HandSCAPE: A Design of Computational Measuring System

by Jae-Chol Lee

Master of Science in Visual Studies, Massachusetts Institute of Technology, Feb. 2000

Master of Fine Arts, Graduate School of Hong-Ik University, Korea, Feb 1997

Bachelor of Fine Arts, School of Fine Arts and Design, Hong-Ik University, Korea, Feb. 1995

Submitted to the Prog ram in Media Arts and Sciences,
School of Architecture and Planning,
in Partial Fulfillment of the Requirement for the Degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology

June 2001

Author [illegible]

Jae-Chol Lee
Program in Media Arts and Sciences
May 11, 2001

Certified by [illegible]

Hiroshi Ishii
Associate Professor
Program in Media Arts and Sciences

Accepted by [illegible]

Stephen A. Benton
Chair, Departmental Committee for Graduate Students
Program in Media Arts and Sciences

HandSCAPE: A Design of Computational Measuring System

by Jae-Chol Lee

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning, on May 11, 2001,
in Partial Fulfillment of the Requirement for the Degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology

ABSTRACT

This thesis introduces HandSCAPE, a computational measuring system that provides designers and workers with a fluid means of bridging physical measuring and three-dimensional computer modeling. Using embedded orientation awareness and wireless communication, the HandSCAPE system first captures vector measurements using a digitally augmented tape measure, and then displays in real time the resulting dimensions in three-dimensional computer graphics. With efficiency of human-scale interaction using a tangible user interface, prototype systems are applied for specific on-site space planning, packing configuration, and archaeological field excavation in order to give users in the field immediate access to computational information. Underlying technology for custom orientation sensing and simulating three-dimensional graphics, as well as a concept of digitally constructed physical space are described. The success of the second generation of the system is also equipped with configurable parameters to increase the usability.

Thesis Supervisor

Hiroshi Ishii
Associate Professor
Program in Media Arts and Sciences
Massachusetts Institute of Technology

This work was supported by
Things That Think consortium and **Digital Life** consortium
at the MIT Media Laboratory

HandSCAPE: A Design of Computational Measuring System

Thesis Committee

Thesis Advisor [illegible]

Hiroshi Ishii
Associate Professor
Program in Media Arts and Sciences

Thesis Reader [illegible]

Joseph A. Paradiso
Principal Research Scientist
Media Laboratory

Thesis Reader [illegible]

Robert J.K. Jacob
Associate Professor in the
Department of EECS at Tufts University

Acknowledgements

The exploratory research topics in this thesis have evolved over countless discussions with colleagues in Tangible Media Group at the MIT Media Laboratory. The below, I would like to thank people who have participated in this research for their ingenious contributions.

I would first like to thank my thesis and research advisor, Professor Hiroshi Ishii, who has rescued me from the multitude of initial ideas through innumerous hour of insightful discussions for his endless support over the years. I especially respect his frontier research practice in Human-Computer Interaction and User Interface Design. His vision, "Tangible Bits" has been the most important aspects of conducting this exploratory research.

I also thank my thesis readers, Joseph A. Paradiso, a principal research scientist at the MIT Media Laboratory for his technical advice on the result, and Professor Rob Jacob at Turf University and for his broad guidance in articulating this project until the last minute.

The whole process of developing the HandSCAPE system in the Tangible Media Group was organized by a group of people under Professor Hiroshi Ishii's advice since 1998. I really appreciate Victor Su, Sandia Ren, Blair Dunn, Rujira Hongladaromp, James Hsiao, and Roy Gross, whose talent and trust I have engendered devote to develop this research over the years. Especially, it was impossible for me to implement this research their ingenious technical talents on Hardware and Software implementation.

In addition, I am grateful for the advice and support of the many people who have shown their interests with their professionals. I would like to thanks Joseph Blanc at Steelcases Inc. who encourages us in design with valuable real world comments. I would also thank Dr. Louise Krasniewiz at UCLA Costen Institute of Archaeology and Dr. Murizio Forte at National Research Council in Italy for their expertise in the domain of Virtual Archaeology. Moreover, I would thank the research support of the Things That Think and Digital Life consortia at the MIT Media Laboratory.

As working colleagues, I thank Brygg Ullmer, Rich Fletcher, Paul Yarin, Phil frei, James Pattern, Ali Mazalek, Ben Piper, Angelar Chang, and the rest of Tangible Media Group. I also thank the members of Program in Media Arts and Sciences and the MIT Media Laboratory. Especially, I would like to thank Professor Stephen A. Benton who encourages me to explore myself at MIT for last five years and Professor Nicholas Negraponte whose visionary integrity between Fine Arts and Fine Science lead me to come to this place since I was a young collage student.

I would thank Tal Streeter and Romig Streeter, who always encourage me to establish myself in this country and give me their sincere hospitality and support. Finally, I thank my parents who have encouraged their child to become more ambitious and challenged in his navigation of a life-sea without the expectation that they will receive anything in return.

Table of Contents

1.0 Introduction

"The ultimate job of design is to transform man's environment and tools and, by extension, man himself." [1] [Victor Papanek, 1963]

1.1 Measuring and Modeling

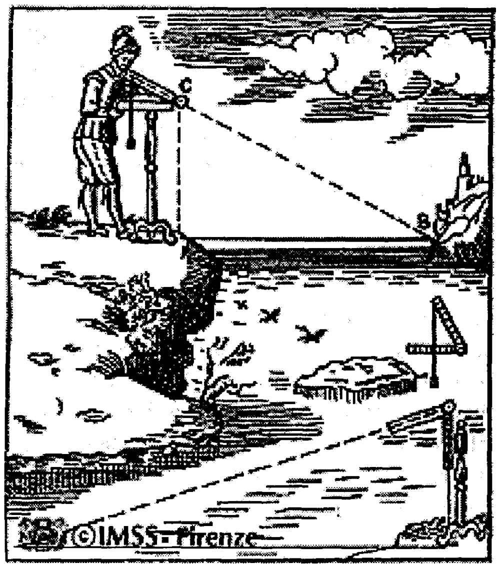


Figure 1. A Measuring Task, 1500, © Institute and Museum of the History of Science of Florence, Italy

Traditionally, people have represented spaces on paper by measuring and drawing approximate sketches of the measurements. In this manner, sketches, blueprints, and drawings have always been 2D representations of objects and spaces being measured and drawn. Recently, the advent of electronics and computing has given rise to a great number of new tools and systems for allowing these 2D representations to be displayed in precise 3D models. To generate the 3D models of physical objects and spaces, the usual approach involves typing all the measurements into a computer-modeling program, which visualizes the spaces and objects on computer screen.

Despite the accuracy and versatile usage of these devices, the user interfaces are limited to desktop scale interaction and are often awkward when used interactively for visualizing "human-scale" planning and construction. Virtual Reality (VR) systems allow the users to perform the "life-scale" interacting with 3D immersive simulation. Although these virtual environments enhance

[1] Papanek, Victor, (1963) Design for the Real World: Human Ecology and Social Change, New York: Van Nostrand Reinhold, p.28

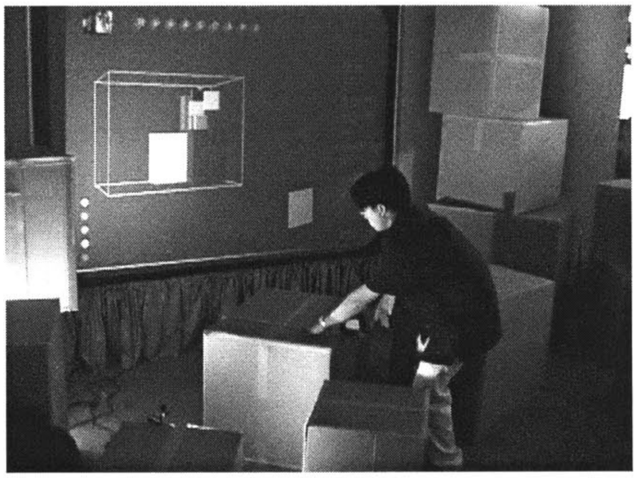
desktop interfaces, theses are not suited for "on-site" in which physical measuring, constructing, and surveying take place. In the field, designers and builders perform various tasks such as planning, measuring, sketching, and constructing. These tasks are becoming more complicated since workers now need to organize and communicate their expertise. In particular, on-site measuring tasks of interior construction, optimal use of storage, and archaeological field survey are extremely complex and require both efficient measuring and modeling process.

For these reasons, there is a need to create a simple handheld measuring tool which would allow workers in the field to gain efficiency (e.g. orientation-awareness, vector calculation, rapid 3D computer modeling and simulation) from modern-day computer technology that combines physical measuring and computer modeling in a single step interaction. This thesis, therefore, explores the design of a "Computational Measuring System", a computational mechanism for modeling digital objects and space incorporated with physical measuring. Providing a way to directly couple measuring with a computer program that simultaneously allows users to input and view their measurements, would simplify the complexity of the worker's tasks. As a non-desktop interface, this system generates digital models of physical objects while the users perform on-site measuring tasks, which employ peoples' existing skills and familiarity in physical environments.

1.2 Motivation

The initial goal of this exploration was to provide field workers with immediate access to the efficiency of computational information, while still using traditional measuring techniques. To extract the needs from the real world, we have closely studied the intended users (interior designers, storage workers, and archaeologist), and their tasks (measuring, planning, allocating, packing, and analyzing), and the physical workplaces (architectural space, loading dock, and field excavation). As a result, the requirements of portability and robustness led me to investigate what could be done with the existing tool commonly known as the tape measure. The following scenario well describes the basic motivation of this exploration:

- "Imagine if you could measure a room with a tape measure and have your computer instantly convert the rough figures into a 3D graphics representation of the room. A traditional tool like new measuring device measures linear distance, but also orientation and angle of tilt.
- When you pull the tape out, the linear distance is sensed by light encoders. When you move the device around, the orientation sensing hardware located underneath the tape captures position coordinates. It then transmits the measurements to a host.
- This device is designed for a variety of on-site measuring applications, where the combination of measuring and modeling increases efficiency such as storage allocation, archaeological sites, and interior design." [2]



By using a computer augmented digital tape measure and a corresponding real-time 3D simulation, this input device will allow the users to measure physical objects that can be instantly visualized on computer screen in real time. HandSCAPE will be portable, simple to use, and low-cost. The benefits of HandSCAPE include ease of measurement data entry (liner distance, orientation and angle of tilt) and visualize the physical objets in digital form in real-time, without the need to abstract physical space into units of measurement.

The successful innovation of designing such a tool demands more than augmenting a traditional tape measure with digital functionality. To provide field workers the above benefits, we considered that the interfaces should have: 1) comprehensible interaction through the manner of traditional measuring techniques, 2) carefully integrated sensing technology, display, and an interface suitable to the environment of use, and 3) an application-specific interface that uses particular measuring tasks.

[2] New Scientist, weekly science and technology Magazine, an article of HandSCAPE: "You've got it taped", reported by Catherine Zandonella, London: Reed Business Information Ltd., August 21, 2000, p.8

1.3 Thesis Overview

This written thesis introduces the motivation of the HandSCAPE concept. In Chapter 2, I continue with detailed information describing backgrounds of this design exploration. These include historical perspectives of measuring techniques from human body measurement to physical tools. As related work, current implementations of 3D input devices and interaction techniques are also introduced.

In Chapter 3, I describe the concept of orientation awareness and the overall system of HandSCAPE, including technical implementations and design rational. In Chapter 4, I demonstrated various prototypes and interaction techniques through applying the HandSCAPE systems to potential exploratory usages. With application-specific visualizations and usage scenarios for storage space configuration, furniture allocation, and archaeological survey, I also examined the initial system and usage results.

In Chapter 5, second-generation technical enhancements are introduced. After resolving and working around the constraints of the initial prototype, we enhanced the custom electronics for the stable orientation sensing mechanism, developed the software algorithm for interaction with the electronics, and added configurable parameters.

In Chapter 6, I generalized the value of traditional interaction techniques with everyday tools. This supports the notion of a "Computational Measuring System" as a means of digital construction of physical human-scale measuring. With the success of developing the second generation of the HandSCAPE system, I also envision possible additions as future work that can improve the system.

In Chapter 7, I summarized the design and implementation of the HandSCAPE system with incorporated technical documents in the appendices.

2.0 Background

2.1 Act of Measuring

The act of measuring is a human task that dates back thousands of years which evolved from a need to describe physical structures for the purpose of construction or surveying. The act of measuring is originally associated with body gestures, such as the way we stretch our arms when describing the length of an object (e.g. "I caught a fish this big").

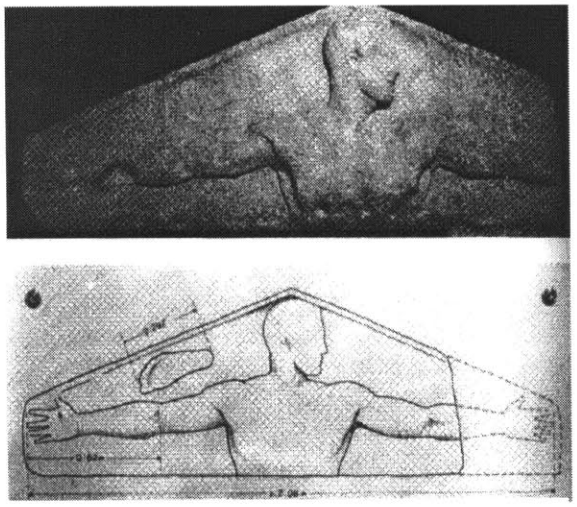


Figure 2. Greeks' Olympic Cubit

In fact, the earliest standard measurements were based on parts of the body. The Egyptian *Cubit* (3000 B.C.) is generally recognized to have been the most widespread unit of linear measurement in the ancient world [Figure 4]. It came into use based on the length of the arm from the elbow to the extended middle finger tips. "Measuring many items by the cubit, the ancient Egyptians developed chairs, ventilated beds, fast chariots, and seaworthy boats. Similarly, ancient Greeks used basic measurements. They used a finger as their basic unit. Sixteen fingers equaled one foot and twenty-four fingers equaled one foot and a half. Twenty-four fingers also equaled one *Olympic Cubit* and so did sixteen palms. The Roman later took the Greeks measurement and used them. In the Middles Ages, measurements were also made by body parts: a seat height would equal five fists, or a half-length."[3] Although the cubit gave an order of magnitude, it was hardly a standard and it varied widely in different time and places.

[3] Tilley R. A., (1993) The Measure of Man and Woman: Human Factors in Design, New York: Henry Dreyfuss Associates, Introduction, p. 9

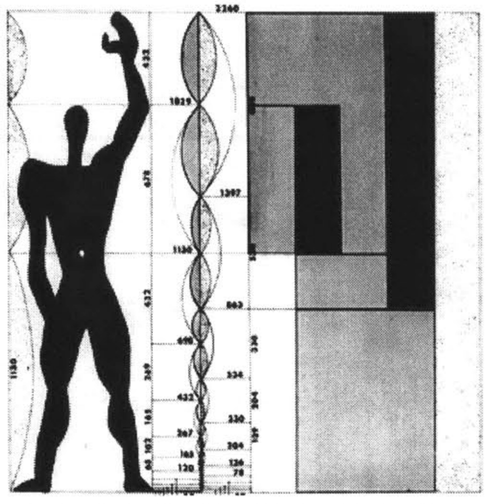


Figure 3. Le Modular, Le Corbusier, 1947

Relating to the physical measurements, later people introduced proportion applied to the human body. This sense of scale was used for building structures. "In discussing classical style of architecture, people often use the expression *designed to the human scale*."[4] The notion of human scale has also been emphasized by all fields of design and used for perceptive aesthetics and spatial interaction with our surrounding environment. The most detailed system of human proportion was first recorded during classical Roman times by theorists, artists, and architects. The Vitruvius Man, by Leonardo Da Vinci, in which a human figure is drawn circumscribed within a square and a circle, must be one of the most overworked visual images around.

Le Corbusier's "*Le Modular*" [Figure 3.] is also a modular system of architectural design, which reflects human proportion conveying a certain sense of rightness and harmony. Le Corbusier's proportional equivalents in terms of mathematical ratios between the dimension of human body parts evolved initially as simple aids to architectural drawing.

Throughout the Industrial Age, design, architecture, and engineering became global fields in which the professional faced the challenge of fashioning products and environments to the population. Recently, the advanced computer-aid design programs have replaced our traditional design practices by making more accurate drawings and modeling through computer technology. With the advancement of technology, we have regressed from the pinnacle of human proportional design used by classical designers. Since we can easily manipulate the scale in the digital domain, people have gradually lost the sensation of physical dimensions. By only using visual tools and typing abstract numbers on keyboards, designers have difficulty to perceptually correlate 2D visual representation on computer display with 3D physical space and proportion in human scale. I discussed this further in Chapter 6.1.3, Digitally Constructed Physical Space.

[4] Pheasant, S., (1998) Bodyspace: Anthropometrics, Ergonomics and the Design, London: Tayor and Francis, p. 7

2.2 Quantitative Measurement Tools

Over the years, measurement tools have evolved from using our arms and feet to more quantitative tools such as rulers and tape measures. Some Royal Egyptian Cubits were made of stone. [Figure 4] These may have been ceremonial rods, or perhaps, master gauges for calibration and comparison; their brittleness would make them unsuitable for the rough handling received by mason's tools. Workers engaged in building tombs, temples, pyramids, etc. were supplied with the cubits. The Romans introduced folding rules of bronze in six-and-twelve inch sizes [Figure 5]. These were probably "pocket" instruments for officials that were too expensive to be used by ordinary craftsmen who probably used plain strip rules.

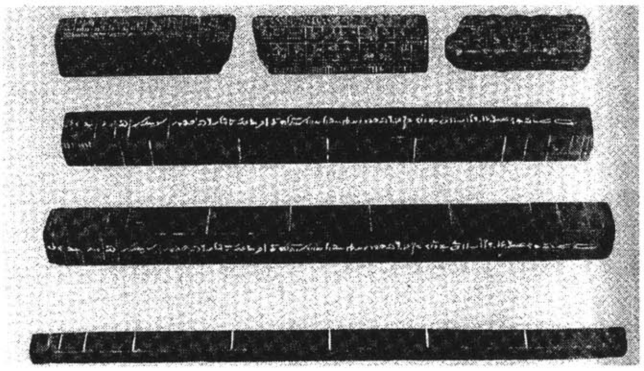


Figure 4. Royal Egyptian Cubit

Only limited evidence exists that graduated rules were used in the Middle Ages and the Renaissance; plain straightedges seem to have predominated. In 1683, an English writer described a foot rule as having 1/8-inch subdivisions. The folding rule, now made of wood, reappeared at the end of the 17th century. Measurement was long characterized by great national and regional differences because every large city in Europe had a different, but local standard.

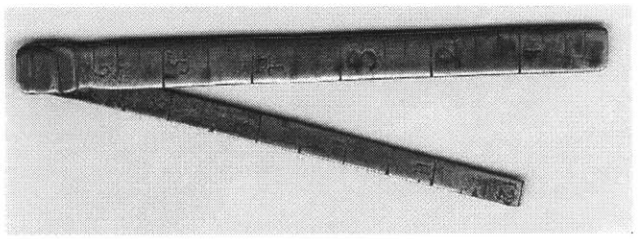


Figure 5. Roman Folding Rule: 6' 12", bronze

Since human beings have progressed, tools have been deeply connected with people and their lives. From the beginning of the human settlement, we can easily find evidences for how and why human beings invented tools. Before the Industrial Revolution of the 18th century, hand tools were used to cut and shape materials for the production of goods. In particular the tape measure as a hand tool was invented to accurately measure objects and spaces, while it provides mobility and ease of use.

As a precursor to measuring tape, the steel tape tool for comparative measuring was introduced [Figure 6], but did not provide very accurate measurements. The tape was used by surveyors, builders and architects for measuring distances between two points, consists of a long narrow steel strip (about 12mm wide) which is wound into a flat circular enclosed reel covered in leather. The winding handle located on the side of the case is used to wind or unwind the tape, and folds down on top of the case when the tape is not in use. A compact size graduated measuring tape was also introduced for portability and accuracy of measurements [Figure 7]. This coin-size measuring tape has only six inches at length, manufactured by Stanley in 1940.

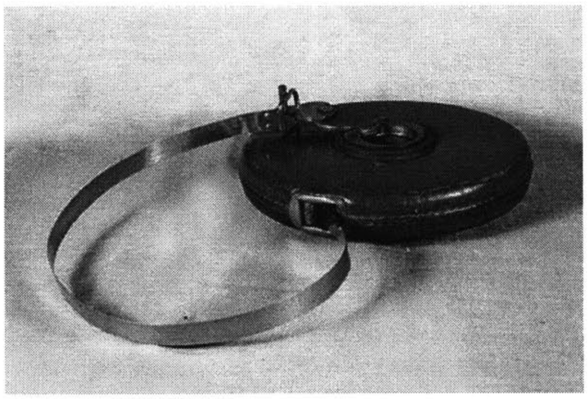


Figure 6. Manufactured in 1900, approx. 30m

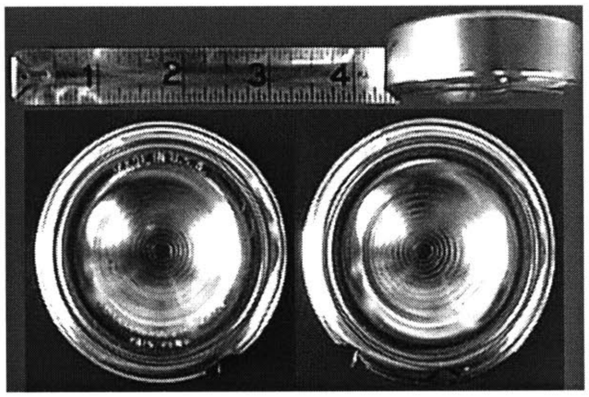


Figure 7 Six inches Nickle Plated Tape Measure, actual size

Throughout these evolutions of technology, the measuring tools were augmented with new capabilities for greater accuracy and ease of use. More recently, the advent of electronics and computing has given rise to a great number of new tools and systems for digitizing and precisely measuring three-dimensional objects (See 2.5 Related work). Nevertheless, many of these new tools are not practical or appropriate (and often too expensive) for common and portable measurement tasks that we have employed in such as use warehouses, construction sites, shipping yards, sporting events, and archeological field surveys. For this reason, the traditional measuring tape continues to be the tool of choice for common everyday measuring tasks.

2.3 Tool as Extension of Body

Mostly tools are hand-held devices that the users easily manipulate to complete a task. According to the inclusive definition of tools, "A tool is a moving entity whose use is initiated and actively guided by a human being toward a specific purpose." [5] In the use of tools, the degree of personal participation, more than any degree of independence from machine technology, influences perceptions of craft in work. In the real world, people wish to perform some tasks using a given physical tool, while most people use just their hands, some may use other body parts as well. Whether we stretch our limbs to grasp, pull, touch, strike, or reach for and grasp tools that facilitate the extension of human capability, physical action in body-space is clearly an area where technology extends our intentions, becoming an extension of our body. Thus, a tool becomes an extension of the body and functions as an augmentation of human abilities in their everyday life.

