A System for Tracking and Characterizing Acoustic Impacts on Large Interactive Surfaces

by

Nisha Checka

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

Bachelor of Science in Electrical Science and Engineering

and

Master of Engineering in Electrical Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2001

Author .....
Department of Electrical Engineering and Computer Science
May 24, 2001

Certified by .....
Joseph A. Paradiso
Principal Resesarch Scientist
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

A System for Tracking and Characterizing Acoustic Impacts on Large Interactive Surfaces

by

Nisha Checka

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2001, in partial fulfillment of the
requirements for the degrees of
Bachelor of Science in Electrical Science and Engineering

and

Master of Engineering in Electrical Science and Engineering

Abstract

In this thesis, I designed and implemented a system to track and characterize acoustic impacts on large interactive surfaces. This involved choosing the appropriate sensors, designing the hardware, and finally developing algorithms to determine the location of a tap. The hardware was designed so that low-noise, high quality signals were produced so that some of the problems that caused inaccuracies in the position determination were eliminated. Various algorithms were designed and implemented to best locate the position of the tap. Finally, an application for the acoustic tap tracker was designed.

Thesis Supervisor: Joseph A. Paradiso
Title: Principal Resesarch Scientist

Acknowledgments

To **Joe Paradiso**, for his guidance, encouragement, and advice. In the time that I have worked for him, I have learned so much. It has been a pleasure to work for him.

To **Josh Lifton**, for his insightful conversations.

To **Ari Benbasat** and **Stacy Morris** for reviewing my thesis and for their willingness to help me when I needed help.

To my colleagues of the **Responsive Environments Group** for making my experience at the MIT Media Lab stimulating and rewarding.

Finally, to my **family** for their encouragement, love, and support.

Contents

1 Introduction **15**

1.1 Motivation 15

1.2 Goals 16

1.3 Outline of Thesis 16

2 Sensing Systems **19**

2.1 The Un-Private House Exhibition 19

2.2 SoftBoard 20

2.3 Intrepid Touch System 20

2.4 Computer Vision Techniques 20

2.5 Ping-Pong Plus (PP+) 21

2.5.1 Ball-Tracking Electronics 21

2.5.2 Software Algorithms 22

2.6 Initial Prototype of the Acoustic Tap Tracker 22

2.6.1 Method 1: Data Acquisition using the NI PCI-6024E 24

2.6.2 Method 2: Data Acquisition using the Hitachi SH microprocessor . . 24

2.7 Problems with the prototype 25

3 Background **27**

3.1 Polyvinylidene Fluoride (PVDF) Sensors 27

3.2 Wave Propagation in Glass 28

4 Electronics **31**

4.1 Pre-Amplifier Board 31

4.2 Signal Conditioning Board 38

5 Algorithms **41**

5.1 Data Acquisition 41

5.2 Algorithms using Time-of-Flight Analysis 41

5.2.1 Calibration 42

5.2.2 Foward Determination 44

5.3 Algorithms incoporating Normalized Integrated Amplitude 49

5.4 Algorithms incorporating Cross-Correlation Data 51

5.4.1 Peak Extraction Using Averaging 53

5.4.2 Peak Extraction Using Polynomial Fitting 54

5.4.3 Peak Extraction Using the Ratio of Peaks 54

5.5 Example 56

5.6 Hard Impacts 62

6 Further Testing of the Portable Acoustic Tap Tracker 65

7 Conclusions and Future Research 71

7.1 Applications 71

7.2 Future Applications 74

7.3 Future Research 75

A Abbreviations and Symbols 77

B Schematics and PCB Layouts 79

B.1 Pre-Amplifier Schematic 80

B.2 Pre-Amplifier PCB Layout 81

B.3 Signal Conditioning Board Schematic 82

B.4 Signal Conditioning Board PCB Layout 83

C Matlab Code 85

C.1 calibrate.m 85

C.2 tapper.m 88

C.3 finalize.m 89

C.4 initializeDAQ.m 90

C.5 once.m 91

C.6 pinpoint.m 92

C.7 takeOneTap.m 95

C.8 determinePeaks.m 96

C.9 peakRatio.m 97

D C Code **99**

D.1 client.cpp 99

List of Figures

2-1 Ping-Pong Plus Diagram [21]. 22

3-1 PVDF sensors [9]. 28

3-2 Waveforms generated by a knuckle tap and a metal ring hitting the glass. 29

3-3 The above waveforms display the dispersion phenomenon. 30

4-1 Overview of the acoustic tap tracking system. 32

4-2 Pictures of the first version of the pre-amplifier board (Velcro mounting scheme). 34

4-3 Mounting scheme of the pre-amplifier board. 34

4-4 Waveform from sensor 1 generated by a tap at location (100,80). The Velcro mounting scheme was used. 35

4-5 Noise comparison between the prototype acoustic tap tracker and the new version. The spiked waveform represents the noise on the line for the prototype. The blue waveform represents the noise on the line for the prototype tap tracker. 36

4-6 Positions where significant attenuation can be observed due to the presence of the Velcro in the signal path. 37

4-7 Comparison of waveforms between the original Velcro mounting scheme and the direct glue (half-velcro) mounting scheme. An impact at location (14,53) generated these waveforms at sensor 4. 38

4-8 Mounting scheme of the final version of the pre-amplifier board. 39

4-9 Pictures of the final pre-amplifier board. 40

4-10 Picture of the signal conditioning board. 40

5-1 Signals from sensor 4 for a series of knuckle taps moving across the pane of glass from left to right. 43

5-2 The two waveforms above demonstrate the inhomogeneities associated with sensor 2. Although the lower waveform was generated by a tap closer to the sensor 2, it exhibits more attenuation than a signal generated by a tap farther away. 46

5-3 Waveforms generated by each sensor with an impact at (18,77) (closest to sensor 4). . . . 47

5-4 Dispersion comparison between two taps both at location (18,77). 48

5-5 Waveforms showing normalized amplitudes of taps generated at location (18,22). 50

