Hardware Design Optimization
for
Human Motion Tracking Systems

by
Bonnie Danette Allen

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2007

Approved by

Advisor: Gregory F. Welch

Reader: Gary Bishop

Reader: Henry Fuchs

Reader: Russell M. Taylor II

Reader: Leandra Vicci

Reader: Mary C. Whitton

ABSTRACT

BONNIE DANETTE ALLEN: Hardware Design Optimization
for
Human Motion Tracking Systems

(Under the direction of Gregory F. Welch)

A key component of any interactive computer graphics application is the system for tracking user or input device motion. An accurate estimate of the position and/or orientation of the virtual world tracking targets is critical to effectively creating a convincing virtual experience. Tracking is one of the pillars upon which a virtual reality environment is built and it imposes a fundamental limit on how real the “reality” of Virtual Reality can be.

Whether working on a new or modified tracking system, designers typically begin the design process with requirements for the working volume, the expected user motion, and the infrastructure. Considering these requirements they develop a candidate design that includes one or more tracking mediums (optical, acoustic, etc.), associated source/sensor devices (hardware), and an algorithm (software) for combining the information from the devices. They then simulate the candidate system to estimate the performance for some specific motion paths. Thus the predictions of such traditional simulations typically include the combined effect of hardware and algorithm choices, but only for the chosen motion paths. Before tracker algorithm selection, and irrespective of the motion paths, it is the choice and configuration of the source/sensor devices that are critical to performance. The global limitations imposed by these hardware design choices set a limit on the quantity and quality of the available information (signal) for a given system configuration, and they do so in complex and sometimes unexpected ways. This complexity often makes it difficult for designers to predict or develop intuition about the expected performance impact of adding, removing, or moving source/sensor devices, changing the device parameters, etc.

This research introduces a stochastic framework for evaluating and comparing the expected performance of sensing systems for interactive computer graphics. Incorporating models of the sensor devices and expected user motion dynamics, this framework enables complimentary system- and measurement-level hardware information optimization, independent of algorithm and motion paths. The approach for system-level optimization is to estimate the asymptotic position and/or orientation uncertainty at many points throughout a desired working volume or surface, and to visualize the results graphically. This global performance estimation can provide both a quantitative assessment of the expected performance and intuition about how to improve the type and arrangement of sources and sensors, in the context of the desired working volume and expected scene dynamics. Using the same model components required for these system-level optimization, the optimal sensor sampling time can be determined with respect to the expected scene dynamics for measurement-level optimization.

Also presented is an experimental evaluation to support the verification of asymptotic analysis of tracking system hardware design along with theoretical analysis aimed at supporting the validity of both the system- and measurement-level optimization methods. In addition, a case study in which both the system- and measurement-level optimization methods to a working tracking system is presented.

Finally, Artemis, a software tool for amplifying human intuition and experience in tracking hardware design is introduced. Artemis implements the system-level optimization framework with a visualization component for insight into hardware design choices. Like fluid flow dynamics, Artemis examines and visualizes the “information flow” of the source and sensor devices in a tracking system, affording interaction with the modeled devices and the resulting performance uncertainty estimate.

ACKNOWLEDGEMENTS

As a civil servant at the National Aeronautics and Space Administration (NASA) Langley Research Center (LaRC), I had the opportunity to pursue a Ph.D. as part of LaRC's Graduate School Program. In the Fall of 1999, I handed off my project design work and began the first of four semesters on campus at UNC Chapel Hill. During that time, I joined the tracker research group, found a mentor and advisor, formed a committee and proposed. This tremendous progress was made possible through the efforts of many people, some of whom I thank specifically below but there are many more who contributed in more ways than I can enumerate. In 2001, I moved back to NASA LaRC where I went back to project work fulltime and focused on my research between projects and as my personal schedule permitted. This meant that spans of time would pass when I was unable to make research progress and I am grateful to everyone at UNC who helped me to stay focused and advanced the research in my absence.

Research rarely occurs in a vacuum and this dissertation is no exception. As acknowledged above, many people contributed to the body of work presented here. The idea for using steady-state analysis for tracking in virtual environments is due to to Dr. Greg Welch who also contributed exploratory analysis prior to my arrival at UNC-CH. The value of our close collaboration from my arrival through to the finish cannot be overstated. I have also been the fortunate recipient of contributions from faculty, staff and student colleagues from the several research groups at UNC-CH. Some of the prose included in this dissertation is taken directly from papers (both published and unpublished) written from 1999 to 2007 and is the product of all contributors. When I refer to “we” or “our” and am not referring to the reader, I am referring to all of the contributors to this research but especially to collaboration with Dr. Greg Welch.

There are so many people who helped me along the way that I know I run the risk of forgetting someone in any attempt to acknowledge specific individuals. With apologies in advance to anyone I have overlooked, I want to express my gratitude to the following people:

- my advisor, Greg Welch, for his unwavering enthusiasm and many ideas, especially the idea of steady-state covariance as a performance metric.
- my Committee as a whole for their continued support of my stop-and-go research while working full time.
- Gary Bishop for sharing his uncanny ability for making the most complicated concepts seem intuitive.
- Henry Fuchs for always taking the time to stop by when he saw me in Sitterson to offer his support and encouragement.
- Russell Taylor for his feedback on my defense practice talk and his “expert” discussion.
- Leandra Vicci for her insights into the murky world of analog noise and her always interesting and educational “tangents”.
- Mary Whitton for her heroic efforts leading up to my defense and invaluable application focus.
- Marcel Prastawa and Erica Stanley for advancing the research with their work on the Artemis prototypes.
- Adrian Ilie for his assistance with the 3DMC setup and DragonFly camera data acquisition.
- John Thomas and David Harrison for their advice and assistance in setting up experiments.

- Tim Quigg, Janet Jones and Karen Thigpin for always finding a desk for me, helping with registration and making sure I had the forms I needed while studying remotely.
- Nick Englund and Ben Elgin of 3rdTech, Inc. for access to the inner workings of the HiBall tracking system and unending patience for my barrage of questions.
- Duane Armstrong, NASA LaRC's analog guru, for continuing to take my calls to answer my many questions about noise.
- George Allison, the head of LaRC's Graduate School Program, for seeing me (and others) through the program in the face of funding cuts.

And on a more personal note:

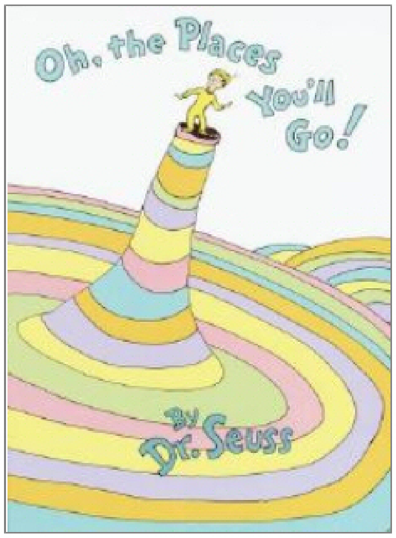
- Michael Nelson for his love and support. It's been a long road and you can finally rest easy. I'm finished.
- My parents, Tom and Bonnie Allen, for allowing their grown daughter (and her dogs!) to intermittently move back in with them from 1999 to 2007.
- My sister and best friend, Julianne Allen, for always being there.

Financial support for this work came from:

- NASA Langley Research Center (LaRC) Graduate School program
- National Science Foundation Information Technology Research grant IIS0121657, "Electronic Books for the Tele-Immersion Age: A Paradigm for Teaching Surgical Procedures"
- National Library of Medicine (NIH) contract N01LM33514, "Telepresence for Medical Consultation: Extending Medical Expertise Throughout, Between and Beyond Hospitals"
- DARPA DSO contract for "Wide Area Visuals for a Simulator in a Box", Award No. N000014-03-1-0589, with Henry Fuchs and Herman Towles. Ralph Chatham, DARWARS Program Manager.

- National Science Foundation grant "High-Fidelity Tele-Immersion for Advanced Surgical Training", UNC/UPenn/Brown, with Henry Fuchs, Kostas Danillidis, and Andy van Dam.
- Naval Research Lab contract "Technology for Full-Body Tracking", with Gary Bishop. Larry Rosenblum, NRL program manager.

The motion capture data used in this thesis was obtained from mocap.cs.cmu.edu [**22**]. The CMU database was created with funding from NSF EIA-0196217.



You have brains in your head.
You have feet in your shoes.
You can steer yourself
any direction you choose.

Oh, the Places You'll Go!
Dr. Seuss (Theodor S. Geisel)

The charm and challenge of life and human tracking.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

Virtual Reality (VR) is a human-computer interface (HCI) modality that involves real-time simulation and interactions through multiple sensory channels [**19**], most commonly visual. VR applications immerse the user in a computer-generated virtual environment (VE) that simulates reality through the use of interactive devices, which send and receive information to detect a user's input and modify the rendered virtual world accordingly. Thus, a VR system or environment can be separated into four components [**45**]:

- Input
- Simulation
- Rendering
- World Database

If any one of the components does not perform as required, the result is a less than satisfactory virtual experience that can cause a wide variety of negative effects ranging from misinformation (object location for example) to simulation sickness [**53**].

A key *input process* into any interactive virtual reality application is the system for *tracking* user or input device motion. An accurate estimate of the position and/or orientation of both people and objects of the virtual world is critical to effectively creating a convincing virtual experience.

Fidelity in VR is limited by the selected tracking system's precision and accuracy [**8**]. While the estimation circumstances and performance characteristics vary with the selected tracking system, the fundamental source of information is the same: pose estimates are derived from electrical measurements of mechanical, inertial, optical, acoustic, and magnetic source/sensors [**104**]. Each type of hardware device has inherent limitations related to the physical medium and practical limitations imposed by the measurement systems. The limitations imposed by hardware design affect the rate and quality of the information throughout the working volume for a given system configuration in complex and often unintuitive ways. This complexity often makes it difficult for designers to predict or develop intuition about the expected performance impact of adding, removing, or moving source/sensor devices, changing device parameters, etc. For example, what would be the likely effect of removing or re-arranging a tracking system's optical or acoustic beacons? How and where will the addition of nearby light or sound-occluding objects affect performance? How will moving or redirecting one motion capture camera affect the precision? Will it help to add another camera? How many do you need? What happens if you change the camera lenses or field-of-view (FOV)?

1.1. System Design

Whether working on a new or modified tracking system, designers typically begin the design process with requirements for the size and shape of the working volume, the expected user motion, and the allowable infrastructure. Considering these requirements, they develop a candidate design that includes one or more tracking mediums (optical, acoustic, etc.), medium appropriate source/sensor devices (hardware), and some algorithm (software) for combining the information from the devices. They then simulate the candidate system in order to estimate the performance for some specific motion paths. This traditional design process is illustrated in Figure 1.1.

The results of such traditional simulations reflect the *combined* effect of hardware and algorithm choices, and only for select motion paths. While these simulations provide an accurate estimate of the system performance and are a valuable step in the tracking system

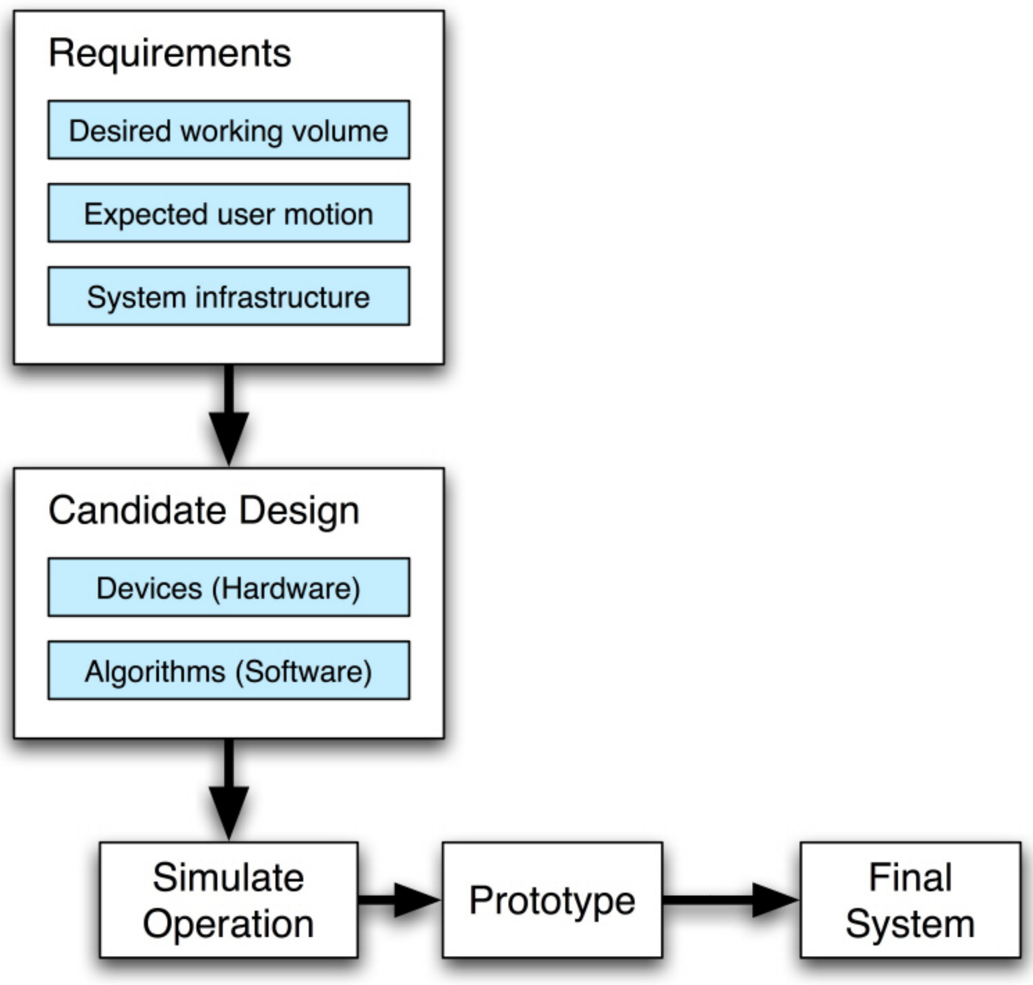


FIGURE 1.1. The traditional tracking system design process

design process, it is the choice and configuration of the source/sensor devices (i.e. the *hardware*) before tracker algorithm selection, and irrespective of the motion paths, that imposes an upper bound on performance. If the necessary information is not available at a sufficient rate and quality throughout the desired working volume, the performance is inherently limited in those areas. In effect the hardware device choices set an upper bound on how well the system as a whole will perform and no estimation algorithm can or should be expected to overcome suboptimal choices of devices, parameters, or geometric arrangement.

A major aspect of the work in this dissertation is a new hardware design optimization approach to be used prior to the traditional techniques of simulation and prototyping. This hardware design optimization serves to gain insight into the expected overall system performance as a function of the hardware choices alone. Ideally, one would specify desired performance goals throughout the working volume, and have a computer search the entire solution space and present the optimal (i.e. minimal) design – a tracking “oracle”. However for all but the most trivial systems the design space is so large as to render such a search intractable, thus making automatic optimization impractical, if not impossible, except in relatively restricted circumstances [**74**]. This oracle-like approach can be compared to a class of problems known as the Art Gallery problems.

1.2. Problem Complexity

Victor Klee originally posed the problem of determining the minimum number of security guards sufficient to cover the interior of an n-wall art gallery room [**75**]. Vasek Chvatal used triangulation to prove that $\lfloor n/3 \rfloor$ guards are sometimes necessary and always sufficient to cover a two dimensional polygon of n-vertices [**26**]. Ntafos extended the original Art Gallery problem when he considered the problem of finding the minimal cover for guards in a 2D grid [**71**]. He found this problem to be solvable in polynomial time. However, he also found that the *minimal* guard coverage of a *three-dimensional* grid is NP-complete. A solution for an NP-complete problem can be verified for correctness in polynomial time but cannot be solved in polynomial-time so that every possible solution in a solution set must be tested [**28**]. As posed by Ntafos, in both the 2D and 3D problems, each point in the plane or volume must be visible by at least one guard without the occlusion by walls as with the original art gallery problem.

If instead of guards, we consider cameras (Figure 1.2) or sensors in general, tracking is an extension of the Art Gallery problem with the added complexity that *multiple* sensors (guards) must have visibility to each point in the tracking volume in order to provide 6D

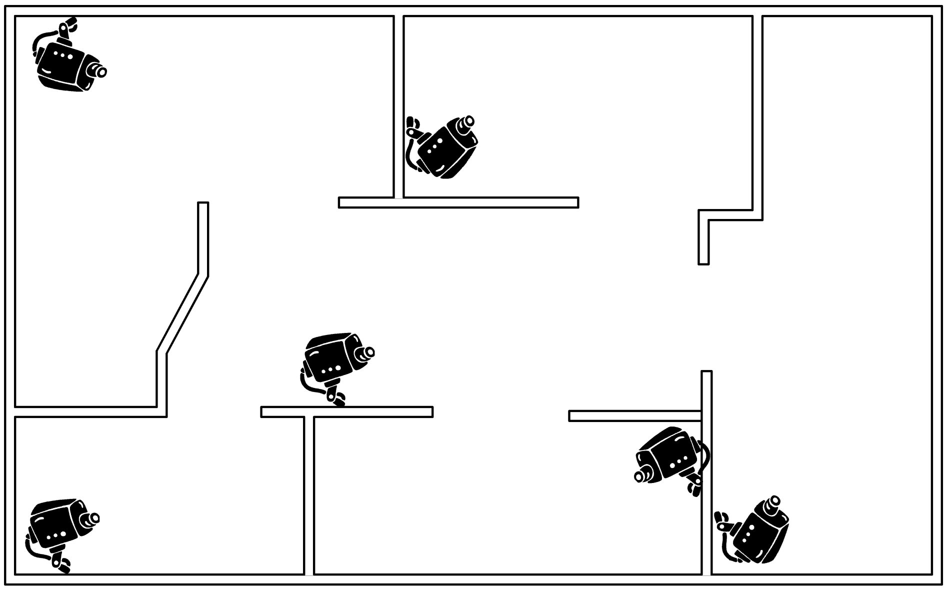


FIGURE 1.2. An Example Art Gallery

measurements. More generally, we need to "see" every point in a tracking volume without interference and we need to see each point more than once because it takes multiple measurements to solve for pose. This interference includes the aforementioned occlusion but also includes parameters such as drift in inertial sensors and the effect of ferrous materials on magnetic sensors. Interference can prevent signal reception between sources and sensors or degrade it so that the signal is unusable, blinding the system (at least partially) to the occluded points in the tracking volume.

1.3. Intelligence Amplification

Given the intractability of the "oracle" solution, we focus on amplifying human intuition and experience in tracking design with a method and tool to be used in conjunction with traditional design techniques and employing a human-in-the-loop approach to "augment the human intellect [beyond that of an] unaided human" [**33**]. This "intelligence amplification" [**6**] should help tracking system designers understand the consequences of every hardware design specification and decision, their interdependencies and how one may be balanced with another in order to achieve optimum performance [**53**].

An area where a similar type of intelligence amplification is already in use is computational simulation, specifically computational fluid dynamics (CFD) and finite element modeling (FEM). In CFD, fluid or air-flow visualizations make "invisible" information "visible" as shown in Figure 1.3. Example FEM applications include simulation of electromagnetism, heat transfer, climate, atoms, traffic, and network systems [**7**]. While the tracking domain discussed here is not CFD or FEM, it does fall into the broader category of *Computational Simulation* (see Section 1.6.3). Computational simulation provides qualitative and quantitative insights into natural phenomena (typically from an area of science or engineering) via mathematics and computer science to gain an improved understanding of a problem.

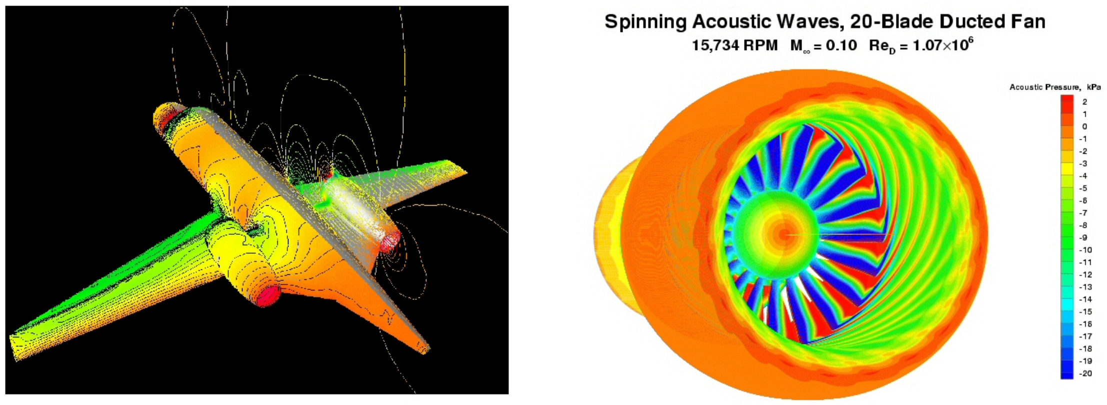

Spinning Acoustic Waves, 20-Blade Ducted Fan
15,734 RPM $M_\infty = 0.10$ $Re_D = 1.07\times10^6$
Acoustic Pressure, kPa
2
1
0
-1
-2
-3
-4
-5
-6
-7
-8
-9
-10
-11
-12
-13
-14
-15
-16
-17
-18
-19
-20

FIGURE 1.3. Computed pressure contours on the body surface, a wing cross-sectional plane and surface oil-flow patterns for a generic business jet configuration (left) and an Blade Acoustic Pressures (right). Both visualizations courtesy of NASA LaRC.

Similar to CFD and FEM, a hardware design optimization framework with meaningful visualizations that made invisible sensor information visible throughout a working volume (Figure 1.4) could help tracking system designers develop insights into the effects of their design choices. The articulation of such a framework and its exploration as an intelligence amplification tool for tracker design is the primary result of this dissertation. In the following sections I introduce the framework and its use for both system- and measurement-level design optimization.

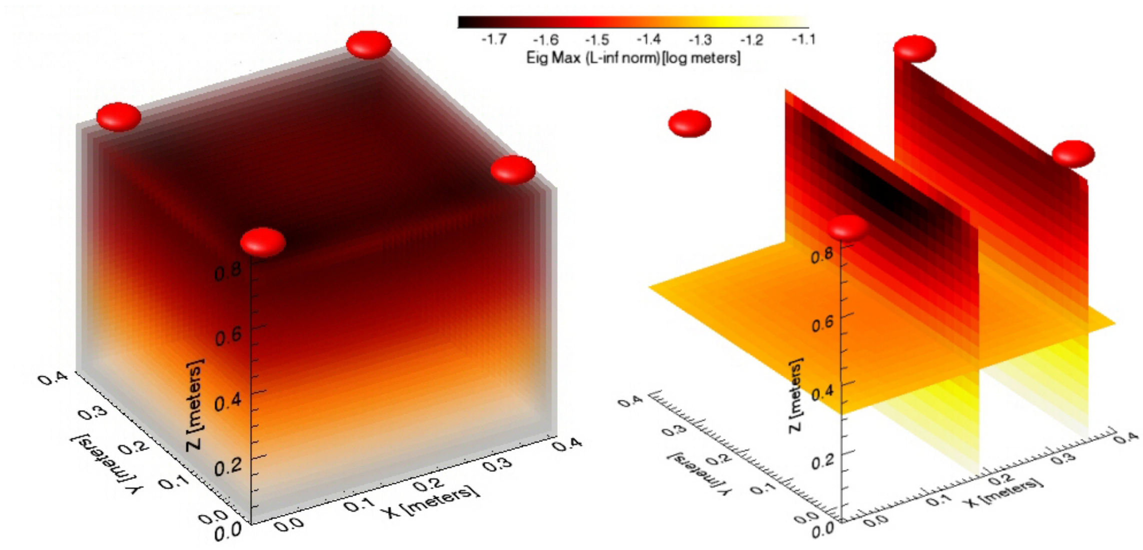


FIGURE 1.4. Two volume visualizations of the "flow" of acoustic sensor information in the tracking domain

1.4. A Stochastic Framework for Tracking System Design Optimization

Along with many dimensions to the design space, there are many possible criteria for performance of a tracking system. For example one might be concerned about resolution/precision, noise, static accuracy, or dynamic accuracy. In a 6D system one might be more concerned about orientation than position, or visa-versa. See [**53, 19, 4**] for more specific criteria and discussion of performance. Without presupposing a particular performance metric, the framework for an intelligence amplification tool in the tracking domain should meet several requirements:

(1) Accommodate currently-known device types and update rates;

(2) Accommodate different device arrangements (e.g., on user, in environment);

(3) Accommodate and account for occlusion and interference (i.e. observability);

(4) Account for growing uncertainty due to expected user motion;

(5) Avoid path and algorithm-dependent analysis;

(6) Support designer interaction with the proposed design; and

(7) Calculate and effectively display estimated performance.

A few of these considerations bear further explanation. The term observability in the third point refers to the amount of information provided by the devices. It is formally described in Section 3.7.4, but it suffices to say here that different devices provide different degrees and amounts of information depending on, for example, the distance and angle between a source and sensor. The fourth point recognizes that the performance depends on the rate of information from the devices, when compared to the expected user motion. The notion of path and algorithm-dependent analysis in point five refers to the common practice of simulating the results of a particular user motion "track" through space and time, using a particular system design and tracking algorithm (see [**101**]).

After investigating many possible performance metrics, we decided on a stochastic approach based on the asymptotic variance (uncertainty) throughout the working volume. This stochastic framework is attractive for many reasons, including meeting the above requirements. In particular, as described in Section 3.3, any tracking system can be considered and described using state-space models. In addition, as described in Section 2.2, there are relatively well-understood methods for estimating the asymptotic (a.k.a. steady-state) performance of systems described in this form.

1.5. Hardware Information Optimization

Hardware device choices set an upper bound on tracking system performance and both the device configuration and the devices themselves contribute to achieving the best possible performance. One of the primary benefits of a stochastic framework is that it enables a designer to perform optimization of the hardware design at the system level independent of software algorithms and motion paths as described in Section 1.5.1. In addition, the parameters required as input to compute the system-level analysis can be evaluated and compared for optimization of the hardware devices themselves as described in Section 1.5.2. This is another contribution of the work presented in this dissertation.

Figure 1.5 shows the human-in-the-loop hardware design optimization process inserted into the traditional design sequence. Notice that it occurs before algorithm selection (i.e. it is algorithm-independent) and does not replace simulation and prototyping. Given descriptions of hardware devices, the desired working volume, and a model for the expected user motion, a tracking system designer can iterate through various system configurations to produce a graphical depiction of the expected position and/or orientation uncertainty throughout the working volume of *any* tracking system. In this way, a designer can explore the variations between designs for a single tracking system. In addition, a designer can compare the performance of multiple systems that may use completely different physical mediums.

1.5.1. System-Level Optimization. The fundamental metric computed in the stochastic framework for hardware design optimization is steady-state covariance, P^{∞}. It can be calculated before either real or simulated system measurements are available. P^{∞} is defined as

$$P^{\infty} = \lim_{t \to \infty} \mathsf{E}\left\{(\bar{x}(t) - \hat{x}(t))(\bar{x}(t) - \hat{x}(t))^{\mathsf{T}}\right\} \tag{1}$$

where $\bar{x}$ and $\hat{x}$ represent the *true* and *estimated* states (position and/or rotation, for example) and E denotes statistical expectation.

An example of P^{∞} analysis for an acoustic tracking system is illustrated in Figure 1.4. The red spheres at the top of the visualization represent four speakers and performance is better (darker) near the transmitters (speakers) and towards the center of the volume where the four acoustic sensor ranges overlap and declines (lighter) in the downward direction.

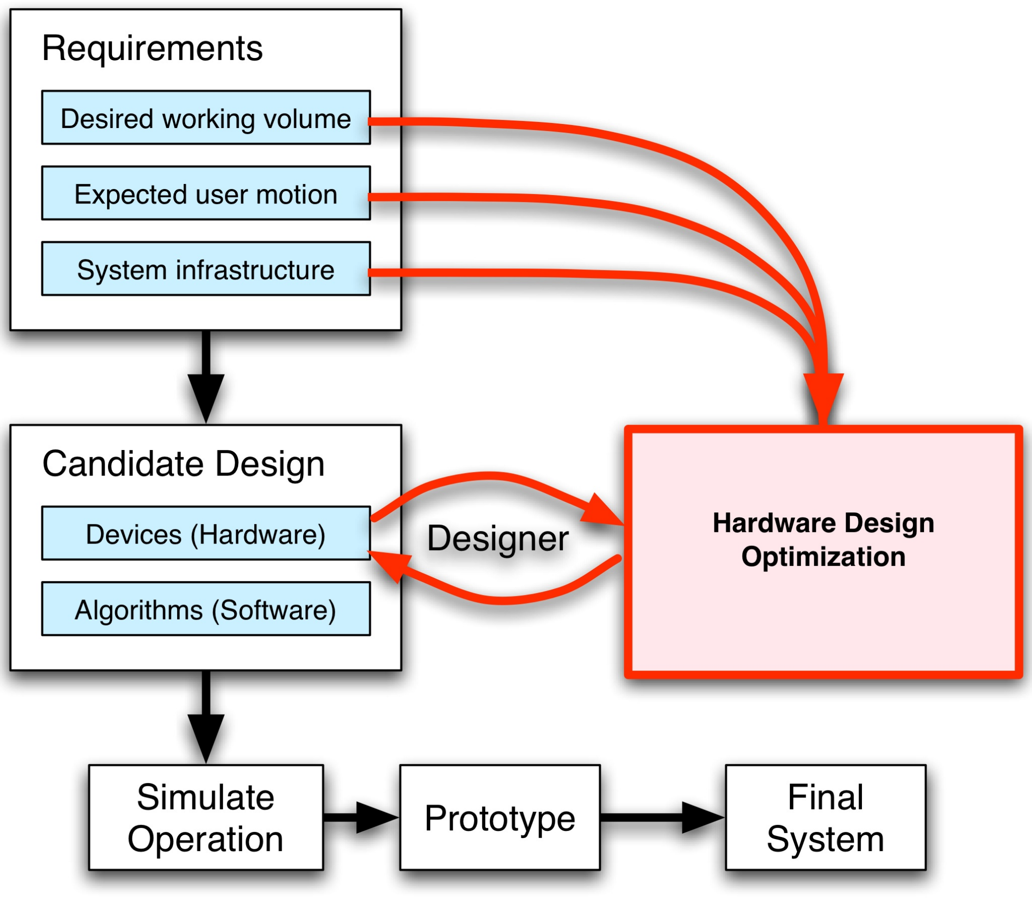


FIGURE 1.5. Hardware Optimization (in red) integrated into the traditional tracking system design process

Once a system has been defined and analyzed, one can alter or interact with a candidate hardware system, varying the types and configurations of devices and graphically visualizing the corresponding effects on the system performance. From this interaction, designers gain insight into *particular* design choices, as well as the relative effects of variations *between* candidate designs, independent of the tracking algorithm chosen for the real system and specific motion paths.

1.5.2. Measurement-Level Optimization. In addition to computing the fundamental metric (P^{∞}) for system-level performance evaluation, the elemental building blocks of this stochastic framework (descriptions of hardware devices and a model for the expected user motion) can be used for analysis of measurement devices. For some types of measurements, the quality may increase with sampling duration (i.e., averaging or correlation) but during this sampling time, a tracked target may be moving. So while the system is waiting to acquire improved measurements, it is unable to observe to the movement of the tracked target in discrete steps and may ironically see increased noise in the measurement due to the motion of the target. By examining the relationship between user motion and measurement behavior over time, the optimal balance between these opposing system elements can be determined. This measurement-level analysis is presented in Section 6.2 and is applied to a working tracking system in Section 7.3.

1.6. This Dissertation

1.6.1. Thesis Statement. *Characterization of a human motion tracking system in a state-space stochastic form offers new opportunities to optimize the hardware design, independent of the estimation algorithm used in the actual system. One can employ closed-form steady-state analysis to explore and compare the expected overall performance corresponding to different hardware configurations. Further, one can determine the optimal sensor sampling time to maximize the filtering of random measurement noise while minimizing the impact of intra-measurement target motion.*

1.6.2. Contributions.

- Establish that the stochastic framework (Chapter 3) enables complimentary system- and measurement-level hardware information optimization, independent of algorithm and motion paths;
    - The system-level analysis (Section 1.5.1) provides human-in-the-loop intelligence amplification for optimization of aggregate hardware information against expected user motion;

    - The measurement-level analysis (Section 1.5.2) provides a method for optimization of individual measurement information against expected user motion;

- Mapping of the stochastic framework to a software architecture and prototype (Chapter 4).
- Experimental evaluation (Chapter 5) in support of verification of system-level steady-state analysis of tracking system hardware design.
- Theoretical analysis (Chapter 6) aimed at supporting the validity of both the system- and measurement-level optimization methods;
- Application of both the system- and measurement-level optimization methods to a working system (Chapter 7);
    - The conclusions reached in the case study analysis could improve the already high performance of the HiBall tracking system;
    - The conclusions reached in the case study analysis enable reduction of the infrastructure of the HiBall tracking system;

1.6.3. Computational Simulation. The American Institute of Aeronautics and Astronautics (AIAA) Guide for the Verification and Validation (V&V) of Computational Fluid Dynamics Simulations [**2**] provides a means for assessing the credibility of modeling and simulation in computational fluid dynamics. This research will be presented within this context and will borrow from its widely-accepted vernacular. The main principles necessary for assessing the credibility of modeling and simulation are *qualification*, *verification* and *validation*. As defined by the AIAA, qualification is the determination of adequacy of the conceptual model to provide an acceptable level of agreement for the intended domain application. Verification is the process of determining that a model implementation accurately represents the conceptual description and specification. Validation is the process of determining the degree to which a model is an accurate representation of the real world. In short, verification deals with mathematics and validation deals with physics (i.e., the physical world). Both qualification and validation attempt to establish the representational fidelity of the conceptual (qualification) and computation (validation) models to the physical system and so qualification may be considered another form of validation [**67**]. Therefore, instead of QV&V, most of the literature speaks to V&V where validation may mean either conceptual or computation validation. Once a system has attained a level of validation, it can be used for *prediction*, the use of a computational model to foretell the state of a system under conditions for which it has not been validated [**2**]. Prediction is an inference based on validation evidence. Examples of the predictive capability of the framework will also be presented.

1.6.4. Nomenclature. Throughout this thesis I use lower-case variables with over-bars, hats or tildes ($\bar{x}$, $\hat{x}$, $\tilde{x}$ for example) to denote a vector, and upper-case variables to denote matrices. I use the term *designer* or *user* to refer to the engineer or researcher evaluating the system, and the term *target* to refer to the object being tracked. Example targets include a sensor on a person's head, a retroreflective sphere on a joint or limb, and potential 3D surface points that one wants to reconstruct using cameras and image/vision-based techniques.

I use a simple acoustic 3D position-tracking system to provide a concrete basis for discussion. This system is depicted in Figure 1.6, and a corresponding visualization of the estimated performance from one possible set of descriptive system parameters is shown in Figure 1.4. There are four speakers permanently mounted in the upper corners, and a microphone mounted on the moving tracking target (or user). The curve in the middle represents an example target motion path through the 3D space over time, and the point $\bar{x}(t) \in \Re^3$ represents the 3D position or *state* of the target at time t.

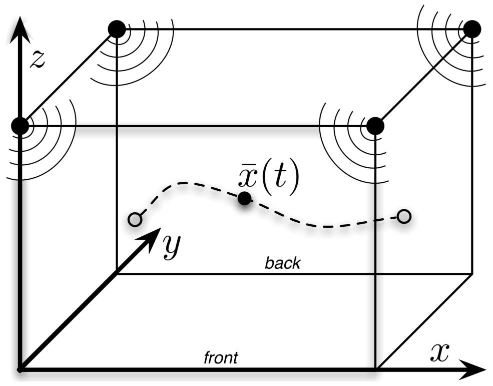


FIGURE 1.6. A simple acoustic 3D position tracker example.

1.6.5. Structure of this Dissertation. In Figure 1.7, the AIAA QV&V model is presented on the left and a mapping to the structure of this thesis is shown on the right.

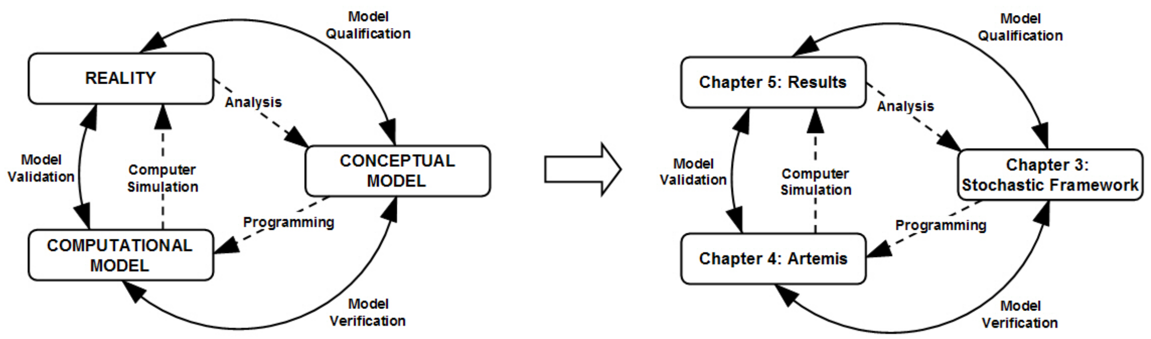


FIGURE 1.7. Phases of Modeling and Simulation: Qualification, Verification and Validation (QV&V) [**2**]

I begin with related research in Chapter 2 highlighting how tracking system performance has been analyzed and reported in the past. Chapters 3 through 7 map to the AIAA V&V process. In Chapter 3, I describe the conceptual model in terms of the specific mathematical framework used to quantify the uncertainty corresponding to a candidate design. The use of asymptotic analysis for performance estimation is a well-studied and accepted method for characterizing the performance of a tracking sensor system [**42**] [**18**] and so I will speak specifically to domain qualification (human motion) for the presented framework. In Chapter 4, I describe Artemis, the software tool developed to implement this framework and visualize the resulting performance estimate. Typically, scientists and engineers speak to verification before validation. However, I will present validation before verification in order to better provide an understanding of what steady-state analysis does and a context for verification. In Chapter 5, I present experiments aimed at validating the use of asymptotic estimates and visualizations, and concrete examples of the approach used to evaluate systems. After presenting the validation results, I address the importance of accurate modeling parameters in Chapter 6 and apply the techniques from this chapter in a case study (Chapter 7) and predict the behavior of modified tracking systems. In Chapter 8, I discuss future plans for this research.

CHAPTER 2

Related Work

2.1. Tracking System Performance Analysis

Tracking and motion capture for interactive computer graphics have been explored for over 35 years [**94, 70, 9, 66, 104**] by both commercial and academic researchers who have utilized mechanical, magnetic, acoustic, inertial, and optical technologies. For example, commercial magnetic tracking systems from Ascension and Polhemus have enjoyed popularity as a result of a small user-worn component, relative ease of use, and robustness for many applications. Optical systems include the HiBall-3000^{TM} system by 3rdTech, the FlashPoint^{TM} and Pixsys^{TM} systems by Image Guided Technologies, and the laserBIRD^{TM} system by Ascension Technology. Foxlin et al. at Intersense in particular have had success developing hybrid systems that combine inertia measurements with acoustic signals [**37, 38, 40**], and with passive optical signals [**37**]. Similarly, optical systems for 3D motion capture have a long history, having been explored for over 30 years [**107**]. Today, companies like Vicon, Motion Analysis, and Ascension make turn-key optical systems that are used in human and animal motion analysis and industrial applications. Much work has also been done on vision-based approaches to motion capture [**68**].

Despite this long history of research and the availability of commercial systems, the tracking community faces two challenges with respect to performance analysis. The first is performance prediction and the second is performance assessment. While this research focuses on the former, the types of metrics used for performance assessment during the test phase of tracking system design are germane.

The U.S. Army Research Laboratorys Army Research Office (ARO) stated in a 2005 BAA [**95**] that “there are no effective methods for predicting the performance of an Image Analysis and Processing system, given the input signal or parameters of the scene such as time of day or nature of clutter”. This extends to broader category of tracking systems where, instead of or in addition to parameters such as time of day or nature of clutter, we speak of camera placement, field-of-view, occlusion, diffraction, multipath, etc. With or without these “effective methods”, tracking system researchers and commercial developers must analyze their systems and offer some measure of performance.

Since tracking systems often exhibit non-uniform performance over space, the common practice of using a single set of statistics to specify a system’s global performance is inadequate and potentially misleading [**39**]. The engineers at Northern Digital, Inc. (NDI) contend that typical assessments are not likely to accurately reflect performance in most users’ intended environments and that a tool is needed to enable users to sample performance throughout regions of the tracking volume. To this end, NDI provides an Accuracy Assessment Kit to test the accuracy of Polaris^{TM} positions sensors in the field. While such tool is useful to the end-user and is a step in the right direction for analysis and assessment of tracking system performance, this tool is specific to a single tracking technology, algorithm and medium.

Whether predicted or assessed, performance measures vary not only in magnitude (one system is twice as accurate as another, for example) but also in modality (accuracy vs. precision metrics, for example). The ultrasonic Lincoln Wand [**83**] (1966) claims a positional resolution of 0.02 inches and absolute accuracy of 0.2 inches. In 1974, Burton and Sutherland reported accuracy of 7.3 mm for their optical position tracker, Twinkle Box [**20**]. The laser-based Minnesota Scanner [**88**] reported accuracy of 0.04 inches RMS and 0.034 in tangential resolution in 1989. In 1999, Welch reported estimation errors of 0.2 mm at 1.9 meters and 0.5 mm at a height of approximately 1 meter for nearly all of the working volume of the HiBall optical tracking system [**103**]. Polhemus advertises accuracy and resolution of 0.03 inches RMS and 0.0002 inches per inch [**78**], respectively for its FASTRAK^{TM} electromagnetic tracking system. With the exception of the Constellation tracking system [**38**] in Figure 2.1, published literature about the performance of tracking systems from the first acoustic system [**83**] through today, typically reports performance metrics in terms of a handful of statistics that may or may not characterize the complete system performance nor permit direct comparison between tracking systems.

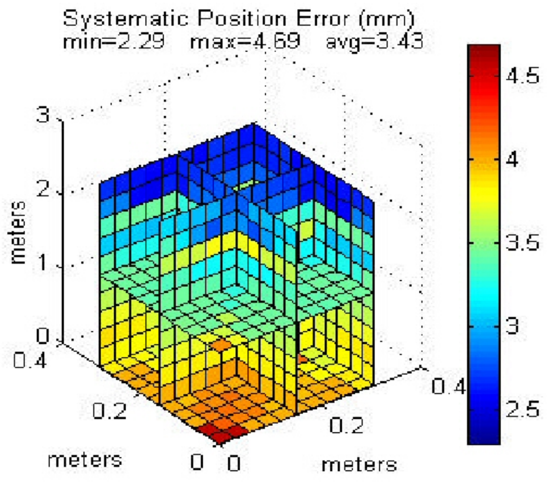


FIGURE 2.1. Example of visualized error in the Constellation tracker [**38**]

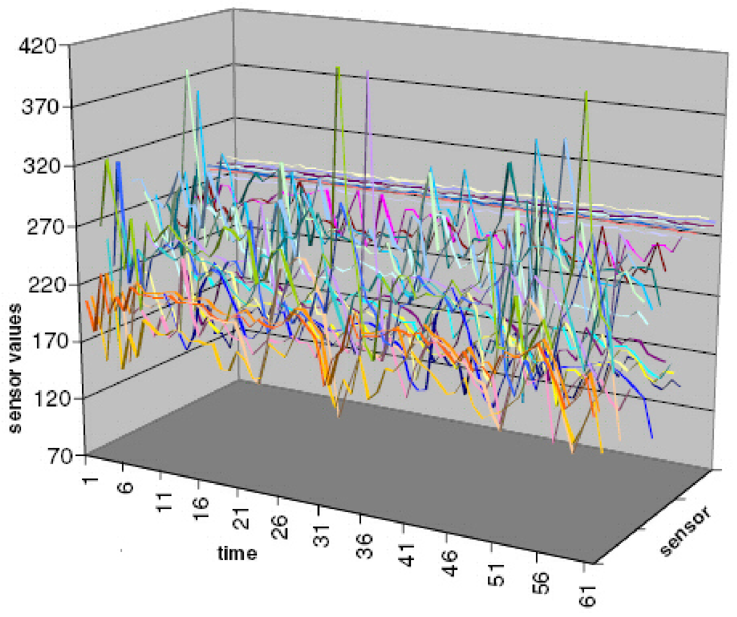
While there has been interest in the design, performance and visualization of sensor systems, historically this interest and subsequent research has focused on specific implementations and not with an eye towards the general. Further, these tools and approaches may provide analysis for a "point design" (i.e., a specific system with a specific target for a specific application) and not support interactive exploration of the design space. The following sections survey both predictive and assessment methods and metrics and categorizes them in terms of qualitative vs. quantitative and interactive vs. non-interactive techniques. Qualitative analysis has proven useful and there are many examples of successful applications (see Section 2.2 and Section 2.3). However, if a certain performance level is required for an error-intolerant application, a quantitative analysis is imperative (see Section 2.4 and

Section 2.5). Further, some systems also consider the type of motion to be tracked (see Section 2.6). Within each of these sections, *prediction* and *assessment* (or some variant) will be used to distinguish between analyses that provide predictive performance estimates and those that provide performance measurements of an existing tracking system, respectively.

2.2. Qualitative Performance Analysis

Laerhoven et al. [**57**] instrumented a wearable harness with thirty accelerometers to assess whether system performance improves with the addition of sensors. The authors defined performance as the system's context-awareness algorithm's ability to distinguish between twenty defined contexts such as walking, standing, running, in a meeting, etc. Figure 2.2 shows the output stream from the thirty accelerometer configuration while walking and the performance metric of context distinction (i.e., the ability to successfully discern the correct context) as the number of possible contexts are increased from 1 to 20. Note that the number of sensors increases as expected from right to left on the left axis (# sensors) but that the number of contexts decreases from left to right on the right axis (# contexts). The system performs better per this qualitative metric given the greatest number of available sensors and the fewest number of contexts from which to select (the back corner of the plot where the z-axis value is highest and largest).

Hollerer et al. [**43**] qualitatively assessed the accuracy of magnetic and inertial tracking systems in indoor environments by tracking users traveling along a known path (an outer hallway of their research building) and plotting the tracked path on the floor plan of the research building. The loop in the path on the left edge of the left image in Figure 2.3 shows the most dramatic effect of magnetic distortion from a nearby magnetic resonance imaging (MRI) device (two floors directly above) on the performance of the magnetic tracking system and the effect of "drift" on the performance of the inertial device is shown on the right. The arrows on both the left and right plots show the start and direction of the traveled track.



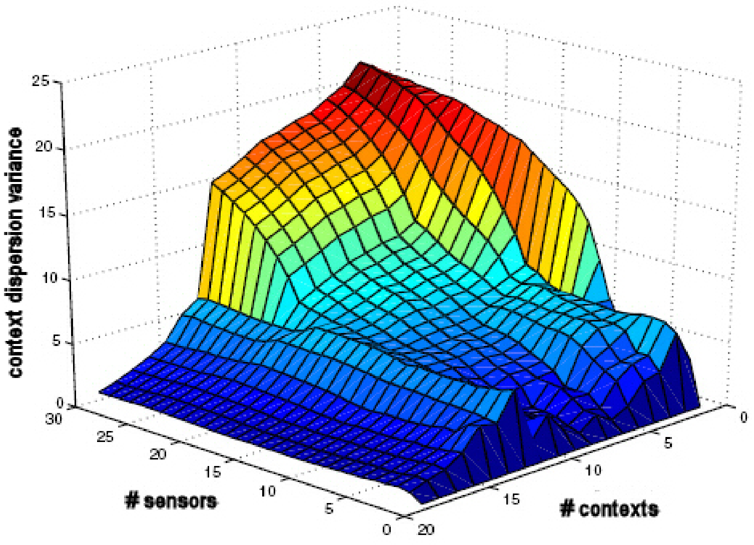


FIGURE 2.2. Accelerometer output while walking (top) and context detection performance (bottom) [**57**]. The vertical axis label on the bottom plot was added for clarification.

