has also contributed unendingly to the writing of this manuscript and the ideas within. His PRESENCE is always enlightening and very human.

It is necessary to thank other people in the lab. Officemates Walter Aviles, Barbara Shinn-Cunningham and Hong Tan provided much amusement and even some useful discussions. I especially would like to thank computer gurus Joe Frisbie and Kiran Dandekar for giving generously of their time to help me out of many a binary muddle. William Rabinowitz provided valuable advice about sensors and measurement techniques and almost all the engineering problems discussed herein.

I would like to take this opportunity to express my gratitude to my parents, who created me, sent me to college and waited patiently while I stumbled along towards graduate school. I appreciate the tremendous love and support they have given me in pursuit of my career and I love them too, although I wouldn't say it to their faces.

Finally, I wish to thank my wife, Susan, whom I met during the course of this thesis. She has made my life wonderful since I met her. She has also contributed directly to this research topic by keeping my head on track and helping me to overcome inertia. Although English is not her native language, her red marks on the manuscript were extremely valuable, entertaining, and artistic. This work, and everything I do, is for her.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Virtual Environment Technology

The recent growth in the power of computers to perform real-time 3-D graphics has led to an explosion of interest in the concept of Virtual Reality, or Virtual Environments, originally explored by Ivan Sutherland in the 1960's. A virtual environment(VE) is a type of immersive computer simulation designed to give a person the feeling of being present in some artificial world or environment. The illusion of presence is accomplished by attaching special hardware to the user which blocks stimuli from the real environment from entering the user's sense organs and presents stimuli from the virtual environment instead. Acoustically insulated earphones block any sounds from the user's true environment and present sounds generated by objects in the VE, processed to make them sound like they are coming from the correct location in 3-space. The visual analog is called a Head-Mounted Display(HMD) — a pair of display screens worn in front of the eyes which block the view of the true environment and present the images that would be seen from the locations of the user's eyes in the virtual environment. For a complete VE simulation there should also be virtual displays of touch, smell and taste, but these devices are more challenging to develop.

Applications of VEs include any pursuit in which cost and/or risk can be reduced by trying something first in a simulation to prevent mistakes, or by substituting a virtual operation for the real one altogether. The classic example is architecture:

design the building in a virtual environment and experience what it would be like to walk through the building and use it's facilities before you actually build it! Another important application area is training. Many calamities could be prevented if pilots could first attempt new maneuvers in a VE flight simulator, or surgeons could practice difficult operations in a VE before performing them on live patients. The training applications of VE technology may also include educational programs, where the goal is not to prevent costly mistakes but to take advantage of the immersive nature of the medium to teach something in a more attention-grabbing, enjoyable and unforgettable way. A third area of application is virtual teleconferencing. Instead of having participants travel around the country or the world to be physically together in a conference room or a convention center or a loved one's living room, they may someday be able to just enter VE systems at their various localities and meet together in a shared virtual world, designed appropriately for the type of work or play intended. If offices were also virtual, it would save tremendous amounts of money and eliminate rush hour and the pollution it generates. Other uses of VE technology include research on human psychophysics and psychology, research in the physical sciences where advanced simulation and data visualization are needed, and entertainment.

Having seen that VEs could be useful, we focus now on the current technological challenges in their development. Most virtual environments are primarily visual, with auditory simulation sometimes added. The basic functions in a simple visual VE are to 1) track the position and orientation of the user's head, 2) compute a stereo pair of images which depict what the user would see if the user's head were at that position and orientation in the virtual world, and 3) present those images to the user's eyes. With these three elements it is only possible to passively walk through the VE and observe it. To enhance the feeling of interacting with the VE as if present, the user's own body should be represented within it, and therefore it is necessary to track not just the head but at least some of the rest of the body (usually the hands), and also to do some sort of modeling of the objects in the world and their interactions so that the user can move them manually. All five of these tasks — head-tracking,

hand shape and position-tracking, computer modeling, computer graphics, and head-mounted display — have been accomplished to some extent but badly need further development to make a truly convincing VE simulator.

The problem of making a fast, accurate, and economical head-tracker which operates throughout a large workspace has been researched extensively but remains very challenging. Tracking the position and orientation of the hand or other body part is a closely related problem and is usually done with one of the same technologies used for head-tracking. The head-tracking problem is more fundamental because there can be no visual or auditory VE at all without it, and it is likely that the requirements are more stringent because errors in head-tracking pose a greater risk than do hand-tracking errors of leading to serious sensory-motor conflicts that would destroy the illusion of presence or even cause simulator sickness. This thesis will consider only the head-tracking problem, but it should be understood that the results may also be applied toward tracking other body parts where appropriate. The next section reviews the head-tracking solutions that appear in the literature, to provide background and motivation for this study of inertial head-tracking.

1.2 Overview of Head-Tracking Technologies

Head-trackers to date have been based on four basic technology categories: mechanical, magnetic, acoustic, and optical. Table 1.1 compares the potential of these four technologies and the proposed inertial technology according to several different specifications. The comparison is based on the intrinsic limitations of the technologies, and has little to do with the current state of the art or commercially available systems. While reading the following discussion of the four basic technologies, it may be helpful to refer to the table to make comparisons.

1.2.1 Mechanical Head-Trackers

Perhaps the most straightforward approach to head tracking is to make some kind of direct physical connection to the head, the displacement of which can be easily mea-

	Mechanical		Magnetic		Acoustic		Optical			Inertial	
	Arm	String	AC/DC Pulsed	Environmental	Time-of-Flight	Phase Coherent	Beacon	Pattern Recognition	Optical Ranging	Angular	Linear
Speed	instantaneous	?	limited by noise, filtering	?	at least 1ms per foot of range	instantaneous	limited only by video rates and matrix computation	very computation bound	?	instantaneous	instantaneous
Resolution	excellent	good	noise limited, range dependant	could be good	range / no. of quantization levels	independant of range	range dependant	?	very fine	depends on head speed	depends on head acceleration
Repeatability	excellent	fair	depends on workspace ferromagnetic stability	depends on workspace ferromagnetic stability	depends on workspace atmospheric stability	poor because of wavenumber ambiguity	excellent–as good as the resolution	excellent–as good as the resolution	excellent	poor due to drift–especially in yaw	extremely poor due to drift
Range	about 6 feet	about room size	SNR limited	unlimited	inversely proportional to desired resolution	room size	about half the linear dimensions of the room for outside-in systems, inverse to desired resolution	large	?	unlimited	unlimited
Line-of-Sight	worse yet, requires mechanical connection	worse yet, requires mechanical connection	unnecessary	unnecessary	required	required	required	not as restrictive as other optical methods	required	unnecessary	unnecessary
Interference	immune	immune	EMI, ferromagnetic distortion	distortions of earth's field by metal, stray magnetic fields	acoustic reflections, percussive sounds	acoustic reflections, narrowband ultrasonic sources	reflections, small lights or highlights	variations in lighting, shadows	?	immune	immune
Angular vs Linear Performance	depends on design	linear only	probably angular better	angular only	angular measurement depends on linear measurement	angular depends on linear, angular may be very difficult	outside-in: angular depends on linear, inside-out: angular resolution superior	?	angular depends on linear	angular only–excellent performance	depends on angular measurement, much worse drift problem
Calibration to Workspace	not necessary	not necessary	could compensate stationary ferromagnetic features	magnetic mapping could help	simple temperature compensation	?	not necessary	not necessary	not necessary	simple temperature calibration	simple temperature calibration
Complexity/Cost	low	low	low–medium	low	low	low-medium	medium-high	very high	high	low	low-medium
Multiple Tracking	not practical	not practical	must multiplex – degrades speed	no problem	must multiplex – degrades speed	probably very difficult	doable	doable	?	no problem	no problem
Other Comments	good for seated applications	may pull head	currently most popular	only 2 DOF		reflection problem particularly severe because the sound source is continuous	inside-out requires mounting multiple imaging devices on head – can be bulky				
Biggest Problems	small range, clumsy	speed + repeatability?	interference, speed, range	very workspace dependant	line-of-sight, lag increases with range	reflections, wavenumber ambiguity	outside-in: resolution-range tradeoff, inside-out complexity	computationally expensive	?	?	drift
Room to Improve	very little	not much	probably alot	not yet implemented	signal processing could make it more robust but not faster	?	fair	alot in the far future	?	not yet implemented – good potential for orientation tracking	not yet implemented – less potential than angular

Table 1.1: Comparison of various head-tracking technologies.

sured using potentiometers, optical encoders, linear variable differential transformers (LVDTs), rotary variable differential transformers (RVDTs), magnetorestrictive displacement transducers, or cable-extension transducers. Simplicity of mechanical design has dictated the two most common linkages for 6 degree-of-freedom (DOF) measurement: a single segment which extends telescopically, or two rigid segments jointed together at an "elbow." In either case, the arm is attached at one end with a 2 DOF "shoulder" joint to some fixed reference point in the room, and at the other end is attached with a 3 DOF "wrist" to the head. Compared to other trackers, these are very simple and inexpensive. The only causes of lag or inaccuracy are flexion of the linkages, backlash in the joints, noisy transducers, analog-to-digital conversion time and resolution, and digital communication to the host computer. With careful mechanical design, flexion and backlash can be reduced within acceptable limits, particularly if the arms are short. Mechanical transducers such as potentiometers and optical encoders are available with extremely good performance, and A/D conversion and digital I/O should not be problematic either.

The biggest problem with mechanical arm trackers is range. Making the arm segments longer lowers the mechanical resonance frequency, which may lead to unacceptable lag or oscillation. It also increases the inertia felt by the user's head. Even if one is willing to make such sacrifices to achieve larger range, the range of a two-segment arm is ultimately limited to about twice the distance from the user's head to the ceiling, as illustrated in Figure 1-1. In short, the mechanical tracker is very well suited for applications where the user remains seated and only needs to make head motions within a small volume. In this case, when the arms are each less than 2–3 feet long, the mass and cumbersomeness can be minimal. There is a small, lightweight mechanical head-tracker commercially available from Shooting Star Technologies for a considerably lower price than any other commercially available head-tracker[26].

Another mechanical approach to head-tracking would be to use pullboxes mounted on the wall or ceiling, with the free ends of the pullstrings attached to the head. High precision instrumented pullboxes, called "cable-extension transducers," are available at low cost from a variety of manufacturers for distance-measurement applications.

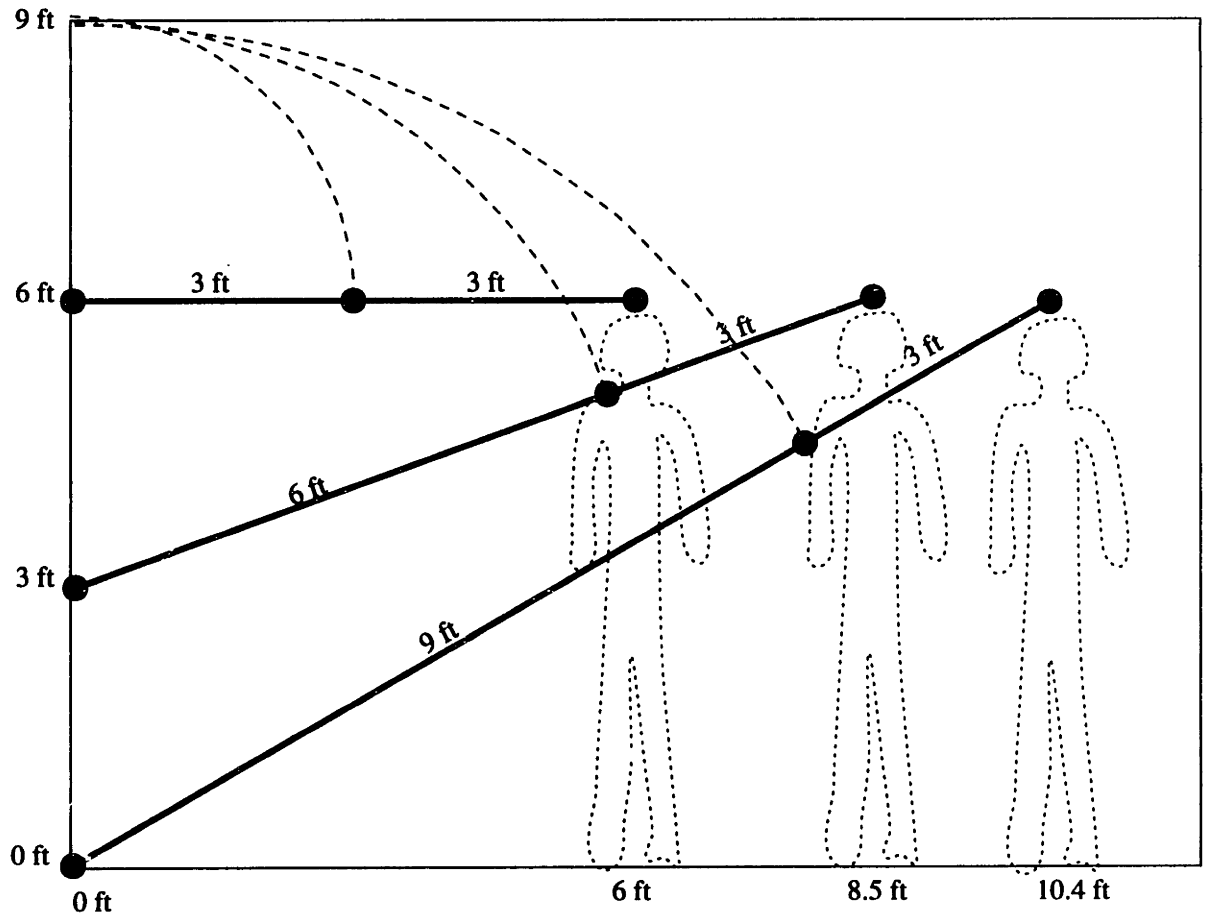


Figure 1-1: Range limitations of mechanical arm head-trackers.

There are two possible ways to measure head position using strings. The first method uses three pullboxes with the free ends of all three strings attached to a point on the head. The cartesian position of the point in space is found by triangulation. A second method would be to use a single pullstring to measure distance and also direction. The azimuth and elevation with which the string departs the pullbox could be measured by running the string through the shaft of a very light, low-friction joystick on its way out from the pullbox. The pullstring technique would not be practical for measuring orientation, but it is a very simple way to get position, and could be used as an add-on to an inertial orientation tracker. It should be noted that to get faster response requires a tighter string-retraction spring, which might result in an annoying sense of being pulled around by the head. To the author's knowledge, nobody has implemented or discussed the use of pullstrings for head-tracking, but they are included in the table as an alternative.

1.2.2 Magnetic Head-Trackers

The most common trackers by far are magnetic. Where an affordable off-the-shelf tracker is needed, the de facto standard has been the Polhemus Isotrak for several years. The biggest reason for their popularity is their comfort and convenience. The magnetic field detection coil that must be worn on the head is extremely small and light. The transmitter coil block can be mounted anywhere within range of the sensor; there does not need to be a line-of-sight between them because magnetic fields can pass right through any non-metallic objects between the transmitter and the receiver. Unlike with mechanical systems, it is possible to track multiple users in the same workspace by time or frequency multiplexing.

There are, however, many serious disadvantages to the magnetic tracking techniques currently in use. The range of the Polhemus Isotrak and its recent rival, the Ascension Technologies "Bird," is only about 2–3 feet. A new Polhemus model called the "Fastrak" is being advertised with better range and much better speed, but there are not yet any reports confirming its specifications. Because magnetic dipole fields fall off as $1/r^3$, to double the range would require an eight-fold increase in current or sixty-four-fold increase in power. Magnetic trackers are also very vulnerable to interference. Stray electromagnetic signals in the environment cause noise in the output of the tracker, which shows up as scene jitter in the VE display. In order to reduce the noise, magnetic tracking systems employ filtering which causes substantial delay. The amount of delay specified in the Ascension Technology Corporation literature [3] is 67 ms for the Bird and 135 ms for the Isotrak. This is approximately consistent with results reported in [1] and [18]. Another cause of interference is the presence of metallic objects in the workspace. The pulsed AC magnetic fields used by the Polhemus trackers invoke eddy currents in nearby conductive metals, which lead to inaccurate readings. The Ascension Technology trackers use pulsed DC magnetic fields and are therefore less susceptible to eddy currents. However, both AC and DC systems also suffer from the distortion of magnetic fields by ferromagnetic materials.

By measuring the earth's field instead of fields emmitted by coils it is possible to overcome the limited range problems of AC and DC pulsed magnetic systems.

However, it is only possible to obtain orientation. If the field were homogeneous it would only be possible to measure two DOFs, but there is a sensor available from Sensor Applications [9] which utilizes the existing inhomogeneities to get all three orientational DOFs. The company is preparing to release a very inexpensive, but slow, head-orientation tracker, and they are at this time unwilling to divulge how it works.

1.2.3 Acoustic Trackers

Acoustic head-trackers in use today measure the time-of-flight for an ultrasonic chirp to get from the user's head to three microphones at fixed locations in the environment. Multiplying by the speed of sound yields three distances which can be used to compute the point in space where the chirp originated. By having three separate ultasonic beacons fixed to different points on the head, the orientation of the head can also be computed from the locations of the three fixed points. This system is even simpler to implement than the pulsed magnetic systems, and can achieve a much greater range. In fact, ultrasonic systems can track a user throughout a room of virtually any size. However, increasing the range incurs a speed penalty. Since the speed of sound is approximately 1 ft/ms, obtaining the position and orientation of a head that is 10 feet from the most distant receiver would require at least 10 ms each to localize the three beacons, making a total of 30 ms plus computation time. The time between updates is even longer because you must wait for echos to die out before initiating a new measurement. To track multiple users requires time multiplexing, thus dividing the update rate by the number of users. The TOF trackers measure time by using counters which stop counting as soon as the microphone detects a signal of the appropriate frequency. If someone jingles keys, types, or otherwise generates sounds which contain energy at the frequency used by the ultrasonic head-tracker, the system may be fooled into thinking it has received the ultrasonic chirp and terminate the counters early, leading to gross ranging errors.

The reproducibility of ultrasonic position measurements is limited by air currents, temperature and humidity changes, and the phase distortion of an ultrasonic pulse

travelling through air. Because ultrasonic systems derive head orientation from the positions of three fixed points on the head, the angular resolution depends entirely on the translational resolution. Since the human head is rather small, the three ultrasonic beacons are usually placed at the vertices of a triangle with about 10 cm separation between them. For this separation, a positional accuracy of ± 1 mm limits the orientational accuracy to about $\pm 1°$. Consistent with this calculation, a commercial tracker made by Logitech called the 6D Mouse advertises a positional resolution of 0.5 mm and an orientational resolution of 0.5°.

An alternate approach called Phase Coherent (PC) ultrasonic tracking tries to get around the sluggishness problem by eliminating the need to await the arrival of a chirp. An emmitter continuously sends out an ultrasonic sinusoid. By measuring the phase difference between the emmitter and receiver, it is possible to measure changes in the distance between them. Distances larger than a wavelength can be measured by keeping track of how many times the phase has wrapped around past 360°. Three distances may be measured simultaneously using three separate carrier frequencies. Because the PC approach only measures relative distance, it is necessary to initialize tracking from a known reference position, and subsequent errors will accumulate. Time-multiplexing is not possible, and therefore tracking more than one position point (as you must if you want to track orientation or multiple users) requires the use of more discrete carrier frequencies. The usable ultrasonic frequency band could become crowded. PC systems are also vulnerable to acoustic interference and reflections.

1.2.4 Optical Trackers

Optical head-tracking systems tend to be complex and expensive, but in some cases offer very high performance. There are a particularly large number of different designs which use visible or infrared light in some way to track human motion. For the sake of taxonomy these can be divided into beacon-tracking systems and other strategies. Beacon-trackers can be further classified into "outside-in" and "inside-out" systems. Other strategies include pattern recognition, light polarization, various methods of

laser rangefinding, and perhaps others.

Outside-in beacon-tracking is the simplest and most common arrangement. Two or more imaging sensors (usually CCD cameras or lateral-effect photodiodes) are mounted on the walls or ceiling looking in on the workspace. The imaging devices detect the LED beacons worn on the user's head, and a computer then computes the 3-D positions of the beacons by combining the 2-D information from multiple imaging devices having different viewpoints. With CCD cameras, the digitized image frames must be analyzed to determine the locations of the brightest points, which are assumed to be the LEDs. Lateral-effect photodiodes are specialized imaging sensors which output the location of the brightest point instead of the complete image. This saves the steps of image digitization and analysis, but since they can only output the location of the single brightest point at any moment, the LED beacons must be time-multiplexed. Infrared LEDs and imaging sensors are commonly used, resulting in less interference from room lighting sources and reflective highlights.

The biggest problem with outside-in systems is the trade-off between resolution and working volume. If the cameras employ telephoto lenses, then smaller distances may be resolved, but the volume of intersection of the field-of-view (FOV) cones of the cameras will be small. With wide-angle lenses you can increase the working volume at the expense of resolution, but even if you were willing to do so the maximum working volume would still be limited by the size of the room and the field-of-view of the widest-angle lenses practical for the application. Figure 1-2 illustrates in two dimensions how the range would be limited using 70° FOV wide-angle lenses. A possible way to circumvent this trade-off might be to use telephoto lenses, but mount the cameras on servo-motorized platforms that are programmed to track the target. This type of system could track only one head at a time, of course, and would represent a great increase in complexity, which may explain why nobody has tried it. Like acoustical head-trackers, outside-in systems determine head orientation by first measuring the position of three beacons at fixed positions on the user's head. If the positional resolution has already been compromised to obtain enough range, the angular resolution will likewise suffer. To give an idea of the performance available, a

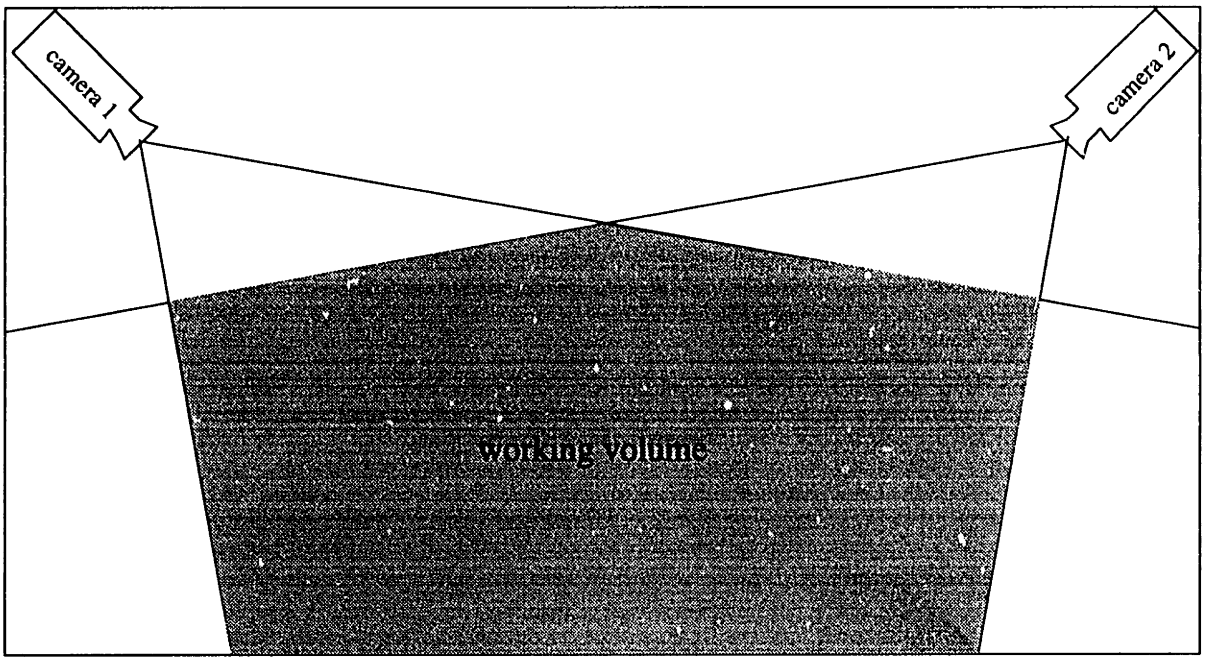


Figure 1-2: Range limitations of optical outside-in head-trackers.

current commercial system, the SELSPOT II, offers 0.25 mm translational resolution and 5 mm translational accuracy when tracking over a 1 m^3 working volume [11].

Another approach, which has been aggressively pursued at the University of North Carolina, is inside-out beacon tracking [27]. The cameras are worn on the head pointing out towards the ceiling, which is covered with infrared LED beacons. This technique is complex and expensive, loads the user's head with 4 CCD cameras, and requires the installation of a special ceiling containing hundreds of precisely positioned LEDs. However, it is one of the highest performance tracking systems currently in existence. The translational resolution is the same as what would be achieved by an outside-in system using the same telephoto lenses, but it is possible to extend the range to any size by increasing the size of the ceiling array while still using telephoto lenses. The really important advantage is in orientation tracking, which becomes extremely sensitive when the sensors are head-mounted. The UNC tracker, which was demonstrated at SIGGRAPH '91, has a 10' X 12' scalable work area, 30 ms lag, 2 mm positional resolution and 0.2° angular resolution.

Other optical approaches to head-tracking, though not yet in widespread use, may someday offer methods which don't require the user to wear anything special or don't

restrict the user's range. With future advances in computer vision it may be possible to rely on pattern recognition to track the users' bodies directly rather than using beacons. A concept proposed by Gary Bishop [27] is to use integrated vision chips mounted on the head to sense motion through space — even unstructured space such as the outdoor environment. Another technique which has been used is to scan a laser beam through a space in which there are reference sensors on the wall as well as movable sensors attached to a person. The location of the worn sensors can be derived from the time at which the beam strikes them. Lasers may also be used for rangefinding using a technique related to interferometry [25] or triangulation [2].

1.3 Inertial Head-Tracking

You may notice that all of the trackers described above make a measurement of head-position relative to some fixed equipment in the room. This is why they all have range limitations. However, it is also possible to make measurements that are not relative to anything except an inertial reference frame or some large scale environmental field such as the earth's magnetic or gravitational field. This is the technique used in Inertial Navigation Systems (INS), which have a long history of success in guidance and navigation of ships, planes, missiles and spacecraft — all of which require accurate position and orientation information as they travel over long distances.

The application of inertial techniques to the VE head-tracking problem has not been widely discussed in the literature. CAE has made a very high fidelity optical tracker which uses accelerometers and angular rate sensors for quickening purposes, and Gary Bishop at the University of North Carolina has proposed a similar system, but neither of these systems rely heavily on inertial data as do INS systems. The reason may be that despite the potential advantages, the difficulties seem too daunting. This section will briefly explain the operation of an INS, then examine the advantages and difficulties and propose some approaches to capitalize on the advantages while getting around the difficulties.

1.3.1 INS Principles of Operation

The position and orientation of an object in space can be described by a 6-dimensional state vector whose elements are the 3 cartesian coordinates x, y, and z, describing position of the object's center of mass (C.O.M.), and three euler angles sometimes referred to as yaw, pitch and roll, describing three rotations about axes through the C.O.M. necessary to get the object from an orientation aligned with the axes of the cartesian coordinate frame to the current orientation of the object. These six state elements as functions of time are the desired output of a head-tracker. They can be obtained directly, as they are in the tracking systems currently in use, or they can be obtained by measuring their derivatives (velocities) and integrating once, or by measuring their second derivatives (accelerations) and integrating twice. Due to Gallilean invariance, it is impossible to measure the linear velocities with any self-contained head-mounted sensor, but the accelerations can be measured with a device called an accelerometer which measures the force necessary to keep a small test mass moving together with the head. The angular velocities can be directly measured by a self-contained sensor because steady rotation produces measurable centripetal and Coriolis forces. The traditional sensor for angular velocity is called a rate gyroscope.

There are two strategies used in INS. The original method is to build a stable table, which is a platform connected to the vessel by motorized gimbals which act to keep the orientation of the platform constant even as the vessel rotates, using angular velocity signals from table mounted gyroscopes. Accelerometers are mounted along three orthogonal axes on the stable table. Since the direction of the accelerometers never changes, it is possible to independently double integrate the three accelerometer outputs to obtain x, y, and z. The pitch, roll and yaw of the vessel are simply measured from the gimbal angles. This technique is conceptually simpler and requires no computation, but it does require sophisticated gyroscopes, servomotors, gimbals and control electronics to maintain the orientation of the stable table with high accuracy for long journeys.

A more recent approach is called Strapdown INS. The accelerometers and rate gyros are simply strapped down to the body of the vessel. The orientation of the

vessel cannot be simply read off from some gimbals; it must be obtained by jointly integrating the outputs of the rate gyros. Since rotations in 3-space do not commute (e.g. [14]), the angular velocities output by the three orthogonal rate gyros cannot be integrated independently. A computer must be used to maintain the vessel's state and compute the increments to the state when it is necessary to update the state. When nonzero signals are detected from the rate gyros, the computer increments the angular state elements, making use of the angular state at the moment the nonzero angular rates were detected. Likewise, when nonzero signals are detected from the linear accelerometers, the computer increments the linear velocity state based on a knowledge of the orientation at the moment the accelerations were measured. This approach requires much less hardware, but the integration needs a computer.