5-6 Cross-correlations between sets of signals for a tap at (55,65) 51

5-7 Both waveforms are the cross-correlations between sensors 2 and 4 for a tap located at (78,59). The waveform on the left was generated by one trial, and the waveform on the right was generated by a different trial. The waveforms are virtually identical showing the consistency of the cross-correlation. The left waveform peaks at location 444 while the right waveform peaks at 443. 52

5-8 Cross-correlation between the signals from sensors 2 and 4 for a tap at location (98,22). . 53

5-9 The cross-correlation on the left was generated by a tap at location (101,29). The cross-correlation of the right was generated by a similar tap at the same location. These two figures show the ambiguity in resolving the maximum peak. 54

5-10 Predicted distribution of guesses. 55

5-11 Distribution of guesses for a tap at location (55,65). The crosses represent guesses. The diamond represents the actual location of the tap, and the star represents the closest calibration point. 57

5-12 Figures generated by running the forward algorithm with the correlation data of the calibration points for the second order model. 58

5-13 Figures generated by running the forward algorithm with the correlation data of the calibration points for the third order model. 60

5-14 Flowchart of algorithm decision process. 61

5-15 Signal generated by a metal tap at location (40,50). 63

6-1 Picture of the acoustic tap tracker setup in the atrium of the MIT Media Lab. The effective area of enclosed by the sensors is approximately 35 inches by 47 inches. The computer was used for DAQ. 66

6-2 Waveforms generated by a tap at location (18,77) using the pane of glass in the atrium. . 67

6-3 Signals from sensor 1 generated by a tap at location (100,80) using two different panes of glass. 68

6-4 Cross-correlation between sensors 1 and 4. Generated by a tap at location (58,52). . . . 69

6-5 Signals from sensors 1 and 3. The size of the effective surface is 55 inches by 60 inches. . 69

6-6 Signal from sensor 1 generated by a metal tap at location (98,77). The pane of glass is in the atrium. 70

7-1 Picture of the showcase wall demo for the acoustic tap tracker. 72

7-2 Picture of the Magic Circles demo. The circles drawn are centered at the tap location, and the radius represents the uncertainty region. 73

7-3 LaserWall. 75

B-1 Schematic of the pre-amplifier board. 80

B-2 PCB Layout of the pre-amplifier board (top and bottom layers). 81

B-3 Schematic of the signal conditioning board. 82

B-4 PCB Layout of the signal conditioning board (top layer). 83

B-5 PCB Layout of the signal conditioning board (bottom layer). 84

Chapter 1

Introduction

The purpose of this work is to develop a new interactive surface whereby the location of finger taps on the surface is tracked. This involves choosing the sensors, the design of hardware to condition the signals before being processed by a data acquisition system, and the development of the algorithms to determine the position of the tap on the surface in real time. This chapter provides the objectives and motivation of the work and concludes with an overview of the remainder of the thesis.

1.1 Motivation

When asked to discuss computers, people usually describe the interfaces because for most users, the interface is the entire system [12]. As technology becomes more pervasive in the world around us, we are beginning to live between two realms: our physical environment and cyberspace. Despite this duality, there exists no seamless bridge to connect these two realms. This research explores one type of seamless bridge to connect people, digital information, and the physical environment that goes beyond traditional graphical user interfaces [19].

Large, flat surfaces are commonplace in our environment but have been limited to a passive role. Much technology has been developed in disparate pieces to enable the vision of embed-

ding computing power into everyday objects such as clothing and furniture. For instance, existing technologies, based on the idea of "smart rooms", have tried to make these surfaces more interactive by embedding small or moderately sized devices such as touchscreens [17]. However, in these systems, the surface itself remains passive, but the system becomes more active by the addition of an active component. Using large portions of walls and tables directly as interactive surfaces still remains unusual.

One of the main challenges in transforming ordinary surfaces into sensate surfaces resides in the characteristics of the surfaces themselves: their size. For instance, while numerous technologies exist for implementing touchscreens, applying these technologies to a wall is impractical because of their expense and issues with scaling. The acoustic tap tracker that will be described in this thesis is relatively inexpensive, scaleable to large surfaces, and can be easily retrofit to many existing surfaces without resulting in a corresponding degradation in performance or increase in complexity.

1.2 Goals

The primary goal of this research is to develop a new interactive sensate system that is scaleable to large surfaces while maintaining good performance. The research consists of two parts: the design of electronics to condition the sensor signals for analysis and the development of algorithms to accurately determine the location of the tap. The system will then be characterized to determine possible improvements to the system.

1.3 Outline of Thesis

This is chapter one, the Introduction, which introduces the thesis subject matter, the motivation, the goals, and the outline of this thesis.

Chapter two, Sensing Systems, describes the theory and implementation of other sensing systems.

Chapter three, Background, describes the sensors as well as the properties of the glass.

Chapter four, Electronics, describes the signal conditioning hardware of the system.

Chapter five, Algorithms, describes the theory behind determining the location of the tap.

Chapter six, Further Testing of the Portable Acoustic Tap Tracker, includes results from porting the system to a different pane of glass and testing the system over a varying surface size.

Chapter seven, Conclusions and Future Work, describes the incorporation of the acoustic tap tracker with other sensate systems and proposes future extensions.

Chapter 2

Sensing Systems

Numerous technologies are available that transform ordinary surfaces into interactive surfaces. By analyzing the techniques and shortcomings of each of these systems, an amalgam of ideas from these systems was used in developing the acoustic tap tracker. The technologies that will be discussed are a pixilated capacitive matrix used for a sensor table, SoftBoard, the Intrepid Touch System, systems employing computer vision techniques, and the Ping-Pong Plus (PP+) installation. Emphasis will be on the PP+ installation as this project has had the most impact on the current research.

2.1 The Un-Private House Exhibition

The Un-Private House is an exhibition of architectural projects at the New York Museum of Modern Art. A significant amount of information about the projects was presented through computer kiosks away from the exhibit. Researchers at the MIT Media Lab were presented with the task of presenting this information to the exhibit's visitors within the context of the exhibit itself [15]. They designed a pixilated capacitive matrix for a dining table that was used to present the information. This technique can detect and track nearby hand movements through capacitive loading; however, extending these concepts to large displays becomes complicated and expensive. The surfaces must be opaque since the technology

used to implement the system is embedded under the surface itself.