2.3. Qualitative Performance Analysis with Interaction

An example of an effective interactive, qualitative analysis tool is Pandora [**89**], a software simulator for multiple camera placement within a pre-modeled 3D environment. Pandora provides human-in-the-loop interaction enabling a user to move and point the available

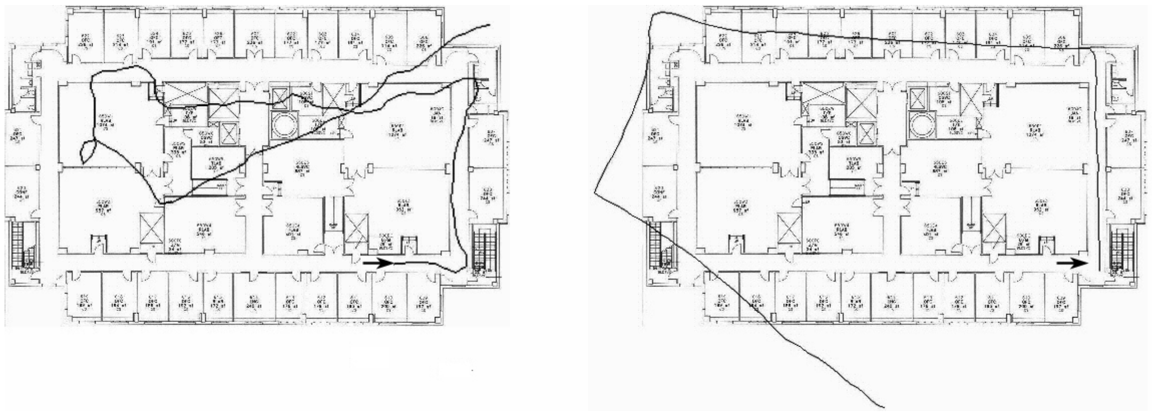


FIGURE 2.3. Tracking performance of magnetic (left) and inertial (right) systems [**43**]

cameras at will to predict system performance. It provides a realtime visualization of the mapping of camera pixels to a targeted surface for a qualitative assessment of performance as shown in Figure 2.4. With this tool, a user can locate occluded areas or, for example, determine whether the areas of interest are covered by two cameras for stereo. Further, in terms of available information, four cameras is qualitatively better than three is better than two is better than one or none. A particularly useful output of Pandora are the calibration matrices for the cameras in the scene which can be used as input for further analysis.

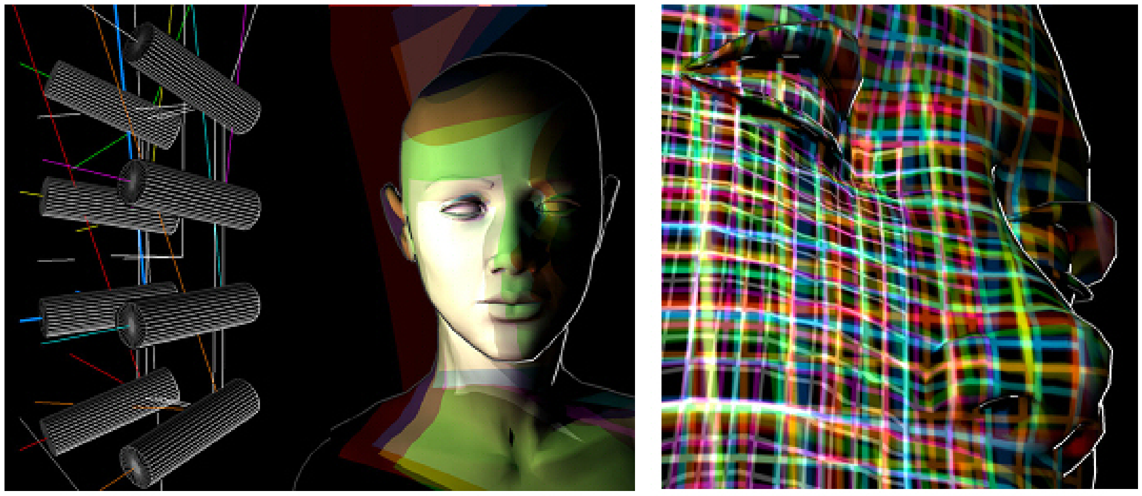


FIGURE 2.4. The *Pandora* Interactive Camera Placement and Visibility Simulator [**89**]

2.4. Quantitative Performance Analysis

In an environment like telesurgery, it may not be sufficient to know that the tracking system configuration in use is the best possible one. Instead, numerical performance metrics are mandatory to show that the tracking system performance requirements are being met. Due in part to applications of this kind, methods and tools for camera placement for vision-tracking are a popular topic for both performance prediction and assessment.

Davis et al. proposed a method for designing marker-based tracking probes [**30**] and for predicting accuracy in pose estimation of these same type systems [**29**]. Using their "Viewpoints Algorithm" and a first-order error propagation to apply the errors from individual markers to overall pose estimation, an optical system using fiducials can be designed and analyzed but this approach does not extend beyond this context.

In 2000, Fleishman et al. [**36**] presented a predictive, automatic camera placement for image-based modeling from scenes with known geometry. Beginning with a large database of potential camera positions, they present a visibility algorithm to produce a minimal subset of camera positions that covers every visible polygon in a scene. This work draws heavily from the work presented by McMillan and Bishop [**65**], specifically the plenoptic function which describes all of the image information visitable from a particular viewing position.

Okansen [**73**] and Ailisto [**3**] both presented methods for vision-systems using Computer-Aided Design (CAD) models. Ailisto developed a Measurement Model Design (MMD) Tool for CAD-based graphic measurement predictive planning for range-finder-based 3D coordinate measurements. It builds a "measurement model" from the available sensors and points or features to be measured for video digitization. Okansen extended this research into photogrammetry (as shown in Figure 2.5) and developed a simulator using least-squares to build a variance-covariance matrix to visualize error ellipses and ellipsoids of error.

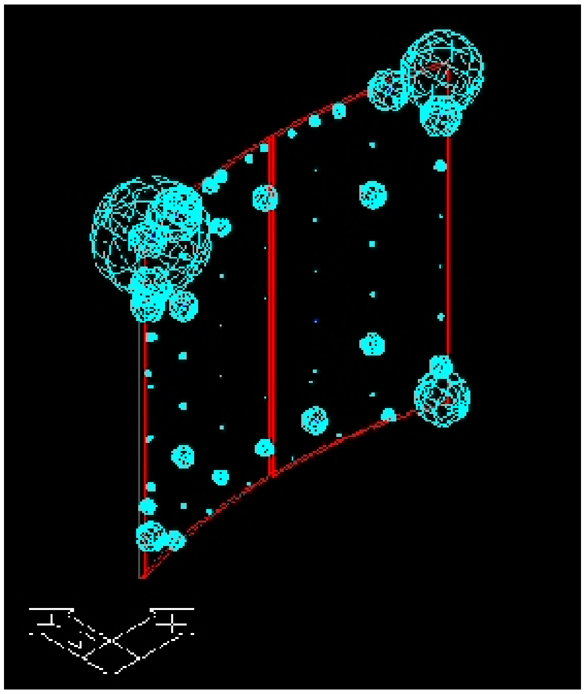


FIGURE 2.5. Example of visualized mean radial spherical errors [73]

Olague [74] addresses the problem of minimizing error in 3D measurements for predicting optimal camera placement for accurate reconstruction. A criterion using the maximum eigenvalue of a computed covariance matrix is established and a global optimization process employing genetic algorithms is applied to minimize this criterion. While this approach is similar to the one presented here, it is specific to camera network design.

Livingston and State [62] created a noise metric for use in magnetic tracker calibration assessment for augmented reality applications within a defined working volume. After collecting sample data from both Ascension Technology's Flock of Birds magnetic tracking system and Faro Metrecom's IND-1 mechanical Faro arm, they defined error metrics in terms of the local coordinate systems of the two tracking systems. For both position and orientation, the difference in either distance or quaternion angles (respectively) was calculated for multiple readings at each sample point. In both cases, the length of the diagonal of a bounding box around the plotted errors is the noise metric. In a coincident publication, Livingston [61] presented four methods for the visualization of rotation fields resulting from the calibration of the Flock of Birds magnetic tracking system as described above. Animated axis stream surfaces and axis streamlines were shown to be the best for highlighting heterogeneity in the rotation field and areas of large tracker error, respectively. Figure 2.6 shows position error (left) and rotation error using the axis stream surface visualization technique (right).

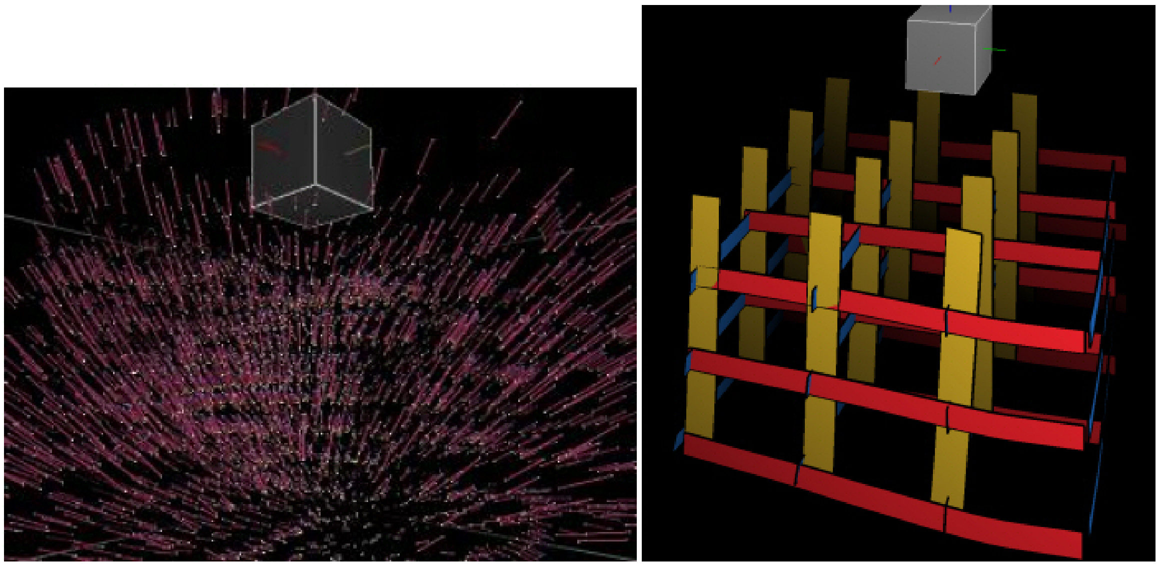


FIGURE 2.6. Visualizations of position [**62**] and rotation [**61**] error from calibration of Ascension's Flock of Birds for use in Augmented Reality environments

2.5. Quantitative Performance Analysis with Interaction

A tool developed for real-time prediction and visualization of acoustic sound fields [**55**] was applied to the design of the Center for New Music and Audio Technologies (CNMAT). Real-time interaction with proposed design elements such as shape of the room, position and orientation of the sound sources, microphones and audience seating aided sound engineers in tradeoff studies for the varied uses of the theater. Example visualizations of acoustic field behavior are shown in Figure 2.7.

Using an accepted measure of acoustic clarity (C_{80}) as a metric, Stettner and Greenberg [**91**] proposed a method to predict the spatial distortion of sound using specular and

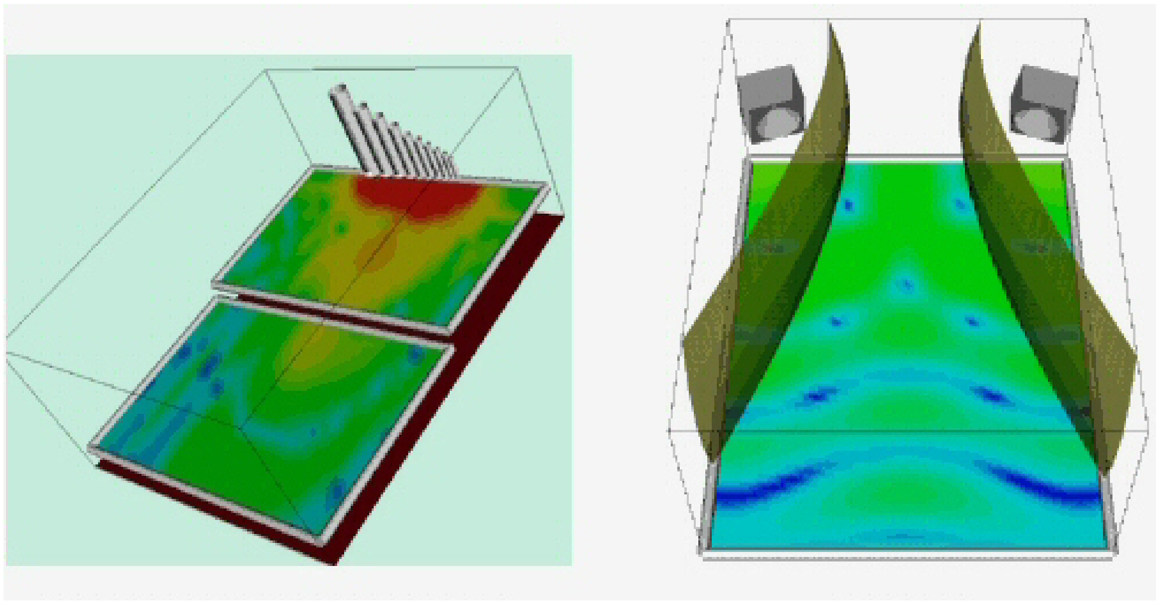


FIGURE 2.7. Organ pipe sources with performer and audience cut planes (left) and time delay isosurface (right) [**55**]

diffuse ray–tracing along with Monte Carlo techniques to generate visualizations of acoustic simulation. C_{80} (measured in decibels) is defined as the logarithmic ratio of the early arriving sound energy from 0 ms to 80 ms divided by the late sound energy arriving after 80 ms. Figure 2.8 shows a visualization of simulated sound clarity in (from left to right) fan-shaped, box-shaped and reverse-fan-shaped concert halls. The reverse fan-shape (right) has the best C_{80} values and the least variation in those values of the three possible concert hall shapes.

More recently, Pulkki [**80**] introduced Vector Base Amplitude Planning (VBAP) to create two- and three-dimensional sounds fields into which any number of virtual sound sources can be inserted interactively for predictive planning. A digital simulator that implements this method for up to eight loudspeakers was also developed.

2.6. Performance Analysis with User Motion

Incorporation of user motion into the analysis of tracking system performance could be the difference between an accurate simulation of performance and not. Systems that

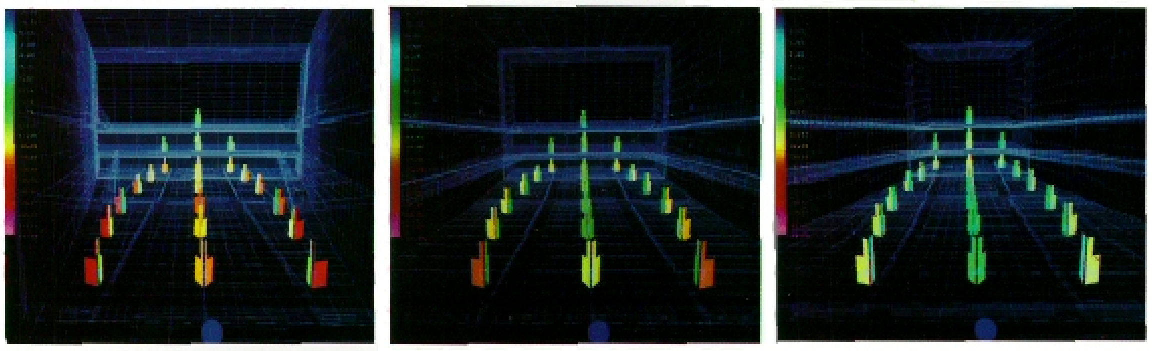


FIGURE 2.8. Sound Clarity in (from left to right) fan-shaped, box-shaped and reverse-fan-shaped concert halls [**91**]

work well in dynamic environments may not perform as effectively in static environments and vice versa. Research in the area of acoustic system performance is typically specific to quantitative acoustic behavior (as opposed to a measure of system performance quality) but must take into account the inherently dynamic environment of a concert hall for example. Godot [**97**] is a predictive software system for room acoustics design. Sound beams are traced as they traverse a polygonal model of a room and linear least-squares prediction is used to determine the coefficients of a digital filter that matches the frequency response of each acoustic path. From this an audible simulation of the proposed acoustic design can be generated. Unlike many point-design analysis tools, Godot II [**98**] accommodates moving objects (i.e. people) in a room.

Chen [**24**] describes the design and application of *M-Track*, a scalable resource allocation tool for the design of tracking systems using tens-of-cameras and considers user motion. Using an extended Kalman Filter, asynchronous information from camera-processor pairs is integrated and coupled with a quantitative uncertainty metric for evaluating the predicted tracking performance of a multi-camera setup. Of particular interest is the fact that this metric (see Figure 2.9) considers both the camera parameters and the likelihood of target occlusion due to user motion. For dynamic occlusion (blocked visibility from a camera due to occlusion by the moving target in the scene), occluder positions and orientations are drawn from a probability distribution function. While M-Track does accommodate heterogeneous cameras (different frame rates, focal lengths, etc.), it is limited to the analysis of camera-based systems in determining optimal configurations.

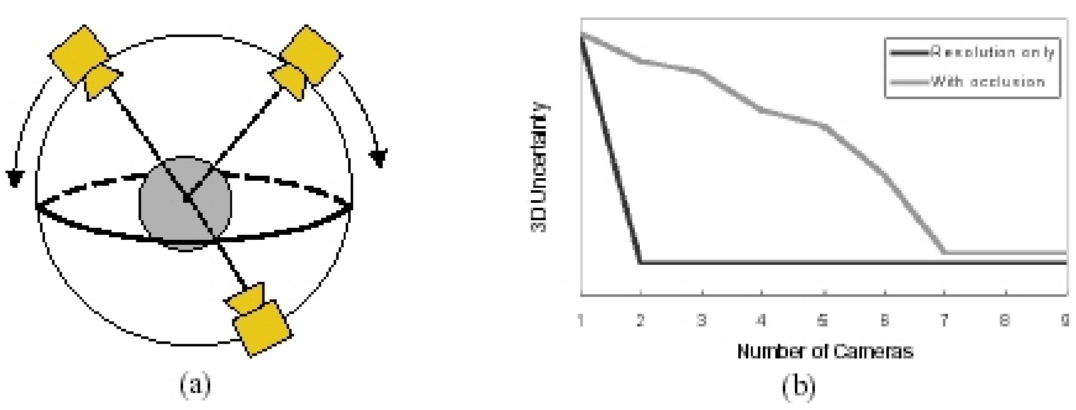


FIGURE 2.9. Example of error (3D uncertainty) vs. number of cameras surrounding a sphere with and without occlusion [**24**]

2.7. Intelligence Amplification and Augmentation

Whether an analysis is qualitative, quantitative, interactive or sensitive to user motion, the goal is a common one. All calculate and communicate information hidden from the unaided human. In his classic 1956 paper "Design for an Intelligence-Amplifier" [**6**], Ashby presented a possibility proof of a machine that would solve problems its creators were incapable of solving. In his 1962 report [**33**], Engelbart presented a conceptual framework for "augmenting human intellect". This framework for increased intellectual capacity aimed to gain comprehension in previously overly-complex problems in order to find better solutions to seemingly insoluble problems faster. In both cases, the objective is not to increase native intelligence, but to augment the human being with means for organizing experience and solving problems so that an intelligent system results in which the human being is the central component [**87**]. The fundamental difference between Ashby's intelligence *amplification* and Engelhart's intelligence *augmentation* is that Ashby's intelligence-amplifier was conceived of as a stand-alone system like the steam engine while Engelhart's intelligence augmentation concept presents a system made up of a human *and* the means for augmenting human intellect such as Vannevar Bush's *memex* [**21**], a hypertext workstation that analyzes, categorizes and presents information in a comprehensive way.

Ouh-Young et al. [**76**] present an "intelligence augmentation man-machine system" for force display in molecular docking to assist scientists already skilled at identifying the patterns, shapes, and packing of molecules. Their manipulator allows for six-DOF manipulation and generates force output to aid biochemists in molecular docking problems. Biocca [**11**] asserts that avatar representation of body image in virtual environments is a form of intelligence augmentation and that this embodiment is a critical component in the advancement of virtual reality systems [**10**]. Intelligence augmentation is a popular goal in the research area of wearable computing where the sensors for a real-time clock and calendar, the exact room temperature, or CO-level might be fused into a multi-sensor network [**57**]. Given the intractability of the tracking "oracle" (see Section 1.2), intelligence augmentation is also a goal of the approach presented here.

CHAPTER 3

Conceptual Model: Stochastic Framework

The highlighted portion of Figure 3.1 (outlined and in red) indicates where this chapter falls in in the QV&V process. Here we map the problem domain (evaluation of human tracking systems) to the proposed stochastic representation.

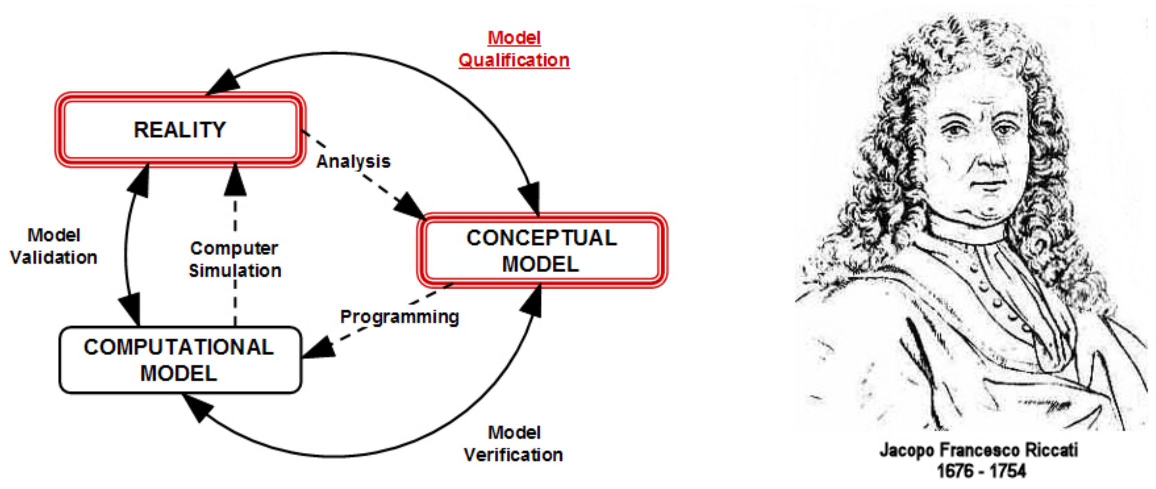


FIGURE 3.1. The Qualification Process

3.1. Statistical Uncertainty

There are many possible quantitative metrics for tracking performance estimation. For example one might be concerned about resolution or precision, noise, static accuracy, dynamic accuracy, latency, or some combination. See [**4, 19, 53**] for more examples and general discussion of performance. The fundamental metric computed in the optimization framework developed in this thesis is statistical uncertainty. More specifically, this is a stochastic estimate of the asymptotic or *steady-state* error covariance (P^∞) throughout the working volume.

3.2. Steady-state covariance

Consider the example acoustic 3D position tracking system. At a representative set of 3D points $\{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_p\}$ throughout the working volume we can estimate and graphically depict the steady-state error covariance (P^∞)

$$P_i^\infty = \lim_{t\to\infty} \mathsf{E}\left\{(\bar{x}_i(t) - \hat{x}_i(t))(\bar{x}_i(t) - \hat{x}_i(t))^\mathsf{T}\right\}, 1 \le i \le p \tag{2}$$

where $\bar{x}_i$ and $\hat{x}_i$ represent the *true* and *estimated* states (respectively) at point i, and E denotes statistical expectation. Note that the method does not attempt to estimate $\bar{x}_i$, $\hat{x}_i$ or the residual $\tilde{x}_i$ where $\tilde{x}_i = \bar{x}_i - \hat{x}_i$ which would require measurements. Instead, P^∞ is estimated directly from state-space model of the system and stochastic estimates of the various noise sources.

3.3. State-Space Models

To estimate the statistical uncertainty (i.e. steady-state error covariance) of the state, we begin by mathematically describing the expected target motion and the measurements using state-space models. State-space models are essentially a notational convenience for estimation and control problems, developed to make what would otherwise be a notationally-intractable analysis tractable [**52, 4**] by representing the variables (inputs, outputs and states) as vectors and the differential and algebraic equations (dynamics) as matrices.

3.3.1. The State Variables. The internal state variables (aka the state) are the smallest possible subset of system variables that can represent the entire state of the system at any given time. In the case of tracking, the state typically represents the user pose and (if needed) pose derivatives. For example, a 3D-pose ($\bar{x}$) as described above that includes position without derivatives is shown in Equation (3).

$$\bar{x} = [x \quad y \quad z]^T \tag{3}$$

A 6D-pose that includes both position and orientation without derivatives is

$$\bar{x} = [x \quad y \quad z \quad \phi \quad \theta \quad \psi]^T \tag{4}$$

where ϕ, θ and ψ are roll, pitch and yaw euler angles (rotation around the x-, y- and z-axis) respectively. Other rotation angle representations (such as quaternions) can also be used.

An important factor in determining the appropriate state variables is the type of motion expected in the tracking environment. These *motion models* (Section 3.3.2), along with the *measurement systems* (Section 3.3.3) themselves, influence whether the state should include derivative measurements and, if so, what derivative order.

3.3.2. Motion Models. Traditionally, motion model types are described in terms of the physical parameter that is constant or uniform over time. The types of constant or uniform motions of interest [**23**] are *constant position* (CP), *constant velocity* (CV) and *constant acceleration* (CA) which display *zero* velocity, acceleration and jerk, respectively, as illustrated in Figure 3.2.

In stochastic estimation, noise is injected into the process in place of the "zero" term so that, in the case of constant velocity motion, for example, acceleration is not zero but is instead normally-distributed random noise. Thus, position would instead be modeled as $x = x_0 + vt$, where $v = \int a$, and $a \sim N(0,q)$. This is a Position-Velocity Motion Model. The state variables for a position only tracker with a PV motion-model (i.e. includes first derivatives) is shown in Equation (5).

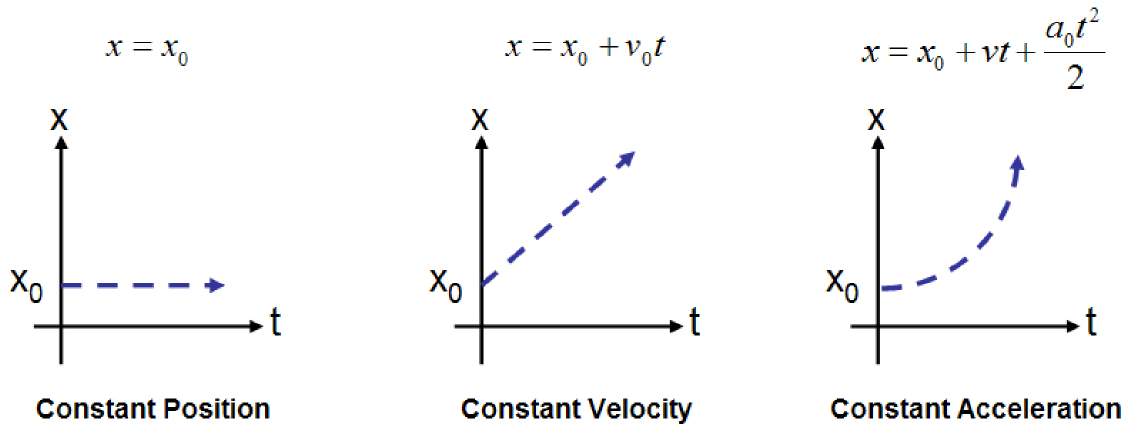


FIGURE 3.2. Traditional Motion Models

$$\bar{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T \tag{5}$$

Position (P), Position-Velocity (PV) and Position-Velocity-Acceleration (PVA) motion models map to the traditional constant position, velocity and acceleration models, respectively. Table 3.1 shows the process of noise integration through to 1D position for each of the three stochastic motion models.

TABLE 3.1. Stochastic Motion Models

$N(0,q) \rightarrow \int \rightarrow x(t)$	$N(0,q) \rightarrow \int \rightarrow \int \rightarrow x(t)$	$N(0,q) \rightarrow \int \rightarrow \int \rightarrow \int \rightarrow x(t)$
Position (P)	**Position-Velocity (PV)**	**Position-Velocity-Acceleration (PVA)**

These integration processes apply to 3D position, 3D orientation plus 6D position and orientation state spaces. In all cases, integration is over some time interval, δt, as described in Section 3.3.

Each stochastic motion model corresponds to some actual target motion. For example, P-motion might be an appropriate choice for surgical environments where the surgeon moves slowly and methodically and perhaps not at all over some δt. In this case, the pose estimate uncertainty over δt is small as we expect little change between observations and position is modeled as a random walk. PV motion is typical of most larger-scale environments in which a human target is walking or moving at a normal pace (i.e. not running or quickly changing direction) and velocity is modeled as a random walk. PVA motion is rapid and dynamic with sudden changes in speed and direction as one might observe in dancing, jumping or high-energy sports such as baseball (swinging a bat) or boxing (throwing punches). In this case, acceleration is modeled as a random walk and the uncertainty in pose between observations (over δt) is high. It is possible (even probable) that a single motion type will not completely describe any single given activity over a non-trivial time duration. For example, a person might switch from PV-motion to P-motion if s/he stops to observe an object or activity in a virtual environment after crossing a room for a closer look. PVA-motion in humans is not maintainable over any significant length of time and can be thought of as short, even instantaneous, bursts of energy. P-motion is perhaps the easiest of the three motion model types to imagine while PV- and PVA-motion are more difficult. While not intended to serve as a rigorous proof, the following discussion should provide intuition about the different types of motion models, specifically PV- and PVA-motion models.

Carnegie Mellon University's (CMU) Graphics Lab Motion Capture Database offers over 2600 motion trials in many categories from common behaviors such as typing on a laptop (an example of P-motion) to more dynamic activities, including sports. Trial 13 for Subject 15 [**22**] contains approximately 78 seconds of shadow boxing motion capture data at 120 frames per second (fps) using only the right-hand to punch initially and then switching to the left-hand alone. For approximately the first 50 seconds, right-hand punches are thrown and for the remaining 38 seconds left-hand punches are thrown. A video of the motion capture trial (30 fps) can be found at the CMU site [**22**] and still frames for a portion of the video are captured in Appendix B for reference. The video frame numbers (1273 through 1632) approximately correspond to seconds 42 through 54.

Figure 3.3 shows the y-axis position as captured along with the calculated velocity, acceleration and jerk data for the boxer's right hand during the entire trial. The apices of the final four right-hand punches can be observed in frames 1290, 1340, 1430 and 1480. All four graphs in Figure 3.3 are repeated in Appendix A on a larger scale and with annotation. For readability and proximity, the annotations are not included in Figure 3.3. The boxer throws 28 punches as marked in Appendix A and these punches are dynamic, sometimes covering as much as 50 inches from start to finish as the boxer moves forward and backward with his punches. To the right of the dashed line marked "Transition to left-hand punches" the position data becomes much less dynamic as the right hand is being held up in a defensive stance as the boxer bobs and weaves, punching with his left hand. By taking derivatives of this data, we can observe PVA-motion in the boxer's right hand when the boxer is throwing punches with his right hand and and PV-motion when he punches with his left hand.

The first derivative (velocity) reveals observable signal content in both the right-hand and left-hand portions of the data. If we were observing P-motion, we would expect to see random noise as the velocity input into the noise model. The second derivative (acceleration) is beginning to look more like random noise but we observe signal content in the right-hand section (left side) of the data. The data to the right of the right-to-left-hand transition, however, contains little to no observable content suggesting that right hand motion while punching with the left hand could successfully be modeled as PV-motion. It is also possible (even likely) that the motion of the right hand between punches could be successfully modeled as PV-motion. However, we still observe signal content during the boxer's punches. The third and final derivative (jerk) reveals a signal that appears to be indistinguishable from a noise prior to the transition mark, suggesting that the punches in right-punching phase of the shadow boxing trial could be modeled successfully as PVA-motion.

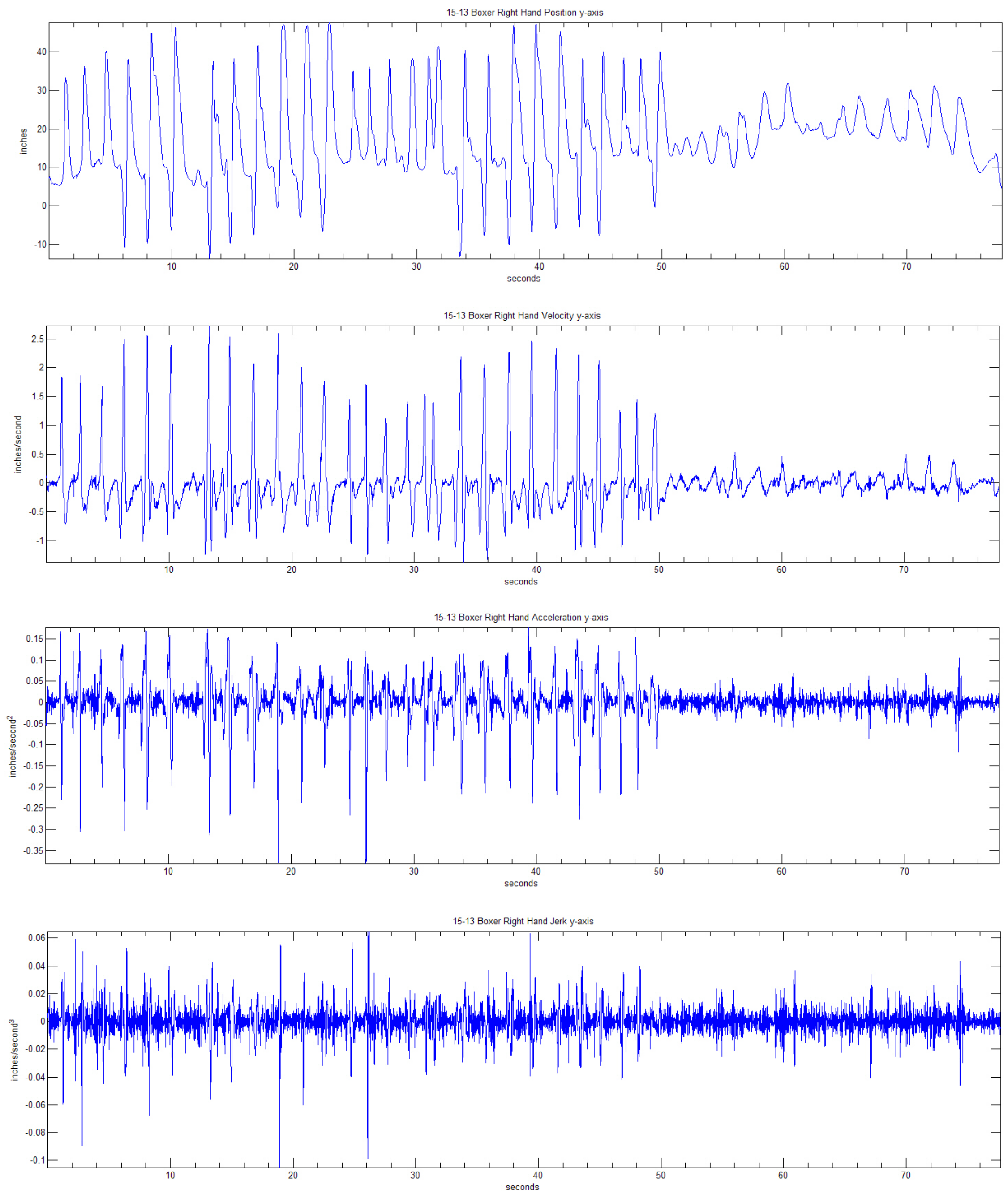


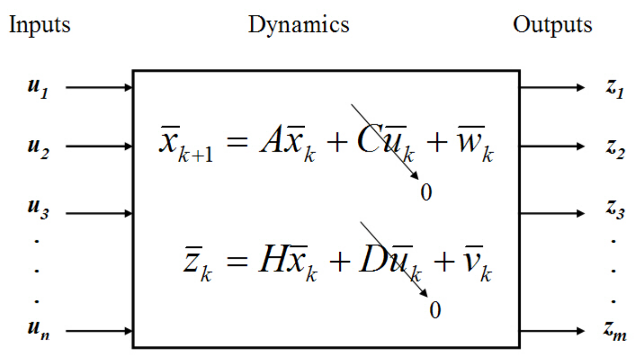
FIGURE 3.3. Boxing Motion Capture right hand y-axis data (from top to bottom) position, velocity, acceleration and jerk

3.3.3. Measurement Systems. The measurement system is the collection of sources and/or sensors used to observe the human in motion. Examples of commercial and research systems that utilize these mediums are listed in Section 2.1 and can be classified in terms of employed medium(s), geometry, etc. The five physical mediums employed in tracking measurement systems are mechanical, magnetic, acoustic, inertial and optical. Beyond the physical medium, measurement systems can be described as inside-looking-out or outside-looking-in (sensors on the user versus sensors in the environment). System measurements can be absolute or relative (at a specific time or a change from the previous estimate), derivative measurements, instantaneous or averaged over time. The system can be active or passive; linear or non-linear. The geometric arrangement of the source/sensor devices is another facet of the measurement system. Each measurement system will have a unique set of parameters that calibrate and map its measurements to the mathematical framework.

3.3.4. System Dynamics. The state variables and measurements are related by a pair of differential or difference equations. One equation moves the state over time and the other maps the measurements to the state space. These two first-order stochastic equations describe the *system dynamics* as shown in Figure 3.4 where, for tracking systems, the input vectors ($C\bar{u}_k$ and $D\bar{u}_k$) are zero (i.e. input elements $\bar{u_1} = \bar{u_2} = ... = \bar{u_n} = 0$ so there is no controllable input), $\bar{z}$ is an m-length vector of available measurements from devices such as accelerometers, gyroscopes, cameras, etc. and $\bar{w}$ and $\bar{v}$ are white, normally-distributed, random noise with zero-mean.

The top equation in Figure 3.4 is the *process model* and the bottom equation is the *measurement model.* These two equations serve in some form as the basis for most stochastic estimation methods. Both models are composed of a deterministic component (matrices A and H) and a random component ($\bar{w}$ and $\bar{v}$).

From Figure 3.4 , the vector state-space notation that describes the change in a state $\bar{x}$ over time is shown in Equation (6).



Inputs
Dynamics
Outputs
u_1
u_2
u_3
u_n
$\bar{x}_{k+1} = A\bar{x}_k + \cancel{C\bar{u}_k} + \bar{w}_k$
0
$\bar{z}_k = H\bar{x}_k + \cancel{D\bar{u}_k} + \bar{v}_k$
0
z_1
z_2
z_3
z_m

FIGURE 3.4. Block diagram of a linear stochastic dynamic system in discrete time (based on a Figure in [**42**]). Both $\bar{u}_k$ vectors are zeroed because there is no controllable input.

$$\hat{x}_{k+1} = A\hat{x}_k + \bar{w}_k \tag{6}$$

where $\bar{w} \sim N(0,Q)$. The corresponding (continuous-time) differential equation can be modeled as shown in Equation (7).

$$\dot{x}(t) = A_c x(t) + q_c(t) \tag{7}$$

where A and A_c are $n \times n$ discrete- and continuous-time state transition matrices, respectively.

The measurement system is described by an equation that maps measurement-space to state-space. It is common to model the m-dimensional device measurements $\bar{z}$ at discrete time k as

$$\bar{z}_k = H\hat{x}_k + \bar{v}_k \tag{8}$$

where H is an $m \times n$ matrix relating the n-dimensional state to the m-dimensional measurements, and $\bar{v}$ (like $\bar{w}$) represents zero-mean, white measurement noise, presumed to be uncorrelated with $\bar{w}$.

In practice the actual noise signals $\bar{w}$ and $\bar{v}$ are not known or estimated as part of a stochastic estimator. Instead, designers typically compute the process and measurements as

$$\hat{x}_k = A\hat{x}_{k-1}, \tag{9}$$

$$\hat{z}_k = H_k\hat{x}_k, \tag{10}$$

then estimate the process and measurement noise covariances Q and R of the presumed normal distributions $\bar{w} \sim \mathsf{N}(0,Q)$ and $\bar{v} \sim \mathsf{N}(0,R)$, and use those covariances to weight the measurements and to estimate the state uncertainty. It is the deterministic parameters, A and H, and the random parameters, Q and R, that the designer must specify to perform a steady-state analysis.

3.4. Non-Linear State-Space Models

In cases where the process and/or measurement models are non-linear, equations (9) and (10) would be written as shown in Equation (11) and Equation (12).

$$\hat{x}(t) = f\left(\hat{x}(t-\delta t)\right) \tag{11}$$

$$\hat{z}(t) = h\left(\hat{x}(t)\right) \tag{12}$$

These non-linear functions can be linearized about the point of interest $\bar{x}$ in the state space. To do so one would compute the Jacobians of the respective functions,

$$A = \left. \frac{\partial}{\partial \bar{x}} f(\hat{x}) \right|_{\bar{x}} \tag{13}$$

$$H = \left. \frac{\partial}{\partial \hat{x}} h(\hat{x}) \right|_{\hat{x}} \tag{14}$$

and use them in place of their corresponding matrices in equations (9) and (10). While such linearizations can lead to sub-optimal results (Section 6.1.1), they provide a computationally efficient means for estimation, and in most cases should offer a reasonable basis for comparison of steady-state results. For linear models, the designer would write functions that implement A and H (linear functions in matrix form) from equations (9) and (10). For non-linear models, the designer would instead write functions that implement the respective Jacobians from equations (13) and (14). Note that the Jacobians resulting from equations (13) and (14) will also be correct for linear models as well, resulting in equations 9) and (10. An alternative to the non-linear (aka Extended) Jacobian equations is the Unscented filter approach [**48**].

3.4.1. Process Model. In the discrete-time process model described by Equation (6), the state transition matrix A moves the target's state forward over some interval of time, δt. The term $A\bar{x}$ models the deterministic portion of the process, while the term $\bar{w}$ and corresponding covariance Q model the random portion of the target's motion.

To start, let's examine a simple 1D position tracking system with derivatives such that the state is defined as $\bar{x} = [x \quad \dot{x}]^T$. To move the estimated state vector $\hat{x}_k$ forward in time δt, the new position $\hat{x}_{k+1}$ is a function of the previous value $\hat{x}_k$, the corresponding velocity element $\hat{\dot{x}}_k$, and the time δt since the last update as shown in Equation (15). The velocity element $(\hat{\dot{x}})$ does not change as shown in Equation (16).

$$\hat{x}_{k+1} = \hat{x}_k + \hat{\dot{x}}_k \delta t \tag{15}$$

$$\hat{\dot{x}}_{k+1} = \hat{\dot{x}} \tag{16}$$

Putting these equations into matrix form,

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

where A is the deterministic portion of the discrete-time process model given in Equation (9).

It is in determining the random component (Q) of the process model that Equation (7) becomes important. The discrete-time (sampled) Q is a function of the continuous-time process components, A_c and Q_c, and δt such that

$$Q = \int_0^{\delta t} e^{A_c t} Q_c e^{A_c^T t} dt \tag{17}$$

as described in [**42**]. The continuous-time process noise is an $n \times 1$ vector such that $q_c = [0, ..., 0, N(0, q)]^T$ with corresponding $n \times n$ noise covariance matrix $Q_c = E\{q_c, q_c^T\}$. Putting these equations into matrix form, we have A_c and Q_c as shown below.

$$A_c = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad Q_c = \begin{bmatrix} 0 & 0 \\ 0 & q \end{bmatrix}$$

For completeness, Table 3.2 shows the continuous-time process parameters corresponding to P, PV and PVA motion models [**99**].

Using Equation (17), the discrete time Q matrices for a 1D position tracker for the motion-models in Table 3.1 are shown in Equations (18), (19) and (20).

TABLE 3.2. Parameters for P, PV, and PVA Motion Models

Model	$\bar{x}$	A_c	Q_c
P	[x]	[0]	[q]
PV	$\begin{bmatrix} \dot{x} \\ \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 \\ 0 & q \end{bmatrix}$
PVA	$\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & q \end{bmatrix}$

$$Q_P = [q(\delta t)], \tag{18}$$

$$Q_{PV} = \begin{bmatrix} q\frac{\delta t^3}{3} & q\frac{\delta t^2}{2} \\ q\frac{\delta t^2}{2} & q(\delta t) \end{bmatrix}, and \tag{19}$$

$$Q_{PVA} = \begin{bmatrix} q\frac{\delta t^5}{20} & q\frac{\delta t^4}{8} & q\frac{\delta t^3}{6} \\ q\frac{\delta t^4}{8} & q\frac{\delta t^3}{3} & q\frac{\delta t^2}{2} \\ q\frac{\delta t^3}{6} & q\frac{\delta t^2}{2} & q(\delta t) \end{bmatrix} \tag{20}$$

Here we use the acoustic tracker example to provide a more concrete notion of the process model parameters A and Q. A 6D state $\bar{x}$ includes the target position and derivatives (velocities) of the target. The six-element state is shown in Equation (21).

$$\bar{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^{\mathsf{T}} \tag{21}$$

To move the single element x of the state vector $\bar{x}$ forward over time δt one would compute the new position x as a function of the previous value $x(t-\delta t)$, the corresponding velocity element $\dot{x}$, and the time δt since the last update: $x(t) = x(t-\delta t) + \dot{x}(t-\delta t)\delta t$. The complete corresponding state transition matrix A, which is actually a function of δt, is shown in Equation (22).

$$A = \begin{bmatrix} 1 & 0 & 0 & \delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{22}$$

Now consider the random component of the process given by Equation (6). The process noise $\bar{w}$ is presumed to be a normally-distributed, zero-mean, spectrally white random variable with probability distribution $\bar{w} \sim \mathsf{N}(0, Q)$.

If we assume the process noise is shaped by the same system of integrators represented by Equation (22), then the covariance Q can be described as

$$Q[i,i] = q\frac{(\delta t)^3}{3} \tag{23}$$

$$Q[i,j] = Q(\delta t)[j,i] = q\frac{(\delta t)^2}{2} \tag{24}$$

$$Q[j,j] = q\delta t \tag{25}$$

for each pair $(i, j) \in \{(1,4),(2,5),(3,6)\}$ and some noise magnitude q. The above derivation of Q can be found in [**18**], and discussion about choosing q can be found in [**102**].

It is worth noting here that while one might imagine the need for many different process (target motion) models, experience indicates that the above position-velocity model is a reasonable match for the average human motion. If one expected the target to be primarily still, one might want to eliminate the velocity states in $\bar{x}$. Similarly if one expected the target to undergo coherent accelerations, one could add acceleration states. For more about motion types (P, PV, PVA), see Section 3.3.2.

3.4.2. Measurement Model. In the measurement model described by Equation (8) the *measurement matrix* H determines the relationship between the state and the measurements. Each type of device (combination sensor and/or source) will likely require a different measurement model of a common model with different parameters. In the acoustic example (Figure 1.6) there are four speakers fixed in the environment, and the target is a single moving microphone. In this example tracking system will continually measure the range from the microphone to each of the four speakers using a time-of-flight approach. In this case $m = 4$ and the measurement function would be

$$\bar{z}[i] = \bar{h}_i(\bar{x}) = \sqrt{(\bar{x}[x] - \bar{t}[i,x])^2 + (\bar{x}[y] - \bar{t}[i,y])^2 + (\bar{x}[z] - \bar{t}[i,z])^2}$$

for each transmitter $1 \leq i \leq 4$, where $\bar{t}[i,*]$ represents the position of transmitter i. Because the acoustic system uses four scalar range measurements and its state vector is six-dimensional, the measurement matrix H must be a 4×6 matrix. In fact because the measurement function is non-linear, we would have to use the linear approximation given by the measurement Jacobian as in Equation (12). For example, the Jacobian element corresponding to transmitter number one and the x element of the state would be

$$H_{1,x} = \frac{\bar{x}_x - \bar{t}_{1,x}}{\sqrt{(\bar{x}[x] - \bar{t}[i,x])^2 + (\bar{x}[y] - \bar{t}[i,y])^2 + (\bar{x}[z] - \bar{t}[i,z])^2}}.$$

Referring back to Equation (8), the measurement noise $\bar{v}$ is a normally-distributed, zero-mean, spectrally white, random variable with probability distribution $\bar{v} \sim \mathsf{N}(0,R)$. The magnitude of the covariance R represents the expected measurement noise for the given combination of sources and sensors. Unlike the process noise q in in (23)–(25), the measurement noise has concrete origins, and in practice R can be quantified. For example, one can arrange a real source/sensor pair in a lab, and gather statistics on the measurement variance under representative conditions (as in Section 5.1.2), then later fit a function to those gathered statistics and use this function in the asymptotic analysis. Or one can simply estimate the form and magnitude of the noise based on past experience or simulation as in [**25**]. For the acoustic tracker example, R is the expected variance in the range measurements, which is a function of the range itself.

3.5. Steady-State Solution

Once we know A, Q, H and R, there are multiple ways to arrive at a solution for P^{∞}, both iterative and closed-form. Because of the hybrid sensor make-up of some tracking systems each sensor type could have a different δt and a different rate of approach to steady-state. The decay time constant is the algebraic function [**42**]:

$$\tau(A,H,R,Q) = 2\sqrt{A^2 + \frac{H^2 Q}{R}} \tag{26}$$

where A and Q are both functions of δt. An iterative solution to P^{∞} would require (1) calculation of τ for all system sensor types and iteration of an estimator such as the Kalman Filter (see Section 3.7.1) for all sensors through to the longest τ and (2) actual measurements from an implemented system. This non-predictive, iterative approach is computationally and infrastructurally expensive and, therefore, we present the closed-form for P^{∞} used in this stochastic framework.