Martin Heidegger states the example of someone learning to use a hammer, "the famous Heidegger's Hammer" metaphor." [6] If I am just learning to use a hammer, at first I am very aware of the hammer and its activity as kind of alien to me. But with repeated use, I am transformed. My motor systems and neural systems adapt out to the new movements necessary to become a competent hammerer. During this period of transformation, the hammer recedes from my consciousness and becomes more and more an extension of my body-mind and its intentions and activities. Thus the meaning of the hammer becomes its hammering. In this manner, the tool being used in my hands becomes transparent while the hamming is more important task I need to focus. Like a well-used ping-pong paddle becomes transparent and allows a player to concentrate on the task – playing ping-pong. "This good fit of grasp is vital to making a paddle transparent and suggests a direction of transparent physical extensions of our body." [7]

Traditional hand tools and instruments were originally developed and optimized to perform certain tasks easily and efficiently. Tools have come to stand for the processes and this symbolic aspect clarifies the user to relate to their physical work. "One of the feature characters of physical properties of tools are that ease-of-use may be exploited to create a seamless interface to the digital

---

[5] McCollough Malcolm (1998) Abstracting Craft: The Practiced Digital Hand, Cambridge: The MIT Press, p.35

[6] Heidegger, M. (1962) Being and Time. New York: Harper and Row, pp. 150-158

[7] Ishii, H. Wisneski, C, Orbans, J., Junn, B., Paradiso, J., (1999) PingPongPlus: Design of an Athletic-Tangible interface for Computer Supported Cooperative Play, in proceeding of CHI '99, ACM Press, pp. 234-241

[8] Hinckely, K. Paustch, P. Goble, C. and Kassell, F., (1994) A Survey of Design Issues in Spatial Input, in Proceeding of the UIST '94, ACM Press, 2130222

domain that draws upon the task for which the hand tools were designed."[8] Thus, providing digital functionality to a measuring tape enables the user to consistently perform the measuring tasks as they would traditionally. I discuss this further in Chapter 6.1.2, Augmenting Everyday Tools.

2.4 Related Work

A variety of new input devices and interaction techniques have been developed for precisely digitizing, measuring, and modeling three-dimensional model [Table 1]. Examples of these are commercially available 3D digitizing and modeling products (e.g. *Digibot3*™, *Monkey2*™, *Polhemus 3Ball*™, etc.) Especially *ShapeTape*™ tracks the shape of tape while the tape is twisting and bending. It has six degrees-of-freedom (DOF) position and orientation of the two. Life-scale construction and survey make use of more precise and accurate measuring systems such as *Digital Photogrammetry* (producing precise surveying planes and maps directly into 3D CAD program by using digital cameras, digitizer, multi-image calibration software package), *Ultrasonic* (using polaroid sensors to measure distance) and *Laser Trackers* (providing for real-time coordinate determination by combining horizontal and vertical angle measurement with interferometric distance)[9].

Despite the accuracy and versatile use of the device, many of these new tools are not practical or appropriate, and often too expensive, for common and portable measurement tasks. More relevant commercially available portable digital tape measures (e.g. *DigiTape*™, *Bosch's DMB5*™, *AccuTape*™, etc.) are easy to read measurement sensed by light, shaft, or polaroid sensor and display on digital LCD panel. Nevertheless, measuring and computer modeling are always separated. Recently there has been growing interest and exploration for designing new input devices and interaction techniques for 3D modeling. For example, "*Bend and Twist Input Strip*"[10] facilitates the creation and direct manipulation of curves and surfaces in computer 3D graphics using *ShapeTape*™. Although the devices and systems provide accuracy and speed to digitize

[9] Ermes P. and F.A. van den Heuvel, (1998) Measurements with Digital Photogrammetry, International Archives of Photogrammetry and Remote Sensing, Volume 32, part 5, pp. 217-220

[10] Balakrishnan, R., Fitzmaurice, W., Kurtenbach, G., Singh, K. (1999) Exploring Interactive Curve and Surface Manipulation Using a Bend and Twist Sensitive Input Strip, In Proceedings of ACM Symposium on Interactive 3D Graphics (I3DG'99), ACM Press, pp. 111-118

physical 3D, these are designed for a desktop application, and not well suited for physical interaction at on-site applications.

	3D Digitizing	3D Modeling Software	Measuring & Surveying
Accurate	Polehemus 3Ball *TM*, Monkey2 *TM*, Digibot3 *TM*, ShapeTape *TM*, etc	AutoCAD *TM*, FormZ *TM*, 3D studioMax *TM*, TrueSpace *TM*, etc.	Photogrammetry, RazorTracker, Utrasonics, etc.
Sketching	Digital Tape Drawing [Buxton, UIST 1999] Bend and Twist Input Strip [I3DG 1999]	Sketch: 3D Scenes [Balakrishman, SIGGRAPH 1999] Teddy: 3D Freeform Sketching, [Igarashi, SIGGRAPH'99]	HandSCAPE [Lee, CHI 2000]

Table 1. Related Parameter for 3D digitizing, measuring, and modeling

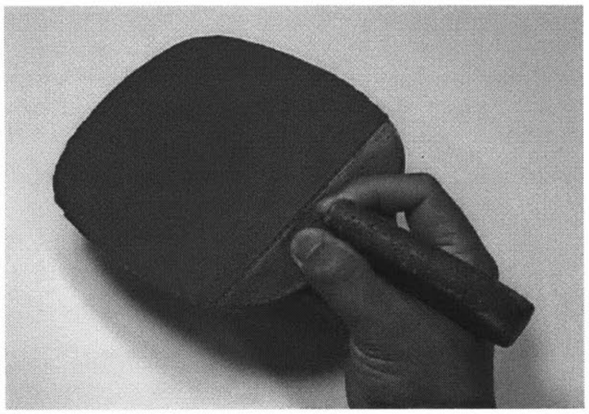
To take advantage of perceptual and comprehensible interaction between physical space and virtual space, "*Digital Tape Drawing*" [11] is an interactive technique using a tape to digitally draw curves where designers use a traditional tape to shape a curve in vehicle design. It offers several fundamental advantages over freeform sketching with pencil, given the large size of these sketches. Both investigations are well designed for conceptual modeling to allow a designer to quickly form, shape, and size. Against the complicated CAD-like 3D modeling programs, there are also emerging computer programs that perform rapid 3D modeling of primitive 3D scene[12] and allow sketching of freeform objects.[13] These are well developed in combining the ideal sketched by hands and computer-based modeling programs to improve the efficiency of sketching approximate models. Their emphasis is ease of low-level correction and simplicity of interface for sketching physical spaces and objects. These are well developed in combining the ideal sketched by hands and computer-based modeling programs to improve the efficiency and speed of sketching models.

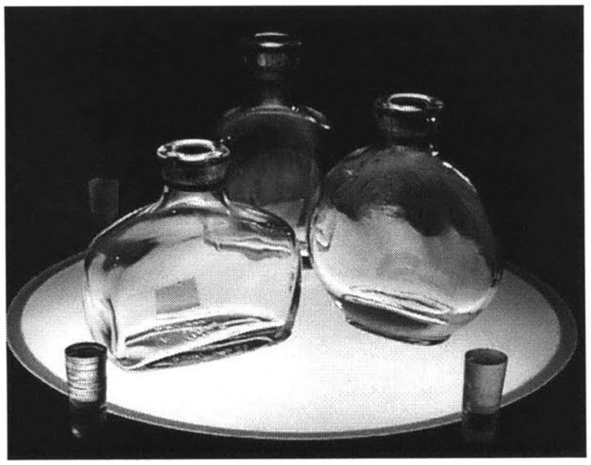
[11] Boxton, W., Fitzmaurice, G., Balakrishan, R., and Kurtenbach, G., (2000) Large-scale Tape Drawing, IEEE Computer Graphics and Applications 20(4), ACM Press, pp. 68-75

[12] Zeleznik R.C., Herndon K.P., and Hughs J.F (1996) SKETCH: An Interface for Sketching 3D Scenes, In *Proceedings of SIGGRAPH '96, pp. 163-170*

[13] Igarashi, T., Matsuoka, S., and Tanaka, H. (1999) Teddy; A Sketching Interface for 3D Freeform Design, In Proceeding of SIGGRAPH'99, pp. 409-416

There has been a recent flood of user interface design paradigms in which the users interact with computers by means of direct physical manipulation. In terms of physicality, these paradigms in the new interface designs take advantage of seamless interaction between the physical world and the digital world. "Tangible Bits"[14] is also strongly influenced to make greater of real physical objects as interfaces. Objects in everyday life are designed to serve and accessed physical form to digital information.





For instance, "*PingPongPlus*" [15] project is designed for digitally augmented cooperative play for ping-pong game. The system is augmented with dynamic graphics and sounds, determined by the position of impact, and the rhythm and style of play." This new class of interaction uses an existing tool (ping pong paddle) that manipulates digital information. The tool becomes transparent when the user is focused on playing ping pong. "*musicBottles*"[16] project is more related work for mapping digital functionality on top of existing objects. As a digital container, the physical empty bottle on top of specially designed sensing table metaphorically creates an illusion where people can control music by opening and closing the bottle cork. This evocative interface accesses digital information seamlessly and playfully.

[14] Ishii, H. and Ullmer, B. (1997) Tangible Bits: Toward seamless Interfaces between People, Bits, and Atoms, in proceeding of CHI '97, ACM Press, pp. 234-241

[15] Ishii, H., Wisneski, C., Orbanes, J., Chun, B., Paradoso, J., (1999) "PingPongPlus: Design of an Athletic-Tangible Interface for Computer-Supported Cooperative Play", in Proceeding of CHI 1999, ACM Press,

[16] Ishii, H., Mazalek, A., Lee, J., (2001) "Bottle as Minimal Interface To Access Digital Information" in Proceeding of CHI 2001, ACM Press,

3.0 Design Approach

3.1 The Basic HandSCAPE Concept

Over the past years, I have developed the first and primary design of an orientation-aware digital tape measure, called HandSCAPE [Figure 8], with Victor Su and Sandia Ren, in the Tangible Media Group at the MIT Media Lab. We augmented digital functionality such as additional orientation sensing, wireless connectivity, and custom designed 3D visualization software package, on top of existing tape measure.

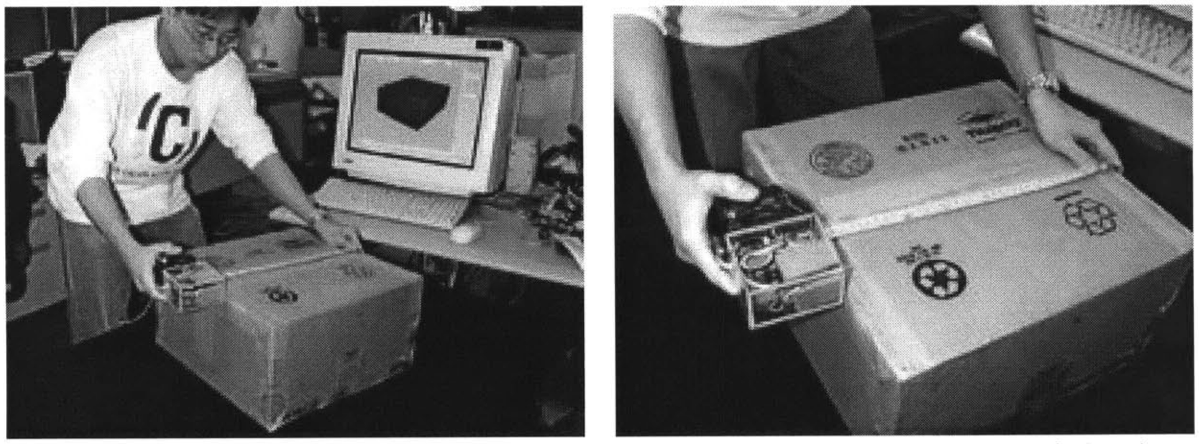


Figure 8. The initial prototype of the HandSCAPE, consisting of a digital tape measure, custom designed orientation, angular sensing electronics with serial communication, and running on SGI O2.

In order to combine physical measuring and computer modeling, we first considered several possible approaches, such as computer vision, ultrasonics, or lasers, but the requirements of portability and robustness led us to investigate what could be done with the existing tool commonly known as the tape measure. Therefore, we set out to augment a traditional tape measure with digital functionality, so that I could employ peoples' existing skills and familiarity in physical environments with this classic tool. As an input device, the HandSCAPE captures relevant vectors on each linear measurement and transmits the vector information --both position and orientation-- wirelessly to a remote computer. It then visualizes the volume of the resulting vectors in computer graphics in real time. A combination of measuring and modeling as a single seamless step interaction, HandSCAPE increases efficiency for on-site measuring tasks. The key innovation is to combine measuring and computer modeling as a single step. Thus, taking measurement and analyzing the data is no longer separated. By using the same communication frequency, multiple users can work together in the field for collaborative data acquisition. Indeed, this research concentrates on field applications in order to provide field workers to access the efficiency and speed of complex measuring tasks.

3.1.1 An Orientation-Awareness

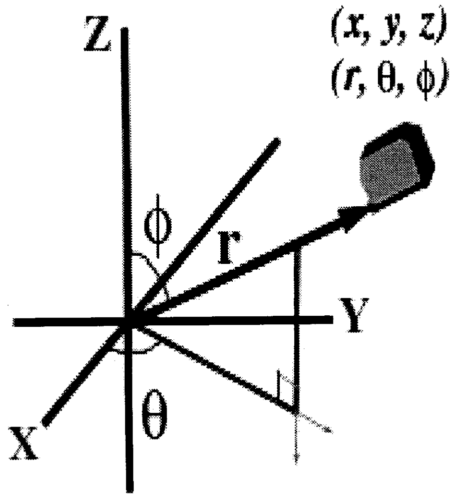


Figure 9. Coordinate System

Although the vector is a quantitative description of the physical dimension, the task of estimation frequently does not require this numerical abstraction. In actual measuring tasks, therefore, we measure linear distances with a series of vectors to recognize the volume of objects and spaces. Thus, orientation awareness is a significant contribution to the demand of measuring tasks. This embedded function enhances the capability of the hand-held tool effectively. Azimuth information can be extracted from its readings much in the same way as a digital compass. We chose to represent 3D orientation in spherical coordinates, as it was the most natural choice given the sensor data. Note that *r*, the radius, is obtained from the length of the tape, θ is produced by the magnetometer data, and ϕ is derived from the accelerometer data [Figure 9].

Since a handheld tape measure simply measures a linear distance, and it is not apparent how such a device could be utilized to capture the necessary spatial information. However, if orientation sensors are added, angular information can be measured as well. Then, knowing the distance and direction enables such a device to measure *vectors*. By measuring a series of vectors, the spatial dimensions of a physical object can be recorded and reconstructed in digital domain automatically. This concept led us to investigate an instrument that bridges the boundary between physical measuring and virtual modeling.

HandSCAPE also illustrates the utility of orientation sensing as an augmentation to common objects. The ability to locate the orientation of an object in physical space has been used in sports medicine and training, for example. These systems, however, require that the user wear special sensors on the body. HandSCAPE demonstrates that embedding orientation sensing as an integral part of objects yields the advantages of ease of set-up and usability by any individual. An orientation-aware golf club or table tennis paddle might help one improve one's golf swing or paddle stroke by exercising the brain's ability to recall learned physical movements without conscious effort.

3.1.2 Interface Design

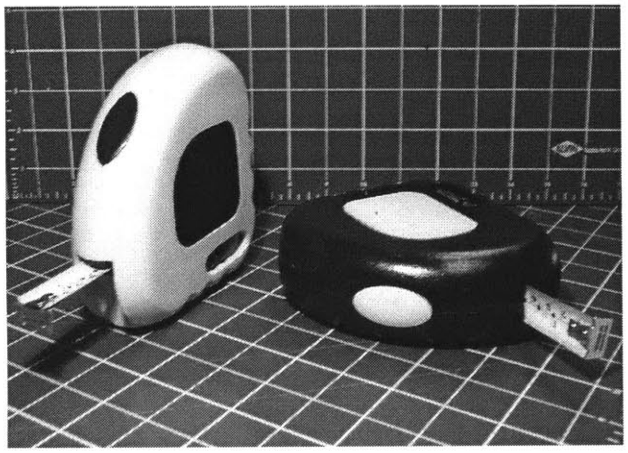


Figure 10. HandSCAPE version 1.0's Handheld Module, 5W x 3H x 1.2Dcm

As an initial step of building the HandSCAPE system, our design rational was to maximize the familiarity of augmenting digital functionality with traditional measuring tape, and to promote the seamless integration between traditional measuring techniques and 3D view of the data on computer display. At the same time, we sought to preserve the level of intrusiveness of user interface components. As such, the user interface widgets that are absolutely necessary are displayed, resulting in a system that emphasizes the coherent measuring workload over user interface components.

Accordingly, the task of complex measuring, such as furniture allocation in interior design and storage space optimization for packing, is directly coupled with computer graphics that allow users to visualize what they measure. While they enhance physical task with additional digital functionality, the interfaces should be 1) comprehensible interaction technique through the manner of traditional measuring techniques, 2) carefully integrated sensing technology, display, and interfaces in a physical form suitable to the environment of use, and 3) capable of convey relevant application-specific information that can be performed according to measuring tasks.

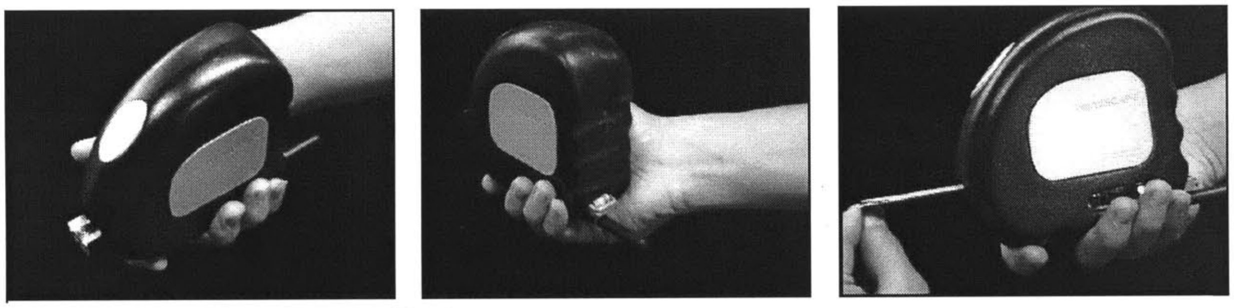
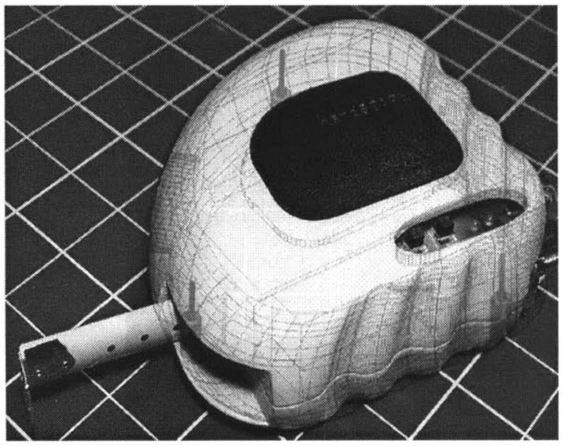


Figure 11. A Handheld Module in Hands

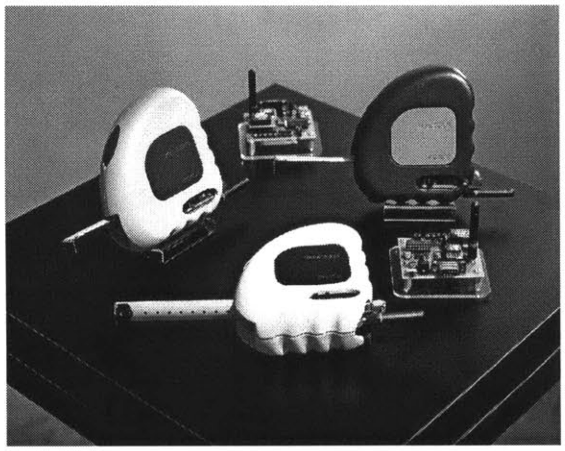
In designing the interaction techniques, therefore, we chose to borrow from existing measuring techniques and modify or enhance the techniques as required. Rather than inventing every from scratch, for example, some techniques like measuring a box without any order in which any side of box can be interpreted automatically as we used to do, others like 3D view with application-specific simulation and animated translation are enhanced, while some techniques like switching input techniques between use of the tape measure and the keyboards are new.

Thus our contribution is not so much industrial techniques themselves, but in the combination of these techniques into a fluid system for bridging the measuring with computer 3D modeling. Emphasis on designing interface with the notion of "transparency" and "simplicity" is also considered, while HandSCAPE provides the user with a simple interface with which to generate digital models of physical objects. The system is capable of handling multiple objects of different scales, ranging from entire architectural spaces to small handheld objects with the same ease. The implications of this technology include the ease of generating these digital models may be adapted to everyday measuring tasks for the dimensions and arrangement of a particular measuring workload.

3.1.3 Physical Form and Appearance



According to the study of Human Factors, "Hand tools should 1) perform well the function for which it is intended, 2) be properly proportioned to the operator's body dimensions, 3) be adjusted to the strength and work capacity of the operator, 4) not cause premature fatigues, 5) be adapted to the operator's sensory capacities, and 6) be inexpensive to purchase and maintain." [17] The appearance is the analogous to a traditional tape measure, so users would not have to adjust and learn how to use this device.



HandSCAPE's handheld module is made of thermoformed polystyrene and latex rubber. The device is designed to be robust in order to be used in field. Latex rubber gives more friction to prevent slipping. The shape of the cover material served both ergonomics and functionality by incorporating grips into the case. By combining case to the form factor. This product could be effectively cost produced by retrofitting mass produced tape measures with custom designed electronics. Packaging with software for interface with existing CAD computer programs should be relatively low cost.