Inertial navigation systems of either type operating within a gravitational field will be unable to detect the acceleration of the vessel separately from the acceleration of gravity. That is, if the actual acceleration that the vessel is undergoing is $\vec{a}$ and the acceleration of gravity is $\vec{g}$, then the acceleration measured by the INS's triaxial accelerometers will be $\vec{a}_{measured} = \vec{a} + \vec{g}$. To obtain the position it is desired to double integrate $\vec{a}$ only, so it is important to know the direction and magnitude of $\vec{g}$ relative to the vessel at all times so that it may be subtracted before the integration. Detailed information about INS is available in many books on the subject, such as [8, 24, 6].

1.3.2 Advantages

If it were possible to attach an inertial navigation system to a person's head it would be possible to track the head with undiminished performance over an unlimited range or working volume. This is a significant advantage over all other tracking technologies, which cannot increase the working volume without degrading the resolution because of limited SNR. The limited working volume of other systems is a significant problem which rules out certain applications. For example the sense of presence in an architectural walkthrough might be greatly enhanced if the user could explore the virtual building by walking around naturally instead of by flying around the virtual space while remaining seated in the non-virtual world.

A second important advantage of inertial head-tracking is that it can be very fast. The integrated outputs of angular rate sensors and accelerometers are ready to be used without any further processing and contain no delays from filtering or finite measuring time. The only other head-trackers which have essentially instantaneous response are mechanical trackers, which suffer from very strict range limitations. Speed, or responsiveness, is known to be a very important aspect of tracker performance. Excessive delay added into the head-motion-to-visual-feedback loop destroys the illusion of presence and can cause simulator sickness [22]. How much of each type of delay is acceptable is an important consideration for designers of virtual environments and an analysis of this problem will be included in the next chapter.

A third advantage of INS techniques for head-tracking is that they are free of the interference and line-of-sight problems that plague all the other trackers. A pure inertial system does not send or receive any signals from its environment and therefore its performance will not be affected by any kind of EMI or occlusions of acoustic or optical sources, or the operation of any other tracking systems in its vicinity. The only external field that an INS senses is gravity, and gravitational interference is not a problem because the movement of mass in the vicinity of the tracker produces gravitational variations so much tinier than the Earth's gravitational field that they will have no effect on the accuracy of the INS. Therefore, inertial trackers are the only ones which provide perfect "sociability" as defined by Meyer, Applewhite, and Biocca [21].

One final advantage that is worth mentioning has to do with a different design approach to the head-tracking problem that comes about when using inertial techniques. In optical and acoustic trackers, the fundamental capability is to measure position. To obtain the orientation of the head, the tracker must measure the positions of three fixed points on the head and then calculate the orientation from them. Therefore, the angular resolution is limited by the uncertainty in the position measurements as well as the distance between the three fixed points on the head. If we make a realistic assumption that the spacing between the points on the head can't be made much more than about 100 mm, then a positional jitter of ± 1.0 mm causes an orientational jitter

of $\pm 1.1°$. Inertial systems, on the other hand, measure the angular degrees of freedom directly, so that it is possible to make a self-contained orientation-tracking module whose performance is in no way affected by the quality of the position-measuring subsystem, or even the lack thereof. The existence of such independent orientation and position subsystems would make possible a more modular approach to VE system design. The designer would be able to select an orientation-tracker and a position-tracker with appropriate specifications for the application. In an extreme case, such as a VE in which the user navigates by making hand gestures and not by walking about, it may be perfectly workable to use no position-tracker at all.

1.3.3 Difficulties

Until recently, the foremost problem with using an INS system to track a human head was probably weight. A traditional navigation gyroscope is at least the size of a soda can and much heavier. Mounting three such gyroscopes plus all the other stuff on the head would be an ergonomic nightmare. Furthermore, they are extremely expensive. In the past year two new devices have come to the market which radically change this picture. Gyration, Inc. in Saratoga, California has announced a miniature plastic spin gyroscope called the GyroEngine which weighs 1.2 oz. and is the size of a film canister. Because it is a gimballed spin gyroscope, one can measure two degrees of freedom, so only two are needed on the head. Systron-Donner, Inc. in Concord, California has introduced a product called the GyroChip. It uses a vibrating quartz tuning fork to detect angular velocity dependent Coriolis forces and convert them into electrical signals through the piezoelectric property of quartz. Both of these products cost about $3000.00 for a three axis system in unit quantities, and promise to become much cheaper in mass production.

A second difficulty with using gyroscopes for head-orientation tracking is drift. All spin gyroscopes will have some tendency for the spin axis to gradually drift away from it's original orientation in space due to friction in the gimbals, acceleration, etc. Angular rate sensors will have some output bias, which when integrated to get angular position appears as a steady drift. In quartz angular rate sensors the bias is

affected by temperature, acceleration and pressure. The effect of drift in a VE is that the virtual world will appear to gradually rotate about the user's head. The question of how slow this rotation must be in order not to be noticed will be addressed in Chapter 2. In an opaque VE with no physical interaction with the real world, a slow rotation in yaw that cannot be noticed is acceptable. A slow drift in pitch and roll will probably not be noticed either for a while, as the user's head will probably tilt gradually to compensate for it. However, when a significant tilt has accumulated, proprioceptive and vestibular cues will alert the user, so tilt must not be allowed to grow unbounded. How much tilt can go unnoticed will also be explored in Chapter 2.

Difficulties with calculating position from the output of accelerometers are much more severe. Whereas the bias in the output of an angular rate sensor, when integrated, leads to an orientation error that grows linearly with time, the output bias of an accelerometer, when double integrated, leads to a position error that grows quadratically with time. Under certain conditions it may be possible to null the input bias so perfectly that even under quadratic growth the error won't explode for a very long time. However, there is inevitably some random noise produced by the accelerometer in addition to it's bias. Let us analyze what becomes of this noise when it is integrated. Let $f(t)$ be a zero-mean unit white noise, which we pass through an integrator to get the output

$$x(t) = \int_0^t f(u)\,du.$$

The integrated noise signal, $x(t)$, is known as a Wiener Process or Brownian Motion Process and has expected value

$$\begin{aligned} E[x(t)] &= E\left[\int_0^t f(u)\,du\right] \\ &= \int_0^t E[f(u)]\,du = 0 \end{aligned}$$

and autocorrelation

$$R_{xx}(t_1, t_2) = E[x(t_1)x(t_2)]$$

$$
\begin{aligned}
&= E\left[\int_0^{t_1} f(u)\,du \int_0^{t_2} f(v)\,dv\right] \\
&= \int_0^{t_2}\int_0^{t_1} E[f(u)f(v)]\,du\,dv \\
&= \int_0^{t_2}\int_0^{t_1} \delta(u-v)\,du\,dv \\
&= \begin{cases} t_1, & t_1 \le t_2 \\ t_2, & t_1 \ge t_2 \end{cases}.
\end{aligned}
$$

The mean square value is

$$E[x^2(t)] = R_{xx}(t,t) = t,$$

so the standard deviation, σ_x, of the output increases as $\sqrt{t}$. This is the ultimate limitation on the performance of angular rate sensors integrated once to obtain angular position. It is impossible to null out the bias perfectly, and no matter how small the bias, it leads to an error that grows as t, which will eventually catch up to the Brownian Motion error that grows as $\sqrt{t}$. Thus, for the short term, the performance of a well-calibrated system will be limited by stochastic fluctuations with $\sigma_x \propto \sqrt{t}$, but in the long term the drift will be dominated by a deterministic linear growth.

Let us now go on to see what happens if we pass the noise source through a second stage of integration, as would be necessary if it were an accelerometer:

$$y(t) = \int_0^t x(u)\,du.$$

As before,

$$E[y(t)] = \int_0^t E[x(u)]\,du = 0.$$

To find the mean-square output of the second integrator, we will make use of the autocorrelation, R_{xx}, at the output of the first integrator, which we computed above:

$$
\begin{aligned}
E[y^2(t)] &= E\left[\int_0^t x(u)\,du \int_0^t x(v)\,dv\right] \\
&= \int_0^t\int_0^t R_{xx}(u,v)\,du\,dv \\
&= \int_0^t \left[\int_0^v R_{xx}(u,v)\,du + \int_v^t R_{xx}(u,v)\,du\right] dv
\end{aligned}
$$

$$
\begin{aligned}
&= \int_0^t \left[\int_0^v u\,du + \int_v^t v\,du \right] dv \\
&= \int_0^t \left[v^2/2 + vt - v^2 \right] dv \\
&= t^3/3 \qquad (1.1) \\
\sigma_y &= t^{3/2}/\sqrt{3} \qquad (1.2)
\end{aligned}
$$

Once again we find that the RMS error due to noise grows a factor of $\sqrt{t}$ more slowly than the deterministic drift. However, in this case they both grow faster than t, which makes the problem of obtaining position from accelerometers much more sensitive than the problem of obtaining orientation from gyroscopes.

1.3.4 Hybrid and Aided Inertial Systems

The construction of a pure INS to solve all the VE head-tracking problems is likely to be too difficult. There are alternatives, however, which can possibly lead to a workable system that retains at least some of the advantages of the INS approach. If the inertial system basically works, but produces intolerable amounts of drift, we may try to curtail the drift by periodically resetting the system with values obtained from other sensors. For the rotational degrees of freedom it is possible to correct drift without reference to any equipment in the room, and therefore without loss of the advantages of being self-contained. For a static body it should be possible to measure the orientation just by measuring the direction of $\vec{g}$ and of the earth's magnetic field, $\vec{H}$. Measurement of $\vec{H}$ can be accomplished using a three-axis fluxgate magnetometer [20]. The frequency response of a fluxgate compass is very limited, which is why it cannot replace the yaw gyroscope altogether. For a static head, measuring $\vec{g}$ is easy: $\vec{g} = \vec{a}_{measured}$. The human head is often quite still for periods of several seconds, but almost never undergoes uniform acceleration for that long. Therefore, if the measured acceleration is a constant for several seconds, and, as an additional check $|\vec{a}_{measured}| = |\vec{g}|$ during that time, it is very probable that the head has been still, so the pitch and roll may be corrected to make that value of $\vec{a}_{measured}$ point down. In practice, resetting the sensor biases suddenly would subject the user to "virtual

earthquakes." A simple way around this would be to ramp the biases back to zero over a timecourse of several seconds. While this may work adequately, to get the best possible performance, in the minimum mean square error sense, from a given combination of sensors, their outputs should be integrated together by a Kalman filter. This is what is typically meant by the term "aided inertial navigation." There is an excellent textbook on the subject by Brown and Hwang [7].

For position-tracking there is no way to aid the accelerometers with any other self-contained sensor. This is particularly unfortunate because position-tracking is the half of the problem that poses the far greater challenges for pure inertial approaches. If inertial position tracking had to be abandoned in favor of an ultrasonic or optical system, much would still have been gained by tracking the orientation with an inertial subsystem. First of all, the responsibility for orientation-tracking would have been offloaded from the ultrasonic system. This means that with less hardware it would be possible to collect position data at three times the rate, and be spared the geometrical calculations. This would increase responsiveness significantly. Phase-coherent ultrasonic tracking is under development which may offer improved responsiveness and immunity to acoustical interference [21]. PC ultrasonics cannot be time-multiplexed for the purpose of tracking three independent points, so orientation-tracking may be a problem. A PC position-only tracker combined with an inertial orientation-only tracker might make an excellent hybrid for many applications. An even more exciting possibility is to use a set of accelerometers aided by a cheap, low data rate ultrasonic system. The ultrasonic system only needs to produce a reliable position fix every few seconds, so it can be heavily filtered to remove noise caused by other acoustic sources and even brief obstructions of the line of sight. Furthermore, it can operate over an large room-sized range since fast TOF measurement is not required.

1.4 Thesis Scope

As argued in the previous section, the part of INS most likely to succeed in head-tracking is the orientation-tracking subsystem. There is little point developing an inertial position-tracker if the orientation-tracker can not work. Therefore, the first step in evaluating the usefulness of INS techniques for head-tracking is to build and evaluate an inertial orientation sensor. This thesis discusses the design, implementation and evaluation of such an orientation-tracker. Chapter 2 discusses the design phase of the project, starting with some analysis of the problem and the literature to determine what tracker performance specifications are desired, and ending with a presentation of the design that was chosen to best meet these specifications. Chapter 3 concerns the implementation of the project and includes detailed descriptions of the hardware and software used. Chapter 4 discusses the experiments done to evaluate the performance of the orientation tracker, the reasons for choosing these particular experiments, the results, and what other experiments might be done to yield additional information. The final chapter interprets the results and lays out directions for further work.

Chapter 2

Design

This chapter is about the process of designing an inertial head-orientation tracker. The actual design process was highly convoluted; all sorts of solutions including various hybrid combinations of acoustic, inertial, mechanical and optical technologies were considered simultaneously. New sensors of all these types were unearthed and contemplated at the same time that the desired specifications were formulated. Decisions were made based on incomplete information, and inertial angular rate sensors gradually emerged as the preferred technology based on an unquantifiable interaction between such factors as price, availability, advertised specifications, rumored specifications and an incomplete but evolving understanding of psychophysical and ergonomic human factors. This chapter, however, presents the design process in the order in which it should have occured. First an attempt is made to define the desired specifications for the head-tracker, drawing on psychophysical information where it is available and guessing where it is not. Next, the specifications of candidate sensors are presented and compared to the desired specifications, and a selection is made. Finally, a system is designed around these sensors.

2.1 Desired Specifications

Some of the important specifications for head-trackers, such as range, speed, accuracy, resolution, and sensitivity to interference, have already been mentioned in the

overview of existing tracking technologies in Chapter 1. However, it is very difficult to generate a comprehensive list of the relevant specifications needed to characterize the performance of any type of tracker and define the meaning of those specifications clearly. A second, even more difficult problem is to figure out what would be an acceptable level of performance for each of those specifications for every application. To tackle both of these challenges would be a suitable subject for an another entire thesis. This section will only make a cursory attempt at answering these two questions.

2.1.1 What are the Relevant Specifications?

A person's head position and orientation at any moment can be expressed as a six component state vector,

$$\vec{v}(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \\ \phi(t) \\ \theta(t) \\ \psi(t) \end{bmatrix},$$

where x, y, and z are the cartesian coordinates of some particular point on the head, and ϕ, θ, and ψ represent the yaw, pitch, and roll rotations respectively about that point. An ideal head-tracker is a device whose output, $\vec{w}(t)$ exactly equals $\vec{v}(t)$ for all t, no matter how the user's head moves, and which does not distract, annoy, or restrict the user. Any practical device falls short of this ideal in a variety of ways. In designing a tracker, one would like to first make a list of specifications which completely characterize how the tracker differs from an ideal tracker, then decide how much performance is desired in each of these specifications and design the tracker accordingly. The list of specifications has to be specific to the type of tracker being considered. For example, "minimum distance from nearest metallic object" is a very important specification for magnetic trackers, but is irrelevant to mechanical trackers. For this reason, only specifications needed to characterize an inertial orientation tracker are considered in this section. A further difficulty is finding

a set of specifications which are orthogonal. Many of the specifications are closely related. For example delay, drift and noise could be considered as three specific kinds of inaccuracy, but it would not suffice to just state the desired accuracy. Therefore, no attempt is made to find a minimal set of mutually exclusive specifications and define them precisely. Instead, an extensive list of loosely defined specifications is provided, which together provide an approximate but fairly complete characterization of tracker performance.

It is helpful to break the specifications down into those which describe the range of input motions, $\vec{v}(t)$, that the system must be able to track, and those which describe the kinds and amounts of imperfections in the output $\vec{w}(t)$ that will be considered tolerable. The first category has to do with the nature of human head motion, and the second category has to do with human perception.

In the first category, specifications on the input, the most obvious ones are the ranges of all six components of $\vec{v}(t)$ as well as its first and second derivatives. Lumping these together according to cartesian coordinates versus angular coordinates, the complete list consists of **translational range, angular range, maximum velocity, maximum angular velocity, maximum acceleration,** and **maximum angular acceleration**. These are not however, the only things we need to know about head motion. It would also be useful to know the **head-motion bandwidth**, which may depend on amplitude of head oscillation as well as other factors if the head-movement system is nonlinear. It would also be valuable to have a statistical description of head motion for the application of interest. If the tracker were going to use stochastic signal processing, a very complete probabilistic model of head motion would be needed. Short of this, it might be very useful to know certain things like how frequently very violent motions occur and how long they last. Since the tracker design in this thesis relies on head-motion pauses to obtain orientation updates from auxilliary sensors, an important specification is **maximum time between quarter-second head pauses**. Finally, input specifications include environmental factors like **operating temperature,pressure and humidity ranges**.

Output specifications basically concern accuracy, resolution, delay, drift and noise,

but these things are tricky to define. The classical definition of accuracy for static measurements, w, of an underlying physical quantity, v, is $\max |w-v|$. Since the head-tracker measurement is time-varying, we will define **static accuracy** as the maximum measurement error obtained when the tracker is at rest, i.e. when $\vec{v}(t) = \vec{v}$. This accuracy may depend on $\vec{v}$, and it is intentionally left ambiguous whether "static accuracy" refers to the worst case or is to be thought of as a function of $\vec{v}$. **Resolution** is the smallest change in the input that can be detected, and depends either on the **noise**, which is the amount of rapid random jitter of the output when it is supposed to be still, or on the number of quantization levels that the system uses to encode $\vec{w}(t)$. The accuracy can be no better than the resolution, because the measurement errors result from the combination of the bias and the irreproducible deviations of the measurements about the mean, the magnitude of which is specified by the resolution. If the bias does not vary with time, and if there is another more accurate measurement tool available, the bias could, in principle, be completely mapped out and subtracted from the measurements, yielding accuracy as good as the resolution. In practice this is often not worth doing. Although the resolution provides some information about the magnitude of the output noise, it may be quite useful to the VE system designer to know more about the nature of the noise. The **noise power spectral density** provides the most complete characterization generally available, and its definition can be found in [4]. **Drift rate** is only a meaningful specification for those degrees of freedom which don't have any stringent accuracy requirements, because any rate of drift will eventually lead to inaccurate readings. Where large discrepencies between $\vec{v}(t)$ and $\vec{w}(t)$ are tolerable, the drift rate is used to specify the amount of discrepancy allowable between $\dot{\vec{v}}(t)$ and $\dot{\vec{w}}(t)$. In inertial systems, the drift results from the integration of sensor noise and bias. Since the integrated bias is steady and can easily be compensated, while the integrated noise is random, it is often necessary to consider these two kinds of drifts separately. One can tell which kind of drift a specification refers to by the units: degrees/sec for a bias and degrees/$\sqrt{\text{sec}}$ for a random drift. Delay is the most abused specification, often given by tracker manufacturers as the only information about the dynamical response of the system.

There are a variety of different features of the dynamical response which have the general effect of delaying the output. The most straightforward is called "transport delay" or simply **lag**: $\vec{w}(t) = \vec{v}(t - t_{\text{lag}})$. This corresponds to a linear time-invariant (LTI) system with all-pass frequency response and linear phase. Another feature of an LTI system function that tells something about delay is **tracker bandwidth**; any causal bandlimited system must impose some delay. Lag and bandwidth alone do not sufficiently characterize the dynamical response of a system. If the system is anywhere close to linear, it would be most helpful to have the complete **frequency response** from which it is easy to compute the **group delay** for all the frequencies within the input bandwidth. Another specification often mistaken for lag is the update rate or **sampling period**. While the average lag is at least half a sampling period, it may be many sampling periods. In fact, the update rate, according to the Sampling Theorem, needs to be at least twice the bandwidth, and therefore need not be specified if the bandwidth is specified and the system is assumed LTI.

2.1.2 What Performance is Desired for the Relevant Specifications?

This section tackles the difficult challenge of answering all the questions set forth in the previous section. As mentioned above, the answers to the questions about head-tracker input specifications should be found by considering human head dynamics and biomechanics, and the answers to the questions about output specifications depend on the perceptual dynamics of the visual-proprioceptive sensorimotor loop. There is very little information in the literature in a form that can be readily applied to answering these questions. Where relevant biomechanical or psychophysical results could be found, they are used. In a few cases, simple experiments are performed to get approximate answers. In many cases, rough order-of-magnitude estimates are obtained through thought experiments or educated guessing.

Throughout this section, it is important to remember that the desired specifications for the head-tracker depend very much on its intended applications. Since the

head-tracker design in this thesis only tracks orientation, it is useful primarily for opaque "fly-through" applications. This is a large and important class of applications in which the user wears an HMD which completely blocks perception of the real surroundings, and navigates through the VE by pointing or looking in a particular direction and informing the computer of a desire to fly in that direction. In such applications, there is no interaction with the real world or motion through it, and it is therefore unnecessary to track head translation. On the other hand, the construction of an inertial orientation-tracker is really just the first step towards the goal of a fast, long-range 6 DOF head-tracker, which is badly needed for walk-through applications with both opaque and see-through HMDs. In opaque walk-throughs, the user translates through the virtual world by translating through the real world, i.e. by walking around the room. There may be haptic interaction with real-world objects such as furniture and doors which are in spatial registration with virtual models of same. In a see-through VE, the HMD uses semi-silvered mirrors to superimpose images of virtual objects onto the natural view of the real world. Precise spatial registration between the real and virtual worlds must be maintained. The demands on the head-tracker are quite different for these three applications, which will hereafter be referred to as No Interaction (NI), Haptic Interaction (HI), and Visual Interaction (VI) with the real world. Some of the specifications will be different for these three types.

Head Dynamics

Characterizing the range of head-motion inputs would be quite easy if a really good head-tracker were already available. As this is just the design phase for the inertial tracker, other methods will suffice. Desired translational range for a large-range tracker is , by definition, large, meaning room-sized. The angular range was measured with a simple protractor attached to a subject's head. Looking "way up" yielded a pitch of $+50°$, and "way down" was $-70°$. Roll was $\pm 35°$. Yaw range should be $\pm 180°$ to allow the user to face any direction. Linear velocity and acceleration ranges are rather application dependant, but specifying them isn't too critical for the design of the orientation-tracker, except that linear accelerations may have a

		Yaw		Pitch		Roll	
		range	avg	range	avg	range	avg
Moderate	EF_{mht}	368–685	489	193–372	285	255–333	307
	WA_{mht}	519–528	524	194–275	235	317–347	332
	WA_{ars}	500–727	612	242–313	282	242–283	263
Violent	EF_{mht}	519–893	689	213–473	362	310–519	392
	WA_{mht}	730–775	753	382–510	458	347–379	363
	WA_{ars}	741–1004	892	368–393	380	454–574	503

Table 2.1: Peak head angular velocities in degrees/second.

slight effect on angular rate sensor biases. Velocities corresponding to a fast sprint, and accelerations of a few g's should be sufficient. These parameters will have to be measured more rigorously before the design of an inertial position tracker using accelerometers. Maximum angular velocity is a critical parameter for a system which utilizes angular rate sensors. As will be discussed in Section 2.2 on sensor selection, there are angular rate sensors available from different manufacturers with angular velocity ranges of $\pm$ 90, 100, 300, 500, and 1000 degrees/sec. It is necessary to measure maximum head angular velocity accurately enough to distinguish which range would be most suitable. To do so, an experiment was done using the ADL-1 mechanical tracker from Shooting Star Technologies [26], connected to a PC programmed to record maximum pitch, roll and yaw rates during 6-second head-motion trials. Two male subjects, EF and WA, each underwent several trials for each of two conditions: moderate and violent head motions. Additional testing was done on WA using an angular rate sensor (ars) instead of the mechanical head-tracker (mht). The results are tabulated in Table 2.1. The results in the "violent" category represent the most rapid head rotations that the subjects were capable of. Such angular rates may be rare or nonexistent in any normal VE application, especially if the user must wear a heavy HMD, but they do provide stringent upper bounds that should be used to design the most robust head-tracker possible. Maximum angular accelerations are not very relevant to the orientation-tracker design based on angular rate sensors, but for completeness they are reported in [11] as approximately $6000°/\text{sec}^2$ for yaw rotations.

Head-motion bandwidth is a rather important specification for designing a head-tracking system, because the bandwidth of the tracker system function, $\vec{H}(f)$, only needs to be as large as the bandwidth of the mechanical input, $\vec{V}(f)$. To actually measure the relevant head-motion bandwidth would be fairly straightforward. A subject would be instructed to carry on normal activities inside a VE for a period, T, of several minutes, while a head-tracker logged a complete record of head position and orientation, $\vec{v}(t)$, for $0 \leq t \leq T$. The input spectrum could then be computed according to the relation

$$\vec{V}(f) = \mathcal{F}\{a(t)\vec{v}(t)\} = \begin{bmatrix} \mathcal{F}\{a(t)x(t)\} \\ \mathcal{F}\{a(t)y(t)\} \\ \mathcal{F}\{a(t)z(t)\} \\ \mathcal{F}\{a(t)\phi(t)\} \\ \mathcal{F}\{a(t)\theta(t)\} \\ \mathcal{F}\{a(t)\psi(t)\} \end{bmatrix} = \begin{bmatrix} A(f) * X(f) \\ A(f) * Y(f) \\ A(f) * Z(f) \\ A(f) * \Phi(f) \\ A(f) * \Theta(f) \\ A(f) * \Psi(f) \end{bmatrix},$$

where the smooth windowing function $a(t)$ has been used to prevent the sharp transitions of $\vec{v}(t)$ at $t = 0$ and $t = T$ from contributing undue high-frequency energy to the spectrum. Instead of setting up the apparatus to actually perform this measurement, an approximation can be made based on some plausible assumptions. Modelling the head as a mass which moves as a result of forces generated by muscles, it seems reasonable to assume that its motion is acceleration-limited. Consider sinusoidal oscillations in yaw of amplitude A:

$$\phi(t) = A \sin 2\pi f t.$$

Differentiating twice to obtain yaw acceleration, and setting the amplitude equal to the maximum yaw acceleration stated above,

$$\begin{aligned} \ddot{\phi}(t) &= -A(2\pi f)^2 \sin 2\pi f t \\ |-A(2\pi f)^2| &= 6000°/\text{sec}^2 \\ A &= \frac{152°/\text{sec}^2}{f^2}. \end{aligned}$$

At 20 Hz, the head oscillation amplitude could be at most ±0.38°, assuming that the ability of the muscles to accelerate the head does not diminish at all at high frequencies. Since the power of the muscles probably also falls off rapidly with frequency, the amount of head motion energy above 20 Hz is probably extremely minute, and so 20 Hz will be used as the input motion bandwidth.

The last specifications on the tracker input that need to be considered are time between head pauses and temperature range. Observing a busy person with a stopwatch, the maximum time between obvious head pauses appears to be on the order of 5–10 seconds. As will be discussed in Section 2.2, the fluid inclinometers need about a quarter second pause to obtain a good reading, and the "obvious" head pauses in this observation experiment are usually much longer than that. The operating temperature requirements on a tracker to be used in an indoor laboratory environment are not too demanding, and a temperature in the range of 70–85° F can be assumed.

Perceptual Dynamics

The psychophysical requirements on head-tracker performance are considerably less obvious. Even the simplest specification, static accuracy, is very complex. The accuracy requirements for VI applications are most difficult to achieve, but easiest to formulate. From this point on, the assumption will be made that the display has perfect resolution, to simplify the analysis and to guarantee a tracker design that will not be obsoleted by the next generation of HMDs. Under this assumption, the accuracy for VI applications must be high enough that any misregistration between the virtual representation and the real world would not be noticed. Given that the human eye's acuity in resolving two parallel lines as separate is about 25 arc-seconds [5, p. 199], and assuming the nearest objects in the visual scene are at a distance of 25 cm, the required accuracies would be on the order of 0.03 mm for translation and 0.009° for orientation. Registration for HI applications requires that when contact is seen between the virtual hand and a virtual object, contact is felt between the real hand and the corresponding real object. This places a bound not on the head-tracker error, but on the sum of the head-tracker error and the hand-tracker error. Since

visually determining the moment of contact between two different objects is more difficult than simply determining whether two matching objects are exactly superimposed, and is highly dependant on the point-of-view from which the objects are seen, this bound on tracker error will generally be looser than in the VI case. In NI applications no registration between the real world and the virtual world is required, with the exception that gravity from the real world will still be felt by the user in the virtual world. This implies that no accuracy at all is required in position and yaw, while the accuracy required in pitch and roll is determined by the ability to adapt to steady discrepancies between self-orientation sensed visually and self-orientation sensed by the vestibular and proprioceptive senses. Based on a study of the perception of the orientation of a bar when the visual field is rotated [5, p.1150], subjects are not aware of any visual-proprioceptive conflict until visual tilt exceeds 15°.