2.2 SoftBoard

Tracking systems for translucent displays usually involve the use of devices that look across the edge of the display [17]. Some of these systems use lasers that track the location of coded targets. One such system is SoftBoard developed by Microfield Graphics where a pair of scanning lasers is attached to the top two corners of the board. Lasers emanate from those positions and can identify and track coded targets. These sensors cannot detect distance; therefore, position is determined by triangulating the two angular measurements [13]. Some of the drawbacks of SoftBoard are the use of coded targets and its inability to track uncoded targets.

2.3 Intrepid Touch System

Similar systems that track uncoded targets, such as a bare hand, do exist. These systems use a dense array of infrared light-emitting diodes (LEDs) that face receivers lining the perimeter of the surface. One such technology based on this approach is the Intrepid touch system. The sensing component consists of two cameras placed in the upper corners of the surface and a bank of LEDs that produce a plane of infrared (IR) light. When an object touches the screen, the path of LED light is reflected into the cameras and processed by a microcontroller that determines the exact location of the touch [1]. While this technique works well for small applications, its sensing component does not scale well to larger surfaces due to the size of the bank of LEDs required for large surfaces.

2.4 Computer Vision Techniques

The techniques described above all use various forms of sensors. Another approach involves the use of computer vision. These systems use multiple cameras that view the plane of the surface. This technique provides information not only on location but could potentially be used to determine hand gesture [17].

One such system is the Holowall[17], whereby an IR camera positioned next to a bright IR source looks from the back of a translucent screen. Although considerable IR light penetrates the screen, the user in front is unable to see this. The IR camera captures reflections (propagating back through the screen) from the user's hands as they approach the surface. Image processing on the resulting frames is used to detect the hands. Some of its drawbacks are that real-time image processing are required and that it has difficulties with clutter and background IR illumination.

2.5 Ping-Pong Plus (PP+)

Ping-Pong Plus (PP+) [13] is a digitally enhanced version of ping-pong developed by the Tangible Media Group with technical collaboration from the Responsive Environments Group of the MIT Media Lab. PP+ served as the inspiration for the acoustic tap tracker. A "reactive table" was designed that displays patterns of light based on the location of the hit of the ping-pong ball. The system consists of three main elements: the ball-tracking electronics, software algorithms for ball-hit location, and a graphics projection system.

2.5.1 Ball-Tracking Electronics

The ball-tracking electronics consisted of the sensors as well as the electronics to track the ping-pong ball. The ball position is sensed solely through the sound. Eight electret microphones (four for each side) mounted on the underside of the table pick up the sound of the ball hitting the table. This is depicted in Figure 2-1.

The acoustic characteristics of the ball impacts had a sharp leading edge and were fairly uniform, enabling sufficient accuracy using constant-threshold discriminator [20] circuits to determine the first arrival at each of the microphones before the signal is contaminated through multipath and reflections. The first part of the electronics involved detecting a ball hit by performing peak thresholding on signals from the microphones. A PIC microcontroller constantly polls one of its digital inputs, and once a hit is detected, the PIC assigns a time value to that microphone input and sends this information to software running on a PC [13].

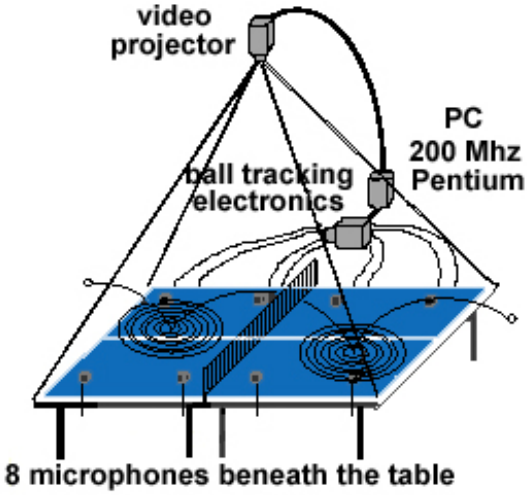


FIGURE 2-1: Ping-Pong Plus Diagram [21].

2.5.2 Software Algorithms

The researchers working on PP+ developed a simple algorithm to calculate ball hit position. This method is based on a comparison of the time difference data to a set of model parameters acquired by a linear least-squares fit of calibration data. This method did introduce some distortion but provided accuracy on the order of inches. This accuracy was acceptable given the application. Several modifications to the algorithms would improve the accuracy of the system. For instance, implementing peak detection and matching the various incoming waveforms (as opposed to simple thresholding) would allow a more accurate determination of the time differences and remove any amplitude dependence. These improvements were incorporated into the design of the acoustic tap tracker.

2.6 Initial Prototype of the Acoustic Tap Tracker

The initial prototype of the acoustic tap tracker[2] was based on the PP+ installation. One of the main differences between PP+ and the first prototype was the sensor choice. While PP+ used electret microphones, pickups made of polyvinylidene fluoride (PVDF)

piezoelectric foil were used in the prototype. Electret microphones were not used because of their sensitivity to noise and poor coupling to the surface. Chapter 3 includes more information on the reasons for choosing PVDF as well as background information on PVDF.

The prototype system consisted of four components: signal transduction, signal conditioning, data capture, and data analysis. The signal transduction stage consisted of four PVDF pickups epoxied to the glass surface. Each sensor is approximately two square centimeters in size and is mounted to the four corners of a tempered, shatter-proof glass pane that makes up part of a glass wall enclosing a conference room. The pane is approximately one meter wide, two meters tall, and half a centimeter thick. The pane spans the entire distance between the floor and ceiling and is further held in place by about half a centimeter of heavy rubber grout connecting it to adjacent panes. The four sensors are arranged on the pane to form a rectangle, which covers the middle third of the pane. That is, the rectangle is as wide as the pane, but only a third as tall and centered at the center of the pane (see Figure 4-6).