[17] Proctor R, Zandt T, (1985) Human Factors in simple and complex systems, Chapter 17: Anthropometrics and Workspace Design, Boston: Allyn and Bacon Press, p. 378

3.2 Technical Implementation

3.2.1 Overall System

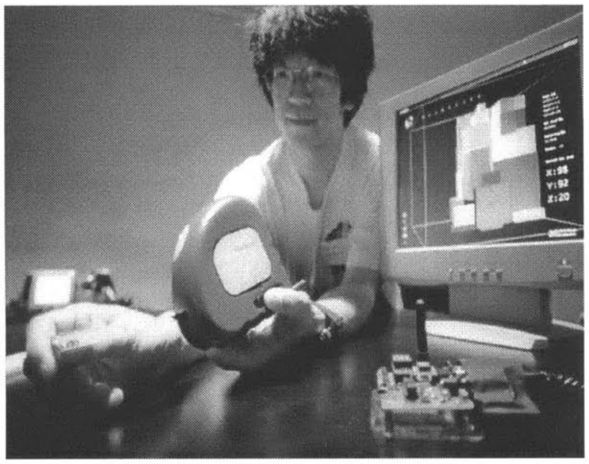
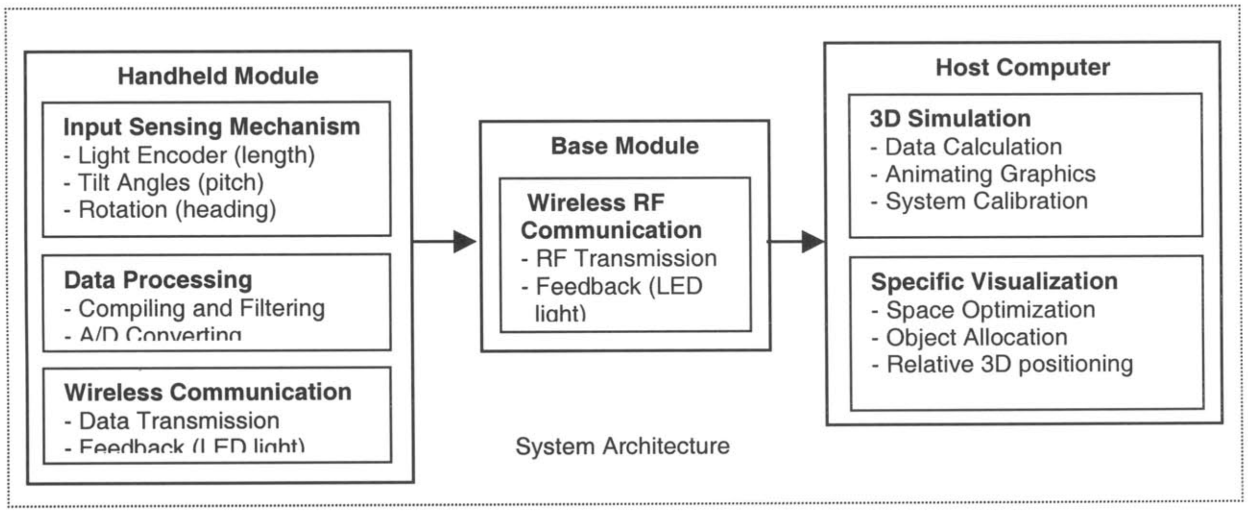


Figure 12. HandSCAPE System Components: handheld and base module, 400Mz laptop

The HandSCAPE system [Figure 12] is a single unit consisting of a measuring tape along with custom sensing electronics located on a printed circuit boards [Figure 13]. It communicates through a Radio Frequency (RF) signal to perform graphics rendering with TGS *Open Inventor*™ and using customized Microsoft *Visual C++*™ programs. To increase the mobility of the system, we especially implemented the whole system on a 400Mhz laptop.



3.2.2 Orientation Sensing and Data Processing

Technically, the digital tape measure tracks the length of the measuring tape by means of a linear optical encoder. HandSCAPE uses a commercially available digital tape. The handheld board interfaces to the encoder in the tape measure to track changes to the tape length. A series of holes are located in the tape, which pass over a row of photo-detectors that allow detection of both direction and magnitude of movement. The handheld electronics module also includes a two-axis micro-machined accelerometer, ADXL202, made by Analog Devices The accelerometer measures the local gravitational acceleration vector, thus acts as a tilt sensor that indicates the displacement of the HandSCAPE device from the horizontal plane. This also senses motion acceleration, which must be filtered out. However above orientation sensors were not accurate enough that has replaced in HandSCAPE version 2.0.

To measure the final degree of freedom, rotation about the vertical axis (heading), we have used a three-axis magnetometer, HMC2003, made by Honeywell, responsive to the Earth's magnetic field in its three sensing axes. The *Microchip PIC* ™ controller, PIC16C715 compiles the sensor data described above and transmits it through a RF interface to a host computer. All sensors reside on the tape measure itself, together with a PIC microcomputer to sample and serialize the data, and RF wireless transmitter. (See more detail of Electronic Parts in Appendix A.2).

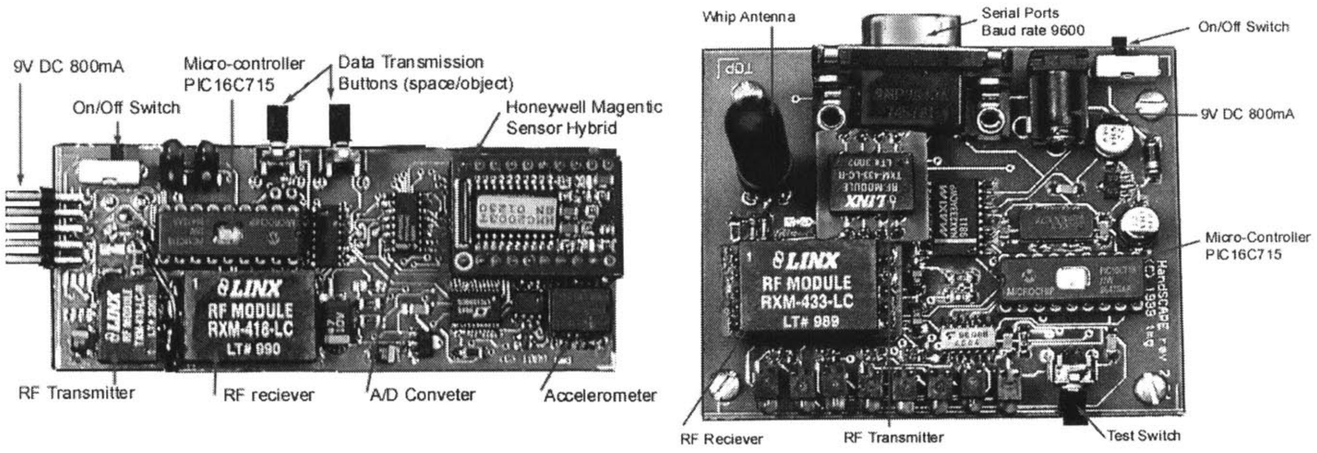

9V DC 800mA
On/Off Switch
Micro-controller PIC16C715
Data Transmission Buttons (space/object)
Honeywell Magentic Sensor Hybrid
RF Transmitter
RF reciever
A/D Conveter
Accelerometer
Whip Antenna
Serial Ports Baud rate 9600
On/Off Switch
9V DC 800mA
Micro-Controller PIC16C715
RF Reciever
RF Transmitter
Test Switch

Figure 13. Handheld-1 board (9 x 4 x 1.5 cm, left) and Base Unit (6 x 8 x 2 cm, right)

Originally we investigated "Multimodal, Compact, Wireless sensing in Expressive Footwear Project" [18], done by Professor Joseph Paradiso at the MIT Media Lab's Responsive Environment Group. The project basically instrumented a pair of dancing sneakers to measure 16 different parameters, including orientation-awareness (a 3-axis solid-state compass), tilt in two axes (a low-G MEMs accelerometer), high-G's/shock in 3 axes (a piezoelectric accelerometer). Thus, we

[18] Paradiso, A. J., Hsiao, K., Bedbasat, Y. A., J., Teengarden, A., (2000) Design and Implementation of Expressive Footwear Sensor, Proceedings of the IBM System Journal (ISJ), New York: IBM Coporation, vol 39, pp. 511-529

extracted the orientation and tilt sensing mechanism from the shoe board and built a more compact version for housing inside of the tape measure

	X-axis (pitch)			Y-axis (roll)		
Accel-erometer data	angle	AX	delta	angle	AY	delta
	90	200	+45	90	230	+45
	0	155	0	0	185	0
	-90	110	-45	-90	140	-45

Processing	`pitch angle = x-axis angle = arcsin[(AX - 155)/45]`
	`roll angle = y-axis angle = arcsin[(AY - 185)/45]`
	`phi = 90 degrees - pitch angle`					

Compass data	`Each axis: 0.5V-4.5V output over a range of 0 to 255`
	`Need to find offset and full-scale reading`
	`XH = CX*cos(pitch angle) + CY*sin(roll angle)*sin(pitch angle) - CZ*cos(roll angle)*sin(pitch angle)`
	`YH = CY*cos(roll angle) + CZ*sin(roll angle)`
	`Heading = 90                       XH=0, YH<0`
	`Heading =270                       XH=0, YH>0`
	`Heading =180-arctan(YH/XH)*180/pi       XH<0`
	`Heading =-arctan(YH/XH)*180/pi     XH>0,YH<0`
	`Heading =360-arctan(YH/XH)*180/pi  XH>0, YH>0`					

Table 2. Angular Sensing and Data Processing

The azimuth angle represents the angle of magnetic north. If we wish to compensate for the discrepancy between magnetic north and geographical north, we can subtract 16 degrees for Boston, according to the declination chart. Since the azimuth angle is already in the X-Y plane, we will take it to be the angle theta in spherical coordinates. If we don't compensate for declination, then we can take 0 degrees as magnetic North. To transform our vector in spherical coordinates into rectangular coordinates once we have obtained phi from the accelerometer, theta from the compass, and the radius from the encoder, we use the following relations:

```
x = radius * cos(theta) * sin(phi)
y = radius * sin(theta) * sin(phi)
z = radius * cos(phi)
```

This coordinate defines the endpoint of the vector that originates from the origin (the base of the tape measure) and passes through this point (the tip of the tape) in three-dimensional space. Note however, that this system does not provide a means for locating the reference frame in absolute space, which would require a tracking system or similar device. The angular data processing is carted in [Table 3].

3.2.3 Wireless Communication Protocol

To increase the mobility of HandSCAPE for on-site applications, wireless Radio Frequency (RF) communication is employed. The RF unit is composed of two bi-directional parts, an on-board RF transmitter / receiver and an external RF receiver / transmitter base unit which communicates with the host computer via a RS-232 serial interface. A cyclic redundancy check has been incorporated to ensure accurate data transmission between HandSCAPE and the base unit.

The RF unit utilizes the ultra-compact, low-cost LC-series transmitter and receiver, communicating at 315, 418, 433MHz from *Linx Technologies*™. The range of this communication is 30 feet to 50 feet, based on the RF interference on the site and a 9-volt battery is used that lasts of continuing operation. Using RF technology also allows HandSCAPE to handle multiple users with just one base unit since the data from each handheld unit can be tagged and processed accordingly by the host computer.

An audible tone through the onboard buzzer, or light signals is also added so that the user recognizes the data transmission, whenever a new measurement is transmitted to the computer. The RF communication protocol between the handheld and the base station consists of a 16-byte sequence consisting of 4-start-bytes and a 12-byte packet. The start byte consists of hexidecimal AA, chosen for its alternating bit sequence. This protocol was revised for interacting with new orientation sensor in HandSCAPE version 2.0, described in 5.1.2. Communication Protocol.

3.2.4 Pitch and Heading Calculation

As a key inventive idea, an orientation-awareness, the orientation of HandSCAPE is determined from the pitch and heading calculations, which are made from the compass and accelerometer readings. The pitch calculation is how much HandSCAPE is being tilted from the horizontal. The heading calculation corresponds to what angle, with respect to magnetic north, HandSCAPE is being held. The compass and accelerometer readings are all normalized to 512. The pitch calculation is the inverse sin of the normalized x reading from the accelerometer:

pitch = $\sin^{-1}$(ax_norm)

The roll calculation is needed to determine the heading. It is simply the inverse sin of the normalized y reading from the accelerometer:

```
roll = sin⁻¹(ay_norm)
```

To determine heading, the normalized horizontal compass x and y are first calculated, using the following:

```
cx_norm_horz = cx_norm * cos(pitch)+cy_norm * sin(roll) * sin(pitch);

cz_norm * cos(roll) * sin(pitch);

cy_norm_horz = cy_norm * cos(roll) + cz_norm * sin(roll);
```

where cx_norm is the normalized x reading from the compass, cy_norm is the normalized y reading. and cz_norm is the normalized z reading. In all cases, the readings are normalized to 512. The heading is then determined from the following [Table 3].

cx_norm_horz	cy_norm_horz	Heading
0	< 0	$\pi/2$
0	> 0	$3\pi/2$
< 0	Anything	$\pi - \tan^{-1}$ (cy_norm_horz/ cx_norm_horz)
	< 0	$- \tan^{-1}$ (cy_norm_horz/ cx_norm_horz)
> 0	≥ 0	$2\pi - \tan^{-1}$ (cy_norm_horz/ cx_norm_horz)

Table 3. Determination of heading from normalized horizontal x and y compass calculations

Once pitch and heading have been calculated, the current orientation of HandSCAPE can be determined. Pitch is how much HandSCAPE is tilted. If it is at $\pi/2$ or $-\pi/2$, it is being held vertically and a y measurement is being read. Heading corresponds to how much HandSCAPE is turned. During initialization, the room heading is calculated. This is the heading that corresponds to a perfect x measurement. If the difference between the room heading and the calculated heading is greater than $\pi/4$, then a z measurement is being made.

Otherwise, the reading is of a x measurement. Thus, with the pitch and heading calculations, HandSCAPE can determine its orientation and therefore determine which dimension of the box is being measured. This pitch and heading calculation has adapted to each application-specific graphics simulation for generating 3D model, relative orientation between two objects, and accurate positioning object's coordination. These particular modes are detailed in Chapter 4.2.2.Generating 3D Model.

3.2.5 Animating Graphics and 3D Visualization

By visualizing objects measured in relative 3D position, the relationship between the physical space and the virtual space is perceptually well defined and constantly monitored. Other visualization information in the graphical simulation is designated to provide a single source of access to information for application-specific information. These include map of physical space, the display of coordinates and dimensions for the object bounding boxes, the ability to create a multi-tier site, and three-dimensional navigation. The scene needs to be viewable from many different directions and ranges to give the user a three-dimensional feel and a chance to analyze the dig from different angles. The origin needs to be visible and there needs to be some indication of direction. The software is implemented using *Microsoft Visual C++ 6.0* ™ and *TGS OpenInventor* ™ for 3D graphical rendering. The RF receiver on the computer is connected to the serial port, and data is read through in the C++ code. The general structure of the code is an *event loop* that then enters a sequence of instructions based on what action is performed. The programming code base was designed with the principles of object-oriented programming which can be easily expandable to contain more functionality yet still maintain acceptable abstraction barriers. Every top-level element of the GUI is contained in its own object. The rendered scene is developed in a tree like fashion. The design of programming code for each application enters a scheduler based on input. *OpenInventor* ™ has its own object called a [TimerSensor]. From buttons in the GUI the [TimerSensor] is activated. The callback procedure operates as a state machine. It stores its internal state as a static variable. It performs an operation and then changes to the next state. After the callback procedure has cycled through all of its states, it then unschedules the [TimerSensor], and the GUI waits for another input. Depending on the state, different actions are performed and other functions are called. *TGS OpenInventor* ™ also has several built-in viewers with extensive capabilities. The [ExaminerViewer] object was chosen to be the viewer for this application since it allows the user to dynamically rotate, translate, and scale the scene.

4.0 Exploratory Usages and Evaluations

4.1 On-Site Measuring Applications

The most important aspect throughout this exploration is to provide a tool to enhance ability of average people using modern-day computer technology. This exploration also bridges Human-Computer Interface design and Industrial Innovation in real world problem solving. Our research, therefore, concentrated on field application where getting information from the physical world becomes more complicated.

For examining our concepts and prototypes from the real world needs, the potential utility of HandSCAPE of simplifying transfer of information from physical space into digital space through measuring is addressed here. On-line collaboration, accurate visual modeling, and rapid feedback between field and analysis may enhance the measuring task with additional digital functionality. With application-specific visualizations and interaction techniques, possible users will save time and efforts in the following scenarios. The following three potential application areas are comparatively examine the aim of this research as a solution to particular measuring problems. Each application describes objective of innovation and issues in both Hardware and Software that arise in using HandSCAPE for particular scenarios.

1) Storage: It is ideal to optimize the usage of space. Imagine a packing situation, there are thousands of different sized boxes to be packed in a container. With HandSCAPE, the users can optimize packing configuration based on the result from computer, while they measure.
2) Interior Design: People rearrange things here and there and renovate a room. An interior designer can show the virtual version of the room and manipulate the digital form of objects to the clients on-site.
3) Archaeology: In a dig, the measurement data incorporates with notes, maps, and photos automatically into a computer database. The user can also retrieve and visualize the data during the course of on-site excavation

4.2 Modeling Architectural Interior Surfaces

One of the basic measuring tasks in everyday life is for purposes of renovation and furniture allocation in interior space. With the advent of computer technology, there seems to be a great advantage to preview the space that will be changed. From designers to builders, the usual approach to create 3D model of the space involves typing all the measurements into a computer-modeling program to visualize the space. From measuring to sketching and typing, the methodology creating 3D model is multi-steps and the measuring and modeling is always performed separately. In order to overcome the need to abstract physical space into units of measurement and subsequently translate those measurements to units usable by graphics, there is a need to combine the process between physical measuring and computer modeling.

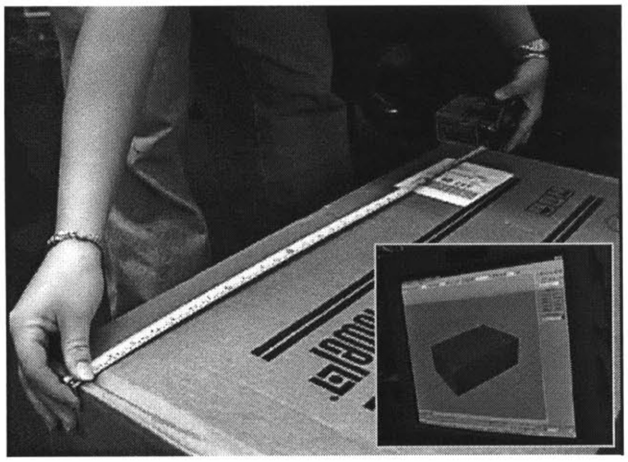


Figure 14. Traditional Measuring and Modeling are always separated.

Imagine, HandSCAPE allows the user to focus on the task of measurement alone when generating digital models. Moreover, it is very complicated to model relations between the objects and spaces without measuring orientation. Note the fact that orientation measure is even more complicated if modeling requires accurate visualization of a physical space that contains several objects. With HandSCAPE, it would also be easy to manipulate the objects on screen before physically moving the objects around. By taking vector measurement of multiple objects in a space, the user can focus on the task of measurement alone when generating digital models.

4.2.1 Scenario: Furniture Allocation

Interior designer steps into a room containing several pieces of furniture that he or she wishes to model. The primary interaction involves taking measurements of these objects and the distances between them. Once the user measures an object, its representative three-dimensional model with corresponding vectors immediately appears on the host computer in real-time. The coordinate defines the endpoint of the vector that originates from the base of the tape measure and passes through this point in the three dimensional space. Now the user measures the vector, *(x, y, z) (r, θ, φ)* between the first and second objects. The user can measure multiple physical objects in the space. It is also possible to make a procedure such that the user can capture the measurement vectors in any arbitrary order as natural as we used to perform [Figure 15]. The digital model generated by the measuring input is displayed on the computer screen and made available for manipulation by a keyboard and a mouse.

4.2.2 Object Mode vs. Space Mode

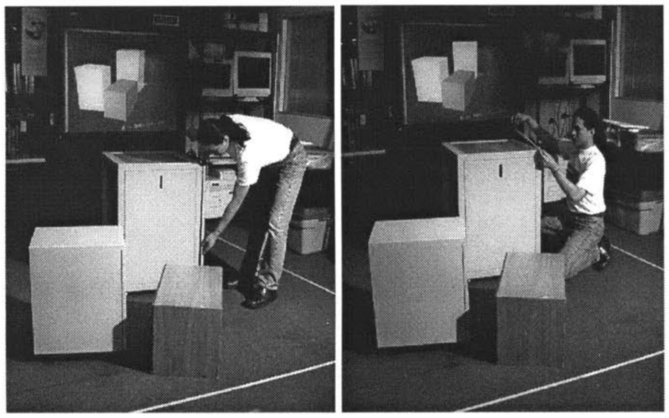


Figure 15. Measuring physical objects and generating the corresponding 3D models

HandSCAPE generates a series of vectors, which can be used to produce a digital representation of the physical object. The vectors generate the correctly oriented models in accurate relative positions to each other. A frame of reference is established for each new object in relation to objects that have already been measured. As long as each set of measurements originates from a certain reference point in both physical and virtual space, and this relationship is consistently observed, the objects will be modeled correctly relative to each other. Each vector is tagged as an object or space measurement by the micro-controller, selectable by a button on the device.

In object mode, the vectors are taken to be the dimensions of a parallelepiped, which represents the object being measured. In space mode, the vector is taken to indicate the spatial relationship between objects, and serves as the reference point for the next object to be measured.

4.2.3 Generating 3D Model in Space

The room modeling application involves measuring the contents of a room as boxes in their correct orientations and relative positions. The orientation of a box can easily be determined from the heading of the x measurement. The angle a box is turned is the difference between the heading of the room and the heading of the x measurement;

```
rotateAngle = heading - room_heading;
```

The placement of a box depends on its relative positions to the box that was measured before it. Each distance measurement is taken from the back right corner of the previous box to the front left corner of the next box. We can again use the pitch and heading calculations (discussed in Implementation section) to determine the relative position of the new box.

```
x_translation= length x cos(heading - room_heading);
y_translation= lengh x sin(pitch);
z_translation= lengh x sin(heading room_heading);
```

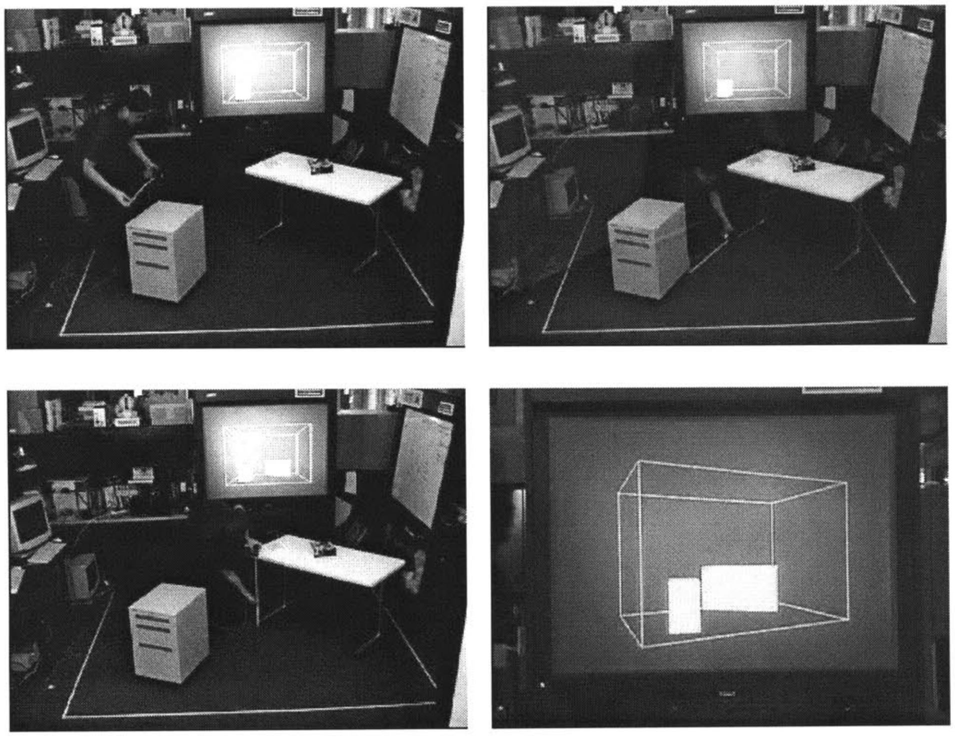


Figure 16.

1) start to measure from left bottom corner of the given space to the closest corner of the first object.

2) then, x, y, z measurements of the object are taken in any order.

3) measure the space between the back right corner of the first box to the front left corner of the next box using Space Mode button.