It has been argued that NI applications don't impose any absolute accuracy requirements on the head-tracker, and therefore drift in the tracker output is alright if it is so slow that a stationary user does not perceive that the environment is moving. Since visual motion detection thresholds for observer-relative motion and long-term exposure are about 3 arc-minutes/sec [5, p. 928], the scene may rotate about the user at up to 3°/minute without appearing to move. Again assuming the closest object is at a distance of 25 cm, the maximum linear drift rate is 1.2 cm/minute transverse to the direction of gaze, and higher in the direction of gaze. The yaw angle may drift at 3°/minute ad infinitum, but the pitch and roll are limited to a maximum error of 15°, and must therefore be corrected after 5 minutes of drift.

While the accuracy requirements can be relaxed in NI applications, the resolution requirements should be the same for all three applications. This is because the purpose of high resolution tracking is to insure smooth motion, which is equally important whether or not there is any correspondance between the virtual world and the real world. The displacement threshold for visual detection of a motion is at least 1.5 arc-minutes [5, p. 926], which is actually larger than the 25 arc-seconds visual acuity used to specify the accuracy requirements for VI applications. Since the resolution cannot be larger than the accuracy, it will be taken as equal for VI applications, or

about 0.03 mm and 0.009°. For HI and NI applications, the resolution should be about 0.1 mm for position and 0.03° for orientation. Note that when the head is moving fast what matters is lag not resolution. Therefore, the resolution has been specified to prevent the perception of discrete motion at very slow speeds, based again on the far-fetched assumption of infinite display resolution.

Because the resolution is limited by either the measurement noise or the size of the quantization cells used, whichever is larger, the maximum noise amplitude has already been implicitly specified. To understand what is meant by this noise amplitude, it is helpful to consider a simple model of how the noise evolves. Assume that the output of the sensor (accelerometer or angular rate sensor) at rest is a zero-mean Gaussian white noise, $x(t)$, with autocorrelation

$$R_{xx}(t_1, t_2) = q(t_1)\delta(t_1 - t_2).$$

If this noise source is passed through a linear system with impulse response $h(t)$ such that the output is $y(t) = x(t) * h(t)$, then it is shown in [23] that

$$E\{|y(t)|^2\} = q(t) * |h(t)|^2 = \int_{-\infty}^{\infty} q(t-\alpha)|h(\alpha)|^2 \, d\alpha. \tag{2.1}$$

Because of linearity, this $y(t)$ is also a zero-mean noise, and therefore its variance $E\{|y(t)|^2\}$ is the square of the RMS noise amplitude in which we are interested. The only problem is that in this case the linear system is an integrator, $h(t) = u(t)$, and the integral in (2.1) blows up. This is because the output $y(t)$ drifts, so its variance tends towards infinity. It is therefore very difficult to interpret the meaning of noise amplitude for this type of system. It seems the most reasonable approach is to measure the RMS amplitude of the output soon after applying the noise source $x(t)$ to the input and before the drift has accumulated and contributed too much to the output variance. Applying the white noise $x(t)$ starting at $t = 0$ is equivalent to using $q(t) = q\,u(t)$ in (2.1), which yields

$$E\{y^2(t)\} = q \int_{-\infty}^{t} |h(\alpha)|^2 \, d\alpha.$$

Substituting $h(\alpha) = u(\alpha)$, we get

$$\text{RMS noise} = \sqrt{E\{y^2(t)\}} = \sqrt{q}\sqrt{t}.$$

Since drift cancellation will occur every 1/4 second when the tracker is not in motion(see Section 2.3), the "peak RMS noise" can be defined as $0.5\sqrt{q}$. Another definition of noise amplitude for noise that drifts is developed in Section 4.3.

Perhaps the most difficult,yet important, requirements on a head-tracker to specify are those relating to speed or responsiveness. Excessive delay causes perceptual decorrelation between actions and their results. Delays of more than 60 ms prevent adaptation to rearrangements of the visual-motor loop, and this adaptation is crucial to the illusion of presence [15]. Delay can also cause motion sickness [22], perhaps even when it is too small to be perceptible as delay. Values anywhere from 5–100 ms have been proposed as the maximum tolerable delay between head-motion and response in a VE. To actually measure it would require the use of a VE system with no noticeable delay, a technology which has not yet been achieved. Rather than trying to guess or discover what is the minimum noticeable delay in head motion to visual feedback, it is certainly safe to arbitrarily choose 1 ms maximum group delay from 0 to 20 Hz as the design goal. This is much smaller delay than really necessary, but with inertial trackers it is easily achieved. It is the total delay between head motion and correction of the visual scene that must be minimized, and it will be a long time before computer graphic rendering speeds of a few ms will be commonplace for scenes of reasonable complexity. Even without any tracker delay, it is very difficult to achieve the desired 15–20 ms total lag with current systems. Any amount of delay introduced by the tracker, therefore, will just worsen the effect of the delay in a regime where it is already noticeable or even problematic. This is why a 1 ms tracker is far more desirable then a 10 ms tracker, even though 10 ms is very fast with respect to the human perceptual system. Table 2.2 summarizes the desired specifications that will be used as a guideline in the design of an inertial head-orientation tracker.

Specification	Angular	Linear
Range	yaw: ±180° pitch: +50°, −70° roll: ±35°	room size
Velocity	±1000°/s	±10m/s
Acceleration	±6000°/s^2	±20m/s^2
Bandwidth	20 Hz	20 Hz
Static Accuracy	opaque HMD: yaw: no requirement pitch & roll: 15° max drift: 3°/min see-through HMD: 0.009°	opaque HMD: max drift: 1.2 cm/min see-through HMD: 0.03 mm
Resolution	opaque HMD: 0.03° see-through HMD: ±0.009°	opaque HMD: 0.1 mm see-through HMD: 0.03 mm
Latency	1 ms	1 ms

Table 2.2: Desired specifications for head-tracker design.

2.2 Selection of Sensors

2.2.1 Gyroscopes and Angular Rate Sensors

The key elements for sensing orientation are gyroscopes and angular rate sensors. There are two basic flavors of gyroscopes: spin gyroscopes and rate gyroscopes. A spin gyroscope consists of a spinning wheel mounted on a three-axis gimbal designed to be as frictionless as possible. When the outer housing rotates, the gimbals swivel to allow the wheel to continue spinning on the same axis. Angular encoders measure the rotations of the gimbals, which corresponds to the orientation of the housing with respect to the spin axis. As rotations about the spin axis are not measured, one spin gyroscope can measure at most two orientational DOFs. The usual arrangement, therefore, involves two spin gyroscopes, one for measuring pitch and roll (called a vertical gyroscope) and one for measuring yaw (called a directional gyroscope). In order to make sure that the spin axis is vertical for a vertical gyroscope and horizontal for a directional gyroscope, the gimbals may be weighted so that they hang in the correct reference orientation before spinning up the rotor. This is called a pendulum-levelled

gyroscope. For the vertical gyroscope the pendulum also prevents unbounded drift by gradually restoring the axis to vertical through a sequence of damped precessions. The disadvantage of this approach is that linear accelerations generate unbalanced torques on the gimbals which cause precession. To avoid this, navigational gyroscopes are instead levelled by active electromagnetic torquers.

Rate gyroscopes measure angular rate instead of angular position. Like spin gyroscopes, they have a spinning rotor mounted on a gimbal, but the gimbal does not rotate freely. When the housing changes orientation, the rotor is forced by the stiff gimbal to change orientation along with it. To change the spin axis requires a torque proportional to the rate of change, and this torque can be measured on the axis of the gimbal and used as a reading of angular velocity. Rate gyroscopes are used for strapdown inertial navigation, and it is common to use three separate gyroscopes strapped down to the three body axes of the vehicle. By a procedure called Euler integration, the angular rates in body-coordinates can be used to compute the orientation in world-coordinates.

There are a variety of angular rate sensors designed to accomplish the same thing as a rate gyroscope without the use of a mechanical spinning wheel. Fiber-optic gyroscopes (FOGs) are the well-established performance leaders in the navigation and guidance arena. By sending a light signal in both directions around a spool of fiber-optic and detecting the phase difference when it recombines, it is possible to measure small angular velocities extremely accurately. Some relatively small FOGs have come on the market, but the prices are still quite high. Recently, a few manufacturers have introduced "solid-state" angular rate sensors which use the piezoelectric effect to measure out-of-plane vibrations of a vibrating quartz tuning fork when it rotates about its axis. There is also development work being done for acoustic [17] and magnetohydrodynamic [16] angular rate sensors.

Table 2.3 lists a variety of commercially available gyroscopes and angular rate sensors that are potentially small enough to be mounted on a human head. The most attractive choice on the table, for miniscule size and amazing drift performance, comparable to an FOG, is the Litton G2000, but it is far too expensive. The Murata

Manufacturer	Model/Price	Specifications
spin position gyros		
Gyration, Inc. 12930 Saratoga Ave.,bldg C Saratoga, CA 95070 408-255-3016	GyroEngine: $3,000 developer's kit, including 1 vertical, 1 directional gyro	±80° pitch & roll range, 5–10°/min drift, 1.75 in. X 1.25 in. dia., 1.2 oz.
rate gyros		
Futaba Corp. P.O. Box 19767 Irvine, CA 92713 714-455-9888	G155: $130 per axis	model airplane servo controller, perhaps can be modified to obtain rate gyro output, 1.26 in. X 1.08 in. X .97 in., .9 oz.
Litton Guidance & Control 5500 Canoga Ave. Woodland Hills, CA 91367 818-715-2909	G2000: $5,500 per axis not including interface electronics	±400°/s range, 0.1°/hr random drift, .75 in. X .75 in. dia., .8 oz.
quartz angular rate sensors		
Watson Industries, Inc. 3041 Melby Rd. Eau Claire, WI 54703 715-839-0628	ARS-C151: $399 per axis	±1000°/s and lower ranges available, 18°/min drift, 1.2 in. X 1.4 in. X 3.2 in., 3 oz.
Murata Mfg. Co., Ltd. 2200 Lake Park Dr. Smyrna, GA 30080 404-436-1360	Gyrostar: $295 per axis	±90°/s range only, .2°/hr random drift, 1 in. X 1 in. X 2.3 in., 1.4 oz.
Systron-Donnor Inertial Division 2700 Systron Dr. Concord, CA 94518 800-227-1625	GyroChip: $1,500 per axis	±1000°/s and lower ranges available, 2–8°/hr random drift, .9 in. X 1.9 in. X 2.2 in., 3.3 oz.
fiber-optic gyros		
Litton Guidance & Control 5500 Canoga Ave. Woodland Hills, CA 91367 818-715-2909	LN-200: $60,000 3-axis inertial measurement unit with accelerometers	±1000°/s range, $.03°/\sqrt{hr}$ random drift, 3.5 in. X 3.5 in. dia., 1.4 lb.
Smiths Industries 4141 Eastern Ave., S.E. Grand Rapids, MI 49518 616-241-7000	?	$.15°/\sqrt{hr}$ random drift, about .6 in. X 1.7 in. dia., 4 oz.

Table 2.3: Small gyroscopes and angular rate sensors

would be an excellent choice, if only it were available with a larger angular velocity range. Considering that the human can detect a drift as slow as 3°/min, the Futaba and the Watson offer inadequate performance, and the FOGs are expensive and bulky, and offer better performance than really necessary. The Gyration GyroEngine has a drift rate slightly higher than the 3°/min threshold, but this is a compromise that would be worth making to have a three-axis weight of 2.4 oz., compared to 9.9 oz. for the Systron-Donnor GyroChips. When the Systron-Donner GyroChips came on the market, an order for GyroEngines had been placed nearly a year earlier which Gyration had still not been able to fill because they had not yet worked out a reliable manufacturing process. So, despite the increased weight and the difficulty of having to integrate them, three GyroChips were ordered with ±1000°/s range and they arrived within two weeks. To corroborate the manufacturer's specifications, a simple drift test was performed using 12-bit A/D conversion, and the drift rate was found to be 2°/min or less on several 6-minute trials.

2.2.2 Inclinometers and Compasses

Having selected the angular rate sensors, the next step is to find appropriate sensors for compensating their drift. This is relatively easy. There are virtually no requirements on their dynamic response. The angular rate sensors take care of the rapid response to movement. The corrective sensors, however, are ultimately responsible for maintaining the absolute accuracy of the system.

Table 2.4 lists several gravimetric tilt sensors, or inclinometers. In principle, all inclinometers work basically the same way as instrumented pendulums, although the proof mass is often a fluid. The Fredericks Model 0717 was chosen for its tiny size, large range, and low price. Since the output is highly nonlinear, it must be converted into actual angles through the use of a look-up table (LUT). If the LUT contains enough entries, the absolute accuracy will be totally determined by the repeatability. Refering to Table 2.2, .03° is only a factor of 3.3 greater than the minimum perceptible misalignment angle in the most stringent of applications. For current uses, this is more than adequate accuracy. If a higher degree of accuracy is needed in the future, the

Manufacturer	Model/Price	Specifications
Lucas Sensing Systems 21640 N. 14th Ave. Phoenix, AZ 85027 800-545-3243	AccuStar: $120 single axis	±60° range, nonlinear past ±45°, .05° repeatability, 1 in. X 2 in. dia.
The Fredericks Co. 2400 Philmont Ave. P.O. Box 67 Huntington Valley, PA 19006 215-947-7464	0717: $66 2-axis sensor, $125 2-axis electronics unit	±70° range (nonlinear), .03° repeatability, .6 in. X .3 in. dia.
Spectron, Inc. 595 Old Willets Path Hauppauge, N.Y. 11788 516-582-5600	SP-5000: $100 2-axis sensor, $126 2-axis electronics unit	±50° range (not quite linear), .01° repeatability, 1 in. X .5 in. dia.

Table 2.4: Gravimetric tilt sensors

tilt sensor can easily be replaced. The Fredericks 0717 has a settling time constant of 245 ms.

Detecting the direction of heading is more difficult than detecting tilt. Gravity points down very consistently, and is barely influenced even by large mountains. By contrast, earth's magnetic field has a dip which depends on latitude, and is affected significantly by power lines, electrical equipment and large amounts of iron or steel in the vicinity. The limitation is not the precision of the sensors, as fluxgate magnetometers are available with 10^6 dynamic range, but rather the complex shape of the magnetic field in a typical laboratory environment. KVH Industries[1] sells a product called the C100 Compass Engine which combines a fluxgate compass with a microprocessor to perform automatic compensation for deviation caused by iron in the installation environment. This processing scheme, which requires an initial compensation procedure to store the mapping pattern of the installation environment, yields absolute accuracies of 0.5°. However, the mapping only compensates for iron that

[1] 110 Enterprise Center / Middletown, RI 02840 / (401)847-3327

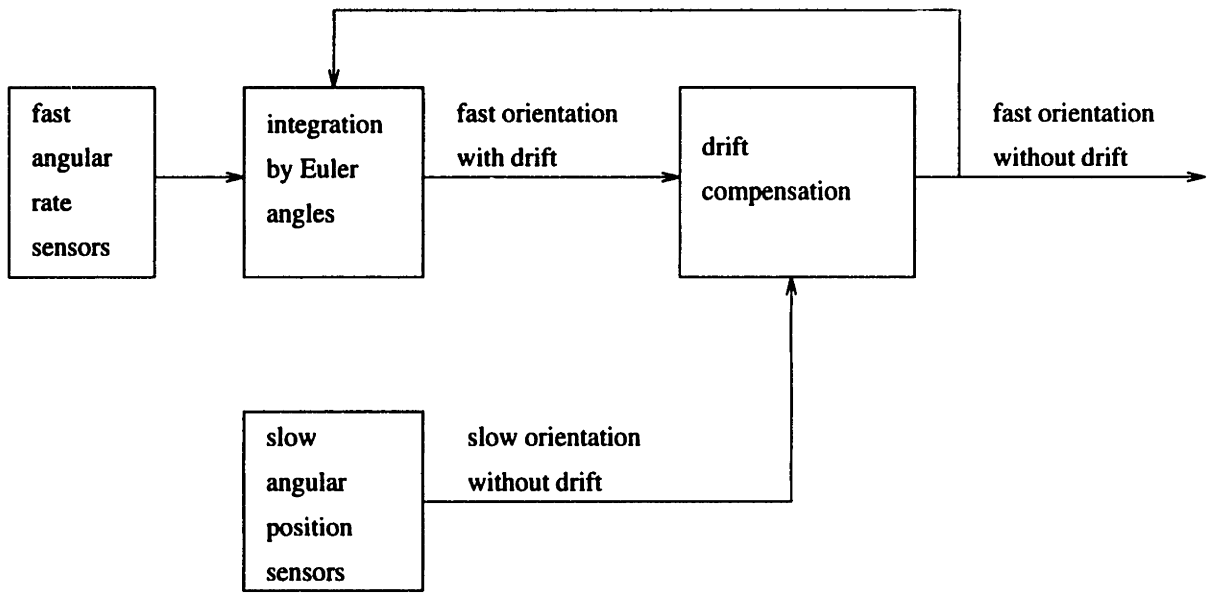


Figure 2-1: Proposed design of inertial orientation tracker with drift compensation.

moves with the sensor (e.g., the hull of a boat), so irregularities in the external field could cause much larger errors.

In this thesis, the orientation-tracker was designed to make use of an Etak[2] model 02-0022 fluxgate compass. Unfortunately, during initial testing with the compass it was discovered that the magnetic field in M.I.T. Building 36 is almost completely vertical! Apparently, large steel plates inserted between the floors to reduce electromagnetic interference also reorient the magnetic flux from floor to ceiling. The compass readings in the horizontal plane were extremely weak and variable, and so the compass was removed from the system. The software still contains provisions for reading a compass, so if the system is ever used in another building the compass can be re-installed.

2.3 Summary of Proposed Design

Figure 2-1 is a block diagram outlining the basics of the proposed design. Having selected the fast angular rate sensors and the slow angular position sensors, it remains to work out the methods of integration and drift compensation that should be used

[2]1430 O'Brien Dr. / Menlo Park, CA 94025 / (415)328-3825

with these particular sensors to come as close as possible to meeting the desired specifications.

The equations for integration by Euler angles are [10]

$$\dot{\psi} = P + Q \sin\psi \tan\theta + R \cos\psi \tan\theta \tag{2.2}$$

$$\dot{\theta} = Q \cos\psi - R \sin\psi \tag{2.3}$$

$$\dot{\phi} = Q \sin\psi \sec\theta + R \cos\psi \sec\theta. \tag{2.4}$$

Here P, Q, and R represent the angular velocities of an object around its x, y, and z body-coordinates respectively. According to the coordinate convention in aeronautics, the x-axis points forward, y points right, and z points down. P, Q, and R can be thought of as the outputs of three orthogonal angular rate sensors mounted on the object. Recall that ϕ, θ, and ψ respectively represent yaw, pitch, and roll which are defined as the amount of counter-clockwise rotation applied about the z, y, and x body-axes in that sequence to get to its current orientation. Note that [10] uses the reverse definitions of ϕ and ψ. Equations 2.2–2.4 aren't actually integral equations at all, but they provide the rates of change of the Euler angles, which can then be integrated to obtain the updated Euler angles. The simplest procedure for integration in a computer is, after each time step Δt, to set

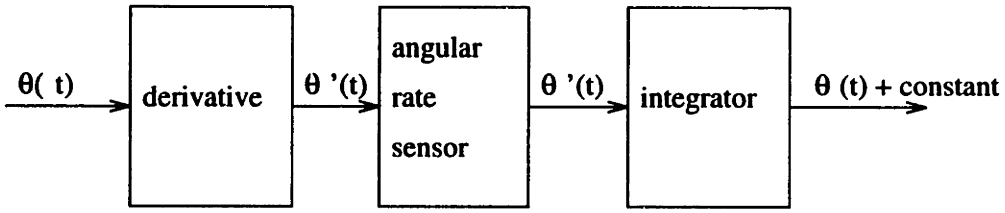
$$\psi(t + \Delta t) = \psi(t) + \dot{\psi}(t)\,\Delta t \tag{2.5}$$

$$\theta(t + \Delta t) = \theta(t) + \dot{\theta}(t)\,\Delta t \tag{2.6}$$

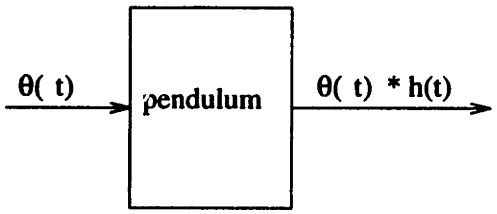
$$\phi(t + \Delta t) = \phi(t) + \dot{\phi}(t)\,\Delta t. \tag{2.7}$$

Note that the increments applied at time t depend on the values of the Euler angles at time t. If those were taken simply as the values computed as the result of the previous integration step, then the accumulation of a little bit of drift would cause the next increment used to be wrong, and the drift rate would accelerate exponentially. To prevent this disaster, the Euler angles used in (2.2–2.4) are taken from the output of the drift compensation step, as shown in the block diagram.

The challenge of designing the drift compensation algorithm is essentially how to combine the signals from two types of sensors, one of which has large bandwidth but has undefined response at 0 Hz, and the other of which has DC response but very small bandwidth. To simplify the analysis, consider a deterministic model for measurement of orientation around a single axis. The basic system without drift correction looks like this:



In the frequency domain, the derivative multiplies the input spectrum by f, and the integration then multiplies it by $1/f$. The result is that the output faithfully reproduces the input except for frequencies extremely close to $f = 0$, where tiny errors introduced by the angular rate sensor will be hugely magnified by $1/f$. At the same time, another sensor is measuring the low frequency changes of the input, like this:



In the case of a fluid pendulum, the system function $H(f)$ may not be exactly that of a damped harmonic oscillator, but it will certainly be some sort of low-pass filter with a bandwidth on the order of 1 Hz. In the frequency domain, the desired processing is to splice the two spectra together. This could be done by low-pass filtering the pendulum output using a filter, $H_{lp}(f)$ with a cutoff around .5 Hz, high-pass filtering the integrated angular rate signal with a filter $1 - H_{lp}(f)$, and then adding the two filtered signals together. Keeping zero phase in such a processing scheme is sufficiently tricky that it is worth looking for a simpler method in the time domain. Fortunately, the human head is rarely in continuous motion for more than 5 seconds or so. When it pauses, the fluid-filled inclinometer will settle to its correct pitch and roll values in

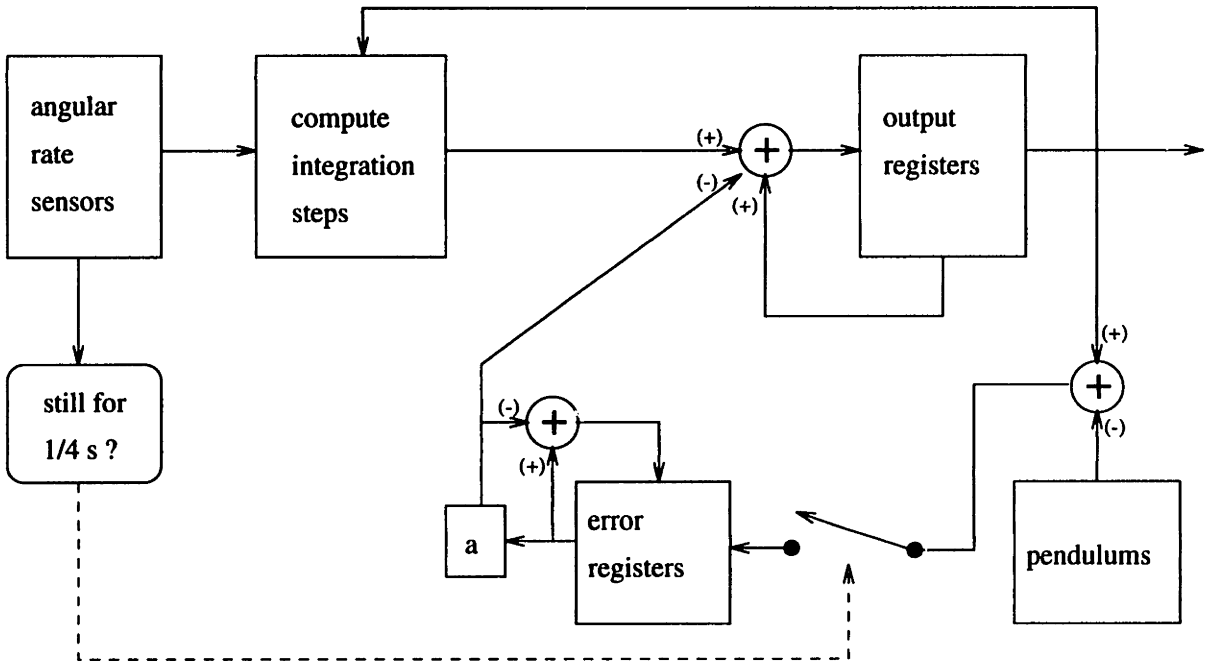


Figure 2-2: The final design as built.

about 1/4 second. At this moment, the absolute head orientation is known with high accuracy, and the output variables can be reset to these values, thus anhilating all the drift that has accumulated since the last reset. To prevent sudden orientation shifts from jarring the user, the drift should be removed from the output gradually, not all at once. Figure 2-2 is a block diagram portraying the processing scheme in greater detail than Figure 2-1. This diagram represents the processing that occurs on each time-step in a discrete-time implementation. The dotted line and switch are meant to imply that if the outputs of the angular rate sensors are zero for 1/4 second, then the pendulums are read and subtracted from the current output values, and these error values overwrite the current values in the error registers. When the switch is open, a fraction a of the error values will be subtracted from the output values during each integration step, and also subtracted from the error registers. This way, the error will decay exponentially with a time constant $1/a$. Since the switch opens and closes at random times, the system is by no means LTI. However it is designed to simulate as closely as possible the trivial LTI system $\vec{w}(t) = \vec{v}(t)$, and linear systems theory will be used to characterize its dynamical performance.

Chapter 3

Implementation

3.1 Description of Hardware

The apparatus that was built is shown schematically in Figure 3-1. Each box in the diagram will be discussed separately.

3.1.1 Head-Mounted Sensor Assembly

The goal in mounting the sensors is to make a package that is as small and light as possible, and can be conveniently yet rigidly attached to the head-mounted display. It is also important to mount the angular rate sensors with their axes as nearly orthogonal as possible.

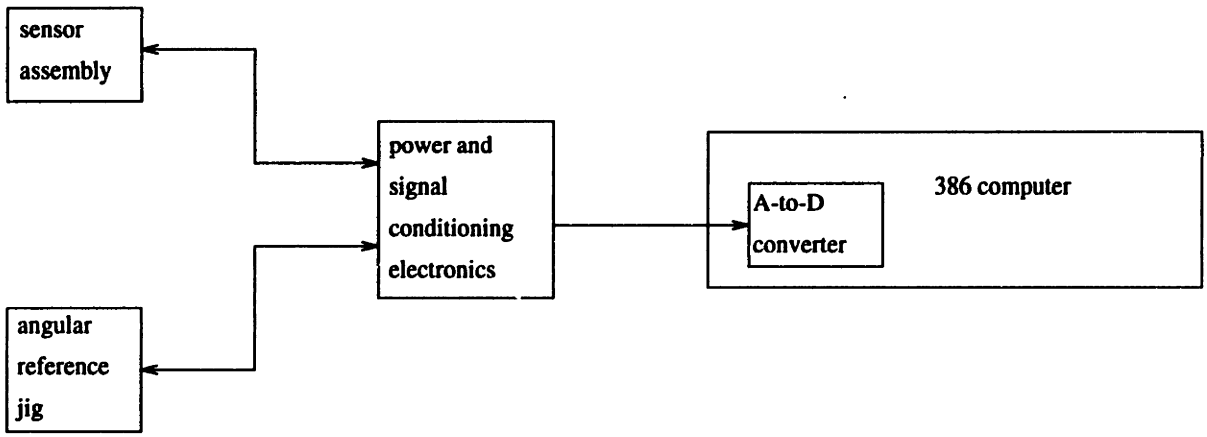


Figure 3-1: Overview of the apparatus.

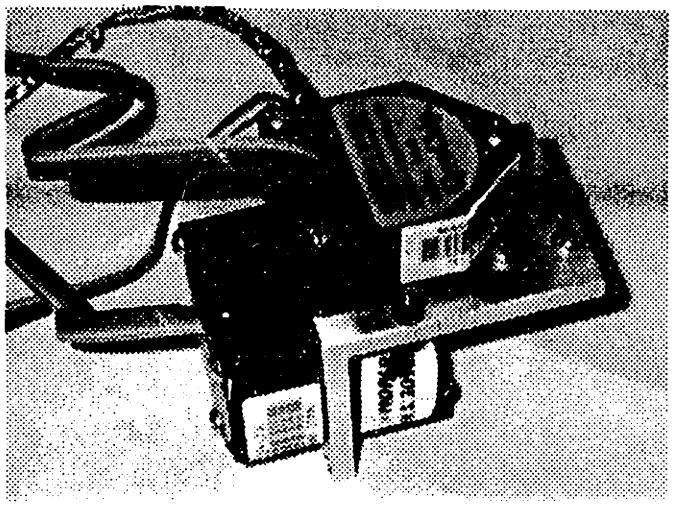


Figure 3-2: Sensor mounting bracket.