The purpose of the signal conditioning stage was to prepare the transduced signal for data capture and for noise immunity for transmission. Signal conditioning consisted of a series of amplifiers, followers, and rectifiers for each channel. Essentially, a low-impedance, amplified, absolute value of the original transduced signal is outputted from the signal conditioning stage. The signal conditioning stage was made up of discrete components and was located on a breadboard powered with a +12 volt DC supply.

The data capture stage was used to collect the data from the outputs of the signal conditioning circuit. Two different methods were implemented for data acquisition (DAQ): using the National Instruments (NI) PCI-6024E [18] data acquisition board and using an analog to digital converter (ADC) onboard the Hitachi Super-H (SH) RISC processor, the SH7032 [7, 6].

2.6.1 Method 1: Data Acquisition using the NI PCI-6024E

Both the data capture and data analysis stages were accomplished using the NI DAQ device in conjunction with the DAQ toolbox of Matlab 5.3. Data is continually sampled from each sensor. Upon detecting a signal above a certain noise threshold, a "knock" event is declared and 10 ms worth of data is stored from all four sensors[1]. Descriptive parameters such as peak height, width, and mean arrival time relative to the initial trigger are extracted for each peak using Matlab.

2.6.2 Method 2: Data Acquisition using the Hitachi SH microprocessor

The SH microprocessor includes an ADC that samples each of the four channels at roughly 10 kHz per channel with 10 bits of resolution. The microprocessor continuously samples the signals from each of the sensors into a circular buffer. The rest of the data acquisition process is then identical to that of the National Instruments card; however, extraction of key parameters is performed using the SH microprocessor instead of using Matlab. These parameters are then transmitted over a serial connection to a PC that processes the timing to determine the location of the tap.

Regardless of the method used for data capture, the same analysis was performed. An overview of the methodology is presented here. A more detailed description can be found in Chapter 5. Essentially, the problem to be solved is to determine the location of an impact, which generates vibrational waveforms at each of the PVDF sensors. Fundamentally, there are two ways to find the location of the impact given the data from the four sensors. One way is to determine the location through differential time-of-flight analysis between the signals from the four sensors. The other involves examining the relative attenuation of the signals between sensors. In general, the farther away the source is from the tap, the greater the attenuation.

[1] This includes 3 ms worth of data before the trigger event.

2.7 Problems with the prototype

The major problem with the prototype was the quality of signals acquired. The received signals were quite noisy for several reasons. First, the wires connecting the sensors to the signal conditioning circuit were very long (approximately 6 ft). This introduced pickup noise into the signal as well as degraded the signal itself. Furthermore, the received signal was not filtered to remove high frequency noise. Because the signal quality was poor, little information was obtained from the amplitude analysis, and the location was determined solely through time-of-flight algorithms. The acoustic tap tracker presented in this thesis solves these problems to improve location determination through better hardware and improved algorithms.

Chapter 3

Background

This chapter presents information on polyvinylidene fluoride sensors (PVDF). In particular, why these sensors were chosen and how the sensors respond to stimuli. Furthermore, information on the propagation of waves in the glass medium is presented.

3.1 Polyvinylidene Fluoride (PVDF) Sensors

Although the system presented in this thesis is an extension of the PP+ project described in Chapter 2, the sensors used were not the same. The implementation of PP+ dealt with impacts of uniform and predictable characteristics due to the hard impacts of the ball hitting the table and the good acoustic propagation characteristics of the supported wood table. The techniques used were well suited for the application but could not be generalized to scenarios such as locating the positions of fingers knocking on a pane of glass. This situation is more complex because finger excitation can vary considerably from one hit to the next. These variations depend on how the glass is struck, the type of glass used, and how the glass is supported. Considering all these factors, sensors made of polyvinylidene fluoride (PVDF) [9] were used instead of electret microphones. These sensors work as contact pickups, are insensitive to ambient sounds in the air, and produce excellent signals when the glass is hit. Figure 3-2(a) is a waveform generated by such a PVDF sensor.

PVDF is a semicrystalline homopolymer, which means that it is a polymer in a mixture of crystalline and amorphous states. One of the states is ferroelectric and responsible for PVDF's piezoelectric properties [18]. In manufacturing PVDF foil, this ferroelectric state is created by rolling the foil and then applying a high electrostatic potential across the foil at an elevated temperature to permanently polarize the PVDF. The foil is then cooled, and a conductive coating is applied to serve as electrodes [16]. Figure 3-1 is a picture of the PVDF sensors used.

FIGURE 3-1: PVDF sensors [9].

When the foil is stretched, the dipoles and their attached crystalline structure move, inducing a change in the polarization charge at the foil surfaces that appears as a voltage across the electrodes. Thus, PVDF can be made to work as an acoustic pickup since the impinging sound pressure waves change the foil shape that induces a voltage [16].

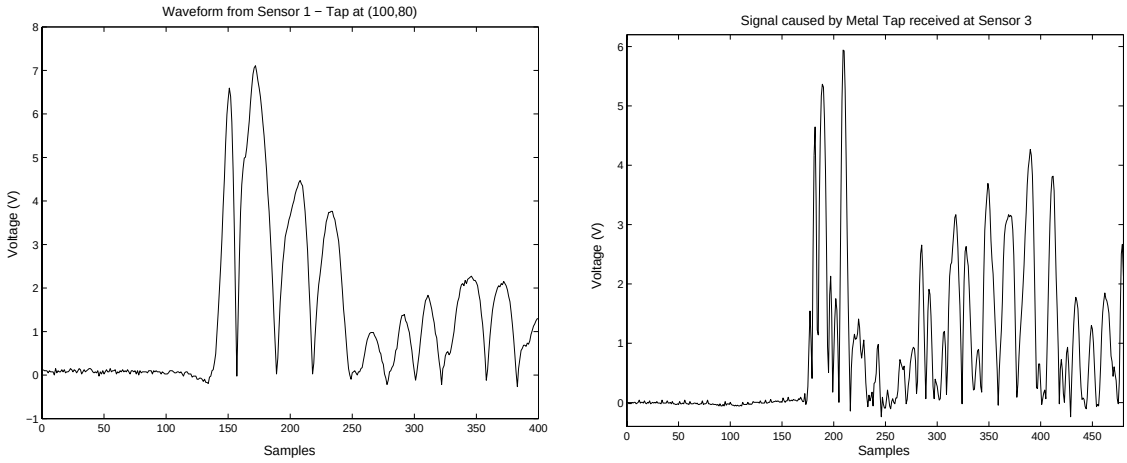
3.2 Wave Propagation in Glass

The acoustic tap tracker current supports two types of impacts: a knuckle tap and a hard knock. Each type of impact has extremely different characteristics, which the system must analyze in order to make an accurate prediction of the location of the tap.