3.5.1. Closed Form Solution (DARE).

The Discrete Algebraic Riccati Equation (DARE) is a closed-form solution for the steady-state covariance P^{∞} [**42**]. Assuming the process and measurement noise elements are uncorrelated the DARE can be written as shown in Equation (27).

$$P^{\infty} = AP^{\infty}A^{\mathsf{T}} + Q - AP^{\infty}H^{\mathsf{T}}\left(R + HP^{\infty}H^{\mathsf{T}}\right)^{-1}HP^{\infty}A^{\mathsf{T}} \tag{27}$$

We use the MacFarlane–Potter–Fath "Eigenstructure Method" [**42, 64, 35, 79**] to calculate the DARE solution for P^{∞} as follows. Given the model parameters A, Q, H, and R from Section 3.3 we first construct the $2n \times 2n$ discrete-time Hamiltonian matrix Ψ as shown in Equation (28).

$$\Psi = \begin{bmatrix} A + QA^{-\mathsf{T}}H^{\mathsf{T}}R^{-1}H & QA^{-\mathsf{T}} \\ A^{-\mathsf{T}}H^{\mathsf{T}}R^{-1}H & A^{-\mathsf{T}} \end{bmatrix} \tag{28}$$

The Hamiltonian (sometimes called the Control Hamiltonian) is a matrix of state and costate variables (thus doubling the state size to $2n \times 2n$). The state equations represent constraints of the minimization problem, and the costate variables represent the marginal cost of violating those constraints. The minimal solution to the Hamiltonian is the best possible control for taking a dynamic system from one state to another. Once we have the Hamiltonian, we then form

$$\begin{bmatrix} B \\ C \end{bmatrix} = [\bar{e}_1, \bar{e}_2, \ldots, \bar{e}_n] \tag{29}$$

from the n characteristic eigenvectors $[\bar{e}_1, \bar{e}_2, \ldots, \bar{e}_n]$ of Ψ (only n eigenvalues are stable [**90**]), and finally using B and C we compute the steady-state covariance as

$$P^{\infty} = BC^{-1}. \tag{30}$$

As described in the next section we do this at a representative set of points $\{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_p\}$ throughout the working volume, computing H and R at each point, and using the same process model (A and Q) throughout.

3.6. Complete Steady-State Computation

Having determined to use a closed-form solution for the DARE to compute P^∞, one needs a process for computing P^∞ throughout a volume (i.e. a 3D grid of points) or for a list of points that define some geometric object. Further the P^∞ solution must reflect the input from all system measurement models each with their own update rate, δt.

To start, one has to define the process model. In particular one must decide on the form of $A(\delta t)$ and $Q(\delta t)$, for example as in equations (22)–(25). The δt parameters are included here to emphasize that A and Q are functions of δt.

Next one needs to define distinct measurement models and corresponding H and R matrices (functions) for each device type. For the example acoustic tracking system, one could think of four separate measurement models or a single parametric model. If using the latter approach, H and R are functions of $\bar{x}$ and any other parameters of interest (e.g., electrical biases, focal length, etc.). In a hybrid system one would typically have multiple parametric measurement models, e.g., one for acoustic devices and one for cameras as in the hybrid example presented in Section 5.2.3. In any case the function that implements each measurement model must handle device-specific processing (e.g., beacon selection) and exceptions such as limited fields of view or occlusions. Any measurement model can be defined to test for occlusion and include its effects in its calculation. This is the case with the system described in Section 5.1.1.2 and, while not included in the analysis presented here, Chen [**24**] presents a probabilistic model for dynamic self-occlusion that could also be incorporated into a measurement model.

It is important to note that the sample time δt, for example in $A(\delta t)$ and $Q(\delta t)$, is defined by the *measurement devices*. If the candidate devices provide measurements at 100 Hz, then $\delta t = 0.01$ seconds. If there are multiple devices with different measurement rates, then there are multiple corresponding δt values, with corresponding instances of $A(\delta t)$ and $Q(\delta t)$.

TABLE 3.3. Pseudo-code for steady-state evaluation

```
For each $n$-dimensional point $\bar{x}_i \in \{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_p\}$
    $\Psi_\Sigma = \text{zeros}(2n, 2n)$
    For each device
        Determine $\delta t$ for the device
        Evaluate $A$ with that $\delta t$ using (22)
        Evaluate $Q$ with that $\delta t$ using (23)–(25)
        Evaluate $H$ at $\bar{x}_i$
        Evaluate $R$ at $\bar{x}_i$
        Test for Observability)
        If Observable
            Compute $\Psi$ using (28)
            $\Psi_\Sigma = \Psi_\Sigma + \Psi$
    Compute the $n$ eigenvectors of $\Psi_\Sigma$ as in (29)
    Compute $P_i^\infty$ as in (30)
```

Finally one has to decide at what points $\{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_p\}$ to evaluate P^∞. One could choose a set of points on a surface or object in the working volume, or a set that spans some 3D volume, perhaps on a regular 3D grid. Examples of both are presented in Section 5.

The pseudo-code in Table 3.3 illustrates the overall process. Notice how the contributions from each device are similarly fused at every point, no matter what type of device (or combination of devices) is being evaluated. Note that the H and R must be evaluated with the appropriate measurement (or Jacobian) function for the device.

Once the P^∞ analysis is complete, one can use surface or volume visualization techniques to render the complete set of points P_i^∞ for $1 \leq i \leq p$ as in the volume visualization for the example acoustic system (Figure 1.6) and throughout Chapters 4, 5, 6, and 7.

3.7. Domain Qualification: Existence of a Solution to the DARE

For readers familiar with the Kalman Filter, Section 3.7.1 provides a segue between the Kalman filter equations and the closed-form equation used in this framework's P^∞ solution. Kalman filter familiarity not withstanding, the proof of the existence of a steady-state covariance solution (P^{∞}) in this domain does rely on one of the equations presented in Section 3.7.1.

3.7.1. Iterative Solution. The Kalman Filter [**54, 100**] is a stochastic estimator for the instantaneous state of a dynamic system (Section 3.3.4) that can be used as a tool for performance analysis [**42**] when actual measurements are available and has been used for both human motion modeling [**108, 92, 77**] and tracking of human motion [**56, 34**]. It might appear that this framework is a Kalman filter-based tracking approach as in [**38**] or [**101**]. In fact, the Kalman Filter is a measurement-dependent, iterative solution to the well-known Riccati (Ordinary Differential) Equation as shown in Section 3.7.1 as compared to the closed form steady-state solution employed by this framework and presented in Section 3.5.1.

If we had real or simulated measurements, P^{∞} could be determined via an iterative solution, the Kalman Filter [**54**]. Given A, H, Q, R, and P_0 (the initial covariance matrix required only if using an iterative approach), we can describe the prediction-correction cycle Kalman Filter algorithm. The first step is to *predict* the system state using the time update. We begin with a state estimate, x_k, and estimate the following state, x_{k+1}, after an elapsed time δt. All steps in the time update are a priori of a real measurement of the outputs of the system. The second and final step is the measurement update that begins when a new measurement is received. First we predict the Kalman gain, K. Then given a measurement vector, z_{i+1}, we *correct* the estimate for x_{i+1} from the time update (prediction) step. We can then correct the covariance estimate. By iterating until until $P_{i+1} = P_i$, the steady-state covariance is determined. The Kalman Filter time update equations (i.e. prediction) are as follows:

$$\hat{x}_i^- = A\hat{x}_{i-1}^- \tag{31}$$

$$P_i^- = AP_{i-1}A^T + Q \tag{32}$$

and the measurement update equations (i.e. correction) are:

$$K_i = P_i^- H^T (HP_i^- H^T + R)^{-1} \tag{33}$$

$$\hat{x}_i^+ = \hat{x}_i^- + K_i(z_i - H\hat{x}_i^-) \tag{34}$$

$$P_i^+ = (I - K_i H)P_i^- \tag{35}$$

where K is called the "Kalman Gain".

Assuming that i is far enough into the future so that continued iterations result in no change (i.e. $P_{i+1} = P_i$ and $P^- = P^+$) we can drop the i subscript and $\pm$ superscripts for simplicity and, by substitution:

$P = (I - KH)P$ *Equation (35)*

$P = (P - KHP)$ *multiplying through*

$P = \left(P - \left(PH^T \left(HPH^T + R\right)^{-1}\right) HP\right)$ *substituting Equation (33)*

$P = A\left(P - \left(PH^T \left(HPH^T + R\right)^{-1}\right) HP\right) A^T + Q$ *substituting Equation (32)*

$P = APA^T + Q - APH^T \left(HPH^T + R\right)^{-1} HPA^T$ *multiplying through, reordering.*

This final equation is the Discrete Algebraic Riccati Equation where $P_i = P_{i+1} = P^\infty$.

3.7.2. Existence Conditions. In order to determine whether we can successfully solve the DARE, we must define conditions to be satisfied for the existence of a solution [**52**] which because of its non-linear nature, is not guaranteed in all cases. A covariance matrix, P, is always non-negative definite or, equivalently, positive semi-definite (p.s.d). This means that all its eigenvalues are less then or equal to one and so all acceptable solutions to the DARE will be positive semi-definite or positive-definite as well. From this, we know that we must find a positive semi-definite (or definite) solution to the DARE. However, the solution of interest is further restricted. Of interest is the stabilizing solution which, when it exists, is unique. Using the state equations from the iterative solution to the DARE presented in Section 3.7.1, we know that

$$\hat{x}_i^+ = A\hat{x}_i^- + K_i(\bar{z}_i - H\hat{x}_{i-1}^-) \tag{36}$$

where $K_i = P_i^- H^T (HP_i^- H^T + R)^{-1}$ (the Kalman Gain) and $\tilde{z}_i = \bar{z}_i - H\hat{x}_{i-1}^-$ is the measurement residual, the difference between the actual measurement ($\bar{z}_i$) and the estimated measurement ($H\hat{x}_{i-1}^-$) as introduced in Section 3.2. Subtracting to find the error in the state estimate ($\tilde{x}_i$),

$$\begin{aligned} \tilde{x}_i &= \bar{x}_i - \hat{x}_i \\ &= A\bar{x}_{i-1} - (A\hat{x}_{i-1} + K_i(\bar{z}_i - H\hat{x}_{i-1}^-)) \\ &= (A - KH)\tilde{x}_{i-1} \\ &= (A - KH)^i \tilde{x}_0 \end{aligned} \tag{37}$$

Since the $(A - KH)$ term will be applied to the state estimate repeatedly over time, the stability of $(A - KH)$ is crucial to avoid exponential growth of error and instability. Therefore, $(A - KH)$ must be stable in order for error convergence to occur. However, because of the random variables, w and v, the stability of $(A - KH)$ alone is necessary but not sufficient. The actual error will fluctuate around a mean value of the state and the value of K minimizes the average value of some function of the error. When this function is the mean-square-error, the K that minimizes the error covariance of the state is the same K_i that we

TABLE 3.4. Existence of the DARE solution [**52**]

Properties of the Solution	Conditions	Remarks
Is there a solution of the DARE such that (A-KH) is semi-stable, i.e. $\lvert\lambda(A-KH)\rvert \leq 1$?	Yes, under detectability of {A,H} (only a sufficient condition).	At least one such solution is p.s.d.
Is there a solution of the DARE such that (A-KH) is stable, i.e. $\lvert\lambda(A-KH)\rvert < 1$?	Yes, iff we have detectability and unit circle controllability of $\{A,Q\}$.	The stabilizing solution is unique and p.s.d. However, there can be several p.s.d. solutions.
When is the stabilizing solution of the DARE its unique p.s.d. solution?	Iff we have detectability and stabilizability (controllability on and outside the unit circle) of $\{A,Q\}$.	

use in the Kalman filter (Equation (33)) and we see contained in the DARE (Equation (27)) as shown in Equation (38).

$$K = K_i = P_i^- H^T (HP_i^- H^T + R)^{-1} \tag{38}$$

Therefore, by definition, the stabilizing solution to the DARE is the solution such that the matrix $(A-KH)$ is stable. In other words, the eigenvectors of $(A-KH)$ must lie inside the unit circle so that their magnitudes are less than one ($\lambda|(A-KH)| < 1$), a condition for positive-definiteness. Table 3.4 summarizes the conditions under which stabilizing solutions to the DARE exist. These conditions move through three phases from the existence of solution(s) to the DARE to the existence of several solutions (one of which is stabilizing) to a *unique* stablilizing solution.

The first step is testing for detectability of $\{A,H\}$. The detectability of $\{A,H\}$ means that there exists a matrix K such that $(A-KH)$ is semi-stable (eigenvalues on and/or inside the unit circle) and, therefore, a solution(s) to the DARE exists such that $(A-KH)$

is semi-stable. Eigenvalues on and/or inside the unit circle is a condition for positive-semidefiniteness (p.s.d.). Without detectability, we cannot claim the possibility of a positive semi-definite solution to the DARE. Therefore, detectability is a necessary but not sufficient condition for a stabilizing solution since without detectability, $(A - KH)$ is not stable for any K and, so, $(A - KH)$ cannot be stable. Once we have shown that a positive semi-definite solution exists, the next step is to test for the existence of a stabilizing solution. Along with detectability, we must show that $\{A, Q\}$ is unit circle controllable.

Unit circle controllability means that there exists some matrix K such that $(A - QK)$ has no unit-circle eigenvalues ($|\lambda_i| < 1$). When A, is stable the unit-circle controllability assumption is automatically met. Detectability and controllability on the unit circle tells us that a stabilizing positive semi-definite solution to the DARE exists. However, it does not eliminate the possibility of several positive semi-definite solutions. From these, we would have to test the stability of $(A - KH)$ directly by substituting the resulting value of P and calculating the eigenvalues of $(A - KH)$. Only one solution will be stabilizing.

Once the existence of a stabilizing solution has been proven, the next test determines whether there is a unique solution to the DARE. If there is, this solution is positive semi-definite and stabilizing. Just as we tested for unit circle controllability above, we can test for controllability outside the unit circle. This means that there exists some matrix K such that $(A - QK)$ has no eigenvalues on or outside the unit-circle (i.e. $(A - Q)$ is stabilizable). In other words, if we have detectability and controllability on the unit circle then the DARE will have only one positive semi-definite solution if and only if $\{A, Q\}$ is stabilizable. This positive semi-definite solution of the DARE is its unique stabilizing solution. As with controllability, when A is stable the stabilizability assumption is automatically met.

The flow chart in Figure 3.5 summarizes the conditional tests for detectability, controllability and stabilizability and the assertions that can be made after each test.

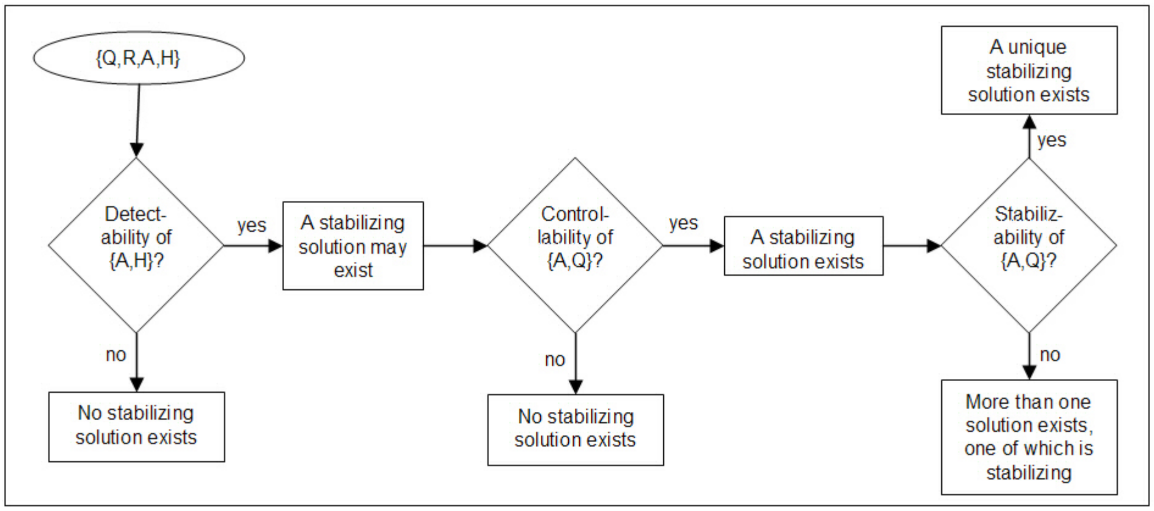

{Q,R,A,H}
Detect-ability of {A,H}?
yes
no
No stabilizing solution exists
A stabilizing solution may exist
Control-lability of {A,Q}?
yes
no
No stabilizing solution exists
A stabilizing solution exists
Stabiliz-ability of {A,Q}?
yes
A unique stabilizing solution exists
no
More than one solution exists, one of which is stabilizing

FIGURE 3.5. DARE Existence Test Flow

3.7.3. Existence in this Tracking Domain. This section will examine9 the P, PV and PVA models for the existence of a solution of the DARE. It is not intended to serve as proof that the DARE can be applied to all tracking system models. Tracking systems models vary depending on the defined values of Q, R, A, H and each should be tested as defined.

Suppose we are designing a 1D-position tracker with a PV-motion model. The state matrix will contain an estimate for $\hat{x}$ and we also want to estimate velocity, $\hat{v} = \hat{\dot{x}} = \frac{d\ddot{x}}{dt}$. A typical A matrix in this position-velocity (PV) system applies the equation $\hat{x}_{i+1} = \hat{x}_i + \hat{v}\delta t$, to determine the new position estimate, $\hat{x}_{i+1}$, and replaces the current velocity estimate, $\hat{\dot{x}}$, with the new one. For example,

(39) $$A = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix} \; given\ state\ definition \begin{bmatrix} \hat{x} \\ \hat{\dot{x}} \end{bmatrix}.$$

Solving for $\hat{x}_{i+1}$,

(40) $$\hat{x}_{i+1} = A\hat{x}_i = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} \hat{x}_i \\ \hat{\dot{x}}_i \end{bmatrix} = \begin{bmatrix} \hat{x} + \hat{\dot{x}}_i\delta t \\ \hat{\dot{x}}_i \end{bmatrix}$$

Most tracking systems measure 3D position or 6D position and orientation and we find that A extends from 1D to 3D such that

$$A = \begin{bmatrix} 1 & \delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad where\ \hat{x}_i = \begin{bmatrix} \hat{x} \\ \hat{\dot{x}} \\ \hat{y} \\ \hat{\dot{y}} \\ \hat{z} \\ \hat{\dot{z}} \end{bmatrix} \tag{41}$$

if the state definition includes estimates for position in $\hat{y}$ and $\hat{z}$. We observe that A is an upper triangular matrix. The upper triangular form of A holds for the 6D position and orientation case and in the position (P) and position-velocity-acceleration (PVA) models in which $\hat{x}_{i+1} = \hat{x}_i + v(\delta t) + \hat{a}(\delta t)^2$ where $\hat{a} = \hat{\dot{v}}$ is estimated acceleration. Regardless of the update time, δt, the eigenvalues of A are all equal to one since the eigenvalues of a triangular matrix are the diagonal elements of the matrix. We know that if A is stable then the stabilizability condition of $\{A, Q\}$ is automatically met and no further testing would be required. However, in the P-PV-PVA domain considered here, A is only semi-stable ($\lambda_i(A) = 1$).

The $\{A, H\}$ detectability test determines whether there exists a matrix K such that $(A - KH)$ is semi-stable. Because the eigenvalues of A are all equal to one, we know immediately that A is semi-stable and a solution to the DARE exists. Regardless of the value of H, if we simply set $K = 0$, the eigenvalues of $(A - KH = A - 0 = A)$ remain equal to one proving that $\{A, H\}$ is semi-stable as long as A is of P, PV, or PVA form. This passes the first condition for detectability and assures the existence of a positive semi-definite solution to the DARE.

Next we test for the existence of a stabilizing solution via the $\{A, Q\}$ detectability condition. To prove that a stabilizing solution exists, we must show that there exists some matrix, K, such that $A - QK$ has no unit-circle eigenvalues. Recall that Q is the process noise covariance that we know to be positive, symmetric and positive-definite and so as long as $K \neq 0$ then we can be certain that $(A - QK)$ will have no unit-circle eigenvalues since the diagonals and the eigenvalues of A alone are one. Therefore, we know that a stabilizing solution to the DARE exists. This is enough to enable us to move forward with the solution(s) to the DARE. However, we really want to prove the existence of a *unique* stabilizing solution since it is the best possible scenario for this problem.

Now that we have met the unit-circle controllability condition, we are assured that a stabilizing solution to the DARE exists. The condition for stabilizability (unit-circle controllability outside the unit circle) tells us whether or not the stabilizing solution is the unique solution to the DARE. While this is not necessary for implementation of the DARE approach to tracker performance evaluation, a unique solution to the DARE simplifies the process considerably. To prove a unique solution, the $\{A, Q\}$ detectability and stabilizability must be met so that $(A - QK)$ can have no eigenvalues on or outside the unit circle. Building on the argument presented in the previous paragraph, if we can show that (QK) is positive then the subtraction from A will result in a negative perturbation of $(A - QK)$ and, consequently, eigenvalues not equal to one. We know that covariance Q contains only positive elements so if we can assert the same for K then (QK) is also positive. Recall the K defined in Equation (38). An inspection of this equation shows that K is a ratio of functions of the process covariance, σ_p, to the measurement covariance, σ_m, such that, generally speaking:

$$K \equiv \frac{\sigma_p}{\sigma_p + \sigma_m} \tag{42}$$

and so will be positive and less than one. We can determine the sensitivity of the eigenvalues of A by treating (QK) as a perturbation on A to see whether $(A - QK)$ does indeed result in eigenvalues of less than one. Because A is not symmetric we know that $A's$ eigenvalues will be sensitive to perturbation. Given the eigenvalues (λ) and eigenvectors (x) of A we know that $A = x\lambda x^{-1}$ and $\lambda = x^{-1}Ax$. If δA represents some change in A then $\lambda + \delta\lambda = x(A + \delta A)x^{-1}$ and so $\delta\lambda = x(\delta A)x^{-1}$ where, in this case, $\delta A = QK$. Taking matrix norms, $\|\delta\lambda\| \leq \|x\|\|\delta A\|\|x^{-1}\| = \|x\|\|x^{-1}\|\|\delta A\| = \kappa(x)\|\delta A\|$ where $\kappa(x)$ is the condition number of the matrix of eigenvectors of A. In the P-PV-PVA cases, we find condition numbers on the order of 10^{13} and so it would be reckless to assert any definitive statement about whether the eigenvalues of $\delta A = QK$ are less than one. Fortunately, there remains another, more efficient way to determine whether a unique stabilizing solution to the DARE exists.

3.7.4. Observability Test. If a stabilizing solution exists then the system covariances converge over time to a steady-state solution or, as discussed above, a solution to the DARE exists [**52**]). However, if a system satisfies the stronger condition of observability then we know a unique solution to the DARE exists [**60**]. Therefore, we can prove the existence of a stabilizing solution via a system observability test. Observability is an important property of dynamic systems that employ feedback such as tracking systems [**46**]. A system is observable if it is possible to uniquely determine the state from the outputs (measurements) at any point in time which is, of course, the very purpose of a tracking system. Using the 3D acoustic example, a single measurement defines a sphere upon which the user may lie, i.e. there are an infinite number of possible solutions. Consequently, we cannot uniquely determine the state with a single measurement. For this system to be observable, you need at least three (four in practice) range measurements to determine the 3D position [**102**]. For it to be practical, the final system must be observable [**85**]. However, this does not mean that every system model a designer tries will be observable, nor that the observability will be satisfied throughout the working volume (occlusion, for example, would not be detected). Before attempting to solve the DARE at any point in the working volume, a test for observability using the following test is necessary [**42**]:

(43) $$Rank_{Observability} \equiv \left(rank\left(O\left(H_\mu, A_\mu, \quad 1 \le \mu \le \mu_m\right)\right) = n\right)$$

where n is the number of elements in the state vector and

(44) $$O(H_\mu, A_\mu, \quad 1 \le \mu \le \mu_m) \equiv \sum_{\mu=1}^{\mu_m} \left\{ \left[\prod_{i=0}^{\mu-1} A_{\mu-i}\right]^T H_\mu^T H_\mu \left[\prod_{i=0}^{\mu-1} A_{\mu-i}\right]^T \right\}$$

for all m measurement model objects and corresponding state transition and measurement matrices, A_μ and H_μ. If the point in question is not observable, a DARE solution may exist but it will no be a stabilizing solution.

CHAPTER 4

Computational Model: Artemis

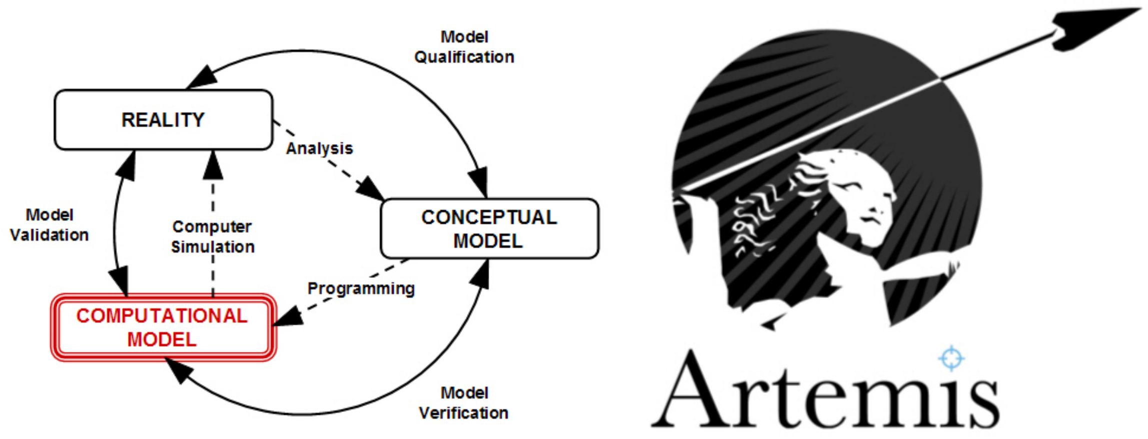


FIGURE 4.1. Artemis: The Computational Model

Named for the Greek goddess of hunting and tracking, Artemis is the software tool that implements the P^{∞} estimation method with built-in support for P, PV and PVA motion models and a variety of measurement models. Artemis is written in RSI's Interactive Data Language$^{\text{TM}}$ (IDL) [**84**], a development environment targeted for data analysis and visualization applications. IDL was chosen because of its built-in image processing, matrix math, Graphical User Interface (GUI) builder, data analysis and visualization capability along with cross-platform portability. IDL's free Virtual Machine supports deployment across multiple platforms and Artemis has run successfully on Windows, OS X and Linux platforms.

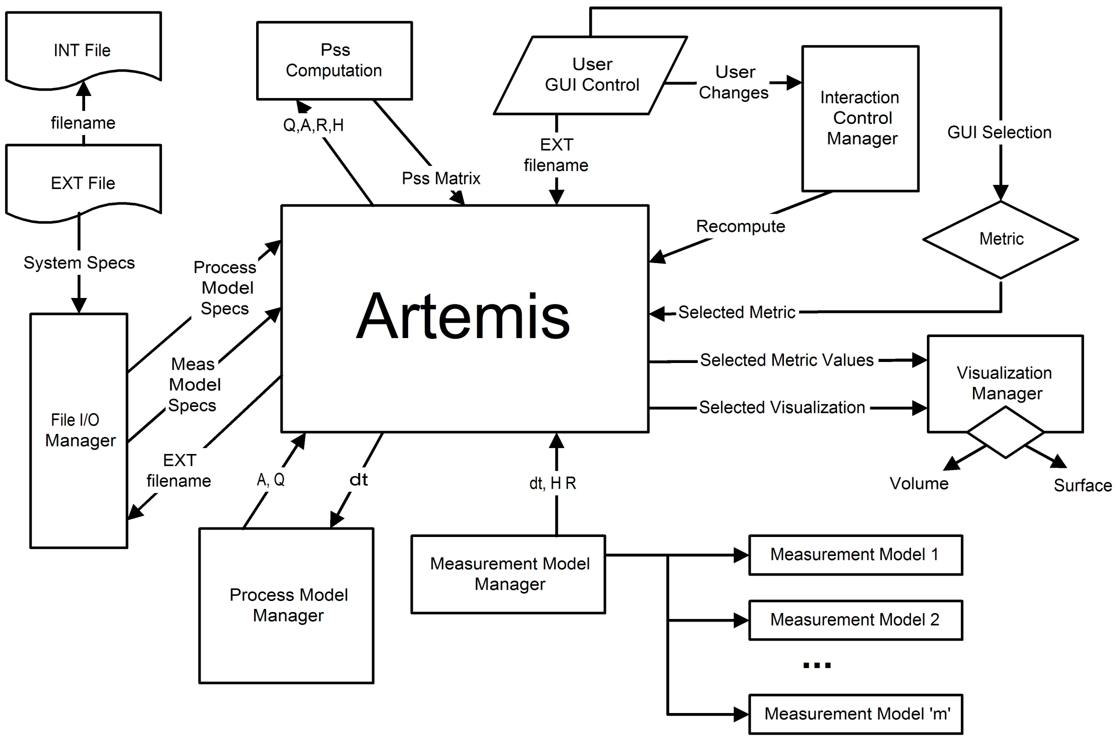


FIGURE 4.2. Artemis Functional Architecture

4.1. Architecture

The Artemis software is comprised of seven primary functional components as shown in Figure 4.2. These functions are:

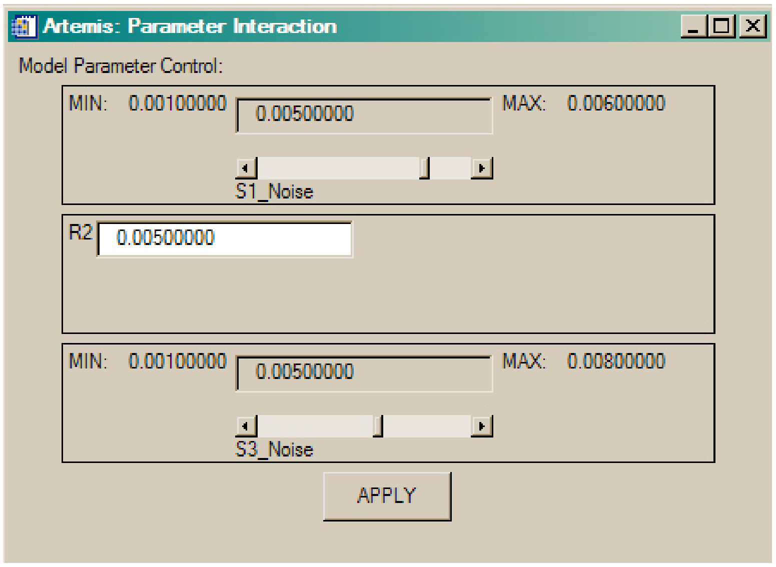
(1) Process Model Manager
(2) Measurement Model Manager
(3) Interaction Control Manager
(4) Visualization Manager
(5) File IO Manager
(6) P^{∞} Computation Manager
(7) GUI Control Manager

The Artemis user interacts with GUIs controlled by the Interaction Control Manager and Visualization Manager. Examples of these GUIs are shown throughout this section with positional numbers in meters and orientation values in degrees. Note that Artemis does not limit the units that can be used and requires only that the user be unit-consistent within any given analysis for correct computation.

4.1.1. Process Model Manager. When provided with a definition of the system state and values for building the process noise matrix, Q, the Process Model Manager instantiates the process model to be used in the tracking system performance analysis. The size and form of the state vector determines the size and form of the Q matrix and the Q matrix components are determined at runtime by the sampling rate (δt) of each measurement model and the "q" values provided by the user. See Section 4.2.1 for details about the process model form and value specification.

4.1.2. Measurement Model Manager. The Measurement Model Manager registers each system measurement model and keeps track of how many have been registered and of which types. During runtime, the Measurement Model Manager provides the sampling rate (δt) for the generation of A and Q by the Process Model Manager and the H and R matrices from each instantiated measurement model for the P^{∞} calculation.

4.1.3. Interaction Control Manger. Every system parameter (Q, R, position, orientation, etc.) is registered with the Interaction Control Manager and assigned a parameter ID after the user specifies the necessary input files. Measurement specific parameters (e.g., FOV) are also registered with the Interaction Control Manager and assigned IDs. Along with the parameter ID and value, an interaction control mode is defined. If the parameter is static (i.e., no real time user control) then its model type is STATIC. For dynamic parameters, users can request control though either SLIDER or TEXT BOX interaction as shown in Figure 4.3. This GUI is displayed automatically if dynamic parameters are specified. The Interaction Control Manager also supports "grouping" of parameters. For example, suppose a system contains a number of sensors placed at various [x,y] positions around a room but at the same height (z). The z-location of all the sensors can be grouped and tied to a single slider or text box so that the user can interactively raise and lower the entire sensor setup. See Section 4.2.5 for further details.



MIN: 0.00100000 | 0.00500000 | MAX: 0.00600000
S1_Noise

R2 0.00500000

MIN: 0.00100000 | 0.00500000 | MAX: 0.00800000
S3_Noise

FIGURE 4.3. Artemis Parameter Interaction GUI with sliders and text boxes

4.1.4. Visualization Manager. The Visualization Manager accepts the values of the metric selected by the user (see the Artemis GUI in Figure 4.4) for display along with the display type (volume or pointlist/surface). If the display type is a surface visualization, then the user must also specify what type of interpolation method to use. The available choices are:

- Inverse Distance
- Linear
- Minimum Curvature
- Natural neighbor
- Nearest Neighbor.

and the user decides which of these interpolations methods to use along with the desired metric to visualize via the main Artemis GUI (Figure 4.4).

4.1.5. File IO Manager. The File IO Manager handles all data read from and saved to files. It parses the input files and passes the parameters specified to the Process Model and Measurement Model Managers. The File IO Manager is activated when the user presses the “Load Data File” and “Save Data File” buttons on the main Artemis GUI (Figure 4.4). P^{∞} results can be saved off for later retrieval without having to recalculate P^{∞} values.

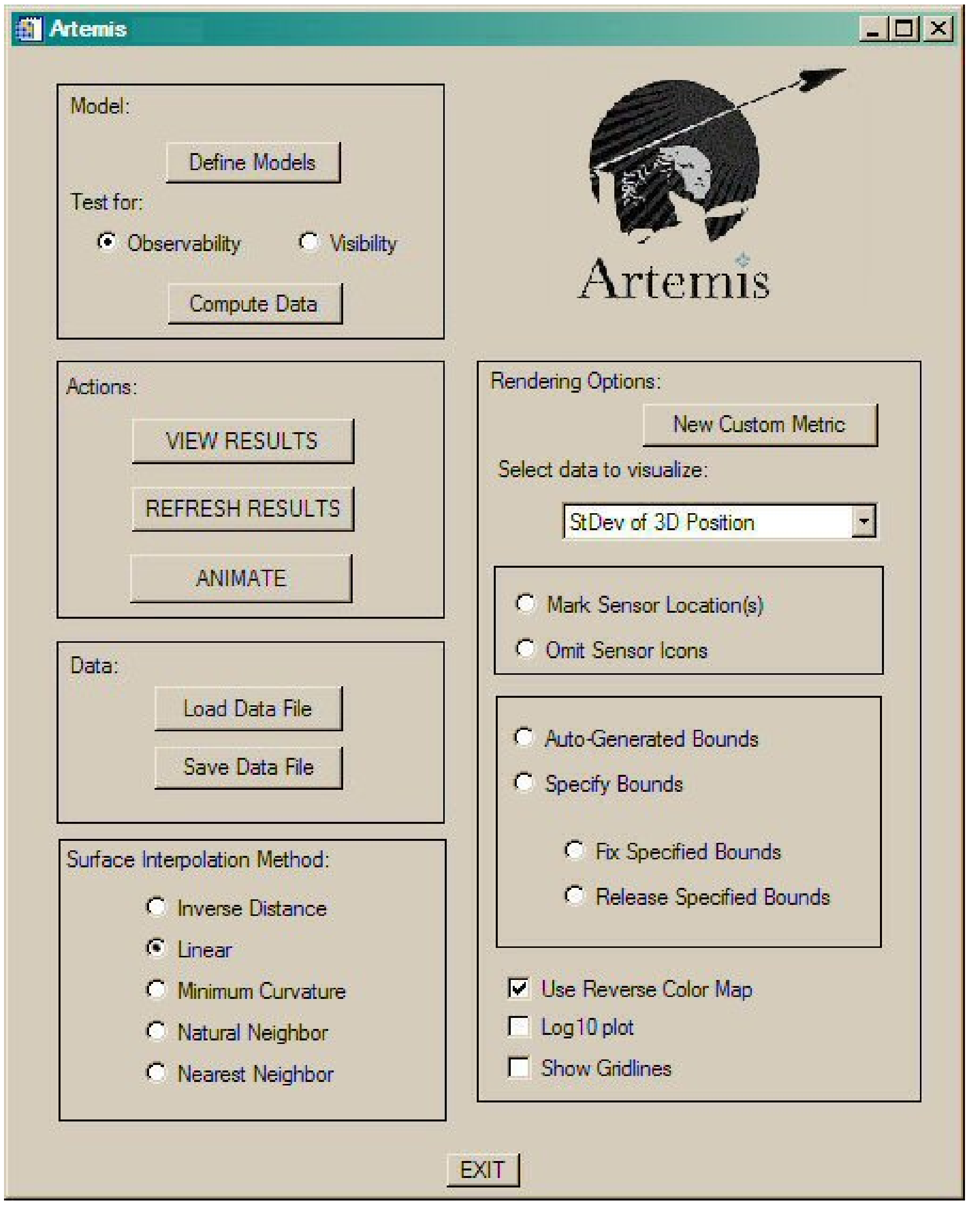
4.1.6. P^{∞} Computation Manager. The P^{∞} Computation Manager interfaces with both the Process Model Manager and the Measurement Model Manager to collect the Q, A, H, R matrices and δt for each measurement model and at every analyzed point. Using the algorithm described in Table 3.3, a P^{∞} matrix is computed for every registered measurement model and these matrices are then summed for a final P^{∞}. When P^{∞} has been calculated at every sampled point, the metrics listed in Section 4.3 are generated and made available for the Visualization Manager.

4.1.7. GUI Control Manager. The GUI Control Manager monitors for user input and routes tasks to the appropriate code modules. The user interfaces and operational procedures are described in the following section (Section 4.2) and the main Artemis GUI is pictured in Figure 4.4.

4.2. Using Artemis

Figure 4.4 is a screenshot on the main Artemis GUI. To define a model for P^{∞} estimation, the Artemis user begins by pressing the “Define Models” button and the system prompts for a model definition file (an ‘.ext’ file extension).

4.2.1. Process Model and Volume/Surface Specification. The tracking system to be analyzed is instantiated with information from the extrinsic (‘.ext’) input file as shown in Figure 4.5. The first line defines whether the state space will contain position and/or orientation parameters and which of the three available motion models to use. In Figure 4.5, a 3D-position state with PV motion is defined (P_PV). Orientation would be included in



Test for: [✓ Observability] Visibility

Surface Interpolation Method: Inverse Distance [✓ Linear] Minimum Curvature Natural Neighbor Nearest Neighbor

Select data to visualize: StDev of 3D Position

☑ Use Reverse Color Map
☐ Log10 plot
☐ Show Gridlines

FIGURE 4.4. Artemis GUI

the state if PO_PV had been requested instead. The second line defines the type of analysis to be performed. "Volume" indicates volume analysis and the following three lines define the volume parameters. The three lines following define the grid or mesh over which P^{∞} calculations will be performed. The first line specifies the starting coordinates for x, y and z. The second line specifies the end coordinates for x, y and z. The third line specifies the spacing or delta between the start and end coordinates for x, y and z. Figure 1.6 shows the result of the volume analysis specified in Figure 4.5 (left).

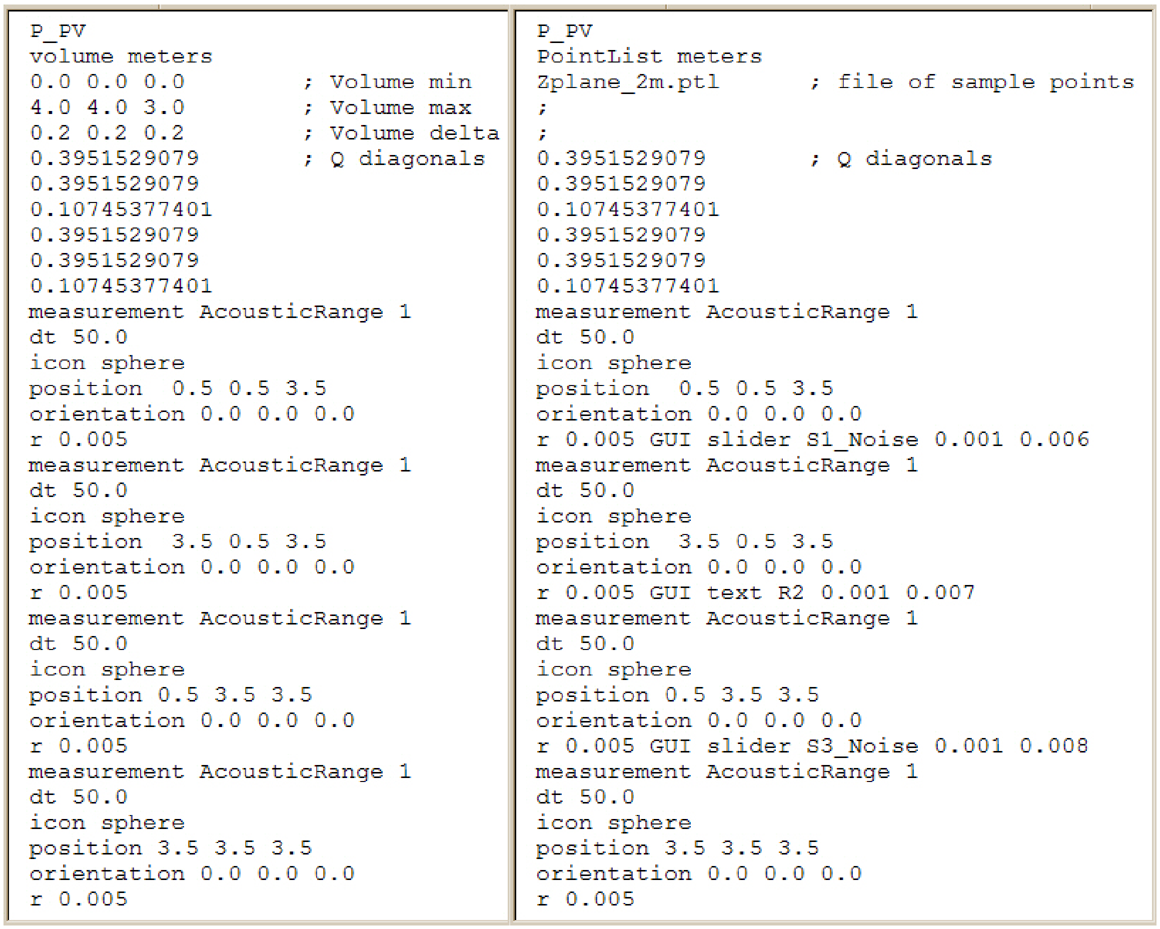
If instead of a volume analysis, an analysis of specific points is desired, a “PointList” can be specified in which Artemis will perform the P^{∞} analysis on the 3D points read in a file as specified in Figure 4.5 (right). “Zplane_2m.ptl” contains lines of 3D points in x, y, z order. For example, the first few lines of the pointlist file “Zplane_2m.ptl” which defines a plane at $z = 1.9$ meters are:

```
3.0      6.0      1.9
3.0      6.1      1.9
3.0      6.2      1.9
3.0      6.3      1.9
3.0      6.4      1.9
3.0      6.5      1.9
```

Pointlists can also be used to define a model such as the torso shown in Figure 4.12. P^{∞} is performed at the vertices that comprise the geometric model.

The next lines define the “Q” matrix. Each “q” value is read from a single line in [qx qy qz qR qP qY] per line order as required by the state definition. The number of lines required to define the Q-matrix varies with the selected state space. For example, if the state is position-only then three lines are needed for qx, qy, and qz. If the state is orientation only, then three lines are needed for qR, qP and qY. If the state is 6D (position and orientation) then six lines are required in the order specified above (position followed by orientation). The Q matrix defined in Figure 4.5 is 6×6. After the Q matrix values have been defined, the state space and associated process model are complete except for measurement update rate which is defined with the measurement model(s). The cognizant Artemis user can define the process noise matrix “Q” in any way he or she wants by circumventing this tool-provided matrix generation inside the IDL file (i.e., code) for the measurement model.

4.2.2. Measurement Model Specification. The remainder of the EXT input file defines the measurements that comprise the proposed tracking system. In both the volume and



```
P_PV
volume meters
0.0 0.0 0.0          ; Volume min
4.0 4.0 3.0          ; Volume max
0.2 0.2 0.2          ; Volume delta
0.3951529079         ; Q diagonals
0.3951529079
0.10745377401
0.3951529079
0.3951529079
0.10745377401
measurement AcousticRange 1
dt 50.0
icon sphere
position  0.5 0.5 3.5
orientation 0.0 0.0 0.0
r 0.005
measurement AcousticRange 1
dt 50.0
icon sphere
position  3.5 0.5 3.5
orientation 0.0 0.0 0.0
r 0.005
measurement AcousticRange 1
dt 50.0
icon sphere
position 0.5 3.5 3.5
orientation 0.0 0.0 0.0
r 0.005
measurement AcousticRange 1
dt 50.0
icon sphere
position 3.5 3.5 3.5
orientation 0.0 0.0 0.0
r 0.005
```

```
P_PV
PointList meters
Zplane_2m.ptl        ; file of sample points
;
;
0.3951529079         ; Q diagonals
0.3951529079
0.10745377401
0.3951529079
0.3951529079
0.10745377401
measurement AcousticRange 1
dt 50.0
icon sphere
position  0.5 0.5 3.5
orientation 0.0 0.0 0.0
r 0.005 GUI slider S1_Noise 0.001 0.006
measurement AcousticRange 1
dt 50.0
icon sphere
position  3.5 0.5 3.5
orientation 0.0 0.0 0.0
r 0.005 GUI text R2 0.001 0.007
measurement AcousticRange 1
dt 50.0
icon sphere
position 0.5 3.5 3.5
orientation 0.0 0.0 0.0
r 0.005 GUI slider S3_Noise 0.001 0.008
measurement AcousticRange 1
dt 50.0
icon sphere
position 3.5 3.5 3.5
orientation 0.0 0.0 0.0
r 0.005
```

FIGURE 4.5. Artemis input file for volume (left) and surface (right) system analysis

pointlist/surface files shown in Figure 4.5, four acoustic range sensors are specified one-at-a-time beginning with the

```
measurement AcousticRange 1
```

line. Following each measurement delineation line, the position, orientation, update rate (δt in Hz) and measurement noise (r) for each sensor are specified. These are order independent. The optional "icon" specification supports the selection of any of the predefined models (camera, sphere, cube, ...) to be used to mark the sensor locations during visualization. In this acoustic example, a "sphere" is chosen and spheres can be seen marking the location of the system sensors in Figure 1.6. An example of the "camera" icon can be seen in Figure 5.15. Note that in the input file for TeX surface analysis (right), three of the four

’r’ values are followed by the word “GUI”. This indicates that the user wants a dynamic input parameter and Artemis will create the required parameter interaction automatically. The requested Parameter Interaction window for this input file is shown in Figure 4.3. See Section 4.2.4 for details.

We could choose to specify all four sensors with one line

```
        measurement AcousticRange 4
```

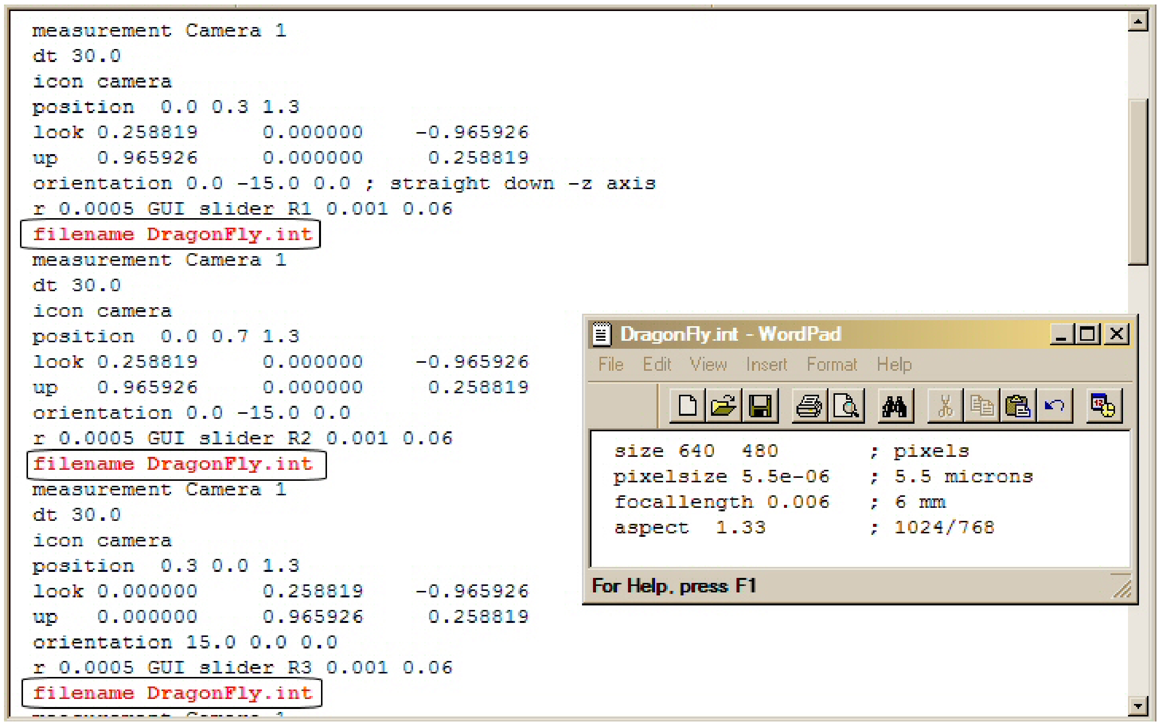
and follow that with four sets of specification flags (δt, icon, position, orientation, r) but this saves only a few lines of input text and is not as maintainable as four separate “measurement” entries.