An initial attempt was made to produce a plastic mounting bracket that would hold the sensors in their most compact arrangement. To do this it would have been necessary to start with a plastic cube whose side was slightly longer than the largest sensor dimension, then cut away most of the plastic to recess the sensors into the block. Because a 2.5" cube could not be found, an effort was made to mold one out of machinable wax, but this did not work well. As a simpler solution, a mounting bracket was machined of aluminum. In order to keep to one-piece design, a section of L-bracket was used, which does not accomodate the sensors in their absolutely most compact configuration, but is still fairly efficient. The 1/4" thick aluminum also weighs more than a similar piece of plastic, but its weight is still only 1/3 of the weight of the entire sensor assembly. A photograph of the bracket with all the sensors mounted is shown in Figure 3-2. Each GyroChip is attached to a surface using three 8-32 x $1\frac{1}{8}$" screws. The bracket surfaces were re-milled to assure that they are flat and perpendicular. The mounting holes were drilled on a Bridgeport milling machine with .001" accuracy, to help assure the orthogonal alignment of the axes. Nonetheless, the alignment of the GyroChip sensitive axis with respect to its housing is only guaranteed to within 0.5°, which is approximately the same as the amount of play in the mounting of the GyroChips to the bracket.

The Fredericks 0717 fluid tilt sensor is mounted in a hole on the top surface of the bracket. A piece of shrink-tubing was used to match the diameter of the glass bubble

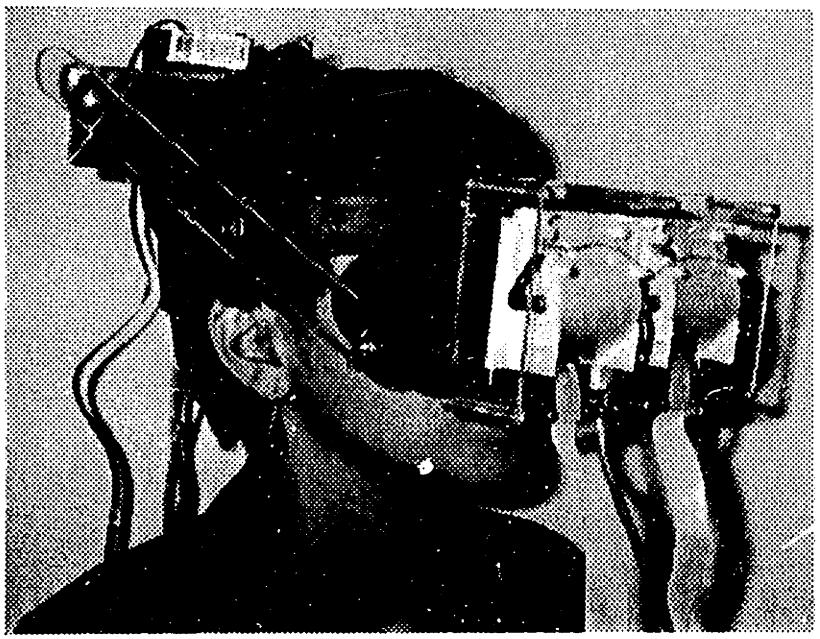


Figure 3-3: Head-mounted display with sensor assembly attached.

to the diameter of the hole and accomplish a very snug fit. Because the bubble is round and has no alignment markings, the alignment of the pitch and roll directions of the tilt sensor was done by eye.

The leads of all four sensors are terminated with 7-pin hex connectors so that the assembly may be removed as a module. It is attached to the HMD using four screws through holes on the top surface of the mounting bracket. A photograph of the HMD with the sensor assembly mounted on the back is shown in Figure 3-3. The assembly weighs 15.6 oz.s, but the HMD incorporates a rear counterweight which can be reduced by this amount to maintain balance when the tracker is attached.

3.1.2 Angular Reference Jig

In order to calibrate the system, a jig was needed to position the sensor assembly at precisely known angles while reading the outputs. This same jig will also be used for the evaluation of the system in the next chapter.

The jig built consists of a 6" diameter motor bolted to a large heavy aluminum plate with rubber feet. The spindle of the motor is outfitted with a bracket assembly which can mount the sensor block in a variety of orientations, and also serves as a

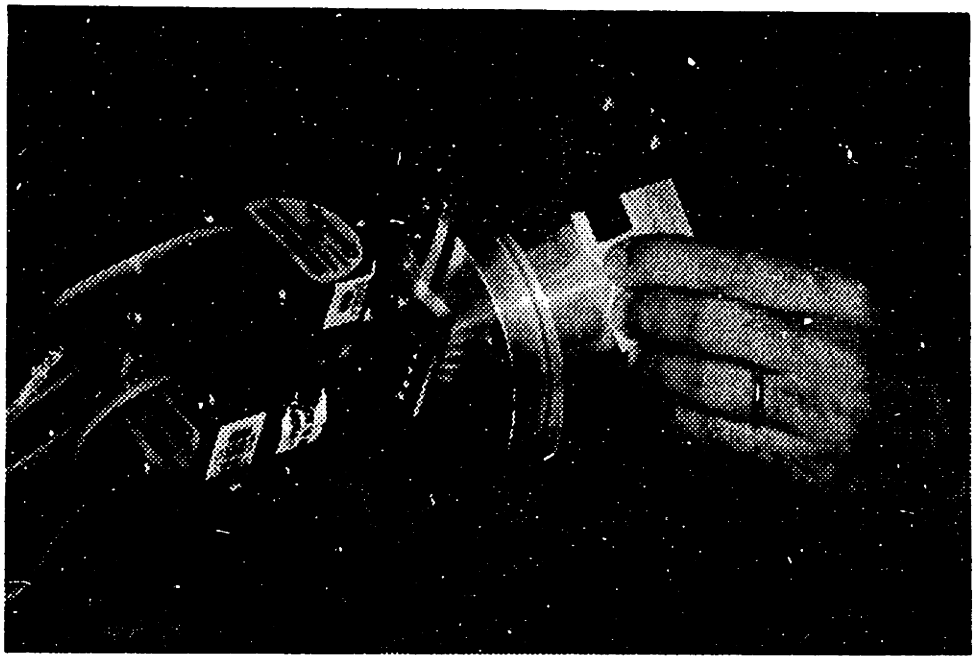


Figure 3-4: Angular reference jig.

convenient handle for rotating the spindle by hand. Hard rubber stops were provided to limit the range of motion to ±90°. The motor itself is never energized — it merely serves as a convenient sturdy bearing for mounting a rotatable platform. The end of the motor spindle opposite from the sensor mounting platform is equipped with a model F78CC502 5K precision potentiometer from New England Instrument Co. Because the potentiometer will serve as the angular reference for the calibration and evaluation of the head-tracking system, its accuracy is very important. The F78CC, with a linearity of ±0.25%, is the most accurate potentiometer compatible with the mounting requirements of the motor. Nonetheless, 0.25% of the ±180° full range only guarantees an accuracy of 0.5°. A photograph of the testing jig, with the sensor block mounted in a suitable orientation for applying roll motions, is shown in Figure 3-4. In each of the three orientations, the tracker can be mounted at five different radial distances from the motor axis. This is useful for testing the effect of acceleration on tracker performance.

3.1.3 Computer

The data processing is performed by a PC Brand computer with an Intel 80386SX CPU running at 25 MHz. The system was initially made to work without a math coprocessor, but it is floating point intensive, and the subsequent installation of an Intel 80387SX math coprocessor allowed the update rate to be increased from 40 to 200 updates/second. The software could be rewritten to perform the trignometry and all the computation using integer arithmetic. Using integer math, it could probably run fast enough on almost any PC platform.

Analog-to-digital conversion is performed by a Data Translation DT2814 board. It provides 40 kHz throughput with 12-bit resolution and 16 single-ended or 8 differential input channels. The input range can be selected from 0–5 V, ±5 V, and ±2.5 V. The ±2.5 V range is used to match the output range of the GyroChips. There is also an on-board clock with software selectable frequencies from .005 Hz to 30 kHz. One disadvantage of this low cost board is that it lacks channel scanning capability; the on-board clock can only trigger repetitive conversions on a single channel. For this reason, the channel scans have to be performed in software from inside the main interrupt service routine. The DT2814 on-board clock can not even be used to generate periodic interrupts for timing the data acquisition, because it is hardwired to trigger serial conversions on a single channel and nothing else. Rather than purchase a separate timer card, the necessary periodic interrupts are obtained from the system clock on the motherboard through the system timer tick interrupt.

The computer is also equipped with a Trident VGA card, an rs-232 serial interface adaptor, and an EtherLink II ethernet interface card. The VGA display is useful for displaying the yaw, pitch and roll angles graphically in real-time. The serial card and ethernet card are for communicating head-tracker output data to a host computer running a VE, but these functions have not yet been implemented.

3.1.4 Interface Electronics

Figure 3-5 shows the interface circuitry between the sensors and the data acquisition

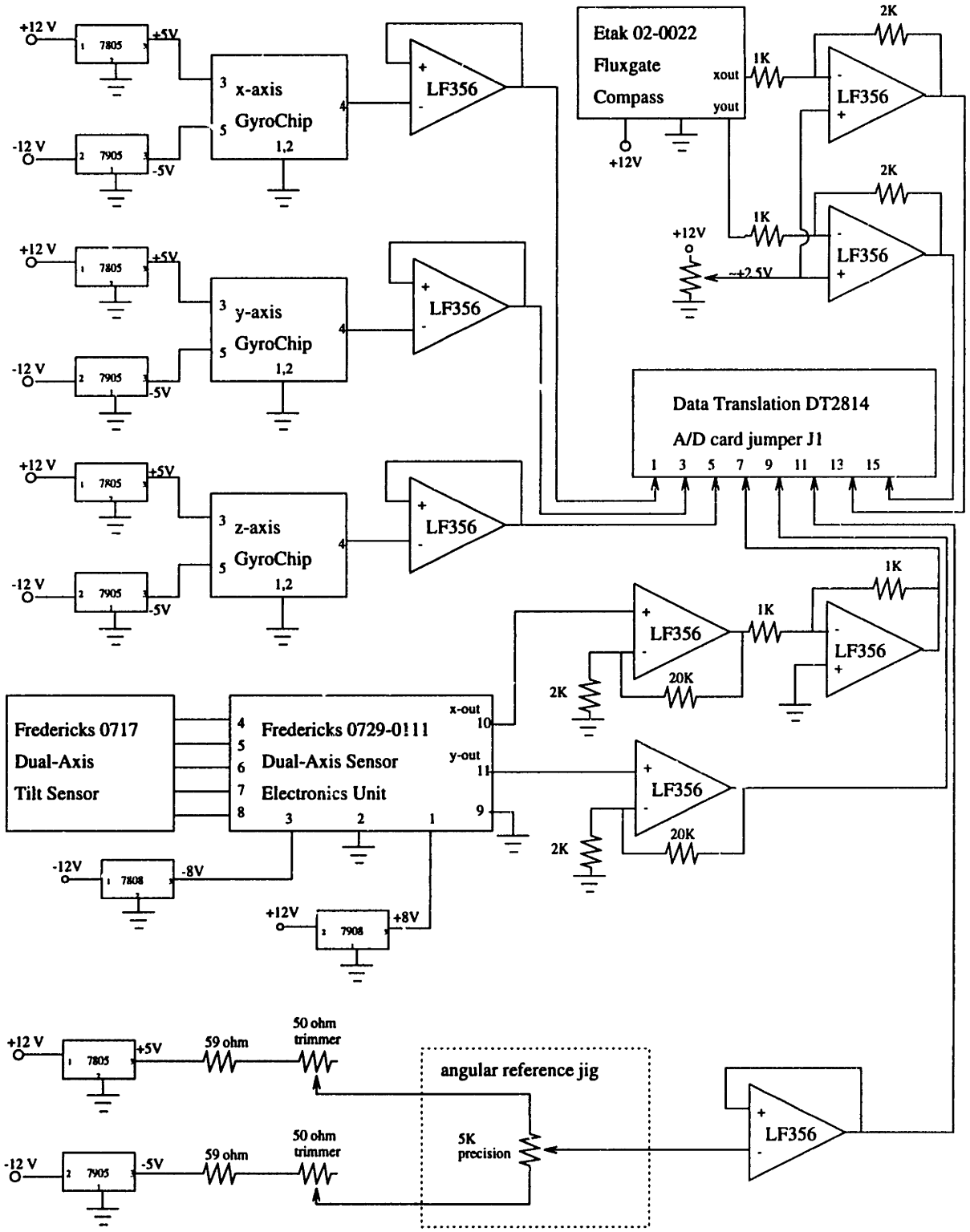


Figure 3-5: Schematic diagram of power and signal conditioning electronics.

card. Following the advice of the manufacturer, each GyroChip is provided with its own separate voltage regulators. Since the GyroChips use an internal 8000 Hz oscillation, they can cause a slight 8000 Hz ripple in the power supply, which can enable multiple chips with a shared power supply to interact unfavorably. The outputs of the GyroChips are ±2.5 V, so the A/D card was jumpered for ±2.5 V input range, and the GyroChips are simply buffered with unity-gain voltage followers. The GyroChips which measure P, Q, and R (the angular velocities counterclockwise about the x, y and z-axes) must be connected to channels 0, 1, and 2, respectively, because this is where the software expects to find them.

Channels 3 and 4 of the A/D converter are for reading the "x out" and "y out" of the Fredericks tilt sensor. With the bubble mounted the way it is in the sensor bracket, the "y out" corresponds to pitch and the "x out" corresponds to negative roll. This is why an extra inverter has been included for the x output in order to allow the software to use symetrical code for processing all the channels. The sensor electronics unit can be run on ±8 to ±14 V supply, and the output ranges are proportional to the supply voltage. To guarantee that the angular sensitivity of the tilt sensors doesn't change if the ±12 V power supply drifts, the power supply is clipped to ±8 V with regulators. With this set, it was found that a gain of 10 was needed to bring the outputs up to slightly more than ±2.5 V.

Channel 5 reads the output of the angular reference jig, buffered by a simple voltage follower. The potentiometer is energized by ±5 V regulators to assure stability. The trimmer potentiometers are hand tuned until the outputs are precisely ±2.5 V when the jig is placed at its ±90° stops. This is a see-saw operation requiring a considerable number of iterations to set both limit values to the desired accuracy. This step is very important and must be done patiently, because the angular reference jig is used to generate the LUTs for the tilt sensors and to test the accuracy of the complete system. Any inaccuracy in the initial calibration of the angular reference jig will cause the measured system performance specifications to be worse than they could be.

When the fluxgate compass is used, it is connected to channels 6 and 7 with

the circuit shown. Although the compass only measures one DOF, yaw, it has two outputs representing magnetic field strengths in two orthogonal directions. To obtain heading direction, the software must compute the ratio of these two components and take the arctangent. This yields the direction of the magnetic field component in the plane of the compass, which is nominally the horizontal. Although the compass has been removed from the head-tracking system, the schematic remains in case it is later necessary to put it back in.

All the sensors and signal conditioning electronics are powered by a single ± 12 V power supply which is floated with respect to chasis ground to prevent a giant ground loop from forming between the ground of the computer and A/D card and the ground of the power supply. A great deal of time was spent hunting down and eliminating ground loops in order to reduce the noise at the input of the A/D converter from 8 LSBs (3 bits out of 12 meaningless) to 1 LSB (11 bits effective precision).

3.2 Description of Software

Appendix A contains a complete listing for the program INTEST, which consists of the inertial head-tracker program INTRACK plus some additional functions for testing the system. This section will overview the structure of the software, elucidate some of the less obvious details of the code, and explain how to use it.

3.2.1 Structure

The software implements the discrete-time processing scheme of Figure 2-2, with the only difference being that the determination of "still for 1/4 s?" is made using the outputs of both the pendulums and angular rate sensors instead of the angular rate sensors alone. Since the computations in each block occur on discrete time steps, and are the same every time, it is necessary to acquire data from the sensors at precisely controlled periodic time intervals. Therefore, the data acquisition is driven by interrupts which are handled by the interrupt service routine (ISR) "int_handler." The most logical way to structure the program would be to acquire the data from

the sensors and call all the processing steps from within the interrupt handler. This would guarantee that on each time step all the processing steps were perfomed once and the euler angles were updated before the next step began. However, this interrupt handler would be quite long. Such a long interrupt handler is very poor programming practice, because other interrupts with lower priority are blocked while the ISR is in effect, leading to anomolous system behavior. It was found that making graphics and floating point calls, both of which are interrupt-driven, from within the ISR led to occasional spurious graphics, and worse yet spurious values being computed for the euler angles.

To overcome the deleterious effects of a long interrupt handler, it was necessary to adopt a somewhat more complicated structure. The interrupt handler was stripped down to the minimum possible functionality — data acquisition and setting a few flags. The data processing is now performed asynchronously in the main loop, using the flags set by the ISR to determine when there is new data to process. The data acquired by the ISR is passed to the main loop in a global ring buffer. The main loop waits until the head of the ring-buffer advances in front of the tail, then computes the new Euler angles and catches the tail up to the head. Normally, the head will have advanced only once, if at all, since the last update step. However, if the timer interrupts are being generated at a rate close to the maximum throughput of the data processing routines, it is possible that occasionally more than one new set of sensor readings will be queued up in the ring-buffer. This is why there is a ring-buffer instead of just one storage variable. If multiple data steps have accumulated in the ring-buffer, they will all be processed together by a single iteration of the function "update_eulers" with an appropriate multiplication of the integration increment. This guarantees that even if the timer ticks are coming too fast to be processed individually, the processing loop will catch up completely on every cycle, with a small loss of precision. This program structure is very robust, and makes it easy to know and to maximize the update rate. The timer can be safely sped up until just below the point where multiple data queing begins. The maximum lag is then the period between timer ticks.

The processing done by the main loop when new data has been queued is done

by two functions, "update_stilltime" and "update_eulers." The first function, "update_stilltime," determines when the outputs of the tilt sensor bubble are reliable, and uses them to update the global error variables that will be used in "update_eulers" to compensate drift. The values read from the bubble sensor are taken as reliable when they have been nearly constant for 1/4 second. Waiting 1/4 second longer after the outputs stop changing not only assures that the readings used are very stable, but also allows averaging of many readings to obtain much better precision than a single reading with 12-bit conversion would accomplish. If on a particular iteration the still time reaches 1/4 second, the averages accumulated over the past 1/4 second are converted into actual pitch and roll angles using the LUTs. The differences between the current Euler angle outputs and the newly computed bubble sensor angles are then stored in the error variables. This updating of the error variables is inhibited if either the pitch or roll reported by the bubble sensor is outside of the angular range within which the bubble sensor provides valid measurements. It can also be turned off by the user in order to observe the nature of the drift without compensation.

The second function, "update_eulers," uses Equations 2.2 – 2.4 to compute the integration step, then adds the step to the Euler angle output variables while subtracting off a fraction of the error. This function requires 4 trignometric functions and 19 double-precision floating-point multiplications, and is the rate limiting step in the loop.

After the Euler angle output variables have been updated, they are copied to mirror variables called "euler_export" so that they can be accessed by an interrupt driven serial communications protocol without causing conflict. During normal operation these exported Euler angles are displayed graphically as meter needles on the screen, updated by the function "display_angles" on each iteration of the main loop. When the user opts to perform a test run, the angles are displayed in a different color and also recorded in an array called "data" for a period of time. The data array can be saved to a disk file for subsequent analysis.

3.2.2 Look-Up Table Generation

The LUTs that are used in the function "update_stilltime" to convert the outputs of the tilt sensor (and heading sensor if used) to angles are generated and managed by routines in the file "0717.c." The LUTs are passed between the functions using an enormous global array of double precision floating-point numbers. To allow for such a large global array declaration, both "intest.c" and "0717.c" must be compiled using the "huge" memory model of the Microsoft C compiler.

The LUT for each axis is created by measuring the outputs of the sensor for 101 equally spaced angles between -1 radian and +1 radian, then essentially calculating the inverse of this function using linear interpolation to assign an angle to every possible measurement value. The sensor output values are measured by the user once, under the guidance of the function "measure_raw_vals." Since the resulting array, "raw_vals" is much smaller than the array "lut" which will be generated from it, "raw_vals" is the one that is saved out to a disk file, "0717.log," and loaded back into memory upon each new invocation of the program. The first index of the two-dimensional arrays "raw_vals" and "lut" selects which channel or axis the data pertains to. An argument of 0 indicates values stored from A/D channel 3 (pitch), 1 indicates channel 4 (roll), and 2 is for channel 5, which may later be used for a heading sensor. The routine "initialize_lut" takes this channel number as an argument and uses the data in "raw_vals" for that channel number to fill the array "lut" for the same channel. At program initialization time, all three channels of "raw_vals" are read in from the file "0717.log", and all three channels of "lut" are initialized. The outputs of the tilt sensor reach their maxima at the edges of the sensor's usable range: about ± 1 radian. During the initialization of "lut" any raw values beyond this are assigned special values outside the range ± 1 radian. Whenever the look-up table returns such a value, the function "update_stilltime" will know that the sensor was tilted beyond its usable range, and will not attempt to update the errors.

3.2.3 User Interface

The main program consists of an initialization section, followed by the main loop described in Section 3.2.1, followed by a user interface section. The main loop runs until the user presses a key to access one of the program's other functions. If the key pressed corresponds to one of the available functions, that function is executed immediately. Otherwise, the user is presented with a menu of the keys that may be pressed with a description of the function that each key activates.

At present the menu consists of the following:

r – roll calibration procedure

p – pitch calibration procedure

y – yaw calilbration procedure

S – save current calibration data

R – retrieve calibration data

s – reset angles and start over

b – remeasure biases, reset angles, start over

f – begin or end fine trimming of biases

3 – display lut for channel 3

4 – display lut for channel 4

5 – display lut for channel 5

l – change length of test run and decimator

t – perform test run

d – display data

c – clear data

o – output data to file

a – toggle attitude drift compensation (currently 1)

h – toggle heading drift compensation (currently 0)

q – quit.

If at any point the user wishes to abandon the menu and return to head-tracking, "s" can be used to re-start, or "b" may be used to re-measure the biases and restart. The biases, i.e. the outputs of the sensors when the tracker is at rest in its reference

orientation, are measured automatically at startup. If the tracker was not at rest at this time, the biases should be re-measured or the system will behave quite poorly. It is also helpful to re-bias the system after about 30 minutes of use, because the biases of the sensors may shift slightly when they warm up. The function "f" for fine-trimming the biases was added after the discovery in Section 4.1.1 that the routine "measure_biases" doesn't work perfectly. When drift compensation is disabled, "f" can be used to reduce the drift rates, but how much benefit it provides has not yet been formally evaluated.

Many of the functions on the menu concern the generation of the LUTs, and can be performed once and then ignored unless the hardware is modified. The roll, pitch and yaw calibration procedures are highly automated and can be performed with a minimum of tedium, considering the precision which the task involves. To generate a new roll LUT, for example, the user must first mount the sensor assembly on the angular reference jig in the correct orientation so that it can be rotated about the roll axis. After typing "r" to start the procedure, the screen will display the line "Measuring output for -1.0 radians." The following line contains a number reflecting the current position of the angular reference jig in radians. The user must move the jig until this number matches the number above: -1.0 radians. As soon as the numbers have matched for enough time for the tilt sensor output to stabilize, the sensor output is automatically recorded into "raw_vals", the user is informed by an audible tone, and a new line appears prompting for the next angle: -0.98 radians. By moving the jig a little bit and waiting for the auditory feedback, it is possible to perform the procedure largely by feel, much more quickly than by only looking at the angles being displayed on the screen. It is possible to measure 101 positions with high accuracy in about 15 minutes. After recording all the measurements, the new LUT constructed from them may be previewed by pressing "3," which displays a graph of the channel 3 or roll contents of the array "lut." If the graph looks sufficiently smooth and shapely, indicating a succesful calibration procedure, the new "raw_vals" should be saved to the file "0717.log" by pressing "S." If it is not desired to save the new calibration, the old one can be restored by pressing "R."

Both static and dynamic testing on the system can be performed by using the function "t," which records the input angle and all three output angles for a number of iterations specified by the variable "record_length," which can be set using the function "l." The length of time spanned by a test run can be changed by using "l" to change the decimator from 1 (no decimation) to n. Then the angles will only be recorded for every n^{th} update. After performing a test, the data acquired can be previewed on the screen by typing "d." When displayed in this manner, the roll angle is graphed in red, the pitch is graphed in purple, and the yaw is graphed in yellow. The angle of the reference jig, which serves as a baseline for comparison, is graphed in blue. The data from the most recent testrun may be output to a file for later analysis by typing the letter "o." The user will be asked for a filename in which to save the baseline, roll, pitch and yaw data as four columns of an ascii flat file. If desired, the most recent test run can be cleared from memory by pressing "c," although this is rarely necessary because the next test writes over the old data anyway.

Finally, there are commands for turning the drift compensation on or off. By default, attitude (pitch and roll) drift compensation is on, and heading drift compensation is off. This is because the system currently has no heading sensor. Turning on the heading drift compensation will cause the yaw output to always tend to zero, no matter what the true yaw. It may be more useful to turn off the attitude drift compensation, in order to observe the drift rate of the system under uncompensated conditions.

Chapter 4

Evaluation

In this chapter, testing is carried out on the system described in the previous chapter. It is impossible, due to limitations on time and equipment, to evaluate every aspect of the tracker's performance, but an effort is made to test the most important specifications discussed in Section 2.1.1.

In particular it is deemed unnecessary to actually carry out testing for the input range specifications. It should be sufficiently obvious from the design that the only really relevant limitation is a maximum head angular velocity of $\pm 1000°/s$. The positional range is limited only by the length of the wires connecting the sensor assembly to the electronics unit. They could be made longer, or even replaced with wireless telemetry devices, but they haven't been. The tracker is capable of tracking any input orientation. However, drift compensation will not occur while the pitch or roll exceeds the range of the Fredericks tilt sensor, about $\pm 60°$. In terms of acceleration, the only limit is that the sensor assembly could be damaged by dropping it on the floor. While it is attached to a person's head there is nothing to worry about; it can take alot more linear and angular acceleration than the head can. No thought has been given to testing the device at different temperatures. All the evaluation described in this chapter is carried out at room temperature, generally between 70° and 80° Farenheit.

4.1 Drift

Since the current prototype does not correct for drift in yaw, it is important to measure the rate of this drift. It is also of intellectual interest to know the drift rates in pitch and roll when they are not being corrected by the inclinometer. This information makes it possible to predict how often attitude updates are needed to keep within a certain desired accuracy tolerance. In principle, the yaw, pitch and roll drift rates should be comparable, because they all result from integration of GyroChips with the same 1000°/s range. However, it is possible that GyroChips of the same model but with different serial numbers may have slightly different performance. It is also true that the integration by Euler angles procedure is not symmetrical with respect to the three axes. Refering to Equations 2.2-2.4, when all the Euler angles are 0, then

$$\dot{\psi} = P \tag{4.1}$$

$$\dot{\theta} = Q \tag{4.2}$$

$$\dot{\phi} = R. \tag{4.3}$$

This implies that at the outset the yaw, roll and pitch drift rates will differ only if the sensors differ. However, as drift accumulates to different degrees in different Euler angles, Equations 2.2-2.4 will become increasingly dissimilar, and the rates may be faster in certain Euler angles than others. For these reasons, the drift rates are measured for the yaw axis singly, and also for all three axes jointly.

4.1.1 Yaw Drift

Long-term drift recordings are made for yaw according to the following procedure. Using the "l" command from the INTEST menu, the record length is set to 3600 and the decimator is set to 203. Since the update rate for the integration process is 202.65 Hz[1], these settings cause the Euler angles to be recorded very close to once

[1]For Section 4.1 only. After this data was collected, it was changed to 145.65 Hz to accomodate changes to the software.