Knuckle taps and hard knocks propagate via different modes. For instance, the observed propagation mode of the low-frequency, finger tap-instigated impulses move quite slowly at roughly 450 m/s [17], which is approximately a factor of 10 below the speed of sound in glass

which is 3,658 m/s [14], depending on the kind of glass. This indicates that these slower waves are mostly likely flexural waves [8]. However, sharper impacts (i.e. when hitting the glass with metal) are seen to propagate at approximately 3500 m/s [17], indicating a faster acoustic propagation mode.

Because of the different modes of propagation, the signals received display different characteristics for these two types of impacts. Figure 3-2(a) is a signal generated by a knuckle tap while Figure 3-2(b) shows a signal generated by a hitting a metal ring against the glass.



(a) Knuckle Tap.

(b) Metal Tap.

FIGURE 3-2: Waveforms generated by a knuckle tap and a metal ring hitting the glass.

The characteristics of the first arrival can vary widely from sensor to sensor and from impact to impact. The dispersive nature of flexural waves poses a significant problem since its wave velocity is proportional to the square root of its frequency [8]. Thus, higher frequency flexural waves will travel faster than lower frequency flexural waves. The signal generated by a knuckle impact contains many frequency components; thus, some components will arrive before others. Figure 3-3 demonstrates such dispersion.

The lower waveform exhibits significant dispersion before the waveform; however, the top waveform does not display as much dispersion. The variable amount of low-amplitude, higher-frequency dispersive deflection that often arrives before the main wavefront as shown

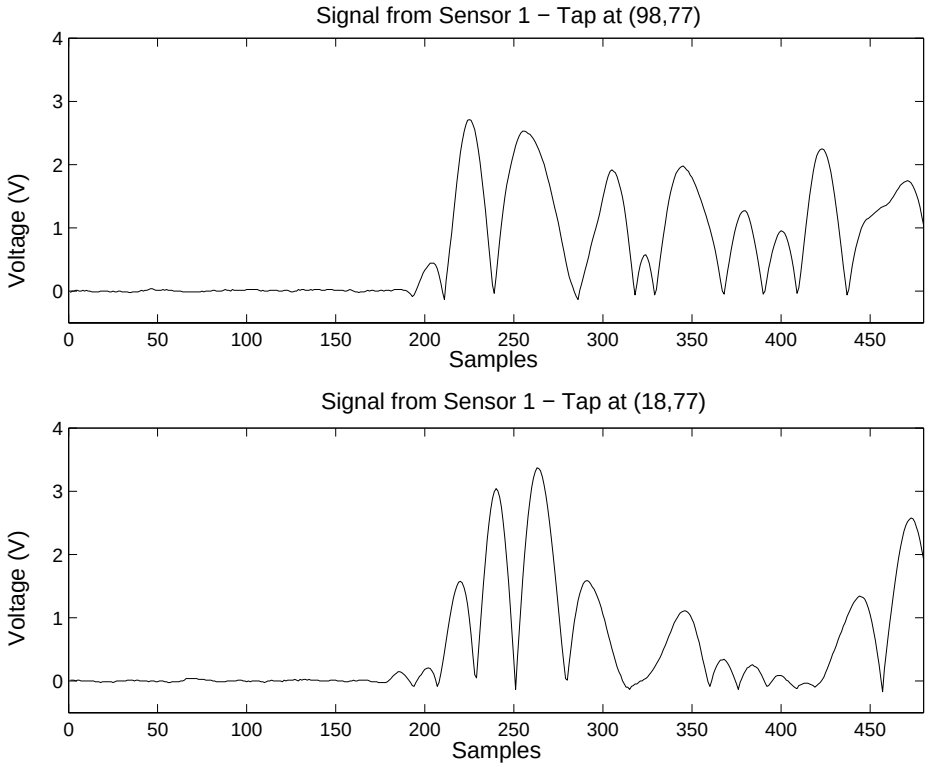


FIGURE 3-3: The above waveforms display the dispersion phenomenon.

in Figure 3-3 poses a significant problem to the timing analysis. Resolution of this problem is discussed in Chapter 5. Sharp impacts propagate via acoustic modes and thus do not display as much dispersion (see Figure 3-2(b)). Thus, leading-edge timing is more straightforward for sharp impacts.

Chapter 4

Electronics

The electronics for this system can be separated into two sub-systems: the pre-amplifier electronics and the signal conditioning board. The entire system consisted of the sensors, signal conditioning board, and data acquisition system. Figure 4-1 shows the overview of the system

The system included four sensors; however, three separated sensors are sufficient to locate an impact on the two-dimensional surface. The fourth sensor adds one degree of redundancy for consistency checking and better resolution.