Whether read-in individually or as a group, each individual measurement definition (four in this example case) is registered with the Measurement Model manager and instantiated. Artemis currently provides measurement models for the following sensor types:

- Acoustic
- Pinhole Camera
- Calibrated Camera (defined by transformation matrices)
- HiBall
- Inertial
- Generic Range
- Custom (user-defined).

Artemis provides a template file that contains the required procedures and function for a user to employ when defining custom metric.

4.2.3. Intrinsic Parameters. Intrinsic parameters can be specified directly in the EXT file or, more conveniently, with an INT file. This is especially useful for sensor system types such as cameras with a number of internal parameters (CCD size, focal length, etc.) when used multiple times in a tracking system specification. This enables a user to specify the position and orientation of each camera in the EXT file and simply reference the INT file for intrinsic parameters. For example, Figure 4.6 shows the specification of several cameras in an eight-camera setup all of which are defined defined using the same intrinsic parameter file (circled and in red).



```
measurement Camera 1
dt 30.0
icon camera
position  0.0 0.3 1.3
look 0.258819     0.000000    -0.965926
up   0.965926     0.000000     0.258819
orientation 0.0 -15.0 0.0 ; straight down -z axis
r 0.0005 GUI slider R1 0.001 0.06
filename DragonFly.int
measurement Camera 1
dt 30.0
icon camera
position  0.0 0.7 1.3
look 0.258819     0.000000    -0.965926
up   0.965926     0.000000     0.258819
orientation 0.0 -15.0 0.0
r 0.0005 GUI slider R2 0.001 0.06
filename DragonFly.int
measurement Camera 1
dt 30.0
icon camera
position  0.3 0.0 1.3
look 0.000000     0.258819    -0.965926
up   0.000000     0.965926     0.258819
orientation 15.0 0.0 0.0
r 0.0005 GUI slider R3 0.001 0.06
filename DragonFly.int
```

```
size 640   480         ; pixels
pixelsize 5.5e-06      ; 5.5 microns
focallength 0.006      ; 6 mm
aspect   1.33          ; 1024/768
```

FIGURE 4.6. Artemis input file (.INT) for intrinsic parameters

4.2.4. Parameter Control and Interaction.

Given the goal of design interaction, Artemis implements slider and text box control of designated parameters. To request realtime interaction with a design parameter, the “GUI” flag must be appended to the parameter specification as shown below.

```
measurement CameraVector 1
dt 50.0
icon sphere
positionX    0.194465 GUI slider positionX -0.5 2.0
positionY    1.83907  GUI slider positionY -0.5 2.0
positionZ   -0.399036 GUI slider positionZ -0.5 2.0
orientationR     90.0 GUI textbox orientationR}
orientationP      0.0
orientationY    -90.0
```

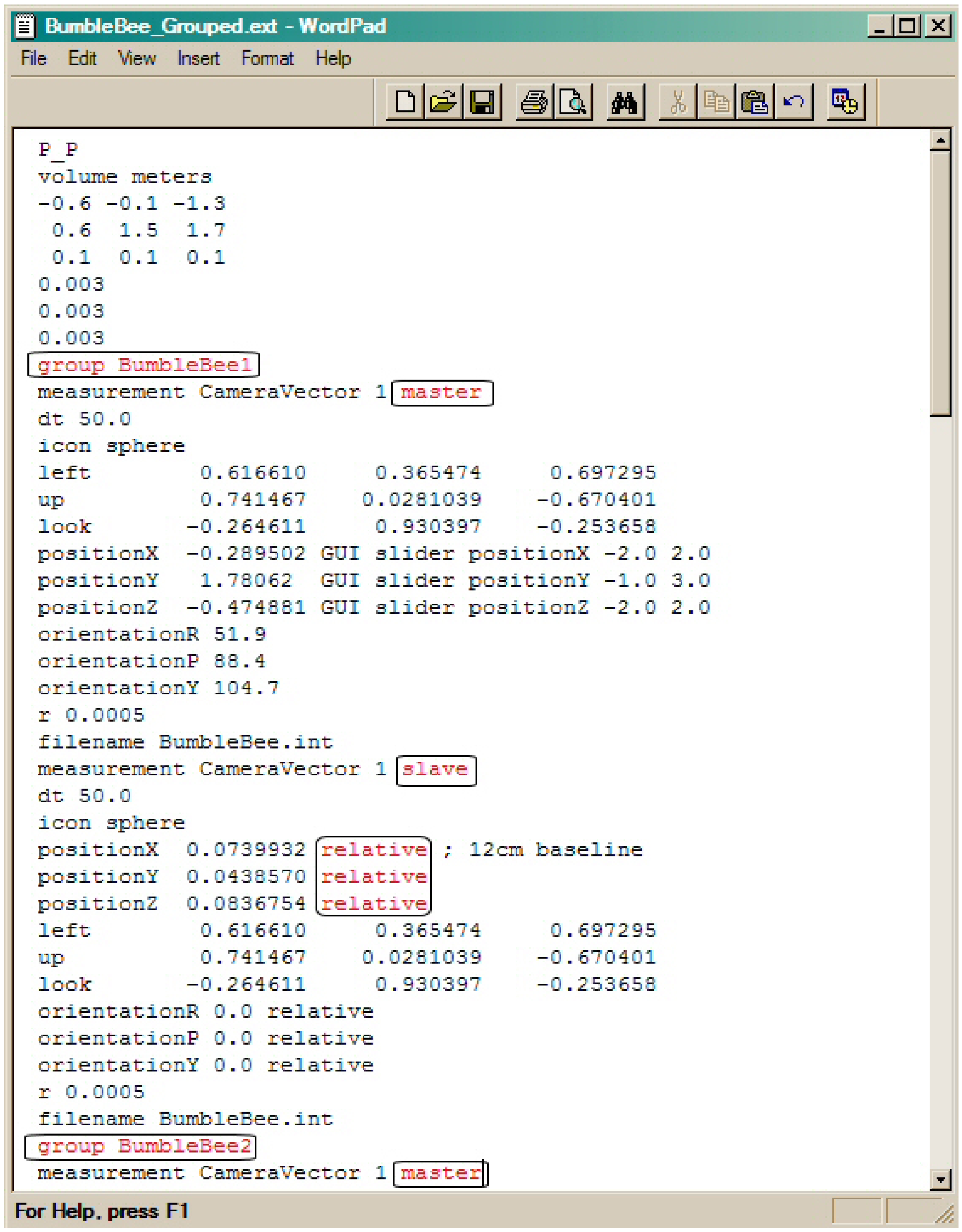
```
left      0.765728    -0.364919     -0.529617
up       -0.587386    -0.0613513    -0.806978
look      0.261988     0.929015     -0.261327
r 0.0005
filename BumbleBee.int
```

In this example, a GUI will be created that contains three sliders and one textbox listed in the order in which they are parsed. The sliders will be labeled "positionX", "positionY", and "positionZ" all ranging from -0.5 to 2.0 [units]. The three sliders will be followed by a single text box labeled "orientationR". An example Parameter Control GUI is shown in Figure 4.3.

4.2.5. Sensor Groups. Artemis supports sensor "grouping" so that one sensor can dictate specifications for other sensors associated with it (i.e., relative specification). Figure 4.7 shows an EXT file that specifies eight cameras (only three are visible in Figure 4.7) where the cameras are grouped into pairs. The first group is named "BumbleBee1" as shown in red and the first "CameraVector" measurement model listed after the group is declared as the "master" measurement model for the group. Its position, orientation, noise models, intrinsic parameters, etc. are specified as described in Section 4.2.2. However, the next CameraVector measurement model listed is a "slave" to the "master" CameraVector and its position in X, Y and Z is relative (as indicated in red in Figure 4.7) to the position of the master CameraVector. Specifically, the "master" in BumbleBee1 is located [-0.289502 1.78062 -0.474881] and the "slave" is located at [-0.289502 1.78062 -0.474881] + [0.0739932 0.0438570 0.0836754] = [x y z]. The second grouping begins with the "group BumbleBee2" statement and proceeds as described for BumbleBee1. Any changes to the master's parameters also change the slave's accordingly.

4.3. Performance Estimation

Once the system model and been defined and instantiated, pressing the "Compute data" button will initiate the P^∞ calculations. The point in the grid or on the surface that is being


BumbleBee_Grouped.ext - WordPad
File Edit View Insert Format Help
For Help, press F1

```
P_P
volume meters
-0.6 -0.1 -1.3
 0.6  1.5  1.7
 0.1  0.1  0.1
0.003
0.003
0.003
group BumbleBee1
measurement CameraVector 1 master
dt 50.0
icon sphere
left         0.616610    0.365474    0.697295
up           0.741467   0.0281039   -0.670401
look        -0.264611    0.930397   -0.253658
positionX  -0.289502 GUI slider positionX -2.0 2.0
positionY   1.78062  GUI slider positionY -1.0 3.0
positionZ  -0.474881 GUI slider positionZ -2.0 2.0
orientationR 51.9
orientationP 88.4
orientationY 104.7
r 0.0005
filename BumbleBee.int
measurement CameraVector 1 slave
dt 50.0
icon sphere
positionX  0.0739932 relative ; 12cm baseline
positionY  0.0438570 relative
positionZ  0.0836754 relative
left         0.616610    0.365474    0.697295
up           0.741467   0.0281039   -0.670401
look        -0.264611    0.930397   -0.253658
orientationR 0.0 relative
orientationP 0.0 relative
orientationY 0.0 relative
r 0.0005
filename BumbleBee.int
group BumbleBee2
measurement CameraVector 1 master
```

FIGURE 4.7. Artemis grouping specification in EXT input file

analyzed is displayed in the IDL command window so that the user can observe the progress of the calculation. When all points have been analyzed, Artemis will display “Complete” in the command window.

4.4. Visualization

Artemis offers known visualization techniques as supplied by IDL (isosurface, volume, image plane, and surface objects) for ten predefined metrics. The available metrics as dereived from P_∞ are:

- standard deviation of X position
- standard deviation of Y position
- standard deviation of Z position
- standard deviation of X velocity
- standard deviation of Y velocity
- standard deviation of Z velocity
- standard deviation of 3D position
- MLE
- EigLen
- EigMax
- Custom

The Maximum Likelihood Estimation (MLE) is a measure of the probability density function of the requested metric. The likelihood estimate (LE) of $\bar{x}$, (the point examined in the volume or surface) is defined as:

$$LE(\hat{x}) = \frac{1}{(2\pi)^{n/2}|P^{\infty}|^{1/2}} e^{\frac{1}{2}[(\hat{x}-\bar{x})^T \frac{1}{P^{\infty}}(\hat{x}-\bar{x})]} \tag{45}$$

where $\hat{x}$ is the state estimate and n is the number of elements in the state vector. For this calculation, we want the *maximum* likelihood estimate that occurs when $\hat{x} = \bar{x}$ and so the MLE formula is shown in Equation (46).

(46) $$MLE(\hat{x}) = \frac{1}{(2\pi)^{n/2}|P^{\infty}|^{1/2}}.$$

While the MLE does not have meaningful "real world" units such as meters or degrees, it is an effective measure of relative performance. A larger MLE is indicative of a smaller covariance and, thus, better performance.

"EigLen" is defined as

(47) $$EigLen = \sqrt[4]{\lambda \bullet \lambda_i}$$

given the set of eigenvalues of the P^{∞} matrix, $\lambda(P^{\infty})$

"EigMax" is the metric selected for all visualizations in Section 5 and is defined as the maximum eigenvalue of the P^{∞} matrix. More precisely, given the set of eigenvalues of the P^{∞} matrix, $\lambda(P^{\infty})$,

(48) $$EigMax = max\{|\lambda_i| : \lambda_i \in \lambda(P^{\infty})\}$$

Ji-guang Sun [**93**] uses this same definition (Artemis' EigMax) for "spectral radius", ρ, and notes that if $\rho < 1$ then the matrix in question is stable (see Section 3.7).

4.4.1. Isosurface. An isosurface is a set of surface or manifold drawn in the volume to represent a surface that has a specific constant value (the isovalue). Figure 4.8 shows examples of isosurfaces both with the volume visualization in place (left) and without the volume. Inside the acoustic volume is a pincushion-shaped (more formally hypocycloid-shaped) isosurface that shows where EigMax is less than -1.76 log meters and indicates a region of peak performance. The isosurface on the right is at -1.6 log meters and the volume information has been deleted.

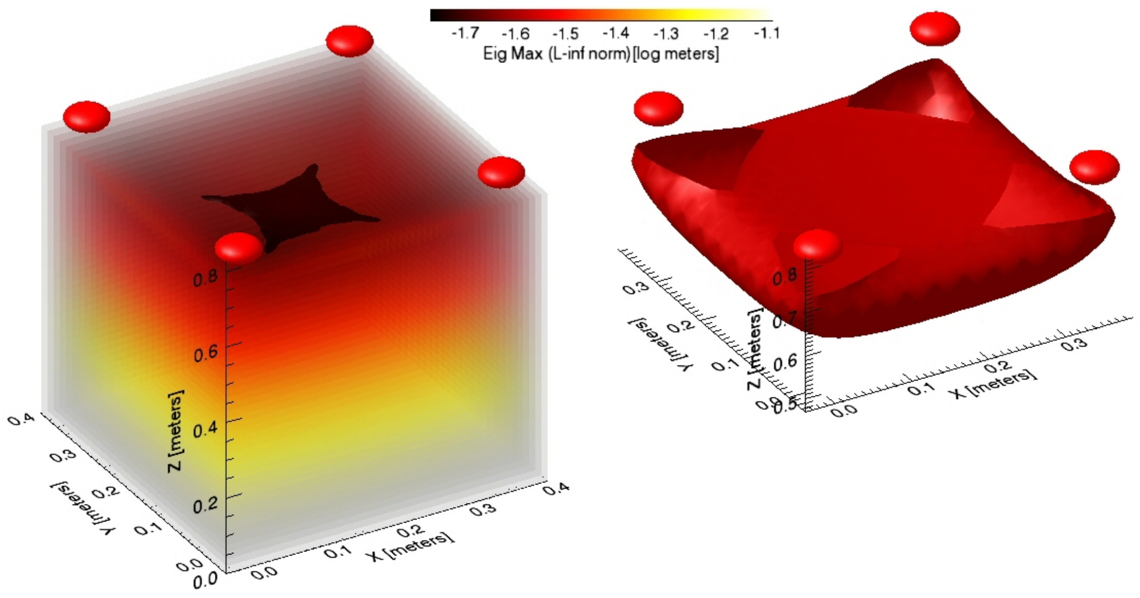


FIGURE 4.8. Acoustic Isosurface for -1.76 log meters (left) and -1.6 log meters (right)

4.4.2. Interval Volume. A volume interval is a set of tetrahedra that span a space between two isovalues within a volume. Figure 4.9 shows an volume interval visualization for our acoustic example in ranges -1.77 to -1.4 log meters and -1.4 to -1.08 log meters.

4.4.3. Image Plane. Figure 4.10 shows the acoustic example visualization using image planes or slices. The left and right figures use common planes with the pincushion-shape from Figure 4.8 (left) clearly visible at the $z = 0.6$ meter plane and the shape of the isosurface from Figure 4.8 (right) at $x = 0.2$ meters with the $z = 0.6$ meter plane removed. IDL does not currently support oblique slices.

4.4.4. Objects. Using a point list, Artemis can read in the description of an object and map the P^∞ metric to the object itself. Figure 4.11 shows a female torso in the Pandora environment [**89**] used for the initial camera setup (see Section 2.3). Figure 4.12 shows the Artemis P^∞ output (left) and the camera coverage (right) as computed by Artemis. Both images show rotated axes to match the Pandora origin and axis setup for easier comparison. The camera coverage from Artemis shows which areas of the torso can be seen by the

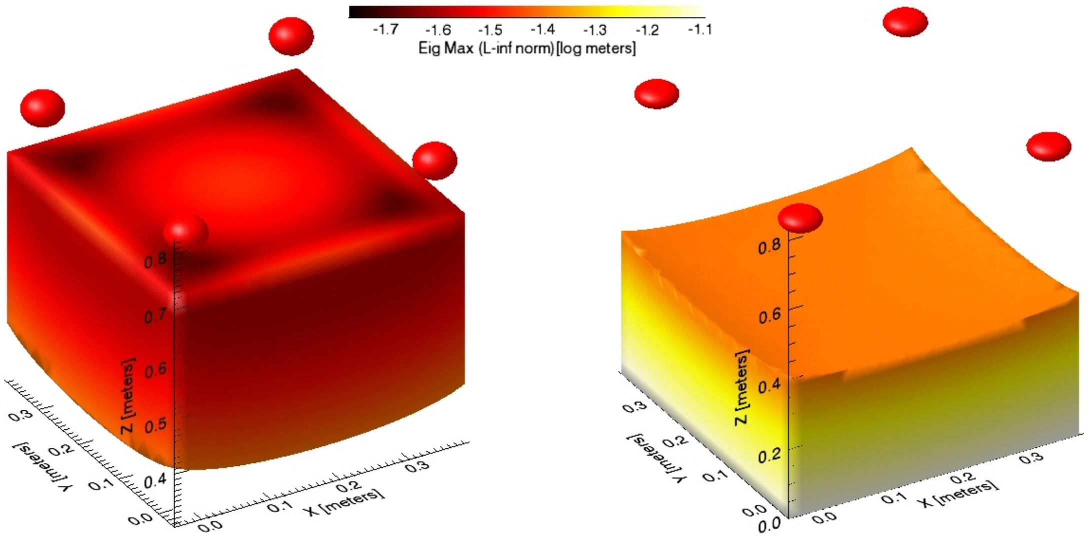


FIGURE 4.9. Acoustic Interval Volume for -1.77 to -1.4 log meters (left) and -1.4 to -1.08 log meters (right)

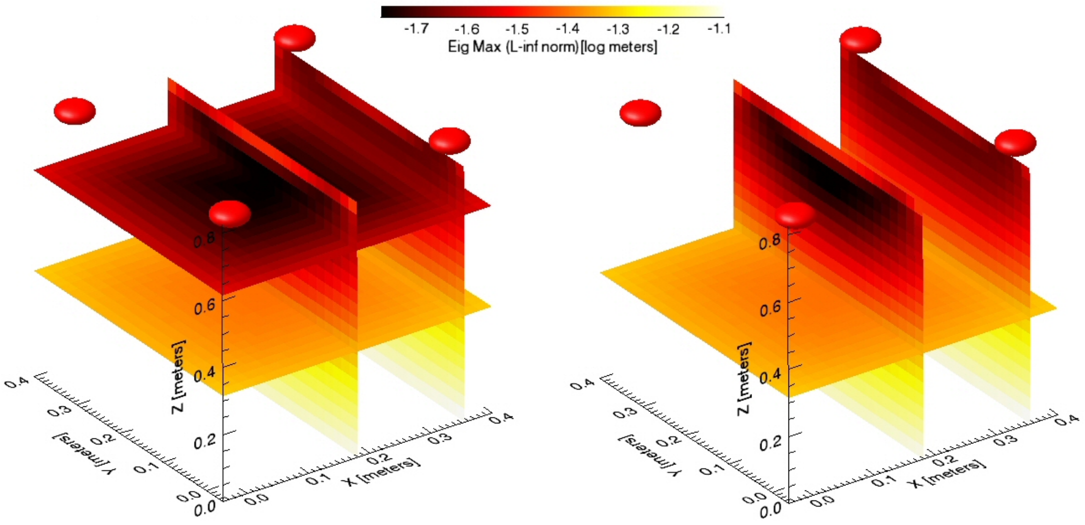


FIGURE 4.10. Acoustic Image Plane at z=0.6 and 0.4 and x = 0.35 meters

sensors defined in Artemis. Colors similar to those used in pandora have been selected to highlight the similarity in camera coverage. In order to perform P^{∞} analysis at a given sample point, at least two cameras must have line-of-sight visibility and so Artemis shows available P^{∞} data only where two or more camera frusta overlap (i.e., when the system is observable). This maps to the turquoise and yellow areas in Pandora and Artemis' camera coverage as shown by the left image in Figure 4.12. Observe that the best performance (darkest area) is approximately where all three camera frusta overlap.

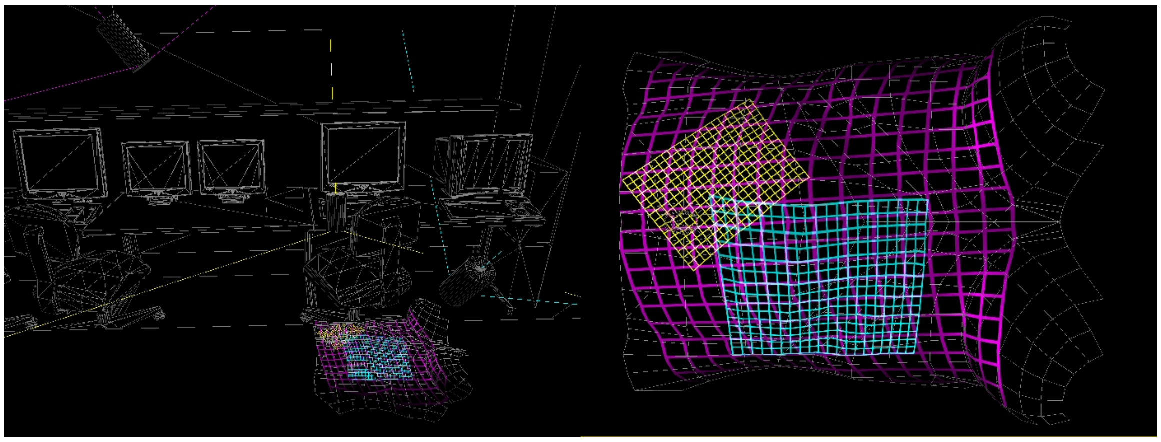


FIGURE 4.11. Patient Biopsy in Pandora.

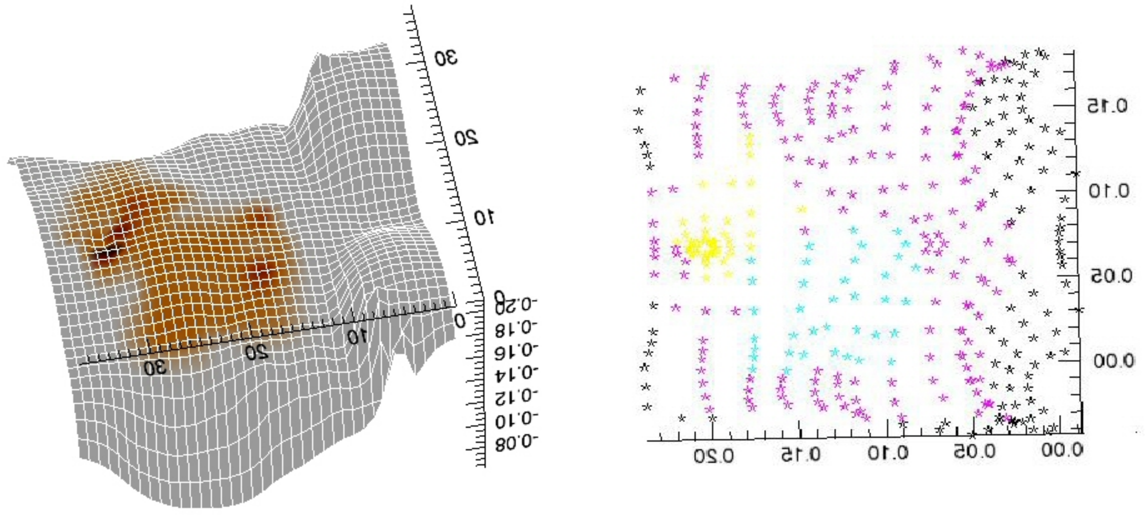


FIGURE 4.12. Patient Biopsy in Artemis with P^{∞} performance on the left and camera coverage shown on the right.

CHAPTER 5

Validation with Reality: Results

Validation is the process of determining the degree to which a computational model is an accurate representation of the real world. The highlighted nodes (outlined and in red) in Figure 5.1 indicates where this chapter falls in in the QV&V process.

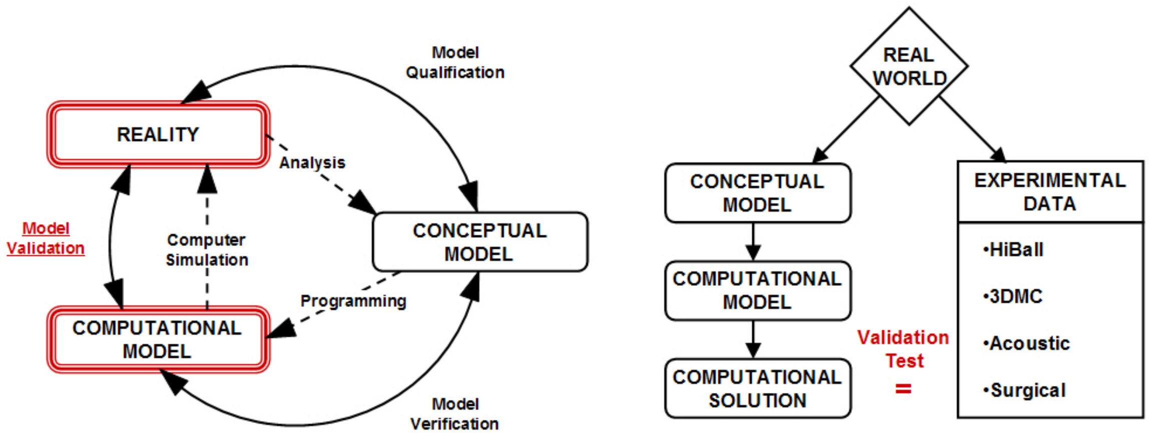


FIGURE 5.1. The Validation Process

Some of the results presented in the chapter were presented at VRST 2005 [**5**] and are the product of both Matlab^{TM} simulations and the working system prototype, Artemis. Artemis is described in detail in Section 4. The results have been separated into two sections that address (1) commercial and research systems for validation with real-world performance and (2) example systems for the demonstration of the flexibility of the P^{∞} framework.

5.1. Commercial and Research Tracking Systems

In this section both a commercially-available tracking system and a research system used for 3D scene reconstruction will be examined for validation. Readers should note that the proposed P^{∞} performance evaluation method was intended to provide a theoretical upper bound on performance and not exact estimates as required for validation. So, compromises had to made in order to perform validation and it may (incorrectly) appear that the P^{∞} method is algorithm-specific. For example, the decision to fuse ten measurements per point in the working volume was made for a fair comparison between measured and estimated performance. For this reason, the measured performance of the working HiBall system presented here should not be interpreted as the commercial system performance. Additionally, a true P^{∞} performance evaluation (as opposed to the validation presented here) would not be limited to ten measurements per point.

5.1.1. A Commercial System: 3rdTech HiBall$^{\text{TM}}$. The HiBall is an optically-based wide-area tracking system originally developed at UNC Chapel Hill and manufactured commercially by 3rdTech, Inc. It is composed of two key integrated components: the HiBall Optical Sensor and the HiBall Ceiling Beacon Arrays. Using what is sometimes called inside-out tracking, infrared LEDs embedded in the ceiling strips are imaged on one of six Lateral Effect Photo Diodes (LEPDs) contained in the HiBall sensor using a single-constraint-at-a-time (SCAAT) algorithm [**101**] for data fusion.

5.1.1.1. *Evaluation.* The approach to evaluating estimated performance was twofold. First, the steady-state predictions were compared to the measurement error reported in [**103**] using a HiBall-3000 optical tracking system. Second, a series of controlled experiments were performed (again using the HiBall-3000), in which the measured performance over a wide area was estimated and compared the results to the steady-state predictions.

Figure 5.2 shows the experimental setup in which the HiBall sensor was pushed along a series of paths mechanically constrained along a rigid rail. 3rdTech engineers graciously

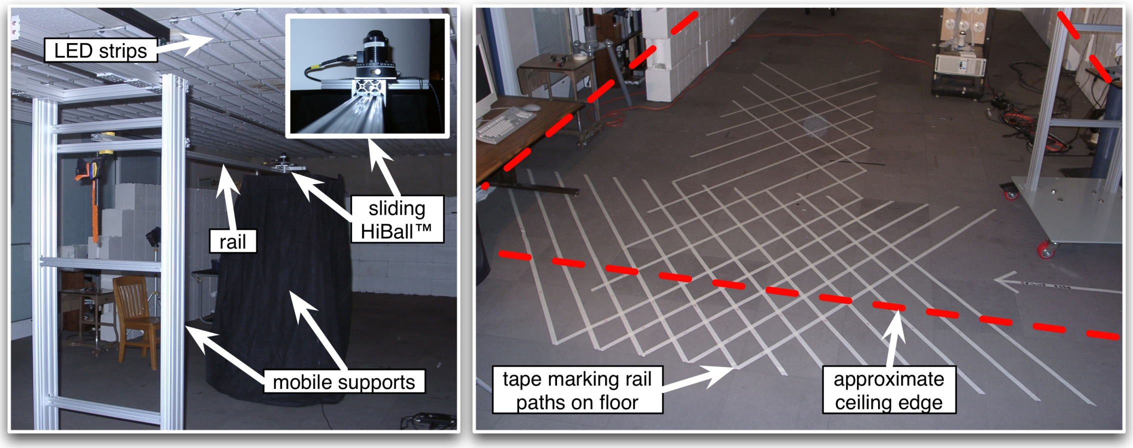


FIGURE 5.2. Experimental setup. Left: shows the LED strips on ceiling, the precision 80/20^{TM} rail, the mobile rail supports, and the sliding HiBall sensor fixture (see also the inset). Right: the tape on the floor marks the 28 different linear rail paths that collectively form a grid intended to span the edge of the ceiling (dashed red line) and some portion of the interior.

provided low-level access to the system software, so HiBall data could be logged at a very high rate. This data was compared to steady-state estimates. In particular, curves fitted to the "rail path" data were generated and the data were examined for deviations from those curves.

The experimental rail rig was approximately seven feet (1.4 m) long, with a constant height of approximately 1.9 meters. In order to perform a wide-area comparison, the rig was positioned along 28 paths crisscrossing the area, forming a grid that ranged from 3.0 to 6.0 meters in the x-direction and 5.5 to 10.0 meters in y. The grid was intentionally arranged to extend beyond one ceiling edge by about one meter, allowing for a region of potentially deteriorating performance where the HiBall is starved for measurements.

5.1.1.2. *The HiBall^{TM} System.* The HiBall-3000 system estimates the sensor (target) pose by sighting a two-dimensional array of ceiling-mounted light emitting diodes (LEDs). For the steady-state estimates the HiBall was modeled using the same process model the system uses, and a single 1000 Hz measurement model. In the measurement model, a grid of beacons identical to an actual installation was specified (spaced 10 cm apart across a 6.3 m by 9.0 m ceiling). Just as the actual system continually chooses a set of nearby LEDs for tracking, the measurement model chooses a subset of visible LEDs for performance analysis. Further, both measurement models incorporate occlusion testing in order to choose visible LEDs.

In 1999, the developers of the HiBall tracking system reported estimation errors of 0.2 mm "for nearly all of the working volume" and 0.5 mm at a height of approximately 1 meter [**103**]. This setup was modeled for comparison and Figure 5.3 shows the resulting surface plots of P^{∞} analysis at 1.9 meters (head height) and at 1.0 m (waist height) across an area at the corner of the current HiBall ceiling. At 1.9 m, the P^{∞} estimates are around 0.20 to 0.22 mm (-3.6 to -3.7 log meters). Note the peaks and valleys in performance corresponding to the location of the LED strips in the ceiling. At first glance, it would seem that the P^{∞} estimates at 1.0 m predict better than the reported 0.5 mm (-3.0 log meters) with the more ceiling-central estimates at or around 0.3 mm (-3.5 log meters). However, upon reexamination of [**103**], it became clear that the test setup had been positioned at the ceiling *edge* and not in the better-performing center area. Performance estimates in the far corner of the surface show values of 0.45 to 0.50 mm (-3.30 to -3.35 log meters). Figure 5.4 shows the HiBall performance estimates over a small working volume (from 1.0 m to 2.0 m) to highlight the performance of both the full and sparse ceiling configurations (see Section 5.1.1.3 for a description of the sparse ceiling configuration). The ceiling strip pattern is clearly visible in both configurations..

Figures 5.7 and 5.8 show some numerical results from the HiBall rail experiments depicted in Figure 5.2. The plot on the top in Figure 5.7 shows the "jitter" of the real system's individual position estimates (deviation from a curve fit to the data) and the plot on the bottom shows the estimated steady-state covariance (P^{∞}). The initial analysis of the real system's deviation from a *line fit* showed a disappointing order of magnitude disparity between the measured data and the predicted estimates. This was due to three unmodeled noise sources in the test setup. First, the rail was sagging several millimeters as a result

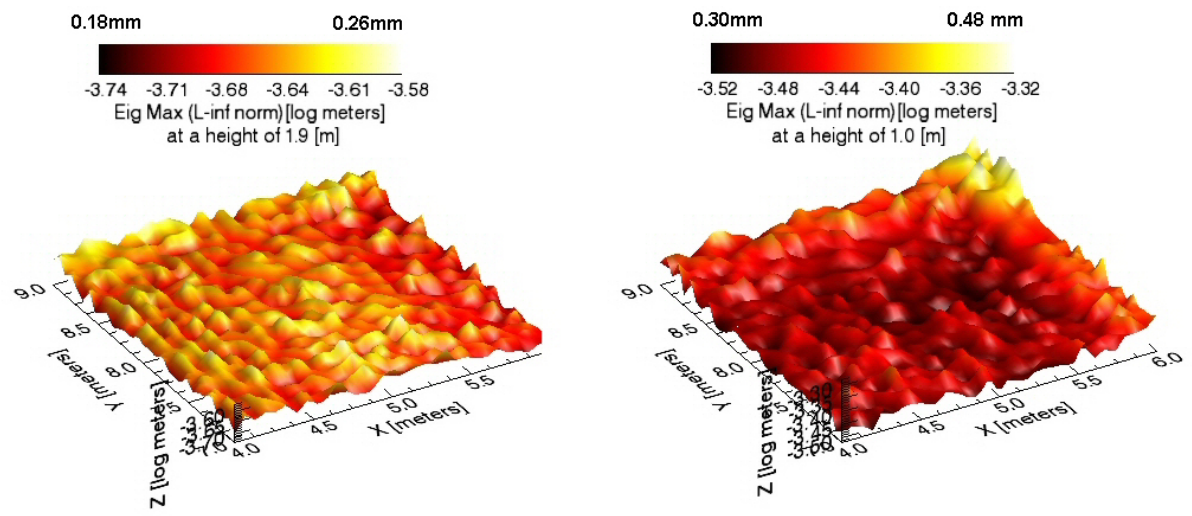


FIGURE 5.3. P^{∞} estimates at Left: 1.9 meter "head" height, and Right: 1.0 meter "waist" height

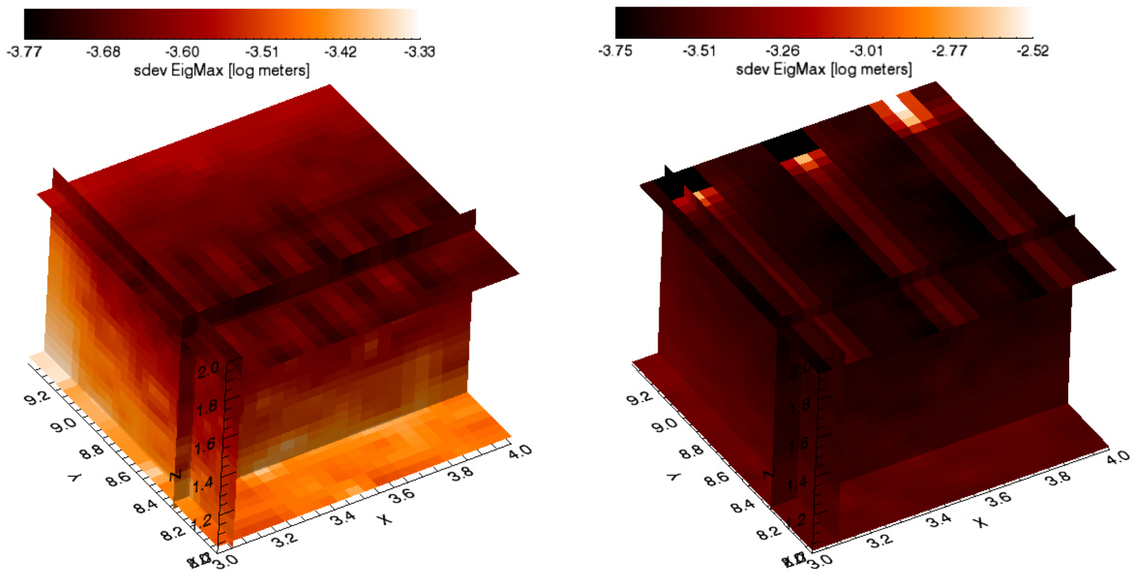
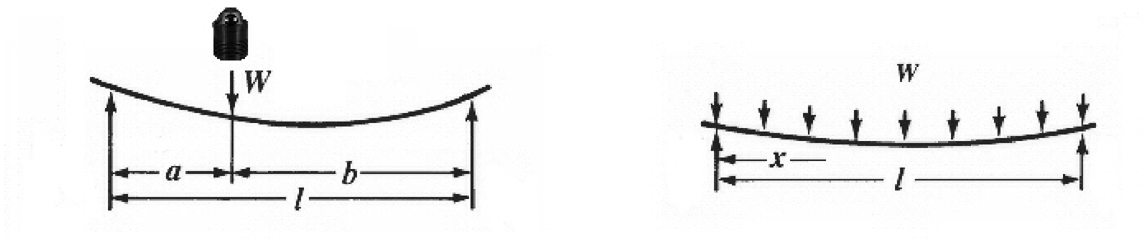


FIGURE 5.4. P^{∞} volume analysis with image planes for full (left) and sparse (right) ceilings

of its own weight and the load of the HiBall. Second, the test execution added mechanical noise (friction, gait, etc.) as the test conductors moved the HiBall sensor along the rails by hand.

To address the first problem the deviation of the real data from a *deflection curve* was computed as described in [**86**] and [**41**]. The total deflection is the sum of the deflection due to the point load weight of the rail itself (distributed) plus the weight of the HiBall sensor and slide apparatus (point) at a distance *x* from the left of the rail.

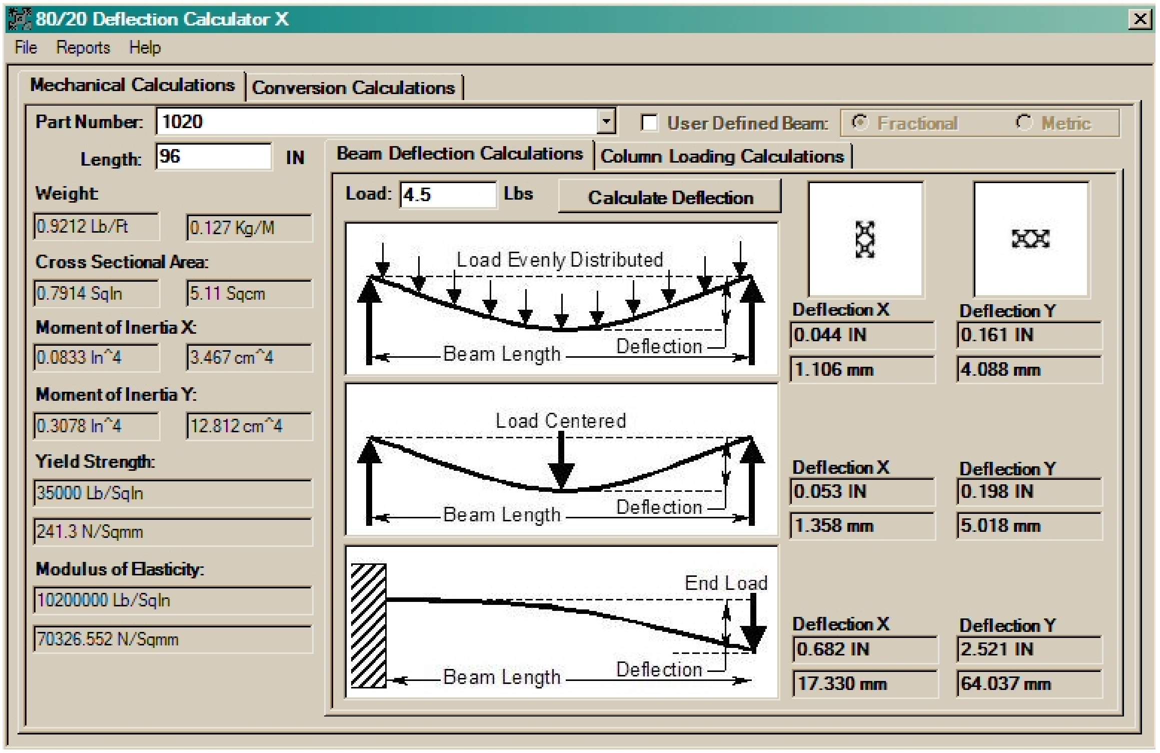


$$Deflection_{point} = \frac{Wbx}{6LEI}(l^2 - b^2 - x^2),\ 0 \leq x \leq a \qquad Deflection_{dist} = \frac{wx}{24EI}(l^3 - 2lx^2 + x^3)$$

FIGURE 5.5. Rail Deflection Formulas for (right) distributed load and (left) a point load

Using the specifications of the 1020-rail from 80/20, Inc., a maximum deflection due to the HiBall sensor of 0.198 in (5 mm) was calculated with behavior along the horizontal rail as pictured in in the upper two diagrams of 08/20 Inc.'s Deflection Calculator Figure 5.6 and specified under "Deflection Y".

To address the second problem, the expected measurement noise was increased slightly. This was necessary because the experiment's exact measurement noise was unknown and noise was being injected into the system through the experiment setup itself. Because we had no method of measuring the noise due to sliding the HiBall sensor along the rail, the additional measurement noise was capped at the value corresponding to the maximum rail deflection as the HiBall sensor and experimenter moved along the rail. The final calibrated results are shown in Figure 5.7. The upper image shows the measured system performance. The surface is interpolated over the stddev values and the blue circles indicate measured values. The semi-transparent vertical planes in both plots marks the edge of the "ceiling" (the LED array) around 9 meters in *y*.


80/20 Deflection Calculator X

File Reports Help

Mechanical Calculations | Conversion Calculations

Part Number: 1020

User Defined Beam: Fractional Metric

Length: 96 IN

Weight: 0.9212 Lb/Ft 0.127 Kg/M

Cross Sectional Area: 0.7914 SqIn 5.11 Sqcm

Moment of Inertia X: 0.0833 In^4 3.467 cm^4

Moment of Inertia Y: 0.3078 In^4 12.812 cm^4

Yield Strength: 35000 Lb/SqIn 241.3 N/Sqmm

Modulus of Elasticity: 10200000 Lb/SqIn 70326.552 N/Sqmm

Beam Deflection Calculations | Column Loading Calculations

Load: 4.5 Lbs

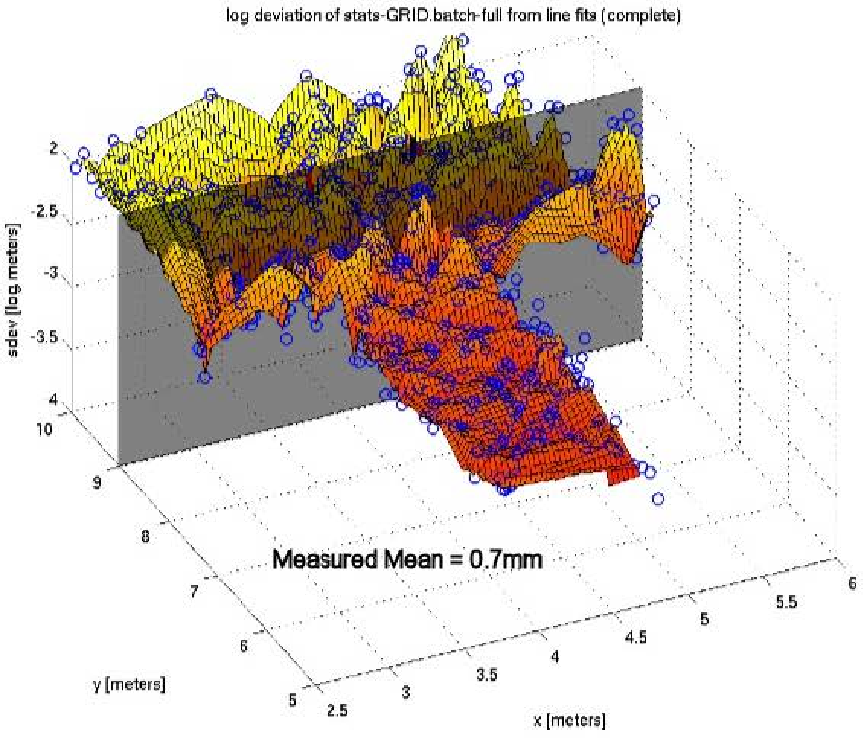
Calculate Deflection

	Deflection X	Deflection Y
Load Evenly Distributed	0.044 IN	0.161 IN
	1.106 mm	4.088 mm
Load Centered	0.053 IN	0.198 IN
	1.358 mm	5.018 mm
End Load	0.682 IN	2.521 IN
	17.330 mm	64.037 mm

FIGURE 5.6. Rail Deflection Behavior [1]

Both measured and estimated system performance are qualitatively and quantitatively consistent under the ceiling. We still observe a greater fluctuation of performance in the measured data (again probably due to unmodeled noise sources) but the measured mean is at the same order of magnitude as the calculated P^{∞} mean. The measured mean of 0.7 mm is only 1.8x that of the calculated mean of 0.4 mm which indicates that the calculated P^{∞} is an upper bound on performance. The performance degrades similarly in both the measured and estimated data as the HiBall sensor moves out from under the ceiling LEDs. At most points where the real system lost tracking, the performance method indicated very high or infinite P^{∞}.