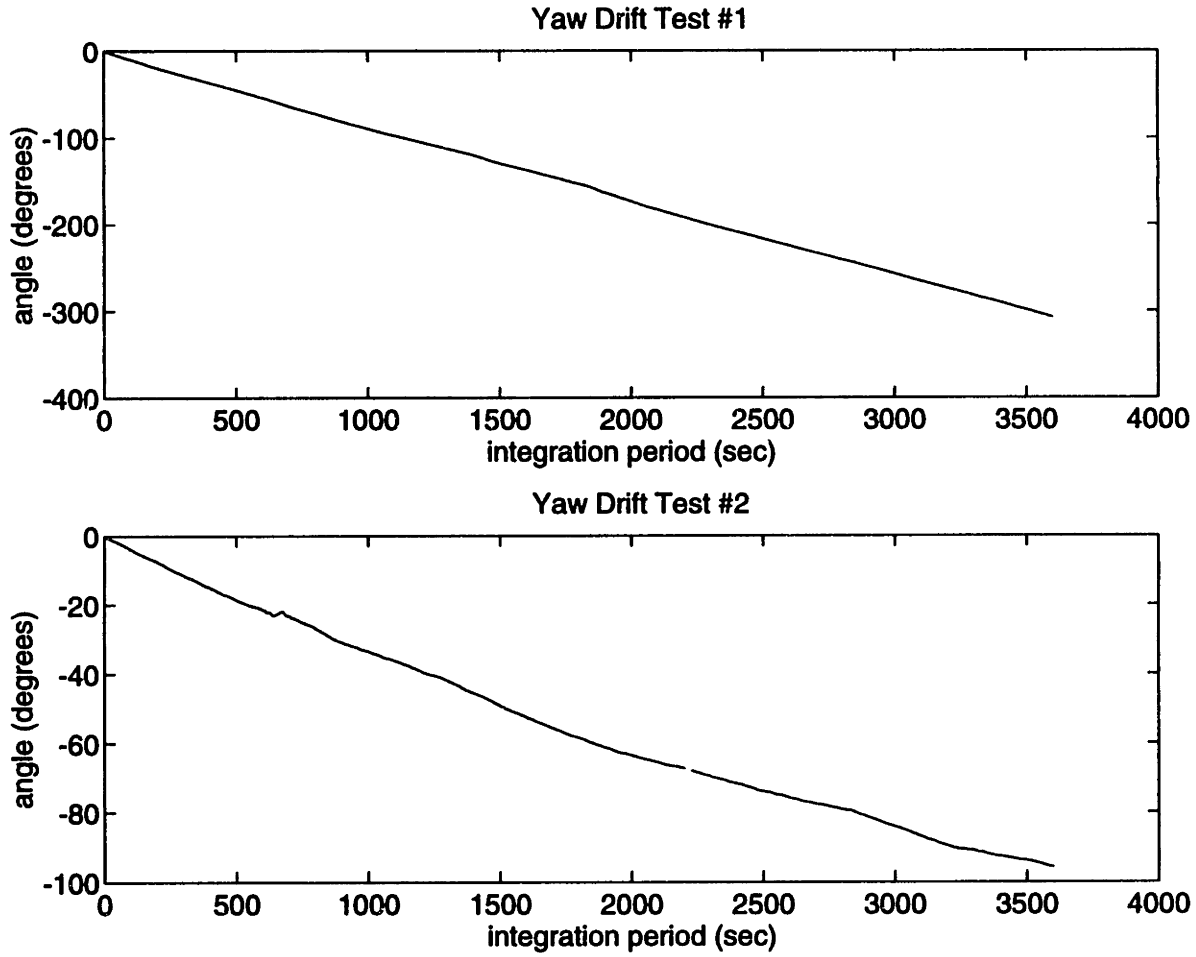


Figure 4-1: Results of 1 hour yaw drift integration tests.

per second for an hour. This protocol is used to permit direct comparison with GyroChip integration test results provided by Systron-Donner [13]. During the hour of recording, the sensor assembly sits still on a table, so any changes in the output angles are the result of drift. The earth's rotation rate of 15°/hour should not appear in the recordings because the bias measurement routine reads the outputs of the rate sensors, which includes a steady angular velocity resulting from earth rotation, and treats it as part of the bias. The results of several hour-long yaw drift recordings are shown in Figure 4-1.

These results of about 100–300°/hour are quite poor compared to the integration test results shown in [13]. On six hour-long integration tests, Systron-Donner obtains peak-to-peak deviations of 1.4–8.3°. Their tests, however, are performed under very tightly controlled circumstances designed to demonstrate the ultimate performance

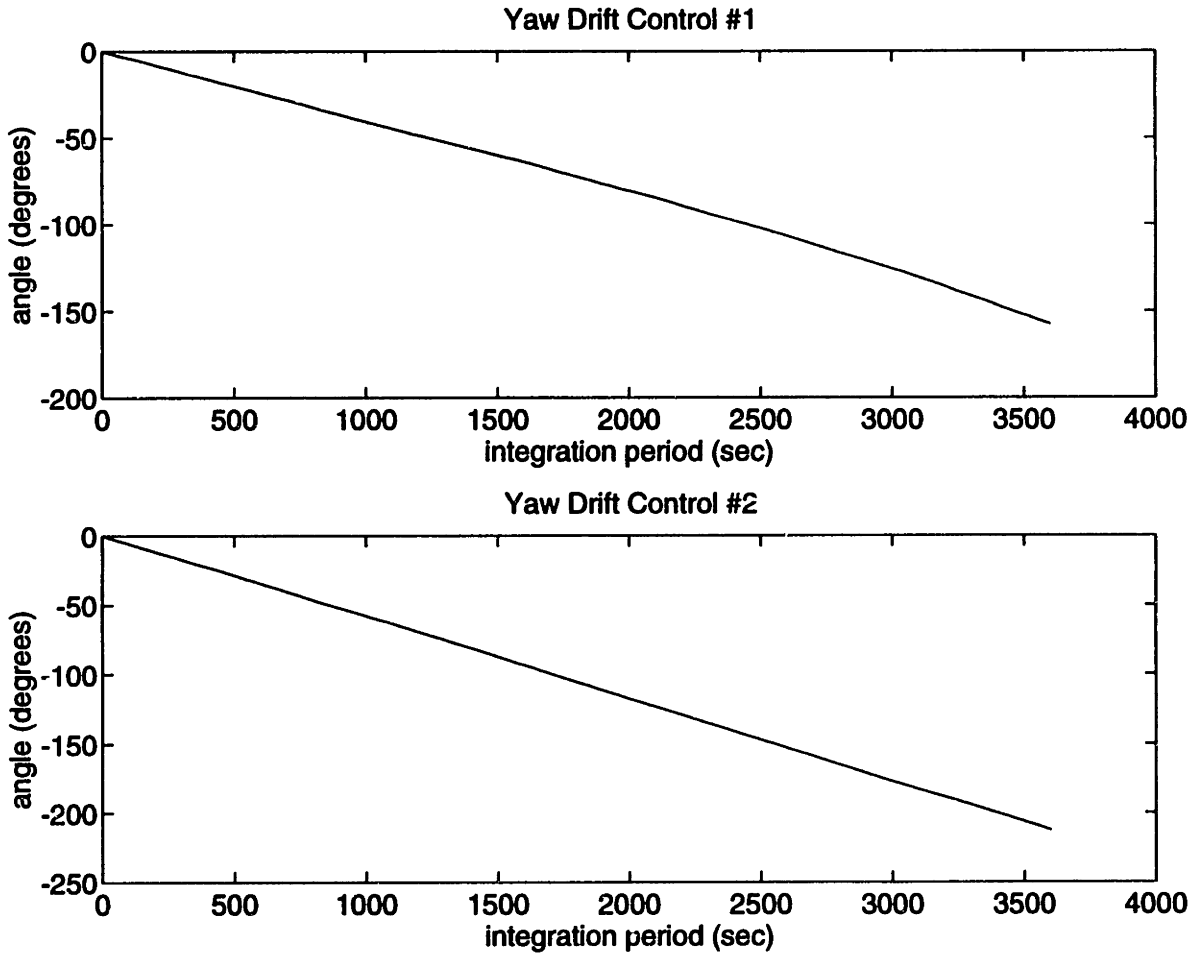


Figure 4-2: Results of 1 hour yaw drift integration tests with inputs grounded.

limitations of the GyroChip sensor. The sensor is isolated in a temperature-controlled chamber for three days up to and including the tests, whereas in the tests of Figure 4-1 the sensor is left on a table next to the door and tested shortly after power-up. However, the major difference lies in the method of integration. The Systron-Donner tests are performed using an analog integrator circuit based on a precision op-amp. When tested with its input grounded, the op-amp only drifts 0.2° in an hour. By contrast, the head-tracker integrates the sensor digitally after 12-bit A/D conversion, using a fairly complicated nonlinear algorithm to integrate the three Euler angles together.

To demonstrate that the A/D conversion and/or the digital computation is at fault, a control experiment was done under the same conditions used to generate Figure 4-1, but with the inputs of the A/D converters all grounded. The results are

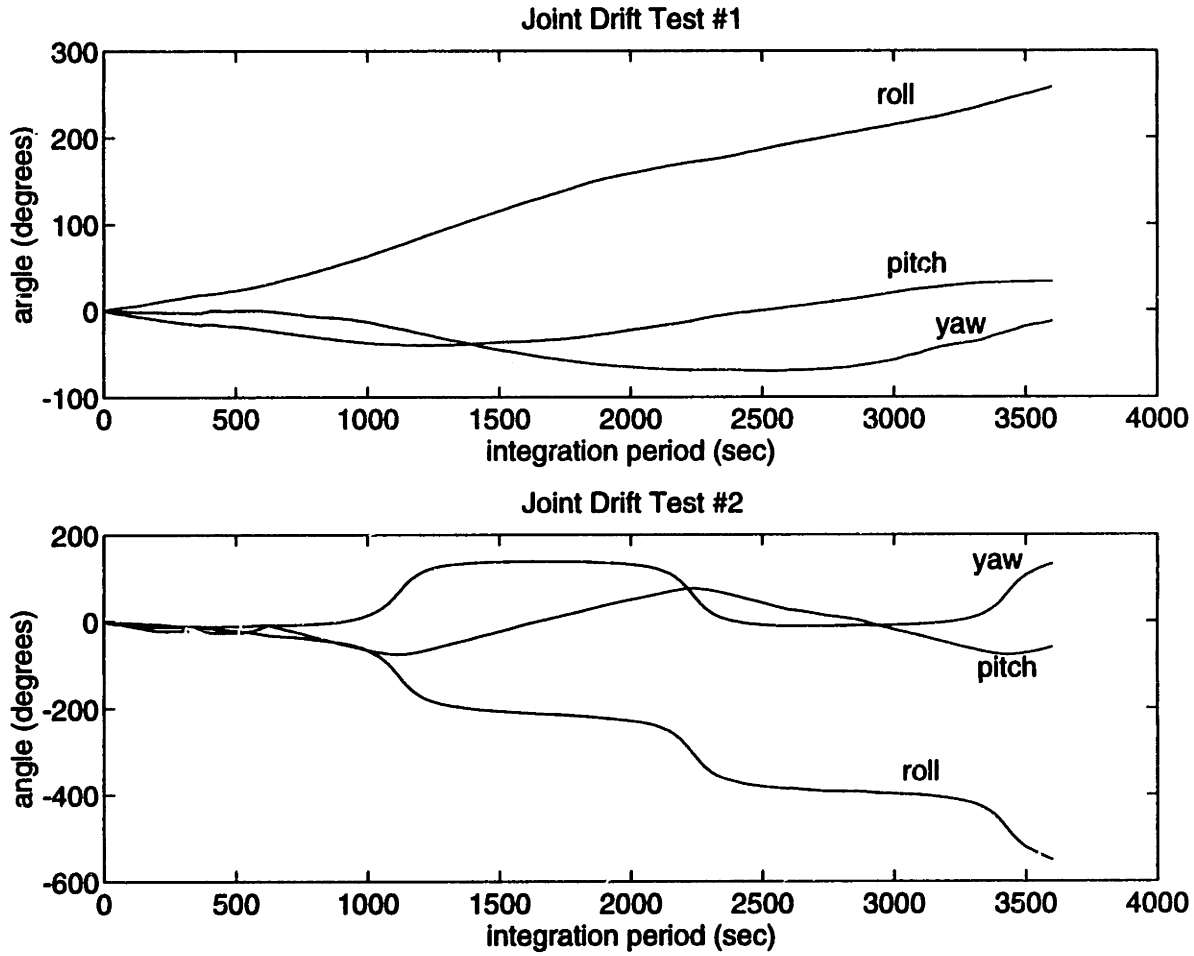


Figure 4-3: Results of 1 hour joint roll, pitch and yaw drift tests.

shown in Figure 4-2. Comparing to Figure 4-1, the sensor noise is clearly not the dominant cause of drift. Another interesting fact that can be observed by comparing the two figures is that the yaw drifted almost steadily counterclockwise in all four trials. This could be a random coincidence, but it does not appear very much like random drift. It is possible that the bias measurement routine does not work properly, but it is not obvious from inspecting the code what is wrong with it.

4.1.2 Joint Pitch, Roll, and Yaw Drift

As was previously stated, it is also interesting to measure the roll and pitch drift rates when the compensation algorithm is turned off. These can be used to predict how often the user must pause in order to maintain the desired accuracy. Recording all

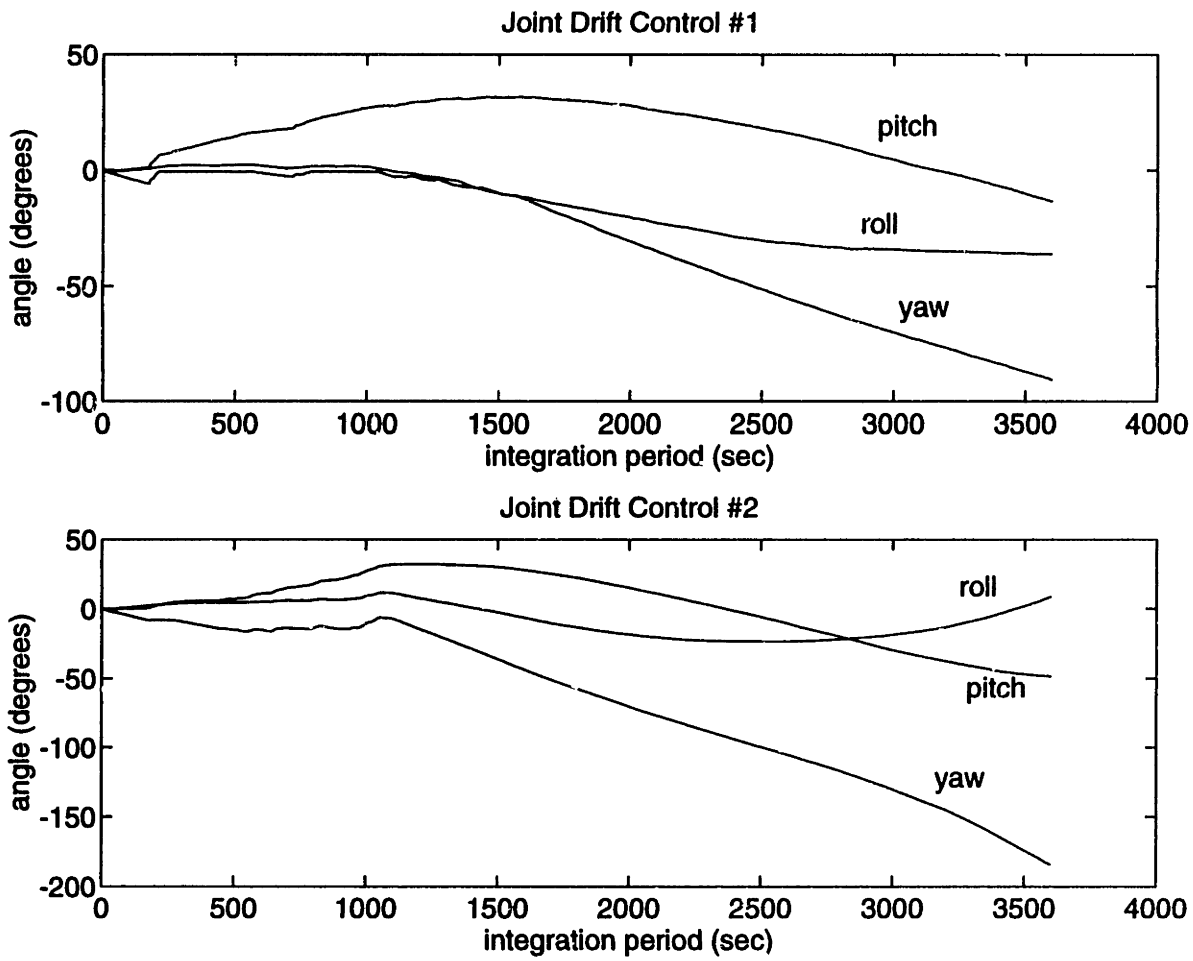


Figure 4-4: Results of 1 hour joint roll, pitch and yaw drift tests with inputs grounded.

three axes simultaneously also captures three times as much statistical information about the drift in a one hour period, assuming that the drifts behave independantly. With this rationale, two additional drift experiments were performed with the compensation algorithm turned off. The results are shown in Figure 4-3 and the results of two grounded-input control experiments are shown in Figure 4-4. In all of these joint drift experiments there are no steady monotonic drifts for any of the Euler angles. This implies that it is something about the attitude drift compensation procedure that caused the consistent counter-clockwise yaw drift in Figures 4-1 and 4-2.

The second drift test in Figure 4-3 exhibits some peculiar correlations that would not be expected in three independant random walks. This is caused by a phenomenon called "gimbal lock" which is inherent in the Euler angle representation of orientation. If an object is mounted on a three-axis gimbal, whose gimbals, from the outermost

to the innermost are yaw, pitch, and roll, then most orientations of the object can be uniquely described in terms of the rotation angles of the gimbals. However if the pitch is elevated to 90°, the roll and yaw axes become coincident, so that rotations in roll and yaw with equal magnitude and same sign cancel each other out and do not affect the orientation of the object. Likewise, at -90° pitch, rotations in yaw and roll with equal magnitude and opposite sign cancel. The net result is that when an object is vertical, yaw and roll become arbitrary. This effect can be seen as a singularity in the Euler integration formulae. Inspecting Equations 2.2 and 2.4 it can be seen that when $\theta \rightarrow \pm 90°$, roll and yaw will both drift wildly with the same rate of $Q \sin\psi \sec\theta + R \cos\psi \sec\theta$ and same or opposite sign depending on the sign of θ. This behaviour can be clearly seen in the second drift test in Figure 4-3. At about 1200 seconds, the pitch drifted close to -90°. The roll and yaw started to rotate equally in opposite directions. As the orientation continued to drift along the same trajectory, it came out of gimbal lock with the yaw and roll values 180° away from where they had been, thus reversing the direction of pitch drift. If the pitch were not restrained to the range ±90°, it would have continued steadily down at a little over 270°/hour, while the roll and yaw would have had very little drift overall. Although the graph of the Euler angles appears quite complicated, it actually represents a very simple somersault motion. If the Euler angles were used to control the orientation of an object being rendered in 3-D, the motion would again appear quite simple.

4.2 Accuracy

As previously stated, the yaw drifts and therefore inherently has no accuracy. Accuracy will be measured for the pitch and roll outputs only.

4.2.1 Static Accuracy

The static accuracy is defined as the maximum (or sometimes r.m.s.) deviation of the reported angle from the actual angle as measured by the angular reference jig, when the tracker is held at a fixed orientation. Since the accuracy may depend on

the orientation, it is measured at a number of arbitrary angles for each axis. Each measurement should be made for a period of time long enough for the output to stray to its greatest deviations. Since the time constant for drift correction is 5 seconds, the tracker is held for at least 40 seconds at each orientation, with the last 20 seconds of data being used for the accuracy determination. In order to reduce the number of data files to be transmitted from the tracker PC to the workstation, the data for all seven angles tested is collected in one continuous experiment and stored in a single large file. The experiment is performed using a record length of 4000, an interrupt frequency of 145.6 Hz and a decimator of 16, thus causing the program to collect data for a little over 7 minutes at a sample rate of 9.1 Hz. At the outset of the experiment, the testing apparatus, with tracker mounted, is set at an angle of 0° and the bias is measured and fine trimmed. Attitude and heading drift compensation are enabled, and when the second-hand of a stopwatch passes zero, the test run is started. After each minute, the testing jig is moved manually to a new angle. The sequence of test angles was determined in advance and marked on the side of the testing jig. The test angles used are approximately $0°, \pm 20°, \mp 40°$, and $\pm 55°$, in that order. However, the angles are basically arbitrary, and no effort is made to move to exactly those angles. Instead, effort is concentrated on holding the angle constant for the duration of the minute, in order to achieve *static* accuracy measurements. After several attempts, it was found that the angle could not be held perfectly constant for a whole minute by hand, so a simple aid was devised: a block of metal with an indentation to match the convex outer surface of the motor was coated on the indented side with double sided tape. By holding the block against the motor and resting the handle against this stop, it is easy to maintain a constant angle for a minute, and quickly move the stop to a new position at the end of the minute. The block can be seen in Figure 3-4.

The results of the static accuracy determination experiment for pitch are plotted in Figure 4-5. The agreement between the "actual" angle (i.e., that reported by the angular reference jig), shown as a dashed line, and the angle measured by the tracker, shown as a solid line, is so good that the two lines almost appear as a single line in the graph. To see the discrepancy between the two lines clearly, Figure 4-6 displays the

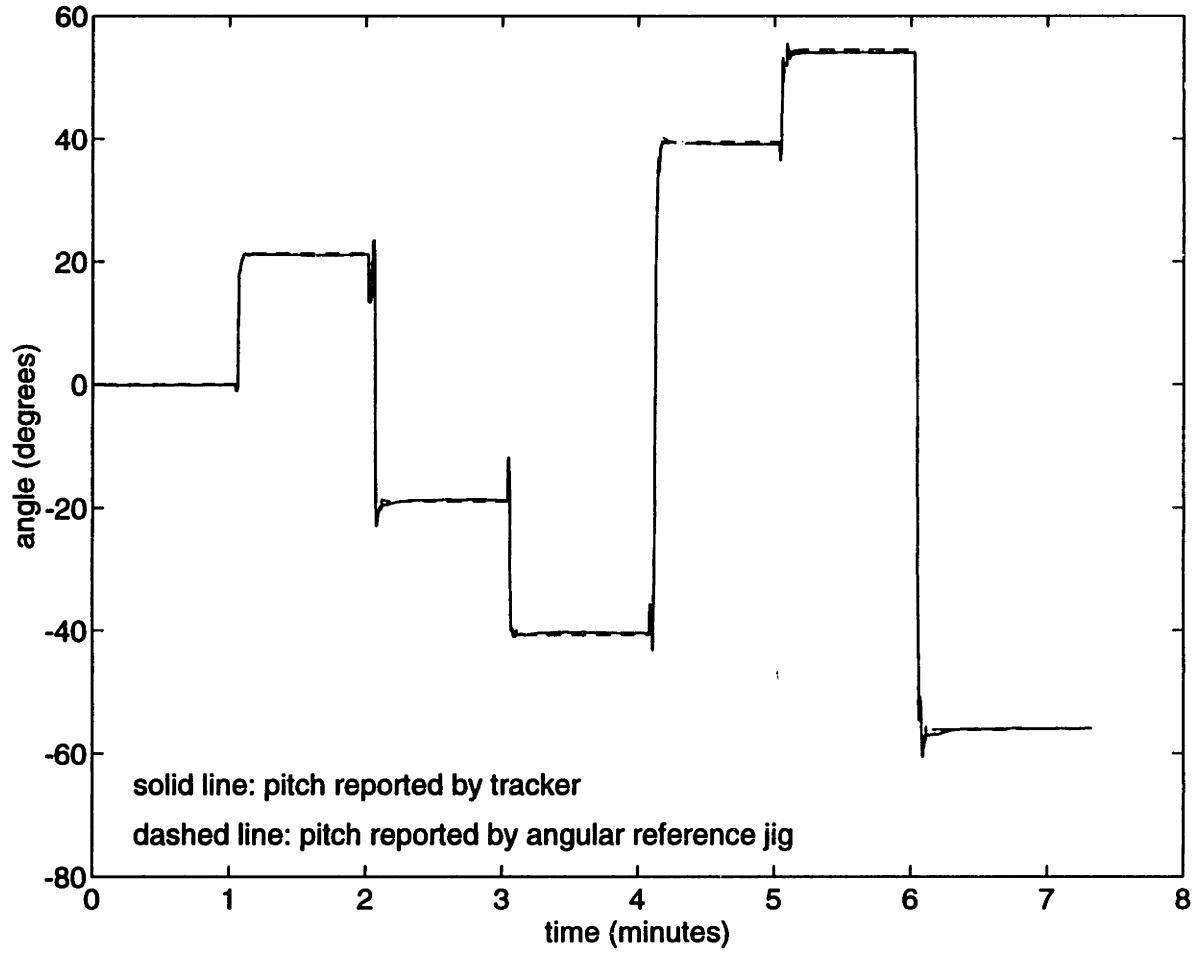


Figure 4-5: Pitch static accuracy at seven angles.

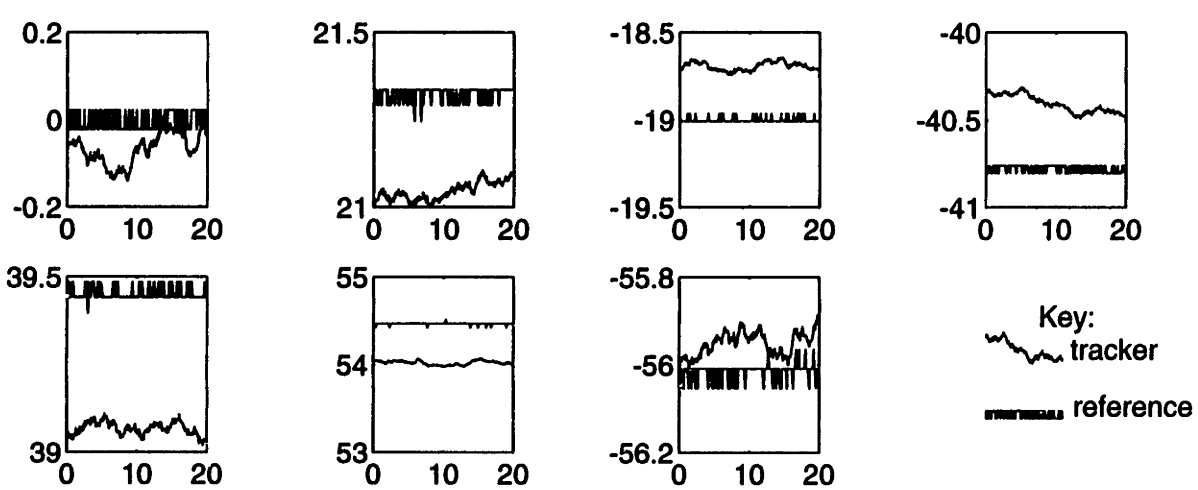


Figure 4-6: Detail of pitch accuracy at each angle.

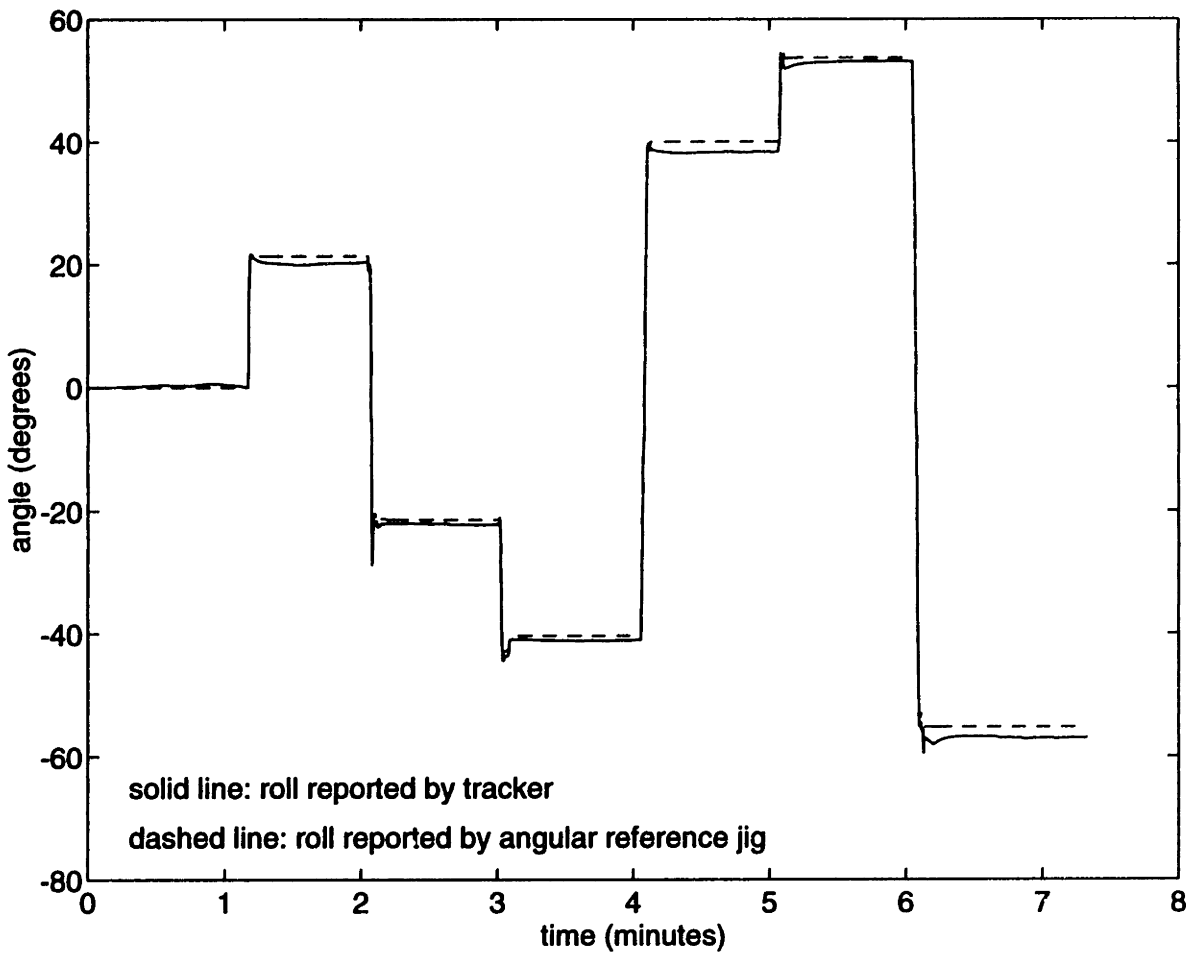


Figure 4-7: Roll static accuracy at seven angles.