4) x, y, z measurements of the second object are taken, then the computer display the relative orientation between two object.

x_translation and z_translation are the distances between the two corners in the x and z direction respectively. y_translation is how much higher (or lower) the corner of the new box is from the corner of the old. From these three calculations, we can determine the position of the new box with respect to the box measured before it. Modeling a room begins by first determining the room heading, for later orientation calculations. The first box is then measured, then the distance between the first and second, then the second, etc. By going through the room and measuring all the boxes and the distances between them, we can rapidly model the room. Later on, the user can move to the computer and manipulate the objects in digital space. For example, he/she can rotate the model in order to gain a greater understanding of the spatial relationships through views from different angle. The user can also simulate object allocation and space configuration.

4.3 Optimal Use of Space in Storage

"Efficient use of space for packing, storing, and transporting goods is vital in the industrial sector." [19] As demand on space and distribution operation increases, investing in material handling and storage equipment to improve efficiency, profit margins, and reduce distribution and storage cost becomes crucial. High costs result from a lack of coordination between transport and storage systems, and between a storage system and its enclosure.

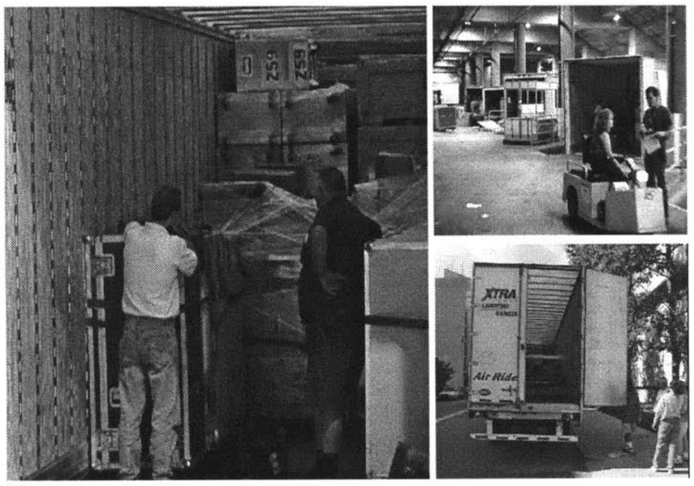


Figure 17. Trucking at a loading dock

The task of space optimization in terms of volume in bulk storage packing is usually a multi-step process involving measurements and calculations done by hand. HandSCAPE transforms this process into a one step interaction using a graphical simulation during the course of packing. With HandSCAPE the measuring has also been directly coupled with space optimization, thus allowing the users to perform the physical task efficiently.

[19] Falconer, P., drury, J. (1975) Building and Planning for Industrial Storage and Distribution, The Architectural Press: London, Introduction

4.3.1 Scenario: Optimal Packing

Imagine two truck drivers who need to transport hundreds of differently sized boxes in a truck container [Figure 17]. Assuming 1) it is not necessary to unload any of these boxes until arriving at the destination, 2) the weight of box is determined by the volume, 3) repeated access to a box is limited due to the truck driver's tight pick-up schedule. How, then, can they pack as many boxes as possible as fast as possible? Given a known volume of available storage, the user employs HandSCAPE to measure the dimension of each box. Whenever a box is measured, the host computer determines the best-fit position of the current box to minimize the use of the space. After measuring all the boxes, the optimal packing configuration is visualized on the screen and the actual physical packing can be performed according to this result.

4.3.2 Volume Sorting and Best-fit Positioning

The goal of our packing application is to determine the most efficient way of packing all the user's boxes. We surveyed approximation algorithms for some well-known and very natural combinatorial optimization problems, such as minimum set covering, minimum vertex covering, maximum set packing, and maximum independent set problems.

The algorithm we derived for the application is an offline algorithm that achieves this goal by performing two steps. First, it sorts the boxes by volume, from largest to smallest. It then uses a greedy algorithm (a step-by-step recipe to perform a single procedure in the recipe over and over again until it can't be done any more and see what kind of results it will produce). and packs the boxes in sorted order, using dynamic programming to determine the best position for each box. The result is an optimal solution for packing all the boxes.

The given space is first broken up into *height* number of floors, where *height* is the height of the space. The algorithm then tries to place the box on each floor by starting from the back right corner and moving outwards, checking at each position whether or not there is space to place the whole box. Once a space is found, the placement is recorded and the algorithm attempts to place the box on the next floor. It does this for each floor and at the end, all possible placements for the box are compared, and the one that minimizes the amount of space used is the placement that used for the box. This is done for each box, in sorted order. The result is that each box is placed in an optimal position and the user can then efficiently pack his or her boxes according to the consequent configurations.

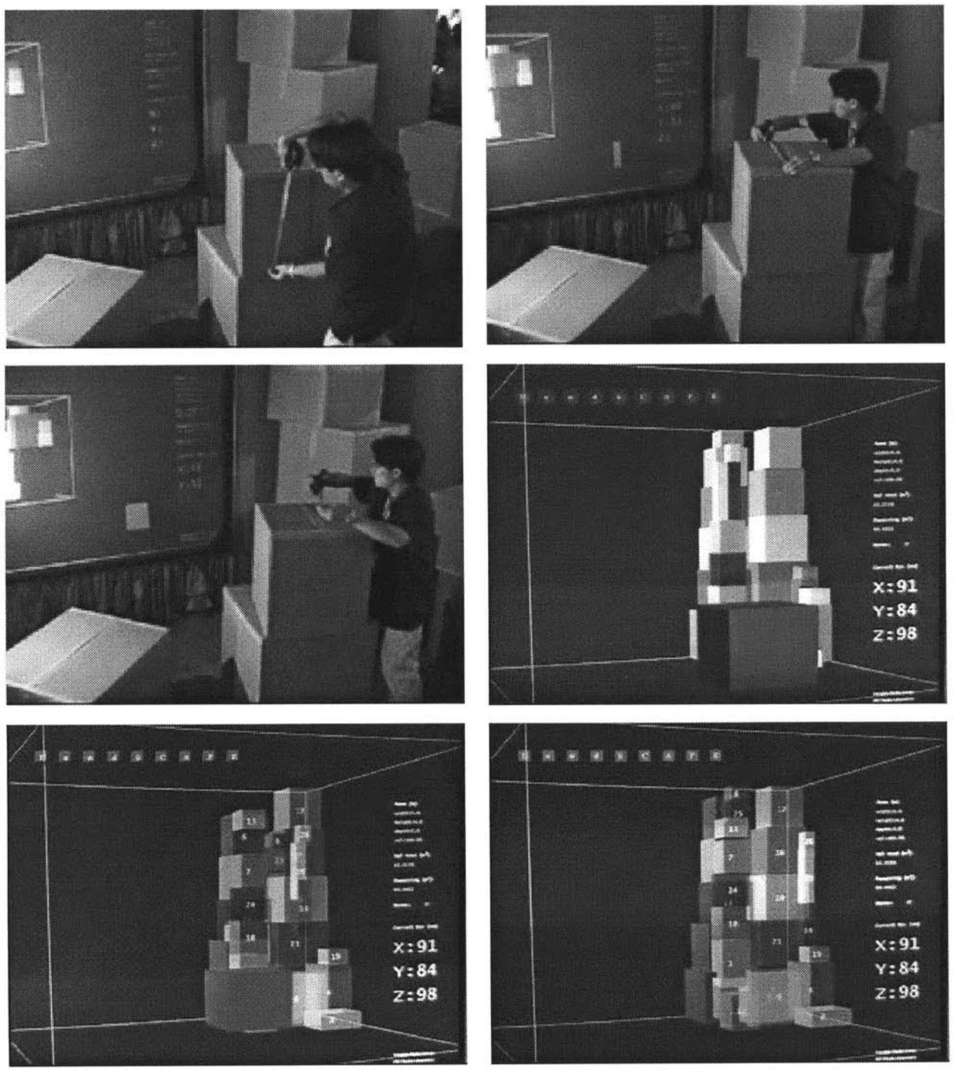


Figure 18.

0) start to measure a given storage space which defines a volume of the room.

1) while x, y, z measurements of boxes are taken, the identical dimension of the measurements are displayed as a box in real-time.

2) Once it's done, the computer animates a box and finds the best-fit position to maximize the use of space.

3) the whole process is animated with dynamic 3D simulation and the dimension of box, the room volume being used are also displayed.

4) Each box is numbered and the optimal packing simulation is animated on screen so that user can perform the optimal packing configuration according to this result

The first step in the algorithm is to sort the boxes by volume from largest to smallest. Since the boxes are then packed in sorted order, this causes large boxes to be packed first. This conserves space because larger boxes tend to have a larger top surface area than smaller boxes so packing them first allows more stacking to occur because the smaller boxes can be packed on top of them. Additionally, we assume that the larger boxes are heavier than the smaller boxes, so packing the larger boxes first will prevent a heavier box from being packed on top of a lighter one. The second step in the algorithm is where the packing is actually performed. A greedy algorithm is used, so each box is packed in the currently optimal place. Since each box could potentially be in many places, we use dynamic programming to determine.

4.3.3 Simulation: The Best Packing Algorithm

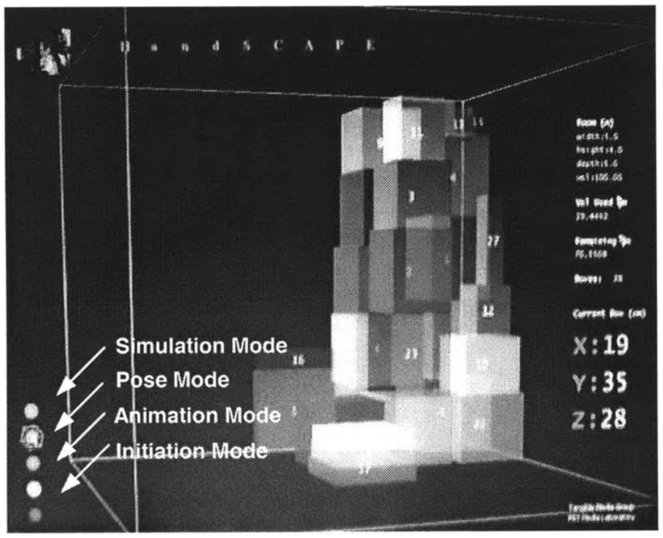


Figure 19. The on-screen visualization

The 3D visualization of optimal packing is designated to be an animation which displays the whole process of space optimization configuration. Once the three measurements are taken, the simulation first creates a box which is identical to the physical box measured and automatically visualizes where the current box should be placed to minimize the use of space. Additional information such as the volume of the room being used is dynamically displayed on the information bar at the right of screen. The user can also control the simulation by click the spheres located in left bottom corner of the screen in order to initiate, pose, and continue the simulation. Each box is numbered on screen so that user can perform the optimal packing configuration according to this result.

4.4 Relative 3D Positioning in Field Excavation

"To reconstruct a virtual model of an archaeological site, archaeologists first define the acquisition of excavated data in the field using hand written measurements, notes, and photography. As a final step of the excavation, the computer reconstructions of sites are usually used for the presentation of complex information in a visual way."[20] However, the manner in which an archeological excavation is recorded and documented has been left unchanged in recent years. With HandSCAPE, the users can gain the accuracy and speed of measuring in a field deposit, in getting surface data informatively. This would also help in research by allowing archeologists to view the relative size and placement of artifacts on computer modeling program. This synchronicity between measurement retrieval and modeling will facilitate analysis of field archaeological excavations.

4.4.1 Existing Archaeological Field Measuring Techniques

"The domain of archaeological field excavation promotes the comprehensive and interdisciplinary study of the human past. The goal is to preserve the great quantity of irreplaceable information associated with archeological excavation." [21] For archeologists, there is the added responsibility of taking primary field data — the innumerable photographs, maps, drawings, and notebooks that make up the archeological excavation record. Therefore, accurate recordings of the field datum of the specimens are crucial at the on-site excavations.

Furthermore, the goal of field measuring becomes increasingly important in a large deposit because having to take the same repetitive set of measurements on hundreds of specimens each day quickly becomes so tedious that even the most conscientious workers may become careless. To achieve the least error, archaeologists traditionally used a primary point for the vertical coordinate as a surface to the ground and lay out a grid system on which lines are no more than a meter apart [Figure 20].

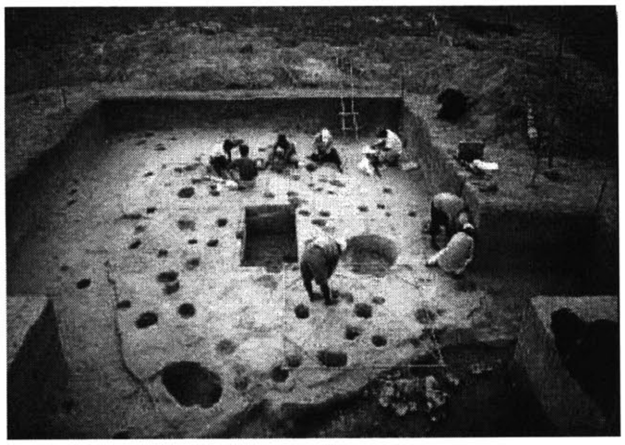


Figure 20 Excavators working in a large block excavation in Chipas, Mexico, using of portable grid. Photo Courtesy of 1999 © Cotsen Institute of Archaeology at UCLA

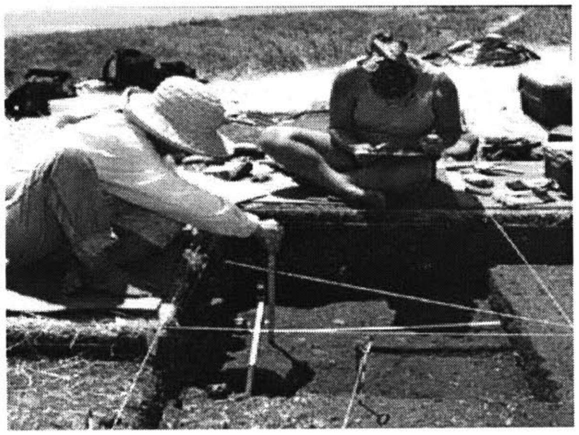


Figure 21. Three units in the process of excavation at Morse Point, Santa Cruz Island, Ca.Photo Courtesy of 1999 © Cotsen Institute of Archaeology at UCLA.

[20] Brien D. Dillon (1993) Practical Archaeology: Filed and Laboratory Techniques and Archaeological Logistics, Institute of Archaeology, University of California Press: Los Angeles, pp. 33-38
[21] Forte, M., Siliotti, A., (eds.) (1997) Virtual Archaeology: Re-creating Ancient Worlds, New York: Abrams, p. 130

The excavators are locating an artifact with respect to two walls of the unit using the two tape measures laying in the pit [Figure 21]. They are also using a third tape measure and a level string to measure depth. The level string is being stretched diagonally across the pit from the "candy cane" - a red and white metal pin near the bottom corner of the dig is set at a certain depth and does not move during the course of the excavation. Despite various efforts and other on-site measuring techniques, it is difficult to reduce chances of error. Besides accuracy, the principal factors in choosing a measuring technique should be speed and possibility of error. In order to reduce innumerous measurement errors, positioning measurements in the field needs a good measuring system to facilitate the accurate determination of the position of any object. These objectives are way for a new measuring technology.

4.4.2 Scenario: Reconstructive 3D Visualization

For capturing measurement data in field archaeology and facilitating a 3D visualization of an excavation rendered in computer graphics, we explored the same digital tape measure interacting with an enhanced archaeological-specific 3D visualization, As to this particular visualization application for archaeology, we called “GeoSCAPE”. The goal was to provide visual reconstruction methods by bridging the acquirement of accurate field measurements and the visualization of the work of an archeologist during the excavation, using computer database and additional information mapping. Following three main functions demonstrates both scenarios and implemented factors in using GeoSCAPE.

1) Dig Space and Anchor Point: Archeological excavations vary dramatically in size and complexity. When a new dig is created it is important that the user will be able to input the new dimension of the dig in order to create digs of varying size. While the excavation area is dig further down, the virtual dig extends on the basis of the scale and number of layers included in the dig. By first defining an “anchor point” that is the identical relationship between physical space and virtual dig. From the “anchor point”, a new reference frame is defined by making a vector in physical space. By first defining some anchor point, the archeologist could easily use the HandSCAPE tool to define a box around an artifact. This information is then transmitted to a receiver on the computer host a short distance away via RF and input into a modeling program, which adds the newly defined box to a rendered scene. This would help in research by allowing archeologists to view the relative size and placement of artifacts on a computer monitor.

Using the same RF frequency, multiple excavators can sequentially input the measurements to perform a collaborative excavation. Adding a GPS positioning device into the HandSCAPE tool, Multi-tier excavation technique can link each "anchor point" to define the relationship between the digs in a large excavated region. One of the difficult tasks for archaeologists is to measure a very large number of artifacts and visualized them in 3D. With HandSCAPE's relative positioning, the locations of any artifact that was found in that area will be displayed.

2) Artifact Creation: During the archaeological excavation, three-dimensional spacing is an important factor to be visualized. Once the relative position of the artifact has been determined, GeoSCAPE then sketches a rectangle that encircles the artifact: two simple measurements in the x and z direction creates the small rectangular area outlining the artifact. One last measurement of the height of the object completes the representative cube of the artifact. This is simply done by using HanSCAPE to measure the distance between the "anchor point" and one corner of the outlining rectangle of artifact. With HandSCAPE, the virtual dig has a localized coordinate system centered on one corner of the dig. This is also the "anchor point" defines the relationship between physical space and virtual dig.

From the "anchor point", a new reference frame is defined by making a vector in physical space. Within this reference frame, three additional measurements are taken. These are used to define a cubic figure in the virtual space that represents the location of an artifact in the physical excavation. With accuracy and speed, HandSCAPE allows the user to transmit the relative size and 3D position to a computer. As the user measures artifacts in real space the representation of the artifacts appear on the computer screen in real time. This synchronicity between measurement retrieval and modeling will facilitate analysis of field archaeological excavations. Using orientation sensing, the single measurement defines a relative position of the artifact in a dig with vector coordinate.

4.4.3 Relative 3D Positioning

The specification for the software could have covered a wide variety of possibilities. The two most significant operations of the program are the addition of an artifact to the dig and the operation of the compass. After the measurements are taken from HandSCAPE, the location of the origin of the new reference frame for the artifact is calculated as follows:

```
float theta = (M_PI/2)-pitch;
float omega = sin(theta) * en;
d = cos(theta) * en;
w = sin(heading) * omega;
h = cos(heading) * omega;
```

The variable "theta" represents the complement of the pitch, and the variable "en": represents the length of the vector. By calculating the sine of theta and multiplying by the length of the vector, we find the magnitude of the projection of the vector in the x, y plane to be omega. With this data we can then calculate the width, depth, and height (x, y, z) of the origin of the new reference frame relative to the anchor point. The cosine multiplied by the magnitude of the original vector gives depth (z-hat), and the width and height (x-hat, y-hat) can be determined using trigonometry with the heading and the magnitude of the projection as illustrated in the code. Once the new reference frame has been established, the fitting of the following measurements is trivial.

Regarding the accuracy and speed of measuring excavation measurement in a field deposit, the awareness of orientation and position of artifacts is a key in getting surface data informatively. As an orientation-aware measuring device, HandSCAPE provides a way for the user to get an accurate relative location of the excavated fossil as well as a visualization of the artifacts found as the archaeologists dig further and further down. This information is determined using the vectors produced by HandSCAPE measurements.

HandSCAPE first begins by drawing out a rectangular area that is to be excavated. Once an artifact is found, HandSCAPE then sketches a rectangle that encircles the artifact. The corners of the original rectangular area can then be used as an anchor point in positioning the new artifact. This is simply done by using HandSCAPE to measure the distance between one corner of the outer rectangle and one corner of the outlining rectangle. Taking the linear *length*, along with the *pitch* and *heading* calculations (discussed in implementation section) the user can determine the relative position of the artifact in a dig. Once the relative position of the artifact has been determined, two simple measurements in the x and z direction creates the small rectangular area outlining the artifact. One last measurement of the height of the object completes the representative cube of the artifact.

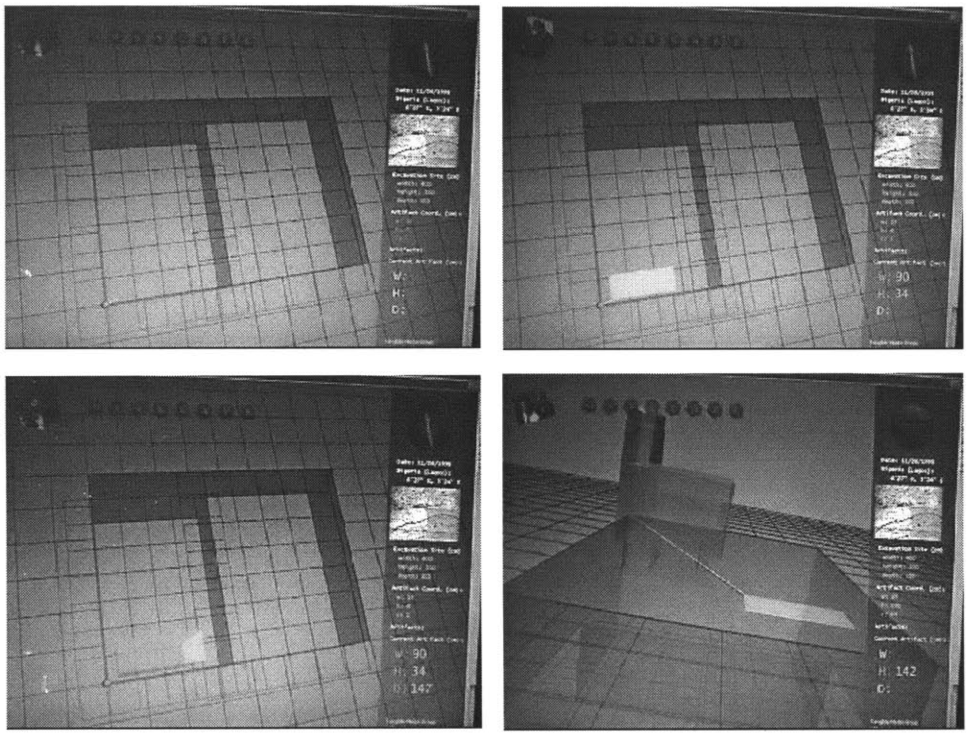


Figure 22

0) start to measure a given excavation area which defines a volume of the room.

1) start to measure the distance between the anchor point and one corner of artifact found.

2) x, y, z measurements of the artifact are taken and the computer displays the process of creating the artifact as a box.

3) to create second artifact, the user repeat the above steps. By clicking a botton on right information bar, the dig can be extended further down.

3) the whole process is animated with dynamic 3D simulation .

The compass is another significant piece of development. The initial idea of adding a virtual compass assumed that the compass would somehow be linked to rotation and movement of the scene or objects in the scene. However, the scene does not actually rotate using the *OpenInventor* [ExaminerViewer]. Rather, the camera moves in a spherical shell around the center of the scene, known as the focal point. The camera always faces the focal point, thereby effectively rotating the camera while translating the camera at the same instant. The program links this camera rotation to the negative rotation of the compass, which in turn is actually viewed by a different, stationary camera.