5.1.1.3. *A Sparse HiBall^{TM} System.* To further validate the performance estimation method, the HiBall system was modified so that every other row of LEDs in the ceiling was disabled, effectively doubling the distance between rows. This was done in both the installed system,



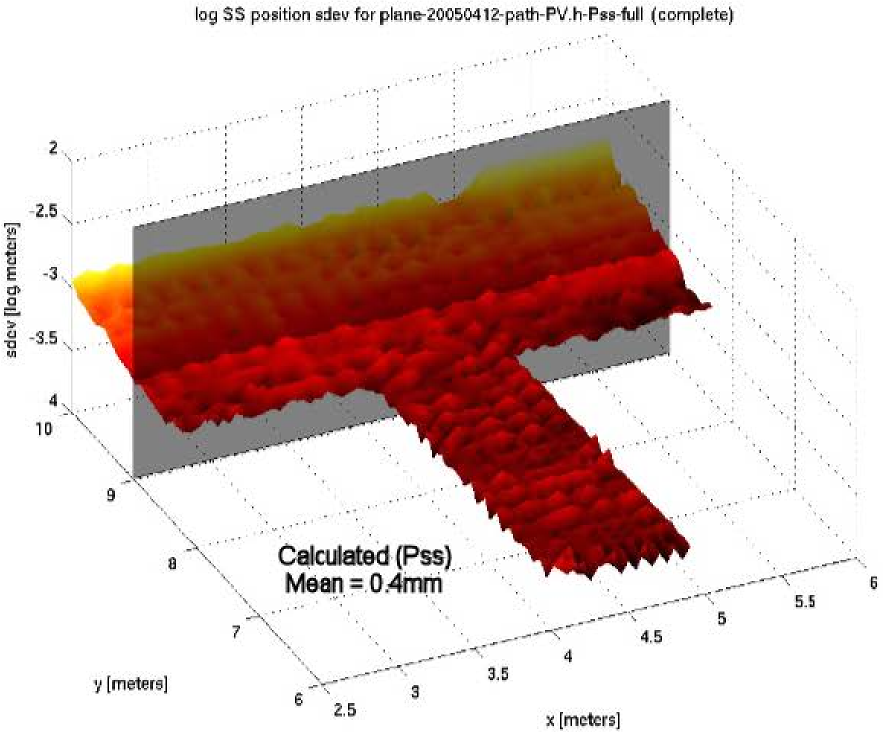
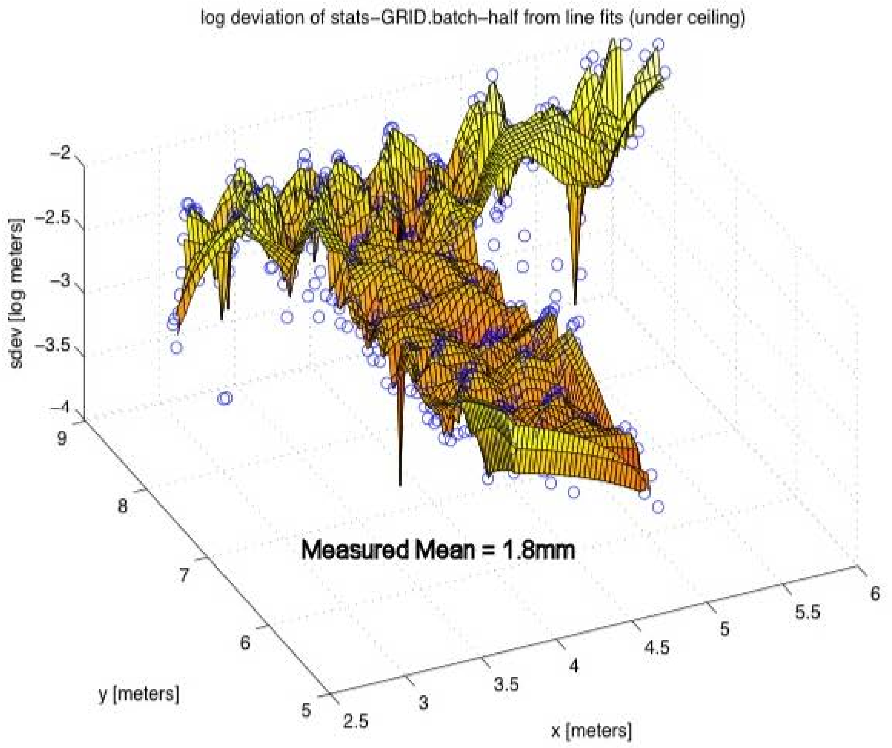


FIGURE 5.7. Comparison of measured HiBall position stddev with P^{∞} estimates. Top: position stddev (from curve fit) of the real system. Bottom: P_{∞} estimates over a plane fit to the rail data.

and in the steady-state models. Here again (Figure 5.8) the performance estimates are similar to the measured system performance (Note that the data off the ceiling was excluded from both plots to increase the dynamic range). While the overall “sparse” system performance remains very good with the standard deviation range increasing only slightly, the performance is less consistent with clear peaks and valleys of accuracy that map directly to the alternating enabled/disabled rows of LEDs. Again we observe greater fluctuation of performance in the measured data with a measured mean of 1.8 mm and a calculated P^{∞} mean of 1.0 mm, a 1.8x increase as in the full-ceiling configuration.

5.1.2. A Research System: 3DMC. Optical systems used for 3D scene reconstruction employ cameras at fixed locations in the environment, looking inward toward the capture area. Figure 5.10 shows the 3D Medical Consultation’s (3DMC) Portable Camera Unit (PCU) used at UNC for development and test of 3D telepresence technologies in remote medical consultations [**105**]. In its analyzed and pictured configuration, it consists of two rows of Point Grey Dragonfly$^{\mathrm{TM}}$ cameras vertically separated by approximately 11 cm and horizontally separated by approximately 15 cm. The Bouguet stereo-calibration algorithm [**13, 14**] was used for camera calibration and the cameras are positioned at heights (z-axis) of 88 cm and 99 cm at x-values of -40, -25, -10, and 6 cm and y-values of 27 and 22 cm (respectively with z-values).

To measure the performance of the 3DMC setup, 1000 images of a pattern of gaussian blobs drawn on a planar piece of foam core were taken over eleven poses of the foam core at distances from 0 cm to 30 cm at 3 cm intervals. The test setup and camera views at the closest distance are shown in Figure 5.10. 3D triangulation was performed for each gaussian blob at every distance and, at each distance, a covariance matrix was generated. These covariance matrices are the same form generated by the P^{∞} analysis and so permit direct comparison between measured and estimated performance.



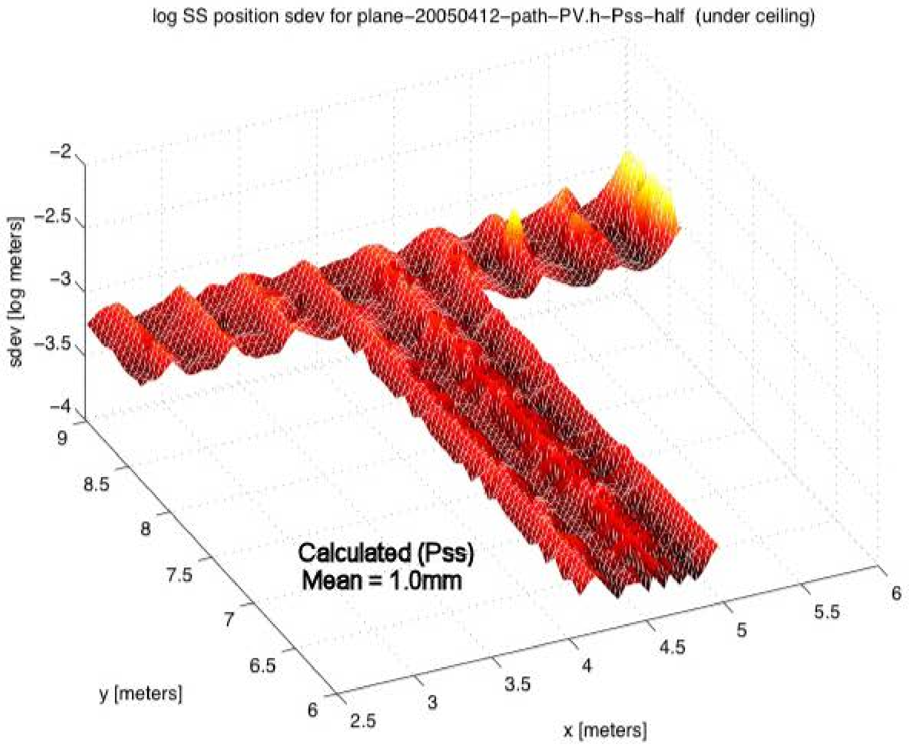


FIGURE 5.8. Comparison of measured HiBall position stddev (top) with P^{∞} estimates (bottom) when alternating rows of LEDs are disabled. See also caption for Figure 5.7.

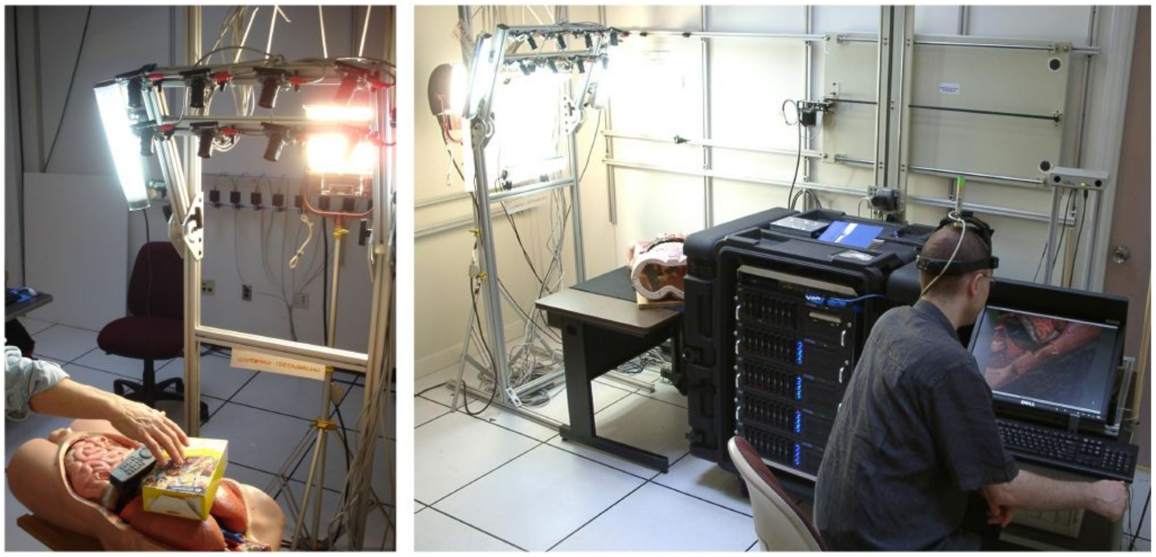


FIGURE 5.9. The 3DMC equipment setup.

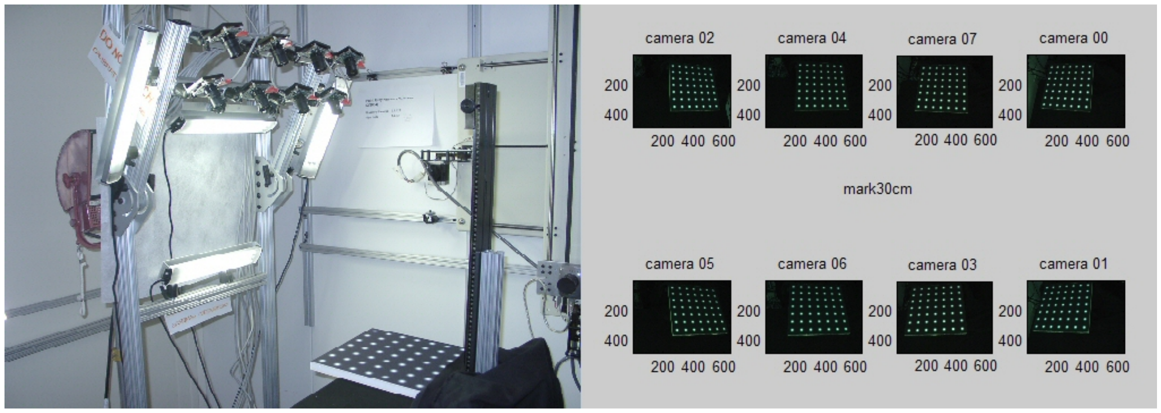


FIGURE 5.10. 3DMC test setup and Camera Views.

To perform the P^{∞} analysis, a noise model for the Dragonfly camera was needed. A 1D gaussian line was photographed over 13 distances (10:5:70 cm) with a single Dragonfly camera positioned directly overhead. At each distance, 1000 images were processed for the gaussian mean and standard deviation of the mean and averaged resulting in the left curve shown in Figure 5.11.

Bundled in with the standard deviation of the blob mean is the size of the imaged blob itself (i.e. there are fewer pixels contained in the blob as the distance away from the camera increases). Dividing the standard deviations at each distance by the standard deviation of the blob itself (in pixels) results in a linear relationship $(0.0004d + 0.002)$ as shown on the right in Figure 5.11.

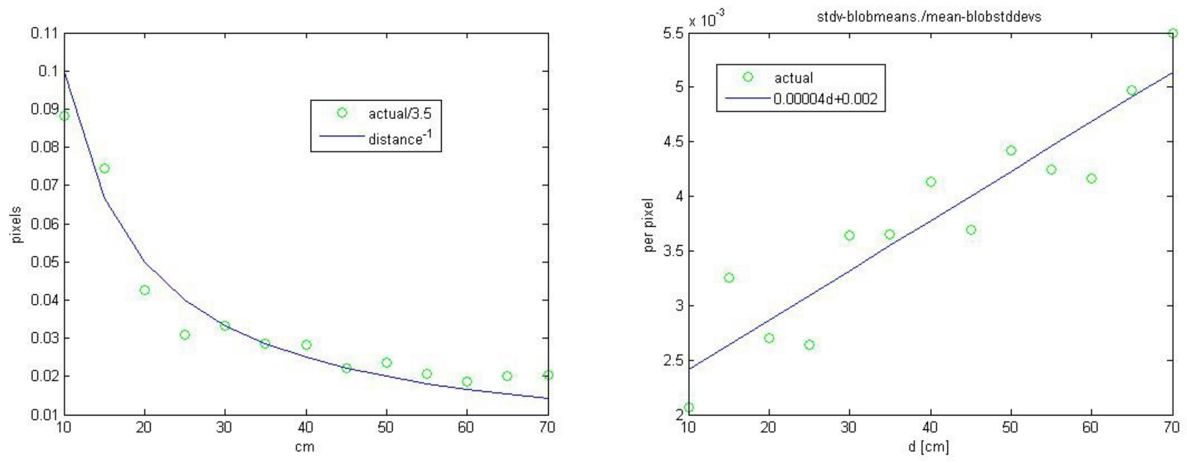


FIGURE 5.11. Standard Deviation of 1D Gaussian Blobs.

This linear relationship is unitless because pixels have been divided by pixels. However, by multiplying by the size of the Dragonfly CCD (640x480 pixels) this can be used as the measurement noise model variance (R) for the 3DMC analysis. Compensating for the off-axis angle, θ, the measurement noise is:

$$CCDnoise = (0.0000046 * distance) + 0.00195 \tag{49}$$

$$R(u) = ((CCDnoise * 640.0)/(cosine(\theta))^2 \tag{50}$$

$$R(v) = ((CCDnoise * 480.0)/(cosine(\theta))^2. \tag{51}$$

Using a P motion model with the q-values set as low as mathematically possible and sampling at double the measured results, the Eigmax metric from the P^{∞} analysis is shown in Figure 5.12 on the bottom and the EigMax metric resulting from the 3DMC PCU test is shown on the top.

In both the measured and estimated P^{∞} performance measurements, the best performance is at the closest height to the cameras in the center of the camera setup at [185 -90 380] mm progressing to the worst performance at the outer corners of the lowest height of approximately 85 mm. The measured EigMax ranges from 0.022 to 0.039 mm while the estimated P^{∞} performance ranges from 0.022 to 0.035 mm. The scale for both figures is set to the largest possible range (0.22 to 0.39 mm) for direct comparison. The difference between the measured and estimated performance ranges from $4.13e^{-6}$ mm to 0.008 mm with a mean of 0.0019 mm. This is shown in Figure 5.13.

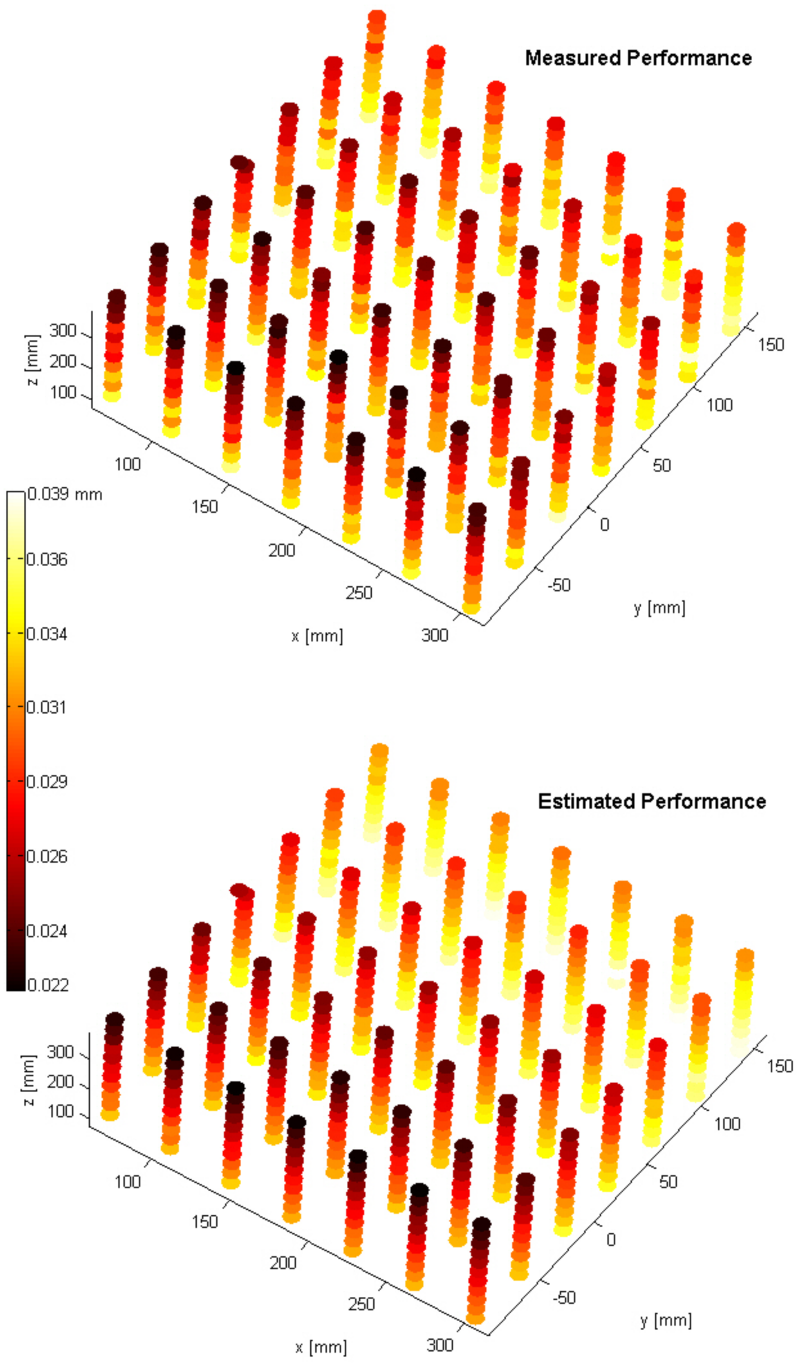


FIGURE 5.12. Standard Deviation of 1D Gaussian Blobs as measured (top) and as estimated (bottom).

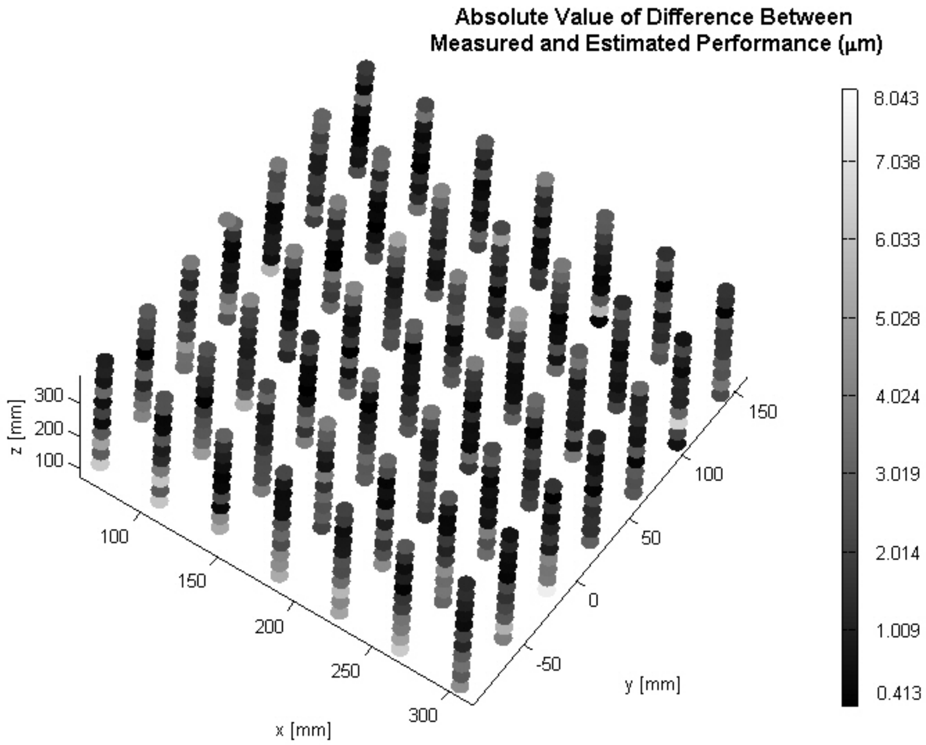


FIGURE 5.13. Absolute difference in measured and estimated performance.

5.2. Example Tracking Systems

5.2.1. An Acoustic Example. Figure 1.4 shows the results from the performance estimation for the hypothetical acoustic system presented earlier. The red spheres mark the positions of the four transmitters (speakers). The system is modeled with a constant update rate of 50 Hz ($\delta t = 0.02$ seconds). The Q matrix is a 6×6 matrix of the form (23)–(25) with $qx = qy = 0.395$ and $qz = 0.107$. The measurement covariance matrix is a 4×4 diagonal matrix with $R[i,i] = 0.005\bar{z}[i]^2$ for $1 \leq i \leq 4$, where $\bar{z}$ is the range given in Section 3.4.2. For simplicity, a measurement model covariance formula that is a quadratic function of distance was selected. This is not to say that this model is the best noise model available. The measurement model can be modified to represent any function desired.

5.2.2. A Multi-Camera Acquisition System. An existing eight-camera acquisition rig used for 3D computer vision-based scene reconstruction research as shown in Figure 5.14 was modeled for analysis to evaluate system performance in different camera configurations.

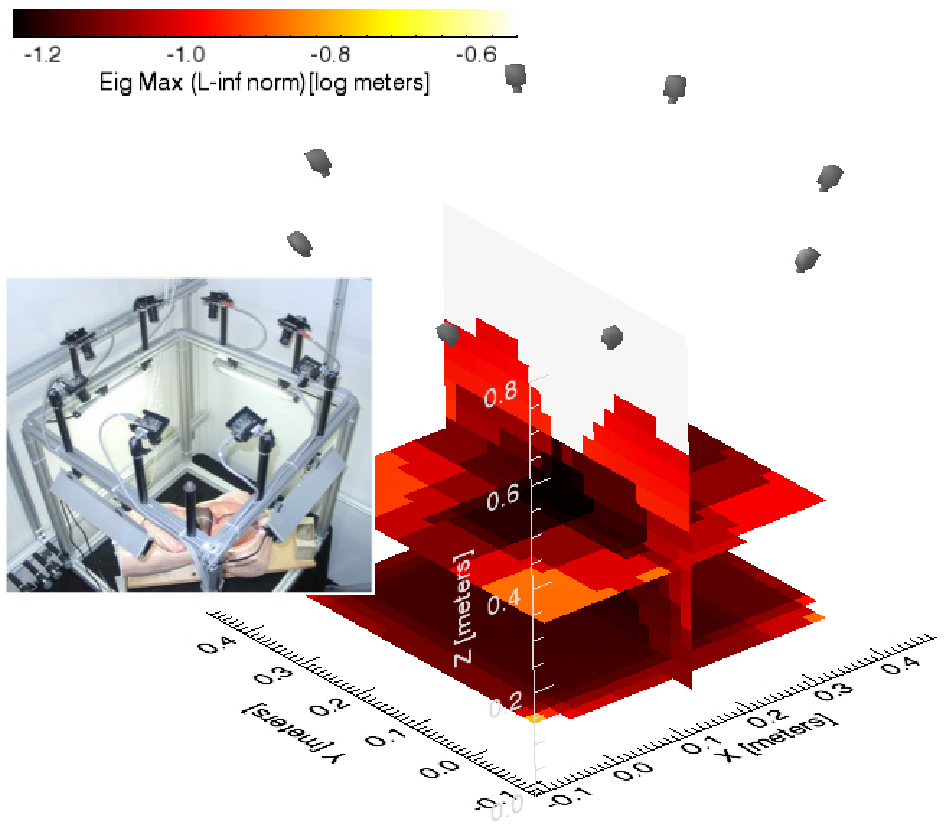


FIGURE 5.14. The P^{∞} analysis of an 8-camera vision-based acquisition rig. The grey camera icons indicate accurate positions and rotations of the cameras.

The process model used was the same as in the preceding acoustic example, with an update rate of 30 Hz. For the measurement function a simple pinhole camera model based on the Point Grey Dragonfly camera specifications (6 mm focal length, 640x480 resolution) was used, where the image coordinates are given by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \bar{x}'[1]/\bar{x}'[3] \\ \bar{x}'[2]/\bar{x}'[3] \end{bmatrix}$$

where $\bar{x}' = R(\bar{x}_i - \bar{c})$, R is a camera rotation matrix, $\bar{x}_i$ is the 3D point where P^∞ is computed, and $\bar{c}$ is the camera position vector. Because the measurement function is non-linear, the linear approximation given by the measurement Jacobian was used,

$$H = \begin{bmatrix} \frac{\partial u}{\partial \bar{x}_i[1]} & \frac{\partial u}{\partial \bar{x}_i[2]} & \frac{\partial u}{\partial \bar{x}_i[3]} \\ \frac{\partial v}{\partial \bar{x}_i[1]} & \frac{\partial v}{\partial \bar{x}_i[2]} & \frac{\partial v}{\partial \bar{x}_i[3]} \end{bmatrix}$$

A function similar to that of the preceding acoustic example was used for R, where the measurement error covariance increases with the square of the distance.

In Figure 5.14 one can see that the view frusta of the individual cameras are recognizable, as the performance increases in the overlapping regions. In this configuration, the best performance is located at the horizontal center at a height of 0.24 meters. The worst performance below the first frusta overlap (0.68 m) occurs at the bottom corners of the volume.

5.2.3. A Hybrid System. To illustrate generality, the acoustic model of Section 5.2.1 can be combined with the multi-camera system model of Section 5.2.2. In the pseudo-code given at the end of Section 3.6, the camera device list and parametric measurement model are simply added to the loop, along with the acoustic devices and model. The results are shown in Figure 5.15 where one can see how overall P^∞ decreases (performance improves) when compared to either the acoustic or multi-camera systems alone. Further, the system can now "see" at heights above 0.68 m due to the inclusion of the acoustic sensors. In fact, the region of peak performance has moved from the horizontal center (camera) and the points closest to the speakers (Figure 1.4) to the top four corners of the working volume (height of 0.68 m).

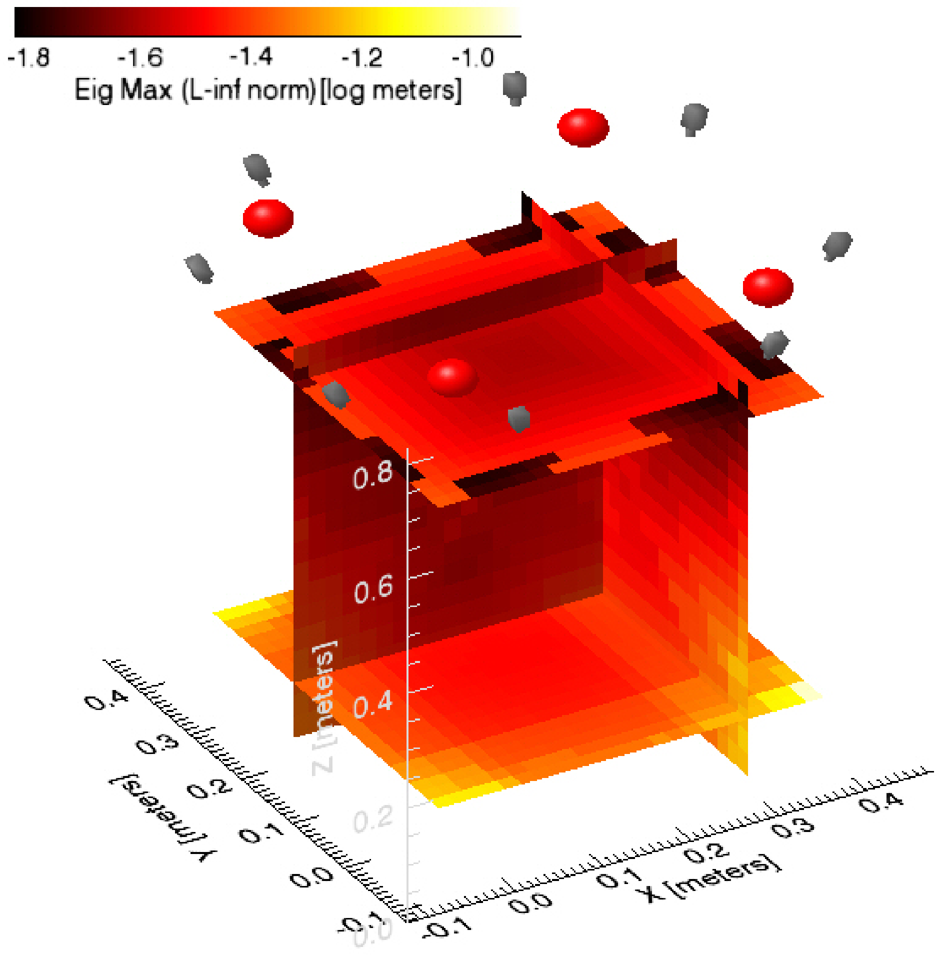


FIGURE 5.15. A P^{∞} visualization for a hypothetical hybrid system with acoustic devices and cameras. Acoustic transmitters and cameras are depicted with small red and gray icons.

5.2.4. A Motion Capture System. Optical systems used for human motion capture typically follow a common paradigm. Cameras are position at fixed locations in the environment, looking inward toward the capture area, so that they can observe active or passive (reflective) targets affixed to the moving human(s). As such the modeling for a P^{∞} analysis is very similar to the preceding multi-camera acquisition system.

Figure 5.16 depicts a hypothetical eight-camera motion capture system, and a corresponding P^{∞} visualization. While the absolute P^{∞} values may not be accurate in this analysis (and so are not included) the relative results from the volume visualization show us that,

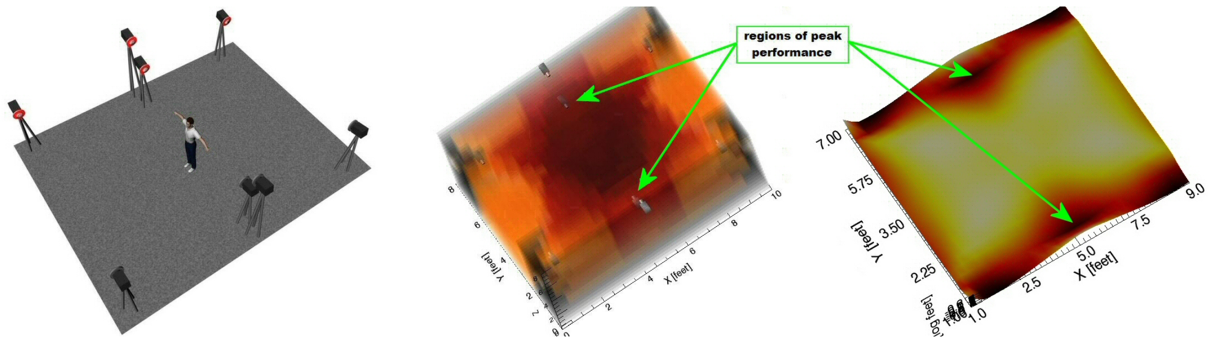


FIGURE 5.16. A hypothetical eight-camera motion capture system. Left: picture depicting the arrangement of the eight cameras in the room. Center: volume visualization of P^{∞} throughout the space. Right: Surface plot at plane of peak performance (z=5.6 ft).

as expected, there are dead spots in the corners and better performance in a hexagonally-shaped pillar running through the center of the volume. Numerical analysis showed that peak performance occurs at a height of 5.6 ft, between the camera pairs at heights of 5 ft and 7 ft. A surface visualization at this height clearly shows that the areas of estimated peak performance are close to the camera pairs as indicated in Figure 5.9, and *not* in the center of the capture area.

CHAPTER 6

Verification: Analysis

Statistician George E. P. Box (creator of the Box-Jenkins ARMA model [**16**]) said “All models are wrong but some are useful” [**15**]. No matter how extensively processes are modeled, they are approximations and our computational models can and will fall short of the true behavior of any system [**32**]. Verification attempts to determine the error and accuracy between the conceptual and computational models. It must address error originating from various sources extending from sensor device noise to algorithmic approximation. The highlighted region (outlines and in red) in Figure 6.1 indicates where this chapter falls in in the QV&V process.

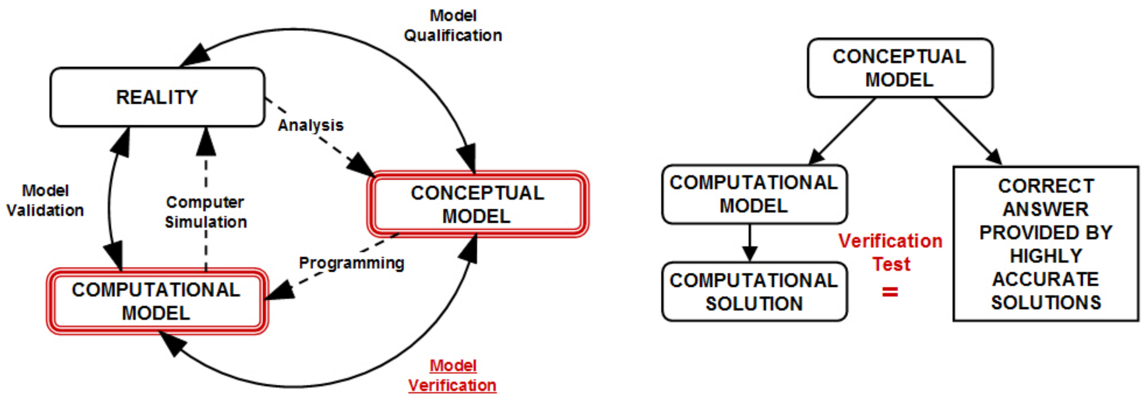


FIGURE 6.1. The Verification Process

Verification can be subdivided into *Code Verification* and *Solution Verification*. The goal of code verification is to find and remove mistakes in the source code and improve software quality in general. While this quality assurance step is important, my focus is on solution verification. This addresses the accuracy of input data (i.e. the model) and numerical algorithms and estimates the solution error to provide some measure of accuracy of output data for the system of interest.

We know that the fidelity of simulation results are dependent on the fidelity of the model itself and the input (i.e. how closely the model resembles the actual system). Errors in the model(s) will result in less-then-optimal performance estimates. Further, some error due to non-linearity (algorithmic approximation) is inherent when linearizing a non-linear problem either by spatial or temporal discretization as with the eKF. Because the eKF is an accepted and proven way of dealing with non-linear systems (see Section 6.1.1), I am not attempting to prove this approach is valid here (i.e. model qualification). However, I do want to address potential sources of model input error for the P^{∞} method. I am calling this "System-Level Analysis".

The parameters required for P^{∞} computation can used to gain an understating of the performance trade-off between sampling time or update rate (δt) and noise. While the systems presented in Section 5 use a scalar value for the measurement noise, R, this value is typically the result of decisions made at the measurement level and examination of those decisions within the context of the P^{∞} parameters can be insightful. This is "Measurement-Level Analysis".

6.1. System-Level Information Optimization

6.1.1. Non-linearity. Both the MacFarlane–Potter–Fath "Eigenstructure Method" (the close-form solution to the DARE) presented in Section 3.5.1 and the extended Kalman filter (the *non-linear* iterative solution to the DARE) presented in Section 3.7.1 accommodate the case of a non-linear process model and/or a non-linear relationship between the process and measurement models. This is accomplished by linearizing the non-linear system around a mean and covariance over some δt. This linearization is prone to error and solving for the linearizations can lead to instability (divergence) and error [**49**].

This suboptimal behavior is especially apparent within the highly non-linear domain of orbital analysis and tracking (the application for which the Kalman Filter was designed) and the Unscented Kalman Filter (UKF) was created to address the issue of linearization in this context [**49, 47, 48, 50, 51**]. Rather than propagating the state mean through a first-order linearization as with the eKF, the UKF uses a deterministic sampling technique. By propagating a set of sample points around the state mean through the non-linear functions themselves, the mean and covariance are captured accurately to the 3rd order for any nonlinearity.

However, LaViola [**58**] found that, within the context of human head and hand tracking in virtual environments, the eKF and the UKF exhibited approximately the same error and that, considering running time, the eKF is a better choice when handling noisy quaternion/rotation measurements. Lemay et al. [**59**] reported a favorable comparison of the eKF with Monte Carlo simulation. Chang found that, for a general performance analysis, covariance analysis is the most appropriate technique and that while the use of the Riccati equation for non-linear problems is an approximate solution due to linearization it can serve as a lower bound on covariance [**23**] or, alternatively, an upper bound on performance.

Given that the eKF is a valid approximation tool for this problem domain, one still might like to be able to quantify the error that results from the first order approximation of a non-linear system. This will vary from sensor type to sensor type but we can examine our acoustic system (or any system that uses range measurements) using the error analysis presented for the Whisper acoustic tracker [**96**]. Given a traveling velocity of v and an update rate of δt, the maximum distance a tracked target can travel in a single δt is $(v * \delta t)$. The maximum error occurs when the target travels in a perpendicular direction to the range measurement (i.e. along the range circle/sphere circumference). The range measurement will not change but the target has moved a maximum of

(52)
$$dr = \sqrt{r^2 + (v * \delta t)^2} - r$$

where r is the current range measurement. Dividing through by r we find the fractional error to be

(53)
$$dr_{fractional} = \frac{\sqrt{r^2 + (v * \delta t)^2} - r}{r}$$

which is significant for small r but falls off as a function of $1/r$. Note that as δt increases, dr increases as well highlighting the fact that *any* linearization of a non-linear process can limit error by minimizing the time between measurement updates.

6.1.2. Global Discretization Error.

6.1.2.1. *Observed Order of Accuracy.* The choice of spatial discretization (i.e. the spacing of our sampled points) can effect the apparent P^∞ result. Because we are interpolating between calculated P^∞ values (Section 3.6), there will be some loss of fidelity due to the spatial frequency chosen. Borrowing from the methods employed by CFD mathematics, we can calculate the spatial *observed order of accuracy* [**82**] of the computed solution and determine whether or not our chosen grid spacing (or element size) is sufficient. Discretization error (DE) is defined as a function of the grid spacing (h) and is proportional to h^p where $p > 0$ is the order of the method such that

(54)
$$DE = Ch^p + HOT$$

where C is a constant and HOT refers to the Higher Order Terms. As h goes to zero (the discretization becomes more dense), the first term in Equation (54) dominates.

Given two solutions, f_1 and f_2, along with an exact solution, f_{exact}, we can calculate the discretization errors, DE_1 and DE_2 where

$$\begin{aligned} DE_1 &= norm_2(f_1 - f_{exact}) \\ DE_2 &= norm_2(f_2 - f_{exact}). \end{aligned}$$

Solutions f_1 and f_2 are calculated at two different grid spacings, h_1 and h_2 where typically $h_2 = 2h_1$ and the *grid refinement factor* is defined as $r = h_2/h_1$ (typically 2). Given these parameters, we can calculate the observed order of accuracy, p, where

$$p = \frac{ln(\frac{DE2}{DE1})}{ln(r)}. \tag{55}$$

We know the theoretical order of accuracy of the eKF and, therefore, this method is first-order ($p = 1$) so observed orders at or near one are desirable.

6.1.2.2. *Acoustic System.* Using a single horizontal plane ($z = 0.76$) of the acoustic system as an example, we calculate f_1, f_2 and f_{exact} where h_1 is spaced at 0.01 m, h_2 is spaced at 0.02 m and h_{exact} calculates exact solutions to P^∞ for comparison to the interpolated values that results from h_1 and h_2. Figure 6.2 and Figure 6.3 show both the coarse (f_2, DE2) and fine (h_1, DE1) grid solutions next to the exact solution (f_{exact}) for comparison. Clearly, the fine solution is closer to the exact solution than the coarse one but both are inexact.

This difference is calculated and shown in Figure 6.4 where we observe maximum errors of 0.0015 and 0.0039 for fine and coarse gridding respectively. Solving for Equation (55), we find $p = 1.57$ so that the global observed order of accuracy of our numerical solution is acceptable given the first-order solution of the eKF.

6.1.2.3. *HiBall System.* Again using a single horizontal plane ($z = 1.9$ m), we can calculate the observed order of accuracy for a HiBall analysis where h_1 is spaced at 0.1 m, h_2 is

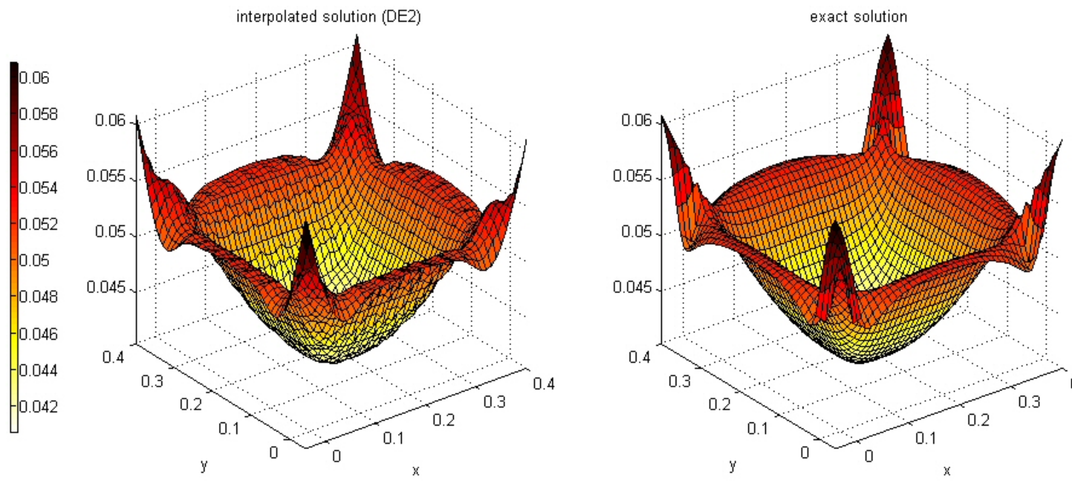


FIGURE 6.2. Coarse grid solution vs. the exact solution

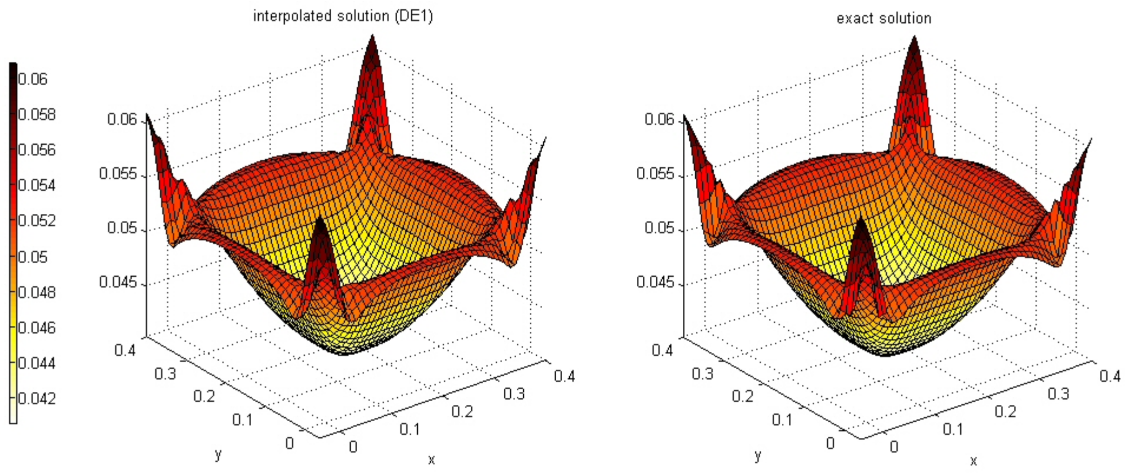


FIGURE 6.3. Fine grid solution vs. the exact solution

spaced at 0.2 m and h_{exact} calculates exact solutions to P^{∞} for comparison to the interpolated values that results from h_1 and h_2. Figure 6.5 and Figure 6.6 show both the coarse (f_2, DE2) and fine (h_1, DE1) grid solutions next to the exact solution (f_{exact}) for comparison. As in the acoustic example, the fine solution (Figure 6.6) is closer to the exact solution than the coarse one (Figure 6.5) but both are inexact.

Figure 6.7 shows the difference between both the fine and course solution and the exact solution. Perhaps unexpectedly, the maximum error from the course solution is 0.000162

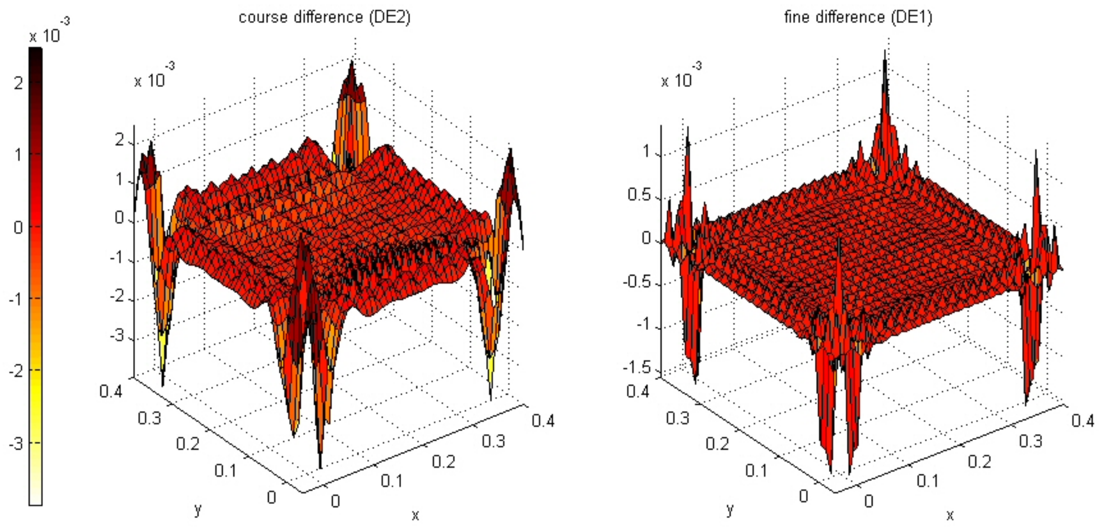


FIGURE 6.4. Difference between two mesh level and the “exact” solution

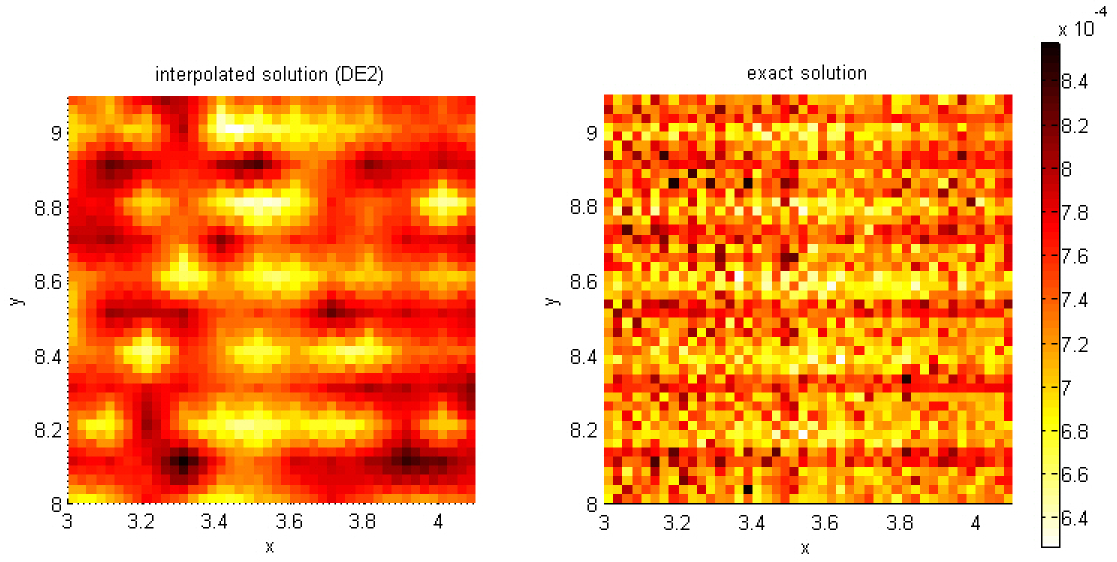


FIGURE 6.5. Coarse grid solution vs. the exact solution

and the maximum error from the fine solution is 0.000164. However, examination of the means shows that the mean of the fine solution error is $2.38e-05$, 4.72e-06 less then the mean of the course solution ($2.85e-05$).