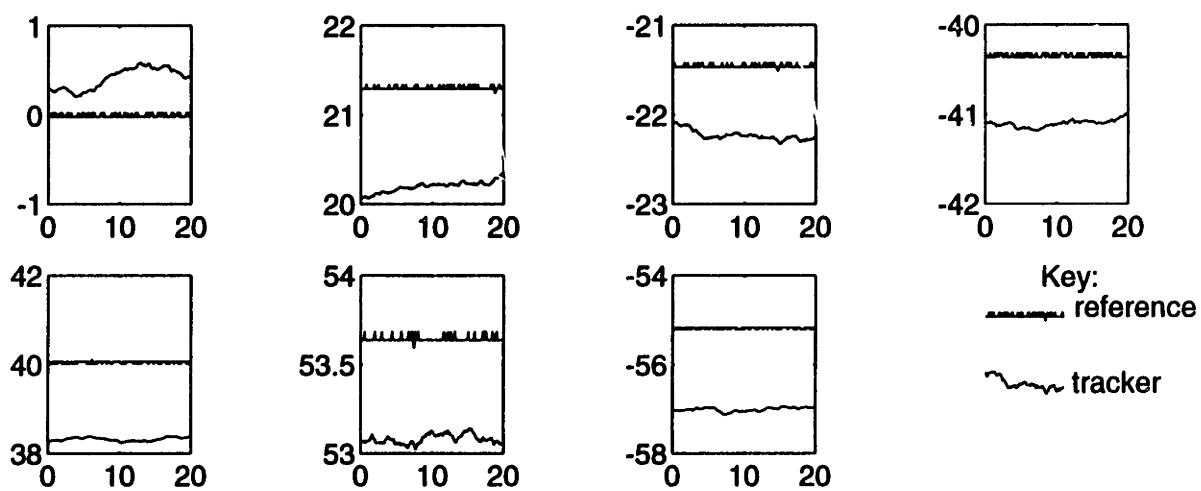


Figure 4-8: Detail of roll accuracy at each angle.

last 20 seconds of each plateau in Figure 4-5 on separate axes. From these magnified views, the maximum errors can be seen to be $-0.15°$ at a test angle of $0°$, $-0.35°$ at $21.3°$, $+0.35°$ at $-19°$, $+0.5°$ at $-40.8°$, $-0.45°$ at $39.5°$, $-0.5°$ at $54.5°$, and $+0.15°$ at $-56°$. Apparently the accuracy is best near $0°$ and decreases at large angles, although the results at $-56°$ are anomalously good. Since the accuracy of the angular reference jig is only $0.5°$, this experiment has only determined that the static accuracy of the tracker is at least as good as the accuracy of the precision potentiometer used to measure it. This is an important result of this thesis: it has now been shown that it is possible to build an inertial orientation tracker with accuracy at least as good as a mechanical tracker, hitherto the most accurate tracker available.

The accuracy results for roll, shown in Figures 4-7 and 4-8 are for some reason inferior to those obtained for pitch. It is not at all obvious why this should be the case, as the methods used for system calibration and accuracy determination were identical. The errors range from $0.5°$ at a test angle of $0°$ all the way up to $2°$ at $-55°$. A peculiar feature of these errors is that, except at $0°$, they are all negative. This is symptomatic of a system that has not been properly calibrated to eliminate measurement bias, but repeating the roll calibration procedure to generate a new LUT did not improve the situation. Because improved accuracy was not a major motivation for the development of an inertial head-tracker, it was decided to let this mystery go unsolved at this time.

4.2.2 Accuracy While In Use

While the static accuracy reveals more about the fundamental limits of the sensors used in the system, it is also interesting to ask how accurate the system is during the whole time it is being used. This question is related to the dynamic performance of the system. For example, if a system has a 100 ms lag and the head turns at $500°/s$, there will be a momentary error of $50°$. This type of error should be considered as lag, not inaccuracy. Since the dynamic performance will be evaluated in Section 4.4, this section does not aim to discover errors of this type. Rather, these tests are done to make sure that the system does not become confused by dynamic input motion.

In particular, the processing scheme makes use of pauses in the input motion, and it is important to verify that the accuracy is not affected too strongly by variations in the frequency or duration of the pauses.

The main experiments were done using a record length of 3000, clock interrupt frequency of 145.65 Hz, and decimator of 3, resulting in a one minute recording sampled at 48.55 Hz. One minute is long enough to generate a variety of interesting input motions. The goals in generating the input motions were 1) to simulate the kinds of motions that would be encountered by a head-tracker in use, and 2) to generate a variety of different types of motions that might "trick" the system. Since the tracker had to be mounted on the angular reference jig and could not be attached to the head, typical head movement had to be simulated by hand. This was accomplished by looking around the room at various objects as if playing in an imaginary virtual environment, while attempting to move the tracker in correspondence with the head. This was difficult to do, and the simulation may not be entirely realistic, but it is far more like head motion than the type of input used to measure static accuracy in the previous section. An effort was also made to provide a variety of motions, fast and slow, with and without frequent pauses.

The results for both pitch and roll are shown in Figure 4-9. The top graph for each experiment shows a recording of all the motions during the minute. A dashed line shows the angles recorded by the angular reference jig, and a solid line shows the tracker output for the DOF being tested, although for the pitch experiment the two lines coincide. The bottom graph for each experiment shows the error, computed by subtracting the dashed line from the solid line. For pitch, the error ranges from $-1°$ to $+0.5°$, with the exception of a positive spike at about $t = 45$ seconds. The spike is easy to explain. At that moment, the angular reference jig was slammed hard against its $+90°$ limit stop, which is made of hard rubber. The rubber has been carefully adjusted so that if the handle is laid on the stop gently or with moderate force it will rest at an angle of 90°. If excessive force is used, the rubber will compress and allow rotation a few degrees past the intended stop point. The A/D converter ignores the voltage increase beyond 2.5 V, and so the angular reference jig still reports 90°, but

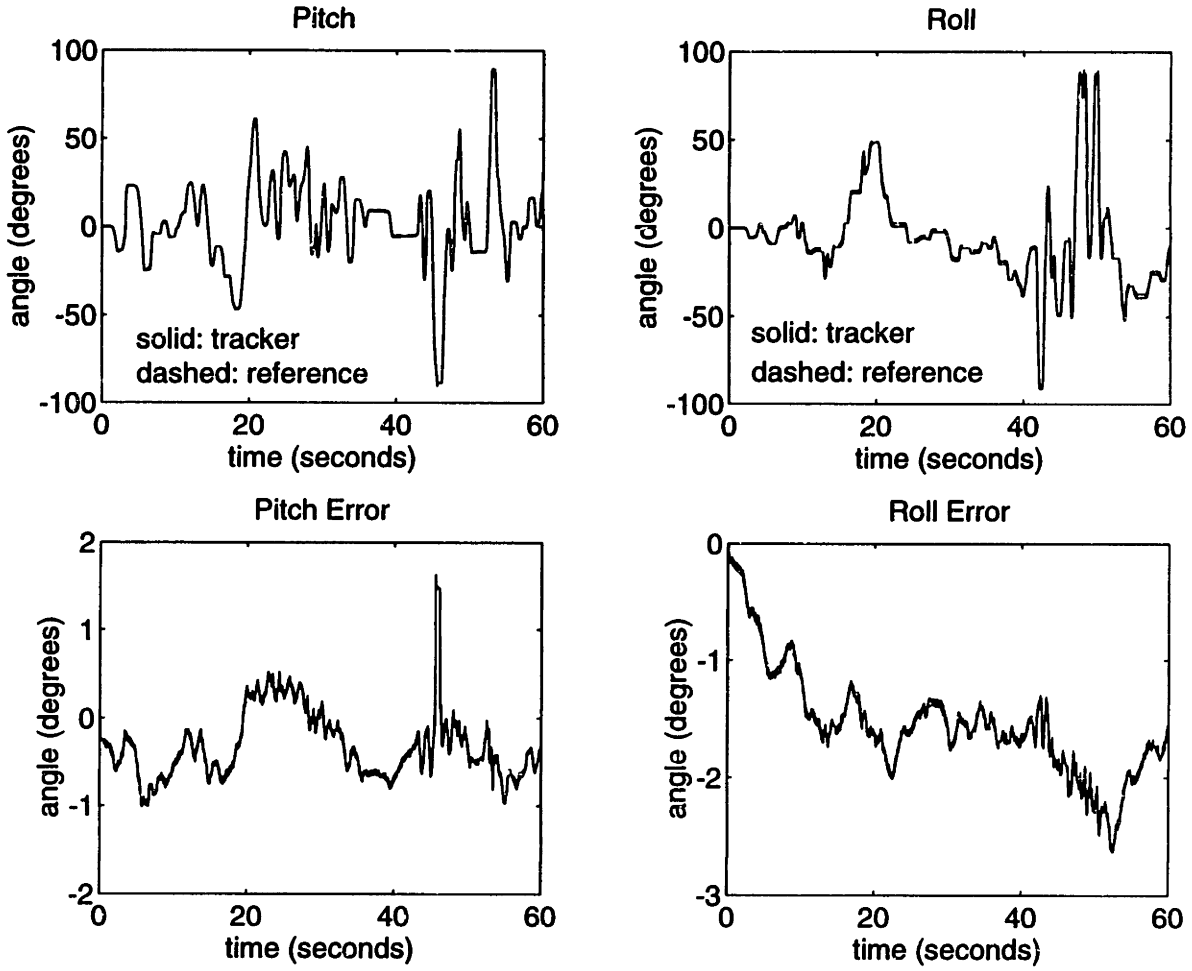


Figure 4-9: Accuracy during simulated "in-use" head movement.

the tracker has no such limit. The error, therefore is due to the angular reference jig, not the tracker. The results for roll are not as good as pitch, but still quite acceptable. Note that the error is always negative, so by adding a constant offset to the tracker roll output the maximum error could be reduced from 2.5° to half that much.

Two additional experiments were done to determine whether accuracy is worse during fast or quasi-static input motions. The former would indicate difficulties in integration of the angular rate sensors, while the latter would indicate inaccurate readings of the fluid inclinometer. Both experiments were done using the pitch output, since it seems to work better. The fast input motion experiment was done with a record length of 512, 145.65 Hz update rate and no decimation, so it only lasted a few seconds. The tracker was shaken fast and furiously during this time, with peak angular rates of about 800°/s. The quasi-static experiment uses data from Figure 4-5.

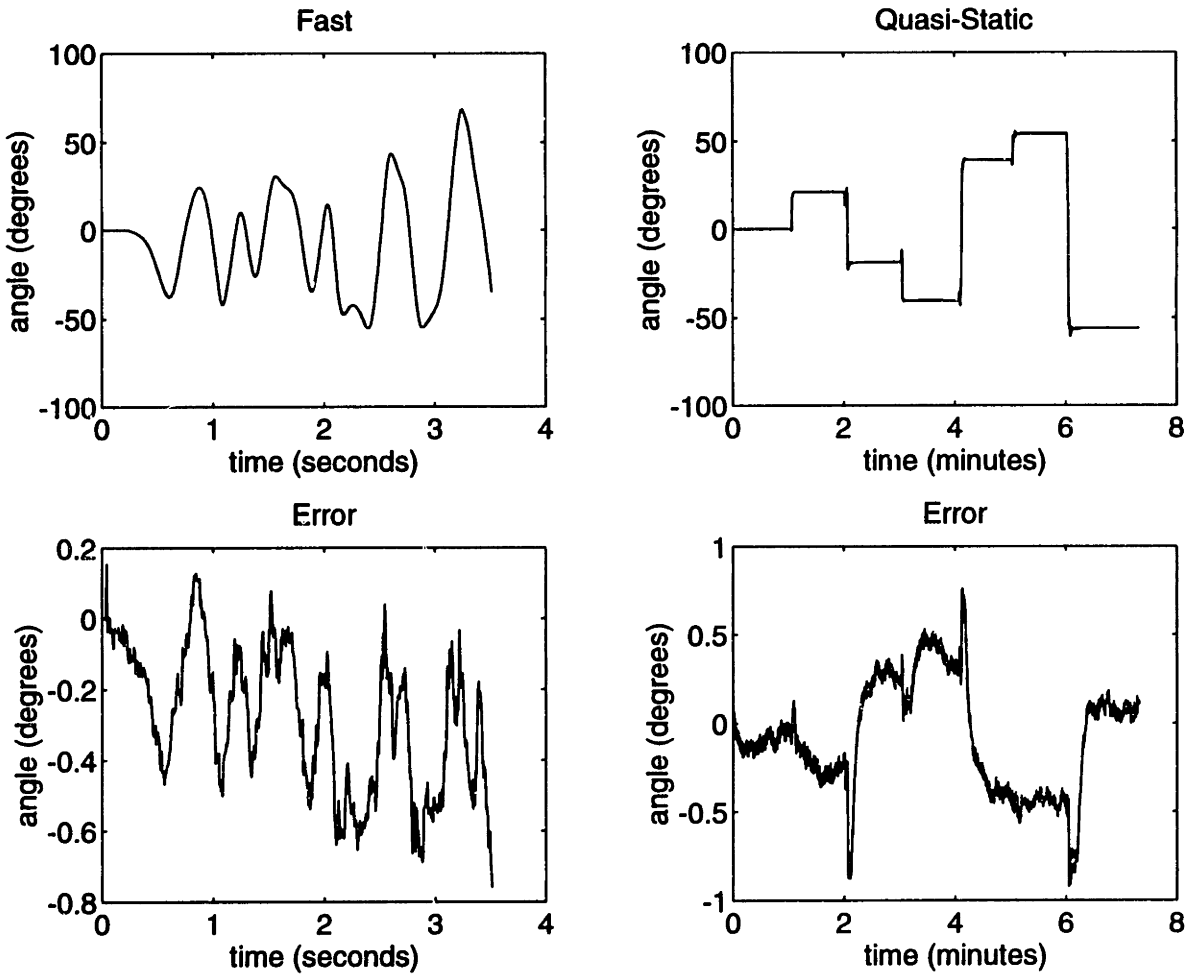


Figure 4-10: Pitch accuracy during fast and quasi-static tracking conditions.

The results from both experiments are presented in Figure 4-10, using the same format as Figure 4-9. Neither experiment reveals any particularly surprising errors, although the quasi-static error graph does show nicely that the largest errors occur during large-angle rotations.

4.2.3 Cross-Axis Sensitivity

The previous two sections concentrated on errors in the tracker output for the DOF that is being stimulated. However, rotations about one axis may also cause some response in the other two outputs, which is, by definition, error. This cross-axis sensitivity is inevitable if the angular rate sensor sensitive axes are not perfectly orthogonal. Since the alignment of the GyroChip sensitive axis within its casing is

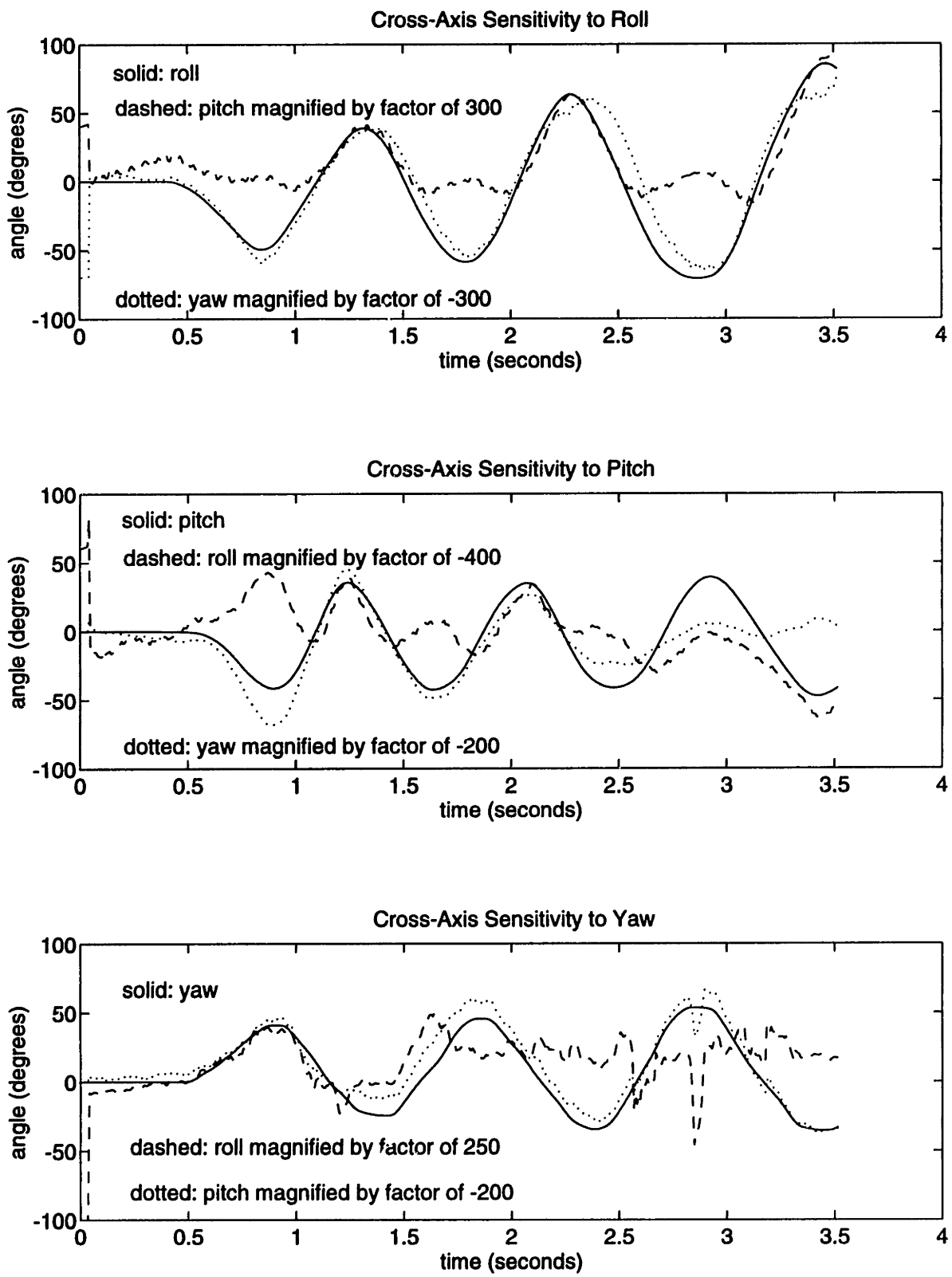


Figure 4-11: Cross-axis sensitivity.

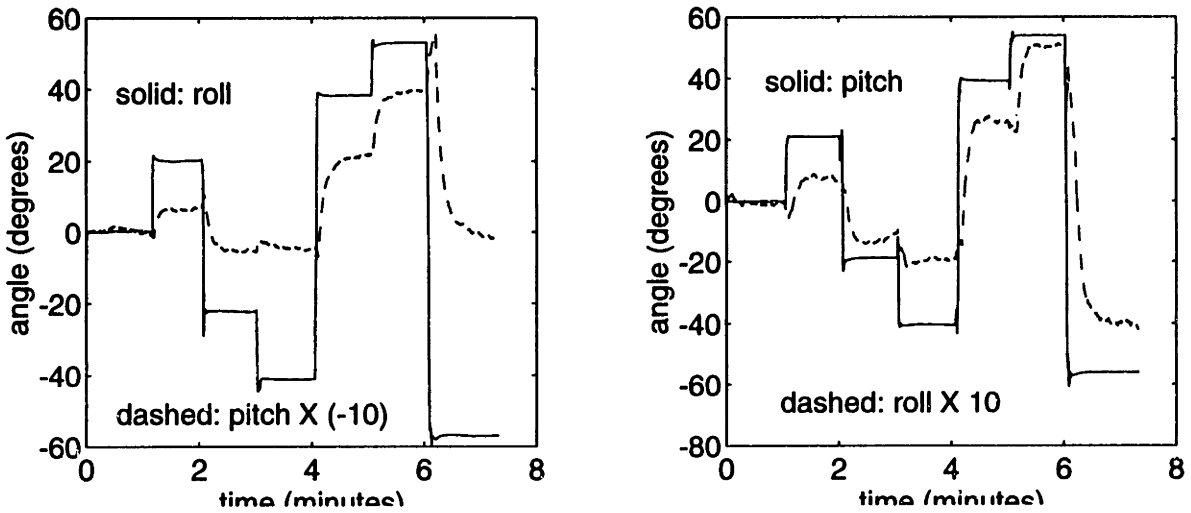


Figure 4-12: Axis misalignment of fluid inclinometer.

only specified to within 0.5° = 0.008 radians, a maximum cross-axis sensitivity of 0.8% should be expected. To verify this, the tracker was wiggled about each axis while the outputs of all three axes were recorded, again using record length 512, 145.65 Hz, and no decimation. The outputs of the two unstimulated axes were magnified until their wiggles appeared to be roughly the same size as those of the stimulated axis. The resulting plots, including the magnification factors, for all three experiments can be seen in Figure 4-11. It should be noted that the dashed and dotted curves were not only scaled, but also shifted vertically to compensate for the effect of magnifying the ordinary errors not produced by cross-axis sensitivity. The cross-axis sensitivities seen in Figure 4-11 are less than 0.5%, which means that the careful machining of all the bracket surfaces to achieve orthogonality was successful.

Another source of cross-axis sensitivity when drift compensation is enabled is misalignment of the fluid inclinometer bubble with respect to the sensor assembly. Figure 4-12 shows that this problem is severe and something should be done about it. This is not surprising, considering that the bubble is tiny and round, and was aligned visually. An important tune-up procedure that was neglected is to repeatedly adjust the orientation of the bubble until the cross-axis sensitivity measured in Figure 4-12 is reduced within acceptable limits.

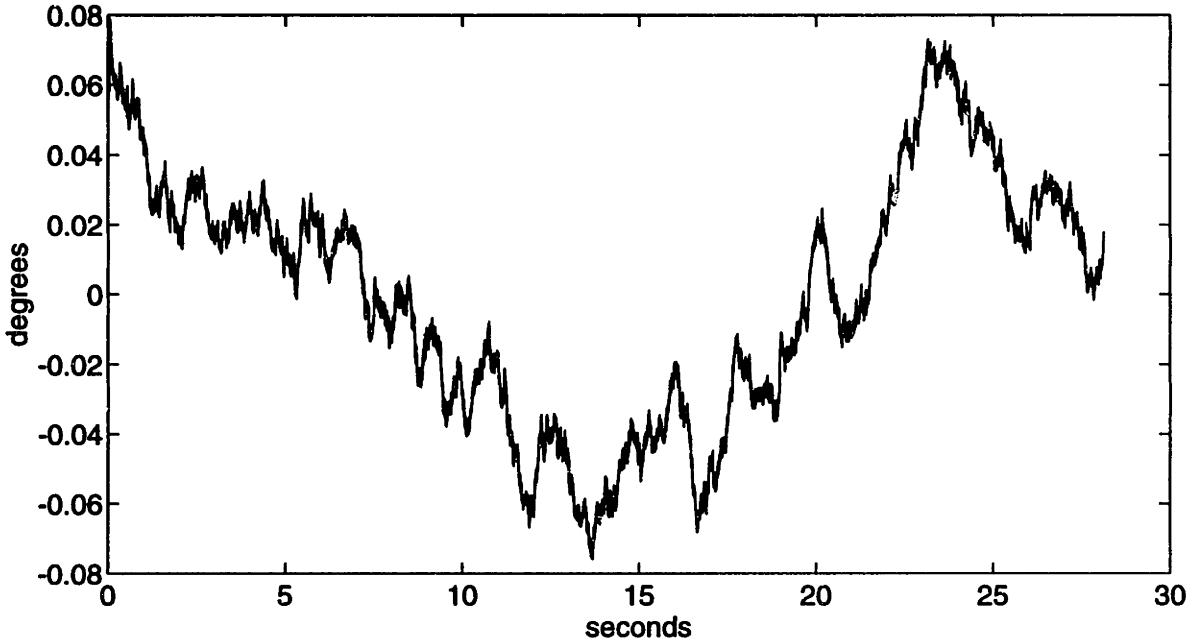


Figure 4-13: Tracker output noise while at rest.

4.3 Resolution and Noise

Figure 4-13 shows a highly magnified view of the noise sample that was obtained for the purpose of evaluating the resolution of the tracker. This data record, which will be used throughout this section, comprises 4096 samples of the yaw output taken at a sampling rate of 145.65 Hz with both attitude and heading drift compensation enabled. The mean has been subtracted, but the data is otherwise unprocessed. The figure clearly illustrates the problem which was discussed in Section 2.1.2: how should the amplitude be expressed for a noise whose variance grows with time? Is the amplitude the size of the largest up and down meanderings, about $\pm 0.08°$, or is it the "thickness" of the line that goes up and down. If the latter, it is very hard to state the "thickness" unambiguously because it depends on the time scale used to plot the signal.

The drift compensation provides a partial solution to the problem. Because the variance does not grow unbounded, the signal can be treated as quasi-stationary over a time period very long compared to the timecourse of drift compensation. By assuming stationarity and ergodicity, the power spectral density (PSD) was estimated according

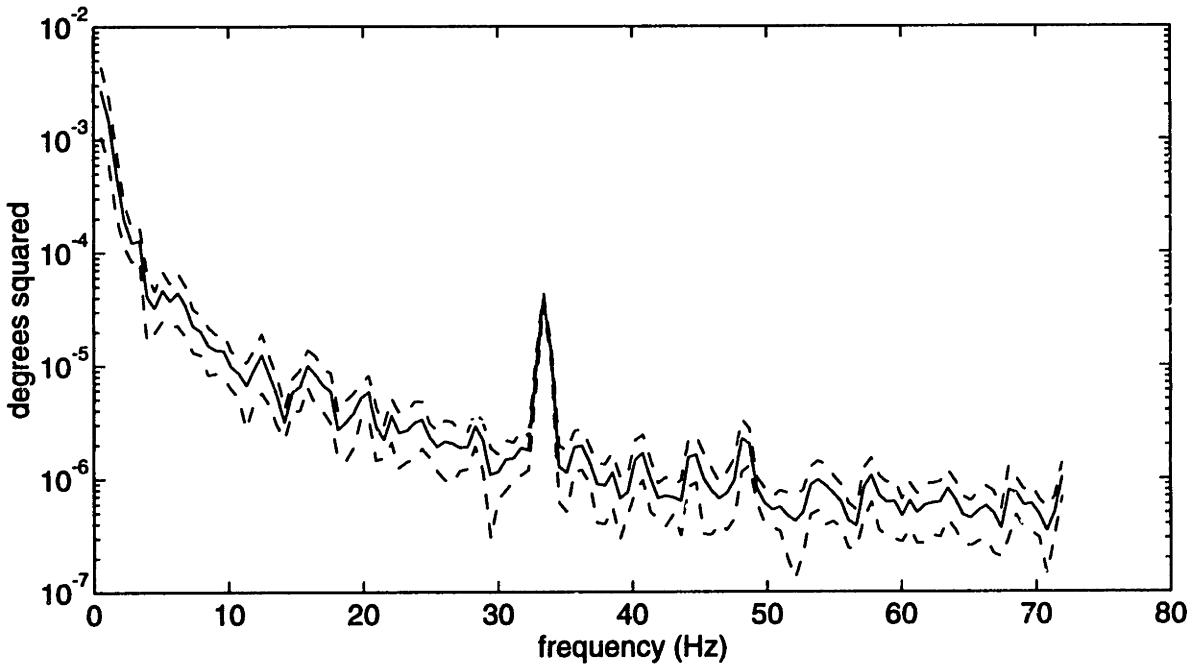


Figure 4-14: Noise power spectral density function.

to the Welch method [19] with 256-point sections, and is displayed in Figure 4-14 together with its 95% confidence intervals. The PSD reveals that more than half of the noise energy is concentrated below 1 Hz, which explains why the noise trace looks like a meandering river more than the straight fuzzy caterpillar typical of white or pink noise. It is not known what causes the spike at 33 Hz.