4.4.4 3D Visualization for Excavated Area

After hearing from archaeologists, we have also realized that archaeologists are more interested in integrating the excavated data together with spatial 3D visualizations. Because a site is often destroyed in order to uncover new information, it is very essential to better visualize the progress of excavation on screen. The significant of GeoSCAPE simulation is to add artifacts to the excavated space with relative position and visualize them. The artifact positioning is done by one measurement between the origin of the new reference frame and the "anchor point". In this manner,

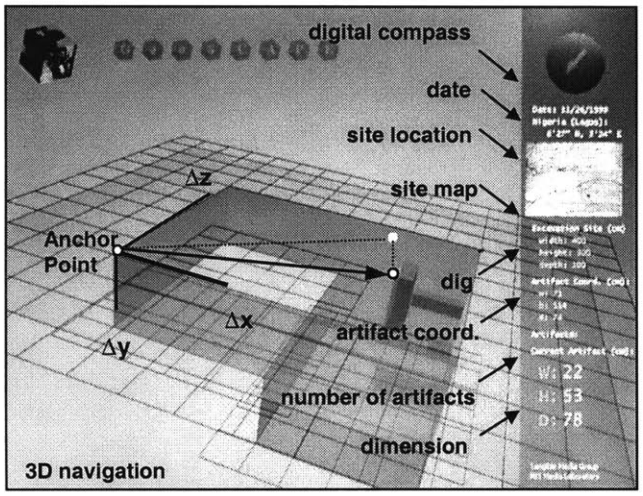


Figure 23. The on-screen visualization

GeoSCAPE allows a very large number of these artifacts to be defined. Additionally the users can change the location of the "anchor point" to set up their own reference frame in operating a large-scale excavation. While the excavation area is dug further down, the virtual dig extends on the basis of the scale and number of layers included in the dig. Using the same RF frequency, multiple excavators can sequentially input the measurements to perform a collaborative excavation. Issues of denoting perspective were also important. The compass, seen in the upper right corner of the screen shot [Figure 23], has its 3D rotation tied to the user's movement of the virtual excavation. Other visualizations include a map of the excavated region, excavation dates, the coordinates of artifacts in a dig, dimensions for the artifacts bounding boxes, the ability to create a multi-tier excavation site, and full three-dimensional navigation.

The remote computer displays the cube on computer screen [Figure 23] in which the artifact is located within an outer three-dimensional box representing the area that has been excavated. As the excavator digs deeper, the outer box will become larger and any additional artifacts that are found will also be represented as a small box within the space. The overall result, therefore, is a 3D visualization of the area excavated containing accurate representations in terms of size and position of any artifacts that were found in that area. In addition to storing measurement data on a computer, HandSCAPE is also able to incorporate geo-graphical and historical reference data while archaeologists perform the excavations. It enhances and verifies innovative archeological research with dynamic and interactive on-site data interpretation. In addition, following information is defined by the needs for the course of archaeological excavation. These are either fully functional or prospective for further development.

1) **Information Bar:** With a GUI based controller for inputting and reconfiguring the system, the users will be able to manipulate each data by typing or selecting. Each content will then be updated depending on the task performance. The additional information include excavation

dates, number of artifacts, scale of dig, a 2D map of the excavated region tied to GPS input, coordinates and dimensions for the artifact bounding boxes, etc.

2) **3D Navigation:** Issues of denoting perspective is also important. The compass, seen in the upper right corner of the screen has its 3D rotation tied to the user's movement of the virtual excavation. This will allow the users to virtually view the site with full three-dimensional navigation.

3) **2D Map (GPS):** GPS location device will be integrated with the host computer that link a map to show the approximate locations of these sites with twenty-five meter accuracy. GPS information will automatically change the map view depending on location and store the coordinate information in the archaeological database.

4) **Capturing Images and Notes:** A potential idea being pursued is to install a small camera on the handheld unit to capture images (photographs and notes). This would allow for recording additional information pertinent to the boxes being viewed and allow the user to better visualize the site with details. Each of the artifacts will contain other information, such as notes about the artifact, photographs, or more specific measurements. This gives the artifacts in the dig some sort of legitimate existence rather than merely defined as boxes.

5) **Connecting to Additional Information:** The boxes in the dig could be linked to a database with both images and notes so that the user can navigate through the site entirely electronically and not be concerned about referring to another source for notes and trying to figure out which box represents what. This could involve clicking on a box to bring up other information or it could involve pasting pictures on the sides of the boxes.

6) **Data archiving and loading:** A methodology needs to be introduced for saving and loading the information stored in these scenes. The archeologist should be able to save the scene when finished working for the day and be able to restore it to continue work the next. There will be two types of the database: one is automatically saved and loaded by a new scene to browse both images and notes linked to each box that represent the artifacts. The other is to connect other relevant information such as stones, bones, pottery, architecture, plants and seeds, inscriptions, wooden items, DNA analyses, dating reports, project items, coins, etc. This pre-archived information for a site-specific will inform archaeologists to analyze and interpret the excavated data under field conditions.

4.5 Usage Results and Initial Observations

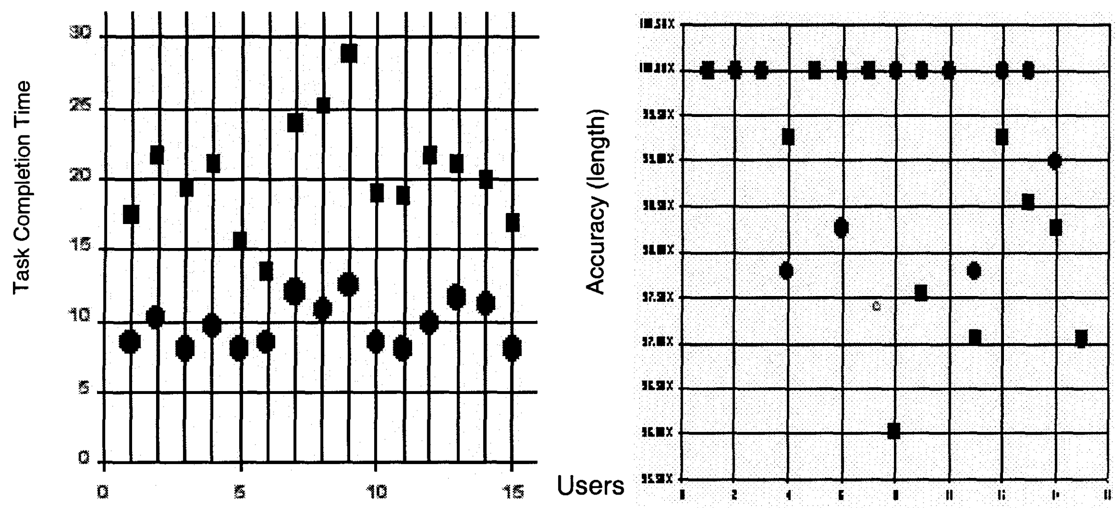
Throughout the initial prototype system implementations applied to three potential application areas, we could examine its functions and verify the innovation of current on-site measuring tasks using the HandSCAPE system. The first important lesson to draw from the usage result is that the HandSCAPE system is capable of increasing efficiency of particular measuring problems. In many ways, it has been an assumption of ours all along that users doesn't need to be interrupted by new interaction techniques while they concentrate on physical measuring. The assumption has been supported by comments from many users who have tried HandSCAPE.

4.5.1 Feedback and Comparative Experiment

We demonstrated the optimal packing application of HandSCAPE to the public at SIGGRAPH'99, where hundreds of users from diverse research backgrounds had the opportunity to try HandSCAPE. It was very easy to inform them about how the device worked because they were familiar with using measuring tapes in everyday life, and the size and shape of HandSCAPE is that of a standard tape measure. Secondly, most users were pleased how fast the data was transmitted to the screen. The interaction was very natural and preserved the senses of doing hand-on measurement, without the need to adjust to a new interface.

By running the optimal space configuration software, users could clearly see that the current box they measured was packed in the best fit position just as it would be in normal real world optimal packing. We proved that HandSCAPE was working in both hardware and software for the application. Although we only presented the packing optimization application, users responded well to adopting the device to various other uses. They even mentioned themselves that HandSCAPE is a very practical device that could be applied immediately to certain real world needs. Some users asked us about the capability of HandSCAPE to measure not only straight lines, but also curves. We also conducted a simple experiment to study two factors of interest, *speed* and *accuracy*. Each subject performed the experiment using two different tape measures --one being HandSCAPE and the other being a normal measuring tape. We asked 15 non-professional users to perform a 3D modeling task, (i.e. measuring a box and modeling it on the computer screen). Using the normal tape measure, users followed the standard approach of entering the measuring value into the modeling program to visualize the box. With HandSCAPE, on the other hand, users simply needed to measure the three dimensions of the box and its visualization immediately appeared on the screen. Results of this experiment are shown in [Figure 25]. From the data we observed 77% of users were more accurate when they used HandSCAPE. Measuring with HandSCAPE was also an average of 2.1 times faster than measuring with a normal tape.

4.5.2 Limiting Factors and Errors



Figrure 25. User data showing subject accuracy and completion time without HandSCAPE ■, and with HandSCAPE ●

As to the first prototype of the HandSCAPE version 1.0, there were many errors we have found while we tried to collect accurate orientation date. This fatal error caused by lack of solid state angular data was not proper to apply the HandSCAPE particular applications. We found that the accuracy of orientation is greatly affected by nearby magnetic fields (digital compass) and drifting resolution of tilt angle (accelerometer). Definitely the archaeology and interior design applications need more accurate and precise orientation measurements. For HandSCAPE version 1.0, therefore, we can only measure straight lines with orthogonal degree of box. We have also observed errors caused by pushing the button two or three times repeatedly. These errors usually resulted in graphical errors. The battery life (a day) and a complex system initialization have also caused a difficulty to let people to use. For these limiting factors, the current system needed to add new features such as add and delete buttons to correct the transmission. To increase the accuracy of the orientation measurements, therefore, we planed to explore another technology to improve the current sensing hardware and refining software package.

5.0 System and Design Refinement

5.1 Objectives

By examining the potential innovations of HandSCAPE in various exploratory usages described in Chapter 4, I was able to identify constrains of the system. Afterwards, the majority of the work has been devoted to the enhancement of the HandSCAPE system that eliminates or works around the constraints of the initial prototype (HandSCAPE version 1.0). Whereas the previous work has focused on implementing the digital tape measure and examining the exploratory usages of HandSCAPE, the work of the second generation of HandSCAPE was to enhance the capability of HandSCAPE.

Three main enhancements have been explored; 1) refining the electronics architecture for accurate orientation sensing, 2) improving software of coordinate measurement algorithm, and 3) adding configurable parameters and optional functionality. With the above technical enhancements, we could also articulate the design philosophy and approach that support the conceptual background of this exploratory design research. Consequently, the validity of this exploration was examined by the completion of followings:

- a robust system architecture with stable orientation sensing and data processing.
- visualization simulation software package for situated measuring applications
- evaluate aspects of the efficiency and interaction of the final implementation

5.2 System Architecture Refinement

As a reminder, HandSCAPE in its initial design (version 1.0) uses a linear optical encoder to tracks the length of the measuring. The handheld electronics module also includes a two-axis micro-machined accelerometer made by Analog Devices Inc. The accelerometer acts as a tilt sensor that indicates the displacement of the HandSCAPE device from the horizontal plane. To measure the final degree of freedom, rotation about the vertical axis, we have used a three-axis magnetometer responsive to the Earth's magnetic field in its three sensing axes.

5.2.1 Accurate Orientation and Inclination Sensing

In an attempt to reduce the rotation measurement error caused by nearby magnetic fields, we first investigated an alternative sensing mechanism using solid state orientation sensor such as a Gyroscope. It is commercially available within affordable prices. However, the resolution and speed of angular data are not good enough for the refinement of HandSCAPE. Finally we found a three-degree of freedom (3DOF) tracking sensor, called *InterTrack2*^{TM}, manufactured by InterSense Inc. The below [Figure 26] shows the basic replacement of the system.

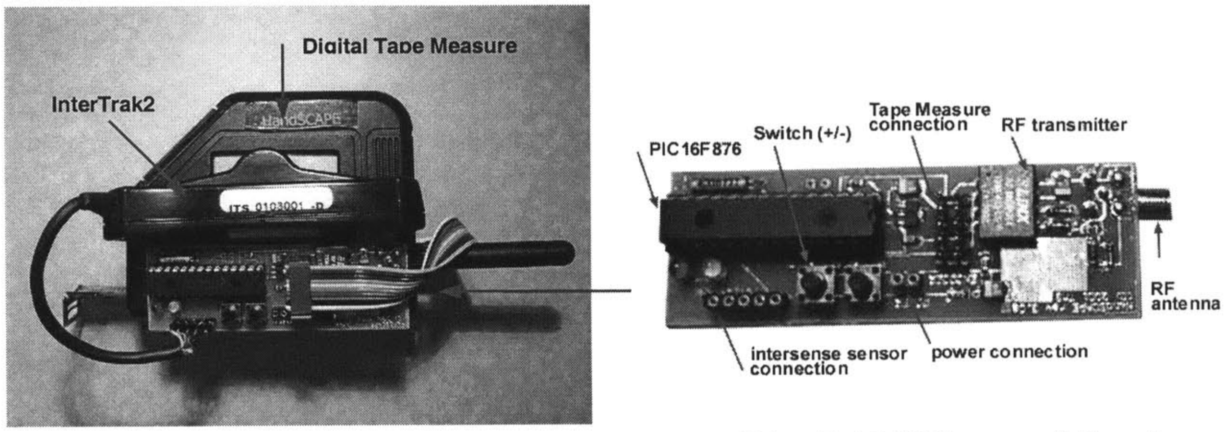


Figure 26. HandSCAPE version 2.0 – a handheld unit, interacting with InterTrak2 3DOF sensors (left), custom electronic board (right)

Therefore, the second-generation implementation of HandSCAPE version 2.0 replaces the custom orientation sensing hardware with the *InterTrack2*^{TM}. This replacement provides a much more stable and accurate measurement of the orientation of the tape measure. The majority of work has been focused on redesigning the custom electronic circuit board which interacts with the *InterTrack2*^{TM}. In detail, *InterTrax2* ^{TM} uses a combination of eight advanced sensors that allow extremely stable angular velocity measurements. The sensor signals are continually analyzed by an on-board microprocessor that uses sophisticated algorithms to compute the orientation of the tracker in space. The basic processing which computes orientation using *InterTrack2*^{TM} uses gyroscopic angular rate which provides the very rapid dynamic response of data transmission and high resolution of the information. The accelerometers and magnetometers are used to stabilize the

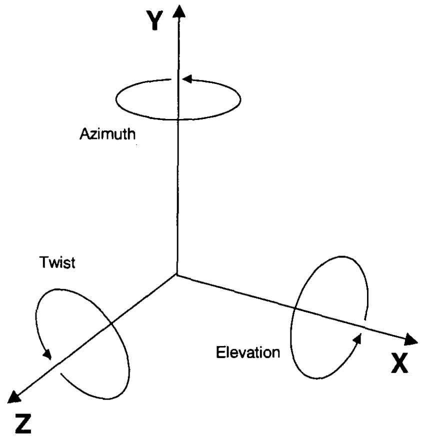


Figure 27. InterTack2's Coordinate Sys.

orientation to the earth's gravitational and magnetic fields, thus eliminating the gradual but unbounded accumulation of gyroscopic drift errors. The replacement provides a much more stable and accurate measurement of the orientation of the tape measure. Technically, the world axes, around which the tracker reports angular position is in a right-handed co-ordinate system, where +X is right, +Y is up, and +Z is out towards the viewer. The angular rotation follows the standard convention: when looking down any one axis in the positive direction, clockwise rotation is positive. In user terms this means that looking to the left is positive azimuth, looking up is positive elevation, and rolling to the left is positive twist, illustrated in [Figure 27].

5.2.2 Communication Protocol

In HandSCAPE version 2.0, the handheld firmware implements an algorithm to read the tape measure and track these changes in a counter variable. The handheld board also communicates with the *InterTrack2*™ sensor through a serial channel. The firmware handles initialization and zeroing of the sensor and polls the sensor for orientation data when the sample buttons are depressed. Initialization of the *InterTrack2*™ occurs at 1200 baud. After initialization, the data communication rate is set to 4800 baud. In addition, the handheld firmware automatically re-initializes the *InterTrack2*™ if an error occurs. The handheld board firmware is executed at 4 MHz on a PIC16F876 Micro-controller. See more details in Appendix A.3, HandSCAPE firmware 2.0.

The start byte serves the purpose of waking the RF receiver circuitry if it is in a low power mode as well as stabilizing the receiver prior to reception of the data packet. The first byte of the data packet is a header byte containing the sequence number of the packet. The sequence number is incremented during retransmissions of a given packet after an error such as a corrupted packet occurs and ensures that a packet is not processed and forwarded to the host computer more than once. The second byte of the data packet holds the node identification number, which identifies the particular handheld in a setup in which a single base station is receiving data from multiple handheld devices. The second byte also contains the sampling mode bit, which is determined by the sampling button that is depressed on the handheld. This byte is followed by the 12-bit tape measure encoder value, which is padded to 16 bits and sent high byte followed by low byte. The next six bytes are the Intersense sensor values, which are 16-bit values of the elevation, azimuth, and twist. These are also transmitted high byte followed by low byte.

The last two bytes in the data packet are a 16-bit CRC code generated from the contents of the packet and is used to verify that a particular packet is received without errors. If the packet is error-free, the base station transmits 4 start bytes followed by an acknowledgement byte consisting of the hexidecimal 53. If the handheld does not receive an acknowledgement after a predetermined timeout period, the sequence number is incremented and the data packet is retransmitted. This cycle repeats itself for a predetermined number of retransmissions, after which an error is returned.

The serial communication protocol consists of a tag delimited text string that is transmitted by the base station to the host computer. The contents of the RF data packet are converted to ASCII text to provide human-readable output. The "/ID" tag is followed by the node identification number of the handheld device that transmitted the data packet. The "/MO" tag is followed by the mode bit value. The "/EN" tag is followed by the encoder value, which directly corresponds to the length of the measuring tape. The "/AX" tag indicates the elevation angle. Similarly, the "/AY" and "/AZ" tags indicate the azimuth and twist values respectively. The serial string is terminated by a new line and carriage return.

5.2.3 Power Supply and Housing

The HandSCAPE version 1.0's communication protocol used continuous mode transmission. This could be altered to a polled protocol to save the battery power on the handheld device. We enhanced this problem by investigating several possibilities. However, the power consumption of the handheld module is dominated by the power requirements of the Intersense device. Since this is a commercial product that has been integrated with the HandSCAPE device, it is not possible to lower the power usage without having access to the design of the sensor itself. Therefore, attention must be paid to switching off the handheld unit when not in use. To solve a particular power off problem present in the first version of HandSCAPE, the power supply of the tape measure is now directly controlled by the power switch of the handheld device. Nevertheless, there was a problem that we have to reset the handheld device for initializing the angular information whenever we power off. It would be idea to have another mode, such as sleep mode so that we are both saving powers and the initial orientation information.

The enhancement of orientation sensing requires reconfiguration of the electronics circuit that contains all sensors on the tape measure itself, together with a PIC microcomputer to sample and serialize the data, a RF wireless transmitter. Therefore, I redesign the handheld unit and housing that fits a layout of the new electric circuit board along a measuring tape. The size of the HandSCAPE version 2.0's handheld module is small than the first prototype in height, but a bit bigger in width. This provides more comfort to grasp and operate.

5.3 Software Algorithm Refinement

5.3.1 Serial Input and Data Filtering

In order to further decrease the potential for instability of rotation data, we have improved the embedded software algorithm for formatting input data. The instability in the current orientation sensing in HandSCAPE version 1.0 is a result of sensor noise on the current micro controller which compiles sensor data. Therefore, we added better microprocessor software for digital filtering before processing orientation data. In HandSCAPE version 2.0, the use of the *InterTrack2*™ device allows us to determine the orientation of the vector measurement with only the elevation, azimuth, and twist readings returned by the device. Thus, these three readings replace the compass and accelerometer readings used in version 1.0. The new input data format is, therefore:

```
/IDaa/MOb/ENcccc/AXddddd/AYeeeee/AZfffff\n\r where
<aa>        4-bit device ID
<b>         mode bit
<cccc>      encoder count (reading in mm)
<ddddd>     unsigned 16-bit elevation reading
<eeeee>     unsigned 16-bit azimuth reading
<fffff>     unsigned 16-bit twist reading
```

5.3.2 Orientation Calculation and Relative 3D Positioning

The orientation of HandSCAPE is determined from the *pitch*, *roll*, and *heading (yaw)* calculations. Once the orientation is precisely measured by the above alternative orientation sensors, this calculation will be refined in the software algorithms which generate accurate vector measurement. As in HandSCAPE version 1.0, the length of the vector measurement is *encoder_reading*/10 (cm). The orientation of the vector measurement is also still determined using the *pitch*, *heading*, and *roll* values of the vector.

However, the way these are determined is different. *Pitch* is determined from the elevation reading, *heading* by azimuth reading, and *roll* by the twist reading. Each of the three readings is an unsigned 16-bit integer representation of an angle between 0 and 2π radians. Therefore, given a reading of x, its corresponding vector value would be:

```
value = (x/65534) * (2π)
```

The radian values of *pitch*, *heading*, and *roll* are then used to determine the orientation of the measurement. Since the *pitch*, *heading*, and *roll* values are much more accurate in v2.0, the orientation of the vector measurement can also be determined more correctly. Whether the measurement represents a width, height, or depth change is determined using the following algorithm:

```
if (π/4 < roll < 3π/4) or (5π/4 < roll < 7π/4) then
      height change
else
      if (π/4 < heading < 3π/4) or (5π/4 < heading < 7π/4) then
            depth change
      else
            width change
```

This algorithm assumes that a width change is a measurement made in the x plane, a height change is a measurement made in the y plane, and a depth change is a measurement made in the z plane. It places the borders of a height vs. width/depth change and the borders of a width vs. depth change at lines that are angled at $\pi/4$ radians with respect to the x, y, and z axes.

The HandSCAPE version 1.0 can generate a series of vectors, which can be used to produce a digital representation of the physical object. However, the system does not produce correctly oriented vectors in accurate relative positions to each other. Thus we add support for having a universal, called an anchor point; two points of reference will be established; one is an absolute frame (e.g. a corner of a room), the other is a relative frame (e.g. a corner of a new object). As long as each set of measurements originates from a certain reference point in both physical and virtual space, and this relationship is consistently observed, the objects is modeled correctly relative to each other.

5.4 Configuration and Additional Functionality

5.4.1 System Initialization

In addition to the hardware and software refinements mentioned above, we have implemented following design and usability features. The initialization sequence in HandSCAPE version 1.0 is inaccurate and complicated. The HandSCAPE version 2.0 supports simple calibration routines. When the user prompt the system initialization, the handheld module should be held motionless for 5 second system to calibrate orientation and set an anchor point.

The Reset button is also located on the top right side of the handheld module which sets the initial orientation of the virtual space. When pressed, the application will use the current position of the tape measure as the straight and level position of the device, thereby orienting the virtual world around that starting position. This initial position may be reset at any time by pressing the button again. The user can then save and restore the calibrated information. This option allows the user to program different sampling and filtering parameters of the system configuration.