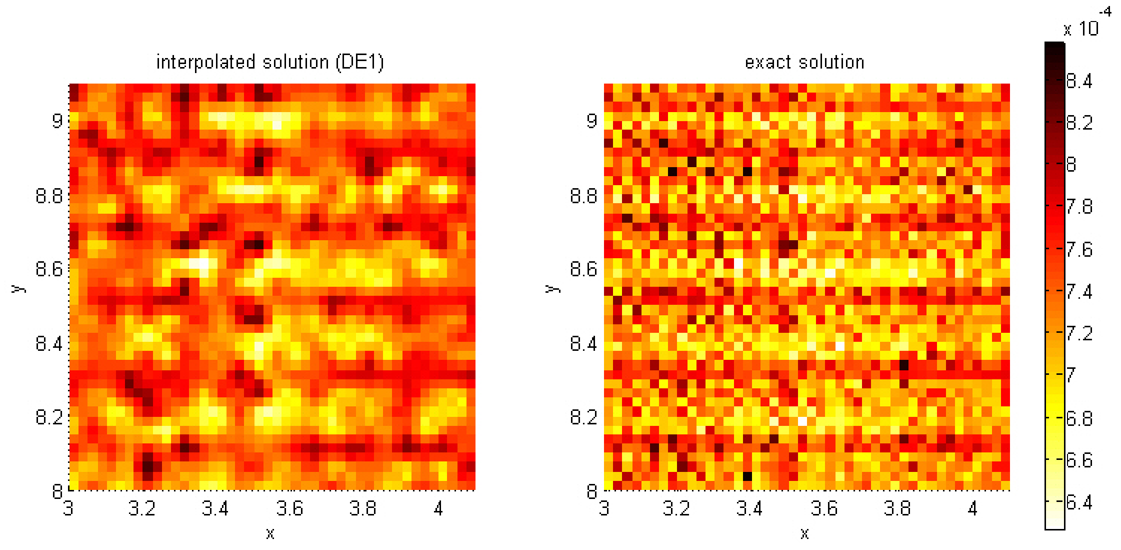


FIGURE 6.6. Fine grid solution vs. the exact solution

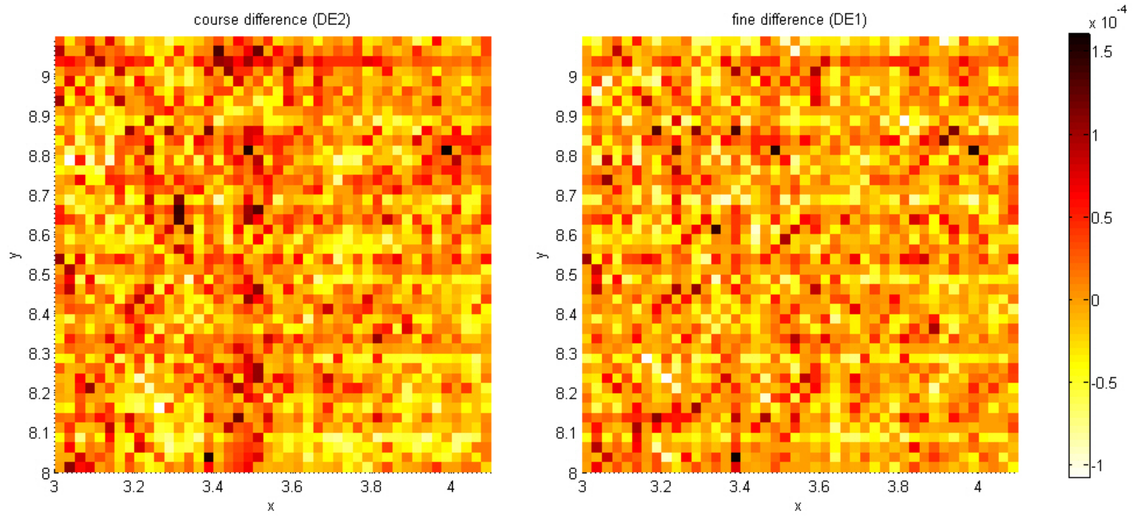


FIGURE 6.7. Difference between both mesh levels and the "exact" solution

Solving for Equation (55), we find $p = 0.08$ with gridding (h_1) of 0.05 meters. This does not approach the first-order solution that we know we have and so we need to more densely sample. Decreasing h_1 to 0.01 meters ($h_2 = 2h_1 = 0.02$ meters) produces a global observed order of accuracy of 0.86, an acceptable observed order of accuracy. Comparisons of these new coarse and fine interpolations are shown in Figure 6.8 and Figure 6.9.

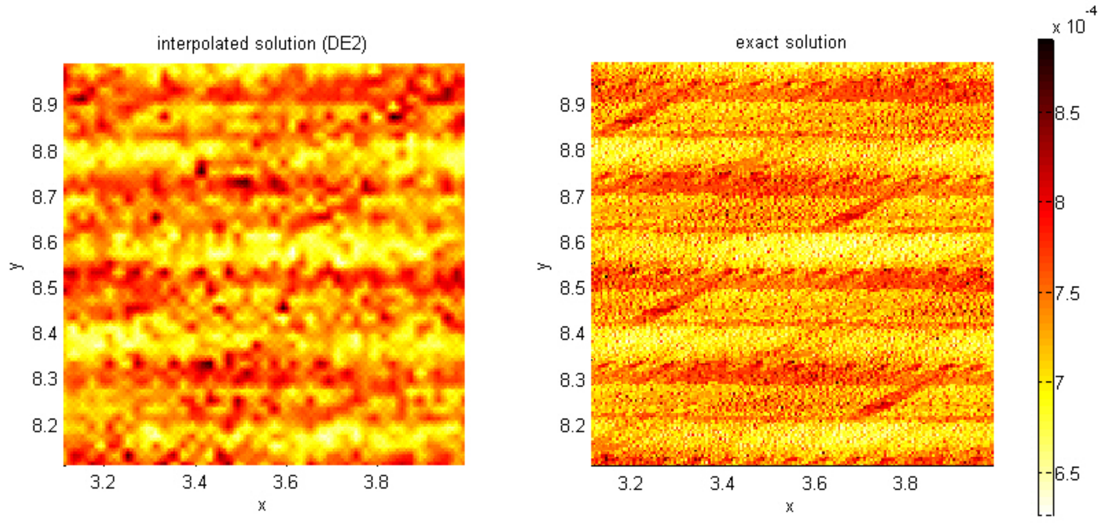


FIGURE 6.8. Coarse grid solution vs. the exact solution with denser sampling

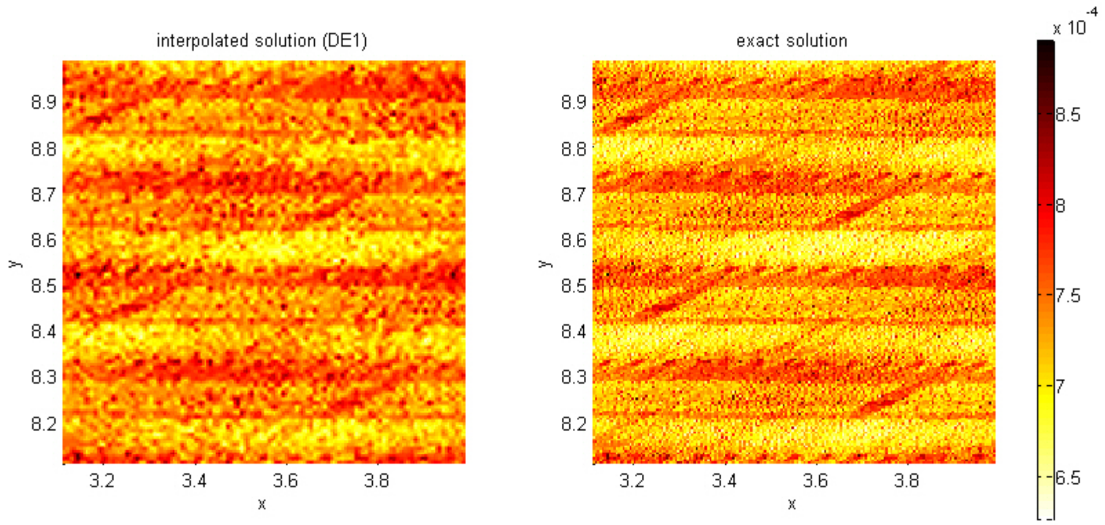


FIGURE 6.9. Fine grid solution vs. the exact solution with denser sampling

Figure 6.10 shows the difference between both the fine and course solution and the exact solution for the denser meshes. Again, the maximum error from the course solution (0.00018) is greater than maximum error from the fine solution (0.00017). Examination of the means again shows that the mean of the fine solution error is $2.08e-005$, $4.72e-06$ less then the mean of the course solution ($2.58e-005$). While the maximum errors in the dense mesh solutions are slightly higher than the maximum errors in the sparse mesh solutions, the means for both the course and and fine meshes are smaller by approximately $2.9e-05$.

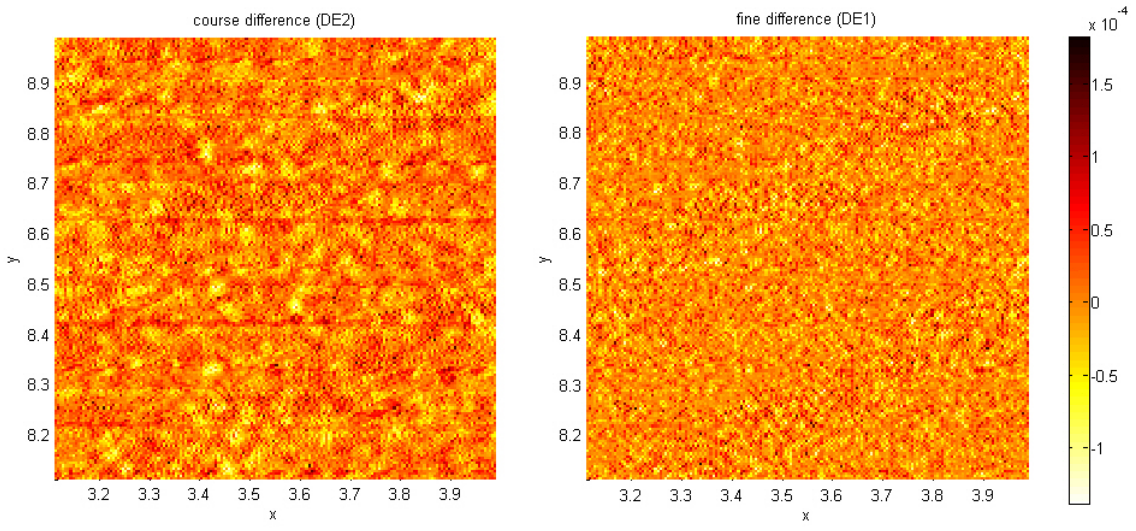


FIGURE 6.10. Difference between both denser mesh levels and the “exact” solution

6.2. Measurement-Level Information Optimization

6.2.1. Noise Covariance Error. The equations for the Hamiltonian (ψ) and the DARE itself (Equations (33), (28) and (27)) reveal a ratio between the process noise covariance, Q, and the measurement noise covariance, R. In all three cases, we see a Q/R relationship which indicates that the ratio of Q to R is important in the P^{∞} result. This relationship is further specified by examining the derivative of P^{∞} with respect to A, H, Q and R as shown in Table 6.1. In order to compute the derivative with an understandable and readable result, the values of A, H, Q, and R were defined as scalars. While this does impact the result (i.e. QH^2 instead of HQH^T), the overall Q and R relationship is preserved as shown by the derivatives themselves. Here we see clearly that as $Q \gg R$, the values in the left columns of the Hamiltonian, ψ, will increase forcing an increase in the eigenvalues and negatively impacting the estimated system performance, P^{∞}.

TABLE 6.1. Sensitivity of the Hamiltonian

Process Model	$\frac{d\psi}{dA} = \begin{bmatrix} 1 - \frac{QH^2}{A^2R} & -\frac{Q}{A^2} \\ -\frac{H^2}{A^2R} & -\frac{1}{A^2} \end{bmatrix}$	$\frac{d\psi}{dQ} = \begin{bmatrix} \frac{H^2}{AR} & \frac{1}{A} \\ 0 & 0 \end{bmatrix}$
Measurement Model	$\frac{d\psi}{dH} = \begin{bmatrix} 2\frac{QH}{AR} & 0 \\ 2\frac{H}{AR} & 0 \end{bmatrix}$	$\frac{d\psi}{dR} = \begin{bmatrix} -\frac{QH^2}{AR^2} & 0 \\ -\frac{H^2}{AR^2} & 0 \end{bmatrix}$

Recall that Q is a function of δt as shown in Equation (23) through Equation (25). Therefore, one approach to reducing the impact of Q on P^{∞} is to minimize δt. The quality of an approximation improves as δt decreases and, in Figure 6.11, the effect of increasing the sampling rate (decreasing δt) on the example acoustic model is shown for six frequencies from 6 Hz to 200 Hz (δt range 0.1667 s to 0.005 s) doubling with each increment. The form of the P^{∞} result remains constant as pictured but the range of P^{∞} EigMax values do change as shown in the colorbars for each frequency. The plot on the right shows that both the maximum and minimum EigMax values decrease with δt and approach the noise floor. The calculated minimum and maximum EigMax values are shown in Table 6.2.

TABLE 6.2. Values for the (false) impact of changing dt on the Acoustic System performance

	6 Hz	12.5 Hz	25 Hz	50 Hz	100 Hz	200 Hz
$EigMax_{min}$	0.0546	0.0323	0.0231	0.0170	0.0126	0.0095
$EigMax_{max}$	0.2063	0.1460	0.1084	0.0817	0.0622	0.0476

From this one could conclude that driving a tracking system as fast as possible (smallest achievable δt) will secure the best performance. While it is true that a smaller δt will decrease the P^{∞} mathematical result and improve the predicted performance, there may be a price to pay for decreasing δt. A smaller δt could effect the operational bandwidth of a system or reduce the time available for signal processing and acquisition. In other words, as δt decrease and Q improves, the measurement noise, R, may be increasing. The potential consequences of changing δt on R are several-fold. One is a change in the operational bandwidth of the circuit electronics and the other is a change in the sampling time that effects the signal-to-noise ratio (SNR). In either case, reducing δt will reduce Q but the effect on the more important Q/R ratio depends upon the relationship between the two noise curves (Q and R) as a function of time or frequency.

6.2.1.1. *Measurement Noise vs. Bandwidth.* One consequence of a changing δt could call into question the noise model from which we generate R. Sampling rate (δt) determines the frequency range (or bandwidth) which can be represented in a digital waveform. Signals sampled at a high sampling rate can represent a broad range of frequencies and hence have broad bandwidth. The maximum bandwidth of a sampled signal is determined by the sampling rate since the maximum frequency representable in a sampled waveform (the Nyquist frequency) is equal to one half the sampling rate. In an effort to capture higher frequency signal content, an increase in the sampling rate might be necessary. Prior to signal sampling (or digitization), analog signal conditioning occurs. These analog electronics

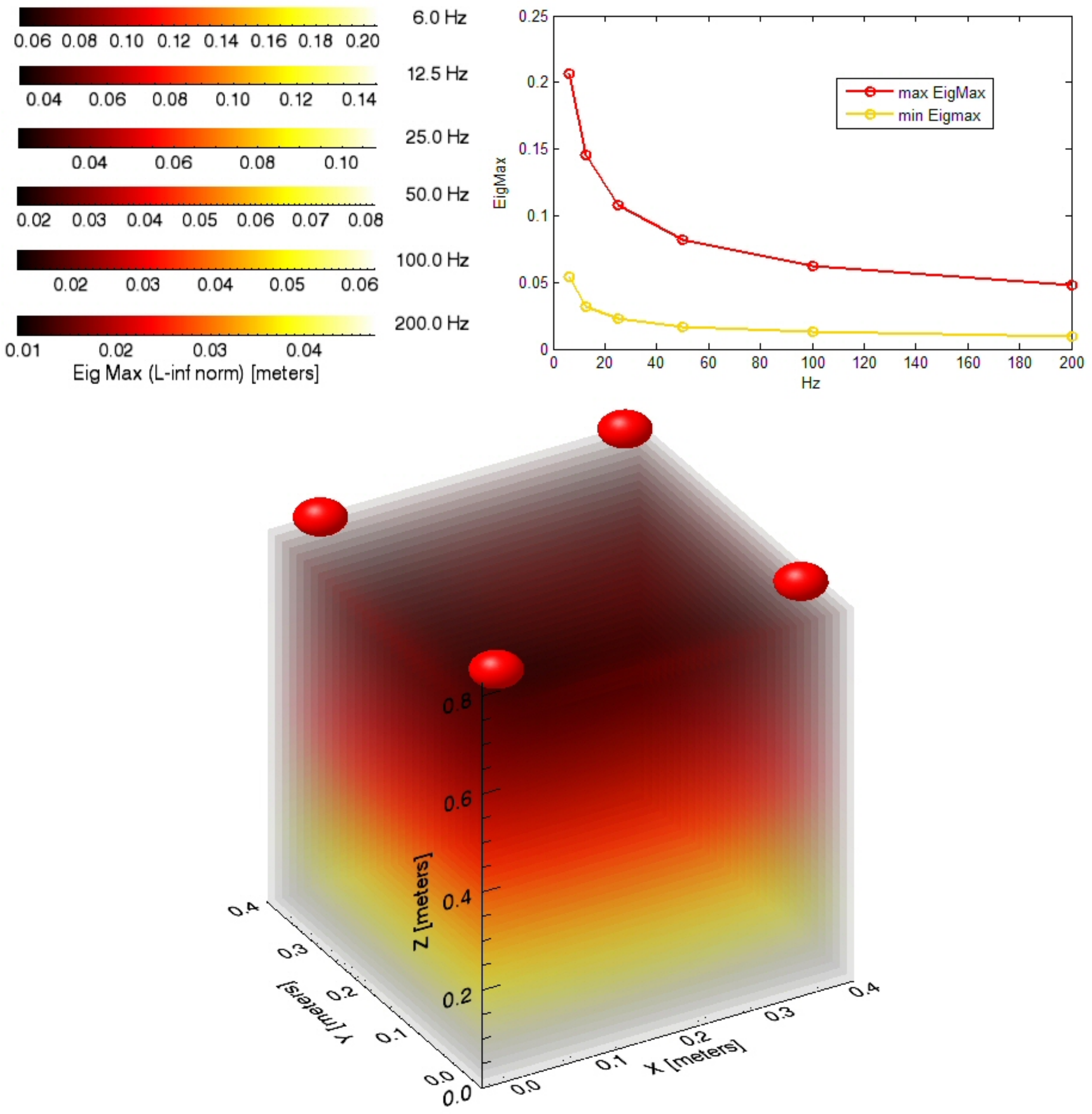


FIGURE 6.11. The (false) impact of changing dt on the Acoustic System performance

impose a frequency band that limits the frequency content of the signal, typically a low-pass band which filters out high frequency signal content. This circuit *corner frequency* (the bandwidth of the circuit) is designed to handle a signal given the signal bandwidth and so simply increasing the sampling rate may not achieve the desired result. A change to the circuit design could be required and this change could change the corner frequency which effects the noise model. Noise models are created given a specific bandwidth, B, and the electronic noise voltages and currents depend on this bandwidth. In general, the greater the bandwidth, the greater the electronic noise. So, even though the noise values used in the R matrices presented here may be constant scaler values or dependent on the distance the signal must travel, there is an inherent frequency/time component. In other words, the scalar value or the multiplier used to compute R is valid for a particular bandwidth or frequency range but not all bandwidths.

Noise is best characterized by the Fourier transform of the time-varying fluctuations in electric current, which is called the noise spectral density [**31**]. Noise spectral density is the average noise in a $1Hz$ bandwidth centered at frequency f Hz and c is a constant amplitude. Therefore, electronic noise is often measured in $V/\sqrt{Hz}$ (volts per root-Hertz) or $A/\sqrt{Hz}$ (amperes per root-Hertz) because the actual noise generated is dependent upon the operational bandwidth. Typical noise models include noise sources arising from white noise components (Table 6.3) such as thermal noise in resistors (Johnson noise) and shot noise due to random fluctuations in the motion of charge carriers in a photodetector junction (i.e. current flow) along with component noise (Figure 6.12).

TABLE 6.3. Electronic Noise Sources

	[volts]	[amperes]
Johnson Noise	$V_{noise}(rms) = V_{nR} = (4kTRB)^{\frac{1}{2}}$	$V_{noise}(rms) = V_{nR} = (\frac{4kTB}{R})^{\frac{1}{2}}$
Shot Noise	$I_{noise}(rms) = I_{nR} = (2qI_{dc}B)^{\frac{1}{2}}R_e$	$I_{noise}(rms) = I_{nR} = (2qI_{dc}B)^{\frac{1}{2}}$

In Table 6.3, it is clear that both Johnson and shot noises are a function of B. The other values are constants (Boltzmann's (k) and electron charge (q)), T is temperature, and I is current. Figure 6.12 shows an example of a typical noise voltage (V_n) curve found in a data sheet. The curve on the left clearly shows that V_n depends upon the frequency or bandwidth of the noise along the horizontal axis. Note that the vertical axis units for V_n are $\mu V/\sqrt{Hz}$. An important point here is to note the *1/f corner frequency* of the input noise voltage. From the shape of the V_n curve, we can observe that $V_n(f) \propto \frac{1}{f}$ which is clearly not Gaussian white noise as it is frequency dependent. However, at sufficiently high frequencies (those above the corner frequency), we can treat this V_n as normally-distributed white noise.

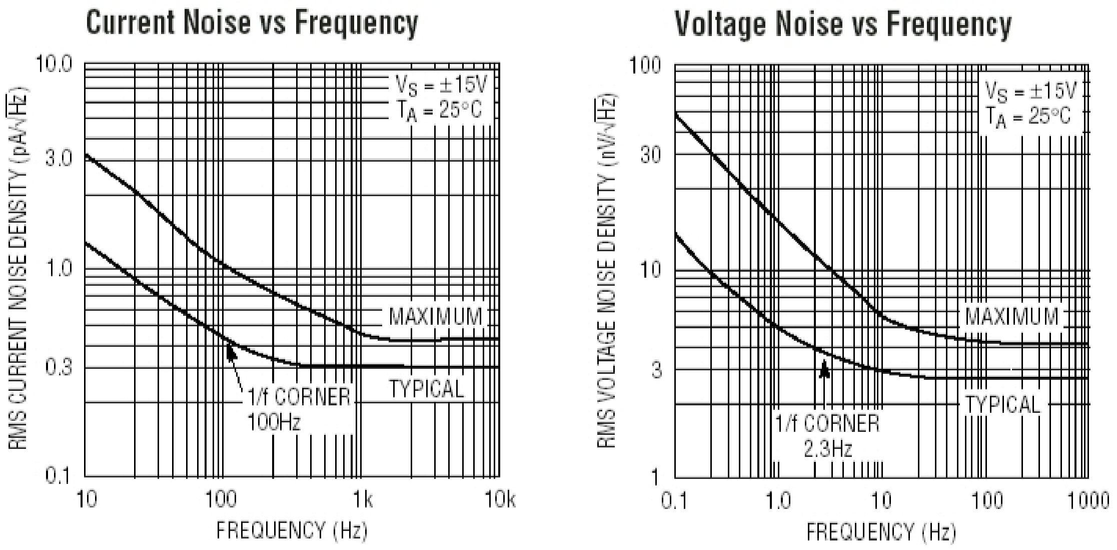


FIGURE 6.12. Noise Density of the LT1124CSB op amp

In all three cases (thermal, shot and component), the noise is a function of bandwidth which could be impacted by a change in δt. Decreased δt could mean an increase in bandwidth which would force a change in R.

6.2.1.2. *Measurement Noise vs. Process Noise.* A second potential impact of shortening or lengthening δt is its effect on the time available for the signal sampling interval. Once we have a noise model at the component level as described in Section 6.2.1.1 and with more detail in [**25**], we need to account for any additional signal processing that might occur such as averaging as seen in CCD dwell time in an optical system like the HiBall or the length of the correlation window in an acoustic tacker. Since the signal integrates (or averages) linearly with time while the noise integrates with the square root of time (see above), the longer the sample window, the better the signal to noise ratio becomes. However, the penalty paid for a longer sampling window is increased uncertainty in pose estimation due to user motion as described in the statistical representation of the user motion, Q.

The potential user motion noise increases as the measurement noise decreases over a δt interval. A well-behaved *averaging* system will operate at or before the "$Q-R$ crossover" where the signal-to-noise ratio can be maximized such that the process noise, Q, does not overwhelm the measurement noise, R. A system that employs instantaneous measurement devices (such as inertial sensors) will not be subject to the effect of the $Q-R$ crossover as described here and will see improved performance by minimizing δt in order to sample as quickly as possible. For an example of a Q-R crossover analysis, see the HiBall Case Study in Section 7.3.1.

In work subsequent to that presented here and at the defense, we determined that the minimum of the $(Q+R)$ curve is the optimal operating point for averaging systems [**99**]. This is shown in the green curve labeled $R+HQH^T$ in Figure 7.7 of Section 7.3.1.

CHAPTER 7

The HiBall^{TM} System: A Case Study

In this case study, I will apply the techniques presented in Section 6 to a working tracking system, the HiBall manufactured by 3rdTech, Inc. Figure 7.1 shows the HiBall-3000 ceiling beacon arrays, the HiBall-3000 sensor and a typical head-tracking application.

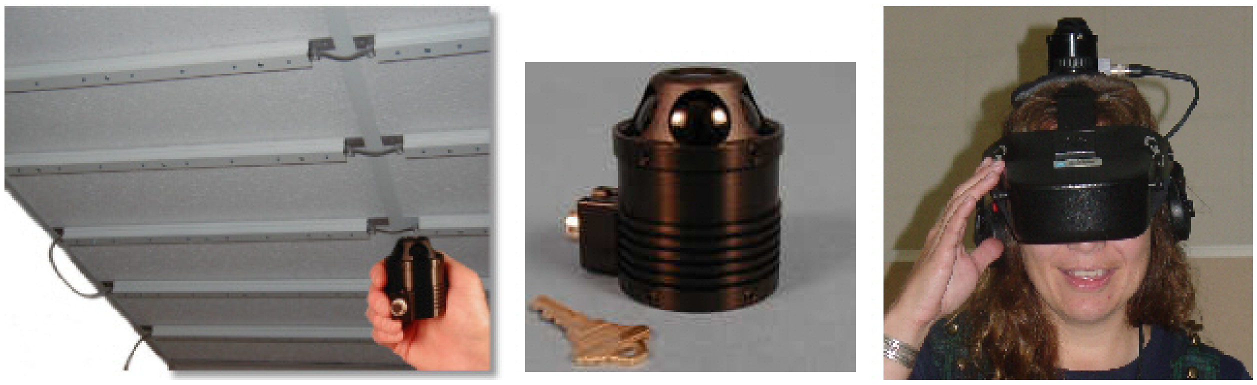


FIGURE 7.1. The HiBall-3000

Specifically, the HiBall-3000 Tracker housed in the computer science department at UNC Chapel Hill was used to gather sensor data for this study. The HiBall was conceived and developed at UNC and the close relationship between the computer science department and 3rdTech permitted an "under the hood" access that I could not have gained anywhere else. I was able to modify source code to access internal filter parameters during runtime and 3rdTech graciously contributed their time and insight to helping me understand the inner workings of the HiBall software.

7.1. The HiBall^{TM} System at Non-Zero Rotations

In Section 5, the performance of the HiBall system was compared to and validated against the P^∞ predicted performance. As noted in Section 5, this validation process forced an artificial configuration of the HiBall system, one element of which is the zero rotation of the HiBall sensor throughout the validation experiments. In a real human tracking application, the target and HiBall sensor rotation would not be static and the tracking performance would be effected. To illustrate the behavior of the HiBall at other rotations, P^∞ predicted performance estimates for rotations of 10°, 20°, and 30° around each individual axis (x, y, and z) in the full-ceiling configuration at a height of $z = 1.9\ m$ (approximately head height) were generated across one corner of the ceiling. An area of nine square meters defined by x from 3.0 meters to 6.0 meters and y from 6.0 to 9.0 meters was analyzed. As with the zero-rotation analysis, this area stretches from the near-center of the tracked space (x=3.0 and y=6.0) to the edge of the HiBall ceiling and the room itself (x=6.0 and y=9.0) and matches the area observed with the actual HiBall system as shown in Figure 5.2. Visualizations of these results are shown in Figure 7.2, Figure 7.3, and Figure 7.4, respectively.

In each figure, we observe that the overall shape of the performances at head-height is similar to those at zero rotations shown in Chapter 5. We can also observe that, for all three axes, the predicted performance improves as we move away from zero degrees. The value ranges for all rotations are shown in Table 7.1. We observe that the range at all rotations are larger then the zero-rotation [0.182, 0.263] mm range and that all show improved performance over zero-rotation towards the venter of the analyzed area. We also observe performance degradation at the ceiling edges. Both the improvement and degradation can probably be attributed (at least in part) to the HiBall sensor rotation. When rotated or turned towards the center of the tracked ceiling, more LEDs are visible and, conversely, when turned towards the edge of the ceiling and as the ceiling edge is approached and the sensor views extend beyond the ceiling edge, fewer LEDs are visible.

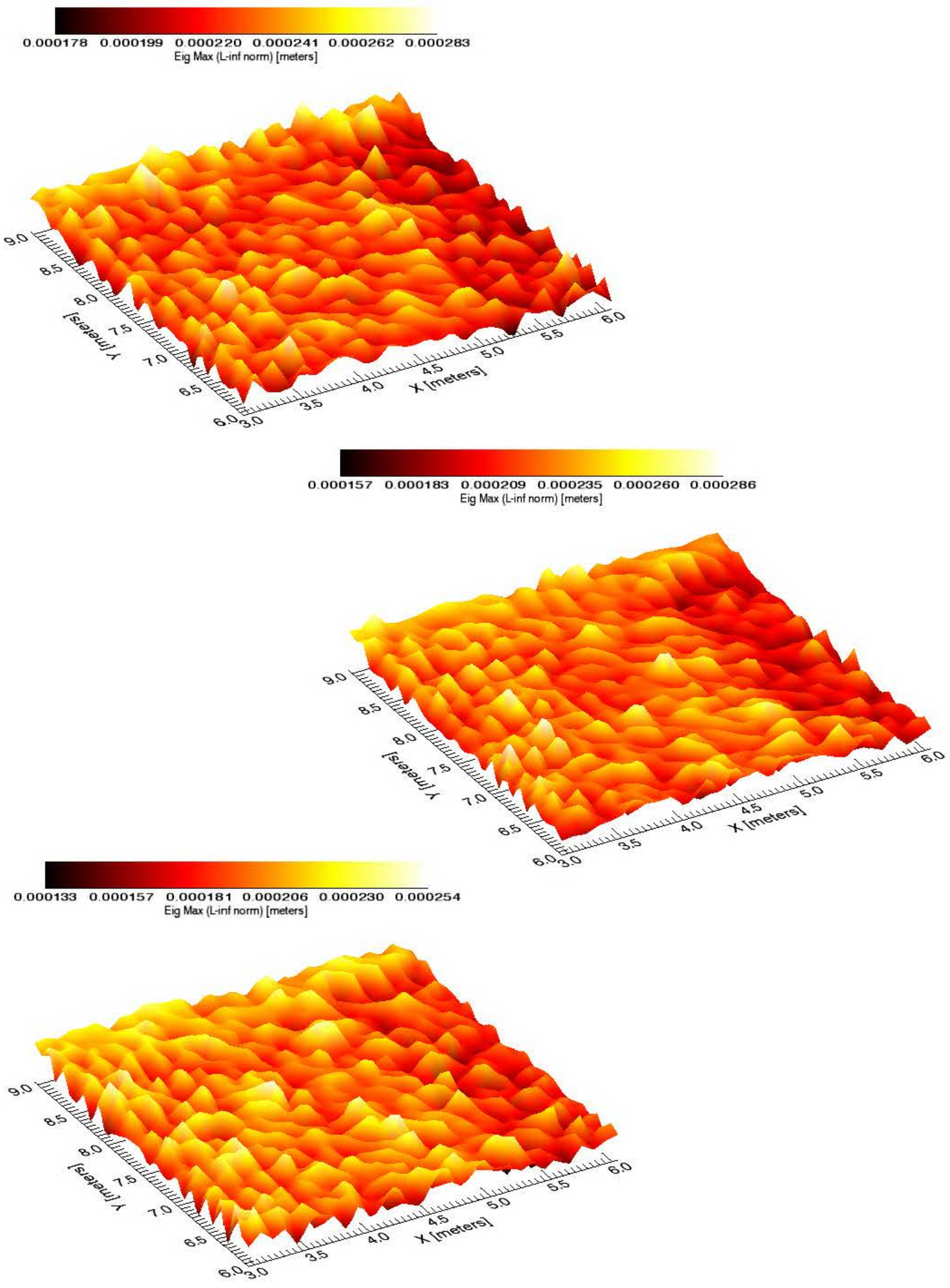
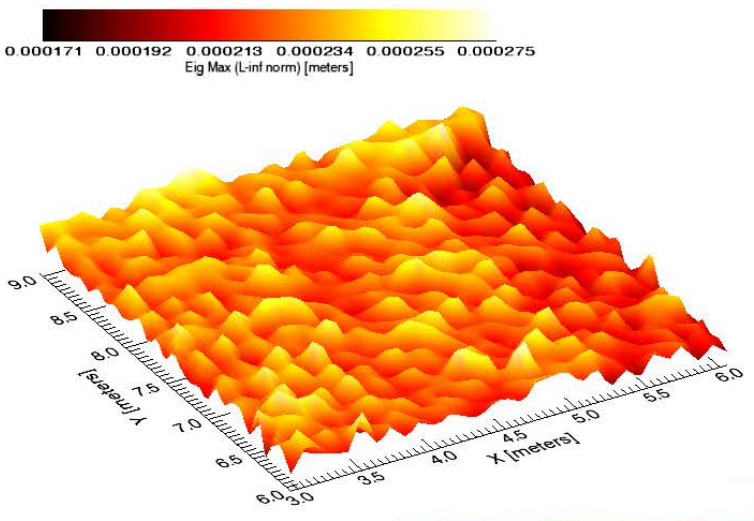
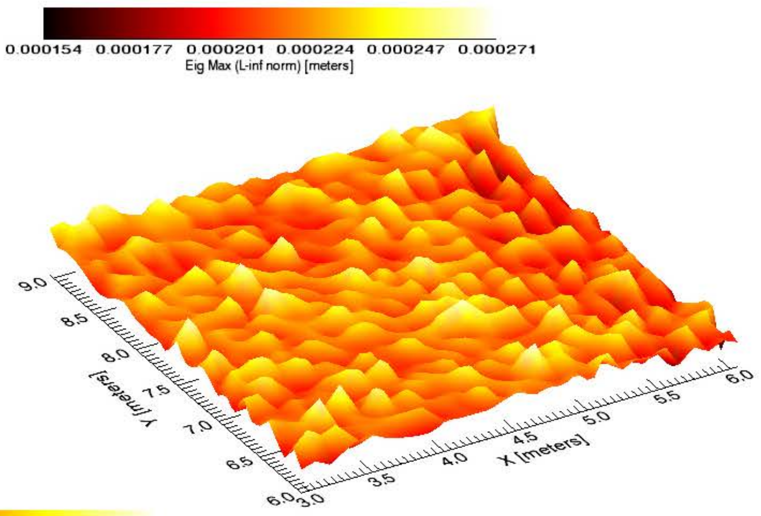


FIGURE 7.2. HiBall P^{∞} estimates rotated 10, 20 and 30 degrees (top to bottom) around x-axis.

TABLE 7.1. HiBall performance estimate ranges [mm] for rotations of 10°, 20° and 30° around the x, y (horizontal) and z (vertical) axes

\ [mm] degrees	x-axis min	x-axis max	y-axis min	y-axis max	z-axis min	z-axis max
10°	0.178	0.283	0.171	0.275	0.179	0.281
20°	0.157	0.286	0.154	0.271	0.171	0.277
30°	0.133	0.254	0.126	0.266	0.150	0.274





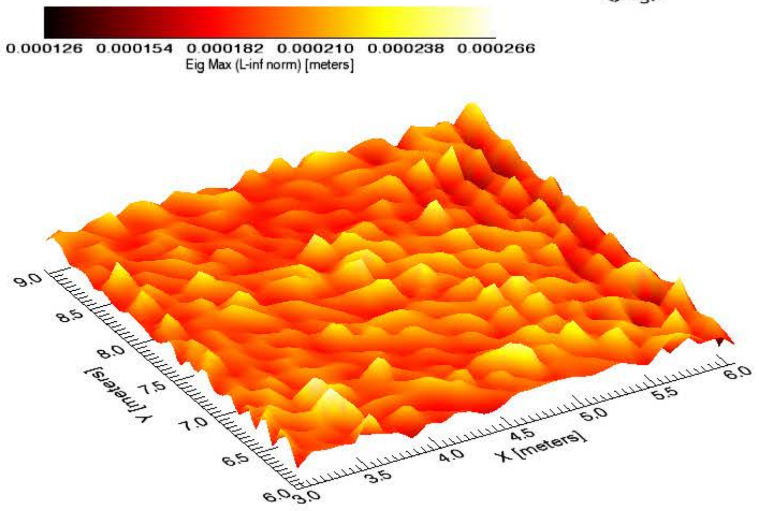


FIGURE 7.3. HiBall P^{∞} estimates rotated 10, 20 and 30 degrees (top to bottom) around y-axis.

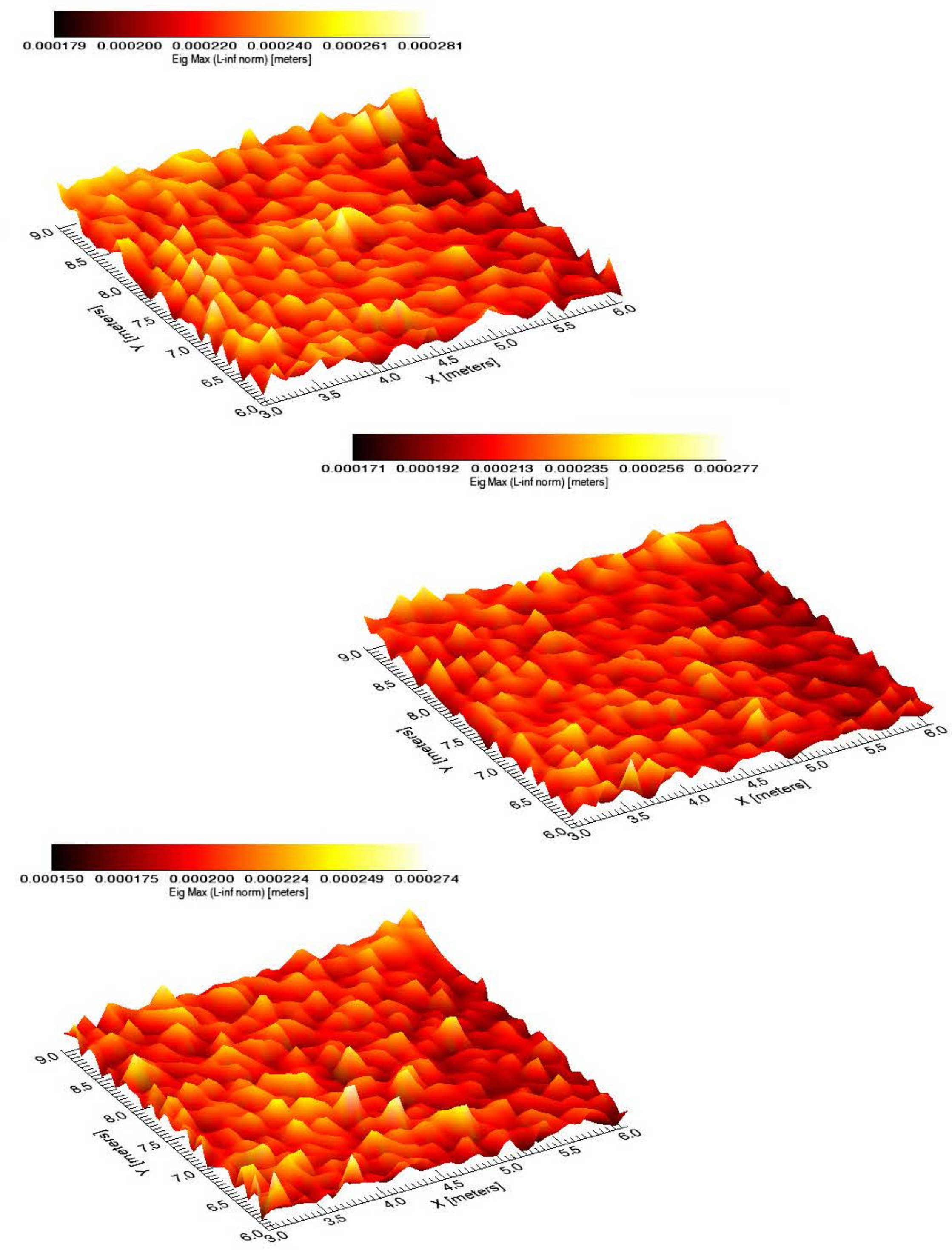


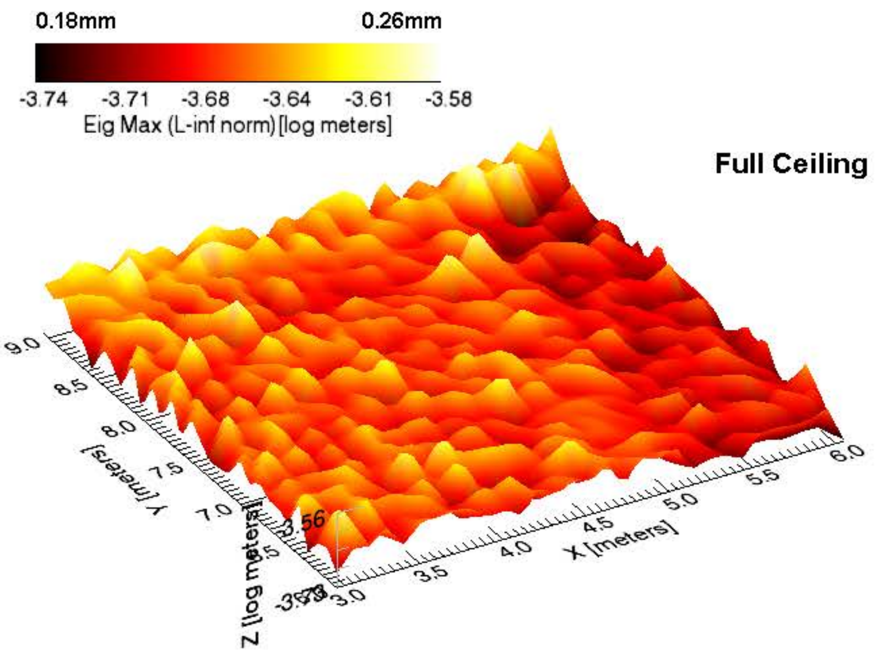
FIGURE 7.4. HiBall P^{∞} estimates rotated 10, 20 and 30 degrees (top to bottom) around z-axis.

7.2. System-Level Information Optimization

In Section 5.1.1.1, the HiBall performance was validated in both the full (operational) and sparse ceiling configurations. Figure 7.5 juxtaposes the validated performance of the current HiBall in both the full- and half-ceiling configurations in a horizontal plane of approximately head-height (1.9 meters) at zero rotation. A side-by-side comparison reveals that the highest penalty paid for halving the number of ceiling LED rows is at the lower end of the performance spectrum and on the ceiling edge. In the full ceiling configuration, the performance ranges from 0.18 mm of error to 0.26 mm, a range of only 0.08 mm. In the half ceiling configuration, the performance ranges from 0.18 mm of error to 0.47 mm, increasing the performance range to 0.29 mm. The best performance (-3.74 log meters = 0.18 mm) remains unchanged between the two configurations while the low performance number degrades by only 0.22 mm from 0.26 mm to 0.48 mm. In the more central locations, the performance range widens only slightly with the most noticeable difference in the form or shape of the performance where we see a marked LED row pattern in the half-ceiling configuration.

This tells us that the number of strips could probably be reduced or a greater area covered with the same number of strips in future while maintaining sub-millimeter accuracy in future HiBall configurations. The sparse (or half) ceiling configuration was chosen because this was a configuration that could be duplicated in a commercial HiBall system without moving ceiling strips so that the full HiBall tracking system could remain operational for other users.

However, we can examine any conceivable ceiling configuration through P^{∞} analysis to determine how many ceiling strips can be disabled before a suffering a serious performance hit. We can consider other LED arrangements such as a checkerboard pattern than rather than a row pattern. The resulting P^{∞} analysis of this ceiling configuration is shown in Figure 7.6.



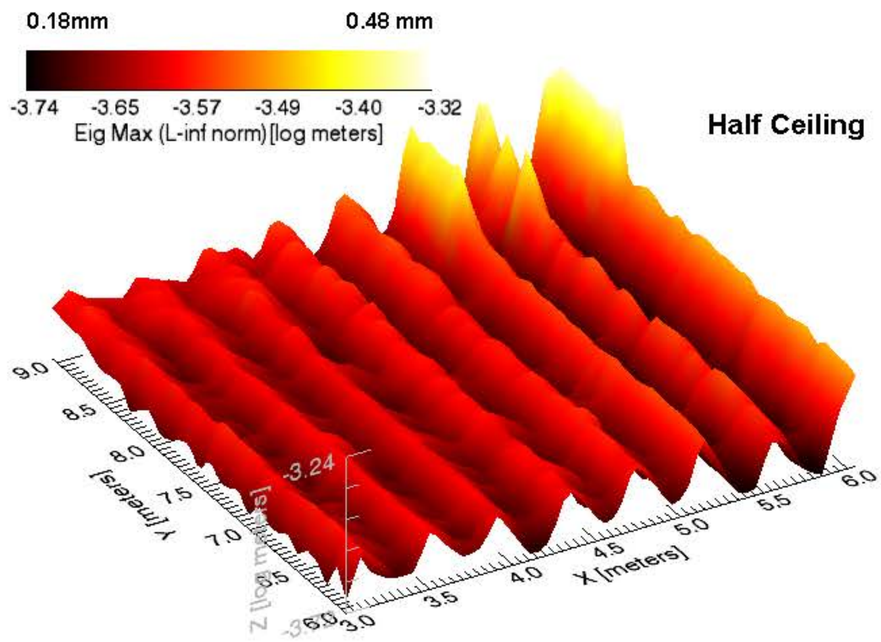


FIGURE 7.5. Full (upper) and Half (lower) Performance with Current (6 LEPD) HiBall Configuration

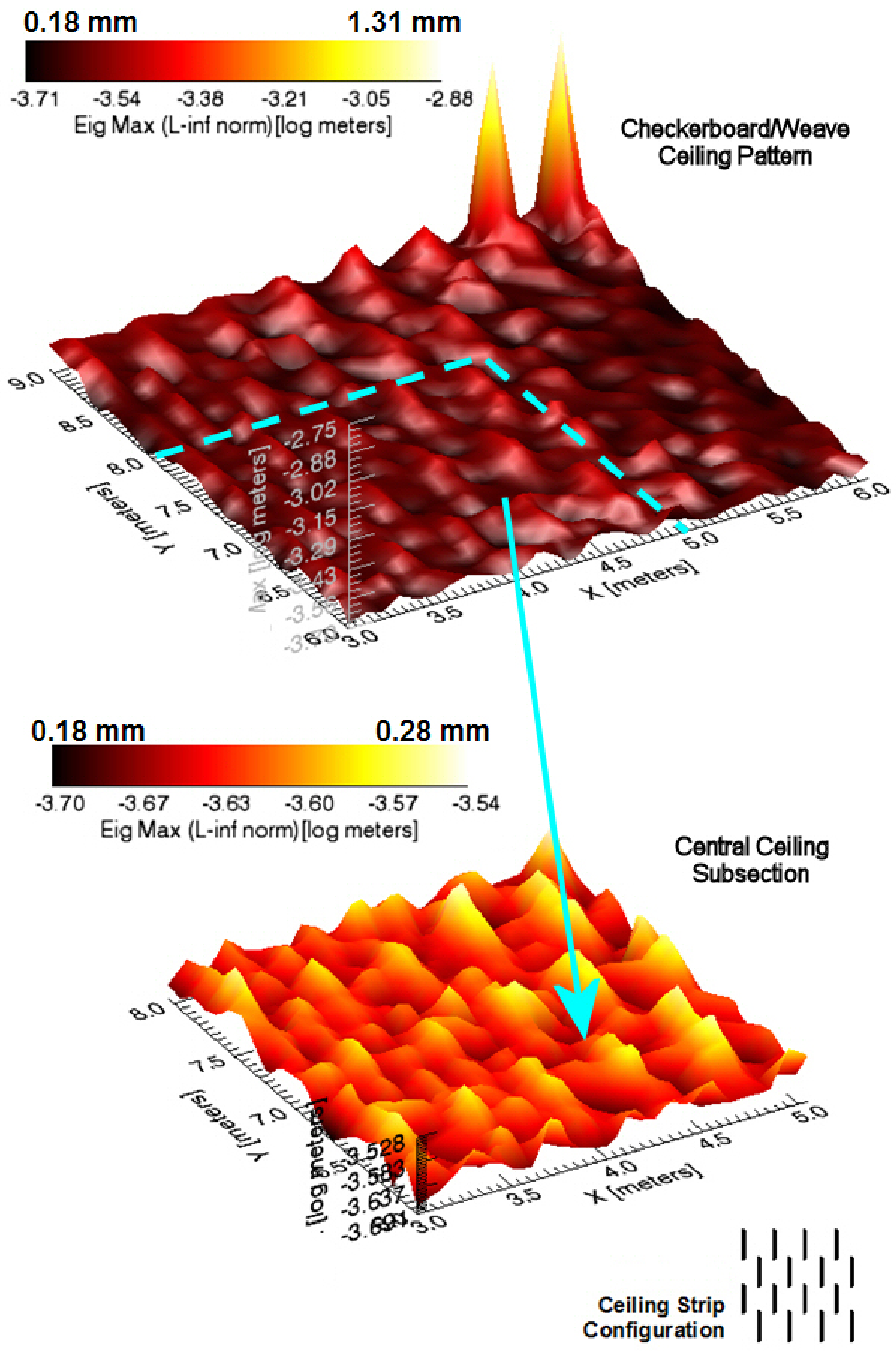


FIGURE 7.6. P^{∞} results for the "checkerboard" ceiling strip pattern

Initially, given the performance metric range of $[-2.75, -3.73]$ log meters ([1.7780.186] mm) on the top in Figure 7.6, it might appear that this configuration does not perform as well as the half-ceiling configuration in Figure 7.5. However, as a result of the checker-boarding, the strip at the outer corner (located at approximately [x,y]=[6,9] was eliminated and so we see a spike (i.e. a decline) in performance at that location. Narrowing the surface to exclude the outer corner (on the bottom in Figure 7.6), we find performance numbers that are very close to those in the full-ceiling configuration in Figure 7.5. Even though the same number of strips in total are used in the checkerboard-ceiling configuration as are used in the half-ceiling configuration, the row demarcation visible in the half-ceiling configuration is missing from the checkerboard-configuration and instead we observe a diagonal flow in the form reminiscent of the full-ceiling configuration. Using the checkerboard-ceiling configuration appears to have eliminated the location of the strips as the dominate influence on performance in favor of the influence of the HiBall sensor rotation. This two observations (dominating influence on performance and performance value range) suggest that the checkerboard-ceiling will outperform the half-ceiling configuration even though the same number of ceiling strips (half) are used in both configurations.