It seems unreasonable to include the very slow fluctuations as "noise" because noise is intended to be a measure of the fast fluctuations in the output of the head-tracker that would lead to distracting scene jitter. As argued in Section 2.1.2, scene rotations slower than 0.05°/s are not even perceptible, let alone distracting. An intuitively reasonable definition of noise for this application is the signal that remains after subtracting from the raw noise signal that low-frequency component which varies more slowly than 0.05°/s. It was found empirically that lowpass filtering the noise with a cutoff frequency of 0.35 Hz extracts a low-frequency component whose derivative has maxima just under 0.05°/s. Figure 4-15 shows the original noise, the lowpass filtered version of the noise, and the highpass version of the noise obtained by subtracting the lowpass version from the original. The RMS amplitude of this "de-drifted"

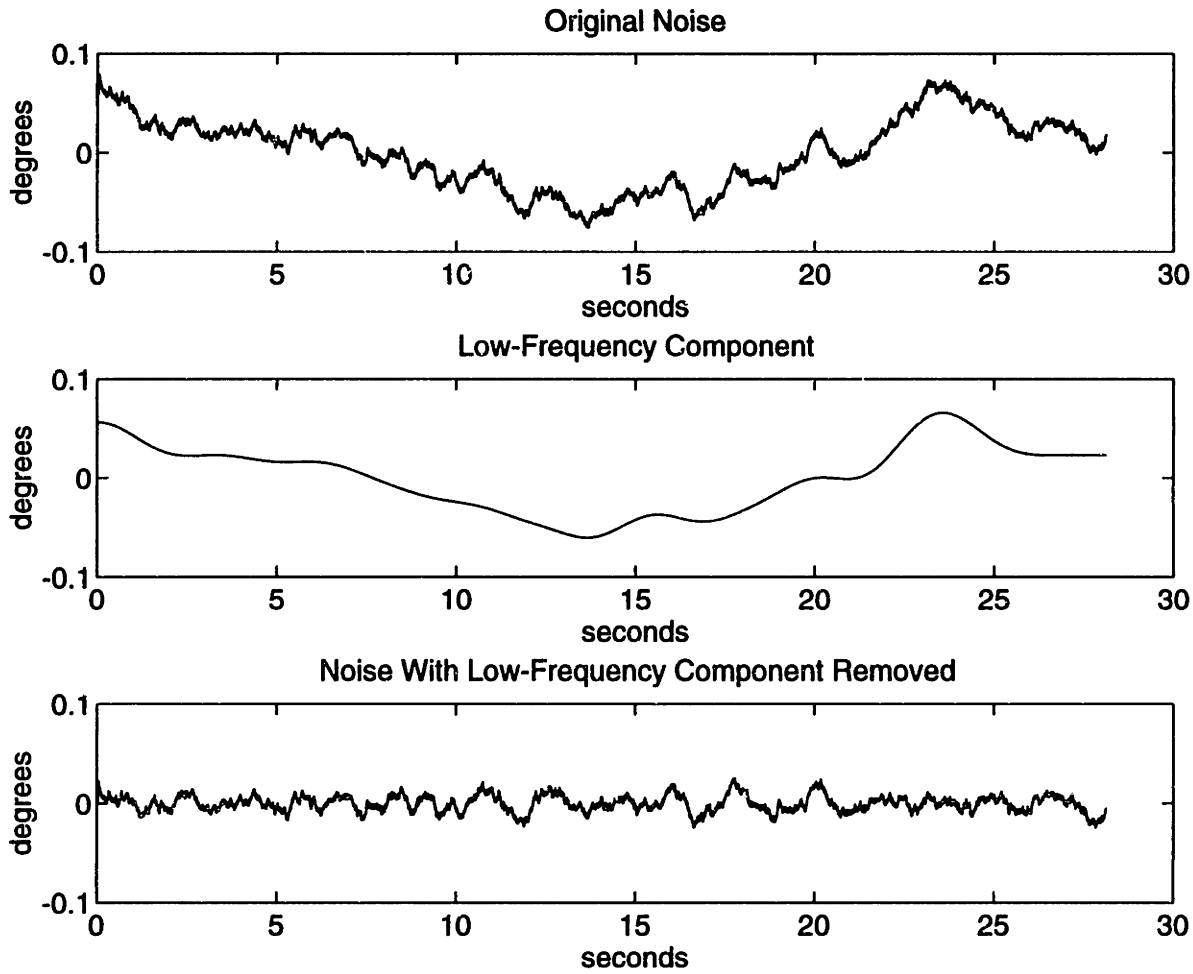


Figure 4-15: Filtering to obtain a more reasonable noise characterization.

noise, 0.0082°, will be taken as the resolution of the head-tracker. By a lucky coincidence, this just meets the most stringent of the desired resolution specifications from Table 2.2, and is comparable to the resolution of a mechanical tracker with 16-bit conversion.

4.4 Dynamic Performance

As discussed in Section 2.1.2, excessive head-tracker delay has a profoundly destructive effect on VE realism, and can even induce nausea and vomitting. The need for a fast head-tracker is one of the major motivations for this study of inertial head-tracking. It is therefore particularly important to correctly evaluate the tracker's dynamic performance. According to Section 2.1.1, complete information about the

dynamic performance is contained in the system function if it is an LTI system. Because the head-tracker is designed to approximate a system $\vec{w}(t) = \vec{v}(t)$, and does a reasonably good job, it is modelled in this section as an LTI system with three inputs (Euler angles describing actual head orientation) and three outputs (measured Euler angles). According to the results on cross-axis sensitivity in Section 4.2.3, the interaction between the three Euler angles is small, and the system can be further simplified into three separate LTI systems, one for each DOF:

$$\begin{aligned} w_\phi(t) &= v_\phi(t) * h_\phi(t) \\ w_\theta(t) &= v_\theta(t) * h_\theta(t) \\ w_\psi(t) &= v_\psi(t) * h_\psi(t). \end{aligned}$$

Since the three system functions, $H_\phi(f)$, $H_\theta(f)$ and $H_\psi(f)$, are expected to be basically the same, it is only necessary to evaluate one of these system functions. Therefore, the subsequent evaluation will focus entirely on determining $H_\theta(f)$. The subscript θ will be dropped for convenience, and it is to be understood that $v(t)$ and $w(t)$ refer to actual and measured pitch. The most important characteristics of the system function are summarized by two specifications: lag and bandwidth.

It is possible to predict an upper bound on the lag of the system before actually measuring the system function, simply by examining the "mainloop" section of the function main() in intest.c. The program waits until an interrupt occurs, then 1) reads the sensors, 2) processes the data to arrive at new Euler angle estimates, 3) displays the new Euler angles, and 4) waits for the next interrupt. The only latency in the output is the time it takes to perform steps 1) and 2), which is the same every time. When step 2) completes and new Euler angles are exported, those angles reflect the orientation input at the moment step 1) was performed. There is no filtering to impose additional delays. By adjusting the speed of the timer that generates the interrupts, the time spent waiting in step 4) can be reduced to almost zero. Using the current version of the software, listed in Appendix A, this adjustment results in an interrupt frequency of 145.65 Hz. This means that the total time consumed by

steps 1), 2), and 3) is 6.87 ms. Since the graphics operations in step 3) may consume a significant portion of this time, this number should definitely be thought of as an upper bound on the lag.

There are several experimental methods to measure the frequency response of a system. If the input is electrical, an impulse or white noise signal may be introduced at the input, while recording the output. Fourier transforming the output then yields the frequency response directly. If the input is mechanical, as for a head-tracker, it is often too difficult to approximate an impulse or white noise with sufficiently flat spectrum. One approach which has actually been used for evaluating other head-trackers [1] is to record the amplitude and phase response to sinusoids at a handful of different frequencies and plot the results against frequency. Since the testing apparatus and software used in this project makes it easy to record both $v(t)$ and $w(t)$ for the same time period $0 \leq t \leq T$, the approach chosen is simply to use

$$H(f) = \frac{W(f)}{V(f)}$$

where $V(f)$ and $W(f)$ are the discrete-time Fourier transforms of $v(t)$ and $w(t)$, computed using a radix-2 FFT algorithm. The benefit of this method is that the excitation signal, $v(t)$, may be anything which contains energy at all the frequencies of interest. Since the motion is generated by hand, it is too difficult to generate sinusoids, and this is the most practical method.

An effort was made to generate good quality step functions, because step functions are known to be wideband. A hammer padded with a thin sheet of rubber is used to tap the handle of the angular reference jig, initially at rest at 0°. The rotation ends equally abruptly when the handle hits a rubber-coated stop at about 50°. With practice, it is possible to hammer with just the right amount of force to cause rotation near the maximum trackable velocity of 1000°/s. Figure 4-16 shows a nice step function (left top) and its frequency response magnitude (left bottom). Although the motion looks quite step-like, it is not very wideband. The top right graph shows an alternative excitation waveform that was generated by manually shaking the appara-

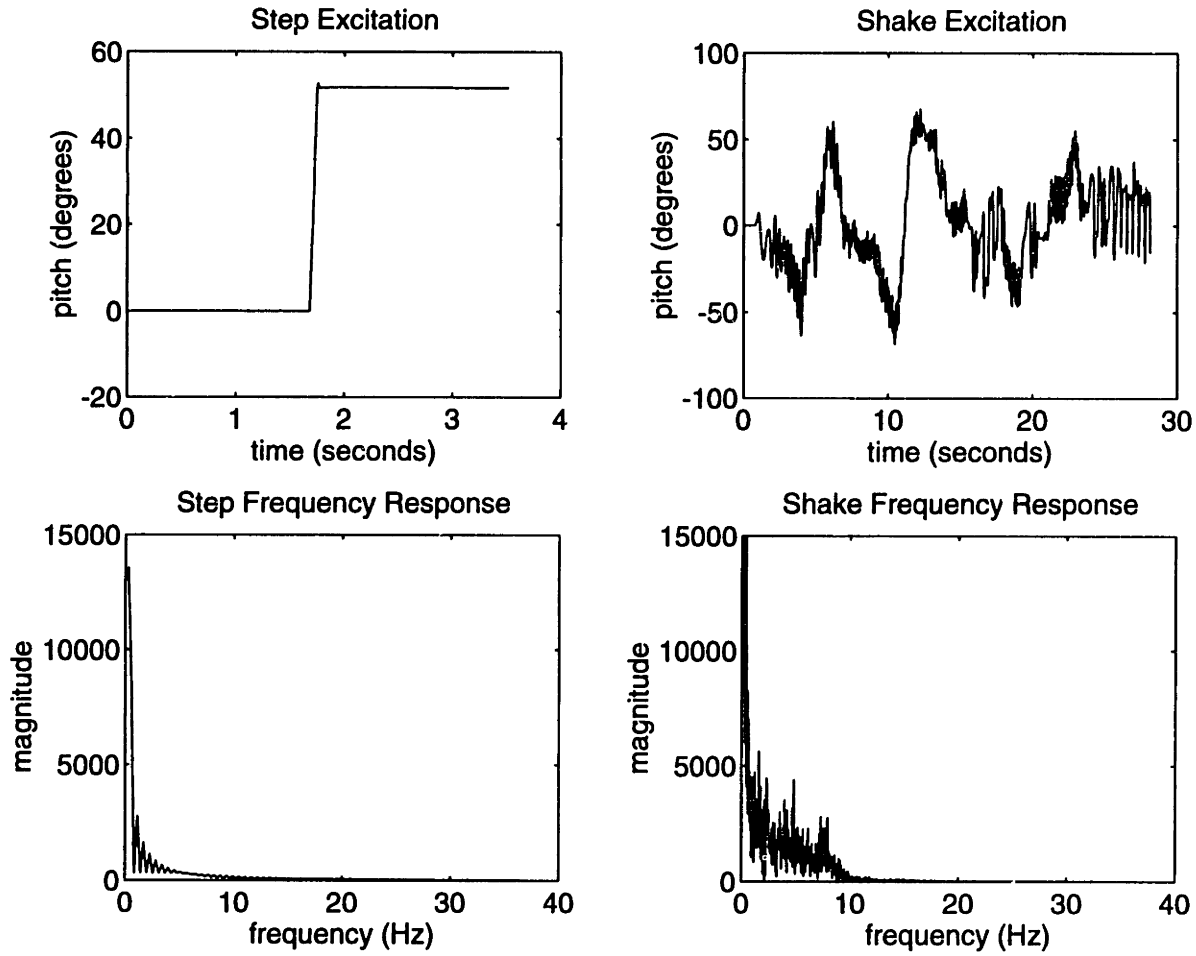


Figure 4-16: Comparison of two excitations used to obtain the system function.

tus as vigorously as possible at a variety of frequencies. The bottom right shows the frequency response magnitude, which contains somewhat more high-frequency energy than does the step excitation. For this reason, the "shake" excitation in Figure 4-16 is the one used for determining the system function.

Figure 4-17 shows the system function obtained in the manner described above. Up to about 10 or 15 Hz, the magnitude is 1 and the phase is 0, as desired. At higher frequencies both the magnitude and phase become sporadic. This is because the excitation contains insufficient energy at high frequencies; it does not indicate that the bandwidth of the tracker is only 15 Hz. Given that the bandwidth of the GyroChips is 100 Hz, it is expected that the system function of the tracker would be flat out to the Nyquist frequency of 70 Hz if it could be measured that far. It is not, however, important to demonstrate flat response all the way out to 70 Hz. In

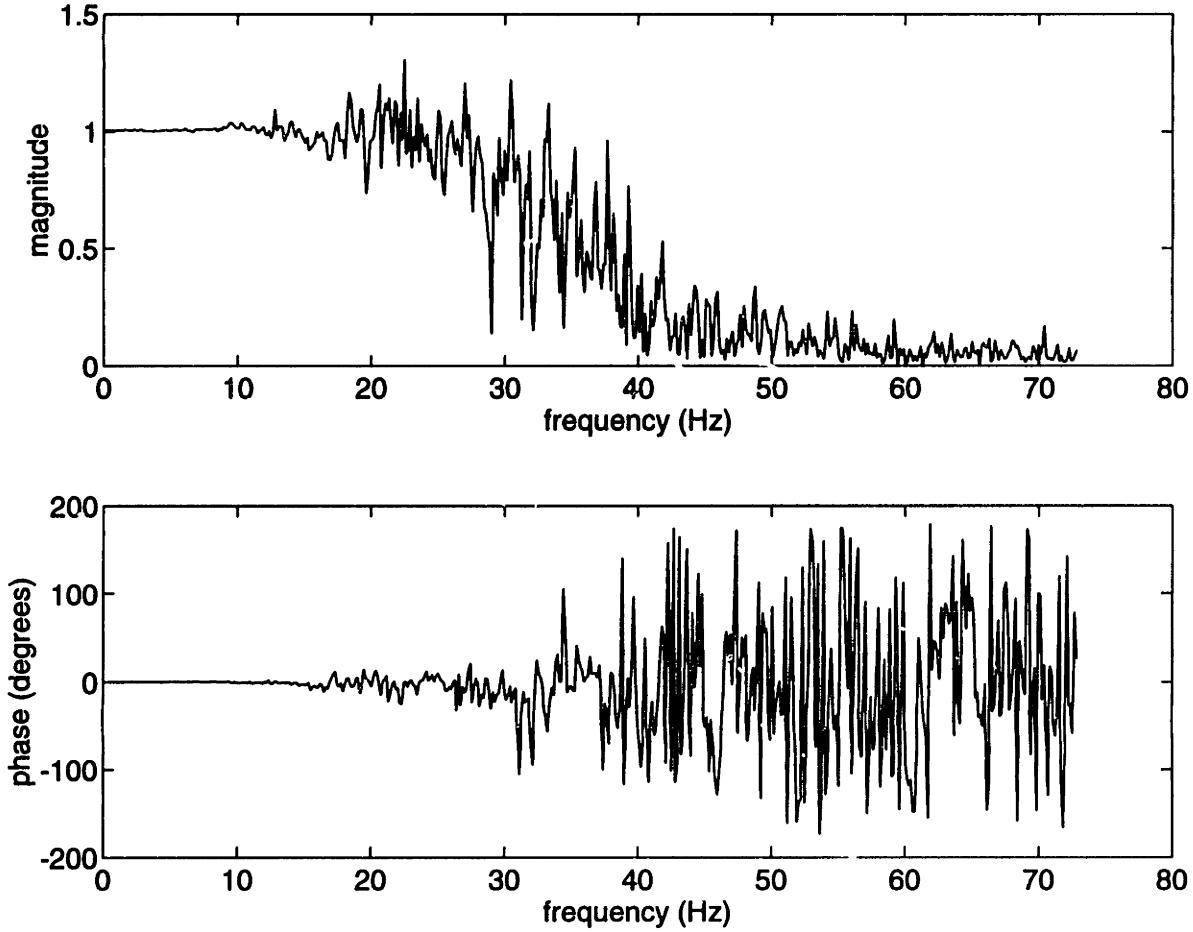


Figure 4-17: Transfer function magnitude and phase for pitch axis.

a paper specifically on testing dynamic performance of head-trackers, Adelstein [1], only tested frequency response up to 3.5 Hz, citing the bandwidth of human volitional movement given in [12] as 3 Hz. Both the Polhemus Isotrack and Logitech 6-D Mouse showed significant gain rolloff below this frequency. In order to further justify the use of system functions that are only valid to 15 Hz, an additional experiment was performed using the inertial tracker mounted on a headband to record 30 seconds of frenzied head-wagging. Fourier transforming the outputs, it was found that the highest frequencies were produced in the yaw output, and died off almost completely by about 8 Hz.

Having determined from the system function magnitude that the tracker bandwidth is more than adequate for tracking human head motion, it is also possible to obtain an empirical measure of the system lag by looking at the phase response in

Figure 4-17. A least squares linear regression to the phase from 0 to 10 Hz has a slope of 0.037°/Hz which corresponds to a lag of 0.10 ms. This result is surprising because it is so much less than the upper bound determined by adjusting the update rate. The implication is that the vast majority of the 6.87 ms duration of each update cycle is spent on the graphical display of three lines to indicate the roll, pitch and yaw angles. To verify this finding, the "display_angles" function was commented out of the main loop and the "TIMER_PERIOD" constant was reduced until the program ran at an update rate of 932 Hz. The fact that it still worked under these conditions makes the submillisecond lag result believable. The design goals for dynamic performance have clearly been achieved.

Chapter 5

Conclusions

5.1 Discussion of Results

Table 5.1 summarizes the results from the evaluation in Chapter 4, and compares them to the desired specifications set forth in Table 2.2. The results show that the prototype inertial orientation tracker was in most ways very successful at meeting the head-tracking needs that have not been met by mechanical, acoustic, magnetic and optical systems.

The range of the inertial tracker is potentially larger than any other head-tracker yet devised. All possible orientations are tracked, although for drift correction to occur the user must occasionally pause with roll and pitch angles inside the range $\pm 57°$. Positional range is also unrestricted. By using long enough wires, the user may wander freely throughout a room of any size. If the application requires the user to travel even greater distances, such as throughout a large building or outdoors, the wires can be eliminated by telemetry. When using this head-tracker, the range-limiting factor is always the HMD, not the tracker. An HMD requires two high-bandwidth cables to bring a stereo pair of high-resolution images from the graphics rendering computer to the user's head. Where long range operation is desired, these cables may also be eliminated by video telemetry[1], but due to their high-bandwidth,

[1]This has been demonstrated by the author using two inexpensive video transmitter/receiver pairs.

Specification	Desired	Achieved
Range	yaw: ±180° pitch: +50°, −70° roll: ±35° position: room size	yaw: ±180° pitch: ±90° roll: ±90° position: unlimited
Velocity	±1000°/s	±1000°/s
Acceleration	±6000°/s^2	> 6000°/s^2
Static Accuracy	opaque HMD: yaw: no requirement pitch & roll: 15° max drift rate: 3°/min see-through HMD: 0.009°	pitch & roll: 0.5° yaw: < 1°/min drift
Resolution	opaque HMD: 0.03° see-through HMD: 0.009°	0.0082° RMS
Bandwidth	20 Hz	tested to 15 Hz probably flat to 70 Hz
Latency	1 ms	0.1 ms

Table 5.1: Summary of desired and achieved performance for orientation tracking.

the video telemetry transceivers will place tighter restrictions on range than will the head-tracker telemetry. To be fair, the inertial orientation-tracker achieves unlimited range in position by not tracking position. In extending the inertial head-tracking concept to 6 DOFs. one of the main challenges will be to preserve its large range. It is likely that translational range will have to be compromised somewhat in order to obtain external position fixes for drift cancellation.

Another extremely important advantage of inertial tracking which is succesfully demonstrated by the prototype is speed. The lag is certainly no greater than 1 ms, as an update rate of 1 kHz is possible with the graphics disabled. According to an analysis of the phase response, the effective lag is only 0.1 ms. The bandwidth is also more than adequate for human motion-tracking applications. The frequency response could unfortunately only be measured up to about 15 Hz because it is impossible to manually generate input motions with enough high frequency energy. However, based on an analysis of the design, there is reason to suspect that the frequency response is actually flat out to 70 Hz. Mechanical trackers are the only other technology

which can provide comparable dynamic performance, but they are cumbersome and range-restrictive.

The noise and resolution results are also very encouraging. Because the inertial tracker integrates the sensor data, the resolution of the outputs is much finer than the resolution of the A/D converters used to sample the sensor data. Even with 11-bit effective conversion precision, the high frequency noise in the outputs is only 0.008° RMS or 0.05° peak-to-peak. By comparison, the orientation jitter shown in [18] for a Polhemus Isotrak appears to be on the order of 0.3° peak-to-peak.

The accuracy results are mixed and present greater problems for interpretation. On the one hand, the static accuracy for pitch is at least within 0.5°, and possibly much better, since the 0.5° errors could be due to nonlinearity of the potentiometer used for calibration and testing. On the other hand, the roll accuracy is inferior to the pitch accuracy, the yaw has no accuracy, and the cross-axis sensitivity to DC displacements is very high. The fact that the roll accuracy is not as good as the pitch accuracy for the current implementation does not indicate an important problem with inertial head-tracking. By spending some time carefully troubleshooting the system, it should be easy to locate the problem and correct it. The most likely cause of the errors is some noise or instability in the electronics which may even be curable by the strategic addition of a capacitor. The magnitude of the cross-axis sensitivity problem is much larger, but the solution is also fairly simple. The fluid inclinometer bubble needs to be aligned with the axes of the sensor mounting bracket within a fraction of a degree. This cannot be done by eye. A method must be developed for adjusting the alignment to the necessary accuracy. The procedure could be as simple as mounting the sensor assembly on the jig in its pitch testing orientation and then adjusting the bubble to minimize the roll axis output changes while the jig is swung back and forth.

The problem of unbounded yaw drift is not as superficial as the other two accuracy problems mentioned. Although the yaw drift rates of about 100–300°/hr achieved by this prototype are perfectly tolerable for opaque VE applications with no interaction with the real environment, the drift accumulated after several minutes would be enough to make even haptic interactions with the environment confusing. The first

avenue of attack on this problem is to try to reduce the drift rate. As the drift rates intrinsic to the Systron-Donner angular rate sensors are two orders of magnitude smaller than those achieved, it should be easy to make significant reductions. The first thing to change is the A/D converters. With 12-bit resolution, a single bit represents $\frac{1}{4096} \times 1000°/s = 88°/s$. The noise level of 3–4 LSBs during the drift recording experiments may therefore fully account for the drift. A 16-bit A/D converter with true accuracy of $\pm\frac{1}{2}$ LSB would bring the grounded-input drift rate down to 3°/hr. In order to maintain this low drift when the sensors are connected to the converter inputs, it would be prudent to add anti-aliasing filters at the front end of the A/D converters. The needed anti-aliasing filters are lowpass zero-phase filters with a passband from 0 Hz to at least 20 Hz. The update rate should also be adjusted to make sure the Nyquist frequency is well into the stopband of the anti-aliasing filters.

The second avenue of attack on the yaw drift problem is to try to find a succesful method of drift correction. The magnetic compass idea definitely deserves further consideration. It may turn out that only in M.I.T. Building 36 is the earth's magnetic field so seriously disrupted that it cannot be used. If the problem turns out to be universal and the earth's magnetic field cannot be used to good effect, then heading information will have to be provided by whatever secondary tracking system is used to provide translational drift correction.

One final specification of the current prototype which deserves some comment is its weight. When attached to the HMD shown in Figure 3-3, the tracker does not really increase the discomfort. The HMD, which was designed by the author for use with the inertial head-tracker, has almost all of its mass concentrated in front. Attaching the tracker to the back helps to counterbalance it and relieve some pressure from the user's nose. However, with the next generation of lightweight HMDs, a 1 lb. tracker will really be too heavy. This problem is not at all intrinsic to inertial techniques for head-tracking. In fact, the next generation GyroChips have 1/3 the size and weight and are already being marketed.

5.2 Directions for Further Work

The purpose of this Master's thesis is to evaluate the potential usefulness of inertial navigation techniques for VE and teleoperator head-tracking. Towards this end, a first prototype inertial head-tracker has been built and evaluated. The results are encouraging and indicate excellent potential for the use of inertial techniques in head-tracking. Many directions for further research are opened by the success of the initial prototype. Further work is planned in three principle areas: 1) development of a second prototype orientation tracker, 2) incorporation of the inertial tracker into a complete VE system, and 3) extension to 6 DOFs.

The purpose of developing a second prototype orientation tracker is to progress from the research stage into the production of a practical inertial head-tracker which can be easily incorporated into any standard VE system and used in real applications. The second prototype will incorporate all the improvements and refinements suggested in the previous sections. The sensor assembly will be of greatly reduced size and weight and will be equiped with standard mounting holes so that it may be screwed onto any HMD designed to accomodate a Polhemus sensor. All the electronics will be integrated on a printed circuit card to be housed in an expansion slot of the PC, so that the entire head-tracker system will consist of a PC and a small sensor block with one long thin cable that plugs into the back of the PC. Anti-aliasing filters and 16-bit A/D converters will be used to greatly reduce drift. If it is found to be effective in other buildings, the sensor block will also include a magnetic heading sensor for yaw drift compensation. The sensors will be carefully aligned to eliminate the cross-axis sensitivity problem that arose with the first prototype, and additional calibration work will be done to improve accuracy. Finally, the software will be extended to provide interface capabilities so that head tracker data may be transferred to a host computer running VE software. One interface that will certainly be provided is an rs-232 interface of the type used by Polhemus and almost all the other commercially available head-trackers. This will allow compatibility with existing VE software, and also permit direct performance comparisons to be made between the inertial

tracker and its rivals. The testing apparatus described in Chapter 3 will be modified to accomodate a variety of different head trackers and receive their outputs via rs-232. It will then be possible to test a variety of head-trackers, including the second prototype inertial tracker, using identical methods. If rs-232 communications are found to add significant lag, the tracker will also be provided with ethernet capability so that lag can be eliminated in VE systems programmed to receive head-tracker data over ethernet.

As soon as the new prototype is built and tested, priority will be given to incorporating it into a complete VE system and trying it out in a VE application. Although the specifications measured in Chapter 4 tell a great deal about the performance of a head tracker, much more can still be learned by actually using it for its intended purpose. Subjective evaluation of the inertial head-tracker in operation may uncover additional problems which have not been addressed by the evaluation procedures of Chapter 4.

The most important, and most challenging, direction for further work is the extension of the inertial head-tracker to track position as well as orientation. Orientation-tracking was chosen as a first goal because it is easier than inertial position-tracking. Having shown that inertial techniques are well suited for tracking orientation, it is still an open question whether they are useful for tracking position. The first question to answer is whether there is any hope at all for tracking position adequately using unaided inertial measurement. It is important to be very certain that the answer is negative before considering aided systems, because the devices used for aiding are bound to place some restrictions on range. If aiding is necessary, the next question to be researched is what type of aiding system should be used to lose as few of the advantages of inertial tracking as possible. The selection of an aiding technology will also be influenced by the outcome of research on magnetic heading sensing — if a succesful method of sensing yaw direction has still not been found, the aiding system will need to track more than 3 DOFs. Acoustic TOF measurement seems to be the most practical candidate for aiding, because it is simple and can measure over a fairly large range if speed is not important. However, no conclusions can be drawn

until some work is done with accelerometers to find out how frequently the position updates must be furnished.

If in the course of further research it is found that neither accelerometers alone nor accelerometers combined with an ultrasonic aiding system can provide a good solution to the position-tracking problem, this should not be taken to mean that the results obtained in this thesis are useless. Firstly, there are applications which only require orientation tracking, for which the inertial tracker is an ideal solution. Secondly, there are other position tracking methods without range restrictions which may someday be viable. For example, the "self-tracker" proposed by Gary Bishop [27] senses self-motion based on optical flow generated by cameras moving through an unstructured visual environment. The "self-tracker" requires computer vision which is currently beyond the state of the art, but if reliable orientation data were available from an inertial subsystem, the computations involved could be simplified tremendously. An excellent solution to the 6 DOF tracking problem will someday be found, and it is the author's opinion that an inertial orientation tracker will be part of it.