5.4.2 Adding (+) and Deleting (-) Mode

In HandSCAPE version 1.0 system, each data point can be tagged as an object or space measurement. In object mode, the vectors are taken to be the dimensions of a parallelepiped, which represents the object being measured. In a relative frame, the vector is taken to indicate the spatial relationship between objects, and serves as the reference point for the next object to be measured. However, the simplest way to eliminate the measurement error caused by pushing the button repeatedly, is to add the mode of adding and deleting function. The ability to retake a measurement makes data acquisition more efficient during the course of measuring.

HandSCAPE version 2.0, therefore, has mode practical feature to sample the measurement data. When either of the sample buttons on the handheld board is depressed, the encoder counter variable is read, a sampling command is sent to the *InterTrack2*^{TM} sensor and the orientation data is received, and these data are transmitted through the wireless channel. The only difference between the two sample buttons is a chance in the sample mode bit. One button results in a sample mode bit value of 0, while the other button results in a sample mode bit value of 1. The host computer software may interpret the sample mode according to the specific application. This feature provides editing capabilities that correct error measurement that often happens in the measuring task.

6.0 Discussion and Future Work

6.1 Design Space

From the beginning of this design exploration, I have primarily focused on improving the methods of simplifying complex measuring tasks. Although HandSCAPE is not designed to be an input device for complex 3D modeling, I believe that incorporating rapid 3D modeling into situated measuring applications is a valid approach to solving complex measuring tasks that can be applied to real world needs. Thorough comprehensive examination of the comparative efficiency issues for particular application domains and the enhancement of the interaction systems, I verified that our work was intuitive as follows:

6.1.1 Value of traditional interaction techniques

The recent trend in user interface design involves the users directly manipulating digital information. In terms of actual tangibility and physicality, these interfaces take advantage of seamless interaction between the physical world and the digital world. Rather than demanding the learning of a new interaction, physical feedback systems are designed to fit the traditional interaction techniques that users already know.

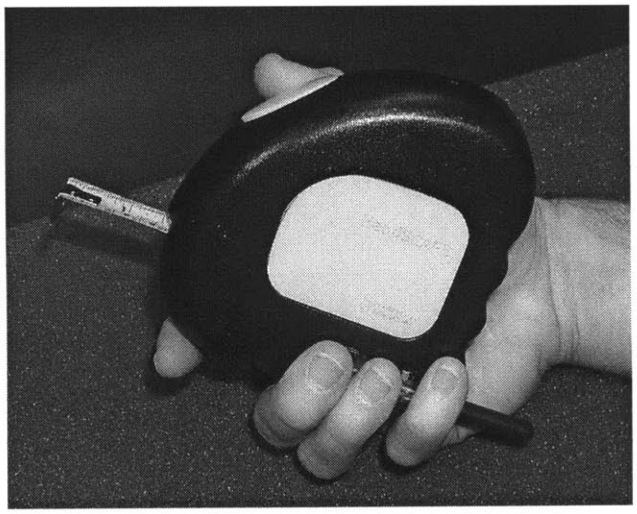
"While well suited to applications in which the computers is the focus of attention, this type of interaction is difficult to carry out in conjunction with inherently physical tasks – sorting of items on a table, the location of physical objects or containers, or the use of physical tools." [22] The core of HandSCAPE's interaction concept draws upon the familiarity of physical tools and

[22] Yarin P. (2000) Towards the Distributed Visualization of Usage History, Master of Science Media Arts and Sciences Thesis, MIT, p. 9

dynamic whole-body interactions in physical space. HandSCAPE provides an exiting new way to measure objects in physical space in real-time. This instant feedback reinforces the means of understanding physical space in the digital domain, not just through measurements, but also through perceptual, tactile, and kinesthetic feedback. As opposed to designing a general-purpose interface, there is a clear advantage in designing interfaces that are application-specific and tightly related to precise tasks. As a solution for these specific real world measuring tasks, HandSCAPE provides an on-site comprehensive interaction, which enhances the consistency of the measuring workload by augmenting the existing interface of the tape measure.

6.1.2 Augmenting Everyday Tools

"Things have rights to 1) have an identity, 2) access other objects, and 3) detect the nature of their environment."[23] These intelligent processes carefully map the properties of physical systems with digital information and functionality. We realized that making use of existing physical tools enables people to correlate new interactions with those that are commonly employed in everyday life.



The property of tape measure is too obvious. Whenever we use a tape measure, we already know what it is, what is used for, and how it works. The generic identity of measuring tasks and shapes is what people are accustomed to. In the beginning of our investigation, we first set out to create a new interface in which we tried to augment the existing design and create a new type of tool. In our attempt, we fit interaction and industrial design to the existing tape measure without requiring new users to learn and to adjust the new interface. Thus, this motivation led us to look closer to the tape measure in terms of shape, scale, and functionality.

[23] Gershenfeld, N., (1999) When Thinks Starts to Think, New York:: Henry and Holt, pp.104-106

When people use tools, they use one or two hands, or other body parts as well. A necessary two-handed interaction uses the natural way of using motion to recognize 3D spaces and objects. When interacting with three-dimensional space, using two hands helps to make spatial input comprehensible to the user. "Using both hands helps users ground themselves in a body-relative interaction space, as opposed to requiring consciously calculated action in an abstract environment–relative space."[24]

Through its functionality, HandSCAPE obtains field measurements efficiently and allows the user to display the scale of objects and spaces in the digital domain, while the user keeps the sensation of physical scale in his or her body. Thus, a measured object in physical space is identical to a visualized model on the computer screen in real-time. This cognitive feedback between physical space and virtual space provides a consistency of sharing the scale of object measured and rendered at the same time. Consequently, the digitally constructed physical space on a computer is not just a visual representation, but also conveys the user's sense of scale (human-scale) mapped onto virtual space.

6.1.3 Digitally Constructed Physical Space

People construct a particular space in their internal representation through various cognitive mappings such as perceptual (visual, audible, touch), sensorimotor systems (eyes, ears, head-movement), and whole-body locomotion. "According to the spatial perception, processes to construct spatial knowledge can be described through primarily view action, the deployment of sensing, and directly engaging their physical body in the space."[25] In fact, the relationship between people and space in a broad and generalized aspect of physical and psychological perceptions can

[24] Mckeinzie, Soukoreff, R. W, Payl, C. A (1997) Two Ball Mouse Affords Three Degrees of Freedom, In Proceedings of CHI '97, ACM Press

be classified. The following diagram [Table 4] illustrates this relationship between people and space, derived from literature in human perception of space as a broad and generalized aspect of various disciplines such as social, ergonomics, behavioral, and aesthetics.

	People	**Space**
	Human Factors; performance with a system environment behavioral variables, anthropometric measurements, and ergonomics	**Architectural space**; human inhabitance, dwelling, standard space usage
Physical Perception	**Five Senses** Visual, Audible, Olfactory, Tactile, and Taste Receptors in distance **Visual Perception** Perceptual space (Polar Coordinates, Direction – Distance) Physical space (Cartesian Space with Orthogonal axes)	**Space Definition** Ground, Platform, Pit, Marker, Focus, Barrier, Roof, Columns, Path, and Openings **Biological needs in space** Orientation, Time, Consistency of light, Sunlight, View, Focus, and Territory
Psychological Perception	**Spatial Organization** Proximity, Similarity, Continuity, Common fate, and Closure Lived-Body Presence of things and body, Sense of extension in tactual experience	**Color and light of space** *Perception of volume, weight and size, estimation of time, temperature, noise and sound* **Dynamism of space** Intermediate, Private, Social, and Public Wayfinding and Navigation in space Sense of Direction, Cognitive Maps

Table 4. People in Space

"Both physical world and perceptual world have structure. The structure of these worlds and the relationships between the structures can be described by geometry." [26] In fact, an object can be displaced or rotated without deformation in physical space. It is different in visual (perceptive) space in which the geometry must be determined empirically and direction and distance are the qualities of perceived space from the pattern of stimulation and the perceptions resulting from the stimulation. With HandSCAPE, we attempted to bridge the gap between physical and perceptual space through a computational mechanism for visualizing three-dimensional objects and space. In

[25] Plesniak, W., (2001) Haptic holography: an early computational plastic, ph.D thesis in the MIT Program of Media Arts and Sciences, p. 28
[26] Hershenson, M., (1999) Visual Space Perception, Cambridge: MIT Press, pp. 1-6

conjunction with physical measuring, the user brings into action all of his or her body part and sequentially translates the measurements to the digital domain. Generally, "body-space" is the physical space close to the human body where user interaction with the environment occurs with perceptual, tactile, and kinesthetic feedback. In HandSCAPE, "body-space" is a spatial mechanism to both perceive and perform the visualization of three-dimensional objects and spaces in conjunction with their physical measuring. This conceptual support helps the user to seamlessly bridge both physical and virtual space.

6.2 Future Work

Throughout the process of developing the HandSCAPE system, we have explored several phases: initial implementation, various exploratory usages, and system refinement. However much work is left to verify the system in order to reach to mass production. With a field test of the final system, potential directions of future research of the HandSCAPE system are outlined below:

6.2.1 Augmented Reality

Although there are benefits of bridging measuring and modeling as a single step, many people are concerned about the looped interaction between input and output. Since HandSCAPE displays 3D models in a computer simulation resulting from physical measuring, the user needs to monitor the results on computer nearby. This could be an additional effort, which is new to workers. Large-scale display can be considered to integrate the two separated tasks. However, it is not suited for outdoor space. Using wearable HMD (Head-Mounted Display) device can also be incorporated with the HandSCAPE system so that the user can see the results of what they measure and simultaneously perform his or her task according to the result. However, this augmented environment would be difficult to recommend to field workers and is often expensive.

An alternative idea being pursued is to attach a compact LCD display on top of the handheld module, so that the user can monitor what they measure. Even adding a small picture camera may help to capture real images incorporated with the measurements. This requires a large number of advanced technologies that should be integrated in a handheld scale. Imagine the handheld module can interact with a host computer nearby for both sending the measurement data and for displaying the information at the same time. Another idea being pursued is the integration of a GPS (Global Positioning System) device into the handheld module. The geographical information of location could link to a map program to browse the map viewed automatically depending on location. The commercially available GPS resolution is up to ten meters. This is not accurate enough to define objects in microscopic measurements such as an archaeological excavation. However, it would be useful to link multiple sites of a large excavation region.

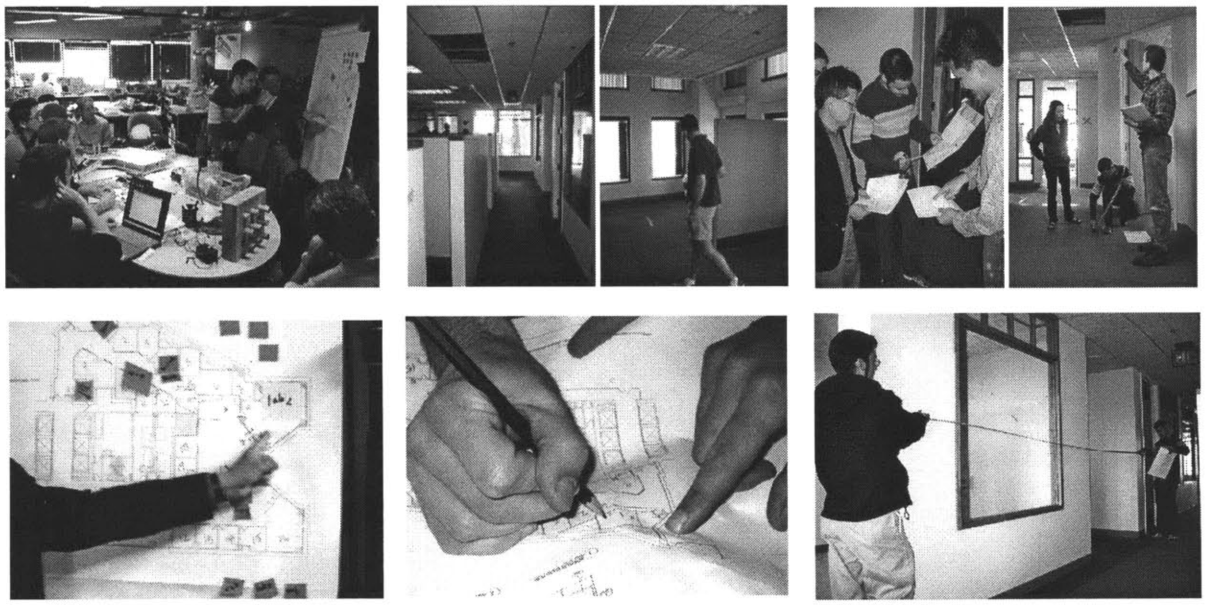
6.2.2 Data Retrieval and Manipulation

To provide a single source of access to information regarding the situated measuring tasks, data such as the size of the space, or a reference point of the space could manually be entered via a keyboard and a mouse. In the current HandSCAPE system, the digital model is only generated and displayed on the computer screen by the measuring input. Once it's reset, the data is lost. This could be improved by linking to a database. By giving the 3D model in the virtual space some sort of legitimate existence rather than merely defining it as boxes, the user could navigate through the site entirely electronically. The user would not be concerned about referring to another source for images and notes and trying to figure out which box represents what. This could involve clicking on a box to bring up other information or it could involve pasting pictures on the sides of the boxes. This is the next step for the project, as it would greatly increase the interactive usability of the system.

The data could also be available for storage, retrieval, and manipulation. These GUIs (Graphical User Interface) could be implemented for application-specific software packages. For any serious use of this software, the users could save the scene when finished working for the day and restore it to continue work the next day. The visualization of the measuring process could also enable those outside of the field to better understand the course of measuring and contextualizing the situated tasks. This significantly will improve collaboration between on-site and laboratory research. Another idea is to tag the measurement information on physical object, artifact, or box. Using RF ID tag, this can track objects being measured and readable for the purpose of monitoring and arrangement. This will provide a collaborative work between physical space and digital information in real world applications

6.2.3 Future Application: Space Planning

Among many other possible real world applications, applying the HandSCAPE system to a large-scale complex measuring and modeling task of space planning could be interesting. Space planning enhances the function and quality of interior spaces for the purpose of improving the quality of life, increasing productivity, and protecting the health, safety, and welfare of the public. The whole process of space planning is a multi-step procedure that requires a long period of time to experiment with changes prior to construction and needs a large number of people, such as directors, contractors, space planners, architects, constructors, and movers. The majority of the work includes executive planning, interior design, construction, and occupation.



Usually the stages of a space planning begin with 1) getting initial blueprint for the new spaces, 2) measuring the actual space that needs to be renovated, and 3) having dynamic group discussions for renovation and moving-in. Although people have diagrams, written measurements, photographs, and detailed information about the new space, the usual next step is to travel on-site to determine or revise their primary plan with additional sketches, notes, and measurements. More often the whole process is also changed during the implementation and upon completion. With HandSCAPE, a fully interactive on-site 3D simulation of the new space can be generated and the field data can be archived automatically. This dynamically updated information could allow people from executives to constructors to share the information available on-line. Future features of

HandSCAPE may be used for this application so that the whole system becomes completely accessible from multi-platforms and tasks.

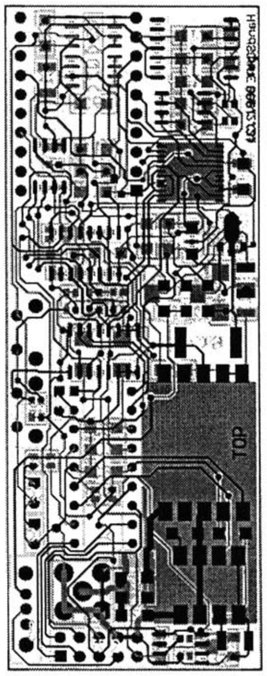
7.0 Conclusion

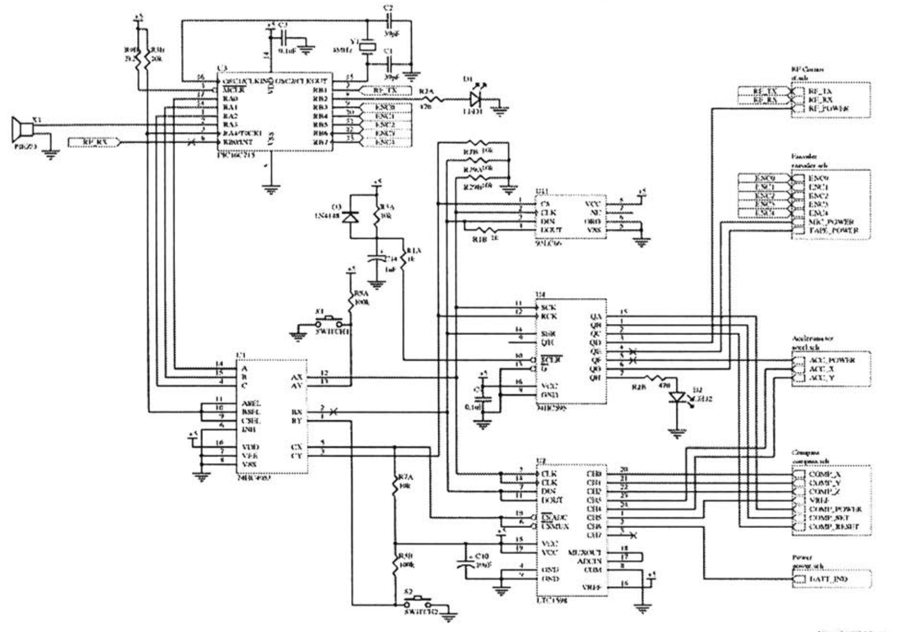
Through this design exploration, I have introduced an innovative and tangible computational measuring system, called HandSCAPE as a means of bridging physical measuring and computer modeling. The HandSCAPE system provides the user with a simple (handheld tool) and comprehensible (human-body scale) interface to generate digital models of physical objects. The result from combining measuring and modeling enables the user to obtain timely and accurate on-site measurements. By applying HandSCAPE in three application-specific simulations, multiple interactions of system refinement and configuration have yielded a system that has demonstrated excellent performance in each of the application environments explored.

Additionally, I have presented the concept of "*Digitally Constructed Physical Space*," a computational mechanism for visualizing three-dimensional models in conjunction with physical measuring in human body scale. Such conceptualization can be examined in two ways. The first approach is to increase computational effectiveness of immediate access to digital information in the manner of traditional interaction techniques. The other is to establish its validity and innovation mapped by combining perceptual sensing (simulated scenario) and physical interaction (task performance). Since these approaches are steps toward handling the complexity of real world problems that are often unpredictable, the number of questions raised from each application have not yet been clearly articulated and implemented. However, these approaches embodied in the HandSCAPE system presents a shift of design that builds upon the notion of breaking out of the "solution-in-a box" appliance mentality that dominates current practice. To reap the benefits that this approach offers will also serve to inform future work in this area.

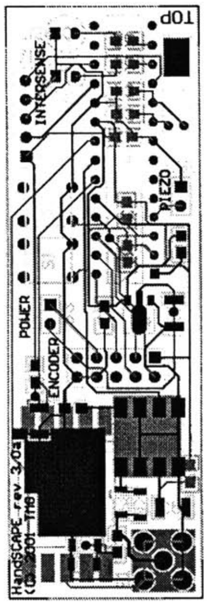
A Appendix : HandSCAPE Custom Electronics

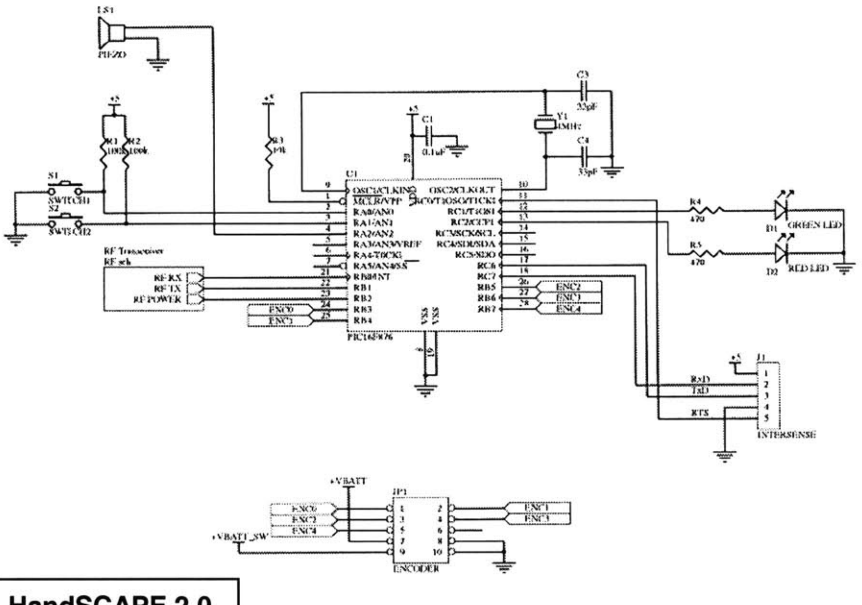
A.1- PCB Circuit Design





HandSCAPE 1.0





HandSCAPE 2.0

A.2 Electronic Parts (HandSCAPE v. 1.0)

Part Type	Footprint	Description	Qty
0.1uF	1206		8
0.22uF	1206		1
1.0uF	TANT_CAP_A		2
10	1206		2
100	1206		1
100k	1206		2
100k	RES743/2		3
100k	POT_3214W		1
100k 1%	1206		1
10k	RES743/2		3
10uF	TANT_CAP_B		4
125k	1206		1
1k	1206		1
1k	RES743/2		1
1M 1%	1206		1
1N4148	DL-35		5
1N5817	MELF		1
21k7 1%	1206		1
22k	RES743/2		1
2k2	RES743/2		1
2N7002	SOT-23_FET		1
39pF	1206		2
4.7uF	TANT_CAP_B		1
470	RES743/2		1
470	1206		1
47nF	1206		2
47uF	TANT_CAP_D		2
4MHz	CSM-12		1
620	1206		1
74HC4053	SO-16		1
74HC595	SO-16		1
82uH	D73C/D75C		1
845k 1%	1206		1
93LC66	SO-8	Microwire serial EEPROM	1
ADXL202	SOL-14	Two-axis accelerometer	1
ENCODER	HEADER, 2x5, 0.1"		1
HMC2003	DIP20/600/42	Compass magnetometer	1
LC_RXM	LC_RXM		1
LC_TXM	LC_TXM		1
LED1	HEADER, LED		1
LED2	HEADER, LED		1
LMC6492	SO-8		2
LT1073	SO-8	Switching DC-DC converter	1
LTC1598	SSOP-24	8-channel 12-bit ADC	1
NDS9952A	SO-8	Logic Level Dual N- and P-Channel MOSFET	2
PIC16C715	DIP18	8-Bit CMOS EEPROM Microcontrollers	1
POWER	SW_RT_SLIDE	Slide switch	1
PIEZO	HEADER, 1x2, 0.1"	Piezo buzzer	1

R1	1206		1
R2	1206		1
R3	1206		1
SMA_VERT	SMA_VERT	Reverse SMA antenna connector	1
SWITCH1	SW_RT_TACT	Tact switch	2
TC55RP05	SOT-89	Micropower voltage regulator	1