7.3. Measurement-Level Information Optimization

7.3.1. HiBall$^{\text{TM}}$ Sample Time. Using the analysis technique presented in Section 6.2.1 [1], the $Q-R$ crossover point for the HiBall system can be calculated. Assuming no changes in the operational bandwidth dictated by the topology of the circuit (Section 6.2.1.1), we can focus on the dark-light-dark operation of the lateral effect photo-diode (LEPD) used to detect the centroid of light in two dimensions on the LEPD. The HiBall system differences repeated samples over dark-light-dark cycles. An integration (or "light") time, i, is set from one to fifteen LED clock cycles as determined by the distance from the HiBall to the sighted LED (greater distance means longer integration time for improved SNR). In all cases, the

[1] see update re: subsequent work on 109

sum of the two dark times is chosen to equal the light time. For example, if $i = 4$ (the LED is on for 4 clocks), then the HiBall first samples for 2 clocks with the LED off, then 4 clocks with the LED on, then 2 with the LED off. In short, the HiBall system differences $-2D + 4L - 2D$. If the light cycle is an odd length such as $i = 5$, then the "extra" dark cycle is added at the end so that the HiBall system differences $-2D + 5L - 3D$. In both cases (odd and even) the iterative result is dark-light cycles with i light cycles followed by i dark cycles followed by i light cycles, etc. for any given i. Given a clock cycle of $21\mu s$ and allowing for filter processing time, δt for a given i is calculated as

$$\delta t = 325 + (303 + 2 * i * 21) \quad \mu s \tag{56}$$

as per information from 3rdTech, Inc.

Chi [25] provides a function for calculating the mean and RMS uncertainty of the optical centroid on the LEPD given key parameters including integration time, noise densities, etc. Using this formula with updated parameters as required due to component substitution and the timing described in Equation (56), we can calculate this uncertainty (i.e. R) due to SNR at various integration times as dictated by i. We can also calculate the Q matrix for each δt value where, for the HiBall system,

$$Q_{HiBall} = \begin{bmatrix} 1/3\, qpxy\, dt^3 & 0 & 0 & 1/2\, qpxy\, dt^2 & 0 & 0 \\ 0 & 1/3\, qpxy\, dt^3 & 0 & 0 & 1/2\, qpxy\, dt^2 & 0 \\ 0 & 0 & 1/3\, qpz\, dt^3 & 0 & 0 & 1/2\, qpz\, dt^2 \\ 1/2\, qpxy\, dt^2 & 0 & 0 & qpxy\, dt & 0 & 0 \\ 0 & 1/2\, qpxy\, dt^2 & 0 & 0 & qpxy\, dt & 0 \\ 0 & 0 & 1/2\, qpz\, dt^2 & 0 & 0 & qpz\, dt \end{bmatrix} \tag{57}$$

for the state state position variables $[x \quad y \quad z \quad x' \quad y' \quad z']$ where $qpxy = 0.261782402027^2$ and $qpz = 0.0815344167^2$ as per the HiBall software.

Figure 7.7 shows a plot of both the measurement noise and potential noise curves over a range of δt values. The crossover point occurs at $\delta t = 0.993$ ms or approximately 1007 Hz. This suggests that the HiBall system could operate with improved estimation accuracy by limiting the update rate to 1007 Hz. Note that 1007 Hz is approximately equivalent to an integration time of eight clock cycles suggesting that allowing for integration times of nine and above, while improving the measurement noise, may adversely effect overall system performance. We also observe the minimum summed $(R + HQH^T)$ noise value of $1.564e-004$ occurs at approximately $\delta t = 0.964$ ms (1037 Hz). Sampling faster then this pushes the measurement noise into the non-linear "knee" of the curve towards a sharp increase and so we might see a negative impact on overall performance at sampling rates higher than 1037 Hz. This suggests that optimal HiBall performance will be achieved at integration times greater than $i = 6$ but less than $i = 9$ (i.e., between minimum summed $(R + HQH^T)$ and the Q-R crossover). Note that the current HiBall system uses and integration time of $i = 8$ at the analyzed ceiling height (i.e. approximate distance from the HiBall to the sighted LED) and so is operating in this optimal range.

This analysis applies for both the P- and PV-models used in the multi-modal Kalman filter implemented in the HiBall system. Figure 7.8 shows the previous PV analysis alongside the crossover point of an example P motion model. With this P motion model, the crossover point occurs later in time at approximately $\delta t = 1.5$ ms allowing for *more* than the maximum 15 cycle integration time currently employed by the HiBall system.

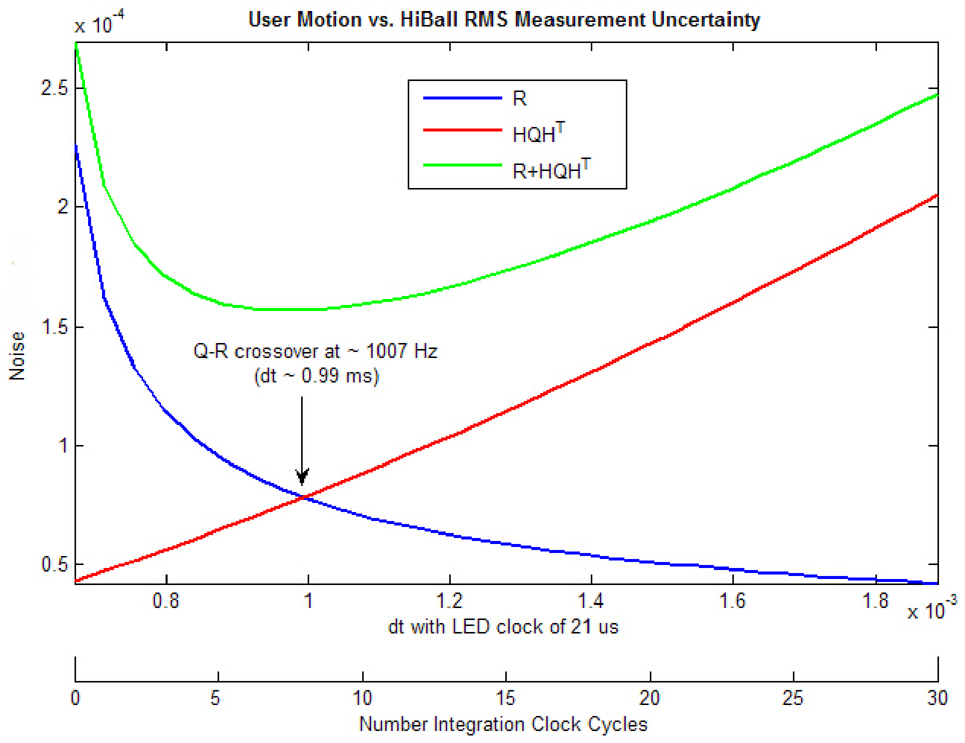


FIGURE 7.7. Analysis of HiBall measurement noise vs. potential process noise

7.4. System-Level Redesign Prediction

P^{∞} analysis can be used to predict the performance for a redesign of the HiBall sensor. Prediction is defined as the use of a computational model to foretell the state of a system under conditions for which it has not been validated [**2**]. Prediction enables one to make *inferences* based on validation evidence but not claims regarding the accuracy of predictions [**72**].

The HiBall sensor has six LEPDs and six lenses each with a narrow six degree (6°) field-of-view (FOV) and a persistent question of interest regarding the HiBall tracking system has been what effect a wide FOV (WFOV) would have on the HiBall performance. Would this effect be such that the number of LEPDs and lenses could be reduced in order to the reduce the size and weight of the HiBall sensor. Figure 7.9 shows the mechanical configuration of the current HiBall sensor. There are six LEPDs and six accompanying

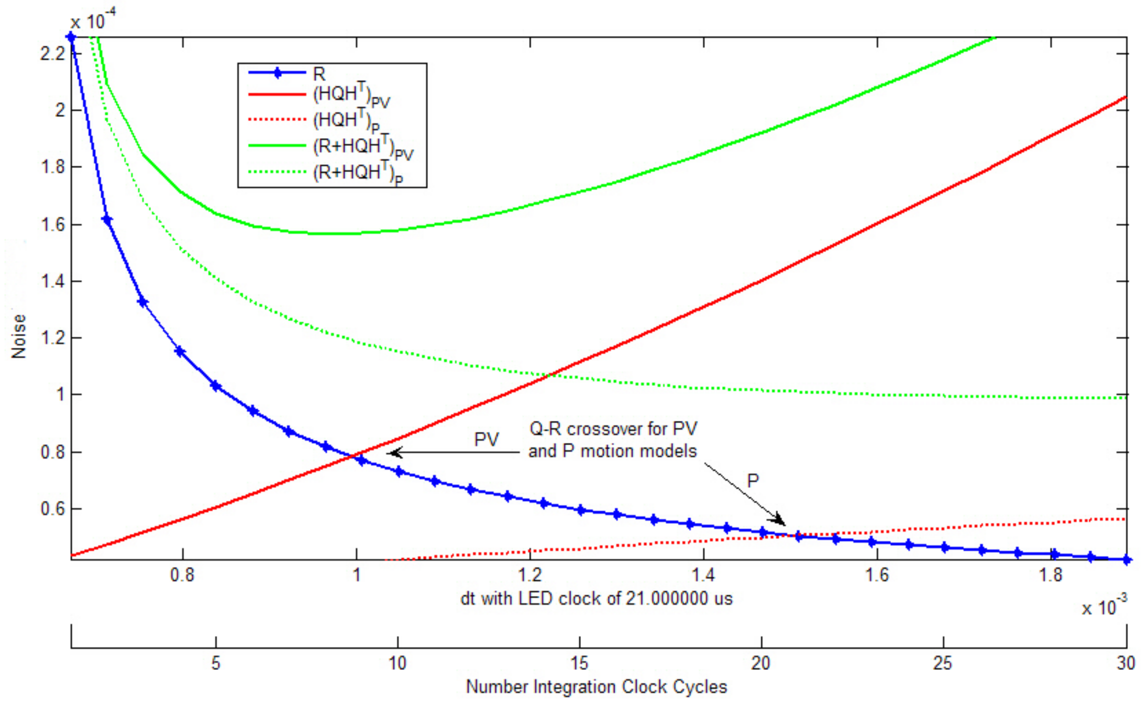
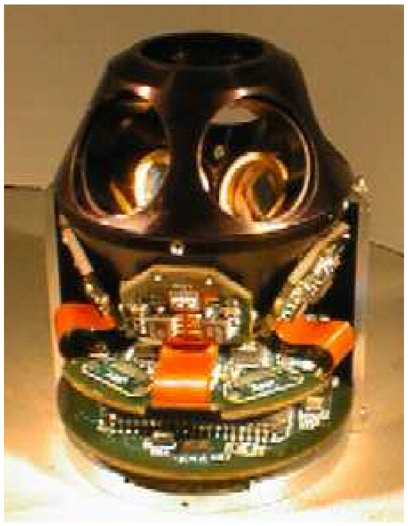


FIGURE 7.8. Analysis of HiBall measurement noise vs. potential process noise for P and PV motion

lenses such that each LEPD has a paired lens directly across from it on the opposite side of the HiBall housing. These are the six primary views of the HiBall. Additionally, each LEPD can see through some of the side lenses (secondary views) resulting in twenty-six total views.

Figure 7.10 shows the top and side views of the LEPD locations of the current HiBall sensor (as-designed) and a proposed reduced-view HiBall. In this reduced-view HiBall, the center LEPD is preserved and the five perimeter LEPDs spaced at 72° apart have been reduced to three equally-spaced (120° apart) ones. Shifting the lenses in the same way, we now have 4 primary views for a total of 12 views if the secondary views are included.

Using the HiBall model, the *predicted* performance of a WFOV HiBall in both its current 6-LEPD configuration and a proposed reduced 4-LEPD configuration can be generated. We will again examine a horizontal plane at approximately head-height (1.9 meters) at zero rotation just as in Section 5.1.1.1. These predictions assume the same noise and



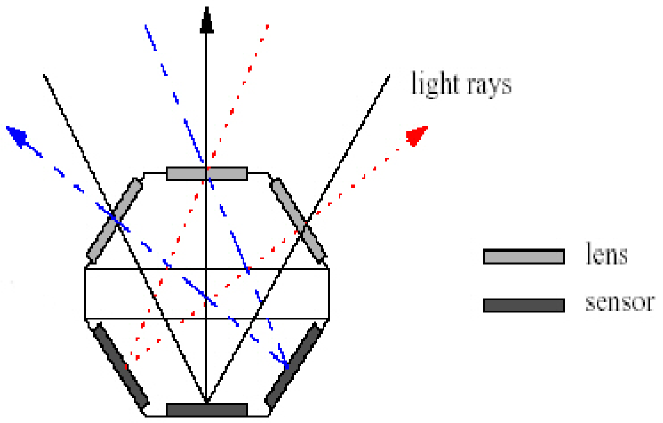


FIGURE 7.9. The HiBall Sensor LEPD Configuration [**106**]

motion models as the previous analysis in Section 5.1.1.1 and actual HiBall system but with twice the FOV as the current as-built HiBall sensor. The number of LEDs included at each sampled point was kept unchanged for ease of comparison.

7.4.1. Current 6-LEPD Configuration. Figure 7.11 shows Artemis' P^{∞} prediction for the performance of a WFOV HiBall sensor in the existing six LEPD configuration with both full- and half-ceilings. Interestingly and perhaps unintuitively, the prediction points to a slight deterioration in performance. The full-ceiling performance estimate value range is nearly identical to the waist-height performance estimate of the current HiBall (Figure 5.3). The waist-height analysis is at twice the distance from the ceiling as the head-height analysis, moving the ceiling further down the view frusta which would enable the HiBall sensor to see a wider patch of LEDs. This is the same patch of LEDs that would be visible at head-height with double the FOV.

The current HiBall system randomly selects an LED from a group of visible LEDs at each measurement update. By widening the FOV, the number of visible LEDs has indeed increased but the newly visible LEDs will be further away than those visible in 6° FOV effectively diluting the pool of visible LEDs with less desirable "far" ones. If the HiBall

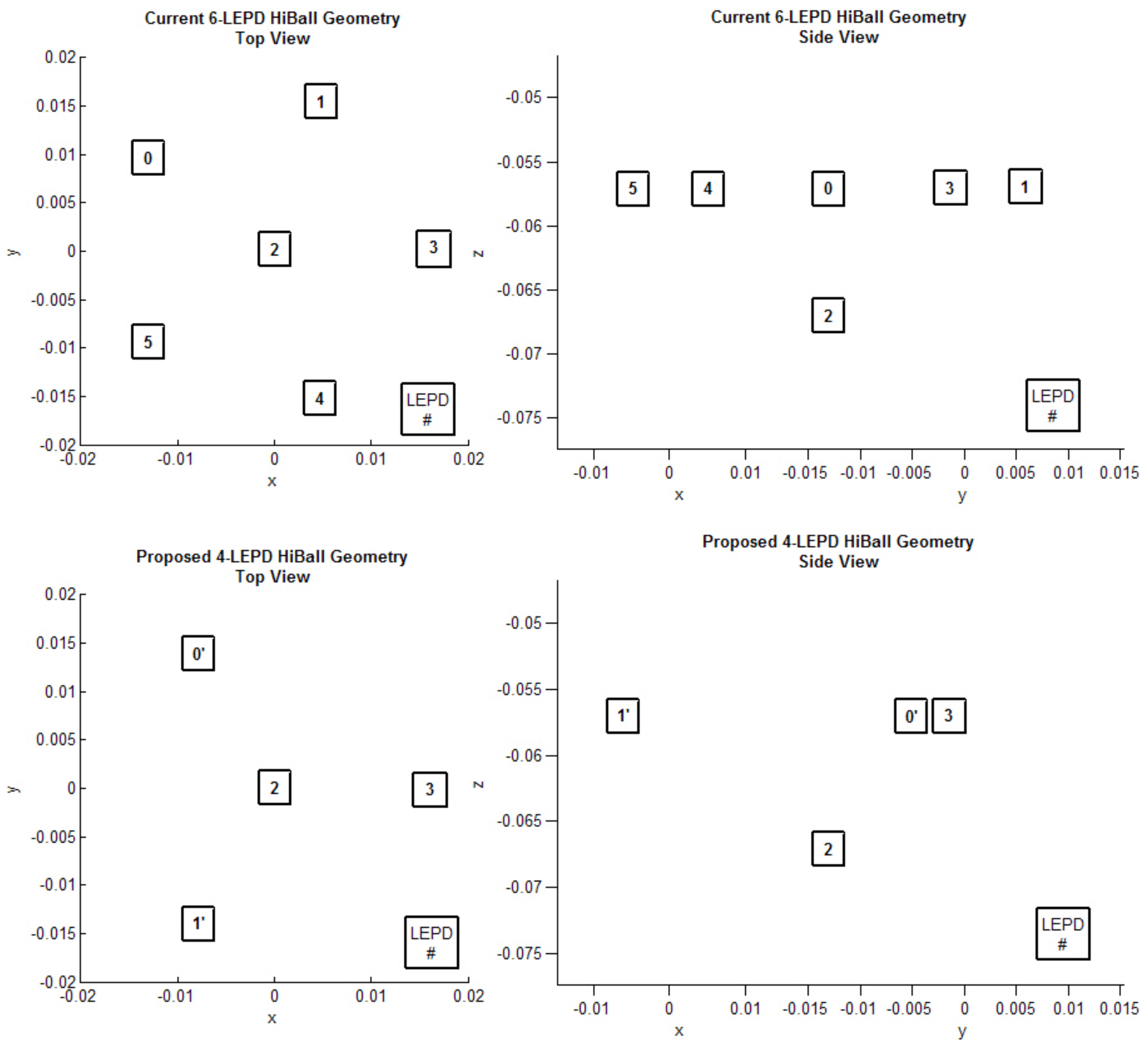


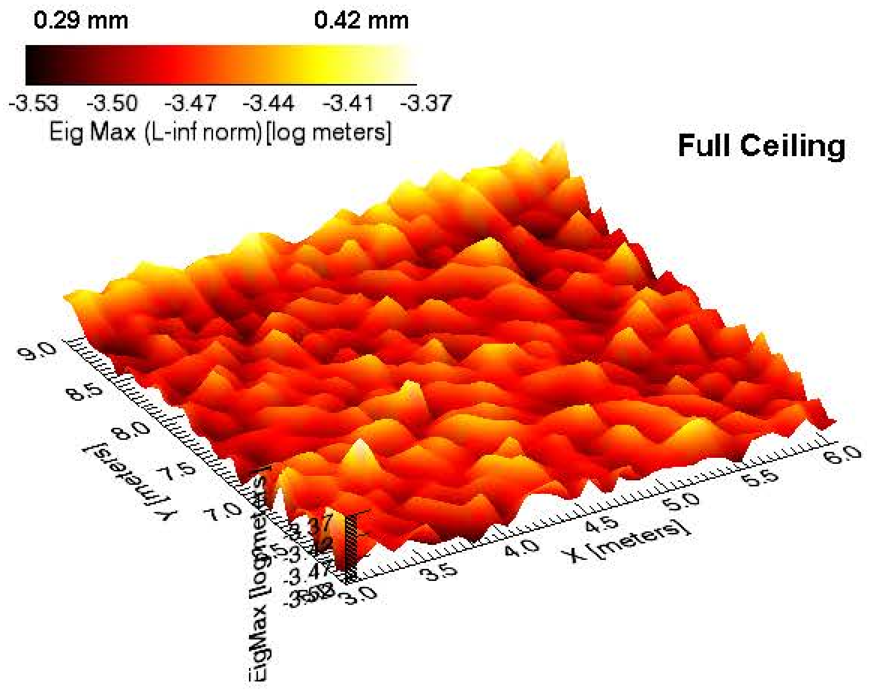
FIGURE 7.10. Current (6 LEPD on top) and Proposed (4 LEPD on bottom) HiBall LEPD configurations with sensor top view on the left and sensor side view on the right

software were modified to discriminate between the near and far visible LEDs, choosing the closer ones when available and the farther ones as needed, it might preserve the better performance in the central locations (under the ceiling) and improve performance towards the ceiling edges or in rotations where some of the views are occluded.

As expected, we find some degradation in WFOV performance from the full- to half-ceiling configuration and, again, we find that the real penalty is at the low end of the performance spectra. The best performance estimate remains at approximately -3.53 log meters (meters) but the performance bottoms out at -3.15 log meters as opposed to -3.37 log meters with the full ceiling. Closer inspection reveals that most of this low-end penalty is at the edge of the ceiling where the number of visible LEDs begins to decline. Performance in the more central positions is comparable between the full- and half-ceiling configurations with a bit more variation due to the ceiling rows of LEDs clearly visible in the half-ceiling performance estimate.

7.4.2. Proposed 4-LEPD Configuration. Figure 7.12 shows Artemis' P^{∞} estimate of a WFOV HiBall with reduced LEPDs (4 in total). In the full-ceiling configuration there is little penalty with respect to the current 6-LEPD configuration for reducing the number of LEPDs and views. We do see a significant penalty in the half-ceiling configuration at the ceiling edge with uncertainly on the order of 2 mm. In both configuration we observe a noticeable decline in performance at approximately 8 m in the y-direction probably due to the reduction of visible LEDs as the edge of the ceiling draws nearer.

Figure 7.13 shows the effect of reducing the number of views to primary only for a total of four views. Once again, we see little effect in the more central portions of the ceiling. However, the effect at the ceiling edge is marked but, overall, no worse than the performance at the ceiling edge in the half-ceiling configuration with secondary views (Figure 7.12).



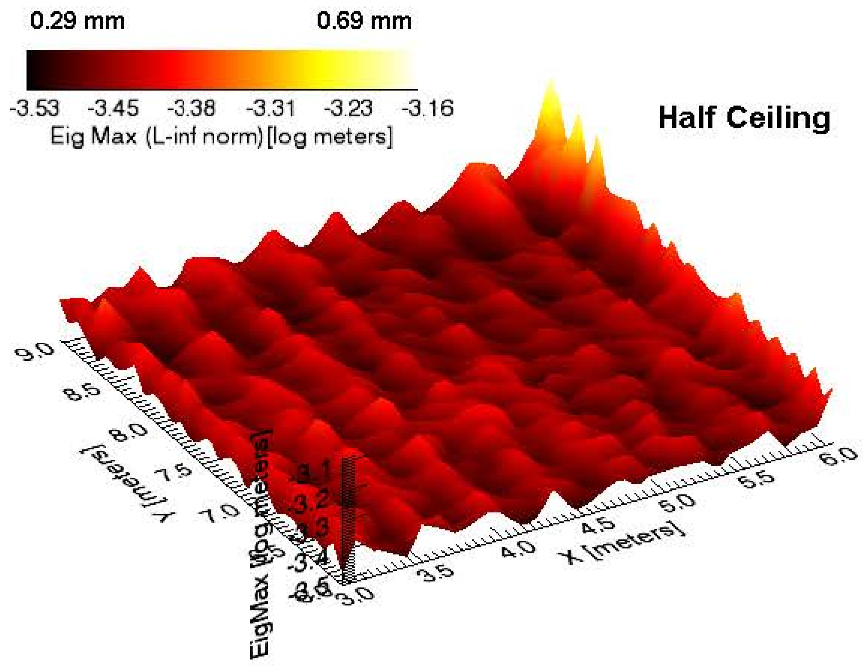
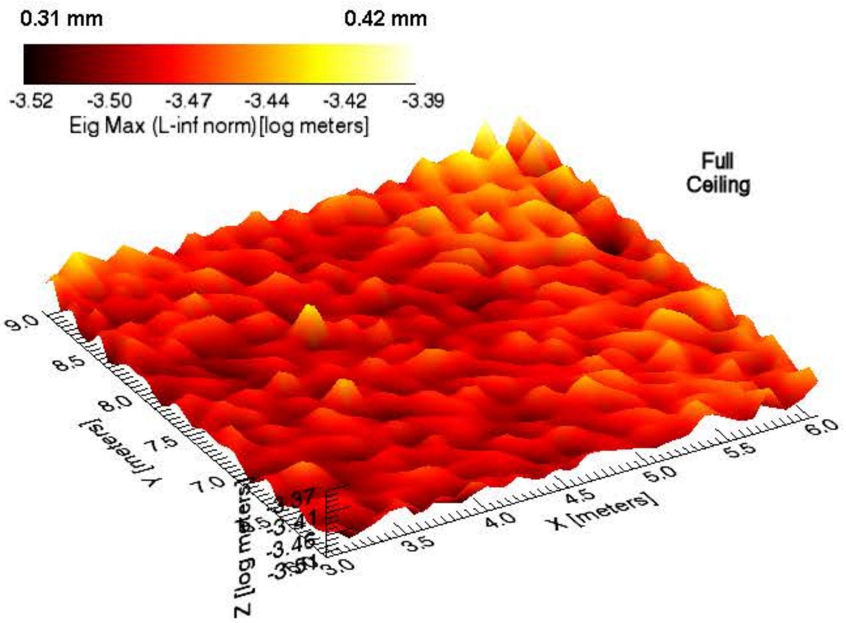


FIGURE 7.11. Full (top) and Half (bottom) Performance with current (6 LEPD) WFOV HiBall configuration at 1.9 meters



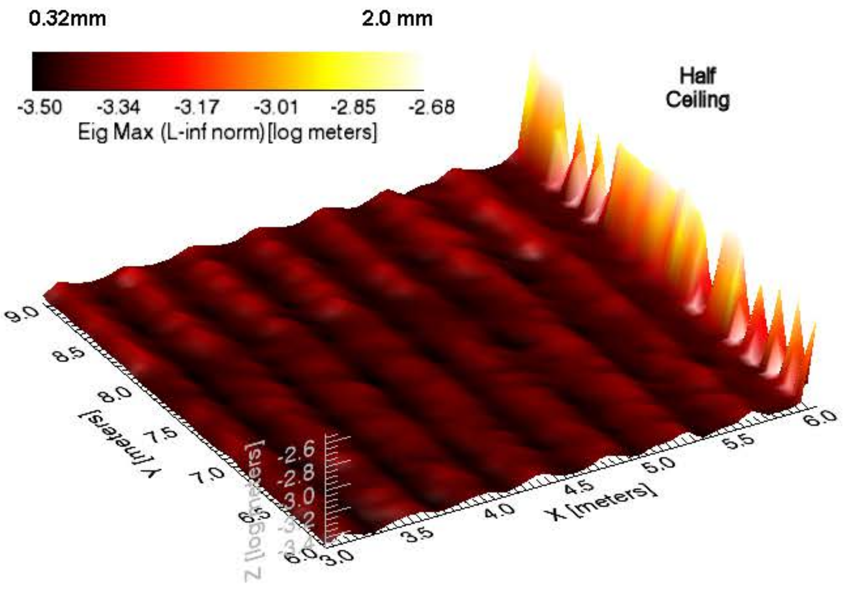
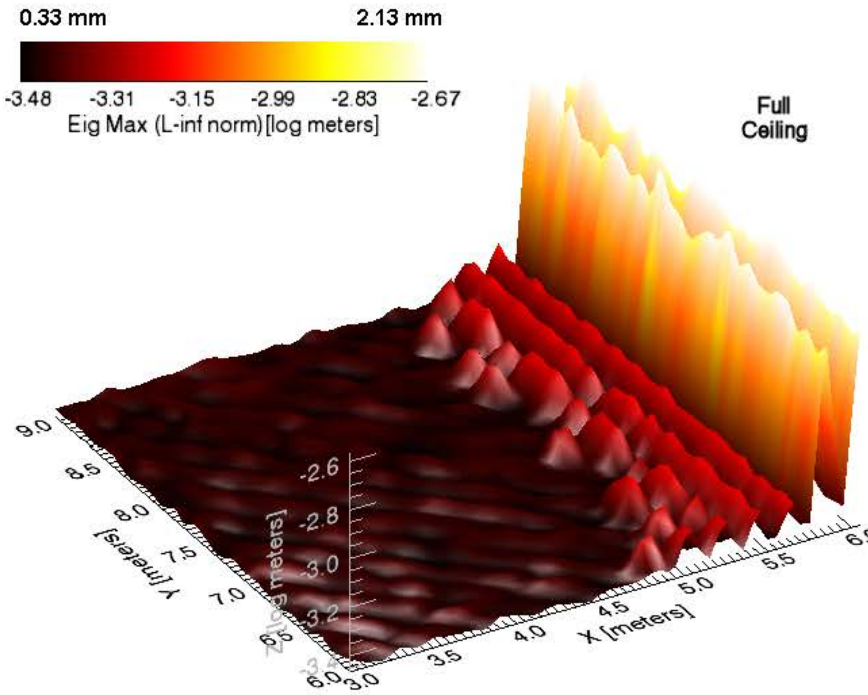


FIGURE 7.12. Full (top) and Half (bottom) Performance with proposed (4 LEPD) WFOV HiBall configuration using primary and secondary views at 1.9 meters



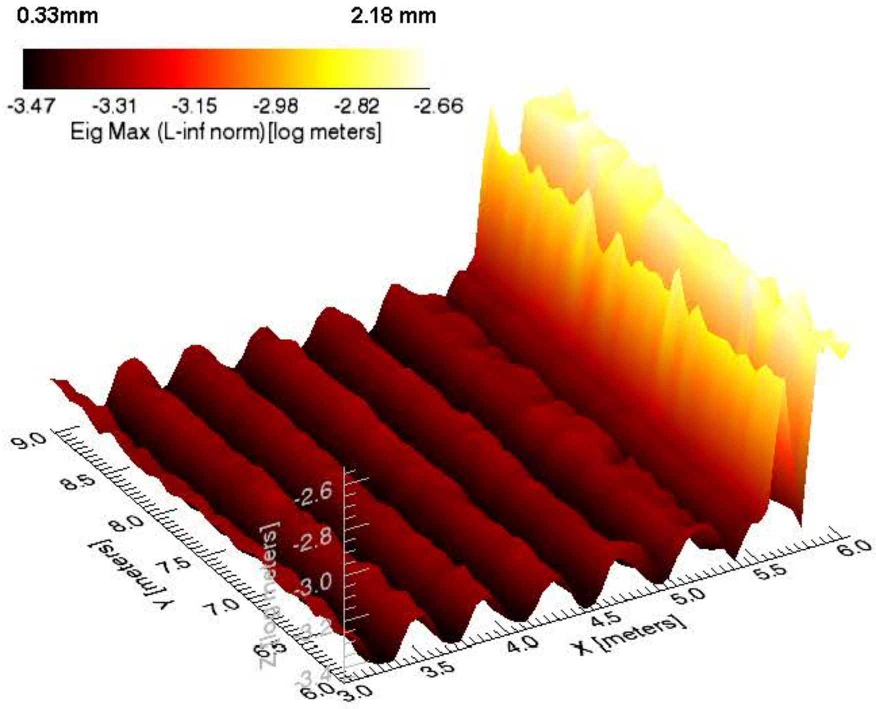


FIGURE 7.13. Full (left) and Half (right) Performance with proposed (4 LEPD) WFOV HiBall configuration using primary views only at 1.9 meters

CHAPTER 8

Future Work

8.1. The Framework

Currently, the P^{∞} method has been applied using evenly-spaced gridding or meshes. In CFD, mesh generation is a complex step in the computation process where the meshes are tighter (dense sampling) in areas of dynamic change and looser (less dense) in more static areas. These techniques could be applied to generate efficient meshes that support appropriate observed orders of accuracy. This would permit Artemis to take advantage of areas of continuity within any global analysis and sample as needed in areas of discontinuities such as where camera frusta overlap.

We would like to decrease P^{∞} computation time using parallel, adaptive, demand-driven computation [**69**] to improve system responsiveness for realtime interaction.

The "measurement-level analysis" presented in Section 6.2 was created and performed after the P^{∞} analysis had been completed and so is not included in the framework at this point. Further analysis is required to look at issues such as stability but, eventually, we would like to add this analysis capability to the framework for inclusion in Artemis or its next incarnation.

We would like to incorporate the generation and visualization of the derivatives of P^{∞} with respect to parameters such as FOV or rotation for sensitivity analysis.

Perhaps the most challenging part of creating a model for P^{∞} analysis is determining the motion model, Q. While the current Artemis system does provide default motion models, we would like to investigate a tool for the creation of user-defined motion models. Perhaps one that would enable a user to describe the motion in an intuitive way and then generate an appropriate Q.

Eventually we'd like to make a repository of process and measurement models to use as building blocks in system design and analysis.

8.2. Applications

We would like to continue applying the framework for more validation with real systems, especially those other than acoustic and optical, and use it in a design effort (as opposed to validation).

We would be grateful for the opportunity to work with 3rdTech, Inc. to see if any of the conclusions drawn here could be implemented and improve system performance. We are especially interested to know whether limiting the number of integration cycles as described in Section 6.2.1.2 could improve performance at longer distances from the ceiling.

We would like to apply Artemis (or the ideas embodied in it) to the computer vision domain. Device questions like how well light reflecting from a surface can be resolved using a camera and how well light can be targeted at a point on a surface using a projective device could be addressed and analyzed.

Just as projectors are used to actively inject signal rather than sense it, we would like to apply Artemis to control system analysis. The control of a robotic arm for example shares complementary physical characteristics: the state of the tip of the arm, and the uncertainty (information) with which it is placed. Just as one wants their tracking or computer vision system to be operating in the peak areas of sensor information as much as possible, one would want the tip of a robotic arm operating near information peak.

There is on-going research at NASA into rendezvous and docking systems. One example is Demonstration of Autonomous Rendezvous Technology (DART) [**44**] which determines the relative positions and attitudes between an active sensor and the passive target at ranges up to 300 meters. Brat et al. [**17**] are exploring how advanced V&V techniques, such as static analysis, model checking, and compositional verification, can be used to gain trust in model-based systems. I believe that Artemis could play a role in the modeling of these autonomous systems and NASA Langley Research Center (LaRC) management has expressed interest in its application.

8.3. New Ideas

At SIGGRAPH 2006, Raskar et al. [**81**] presented a method for handling motion blur caused by moving objects in a single exposure photograph. Assuming an object with constant speed (a first-order motion model), the "flutter" the camera's shutter open and closed at random intervals rather than hold the shutter open during the exposure time. This allows for the recovery of the moving and stationary objects in the scene using deconvolution. It would be interesting to see whether something like this could be applied to the HiBall system. If we could back out the smear or blur caused by the target moving during a δt, the result could be longer LEPD dwell/integration times and better overall performance.

8.4. Statistical Power Analysis

What follows is work for The HiBall System Case Study Chapter 7 that was not mature enough to defend and include in the body of the dissertation. However, a good bit of thought and analysis has been accomplished and is included here as future work so that the idea is not lost.

8.4.1. Z-test. The central limit theorem tells us that, given a sufficient number of samples ($N \to \infty$) of a normally-distributed signal with known mean and variance, the distribution of the sample will approach a normal distribution with the same mean and variance as the signal [**12**]. Since we cannot sample infinitely, a question often asked in statistical analysis is how many samples (N) are necessary to achieve a sampled representation of a signal (or a population) with the same variance and mean within some delta of the known variance and mean. This difference between the two means is called *effect size*. *Statistical power* *analysis* [**27**] provides a method for estimating the required number of measurements, effect size, mean, variance, and confidence intervals (α and β) using the statistical Z-test. The Z-test is a statistical test which determines if the difference between a sample mean and the population mean (i.e. the expected signal mean in this case) is large enough to be statistically significant. We test this stating a hypothesis such as "sample mean equals population mean" or "sample mean less than population mean".

8.4.2. Number Samples. How many data samples are required so that the distribution of sampled data will match that of the expected signal distribution within an acceptable margin of error? In this case, the hypothesis would be that the expected signal mean and the sampled mean are equal such that $\mu_0 = \mu_1$ where μ_0 is the expected mean and μ_1 is the sample mean. α is the risk of assuming that there is a difference between the two means when in fact there is not (a false positive) and β is the risk of concluding that a difference does not exist when in fact a difference does exist (a false negative). β determines power, as in power analysis, and power is defined as $1-\beta$. Typical values for α and $1-\beta$ (power) are 0.05 and 0.80, respectively. This means that the probability of a false positive is 5% and the probability of a false negative is 20% (i.e. β). In the literature, the power analysis equation is often arranged in order to calculate the required sample size, N, where [**63**]

$$N = \frac{\sigma^2 (Z_{\alpha/2} + Z_{\beta})^2}{(\mu_0 - \mu_1)^2}. \tag{58}$$

Note that instead of α and β, Equation (58) contains $Z_{\alpha/2}$ and Z_{β}, the critical values for α and β for the statistical Z-test. These critical values can be determined by look-up and a standard normal distribution (Z-test) table is included in the appendix (Section D).

8.4.3. Effect Size. Given known mean, variance, sample size and confidence intervals ($\alpha = 0.05$ and $\beta = 0.20$), we can determine how much deviation from the "truth" mean we can expect throughout a range of sampling rates. The effect size ($ES = \mu_0 - \mu_1$) is simply a measure of the difference between the known mean and the mean that we will measure. Equation (58) designates N as the dependent variable but with simple algebra we can calculate the effect size instead as

$$ES = \mu_0 - \mu_a = \sqrt{\frac{\sigma^2(Z_{\alpha/2} + Z_\beta)^2}{N}}. \tag{59}$$

8.4.4. HiBall$^{\text{TM}}$ Power Analysis. During a dark-light-dark cycle, the HiBall takes a single sample ($N = 1$) with expected variance (σ^2) over a varying time, δt, which is a function of the number of integration cycles, i, such that $\delta t = f(i)$. Despite knowing the signal variance, we cannot estimate the exact value of a single measurement of a random signal. Any single measurement could be the mean (μ), one sigma (σ) or many sigmas away from the expected value of the signal (i.e. the mean, μ). In the current operation of the HiBall system, a single measurement is taken per measurement update and while this measurement value is probably within one sigma, we might be able to acquire a better representation of the normal distribution of the signal (and thus the mean) with multiple samples. Will the expected performance improve if i individual samples were averaged in place of a single i-duration clock cycle sample even though we know that the shorter samples will be noisier (Section 6.2.1.2)?

In the case of the HiBall system, we know that $N = 1$ regardless of integration time. To simplify analysis, we can choose an LED directly above the HiBall sensor so that the centroid of light will fall directly in the center of the center LEPD and so the mean (μ_0) is zero (recall that the LEPD sensor readings are [-1,1]. The data shown in Figure 7.7 is the noise in mm we expect for the centroid of light on the HiBall LEPD for a given sample/integration time, $\delta t = f(i)$.

The values of $\alpha = 0.05$ and $1 - \beta = 0.80$ correspond (by lookup) to Z critical values of 1.96 and 0.84 respectively such that $Z_{\alpha/2} + Z_\beta)^2 \simeq 7.9$. We use $(Z_{\alpha/2}$ because we want to know whether the signal and sampled means are *equal* as opposed to a directional operator such as $\leq$ or $>$ in which case we would use Z_α. Given these critical values, we can calculate the effect size for a range of δt values as shown in the upper (solid blue) curve labeled "Effect Size for N=1" in Figure 8.1.

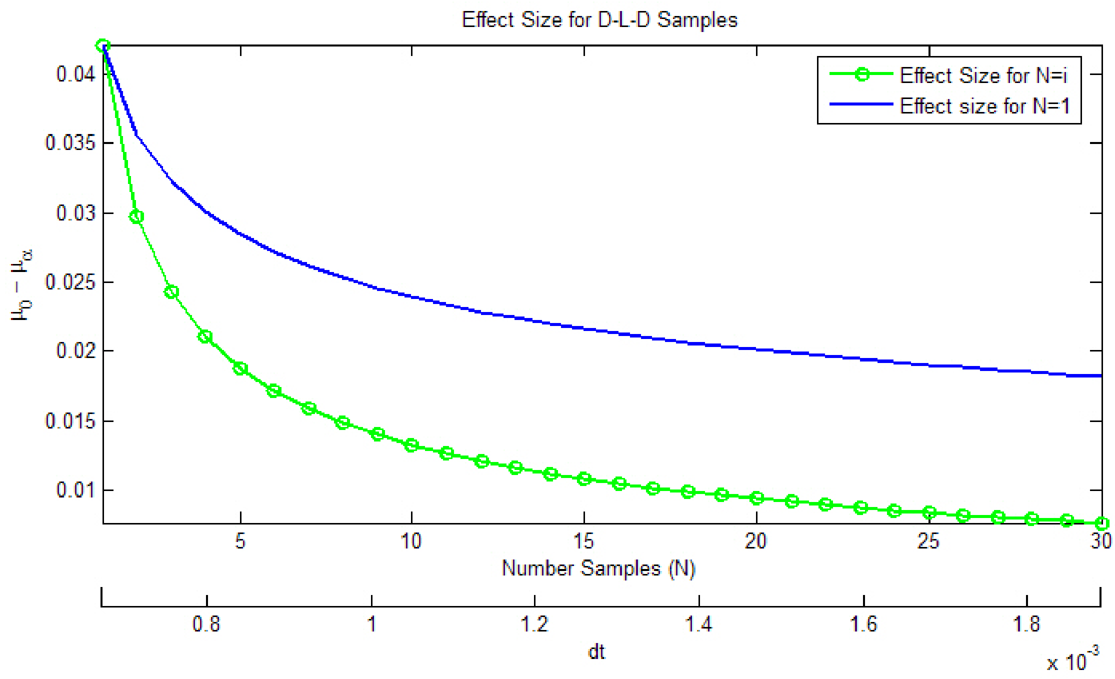


FIGURE 8.1. Effect Size per HiBall Dark-Light-Dark Sample Period from two-sided statistical power analysis

Here we find that the effect size ranges from 0.018 to 0.042 or from 0.9% to 2.1% of the LEPD output range of 2 $(-1:1)$. If we limit our findings to the region below the $Q-R$ crossover point, we find that the effect size minimum increases to 2.52 or 1.2% of the LEPD range.

We can use statistical power analysis to predict the effect size if, instead of integrating over the entire light cycle for a single measurement, we averaged i measurements over the light cycle. For example, if $i=2$, we can measure two light-dark-light cycles of length i in a single δt. To predict the effect size of this change, we use the calculated variance of a single light-dark-light cycle ($\sigma^2 = 4.22e-005$) to compute the effect size at $N=2$.

Calculating for a range of N, we find that the effect size decreases as shown in the lower (circle-marked green) curve labeled “Effect Size for N=i” in Figure 8.1.

Even with the penalty of the higher variance for a one-cycle integration time, we observe that effect size decreases to a minimum of 0.007 (0.38%) as the number of samples per δt increases. This suggests that the HiBall system might operate with greater accuracy by acquiring multiple light-dark-light samples per filter iteration (δt) in order to achieve a more statistically normal measurement. The actual N that could be used will be limited by the physical constraints of the hardware itself.

8.4.4.1. *Number of Samples.* Using Equation (58), we can also solve for the number of samples required for the two means to be separated by some predetermined effect size. If a 10% effect size (± 0.2) is acceptable then a single sample ($N = 1$) is all that is required. A 1% effect size increases N to 9 samples for $i = 1$ and 3 samples for $i = 15$. Recall that the signal variance (i.e. noise) decreases as i increases and so fewer samples are required for the same effect size when $i = 15$ than when $i = 1$. Were we able to integrate for 30 clock cycles, only 2 samples would be required. An effect size equivalent to 0.1% requires hundreds of samples regardless of the chosen integration time and so is currently not achievable. Figure 8.2 shows the number of samples required for HiBall integration clock cycle counts of $1 \leq i \leq 30$ for various net effect sizes ranging from 0.5% to 5.0%. Note that only values to the to the left of 15 integration cycles on the horizontal axis are achievable by the current HiBall system. Therefore, effect sizes from 5.0% down to 1.0% are achievable and 0.5% is achievable for $i \geq 2$.

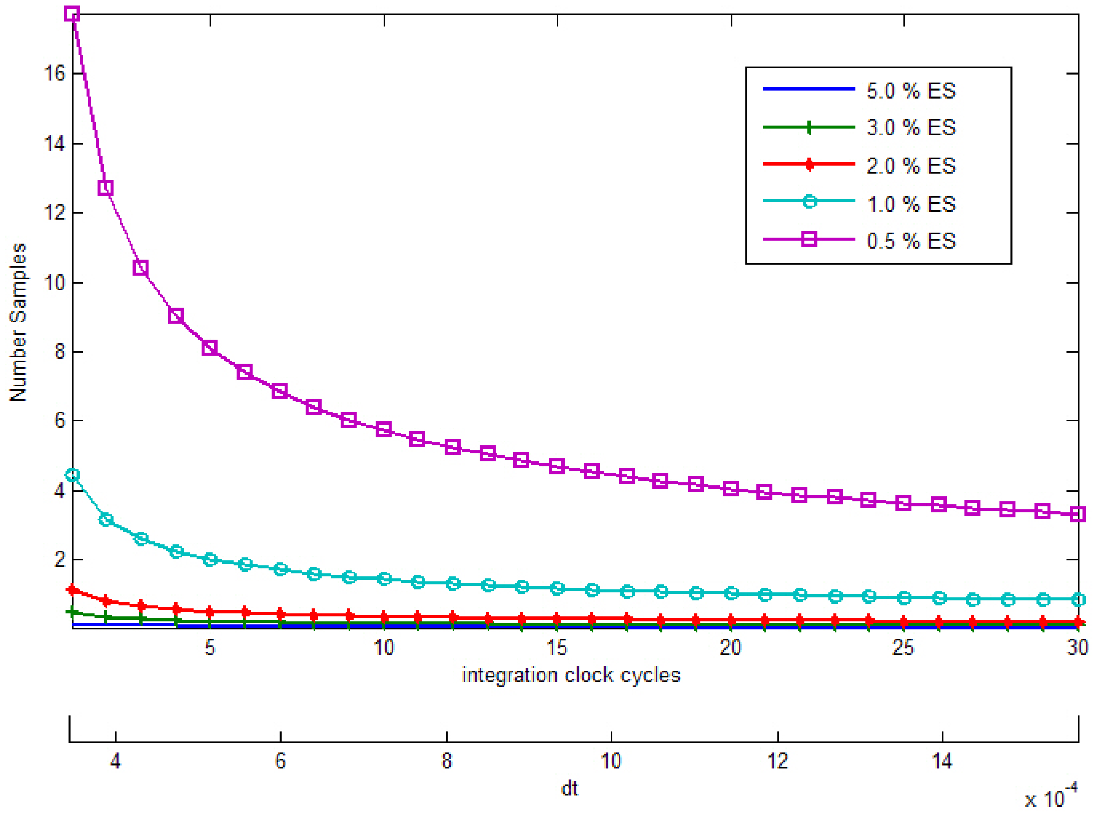


FIGURE 8.2. Number Samples required per HiBall Dark-Light-Dark Sample Period for various net effects

APPENDIX A

Motion Capture Data from Chapter Three (120 fps)

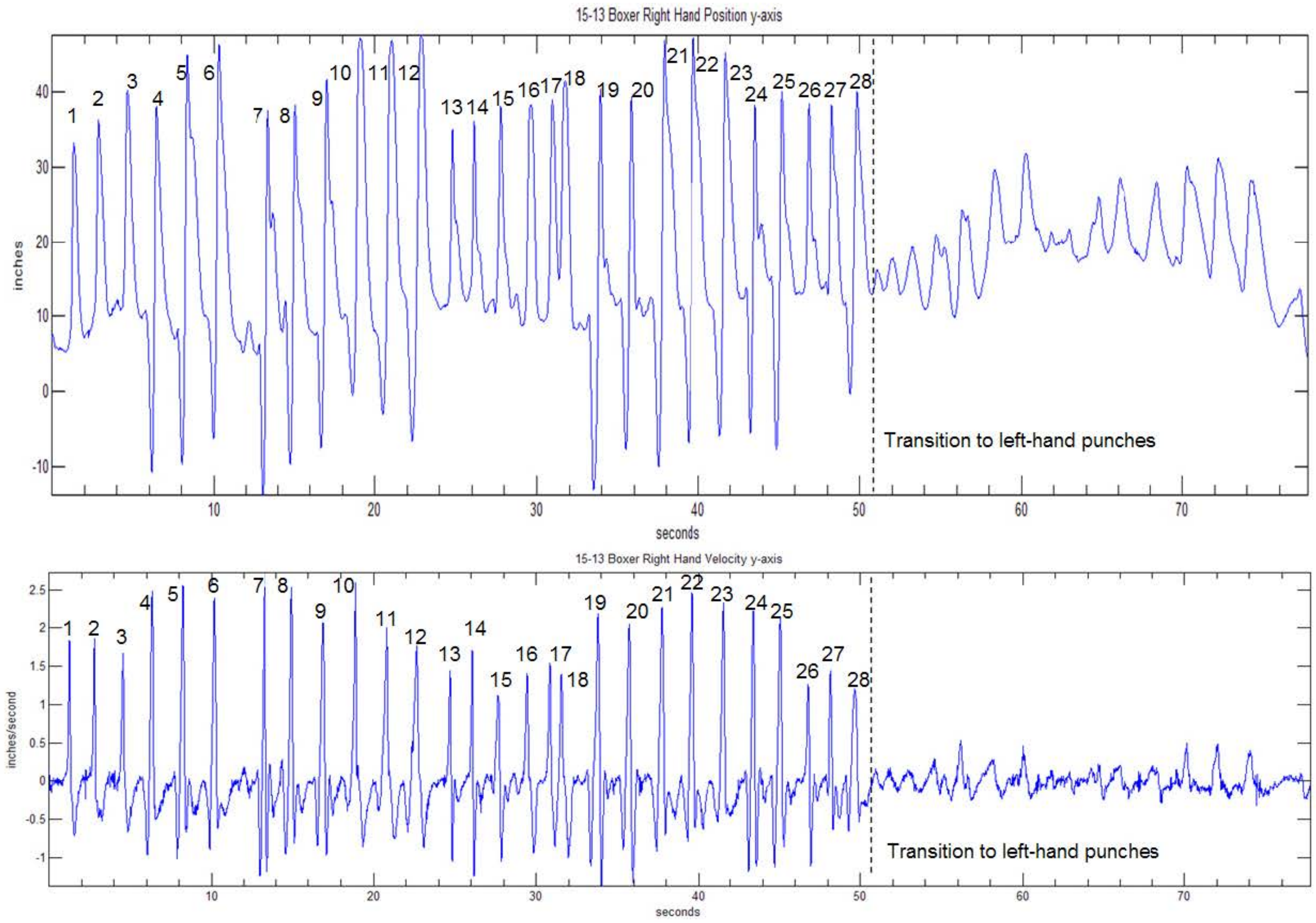


FIGURE A.1. Boxing motion capture right hand y-axis position (top) and velocity (bottom). Note different vertical scale and units.