4.1 Pre-Amplifier Board

The design for the new acoustic tap tracker differs from the prototype system in the use of a pre-amplifier board to amplify the sensor signal before transmitting to the signal conditioning board. One of the main problems with the prototype was the quality of the signal produced at each of the sensors, which were directly glued to the glass surface. A wire approximately six feet in length carried the signal from the sensor to the signal conditioning board. Because of the long length of the wires, the received signal was very noisy. The prototype employed differential time-of-flight analysis to determine the position of the tap

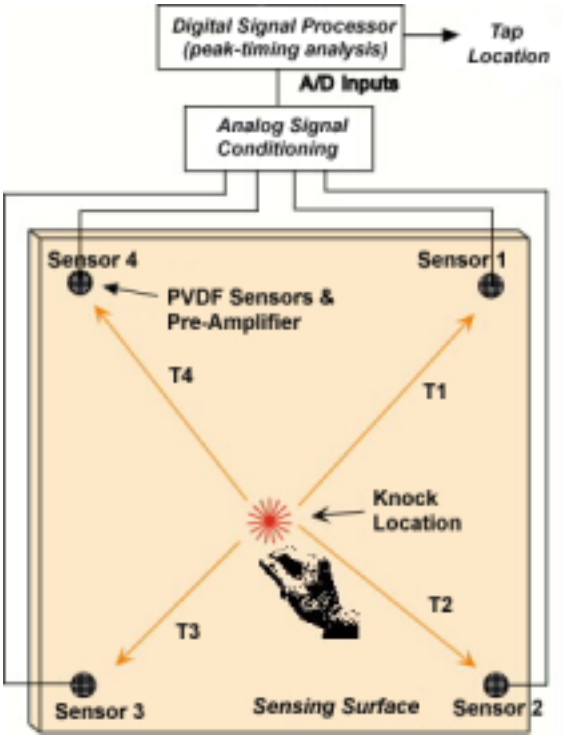


FIGURE 4-1: Overview of the acoustic tap tracking system.

(described further in Chapter 5), the accuracy of which depends critically on the determination of the start of the signal. As discussed in Chapter 3, the waves generated by a knuckle tap are dispersive. The amount of dispersion before the main wavefront makes determination of the start of the signal difficult. Because the magnitude of the noise was on the same order as that of the initial signal, precisely locating the start of the signal was even more complicated. The addition of a low pass filter on the signal conditioning board would have been a straightforward solution to eliminate high frequency noise. However, there was a fair amount of noise in the same frequency band as the signals themselves so any filtering would also attenuate those frequencies of the signal. Thus, the cutoff frequency was made sufficiently high to avoid attenuating components of the signal itself.

The additive noise in the signal also complicated triggering. The data acquisition system started sampling the signal when a threshold was exceeded. Since the noise cannot be differentiated from the signal itself, false triggers were be generated and had to be compensated

for in software.

A major source of the noise problem in the prototype was pickup and attenuation of the high-impedance PVDF signal by the long wires used for transmission. There are two solutions to solve this problem. The first is to simply reduce the length of the wires; however, the size of the surface and the location of the sensors does not easily permit this solution. The sensors are located in the four corners of a large glass surface approximately 47 inches in length and 35 inches in width. Each of the four sensors has to be connected to the signal conditioning board. This board connects to a PC that contains the data acquisition system. Therefore, the length of the wires is determined by the proximity of the PC to the surface. The system should work reliably regardless of this parameter, thus a more robust solution had to be devised.

The solution to this problem was to add an additional amplification stage before the signal is transmitted. This amplifier was mounted directly with the sensor so that the signal is amplified before transmission, and the wire is driven directly through a low impedance op amp output. The pre-amplifier board would have to be small enough so that the board would not stand out significantly when mounted on the glass. The design consisted of using a high-impedance, non-inverting amplifier circuit with a gain of 10. The schematic is included in Appendix B. A stereo phone connector was used to connect the pre-amplifier board to the signal conditioning board, supplying ground and routing the signal. Figure 4-2 shows pictures of the first design of this pre-amplifier board.

Surface mount components were used to keep the size of the board at small. The largest component on the board was the stereo phone connector so it primarily determined the size of the board.

Another design consideration was to make the whole system portable such that the sensors and the pre-amplifier board would be easily mountable and movable. The prototype required gluing the sensors directly to the glass. In the process of removing the sensors from the glass, they would be destroyed.

Several mounting schemes were explored. The first mounting scheme used Velcro. Two

(a) Front view.

(b) Back view.

FIGURE 4-2: Pictures of the first version of the pre-amplifier board (Velcro mounting scheme).

small Velcro strips were attached to the top and bottom of the underside of the board, which would attach to the corresponding Velcro attachments on the glass. The sensor was connected directly to the board itself and was pressed to the glass using a piece of foam. This is depicted in Figure 4-3.

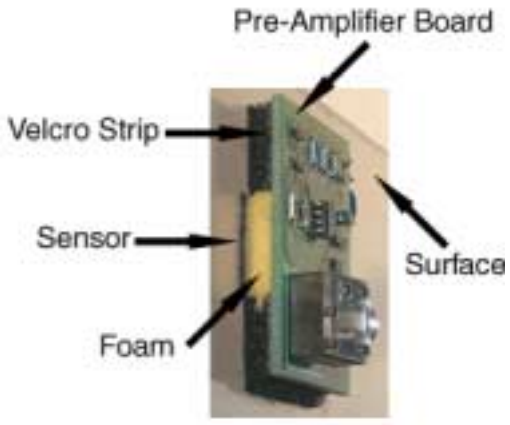


FIGURE 4-3: Mounting scheme of the pre-amplifier board.

Essentially the board presses the sensor to the glass. The system is entirely portable since the board and sensor can be removed easily by detaching it from the Velcro. A waveform generated using this scheme is shown in Figure 4-4.

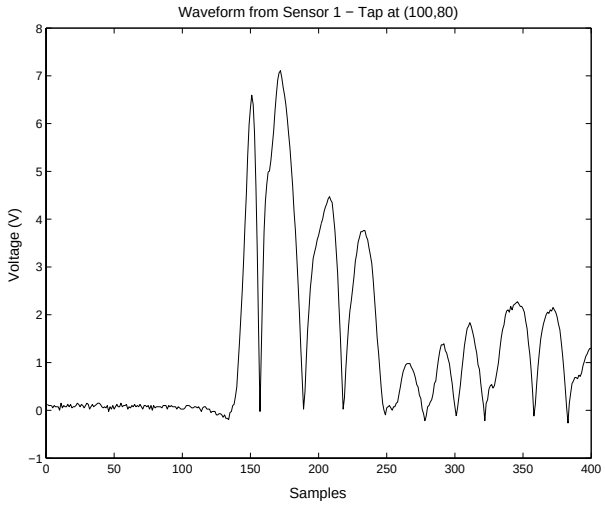


FIGURE 4-4: Waveform from sensor 1 generated by a tap at location (100,80). The Velcro mounting scheme was used.