A.3 - Firmware version 2.0 (interacting with InterTrack2)

```
/*
 * HAND20.C
 * HandSCAPE firmware
 *
 * Handheld orientation-aware digital measuring tape
 *
 * Device: PIC16F876
 * Resource allocation:
 *  port A (RA0) : Switch input 1
 *        (RA1) : Switch input 2
 *        (RA2) : Piezo speaker
 *  port B (RB0) : RF receive data
 *        (RB1) : RF transmit data
 *        (RB2) : RF power control
 *        (RB3) : Linear encoder input bit 0
 *        (RB4) : Linear encoder input bit 1
 *        (RB5) : Linear encoder input bit 2
 *        (RB6) : Linear encoder input bit 3
 *        (RB7) : Linear encoder input bit 4
 *        (RC0) : RS-232 RTS (to Intersense sensor)
 *        (RC1) : Green LED
 *        (RC2) : Red LED
 *        (RC6) : RS-232 TxD (to Intersense sensor)
 *        (RC7) : RS-232 RxD (to Intersense sensor)
 *
 * The data packet format is as follows:
 *  4 START bytes
 *
 *  HEADER byte, bits 0-3   ARQ packet sequence number
 *  HEADER byte, bits 4-7   fixed nibble
 *
 *  DATA byte 0, bits 0-3   node identification code
 *  DATA byte 0, bit 4      sampling mode bit
 *  DATA byte 1, bits 0-7   8 MSB of encoder measurement
 *  DATA byte 2, bits 0-7   8 LSB of encoder measurement
 *  DATA byte 3, bits 0-7   8 MSB of x-axis (elevation)
measurement
 *  DATA byte 4, bits 0-7   8 LSB of x-axis (elevation)
measurement
 *  DATA byte 5, bits 0-7   8 MSB of y-axis (azimuth)
measurement
 *  DATA byte 6, bits 0-7   8 LSB of y-axis (azimuth)
measurement
 *  DATA byte 7, bits 0-7   8 MSB of z-axis (twist)
measurement
 *  DATA byte 8, bits 0-7   8 LSB of z-axis (twist)
measurement
 *
 *  CRC high byte
 *  CRC low byte
 *
 * Modification Log
 *
=========================================
 *  1.0  24 Jun 1998   vcs   initial version
 *  1.1  13 Oct 1998   vcs   added shoe board extensions
 *  1.2   1 Nov 1998   vcs   buttonless UI (auto-sample after
delay)
 *  1.3   3 Nov 1998   vcs   removed zero mode
 *  1.4  17 Nov 1998   vcs   added space/object mode
 *                           selectable beeper tones
 *                           added serial commands
 *                           added fast I/O
 *  1.5  25 Mar 1999   vcs   supports hardware revision 2.0
 *  1.6  14 Apr 1999   vcs   added support for I2C ext
EEPROM (rev 2.1)
 *  1.7  18 Apr 1999   vcs   supports hardware revision 2.2
 *  2.0  22 Apr 2001   vcs   supports hardware revision 3.0
 *
 */


#fuses XT, NOWDT, NOPROTECT

//#include <16C73A.H>
#include <16F876.H>

#use delay (clock=4000000)
#use fast_io (a)
#use fast_io (b)
#use fast_io (c)

// Set software switches
#define _PRED_LOOKUP

#define SWITCH1        PIN_A0
#define SWITCH2        PIN_A1
#define PIEZO          PIN_A2
#define RF_RX          PIN_B0
#define RF_TX          PIN_B1
#define RF_POWER       PIN_B2
#define SER_RTS        PIN_C0
#define GREEN_LED      PIN_C1
#define RED_LED        PIN_C2
#define SER_TX         PIN_C6
#define SER_RX         PIN_C7

#define ASSERT_GREEN_LED   output_high
(GREEN_LED)
```

```
#define DEASSERT_GREEN_LED output_low
(GREEN_LED)

#define ASSERT_RED_LED    output_high (RED_LED)
#define DEASSERT_RED_LED   output_low (RED_LED)

#define ASSERT_RF        output_high (RF_TX)
#define DEASSERT_RF        output_low (RF_TX)

#define RF_RX_ON        output_high (RF_POWER)
#define RF_RX_OFF        output_low (RF_POWER)

#define ASSERT_RTS       output_high (SER_RTS)
#define DEASSERT_RTS       output_low (SER_RTS)

#define RF_ASSERTED       input (RF_RX)
#define SW1_ASSERTED       !input (SWITCH1)
#define SW2_ASSERTED       !input (SWITCH2)

#define START          0xAA
#define HEADER           0x50
#define ACK           0x53

#define NODE_ID          0      // 4-bit node ID
#define SLEEP_DELAY        458    // Inactivity power down
delay in 65.536 ms increments
#define SAMP_ARRAY_SIZE   9      // Sample array size
#define RECV_BUFF_SIZE    6      // Serial receive buffer
size
#define TX_TIMEOUT         3      // No. of transmit retries
before timeout
#define RX_TIMEOUT        100     // Response timeout in
500 us increments
#define START_BIT_TIMEOUT  50      // Start bit timeout in
10 us increments
#define SER_RECV_TIMEOUT  50      // Serial receive
timeout in 10 ms increments
#include "asm_def.h"              // Assembly defines header file
#include "lin_enc.c"             // Linear encoder driver
#include "bit_math.c"             // Bit math library
//
============================================
// Global variables
//

byte rfRecvData = 0;
byte bytesReceived = 0;

signed long gCounter = 0;
signed long lastgCounter = 0;
unsigned long idleCounter = 0;

byte sampleArray[SAMP_ARRAY_SIZE];
byte recvBuffer[RECV_BUFF_SIZE];



//
============================================
// Forward declarations
//

void Initialize (void);

void Beep_Low (void);
void Beep_High (void);

byte Resample (void);

void Receive_RF (void);
void Send_RF (byte data);
byte Transmit_Measurements (void);

void Receive_Serial (void);
void Send_Serial (byte data);

byte Initialize_Intersense (void);
void Zero_Intersense (void);
byte Sample_Intersense (void);

void Service_RTCC (void);



//============================================
// Initialization
//

void Initialize ()
{
  byte err;

  // Power-on delay
  delay_ms (500);

  // Disable all interrupts for initialization
  disable_interrupts (GLOBAL);

  // Set tri-state for ports
  set_tris_a (0x03);     // 0000 0011
  set_tris_b (0xF9);     // 1111 1001
  set_tris_c (0x80);     // 1000 0000

  // Configure port A for digital I/O
  setup_adc_ports (NO_ANALOGS);

  // Set node ID
  sampleArray[0] = NODE_ID;

  // Make sure RF transmitter is off
  DEASSERT_RF;

  // Make sure both LEDs are off
  DEASSERT_RED_LED;
  DEASSERT_GREEN_LED;

  // At 4MHz, RTCC increments at 256us intervals
  setup_counters (RTCC_INTERNAL, RTCC_DIV_256);
  set_rtcc (0);

  // Configure interrupt edge for EXT
  ext_int_edge (L_TO_H);

  // Initialize encoder
  Initialize_Encoder ();

  // Turn on the RDA interrupt to initialize the Intersense
sensor
  enable_interrupts (INT_RDA);
  enable_interrupts (GLOBAL);

  err = Initialize_Intersense ();

  while (err) {
    // Flash red LED on initialization error
    ASSERT_RED_LED;
    delay_ms (300);
    DEASSERT_RED_LED;

    delay_ms (1000);

    // Attempt to reinitialize
    err = Initialize_Intersense ();
  }

  // Zero the Intersense
  Zero_Intersense ();

  // Light green LED if initialization succeeds
  ASSERT_GREEN_LED;

  // Two tone beep
  Beep_Low ();
  Beep_High ();

  // Turn off LED's
  DEASSERT_GREEN_LED;
  DEASSERT_RED_LED;

  // Enable remaining interrupts
  enable_interrupts (INT_RTCC);
}


//
============================================
// Sound output
//

// Low beep
void Beep_Low ()
{
  byte cycleCounter;

  // 1/8 second 880 Hz beep
  for (cycleCounter = 0; cycleCounter < 110; cycleCounter++)
{
    output_high (PIEZO);
    delay_us (568);
    output_low (PIEZO);
    delay_us (568);
  }
}

// High beep
void Beep_High ()
{
  byte cycleCounter;

  // 1/8 second 1200 Hz beep
  for (cycleCounter = 0; cycleCounter < 150;
cycleCounter++) {
    output_high (PIEZO);
    delay_us (417);
    output_low (PIEZO);
    delay_us (417);
  }
}


//
============================================
// Sampling
//

byte Resample ()
{
  unsigned long value;
  byte sampleCounter;
  byte err;


  // Rezero the encoder counter if it's less than zero
  if (gCounter < 0) gCounter = 0;
  value = (unsigned long) gCounter;
  sampleArray[1] = (byte) (value >> 8);
  sampleArray[2] = (byte) value;

  // Read the Intersense sensor
  err = Sample_Intersense ();

  return err;
}


//
============================================
// RF Communications
//

#use rs232 (baud=4800, xmit=RF_TX, rcv=RF_RX,
INVERT)

// This ISR is executed when RF data is being received
#INT_EXT
void Receive_RF (void)
{
  static byte startBitTimer = 0;

  // Sometimes the RF receiver will produce a false logic
high, which may be
  // triggered by noise or the local RF transmitter.  To prevent
blocking of
  // getc() in this case, RF data has a mark and space bit with
timeouts
```

```
  // followed by the serial data.

  // Wait for mark to end
  while ((++startBitTimer < START_BIT_TIMEOUT) &&
(RF_ASSERTED)) {
    delay_us (10);
  }

  if (startBitTimer < START_BIT_TIMEOUT) {      // Valid
mark
    startBitTimer = 0;

    // Wait for space to end
    while ((++startBitTimer < START_BIT_TIMEOUT) &&
(!RF_ASSERTED)) {
      delay_us (10);
    }

    if (startBitTimer < START_BIT_TIMEOUT) {   // Valid
space
      rfRecvData = getc ();              // Receive data
    }
  }

  startBitTimer = 0;
}


// Send RF data
void Send_RF (byte data)
{
  // Send extra start bit
  ASSERT_RF;
  delay_us (208);
  DEASSERT_RF;
  delay_us (208);

  // Send data
  putc (data);
}


// Transmit sample array over RF channel with auto-
retransmission
int Transmit_Measurements ()
{
  byte success = 0;
  byte arrayIndex;
  byte txRetries = 0;
  byte rxTimer = 0;
  unsigned long crc;

  // Power-up the RF receiver
  RF_RX_ON;

  // Compute CRC for error checking
  crc = compute_crc16_int (sampleArray,
 SAMP_ARRAY_SIZE);

  // Transmit, verify, and retry if necessary
  while (txRetries < TX_TIMEOUT) {
    // Send start bytes
    Send_RF (START);
    Send_RF (START);
    Send_RF (START);
    Send_RF (START);

    // Send header byte and sequence number
    Send_RF (HEADER | txRetries);

    // Send sample array with zero balancing
    for (arrayIndex = 0; arrayIndex < SAMP_ARRAY_SIZE;
arrayIndex++) {
      if (bit_test (arrayIndex, 0)) {        // Odd numbered
bytes only
        Send_RF (sampleArray[arrayIndex] ^ 0xFF);
      } else {
        Send_RF (sampleArray[arrayIndex]);
      }
    }

    // Depending on if SAMP_ARRAY_SIZE is even or odd,
the high or low byte
    // of the CRC should be zero balanced
    Send_RF (((byte) (crc >> 8)) ^ 0xFF);
    Send_RF ((byte) crc);

    // Make sure transmitter is off
    DEASSERT_RF;

    // Turn on EXT interrupt to receive RF data
    #asm
    BCF  _INTCON, _INTF
    #endasm

    enable_interrupts (INT_EXT);

    // Wait for acknowledgement byte
    // Note that base station sends four start bytes which are
ignored
    // but are used to allow the receiver to settle
    while (++rxTimer < RX_TIMEOUT) {  // Timeout timer
      if (rfRecvData == ACK) {
        success = 1;              // Set flag
        rxTimer = RX_TIMEOUT;      // Exit loop
      } else {
        delay_us (500);
      }
    }

    // Turn off interrupt
    disable_interrupts (INT_EXT);

    // Reset loop variables
    rfRecvData = 0;
    rxTimer = 0;

    // Check result flag
    if (success) {
      txRetries = TX_TIMEOUT;      // Exit loop
    } else {
      // Device retransmits in the next window after a collision
      // with 1/2 probability
      if (get_random (0) > 127) delay_ms (65);
      // Increment retry counter
      txRetries++;
    }
  }

  // Shutdown the RF receiver
  RF_RX_OFF;

  return success;
}
//
==============================================
// Serial Communications (for Intersense sensor)
//

#use rs232 (baud=1200, xmit=SER_TX, rcv=SER_RX)

#INT_RDA
void Receive_Serial ()
{
  recvBuffer[bytesReceived] = getc ();

  if (bytesReceived < RECV_BUFF_SIZE) {
    bytesReceived++;
  }
}


void Send_Serial (byte data)
{
  putc (data);
}


//
==============================================
// Intersense sensor routines
//

byte Initialize_Intersense ()
{
  unsigned long timeoutTimer;


  // Assert the RTS line for at least 100 ms to reset
  ASSERT_RTS;
  delay_ms (200);
  DEASSERT_RTS;
  delay_ms (200);

  // Command mode baud rate is 1200
  set_uart_speed (1200);

  // Reset variables
  bytesReceived = 0;
  timeoutTimer = 0;

  // Enter command mode
  Send_Serial ('c');

  // Wait for response
  while ((bytesReceived == 0) && (++timeoutTimer !=
SER_RECV_TIMEOUT)) {
    delay_ms (1);
  }

  if (timeoutTimer == SER_RECV_TIMEOUT) {
    return 1;
  } else {
    // Now check response
    if (recvBuffer[0] != 'o') {
      return 1;
    }
  }

  // Reset variables
  bytesReceived = 0;
  timeoutTimer = 0;

  // Set data baud rate to 4800
  delay_ms (50);
  Send_Serial ('b');
  delay_ms (50);
  Send_Serial (1);

  // Wait for response
  while ((bytesReceived == 0) && (++timeoutTimer !=
SER_RECV_TIMEOUT)) {
    delay_ms (1);
  }

  if (timeoutTimer == SER_RECV_TIMEOUT) {
    return 1;
  } else {
    // Now check response
    if (recvBuffer[0] != 'b') {
      return 1;
    }
  }

  // Reset variables
  bytesReceived = 0;
  timeoutTimer = 0;

  // Enable data mode
  delay_ms (50);
  Send_Serial ('d');

  // Wait for response
  while ((bytesReceived == 0) && (++timeoutTimer !=
SER_RECV_TIMEOUT)) {
    delay_ms (1);
  }

  if (timeoutTimer == SER_RECV_TIMEOUT) {
    return 1;
  } else {
    // Now check response
    if (recvBuffer[0] != 'd') {
      return 1;
    }
  }
  // Change to data baud rate
  set_uart_speed (4800);

  delay_ms (50);

  return 0;
}

void Zero_Intersense ()
{
  unsigned long timeoutTimer;

  // Reset variables
  bytesReceived = 0;
  timeoutTimer = 0;

  // Send reset angles command
  Send_Serial ('r');
  delay_ms(50);

  // Send 0 for device number
  Send_Serial (0);
  delay_ms(50);

  // Set the azimuth angle
  Send_Serial (0);
  delay_ms(50);
  Send_Serial (0);
  delay_ms(50);
  // Set the elevation angle
  Send_Serial (0);
  delay_ms(50);
  Send_Serial (0);
  delay_ms(50);

  // Set the twist angle
  Send_Serial (0);
  delay_ms(50);
  Send_Serial (0);

  while ((bytesReceived != 4) && (++timeoutTimer !=
(SER_RECV_TIMEOUT * 4))) {
    delay_ms (1);
  }
}


byte Sample_Intersense ()
{
  unsigned long timeoutTimer;


  // Reset variables
  bytesReceived = 0;
  timeoutTimer = 0;

  // Send 3DOF sample command
  Send_Serial ('3');

  while ((bytesReceived != 6) && (++timeoutTimer !=
(SER_RECV_TIMEOUT * 6))) {
    delay_ms (1);
  }

  if (timeoutTimer == (SER_RECV_TIMEOUT * 6)) {
    bytesReceived = 0;

    sampleArray[3] = 0;
    sampleArray[4] = 0;
    sampleArray[5] = 0;
    sampleArray[6] = 0;
    sampleArray[7] = 0;
    sampleArray[8] = 0;

    return 1;

  } else {
    bytesReceived = 0;

    // Copy the received data to sample array
    sampleArray[3] = recvBuffer[3];
    sampleArray[4] = recvBuffer[2];
    sampleArray[5] = recvBuffer[1];
    sampleArray[6] = recvBuffer[0];
    sampleArray[7] = recvBuffer[5];
    sampleArray[8] = recvBuffer[4];

    return 0;

  }
}

//
=============================================
// Interrupt service routine
//

// This ISR controls power-down during periods of inactivity
// Execution occurs at 65.536ms intervals
#INT_RTCC
void Service_RTCC ()
{
  set_rtcc (0);

  if (lastgCounter == gCounter) {
    idleCounter++;
    DEASSERT_GREEN_LED;
  } else {
    idleCounter = 0;
    ASSERT_GREEN_LED;
  }
  lastgCounter = gCounter;

  // Auto power-down after a delay of SLEEP_DELAY
//  if (idleCounter > SLEEP_DELAY) {
//    idleCounter = 0;
//    Suspend ();
//  }
}


//
=============================================
// Main
```

```
//

void Main ()
{
  Initialize ();
  while (1) {

    // Poll and accumulate encoder
    gCounter = gCounter + (signed long) Read_Encoder ();

    // Check for switch 1 depression
    if (SW1_ASSERTED) {
      disable_interrupts (INT_RTCC);

      ASSERT_GREEN_LED;
      bit_clear (sampleArray[0], 4);       // Sample mode bit

      if (Resample () == 0) {
        if (Transmit_Measurements ()) {
          DEASSERT_RED_LED;
          Beep_High ();
        } else {
          ASSERT_RED_LED;
        }

        delay_ms (200);
        DEASSERT_GREEN_LED;

      } else {
        // Reinitialize Intersense sensor
        Initialize_Intersense ();

      }

      enable_interrupts (INT_RTCC);

    // Check for switch 2 depression
    } else if (SW2_ASSERTED) {
      disable_interrupts (INT_RTCC);

      ASSERT_GREEN_LED;
      bit_set (sampleArray[0], 4);      // Sample mode bit

      if (Resample () == 0) {
        if (Transmit_Measurements ()) {
          DEASSERT_RED_LED;
          Beep_Low ();
        } else {
          ASSERT_RED_LED;
        }

        delay_ms (200);
        DEASSERT_GREEN_LED;

      } else {
        // Reinitialize Intersense sensor
        Initialize_Intersense ();

      }

      enable_interrupts (INT_RTCC);
    }

    // Sleep if flag set
//    if (powerDownFlag) sleep ();
  }
}


=============================================
=
```

B Appendix: API Software

B.1 HandSCAPE v. 2.0 (Serial Reading)

```
#include "serial.h"

#ifdef WIN32
#include "print.h"
#endif

HANDLE hCommPort;
COMMTIMEOUTS ctmoCommPort;
DCB dcbCommPort;
DWORD dwCount;
char c[42];
BOOL okay;


void Open_Serial_Port()
{
  hCommPort = CreateFile("COM1", GENERIC_READ, 0, NULL, OPEN_EXISTING,
         FILE_ATTRIBUTE_NORMAL, NULL);
  if (hCommPort == INVALID_HANDLE_VALUE)
         OutputDebugString("crash!");
  else
         OutputDebugString("successful!");

  ctmoCommPort.ReadIntervalTimeout = 0;
  ctmoCommPort.ReadTotalTimeoutMultiplier = 0;
  ctmoCommPort.ReadTotalTimeoutConstant = 0;
  ctmoCommPort.WriteTotalTimeoutMultiplier = MAXDWORD;
  ctmoCommPort.WriteTotalTimeoutConstant = MAXDWORD;
  SetCommTimeouts(hCommPort, &ctmoCommPort);

  dcbCommPort.DCBlength = sizeof(DCB);
  dcbCommPort.BaudRate = CBR_9600;
  dcbCommPort.ByteSize = 8;             // data size, xmit, and rcv
  dcbCommPort.Parity = NOPARITY;        // no parity bit
  dcbCommPort.StopBits = ONESTOPBIT;    // one stop bit
  dcbCommPort.fOutX = 0;
  dcbCommPort.fInX = 0;
  dcbCommPort.fRtsControl = RTS_CONTROL_DISABLE;
  dcbCommPort.fOutxCtsFlow = 0;
  dcbCommPort.fOutxDsrFlow = 0;
  dcbCommPort.fDtrControl = DTR_CONTROL_DISABLE;
  SetCommState(hCommPort, &dcbCommPort);

}

void auto_get_measurements(int& en, float& pitch, float &heading)
{
  srand((unsigned)time(NULL));
  pitch = -(rand() % 91) * M_PI/180.0;
  heading = (rand() % 91) * M_PI/180.0;
  en = (rand() % 150) + 15.;
}

/**
* Reads measurements in from serial port.
* Format of input:
* /IDaa/MOb/ENcccc/AXddddd/AYeeeee/AZfffff\n\r
* <aa>           4-bit device ID
* <b>            sample mode bit
* <cccc> encoder count
* <ddddd>unsigned 16-bit elevation measurement
* <eeeee>unsigned 16-bit azimuth measurement
* <fffff>unsigned 16-bit twist measurement
```

```
*/
int get_measurements(int& en, float& pitch, float& roll, float& heading)
{
  int mo, ax, ay, az;
  int okay = 0;
  char inputchar[1];

  while (okay == 0)
  {
        ReadFile(hCommPort, c, 42, &dwCount, NULL);
        if (dwCount == 42)
        {
                printf("%s", c);
                if (measureError(c) != 1)
                {
                        mo = measure(c, 'M', 'O');
                        en = measure(c, 'E', 'N')/10. - 1.;
                        if (en != 0)
                        {
                                ax = measure(c, 'A', 'X');
                                ay = measure(c, 'A', 'Y');
                                az = measure(c, 'A', 'Z');
                                okay = 1;
                        }
                }
        }
  }

  printf("en: %d\nax: %d\nay: %d\naz: %d\n", en, ax, ay, az);

  pitch = Find_Pitch_Angle(ax);
  heading = Find_Heading(ay);
  roll = Find_Roll_Angle(az);

  printf("pitch: %f\n", pitch * 180 / M_PI);
  printf("roll: %f\n", roll * 180 / M_PI);
  printf("heading: %f\n", heading * 180 / M_PI);

  return okay;

}
/**
* returns an integer representation of the number following c1c2 in buf
*/
int measure(char *buf, char c1, char c2)
{
  int i = 0;
  int result = 0;
  while ((buf[i] != c1) || (buf[i+1] != c2))
    {
      i = i + 1;
    }
  i = i + 2;
  while ((buf[i] != '/') && (buf[i] != '\n'))
    {
    int digit = ((int)buf[i]) - 48;
    if ((digit >=0) && (digit <= 9)) {
        result = result * 10 + digit;
    }
    i = i + 1;
    }
  return result;
}

/**
* returns 1 if measurement has ER (therefore, an error)
* else returns 0
*/
int measureError(char *buf)
{
  int i = 0;
  float result = 0;
  while (buf[i] != '\n')
  {
        if ((buf[i] == 'E') && (buf[i+1] == 'R'))
              return 1;
        else
              i++;
  }
  return 0;
}


/**
* returns pitch angle given elevation data
* ax is an int between 0 and 65534
* returns a float representing the radian measure of the pitch angle
*/
float Find_Pitch_Angle(int ax)
{
  return (float) ax/65534 * (2 * M_PI);
}

/**
* returns heading angle given azimuth data
* see Find_Pitch_Angle
*/
float Find_Heading(int ay)
{
  return (float) ay/65534 * (2 * M_PI);
}

/**
* returns roll angle given twist data
* see Find_Pitch_Angle
*/
float Find_Roll_Angle(int az)
{
  return (float) az/65534 * (2 * M_PI);

}

void ClearSerial()
{
  PurgeComm(hCommPort, PURGE_RXABORT);
  PurgeComm(hCommPort, PURGE_RXCLEAR);
}
```