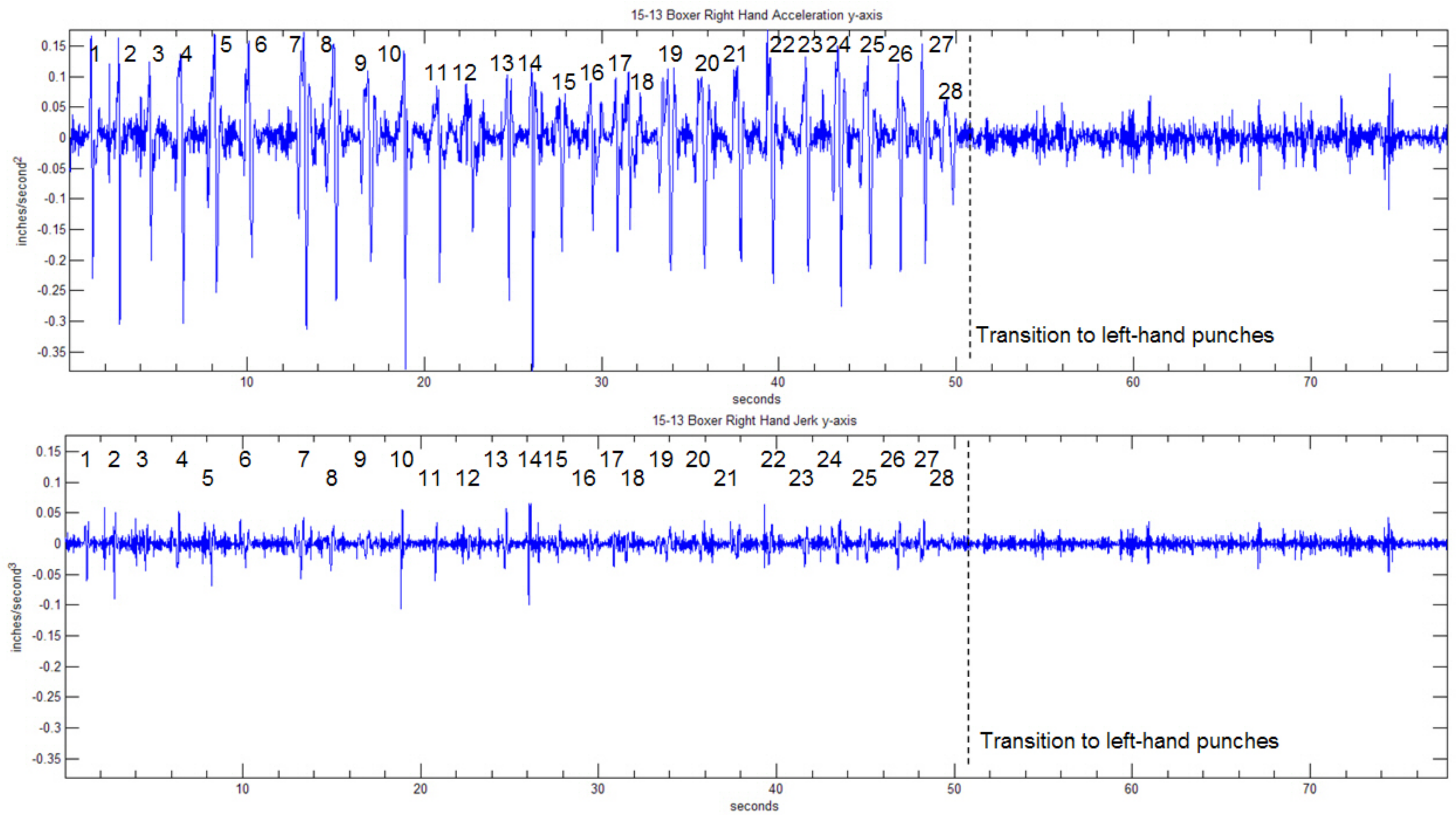


FIGURE A.2. Boxing motion capture right hand y-axis acceleration (top) and jerk (bottom). Note different vertical units.

APPENDIX B

Motion Capture Frames from Boxer MPG Video (30 fps)

Frame 1273

Frame 1274

Frame 1275

Frame 1276

Frame 1277

Frame 1278

Frame 1279

Frame 1280

Frame 1281

Frame 1282

Frame 1283

Frame 1284

Frame 1285

Frame 1286

Frame 1287

Frame 1288

Frame 1289

Frame 1290

Frame 1291

Frame 1292

Frame 1293

Frame 1294

Frame 1295

Frame 1296

Frame 1297

Frame 1298

Frame 1299

Frame 1300

Frame 1301

Frame 1302

Frame 1303

Frame 1304

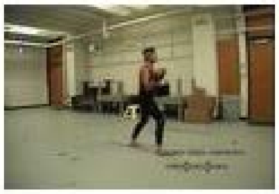
Frame 1305

Frame 1306

Frame 1307

Frame 1308

Frame 1309

Frame 1310

Frame 1311

Frame 1312

Frame 1313

Frame 1314

Frame 1315

Frame 1316

Frame 1317

Frame 1318

Frame 1319

Frame 1320

Frame 1321

Frame 1322

Frame 1323

Frame 1324

Frame 1325

Frame 1326

Frame 1327

Frame 1328

Frame 1329

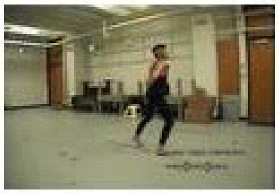
Frame 1330

Frame 1331

Frame 1332

Frame 1333

Frame 1334


Frame 1335

Frame 1336

Frame 1337

Frame 1338

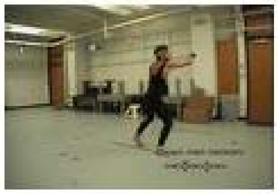
Frame 1339

Frame 1340

Frame 1341

Frame 1342

Frame 1343


Frame 1344

Frame 1345

Frame 1346

Frame 1347

Frame 1348

Frame 1349

Frame 1350

Frame 1351

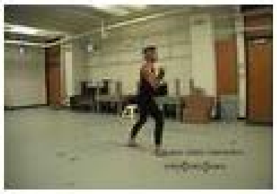
Frame 1352

Frame 1353

Frame 1354

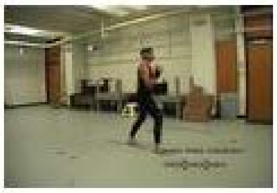
Frame 1355

Frame 1356

Frame 1357

Frame 1358

Frame 1359

Frame 1360

Frame 1361

Frame 1362

Frame 1363

Frame 1364

Frame 1365
Frame 1366
Frame 1367
Frame 1368

Frame 1369

Frame 1370

Frame 1371

Frame 1372

Frame 1373

Frame 1374

Frame 1375

Frame 1376

Frame 1377

Frame 1378

Frame 1379

Frame 1380

Frame 1381

Frame 1382

Frame 1383

Frame 1384

Frame 1385

Frame 1386

Frame 1387

Frame 1388

Frame 1389

Frame 1390

Frame 1391

Frame 1392

Frame 1393

Frame 1394

Frame 1395

Frame 1396

Frame 1397

Frame 1398

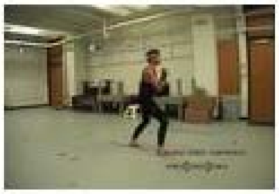
Frame 1399

Frame 1400

Frame 1401

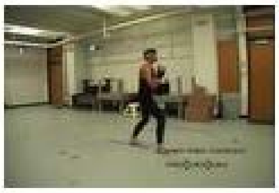
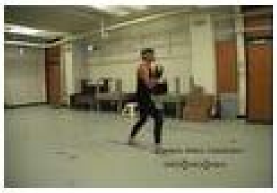
Frame 1402

Frame 1403

Frame 1404

Frame 1405

Frame 1406

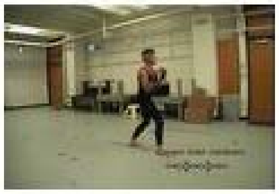
Frame 1407

Frame 1408

Frame 1409

Frame 1410

Frame 1411

Frame 1412

Frame 1413

Frame 1414

Frame 1415

Frame 1416

Frame 1417

Frame 1418

Frame 1419

Frame 1420

Frame 1421

Frame 1422

Frame 1423

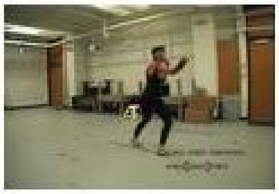
Frame 1424

Frame 1425

Frame 1426

Frame 1427

Frame 1428


Frame 1429

Frame 1430

Frame 1431

Frame 1432

Frame 1433

Frame 1434

Frame 1435

Frame 1436

Frame 1437

Frame 1438

Frame 1439

Frame 1440

Frame 1441

Frame 1442

Frame 1443

Frame 1444

Frame 1445

Frame 1446

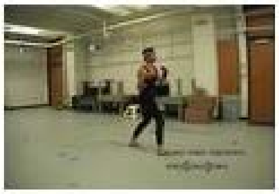
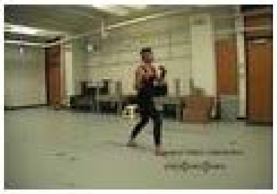
Frame 1447

Frame 1448

Frame 1449

Frame 1450

Frame 1451

Frame 1452

Frame 1453

Frame 1454

Frame 1455

Frame 1456

Frame 1457

Frame 1458

Frame 1459

Frame 1460

Frame 1461

Frame 1462

Frame 1463

Frame 1464

Frame 1465

Frame 1466

Frame 1467

Frame 1468

Frame 1469

Frame 1470

Frame 1471

Frame 1472

Frame 1473

Frame 1474

Frame 1475

Frame 1476

Frame 1477

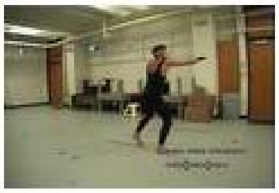
Frame 1478

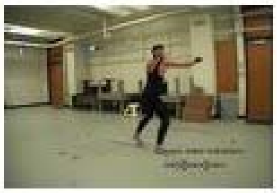
Frame 1479

Frame 1480

Frame 1481

Frame 1482

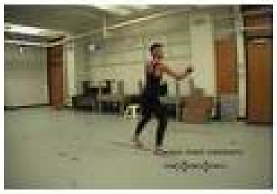

Frame 1483


Frame 1484

Frame 1485

Frame 1486

Frame 1487


Frame 1488

Frame 1489

Frame 1490

Frame 1491

Frame 1492

Frame 1493

Frame 1494

Frame 1495

Frame 1496

Frame 1497

Frame 1498

Frame 1499

Frame 1500

Frame 1501

Frame 1502

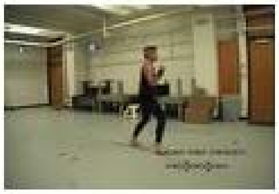
Frame 1503

Frame 1504

Frame 1505

Frame 1506

Frame 1507

Frame 1508


Frame 1509

Frame 1510

Frame 1511

Frame 1512

Frame 1513

Frame 1514

Frame 1515

Frame 1516

Frame 1517

Frame 1518

Frame 1519

Frame 1520

Frame 1521

Frame 1522

Frame 1523

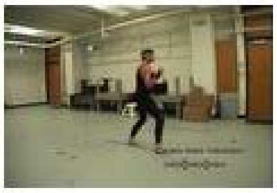
Frame 1524

Frame 1525

Frame 1526

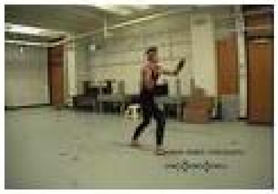
Frame 1527

Frame 1528

Frame 1529

Frame 1530

Frame 1531

Frame 1532



Frame 1533
Frame 1534
Frame 1535
Frame 1536

Frame 1537

Frame 1538

Frame 1539

Frame 1540

Frame 1541

Frame 1542

Frame 1543

Frame 1544

Frame 1545

Frame 1546

Frame 1547

Frame 1548

Frame 1549

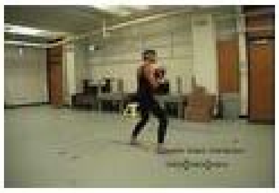
Frame 1550

Frame 1551

Frame 1552

Frame 1553

Frame 1554



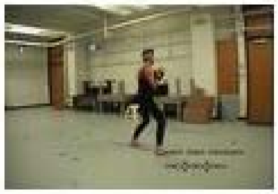
Frame 1555

Frame 1556

Frame 1557

Frame 1558

Frame 1559



Frame 1560

Frame 1561

Frame 1562

Frame 1563

Frame 1564

Frame 1565

Frame 1566

Frame 1567

Frame 1568

Frame 1569

Frame 1570

Frame 1571

Frame 1572

Frame 1573

Frame 1574

Frame 1575

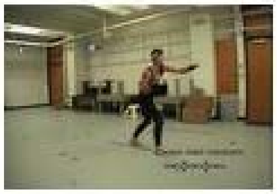
Frame 1576

Frame 1577

Frame 1578

Frame 1579

Frame 1580



Frame 1581
Frame 1582
Frame 1583
Frame 1584

Frame 1585

Frame 1586

Frame 1587

Frame 1588

Frame 1589

Frame 1590

Frame 1591

Frame 1592

Frame 1593

Frame 1594

Frame 1595

Frame 1596

Frame 1597

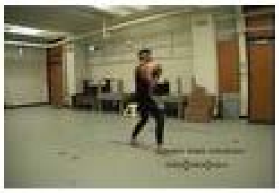
Frame 1598

Frame 1599

Frame 1600

Frame 1601

Frame 1602


Frame 1603

Frame 1604

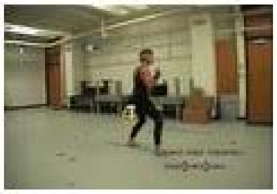
Frame 1605

Frame 1606

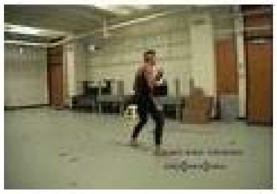
Frame 1607

Frame 1608

Frame 1609


Frame 1610

Frame 1611


Frame 1612

Frame 1613

Frame 1614

Frame 1615

Frame 1616

Frame 1617

Frame 1618

Frame 1619

Frame 1620

Frame 1621

Frame 1622

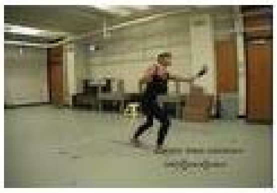
Frame 1623

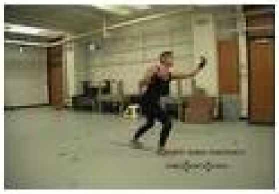
Frame 1624

Frame 1625

Frame 1626

Frame 1627


Frame 1628



Frame 1629

Frame 1630

Frame 1631

Frame 1632

APPENDIX C

HiBall Noise Model [25] in Matlab

```
clear syms v0 f0 f tau T T0 positive

Rx_tau = (pi*v0^2*f0)*exp(-2.0*pi*f0*abs(tau));

var_Dx1 = 2.0*Rx_tau*(T-tau);       % 1st integral in var_Dx
var_Dx2 = Rx_tau*(tau-T0);          % 2nd integral in var_Dx
var_Dx3 = Rx_tau*(T0+2.0*T-tau);    % 3rd integral in var_Dx



D1=1/2*v0^2*(1+2*exp(T*pi*f0)^2*pi*f0*T-exp(T*pi*f0)^2)/exp(T*pi*f0)^2/pi/f0;
D2=1/4*v0^2*(-2*T*pi*f0-1+exp(T*pi*f0)^2)/exp(pi*f0*T0)^2/exp(T*pi*f0)^2/pi/f0;
D3=1/4*v0^2*(1+2*exp(T*pi*f0)^2*pi*f0*T-exp(T*pi*f0)^2)/exp(pi*f0*T0)^2/exp(T*pi*f0)^4/pi/f0;

var_Dx_sym  = (2.0/T^2)*( D1-D2-D3);

x=0.5; phi=60.0;ds=3.0;rf=2.0e5;vn=4.5e-9;in=6e-13;f0=1.6e5;T0=2e-5; %from the TM
%x=0.5; phi=0.0;ds=1.0;rf=2.0e5;vn=4.5e-9;in=6e-13;f0=1.6e5;T0=2e-5; %1m directly below an LED
%f0 = 0.0;

dt_array=20:20:2000;
% insert a number very close to zero as first independent variable in time
%dt_array = [0.001 dt_array];
dt_array=1e-6*dt_array;

X_est_array = zeros(length(dt_array),2);

[v1,v2] = calc_v (x, phi, ds, rf, vn, in);

[v1d,v2d] = calc_v (-1.0,0.0, 1.0, rf, vn, in);
```

```
S1 = v1.v1s; S2 = v2.v2s;

S1 = S1 - v1d.v1s; S2 = S2 - v2d.v2s;

firstNaN = 0;

for t=1:length(dt_array)
    T=dt_array(t);
    v0=v1.v1n;
    % precision problems with very large exp values so use "vpa"
    N1 = subs(expand(vpa(var_Dx_sym,64))); %var_Dxs(v1.v1n,f0,T0,T,Rx_tau);
    N1 = sqrt(subs(N1));
    v0=v2.v2n;
    N2 = subs(expand(vpa(var_Dx_sym,64))); %var_Dxs(v2.v2n,f0,T0,T,Rx_tau);
    N2 = sqrt(subs(N2));

    X_s = (S1-S2)/(S1+S2);
    X_n = 2.0 * (sqrt((N1*S2)^2 + (2.0*N1*S2*N2*S1) + (N2*S1)^2)) / (S1+S2)^2 ;

    X = [ X_s, X_n ];
    X_est_array(t,:)=X;

end

format long g
fprintf('X_est = %g = 0.001751273 ? \n',X_est_array(5,2));

figure; plot(dt_array,X_est_array(:,2));

xlabel('dt'); ylabel('X_n');

title('HiBall RMS uncertainty of estimated position');
```

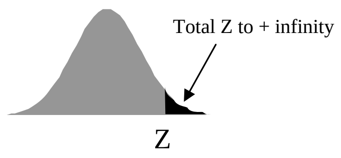
```
% Calculate and show preamp output signal and noise for various parameters
% function [] = vo(x, phi, ds, rf, vn, in, f0)
function [v1,v2] = vo(x, phi, ds, rf, vn, in, f0)

fprintf('============= vo =================\n\n');

[v1,v2] = calc_v (x, phi, ds, rf, vn, in);

[v1,v2] = show_v(f0, v1, v2) ; fprintf('\n');

% Examples
% vo(-1.0,  0.0, 1.0, 2.0e5 4.5e-9 6.0-13 1.6e5)    % dark current
% vo( 0.5, 60.0, 3.0, 2.0e5 4.5e-9 6.0-13 1.6e5)    % 3.0 [m], 60 [deg]
% vo( 0.0,  0.0, 0.6, 2.0e5 4.5e-9 6.0-13 1.6e5)    % 0.6 [m],  0 [deg]
% Display values of signal and noise for bandwidth f0 or spot noise if f0=0

function [v1,v2] = show_v(f0, v1, v2)


fprintf('============== show_v ================\n\n');


fprintf('at f = %g Hz ', f0);


% show_v

if f0==0

    f0 = 1.0;

else

    f0 = 0.5*pi*sqrt(f0);

end


v1.v1s      = v1.v1s;


v1.v1n      = v1.v1n*f0;


v1.v1sh     = v1.v1sh*f0; v1.v1vn1    = v1.v1vn1*f0;


v1.v1in1    = v1.v1in1*f0;


v1.v1iti    = v1.v1iti*f0;


v1.v1vtf1   = v1.v1vtf1*f0; v1.v1vn2    = v1.v1vn2*f0;


v2.v2s      = v2.v2s; v2.v2n      = v2.v2n*f0;


v2.v2sh     = v2.v2sh*f0;


v2.v2vn2    = v2.v2vn2*f0; v2.v2in2    = v2.v2in2*f0; v2.v2iti    =

v2.v2iti*f0; v2.v2vtf2   = v2.v2vtf2*f0;
```

```
v2.v2vn1    = v2.v2vn1*f0;

fprintf('(spot noise equivilent at f = %g Hz): \n\n', f0);

fprintf('v1s    = %g\n'    ,v1.v1s);
fprintf('v1n    = %g\n'    ,v1.v1n);
fprintf('v1sh   = %g\n'   ,v1.v1sh);
fprintf('v1vn1  = %g\n'  ,v1.v1vn1);
fprintf('v1in1  = %g\n'  ,v1.v1in1);
fprintf('v1iti  = %g\n'  ,v1.v1iti);
fprintf('v1vtf1 = %g\n' ,v1.v1vtf1);
fprintf('v1vn2  = %g\n'  ,v1.v1vn2);

fprintf('\n');

fprintf('v2s    = %g\n'    ,v2.v2s);
fprintf('v2n    = %g\n'    ,v2.v2n);
fprintf('v2sh   = %g\n'   ,v2.v2sh);
fprintf('v2vn2  = %g\n'  ,v2.v2vn2);
fprintf('v2in2  = %g\n'  ,v2.v2in2);
fprintf('v2iti  = %g\n'  ,v2.v2iti);
fprintf('v2vtf2 = %g\n' ,v2.v2vtf2);
fprintf('v2vn1  = %g\n'  ,v2.v2vn1);

return;
```

APPENDIX D

The Standard Normal Distribution (Z-Statistic)

Values α in the body of the table are the probability that z is greater than the positive value z_α given in the left and top margins. For example, to find $z_{0.025}$, we locate 0.025 in the table (circled) and find that it corresponds to 1.90 in the left column and 0.06 in the upper row. Summing these two number, we get 1.96, the critical value for $z_{0.025}$. Locating 0.20 in the table and summing the values found in the left and top margins returns the values for $z_{0.200} = 0.84$.



Z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.50000	0.49601	0.49202	0.48803	0.48405	0.48006	0.47608	0.47210	0.46812	0.46414
0.1	0.46017	0.45620	0.45224	0.44828	0.44433	0.44038	0.43644	0.43251	0.42858	0.42465
0.2	0.42074	0.41683	0.41294	0.40905	0.40517	0.40129	0.39743	0.39358	0.38974	0.38591
0.3	0.38209	0.37828	0.37448	0.37070	0.36693	0.36317	0.35942	0.35569	0.35197	0.34827
0.4	0.34458	0.34090	0.33724	0.33360	0.32997	0.32636	0.32276	0.31918	0.31561	0.31207
0.5	0.30854	0.30503	0.30153	0.29806	0.29460	0.29116	0.28774	0.28434	0.28096	0.27760
0.6	0.27425	0.27093	0.26763	0.26435	0.26109	0.25785	0.25463	0.25143	0.24825	0.24510
0.7	0.24196	0.23885	0.23576	0.23270	0.22965	0.22663	0.22363	0.22065	0.21770	0.21476
0.8	0.21186	0.20897	0.20611	0.20327	0.20045	0.19766	0.19489	0.19215	0.18943	0.18673
0.9	0.18406	0.18141	0.17879	0.17619	0.17361	0.17106	0.16853	0.16602	0.16354	0.16109
1.0	0.15866	0.15625	0.15386	0.15151	0.14917	0.14686	0.14457	0.14231	0.14007	0.13786
1.1	0.13567	0.13350	0.13136	0.12924	0.12714	0.12507	0.12302	0.12100	0.11900	0.11702
1.2	0.11507	0.11314	0.11123	0.10935	0.10749	0.10565	0.10383	0.10204	0.10027	0.09853
1.3	0.09680	0.09510	0.09342	0.09176	0.09012	0.08851	0.08691	0.08534	0.08379	0.08226
1.4	0.08076	0.07927	0.07780	0.07636	0.07493	0.07353	0.07215	0.07078	0.06944	0.06811
1.5	0.06681	0.06552	0.06426	0.06301	0.06178	0.06057	0.05938	0.05821	0.05705	0.05592
1.6	0.05480	0.05370	0.05262	0.05155	0.05050	0.04947	0.04846	0.04746	0.04648	0.04551
1.7	0.04457	0.04363	0.04272	0.04182	0.04093	0.04006	0.03920	0.03836	0.03754	0.03673
1.8	0.03593	0.03515	0.03438	0.03362	0.03288	0.03216	0.03144	0.03074	0.03005	0.02938
1.9	0.02872	0.02807	0.02743	0.02680	0.02619	0.02559	$\boxed{0.02500}$	0.02442	0.02385	0.02330
2.0	0.02275	0.02222	0.02169	0.02118	0.02068	0.02018	0.01970	0.01923	0.01876	0.01831
2.1	0.01786	0.01743	0.01700	0.01659	0.01618	0.01578	0.01539	0.01500	0.01463	0.01426
2.2	0.01390	0.01355	0.01321	0.01287	0.01255	0.01222	0.01191	0.01160	0.01130	0.01101
2.3	0.01072	0.01044	0.01017	0.00990	0.00964	0.00939	0.00914	0.00889	0.00866	0.00842
2.4	0.00820	0.00798	0.00776	0.00755	0.00734	0.00714	0.00695	0.00676	0.00657	0.00639
2.5	0.00621	0.00604	0.00587	0.00570	0.00554	0.00539	0.00523	0.00508	0.00494	0.00480
2.6	0.00466	0.00453	0.00440	0.00427	0.00415	0.00402	0.00391	0.00379	0.00368	0.00357
2.7	0.00347	0.00336	0.00326	0.00317	0.00307	0.00298	0.00289	0.00280	0.00272	0.00264
2.8	0.00256	0.00248	0.00240	0.00233	0.00226	0.00219	0.00212	0.00205	0.00199	0.00193
2.9	0.00187	0.00181	0.00175	0.00169	0.00164	0.00159	0.00154	0.00149	0.00144	0.00139
3.0	0.00135	0.00131	0.00126	0.00122	0.00118	0.00114	0.00111	0.00107	0.00104	0.00100
3.1	0.00097	0.00094	0.00090	0.00087	0.00084	0.00082	0.00079	0.00076	0.00074	0.00071
3.2	0.00069	0.00066	0.00064	0.00062	0.00060	0.00058	0.00056	0.00054	0.00052	0.00050
3.3	0.00048	0.00047	0.00045	0.00043	0.00042	0.00040	0.00039	0.00038	0.00036	0.00035
3.4	0.00034	0.00032	0.00031	0.00030	0.00029	0.00028	0.00027	0.00026	0.00025	0.00024

FIGURE D.1. Area in One Tail of the Standard Normal Distribution

BIBLIOGRAPHY

[1] 80/20, Inc. 80/20 Deflection Calulator. http://www.8020.net/.

[2] AIAA. Guide for the verification and validation of computational fluid dynamics simulations. Guide AIAA-G-077-1998(2002), American Institute of Aeronautics and Astronautics, Reston, VA, June 2002. Approved January 14, 1998. Reaffirmed June 25, 2002.

[3] H. Ailisto. *CAD model based planning and vision guidance for optical 3D coordinate measurement*. PhD thesis, University of Oulu, March 1996.

[4] B. D. Allen, G. Bishop, and G. Welch. Tracking: Beyond 15 minutes of thought: SIGGRAPH 2001 course 11. In *Computer Graphics*, Annual Conference on Computer Graphics & Interactive Techniques. ACM Press, Addison-Wesley, Los Angeles, CA, USA (August 12-17), SIGGRAPH 2001 course pack edition, 2001.

[5] B. D. Allen and G. Welch. A general method for comparing the expected performance of tracking and motion capture systems. In *VRST '05: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 201–210, New York, NY, USA, 2005. ACM Press.

[6] R. Ashby. *Design for an Intelligence-Amplifier*, volume 34 of *Annals of Mathematics Studies, Automata Studies*. Princeton University Press, 1956.

[7] D. C. Banks. Interactive computational simulation. http://www.cs.fsu.edu/∼banks/courses/csit/, Fall 2002. Class website for CIS5930.

[8] R. Bernhard, S. Dieter, B. Alexander, and B. Reinhard. Spatial analysis tools for virtual reality-based surgical planning. In *IEEE Symposium on 3D User Interface 2006 (3DUI 2006)*, pages 1–2, Alexandria, VA, March 2006. IEEE.

[9] D. K. Bhatnagar. Position trackers for head mounted display systems: A survey. Technical Report TR93-010, University of North Carolina at Chapel Hill, 1993.

[10] F. Biocca. *Cognitive technology: In Search of a Humane Interface*, chapter Intelligence Augmentation: The Vision Inside Virtual Reality. Amsterdam: North Holland, 1996.

[11] F. Biocca. The cyborg's dilemma: embodiment in virtual environments. In *Second International Conference on Cognitive Technology*, pages 12–26, August 1997.

[12] L. Blank. *Statistical Procedures for Engineering, Management and Science*. McGraw-Hill, first edition, 1980.

[13] J. Bouguet. Camera Calibration from Points and Lines in Dual-Space Geometry. In *Proc. 5th European Conf. on Computer Vision*, pages 2–6, 1998.

[14] J.-Y. Bouguet. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[15] G. E. P. Box. *Robustness in Statistics*, chapter Robustness in the Strategy of Scientific Model Building, pages 201–236. Academic Press, 1979.

[16] G. E. P. Box and G. M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, San Francisco, 1970.

[17] G. Brat, E. Denney, D. Giannakopoulou, J. Frank, and A. Jonsson. Verification of autonomous systems for space applications. In *IEEE Aerospace Conference*, March 2006.

[18] R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering: with MATLAB Exercises and Solutions*. Wiley & Sons, Inc., third edition, 1996.

[19] G. Burdea and P. Coiffet. *Virtual Reality Technology*. John Wiley & Sons, Inc., second edition, June 2003.

[20] R. P. Burton and I. E. Sutherland. Twinkle box – a three–dimensional computer input device. Technical Report 74–5161–3, Bell Laboratories, Salt Lake City, Utah, July 1974.

[21] V. Bush. As we may think. *The Atlantic Monthly*, July 1945.

[22] Carnegie Mellon University Graphics Lab. Motion capture database. http://mocap.cs.cmu.edu/search.php?subjectnumber=15.

[23] C. B. Chang and J. A. Tabaczyinski. Application of state estimation to target tracking. *IEEE Transactions on Automatic Control*, ac-29(2):98–109, February 1984.

[24] X. Chen. *Design of Many-Camera Tracking Systems For Scalability and Efficient Resource Allocation*. PhD thesis, Stanford University, 2002.

[25] V. L. Chi. Noise model and performance analysis of outward-looking optical trackers using lateral effect photo diodes. Technical Report TR95-012, University of North Carlina at Chapel Hill, April 1995.

[26] B. Chvatal. A combinational theorem in plane geometry. *Journal Combinational Theory*, B(18), 1975.

[27] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, Hillsdale, 1st edition, 1988.

[28] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to algorithms*. MIT Press Cambridge, MA, USA, 1990.

[29] L. Davis, E. Clarkson, and J. P. Rowland. Predicting accuracy in pose estimation for marker-based tracking. In *Proceedings of Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'04)*, pages 28–35. Institute of Electrical and Electronics Engineers, IEEE Computer Society Press, October 2003.

[30] L. Davis, F. Hamza-Lup, and J. P. Rowland. A method for designing marker-based tracking probes. In *Proceedings of Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'04)*. Institute of Electrical and Electronics Engineers, IEEE Computer Society Press, November 2003.

[31] M. de Jong. Sub-Poissonian shot noise. *Physics World*, page 22, August 1996.

[32] R. G. Easterling. Quantifying the uncertainty of computational predictions. In *Proceedings of the 2001 SEM Annual Conference*. SEM, June 2001.

[33] D. C. Engelbart. Augmenting human intellect: A conceptual framework. Summary Report AFOSR-3233, Stanford Reserach Institute, Menlo Park, California, October 1962.

[34] M. E. Farmer, R.-L. Hsu, and A. K. Jain. Interacting multiple model (imm) kalman filters for robust high speed human motion tracking. *icpr*, 02:20–22, 2002.

[35] A. Fath. Computational aspects of the linear optimal regulator problem. *IEEE Transactions on Automatic Control*, 14(5):547–550, October 1969.

[36] S. Fleishman, D. Cohen-Or, and D. Lischinski. Automatic camera placement for image–based modeling. *Computer Graphics Forum*, 19(2):101–110, June 2000.

[37] E. Foxlin, Y. Altshuler, L. Naimark, and M. Harrington. Flighttracker: A novel optical/inertial tracker for cockpit enhanced vision. In *Proceedings of Third IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'04)*, pages 212–221. Institute of Electrical and Electronics Engineers, IEEE Computer Society Press, November 2004.

[38] E. Foxlin, M. Harrington, and G. Pfeifer. Constellation: A wide-range wireless motion-tracking system for augmented reality and virtual set applications. In M. F. Cohen, editor, *Computer Graphics*, Annual Conference on Computer Graphics &

Interactive Techniques, pages 371–378. ACM Press, Addison-Wesley, Orlando, FL USA, SIGGRAPH 98 conference proceedings edition, 1998.

[39] D. Frantz, S. Kirsch, and A. Wiles. Specifying 3D tracking system accuracy – one manufacturer's view. Technical report, Northern Digital, Inc., Ontario, Canada, 2004.

[40] E. Fuchs/Foxlin. *Inertial Head-Tracking*. M.S. thesis, Massachusetts Institute of Technology, 1993. Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science.

[41] J. M. Gere and S. P. Timoshenko. *Mechanics of Materials*. PWS, Boston, MA, 2nd edition, 1984.

[42] M. S. Grewal and P. A. Angus. *Kalman Filtering Theory and Practice*. Information and System Sciences Series. Prentice Hall, Upper Saddle River, NJ USA, 1993.

[43] T. Hollerer, D. Hallaway, N. Tinna, and S. Feiner. Steps toward accommodating variable position tracking accuracy in a mobile augmented reality system. In *2nd International Workshop on Artificial Intelligence in Mobile Systems (AIMS 2001)*, pages 31–37, Columbia University, August 2001.

[44] R. T. Howard, A. S. Johnston, T. C. Bryan, and M. L. Book. Simulation and ground testing with the advanced video guidance sensor. In *SPIE Defense and Security Symposium*, March 2005.

[45] J. Isdale. Tutorial: Introduction to VR technology. In *Virtual Reality (VR)*, page 302. IEEE, March 2003. HRL Laboratories, LLC.

[46] O. Jacobs. *Introduction to Control Theory*. Oxford University Press, second edition, 1993.

[47] S. Julier and J. Uhlmann. A general method for approximating nonlinear transformations of probability distributions. Technical report, 1996.

[48] S. Julier and J. Uhlmann. A new method for the nonlinear transformation of means and covariances in filters estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482, 2000.

[49] S. J. Julier, J. K. Uhlmann, and H. Durrant-Whyte. A new approach for filtering nonlinear system. In *American Control Conference*, pages 1628–1632, 1995.

[50] S. J. Julier, and J. K. Uhlmann. Reduced Sigma Point Filters for the Propagation of Means and Covariances through Nonlinear Transformations. In *Proceedings of the*

IEEE American Control Conference, pages 887–892, Anchorage AK, USA, 8–10 May 2002. IEEE.

[51] S. J. Julier, and J. K. Uhlmann. The Scaled Unscented Transformation. In *Proceedings of the IEEE American Control Conference*, pages 4555–4559, Anchorage AK, USA, 8–10 May 2002. IEEE.

[52] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Information and System Sciences Series. Prentice Hall, Upper Saddle River, NJ USA, 2000.

[53] R. S. Kalawsky. *The Science of Virtual Reality and Virtual Environments: A Technical, Scientific and Engineering Reference on Virtual Environments*. Addison-Wesley, Wokingham, August 1993.

[54] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME—Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[55] S. Khoury, A. Freed, and D. Wessel. Volumetric modeling of acoustic fields in CNMAT's sound spatialization theatre. In *IEEE VIS '98: Proceedings of the IEEE conference on Visualization '98*, pages 439–442. Institute of Electrical and Electronics Engineers, IEEE Computer Society Press, 1998.

[56] N. Krahnstoever, M. Yeasin, and R. Sharma". "towards a unified framework for tracking and analysis of human motion". In *"Proc. IEEE Workshop on Detection and Recognition of Events in Video, Vancouver, Canada"*, "July" 2001.

[57] K. V. Laerhoven, A. Schmidt, and H. W. Gellersen. Multi-sensor context-aware clothing. In *Sixth International Symposium on Wearable Computers (ISWC)*, pages 1–8, Seattle, WA, 2002. IEEE.

[58] J. LaViola. A comparison of unscented and extended Kalman filtering for estimating quaternion motion. In *American Control Conference*, pages 2435–2440. IEEE Press, June 2003.

[59] J. L. LeMay, W. L. Brogan, C. E. Seal, and H. T. Hendrickson. Performance predictions for high altitude self–contained navigational systems. *IEEE Transaactions on Automatic Control*, ac-20(6):739–747, December 1975.

[60] F. L. Lewis. *Optimal Estimation: With an Introduction to Stochastic Control Theory*. John Wiley and Sons, Inc., 1986.

[61] M. A. Livingston. Visualization of rotation fields. In *VIS '97: Proceedings of the 8th conference on Visualization '97*, pages 491–ff., Los Alamitos, CA, USA, 1997. IEEE Computer Society Press.

[62] M. A. Livingston and A. State. Magnetic tracker calibration for improved augmented reality registration. *Presence: Teleoperators and Virtual Environments*, 6(5):532–546, 1997.

[63] A. E. Mace. *Sample-Size Determination*. E. Krieger Publishing Company, Huntington, NY, 1974.

[64] A. G. J. MacFarlane. An eigenvector solution of the optimal linear regulator problem. *Journal of Electronic Control*, 14:643–654, 1963.

[65] L. McMillan and G. Bishop. Plenoptic modeling an image–based rendering system. In R. Cook, editor, *ACM SIGGRAPH*, pages 39–46, Los Angeles, CA, August 1995. ACM, Addison Wesley. ISBN 0–201–84776–0.

[66] K. Meyer, H. Applewhite, and F. Biocca. A survey of position trackers. *Presence: Teleoperators and Virtual Environments*, 1(2):173–200, 1992.

[67] J. Michopoulos and S. Lambrakos. On the fundamental tautology of validating data-driven models and simulations. In V. S. Sunderam, G. D. van Albada, P. M. A. Sloot, and J. Dongarra, editors, *5th International Conference in Computational Science (ICCS 2005)*, volume 3515 of *Lecture Notes in Computer Science*, pages 738–745, Atlanta, GA, May 2005. Springer.

[68] T. B. Moeslund and E. Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.

[69] P. J. Moran and C. Henze. Large field visualization with demand-driven calculation. In *Visualization '99*, pages 27–506, San Francisco, CA, USA, October 1999. IEEE.

[70] A. Mulder. Human movement tracking technology. Technical Report TR 94-1, School of Kinesiology, Simon Fraser University, July 1994.

[71] S. Ntafos. On gallery watchmen in grids. *Information Processing Letters*, 23(2):99–102, August 1986.

[72] W. L. Oberkampf and T. G. Trucano. Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences*, 38(3):209–272, April 2002.

[73] K. Oksanen. *International Archives of Photogrammetry and Remote Sensing*, chapter The Design and Simulation of Video Digitizing by Using Three–Dimensional CAD models, pages 432–437. 1996.

[74] G. Olague and R. Mohr. Optimal camera placement for accurate reconstruction. *Pattern Recognition*, 35(4):927–944, 2002.

[75] J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, New York, 1987.

[76] M. Ouh-Young, M. Pique, L. Hughes, N. Srinivasan, and F. B. Jr. Using a manipulator for force display in molecular docking. In *Robotics and Automation Conference*, pages 1824–1829, Philadelphia, April 1988. IEEE.

[77] M. Perse, J. Pers, M. Kristan, S. Kovacic, and et al. Physics-based modelling of human motion using kalman filter and collision avoidance algorithm.

[78] Polhemus, Inc., Colchester, VT. *FASTRAK brochure*, July 2003.

[79] J. E. Potter. A matrix equation arising in statistical estimation theory. Technical Report CR-270, National Aeronautics and Space Administration, 1965.

[80] V. Pulkki. Virtual sound source positioning using vector base amplitude panning. *Journal of the Audio Engineering Society*, 45:456–466, 1997.

[81] R. Raskar, A. Agrawal, and J. Tumblin. Coded exposure photography: Motion deblurring using fluttered shutter. In *ACM Transactions on Graphics (SIGGRAPH)*, pages 795–804, Boston, MA, July 2006. ACM.

[82] P. Roache. Perspective: a method for uniform reporting of grid refinement studies. *Journal of Fluids Engineering, Transactions of the ASME*, 116(3):405–413, September 1994.

[83] L. G. Roberts. The Lincoln Wand. In *AFIPS Conference Proceedings*, volume 29, pages 223–227. Joint Computer Conference, Fall 1966.

[84] RSI, Inc. IDL: The Data Visualization and Analysis Platform. http://www.rsinc.com/idl/.

[85] Schultz and Melsa. *State Functions and Linear Control Systems*. McGraw Hill, 1967.

[86] J. E. Shigley and L. D. Mitchell. *Mechanical Engineering Design*. McGraw-Hill, Inc., 1983.

[87] P. Skagestad. Thinking with machines: Intelligence augmentation, evolutionary epistemology, and semiotic. *The Journal of Social and Evolutionary Systems*, 16(2):157–180, July 1993.

[88] B. R. Sorensen, M. Donath, G.-B. Yang, and R. C. Starr. The Minnesota scanner: A prototype sensor for three-dimensional tracking of moving body segments. *IEEE Transactions on Robotics and Automation*, 5(4):499–509, 1989.

[89] A. State, G. Welch, and A. Ilie. An interactive camera placement and visibility simulator for image-based vr applications. In *Engineering Reality of Virtual Reality 2006 (3D Imaging, Interaction, and Measurement; IS&T/SPIE 18th Annual Symposium on Electronic Imaging Science and Technology)*, San Jose, CA, January 2006.

[90] R. F. Stengel. *Stochastic Optimal Control: Theory and Application*. John Wiley & Sons, Inc., New York, NY, first edition, 1986.

[91] A. Stettner and D. P. Greenberg. Computer graphics visualization for acoustic simulation. In *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, volume 23, pages 195–206. Association of Computing Machinery, ACM Press, Addison-Wesley, 1989.

[92] C. Sul, S. Jung, and K. Wohn. Synthesis of human motion using kalman filter. In *CAPTECH '98: Proceedings of the International Workshop on Modelling and Motion Capture Techniques for Virtual Environments*, pages 100–112, London, UK, 1998. Springer-Verlag.

[93] J. Sun. Sensitivity analysis of the discrete-time algebraic riccati equation. *Linear Algebra and its Applications*, 275/276:595–615, 1998.

[94] I. E. Sutherland. A head-mounted three dimensional display. In *Proceedings of the 1968 Fall Joint Computer Conference, AFIPS Conference Proceedings*, volume 33, part 1, pages 757–764. Thompson Books, Washington, D.C., 1968.

[95] U. S. Army Research Laboratory's Army Research Office. BAA W911NF-04-R-0005. http://tinyurl.com/33y2kh, April 2005.

[96] N. Vallidis. *WHISPER A Spread Spectrum Approach to Occlusion in Acoustic Tracking*. PhD thesis, University of North Carolina at Chapel Hill, May 2002.

[97] J. P. Walsh and N. Dadoun. The design of Godot: A system for computer–aided room acoustics modeling and simulation. In *Tenth Annual Congress On Acoustics*, volume 3, pages E–15.3, Sydney, Australia, July 1980. Abstract Only.

[98] J. P. Walsh and N. Dadoun. What are we waiting for? the developmen of Godot II. *The Journal of the Acoustical Society of Ameria*, 71(S1):S5, April 1982. Abstract Only.

[99] G. Welch, B. D. Allen, A. Ilie, and G. Bishop. Measurement sample time optimization for human motion tracking/capture systems. In G. Zachmann, editor, *Proceedings of Trends and Issues in Tracking for Virtual Environments, Workshop at the IEEE Virtual Reality 2007 Conference*, Charlotte, NC USA, March 2007. IEEE.

[100] G. Welch and G. Bishop. An introduction to the Kalman filter. Technical Report TR95-041, University of North Carolina at Chapel Hill, Department of Computer Science, 1995.

[101] G. Welch and G. Bishop. SCAAT: Incremental tracking with incomplete information in places. In T. Whitted, editor, *Computer Graphics*, Annual Conference on Computer Graphics & Interactive Techniques, pages 333–344. ACM Press, Los Angeles, CA, USA (August 3-8), SIGGRAPH 97 conference proceedings edition, 1997.

[102] G. Welch and G. Bishop. An introduction to the Kalman filter: SIGGRAPH 2001 course 8. In *Computer Graphics*, Annual Conference on Computer Graphics & Interactive Techniques. ACM Press, Addison-Wesley Publishing Company, Los Angeles, CA, USA, SIGGRAPH 2001 course pack edition, August 12-17 2001.

[103] G. Welch, G. Bishop, L. Vicci, S. Brumback, K. Keller, and D. Colucci. The HiBall tracker: High-performance wide-area tracking for virtual and augmented environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST)*, pages 1–11. Association of Computing Machinery, ACM Press, Addison-Wesley Publishing Company, December 1999.

[104] G. Welch and E. Foxlin. Motion tracking: No silver bullet, but a respectable arsenal. *IEEE Computer Graphics Applications*, 22(6):24–38, 2002.

[105] G. Welch, H. Fuchs, B. Cairns, K. Mayer-Patel, D. H. Sonnenwald, R. Yang, A. State, H. Towles, A. Ilie, M. Ampalam, S. Krishnan, H. Maurin, V. Noel, and M. Noland. Remote 3D medical consultation. *Journal of Mobile Multimedia (JMM)*, 2007 (in press).

[106] G. F. Welch. *SCAAT: Incremental Tracking with Incomplete Information*. PhD thesis, University of North Carolina at Chapel Hill, Chapel Hill, NC, 1996.

[107] H. J. Woltring. New possibilities for human motion studies by real-time light spot position measurement. *Biotelemetry*, 1:132–146, 1974.

[108] C. R. Wren and A. P. Pentland. Dynamic modeling of human motion. In *Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 22–27, Nara, Japan, April 1998.