Figure 4-5 compares the noise between the prototype tap tracker and the new system. This noise is present when the system is idle (i.e. there has been no tap). The peaks in the noise of the prototype was mostly likely due to pickup noise that was minimized using the pre-amplifier. As can clearly be seen, the noise has been significantly reduced using the pre-amplifier.

Although the use of the pre-amplifier board significantly improved the quality of the waveforms, a second order effect due to the mounting scheme was observed. For instance, when a tap occurs at the vertical midpoint between two sensors (as shown in Figure 4-6) significant attenuation in the signal is observed when compared to a tap that occurs at the horizontal midpoint between two sensors.

Because a strip of Velcro was located directly in the acoustic path, that piece of Velcro was disturbing the signal before it reaches the sensors. To test this hypothesis, a different mounting scheme was used. The same pre-amplifier board was used; however, the sensor was directly glued to the glass surface, and one piece of Velcro located outside the signal path held the board to the glass. Figure 4-7 compares the waveforms generated by a tap at the same location (14,53) using the Velcro mounting and direct epoxy mounting. The Velcro

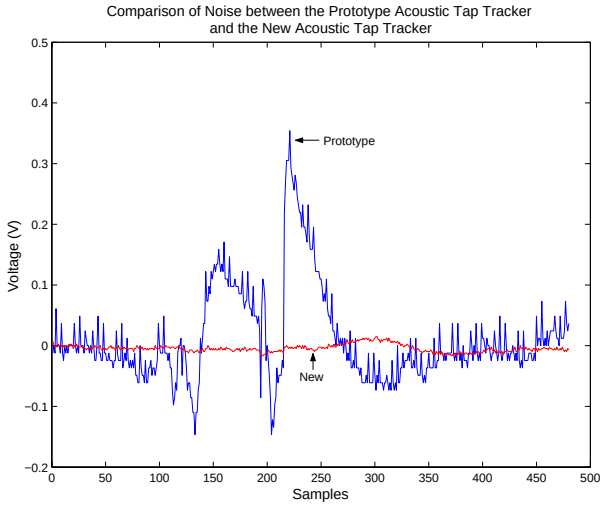


FIGURE 4-5: Noise comparison between the prototype acoustic tap tracker and the new version. The spiked waveform represents the noise on the line for the prototype. The blue waveform represents the noise on the line for the prototype tap tracker.

not only attenuated the signal but also introduced significant distortion. By removing the Velcro strip from the signal path, the attenuation and distortion was no longer present.

While directly gluing the sensor to the glass solved the attenuation problem, it created others. First, the sensors and pre-amplifier were no longer portable. If a board malfunctioned, the sensor would have to be removed, thus destroying it so a new sensor would have to be soldered to the board. Second, the process of applying the super glue to the sensor had to be performed carefully. If any of the glue contacted the electrodes, the signal quality degenerated. The full surface of the sensor had to be coated with the super glue to ensure that good contact was made between the sensor and the glass. Finally, the wire connecting to the pre-amplifier board was quite heavy so the board needed to be securely mounted so that it would not detach from the glass. With only one piece of Velcro (i.e. only one side is not in the direct acoustic path), the board was not very securely mounted and frequently became detached.

The final mounting scheme solved the shortcomings of the direct glue method without any additional problems. The main problem with the two mounting schemes described above

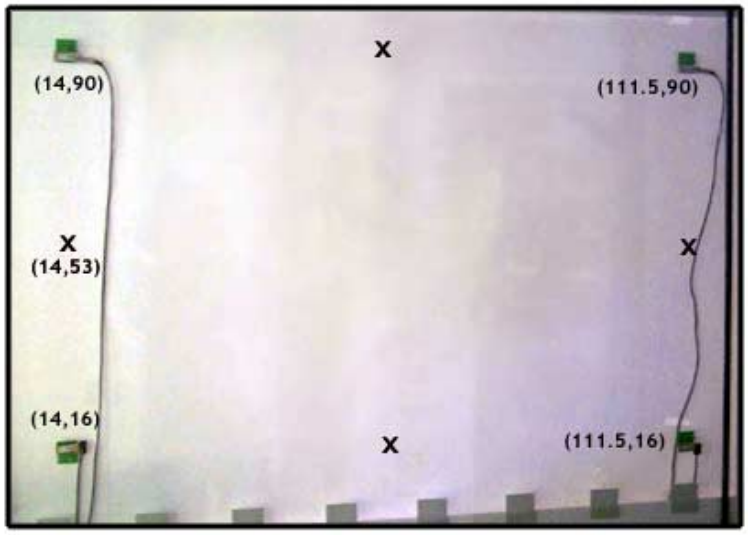


FIGURE 4-6: Positions where significant attenuation can be observed due to the presence of the Velcro in the signal path.

involved the Velcro. For the direct glue mounting scheme, one strip of Velcro was not adequate to support the board and the connecting wire. Thus, another way of attaching the board and sensor to the glass would have to be devised. The scheme developed was based on the first design of using the board to press the sensor against the glass, allowing for good contact without having to directly epoxy the sensor to the glass. Instead of using Velcro to attach the board, the board would be screwed into nuts that are glued to the glass. Figure 4-8 depicts the mounting scheme.

In this scheme, two nuts are glued to the surface of the glass. The board has two holes so that nylon screws could be used to attach the board to the nuts. Foam attached to the underside of the board pushes the sensor into the glass. When the screws are tightened, the sensor presses directly to the glass resulting in good contact between the sensor and the surface. This system is now entirely portable. The only preparation needed before operation is the gluing of eight small nuts (two per sensor) to the surface. Figure 4-9 shows pictures of the final pre-amplifier board.