B.2 HandSCAPE v. 1.0 (Relative 3D Positioning)

```
#include "theDig.h"

float dig_w = 400.;
float dig_h = 300.;
float dig_d = 100.;
float grids = 50.;
// as of right now, the border must be some whole multiple of grids
#define border 400.

extern int child;
extern int Ccamera;
extern int Cmaintrans;
extern int Csun;
extern int Cboxes;
extern int Cdig;
extern SoSeparator** boxes;

void createDig(SoSeparator *globalroot)
{
```

```
//create the first sub-root
SoSeparator *root = new SoSeparator;
//only need to reference the global root
//    root->ref();

  ///////////////// Create camera ////////////////////
  SoPerspectiveCamera *roomCamera = new SoPerspectiveCamera;
  roomCamera->position.setValue(0, 0, 4000);
roomCamera->focalDistance.setValue(4000.);
  roomCamera->nearDistance = 2000;
  roomCamera->farDistance = 6000;
//////////////

/////as of now, unnecessary
//    float orientAngle1 = 0;
//        roomCamera->orientation.setValue(SbVec3f(1.0, 0.0, 0.0), orientAngle1);
child = 0;
root->addChild(roomCamera);
Ccamera = child;
child++;


///////////////// translate scene ///////////////////
// or not...
SoTranslation *maintrans = new SoTranslation;
//maintrans->translation.setValue(-180, -130, 0);
root->addChild(maintrans);
Cmaintrans = child;
child++;


////////////////// add the sun //////////////////////
//the sun is on my right
  SoPointLight *theSun = new SoPointLight;
  theSun->location.setValue(700, 0, 2000);
  theSun->intensity = 1.0;
  root->addChild(theSun);
Csun = child;
child++;

//uh oh, no shadows, can I solve this?
//interesting code, why?
// ahh ha!  This incorporates the lighting concept from above (i.e. the sun)
// using BASE_COLOR instead of PHONG just uses the diffuse color of the objects
//   and does not consider the effects of lighting, PHONG is default so this is unnecessary
//    SoLightModel *roomLight = new SoLightModel;
//    roomLight->model = SoLightModel::PHONG; // versus BASE_COLOR


///////////////// create the hole //////////////////////
SoSeparator *theDig = new SoSeparator();

// the color of dirt (as best as I could tell
  SoMaterial *brown = new SoMaterial;
  brown->diffuseColor.setValue(.84f, .47f, .22f);
brown->transparency = 0;
SoMaterial *trans_grey = new SoMaterial;
trans_grey->diffuseColor.setValue(.4f, .4f, .4f);
trans_grey->transparency = .6f;
theDig->addChild(trans_grey);

//the vertices of the hole (currently #define, needs to come from measurements)
SoCoordinate3 *thePoints = new SoCoordinate3();
float mypoints[12][3] =
{
       {-dig_w/2., -dig_h/2., -dig_d},
       {dig_w/2., -dig_h/2., -dig_d},
       {dig_w/2., dig_h/2., -dig_d},
       {-dig_w/2., dig_h/2., -dig_d},
       {-dig_w/2., -dig_h/2., 0.},
       {dig_w/2., -dig_h/2., 0.},
        {dig_w/2., dig_h/2., 0.},
        {-dig_w/2., dig_h/2., 0.},
        {-dig_w/2.-border, -dig_h/2.-border, 0.},
        {dig_w/2.+border, -dig_h/2.-border, 0.},
        {dig_w/2.+border, dig_h/2.+border, 0.},
        {-dig_w/2.-border, dig_h/2.+border, 0.}

};
thePoints->point.setValues(0, 12, mypoints);
theDig->addChild(thePoints);

//define the faces of the hole
SoIndexedFaceSet *theFaces = new SoIndexedFaceSet();
int indices[9*5] =
{
        0, 1, 2, 3, -1,
        0, 4, 5, 1, -1,
        1, 5, 6, 2, -1,
        2, 6, 7, 3, -1,
        3, 7, 4, 0, -1,
        8, 9, 5, 4, -1,
        9, 10, 6, 5, -1,
        10, 11, 7, 6, -1,
        11, 8, 4, 7, -1
};
theFaces->coordIndex.setValues(0, 9*5, indices);
theDig->addChild(theFaces);

// add grid
SoSeparator *grid = new SoSeparator;
SoDrawStyle *lines = new SoDrawStyle;
lines->style = SoDrawStyle::LINES;
grid->addChild(lines);
SoMaterial *white = new SoMaterial;
white->diffuseColor.setValue(0., 0., 0.);
grid->addChild(white);
float i,j;

for (i=-dig_w/2.-grids; i<dig_w/2.+grids; i+=grids) {
        for (j=-dig_h/2.-grids; j<dig_h/2.+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., -dig_d);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// left top border
for (i=-dig_w/2.; i>-dig_w/2.-border-grids; i-=grids) {
        for (j=-dig_h/2.-border-grids; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i-grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// right top border
for (i=dig_w/2.; i<dig_w/2.+border+grids; i+=grids) {
        for (j=-dig_h/2.-border-grids; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// bottom top truncated border
for (i=-dig_w/2.; i<dig_w/2.; i+=grids) {
        for (j=-dig_h/2.; j>-dig_h/2.-border-grids; j-=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j-grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// top top truncated border
for (i=-dig_w/2.; i<dig_w/2.; i+=grids) {
        for (j=dig_h/2.; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
theDig->addChild(grid);

// the ball that denotes the anchor point
SoSeparator *anchor = new SoSeparator;
SoTranslation *anchor_trans = new SoTranslation;
anchor_trans->translation.setValue(-dig_w/2., -dig_h/2., 0.);
SoMaterial *anchor_col = new SoMaterial;
anchor_col->diffuseColor.setValue(.94f, 0.3f, 0.3f);
SoSphere *anchor_ball = new SoSphere;
anchor_ball->radius.setValue(5.);
anchor->addChild(anchor_trans);
anchor->addChild(anchor_col);
anchor->addChild(anchor_ball);
theDig->addChild(anchor);

// ah yes, it does exist
root->addChild(theDig);
Cdig = child;
child++;
Cboxes = child;


/////// initialize boxes ////////
for (int k=0;k<MAXBOXES;k++)
        boxes[k] = 0;


printf("have the root\n");
if (globalroot->getNumChildren() != 0)
        globalroot->replaceChild(0, root);
```

```
  else globalroot->addChild(root);
  printf("passed the root\n");

  return;
}



void deep(SoSeparator *globalroot) {

  SoSeparator *theDig = (SoSeparator *)(((SoSeparator *)globalroot->getChild(0))->getChild(Cdig));

  SoCoordinate3 *thePoints = new SoCoordinate3();
  float mypoints[18][3] =
  {
        {-dig_w/2., -dig_h/2., -2*dig_d},     //left down deep
        {0., -dig_h/2., -2*dig_d},            //mid down deep
        {0., dig_h/2., -2*dig_d},             //mid up deep
        {-dig_w/2., dig_h/2., -2*dig_d},      //left up deep
        {0., -dig_h/2., -dig_d},              //mid down in
        {dig_w/2., -dig_h/2., -dig_d},        //right down in
        {dig_w/2., dig_h/2., -dig_d}, //right up in
        {0., dig_h/2., -dig_d},                       //mid up in
        {-dig_w/2., -dig_h/2., 0.},           //left down out
        {dig_w/2., -dig_h/2., 0.},            //right down out
        {dig_w/2., dig_h/2., 0.},             //right up out
        {-dig_w/2., dig_h/2., 0.},            //left up out
        {-dig_w/2.-border, -dig_h/2.-border, 0.},     //border left down out
        {dig_w/2.+border, -dig_h/2.-border, 0.},      //border right down out
        {dig_w/2.+border, dig_h/2.+border, 0.},               //border right up out
        {-dig_w/2.-border, dig_h/2.+border, 0.},      //border left up out
        {0., -dig_h/2, 0.},                           //mid down out
        {0., dig_h/2, 0.}                             //mid up out
  };
  thePoints->point.setValues(0, 18, mypoints);
  theDig->replaceChild(1, thePoints);

  //define the faces of the hole
  SoIndexedFaceSet *theFaces = new SoIndexedFaceSet();
  int indices[13*5] =
  {
        0, 1, 2, 3, -1,
        2, 1, 4, 7, -1,
        4, 5, 6, 7, -1,
        6, 5, 9, 10, -1,
        0, 3, 11, 8, -1,
        1, 0, 8, 16, -1,
        5, 4, 16, 9, -1,
        3, 2, 17, 11, -1,
        7, 6, 10, 17, -1,
        13, 14, 10, 9, -1,
        14, 15, 11, 10, -1,
        15, 12, 8, 11, -1,
        12, 13, 9, 8, -1,
  };
  theFaces->coordIndex.setValues(0, 13*5, indices);
  theDig->replaceChild(2, theFaces);

  // add grid
  SoSeparator *grid = new SoSeparator;
  SoDrawStyle *lines = new SoDrawStyle;
  lines->style = SoDrawStyle::LINES;
  grid->addChild(lines);
  SoMaterial *white = new SoMaterial;
  white->diffuseColor.setValue(0., 0., 0.);
  grid->addChild(white);
  float i,j;

  //upper plane
  for (i=-grids; i<dig_w/2.+grids; i+=grids) {
        for (j=-dig_h/2.-grids; j<dig_h/2.+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., -dig_d);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
//lower plane
for (i=-dig_w/2.-grids; i<grids; i+=grids) {
        for (j=-dig_h/2.-grids; j<dig_h/2.+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., -2*dig_d);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// left top border
for (i=-dig_w/2.; i>-dig_w/2.-border-grids; i-=grids) {
        for (j=-dig_h/2.-border-grids; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i-grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// right top border
for (i=dig_w/2.; i<dig_w/2.+border+grids; i+=grids) {
        for (j=-dig_h/2.-border-grids; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// bottom top truncated border
for (i=-dig_w/2.; i<dig_w/2.; i+=grids) {
        for (j=-dig_h/2.; j>-dig_h/2.-border-grids; j-=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j-grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
```

```
  }
  // top top truncated border
  for (i=-dig_w/2.; i<dig_w/2.; i+=grids) {
        for (j=dig_h/2.; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
  }
  theDig->replaceChild(3, grid);

}


void shallow(SoSeparator *globalroot) {

  SoSeparator *theDig = (SoSeparator *)(((SoSeparator *)globalroot->getChild(0))->getChild(Cdig));

  SoCoordinate3 *thePoints = new SoCoordinate3();
  float mypoints[12][3] =
  {
        {-dig_w/2., -dig_h/2., -dig_d},
        {dig_w/2., -dig_h/2., -dig_d},
        {dig_w/2., dig_h/2., -dig_d},
        {-dig_w/2., dig_h/2., -dig_d},
        {-dig_w/2., -dig_h/2., 0.},
        {dig_w/2., -dig_h/2., 0.},
        {dig_w/2., dig_h/2., 0.},
        {-dig_w/2., dig_h/2., 0.},
        {-dig_w/2.-border, -dig_h/2.-border, 0.},
        {dig_w/2.+border, -dig_h/2.-border, 0.},
        {dig_w/2.+border, dig_h/2.+border, 0.},
        {-dig_w/2.-border, dig_h/2.+border, 0.}

  };
  thePoints->point.setValues(0, 12, mypoints);
  theDig->replaceChild(1, thePoints);

  //define the faces of the hole
  SoIndexedFaceSet *theFaces = new SoIndexedFaceSet();
  int indices[9*5] =
  {
        0, 1, 2, 3, -1,
        0, 4, 5, 1, -1,
        1, 5, 6, 2, -1,
        2, 6, 7, 3, -1,
        3, 7, 4, 0, -1,
        8, 9, 5, 4, -1,
        9, 10, 6, 5, -1,
        10, 11, 7, 6, -1,
        11, 8, 4, 7, -1
  };
  theFaces->coordIndex.setValues(0, 9*5, indices);
  theDig->replaceChild(2, theFaces);

  // add grid
  SoSeparator *grid = new SoSeparator;
  SoDrawStyle *lines = new SoDrawStyle;
  lines->style = SoDrawStyle::LINES;
  grid->addChild(lines);
  SoMaterial *white = new SoMaterial;
  white->diffuseColor.setValue(0., 0., 0.);
  grid->addChild(white);
  float i,j;
```

```
for (i=-dig_w/2.-grids; i<dig_w/2.+grids; i+=grids) {
        for (j=-dig_h/2.-grids; j<dig_h/2.+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., -dig_d);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// left top border
for (i=-dig_w/2.; i>-dig_w/2.-border-grids; i-=grids) {
        for (j=-dig_h/2.-border-grids; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i-grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// right top border
for (i=dig_w/2.; i<dig_w/2.+border+grids; i+=grids) {
        for (j=-dig_h/2.-border-grids; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// bottom top truncated border
for (i=-dig_w/2.; i<dig_w/2.; i+=grids) {
        for (j=-dig_h/2.; j>-dig_h/2.-border-grids; j-=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j-grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
                csep->addChild(c);
                grid->addChild(csep);
        }
}
// top top truncated border
for (i=-dig_w/2.; i<dig_w/2.; i+=grids) {
        for (j=dig_h/2.; j<dig_h/2.+border+grids; j+=grids) {
                SoCube *c = new SoCube;
                c->width = grids;
                c->height = grids;
                c->depth = 1.;
                SoTranslation *ct = new SoTranslation;
                ct->translation.setValue(i+grids/2., j+grids/2., 0.);
                SoSeparator *csep = new SoSeparator;
                csep->addChild(ct);
```

```
               csep->addChild(c);
               grid->addChild(csep);
        }
  }
  theDig->replaceChild(3, grid);

}
```

B.3 HandSCAPE v. 1.0 (Packing Optimization)

```
#include "box.h"

int updateBox(SoCube *b, int en, float pitch, float heading, float roll,
                    int& boxwidth, int& boxheight, int& boxdepth,
                    int& widthchange, int& heightchange, int& depthchange)
{
  int w, d, h;
  w = b->width.getValue();
  d = b->depth.getValue();
  h = b->height.getValue();

  // determine type of change
  // 0 = width
  // 1 = height
  // 2 = depth
  int changetype = -1;

  // to determine if it's a height or a width/depth change, look at roll
  // if (PI/4 < roll < 3PI/4) or (5PI/4 < roll < 7PI/4) --> height change
  // else --> width or depth change
  // to determine if it's width or depth change, look at heading
  // if (PI/4 < heading < 3PI/4) or (5PI/4 < heading < 7PI/4) --> depth change
  // else --> width change

  if (((M_PI/4 < roll) && (roll < 3*M_PI/4)) || ((5*M_PI/4 < roll) && (roll < 7*M_PI/4))) {
        changetype = 1;
  } else if (((M_PI/4 < heading) && (heading < 3*M_PI/4)) || ((5*M_PI/4 < heading) && (heading <
7*M_PI/4))) {
        changetype = 2;
  } else {
        changetype = 0;
  }

  if (changetype == 0) {
        // width changed
        printf("Made a width measurement");
        if (w != -1) return 0;
        boxwidth = en;
        widthchange = 1;
        heightchange = 0;
        depthchange = 0;
  } else if (changetype == 1) {
        // height change
        printf("Made a height measurement");
        if (h != -1) return 0;
        boxheight = en;
        widthchange = 0;
        heightchange = 1;
        depthchange = 0;
  } else {
        // depth change
        printf("Made a depth measurement");
        if (d != -1) return 0;
        boxdepth = en;
        widthchange = 0;
        heightchange = 0;
        depthchange = 1;
  }

  animateChange(b, changetype, en);
  return 1;

  heightchange = 0;
}

void animateChange(SoCube *b, int index, int en)
{
  SoOneShot *dim = new SoOneShot;
  dim->duration = 1.0;
  dim->flags = SoOneShot::HOLD_FINAL;

  SoCalculator *dimCalc = new SoCalculator;
  dimCalc->a.connectFrom(&dim->timeOut);
  dimCalc->b = en;
  dimCalc->expression.set1Value(0, "oa = a * b");

  if (index == 0) //width
  {
        b->width.connectFrom(&dimCalc->oa);
  }

  else if (index == 1) //height
  {
        b->height.connectFrom(&dimCalc->oa);
  }

  else //depth
  {
        b->depth.connectFrom(&dimCalc->oa);
  }

  dim->trigger.setValue();
}
```

References

Balakrishnan, R., Fitzmaurice, G., and Kurtenbach, G., Singh, K., (1999) Exploring Interactive Curve and Surface Manipulation Using a Bend and Twist Sensitive Input Strip, *In Proceedings of ACM Symposium on Interactive 3D Graphics* I3DG'99, ACM Press, pp. 111-118

Balakrishnan, R., Fitzmaurice, G.W. and Kurtenbach, G., Buxton W., (1999) Digital Tape Drawing, In Proceedings of ACM Symposium on User Interface Software and Technology, *In Proceedings of User Interaction System and Technology*, UIST'99, pp. 161-169,

Bloomer, K., and Moore, W., (1977). Body, Memory, and Architecture. New Haven: Yale University Press, Ch 4-6.

Boxton, W., Fitzmaurice, G., Balakrishan, R., and Kurtenbach, G., (2000) Interaction Techniques for 3D Modeling on Large Display, *In the Proceeding of IEEE Computer Graphics and Applications*, ACM Press, pp. 68-75

Brien Dillon (1993) Practical Archaeology: Filed and Laboratory Techniques and Archaeological Logistics, Institute of Archaeology, University of California Press: Los Angeles, pp. 33-38

Digibot3^{TM} *Product Information*, Digibotics, Inc. (http://www.digibot.com)

Ermes P. and F.A. van den Heuvel, (1998) Measurements with Digital Photogrammetry, *International Archives of Photogrammetry and Remote Sensing*, Volume 32, part 5, pp. 217-220

Falconer, P., Drury, J. (1975) Building and Planning for Industrial Storage and Distribution, The Architectural Press: London, Introduction

Fitzmaurice, G., Ishii, H., and Buxton, W., (1995). Bricks: Laying the foundation for Graspable User Interfaces. *In Proceedings of CHI'95*, pp. 422-449.

Forte, M., Siliotti, A., (eds.) (1997) Virtual Archaeology: Re-creating Ancient Worlds, New York: Abrams, p. 130

Gershenfeld, N. (1999) When Thinks Starts to Think, New York: Henry and Holt, pp.104-106

Gibson, William (1979) The Ecological Approach to Visual Perception. Boston: Houghton Mifflin, Part Three: "Visual Perception".

Hall, T. Eliot (1966). The Hidden Dimension. New York: Anchor Books, Doubleday. Ch. 4-6, 10.

Hinckley, K., Pausch, R., Proff itt, D., and Kassell, N., (1998) Two-Handed Virtual Manipulation, ACM *Transactions on Computer–Human Interaction*, 260-362

Igarashi, T., Matsuoka, S., and Tanaka, H., (1999) Teddy; A Sketching Interface for 3D Freeform Design, *In Proceeding of SIGGRAPH'99*, pp. 409-416

Ishii, H. and Ullmer, B., (1997) Tangible Bits: Toward seamless Interfaces between People, Bits, and Atoms, in *proceeding of CHI '97*, ACM Press, pp. 234-241

Ishii, H. Wisneski, C, Orbans, J., Junn, B., Paradiso, J., (1999) PingPongPlus: Design of an Athletic-Tangible Interface for Computer Supported Cooperative Play, in *proceeding of CHI '99*, ACM Press, pp. 234-241

Merleau-Ponty, M. (1962). Phenomenology of Perception. Part 2: "Sense Experience" and "Space".London: Routledge,

Lee, J., Su, V., Ren, S., Hsiao, J., Hongladaromp, R., Ishii, H., (1999) HandSCAPE, in *Conference Abstracts and Applications of SIGGRAPH '99* , Emerging Technologies, (Los Angeles, California USA, August 8-13, 1999), ACM Press, pp. 168.

Lee, J., Su, V., Ren, S., and Ishii, H., (2000) HandSCAPE: A Vectorizing Tape Measure for On-Site Measuring Applications, in *Proceedings of Conference on Human Factors in Computing Systems (CHI 2000)* , (The Hague, The Netherlands, April 1-6, 2000), ACM Press, pp.137-144

Lee, J., Dunn, B., Ren, S., Su, V. and Ishii, H., (2000) GeoSCAPE: 3D Visualization of On-Site Archaeological Excavation Using a Vectorizing Tape Measure, in *Conference Abstracts and Applications of SIGGRAPH '00* (New Orleans, Louisiana, July 23-28, 2000), ACM Press. pp.206

Lee, J., Ishii, H., Dunn, B., and Su, V., (2000) GeoSCAPE: A Reconstructive Tool for Field Archaeological Excavation , Design Expo Technical Program in *Conference Abstracts and Applications of CHI 2001* (Seattle, Washington, March 29 – April 5, 2001), ACM Press. (will be published)

Monkey2™. *Product Information*, Digital Image Design Incorporated, New York

Papanek, Victor (1963) Design for the Real World: Human Ecology and Social Change, New York: Van Nostrand Reinhold, p.28

Paradiso, A. J., Hsiao, K., Bedbasat, Y. A., J., Teengarden, A., (2000) Design and Implementation of Expressive Footwear Sensor, Proceedings of the IBM System Journal (ISJ), New York: IBM Coporation, vol 39, pp. 511-529

Ruland, Robet (1993) The Chesapeake Laser Tracker in Industrial Metrology, Proceedings of the Third International Workshop On Accelerator Alignment, CERN – European Laboratory for Particle Physics, pp. 101-118.

SHAPE TAPE ™, *Product Information,* Measurand Inc., Fredericton, New Brunswick, Canada.

Shaper, R., Ashmore, W., (1993). Archaeology: Discovering Our Past, California: Mayfield Publishing Company, pp. 114-120.

Underkoffler J. and Ishii, H.,(1999) Urp: A Luminous-Tangible Workbench for Urban Planning and Design, In *Proceedings of the CHI '99*, ACM Press, pp. 386-393

Weiser, M. (1991) The Computer for the 21 Century, *In Scientific American*, 265 (3), pp. 94-104

Yarin, P. and Ishii, H. (1999) TouchCounters: Designing Interactive Electronic Labels for Physical Containers, in *Proceeding of the CHI '99*, ACM Press, pp. 362-369

Zeleznik C., Herndon K., and Hughs F., (1996) SKETCH: An Interface for Sketching 3D Scenes, In *Proceedings of SIGGRAPH '96,* pp. 163-170