Appendix A

Source Code

A.1 intest.c

```
/*  File:  INTEST.C

    INTRACK.C -- READS ANGULAR RATES FROM THREE ORTHOGONAL GYROCHIPS
        AND INTEGRATES BY EULER ANGLES TO OBTAIN ORIENTATION.
        ALSO READS A TWO-AXIS FLUID INCLINOMETER AND USES THE
        RESULT TO COMPENSATE FOR DRIFT. ANSWERS QUERIES FOR
        ORIENTATION OVER AN RS-232 INTERFACE.
    INTEST.C -- EVALUATION VERSION OF INTRACK. INSTEAD OF ANSWERING
        QUERIES OVER RS-232, IT WRITES DATA FILES FOR SUBSEQUENT
        OFF-LINE ANALYSIS OF THE TRACKER PERFORMANCE.

This program was designed to be used with a DT2814 data
acquisition board configured  for an input voltage range of -2.5 to 2.5 volts.
Measurements are made WITHOUT using LPCLAB. Channel scanning is performed
by program in response to system timer interrupts.

COMPILATION COMMAND LINE --
    CL /c /AH INTEST.C
    CL /c /AH 0717.C

LINK COMMAND LINE --
    LINK INTEST+0717
*/

/* Include necessary C run-time library header files */

#include    <string.h>
#include    <stdio.h>
#include    <stdlib.h>
#include    <math.h>
#include    <float.h>
```

```
#include    <graph.h>
#include    <dos.h>
#include    <conio.h>


#define RINGLEN 16  /* length of ring buffer for data acquisition */
#define NUMCHANS 5  /* data acquired for channel 0 through 4 */
#define NUMEULERS 3 /* number of euler angles */
#define DTBASE  0x220   /* DT2814 card base address */
#define TICK    0x1C    /* system timer tick interrupt*/
#define INTNUM  TICK    /* chosen interrupt on which to install my ISR */
#define RATE_SCALE_FAC  0.000532632218  /* radians/sec per bit */
#define TIMER_PERIOD    0x20    /* the timer will count down from 0xNN00
                                   before causing a sytem tick interrupt
                                   (minimum 0x17 on 25 MHz 386/387)*/
#define QUARTER_SEC 1193/TIMER_PERIOD  /*interrupts per quarter second*/
#define RC      5.0     /* time constant for drift correction in seconds */
#define RANGE   2.0     /* defined in 0717.c */
#define PI      3.14159265359
#define WHITE   15
#define YELLOW  14
#define PURPLE  13
#define RED     12
#define DATALENGTH 4096 /* max. length of data records that can be recorded,
                           displayed, and written to file for analysis */

/* global variables */
extern double lut[3][4096]; /* from 0717.c */
unsigned int values[RINGLEN][NUMCHANS+1];  /* ring buffer for acquired data */
unsigned int biases[NUMCHANS+1];
char    head = 0, tail = 0; /* location of front and back of data queue */
double  trim_bias[3];       /* radians/sec drift rates measured during testrun */
double  motor_bias;
double  period;             /* time between interrupts in seconds */
int     stilltime;          /* time in interrupts the tracker has been at rest */
double  euler[NUMEULERS];   /* Angles in rads about X,Y,Z (world coordinate)
                               axes respectively, called roll, pitch, yaw */
double  euler_error[NUMEULERS];  /* angular drift detected */
double  euler_export[NUMEULERS]; /* mirror variable that can be read by other
                                    processes such as display or RS-232 I/O,
                                    as long as it is unlocked  */
char    locked = 0;     /* for preventing simultaneous access to variables*/
char    tiltcomp = 1;   /* drift compensation for pitch,roll on or off */
char    yawcomp = 0;    /* drift compensation for yaw on or off */
long    int_counter;
double  data[4][DATALENGTH];


/*function prototypes */
void (_cdecl _interrupt _far *save_old_handler)(void);
void _cdecl _interrupt _far int_handler(void);
void display_angles(int, int, double *);
void print_angles(int, double *);
void measure_biases(unsigned int *);
```

```
void update_eulers(unsigned short);
void update_stilltime(unsigned short, unsigned short);
void transfer(int, double *, double *);
void pause(void);
void resume(void);
void display_data(int);
void output_data(char *, int);
double motor_angle(void);

main()
{
    unsigned short changes = 0, oldtail = 0;
    char ch;
    short int color = WHITE;
    char filename[20];
    int data_high, data_low;
    long i;
    int record_length = 512;
    int decimator = 1;
    int trimming_now = 0;
    long trim_start_time;
    double trim_start_angle[3];

    /* initialize the system timer to a faster than normal rate */
    outp(0x43, 0x36);
    outp(0x40, 0x00);
    outp(0x40, TIMER_PERIOD);
    period = ((double)TIMER_PERIOD * 256.0)/1193180.0; /* 1.19 MHz clock*/

    /* initialize graphics */
    _setvideomode(_VRES16COLOR);
    _setbkcolor(_BLUE);
    _clearscreen(0);

    /* initialize the DT2814 board */
    outp(DTBASE + 0, 0x00);
    while(!(inp(DTBASE)&0x80));
    data_high = inp(DTBASE +1);
    data_low = inp(DTBASE +1);
    measure_biases(biases);

    /* install interrupt service routine */
    save_old_handler = _dos_getvect(INTNUM);
    _disable();
    _dos_setvect(INTNUM, int_handler);
    _enable();

    retrieve_raw_vals("0717.log");
    initialize_all_luts();

reset:
    for(i=0; i<NUMEULERS; i++){
        euler[i] = 0.0;
        euler_error[i] = 0.0;
```

```
        euler_export[i] = 0.0;
    }
    int_counter = (long)record_length * (long)decimator;

mainloop:
    _clearscreen(0);
    while(!kbhit()){
        oldtail = tail;
        while(tail!=head){
            changes++;
            tail++;
            tail%=RINGLEN;
        }
        if(changes){
            update_stilltime(oldtail, tail);
            update_eulers(changes);
            transfer(3, euler, euler_export);
            i = (int_counter - 1)/decimator + 1;
            if(i >= record_length){
                display_angles(WHITE, 3, euler_export);
            }
            else{
                data[0][i] = motor_angle() - motor_bias;
                data[1][i] = euler_export[0];
                data[2][i] = euler_export[1];
                data[3][i] = euler_export[2];
                display_angles(RED, 3, euler_export);
            }
            changes = 0;
        }
    }

menu:
    ch = getch();
    switch(ch){
    case 'r':   /* calibrate roll axis */
        _clearscreen(0);
        measure_raw_vals(3);
        initialize_lut(3);
        _clearscreen(0);
        goto mainloop;
        break;
    case 'p':   /* calibrate pitch axis */
        _clearscreen(0);
        measure_raw_vals(4);
        initialize_lut(4);
        _clearscreen(0);
        goto mainloop;
        break;
    case 'y':   /* calibrate yaw axis */
        _clearscreen(0);
        measure_raw_vals(5);
        initialize_lut(5);
        _clearscreen(0);
```

```
        goto mainloop;
        break;
    case 'S':   /* save calibration data */
        save_raw_vals("0717.log");
        goto mainloop;
        break;
    case 'R':   /* retrieve calibration data */
        retrieve_raw_vals("0717.log");
        initialize_all_luts();
        goto mainloop;
        break;
    case 's':   /* start over */
        goto reset;
        break;
    case 'b':   /* remeasure biases */
        measure_biases(biases);
        goto reset;
        break;
    case 'f':   /* fine trim biases */
        if(!trimming_now){
            trimming_now = 1;
            trim_start_time = int_counter;
            trim_start_angle[0] = euler_export[0];
            trim_start_angle[1] = euler_export[1];
            trim_start_angle[2] = euler_export[2];
        }
        else{
            trimming_now = 0;
            for(i=0;i<=2;i++){
               trim_bias[i]=(euler_export[i]-trim_start_angle[i])/
                (period * (int_counter - trim_start_time));
            }
        }
        goto mainloop;
        break;
    case '3':
        display_lut(3);
        goto mainloop;
        break;
    case '4':
        display_lut(4);
        goto mainloop;
        break;
    case '5':
        display_lut(5);
        goto mainloop;
        break;
    case 'l':   /* change length of test run */
        _clearscreen(0);
        printf("\ncurrent values are:");
        printf("\nrecord_length=%d, decimator=%d, ",
               record_length,decimator);
        printf("run will last %f seconds at %f Hz.",
               (float)record_length * (float)decimator * period,
```

```
            1.0/((float)decimator * period));
    do{
        printf("\nenter new record_length: ");
        scanf("%d", &record_length);
        printf("\nenter new decimator: ");
        scanf("%d", &decimator);
        printf("\nnew values are:");
        printf("\nrecord_length=%d, decimator=%d, ",
               record_length,decimator);
        printf("run will last %f seconds at %f Hz.",
               (float)record_length*(float)decimator*period,
               1.0/((float)decimator * period));
        printf("\nis this ok [y/n]?\n");
    } while(getch() != 'y');
    goto mainloop;
    break;
case 't':   /* perform test run */
    int_counter = 0;
    goto mainloop;
    break;
case 'd':   /* display data */
    display_data(record_length);
    goto mainloop;
    break;
case 'c':   /* clear data */
    for(i=0; i<DATALENGTH; i++)
        data[0][i]=data[1][i]=data[2][i]=data[3][i]=0.0;
    goto mainloop;
    break;
case 'o':   /* output data to a file */
    _clearscreen(0);
    printf("enter name of file in which to store data: ");
    scanf("%s", filename);
    output_data(filename, record_length);
    goto mainloop;
    break;
case 'a':   /* toggle attitude drift compensation */
    tiltcomp = 1 - tiltcomp;
    goto mainloop;
    break;
case 'h':   /* toggle heading drift compensation */
    yawcomp = 1 - yawcomp;
    goto mainloop;
    break;
case 'q':
    break;
default:
    _clearscreen(0);
    printf("r -- roll calibration procedure\n");
    printf("p -- pitch calibration procedure\n");
    printf("y -- yaw calibration procedure\n");
    printf("S -- save current calibration data\n");
    printf("R -- retrieve calibration data\n");
    printf("s -- reset angles and start over\n");
```

```
        printf("b -- remeasure biases, reset angles, start over\n");
        printf("f -- begin or end fine trimming of biases\n");
        printf("3,4,or 5 -- display lut for that channel\n");
        printf("l -- change length of test run and decimator\n");
        printf("t -- perform test run\n");
        printf("d -- display data\n");
        printf("c -- clear data\n");
        printf("o -- output data to file\n");
        printf("a -- toggle attitude drift compensation ");
            printf("(currently %d)\n", tiltcomp);
        printf("h -- toggle heading drift compensation ");
            printf("(curently %d)\n", yawcomp);
        printf("q -- quit\n");
        goto menu;
        break;
    }

goodbye:
    _disable();
    _dos_setvect(INTNUM, save_old_handler);
    _enable();

    outp(0x43, 0x36);
    outp(0x40, 0x00);
    outp(0x40, 0x00);

    _setvideomode(_DEFAULTMODE);
    exit(0);
}

/* INT_HANDLER:  this is the interrupt service routine. It must be short,
   so all it does is read all the A/D channels into the head of the ring
   buffer called "values". The main loop detects when there have been
   advances in the head of the ring buffer, and processes the new data. */

void _cdecl _interrupt _far int_handler(void)
{
    unsigned int data_high, data_low;
    int  channel;
    int i;

    head++;
    head%=RINGLEN;
    for(channel=0; channel<NUMCHANS; channel++){
        outp(DTBASE, channel);       /* start conversion */
        while(!(inp(DTBASE)&0x80));  /*wait till conversion finished*/
        data_high = inp(DTBASE+1);
        data_low = inp(DTBASE+1);
        values[head][channel] = 256*data_high + data_low;
    }
    int_counter++;
}

/* UPDATE_EULERS: When there has been new data acquired, the main loop calls
```

```
   this routine, passing the number of integration steps to be computed.
   This routine computes the increments and adds them "steps" times. It also
   subtracts off a fraction of the error for each step. */

void update_eulers(unsigned short steps)
{
    int  i;
    double  euler_dot[3];             /* Rates of change of euler angles
                                         in radians/sec. */
    double  rate[3] = {0.0,0.0,0.0};  /* Angular velocities in radians/sec
                                         about X,Y,Z body axes
                                         respectively, called P,Q,R or
                                         roll,pitch,yaw rate. */
    double  sin0, cos0, sin1, sec1, decrement;

    for(i=0; i<=2; i++){
        rate[i] =((double)values[tail][i] - (double)biases[i])
                  * RATE_SCALE_FAC - trim_bias[i];
    }
    sin0 = sin(euler[0]);
    cos0 = cos(euler[0]);
    sin1 = sin(euler[1]);
    sec1 = 1.0/cos(euler[1]);
    euler_dot[2] = rate[1]*sin0*sec1 + rate[2]*cos0*sec1;
    euler_dot[0] = rate[0] + euler_dot[2] * sin1;
    euler_dot[1] = rate[1]*cos0 - rate[2]*sin0;
    for(i=0; i<=2; i++){
        euler[i] += euler_dot[i] * period * steps;
        euler[i] -= (decrement= euler_error[i] * (period/RC) * steps);
        euler_error[i] -= decrement;
    }
}

/* UPDATE_STILLTIME: Keeps track of how long the outputs of the tilt sensor
   have been approximately constant while the angular rates have also been
   zero. If the stilltime reaches 1 second, the average values of the
   tilt sensor outputs over that 1/4 second are used to update the
   euler_error variables. */

void update_stilltime(unsigned short start, unsigned short finish)
{
    static long avg_ch3, avg_ch4;
    long ch0, ch1, ch2, ch3, ch4;
    long epsilon = 0x0080; /* this value was found by trial and error to
                              be just a little above the peak noise*/
    double bubbleroll, bubblepitch, compassyaw = 0.0;

    while(start != finish){
        stilltime++;
        ch0 = values[start][0] - (long)biases[0];
        ch1 = values[start][1] - (long)biases[1];
        ch3 = values[start][3];
        ch4 = values[start][4];
        avg_ch3=((stilltime-1)*avg_ch3 + ch3)/stilltime;
```

```
        avg_ch4=((stilltime-1)*avg_ch4 + ch4)/stilltime;
        if((labs(ch3-avg_ch3)>epsilon)||(labs(ch4-avg_ch4)>epsilon)
           ||(labs(ch0)>0x0040)||(labs(ch1)>0x0040)){
            stilltime = 0;
            }
        if(stilltime >= QUARTER_SEC){
            bubbleroll=lut[0][avg_ch3>>4];
            bubblepitch=lut[1][avg_ch4>>4];
            if((fabs(bubbleroll)<RANGE/2.0) &&
               (fabs(bubblepitch)<RANGE/2.0)){
               if(tiltcomp){
                   bubbleroll -= lut[0][biases[3]>>4];
                   bubblepitch -= lut[1][biases[4]>>4];
                   euler_error[0] = euler[0] - bubbleroll;
                   euler_error[1] = euler[1] - bubblepitch;
               }
               if(yawcomp){
                   euler_error[2] = euler[2] - compassyaw;
               }
            }
            stilltime = 0;
        }
        start++;
        start%=RINGLEN;
    }
}

/* MEASURE_BIASES: Average 5000 measurements of each A/D channel while
   the sensor assembly is in reference position. */

void measure_biases(unsigned int *biases)
{
    int channel;
    int numtrials = 5000;
    int trial;
    unsigned int data_high, data_low;
    long int accum[NUMCHANS+1];

    for(channel=0; channel<=NUMCHANS; channel++){
        accum[channel] = 0;
    }
    for(trial=0; trial<numtrials; trial++){
        for(channel=0; channel<=NUMCHANS; channel++){
            outp(DTBASE, channel);       /* start conversion */
            while(!(inp(DTBASE)&0x80));  /*wait till finished*/
            data_high = inp(DTBASE+1);
            data_low = inp(DTBASE+1);
            accum[channel] += 256*data_high + data_low;
        }
    }
    for(channel=0; channel<=NUMCHANS; channel++){
        biases[channel] = (unsigned int)(accum[channel]/numtrials);
    }
    trim_bias[0] = trim_bias[1] = trim_bias[2] = 0.0;
```

```
    motor_bias = (double)biases[NUMCHANS];
    motor_bias = -PI/2.0+(motor_bias*PI)/((double)0xffff);
}

/* DISPLAY_ANGLES: Simultaneous graphical display of angles on screen. */

void display_angles(int color, int numchans, double *vals)
{
    int channel;
    int XCENTER, YCENTER = 240;
    int RADIUS = 600/(2*numchans);
    int x,y;
    static int xold[NUMCHANS], yold[NUMCHANS];

    for(channel=0; channel<numchans; channel++){
        XCENTER = (2*channel+1)*(int)RADIUS;

        x = (int)((double)RADIUS * sin(vals[channel]));
        y = (int)((double)RADIUS * cos(vals[channel]));

        _setcolor(0);
        _moveto( XCENTER, YCENTER);
        _lineto( XCENTER + xold[channel], YCENTER - yold[channel]);

        _setcolor(color);
        _moveto( XCENTER, YCENTER);
        _lineto( XCENTER + x, YCENTER - y);

        /* We've plotted the point, now it's the old point */
        xold[channel] = x;
        yold[channel] = y;
    }
}

/* PRINT_ANGLES: For diagnostic printing of data. */

void print_angles(int numchans, double *vals)
{
    int i;
    printf("\n");
    for(i=0; i<numchans; i++){
        printf("%lf             ", vals[i]);
    }
}

/* TRANSFER: Copies one vector to another. */

void transfer(int numvals, double *source, double *dest)
{
    int i;

    if(!locked){
        locked = 1;
        for(i=0; i<numvals; i++){
```

```
            dest[i] = source[i];
        }
        locked = 0;
    }
}

/* PAUSE: During disk operations it is important to restore the system
   timer to its normal condition. */

void pause(void)
{
    _disable();
    _dos_setvect(INTNUM, save_old_handler);
    _enable();

    outp(0x43, 0x36);
    outp(0x40, 0x00);
    outp(0x40, 0x00);
}

/* RESUME: Used in conjunction with PAUSE. */

void resume(void)
{
    outp(0x43, 0x36);
    outp(0x40, 0x00);
    outp(0x40, TIMER_PERIOD);

    save_old_handler = _dos_getvect(INTNUM);
    _disable();
    _dos_setvect(INTNUM, int_handler);
    _enable();
}

/* DISPLAY_DATA: Graphs the first 600 entries in the data record obtained
   by selecting "t" (for "test") from the menu. As a mnemonic, (b)lue is
   used for (b)aseline, (y)ellow for (y)aw, (p)urple for (p)itch, and
   (r)ed for (r)oll. */

void display_data(int record_length)
{
    int x, y;
    int i, end;

    end = (record_length<600)?record_length:600;
    _setbkcolor(_BLACK);
    _clearscreen(0);
    for(x=0; x<end; x++){
        for(i=0;i<4;i++){
            y=240-(int)(data[i][x] * 140.0);
            _setcolor(i+11);
            _setpixel(x,y);
        }
    }
```

```
    while(!kbhit());
    getch();
    _setbkcolor(_BLUE);
    _clearscreen(0);
}

/* OUTPUT_DATA: Writes the data record obtained on the most recent test run
   to a file. */

void output_data(char *filename, int record_length)
{
    FILE *fp;
    int i,j;

    pause();

    if((fp = fopen(filename, "w")) == NULL){
        printf("\a\nCannot open file.");
        return;
    }
    rewind(fp);
    for(j=0; j<record_length; j++){
        for(i=0; i<4; i++){
            fprintf(fp, "%f ", data[i][j]);
        }
        fprintf(fp, "\n");
    }
    fclose(fp);
    resume();
}
```

A.2 0717.c

```
/*  File:   0717.c

    THIS FILE CONTAINS FUNCTIONS TO GENERATE AND ACCESS LOOK-UP TABLES
    FOR THE FREDERICKS 0717-2201 2-AXIS FLUID INCLINOMETER
*/

/* Include necessary C run-time library header files */

#include    <stdio.h>
#include    <math.h>
#include    <float.h>
#include    <dos.h>
#include    <graph.h>

#define RANGE  2.0  /* radians of usable range for each axis*/
#define RESOLUTION 101  /* number of angles measured per axis */
#define DELTA_THETA     RANGE/((double)(RESOLUTION-1))
#define PI      3.14159265359
#define DTBASE      0x220

/* global variables */
int raw_vals[3][RESOLUTION];
double lut[3][4096];

/* function prototypes */
void initialize_lut(int channel);
void initialize_all_luts(void);
void measure_raw_vals(int channel);
void save_raw_vals(char *filename);
void retrieve_raw_vals(char *filename);
double motor_angle(void);
void display_lut(int channel);


/* INITIALIZE_ALL_LUTS: Initialize all luts that are not empty. */

void initialize_all_luts(void)
{

    if(raw_vals[0][RESOLUTION/3])
        initialize_lut(3);
    if(raw_vals[1][RESOLUTION/3])
        initialize_lut(4);
    if(raw_vals[2][RESOLUTION/3])
        initialize_lut(5);
}

/* INITIALIZE_LUT: Convert the raw_vals obtained in the calibration
   procedure into the lut format using linear interpolation. */
```

```
void initialize_lut(int channel)
{
    int i;
    int this = 0, next;
    double this_test_angle, angular_increment;
    int this_raw_value, next_raw_value;


    while(this < RESOLUTION-1){
        this_test_angle = -RANGE/2.0 + (double)this * DELTA_THETA;
        next = this;
        do{
            next ++;
            this_raw_value = raw_vals[channel-3][this];
            next_raw_value = raw_vals[channel-3][next];
        }while(next_raw_value <= this_raw_value);
        angular_increment = DELTA_THETA/
                            (double)(next_raw_value - this_raw_value);
        for(i = this_raw_value; i<next_raw_value; i++){
            lut[channel-3][i] = this_test_angle +
                                (double)(i-this_raw_value) *
                                angular_increment;
        }
        this = next;
    }

    /* fill in the areas at the beginning and end of the lut */
    for(i=0; i<= raw_vals[channel-3][0]; i++)
        lut[channel-3][i] = -RANGE/2.0 - 0.01;
    for(i= raw_vals[channel-3][RESOLUTION-1]; i<4096; i++)
        lut[channel-3][i] = RANGE/2.0 + 0.01;
}

/* DISPLAY_LUT: For previewing the luts on the screen. */

void display_lut(int channel)
{
    int x, y;

    _clearscreen(0);
    _moveto(0,440);
    for(x=0; x<4096/8; x++){
        y = 240 - (int)(lut[channel-3][8*x] * 200.0 * 2.0/RANGE);
        _lineto(x,y);
    }
    while(!kbhit());
    _clearscreen(0);
}

/* MEASURE_RAW_VALS: The calibration procedure for determining the output
   of the tilt sensor as a function of pitch or roll. */

void measure_raw_vals(int channel)
{
```

```
    double theta = -RANGE/2.0;
    double a;
    int i = 0;
    int data_high, data_low;
    long  value, avg, j, previous;

    while(i < RESOLUTION){
        printf("\nMeasuring output for %lf radians.\n", theta);
        j = avg = 0;
        while(j < 200){
            printf("%lf\r", a = motor_angle());
            outp(DTBASE, channel);
            while(!(inp(DTBASE)&0x80));
            data_high = inp(DTBASE+1);
            data_low = inp(DTBASE+1);
            value = 256*data_high + data_low;
            if((fabs(a-theta) <= 0.001) &&
               (labs(value-previous) <= 0x0080)){
                j++;
                avg = ((j-1)*avg + value)/j;
                if(labs(value-avg) > 0x0080) j = 0;
            }
            previous = value;
            if(kbhit()) return;
        }
        printf("\n\a");
        raw_vals[channel-3][i] = (int)(((unsigned int)avg)>>4);
        printf("bubble raw value: %d\n", raw_vals[channel-3][i]);
        theta += DELTA_THETA;
        i++;
    }
}

/* MOTOR_ANGLE: Read the angular reference jig. */

double motor_angle(void)
{
    unsigned short data_high, data_low;
    unsigned int value;
    double angle;

    outp(DTBASE, 5);  /* the motor is on channel 5 */
    while(!(inp(DTBASE)&0x80));
    data_high = inp(DTBASE+1);
    data_low = inp(DTBASE+1);
    value = 256*data_high + data_low;

    angle = -PI/2.0 + (((double)value) * PI)/((double)0xffff);
    return(angle);
}

/* SAVE_RAW_VALS: The raw calibration data is stored instead of the
   precomputed luts because it is much more compact. */
```

```
void save_raw_vals(char *filename)
{
    FILE *fp;
    int i,j;

    pause();

    if((fp = fopen(filename, "w")) == NULL){
        printf("\a\nCannot open file.");
        return;
    }
    rewind(fp);
    for(j=0; j<RESOLUTION; j++){
        for(i=0; i<3; i++){
            fprintf(fp, "%d ", raw_vals[i][j]);
        }
        fprintf(fp, "\n");
    }
    fclose(fp);
    resume();
}

/* RETRIEVE_RAW_VALS */

void retrieve_raw_vals(char *filename)
{
    FILE *fp;
    int i,j;

    pause();

    if((fp = fopen(filename, "r")) == NULL){
        printf("\a\nCannot open file.");
        return;
    }
    rewind(fp);
    for(j=0; j<RESOLUTION; j++){
        for(i=0; i<3; i++){
            fscanf(fp, "%d ", &raw_vals[i][j]);
        }
    }
    fclose(fp);
    resume();
}
```

Bibliography

[1] Bernard D. Adelstein, Eric R. Johnston, and Stephen R. Ellis. A testbed for characterizing dynamic response of virtual environment spatial sensors. In *Proceedings UIST'92: The Fifth Annual ACM Symposium on User Interface Software and Technology*, November 1992.

[2] Aromat Corporation. *MQ Laser Analog Sensor Technical Manual*, 1991.

[3] Ascension Technology Corporation. *Questions and Answers About the Bird Family of 6D Input Devices.*

[4] Julius S. Bendat and Allan G. Piersol. *Random Data: Analysis and Measurement Procedures.* John Wiley & Sons, Inc., New York, second edition, 1986.

[5] Kenneth R. Boff and Janet E. Lincoln. *Engineering Data Compendium: Human Perception and Performance.* Harry G. Armstrong Aerospace Medical Research Laboratory, Wright-Patterson Air Force Base, Ohio, 1988.

[6] Kenneth R. Britting. *Inertial Navigation Systems Analysis.* Wiley-Interscience, New York, 1971.

[7] Robert Grover Brown and Patrick Y.C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering.* John Wiley & Sons, Inc., New York, second edition, 1992.

[8] Charles Broxmeyer. *Inertial Navigation Systems.* McGraw-Hill, New York, 1964.

[9] Mark Clymer. Personal Communication., 1993.

[10] Joseph M. Cooke, Michael J. Zyda, David R. Pratt, and Robert B. McGhee. Npsnet: Flight simulation dynamic modeling using quaternions. *Presence*, 4, 1993. In press.

[11] The Virtual Environment and Teleoperator Research Consortium affiliated with MIT. Virtual environment technology for training. Report 7661, BBN, 1992. Prepared for Naval Training Systems Center, Orlando, FL.

[12] H.J. Freund, H. Hefter, V. Homberg, and K. Reiners. Determinants of tremor rate. In L.J. Findley and R. Capildeo, editors, *Movement Disorders: Tremor*, pages 195–204. Oxford University Press, New York, 1984.

[13] F. M. Garcia. Rate output integration test report of the quartz rate sensor. Report DO 92-23, Systron-Donner Engineering Department, 1992.

[14] Herbert Goldstein. *Classical Mechanics.* Addison-Wesley Publishing Company, second edition, 1980.

[15] Richard Held and Nathaniel Durlach. Telepresence, time delay, and adaptation. In *NASA Conference Publication 10023*, 1987.

[16] Darren R. Laughlin, Andrew A. Ardaman, and Henry R. Sebesta. Inertial angular rate sensors: Theory and applications. *Sensors*, 9(10):20–24, October 1992.

[17] H. Leblond, F. Jupinet, J. P. Colinot, P. Jarri, L. Guillot, and P. St. Martin. New gyrometer for automotive applications. In *Proceedings — Society of Automotive Engineers*, pages 619–625, May 1990.

[18] Jiandong Liang, Chris Shaw, and Mark Green. On temporal-spatial realism in the virtual reality environment. In *Fourth Annual Symposium on User Interface Software and Technology*, pages 19–25, November 1991.

[19] John Little and Loren Shure. *Signal Processing Toolbox for Use With MATLAB.* The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, Mass. 01760, 1993.

[20] Jack M. Loomis, Chick Hebert, and Joseph G. Cicinelli. Active localization of virtual sounds. *Journal of the Acoustical Society of America*, 88(4):1757–1764, October 1990.

[21] Kenneth Meyer, Hugh L. Applewhite, and Frank A. Biocca. A survey of position trackers. *Presence*, 1(2):173–200, Spring 1992.

[22] C. M. Oman. Sensory conflict in motion sickness: an observer theory approach. In S. R. Ellis, M. K. Kaiser, and A. J. Grunwald, editors, *Pictorial Communication in Virtual and Real Environments*, pages 362–376. Taylor and Francis, London, 1991.

[23] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, Inc., third edition, 1991.

[24] Richard H. Parvin. *Inertial Navigation.* Van Nostrand, Princeton, New Jersey, 1962.

[25] George Pratt. Range resolution using interleaved pulse trains. Final Report to Draper Lab, 1993.

[26] Shooting Star Technology, 1921 Holdom Ave., Burnaby, B.C., Canada V5B 3W4, 604-298-8574. *ADL-1 Six Degree of Freedom Tracking System.*

[27] Mark Ward, Ronald Azuma, Robert Bennet, Stefan Gottschalk, and Henry Fuchs. A demonstrated optical tracker with scalable work area for head-mounted display systems. In *Proceedings 1992 Symposium on Interactive 3D Graphics*, 1992.