These board were designed in two orientations: one for the right part of the surface and

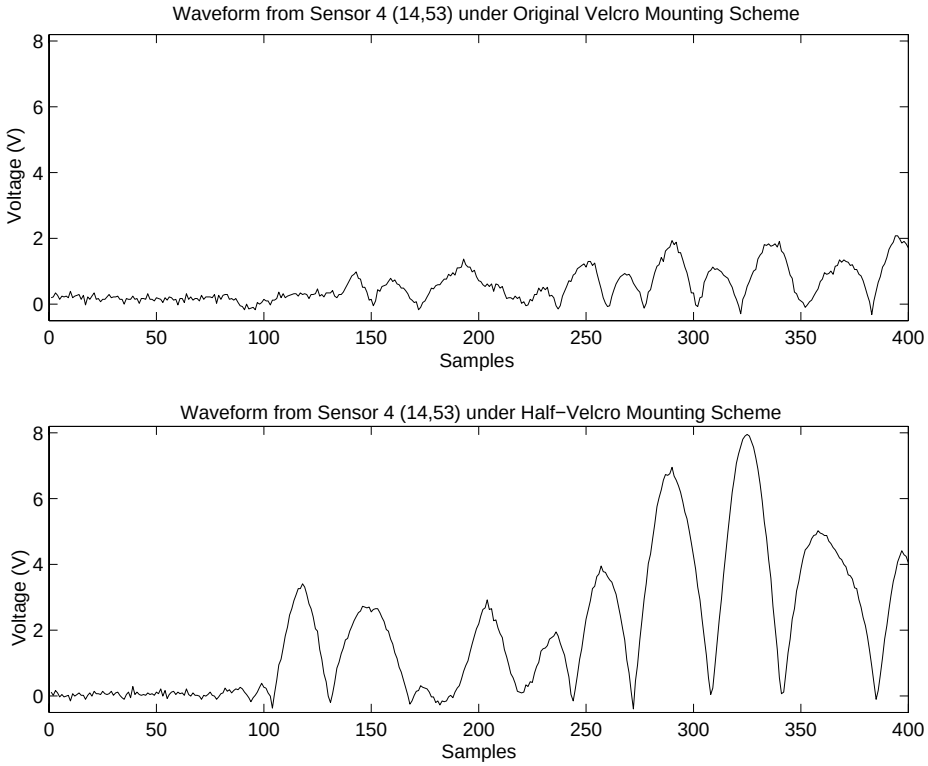


FIGURE 4-7: Comparison of waveforms between the original Velcro mounting scheme and the direct glue (half-velcro) mounting scheme. An impact at location (14,53) generated these waveforms at sensor 4.

one for the left.

4.2 Signal Conditioning Board

The signal conditioning board receives the signals from the sensors and prepares them for data acquisition. The signal conditioning stage of the prototype consisted of amplifiers and comparators for thresholding. The main problem with this design was the absence of filtering. The frequency of the signal is low to mid frequency so any high frequency noise can be filtered out quite easily using a low pass filter. The high frequency cutoff must not be made too low since the signal generated by hard impacts are of significantly higher frequency than that of knuckle taps.

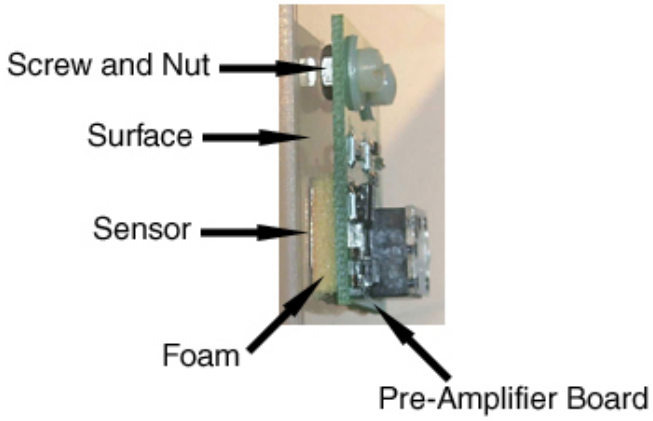
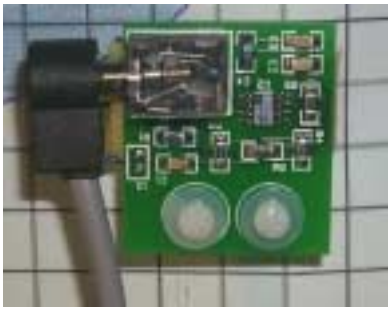


FIGURE 4-8: Mounting scheme of the final version of the pre-amplifier board.

There is also a low frequency component of the noise, which can be similarly filtered out. A bandpass filter with a low frequency cutoff of 260 Hz and a high frequency cutoff of 80 kHz was designed. After the filtering, the signal needs to be further amplified. This was accomplished using a voltage controlled amplifier, allowing adjustable gain.

Furthermore, the sensor signal is bipolar and will be sampled by analog to digital converters in the data acquisition system. It is not necessary for position determination algorithms to know whether a portion of the signal is positive or negative. Information about the sign of the signal is only useful when determining whether the tap was caused by a finger knuckle or by a hard object such as a metal ring (e.g. one can discriminate between the two taps by counting the zero crossings). Thus, the absolute value of the signal was digitized instead of the bipolar signal. This improved the accuracy of the analog to digital (A/D) conversion.

Since information about the polarity of the signal is useful, a comparator was included to provide this information. The comparator switches voltage levels (from 0 to 1) depending on the polarity of the signal. Thus, high frequency signals, such as those generated by metal taps, would result in significantly more transitions than that of a lower frequency signal (generated by knuckle taps) for a given time period. By counting the number of transitions, the type of object that generated the signals can be identified as either a knuckle or metal



(a) Front view. The board is held in place by two screws.

(b) Side view.

FIGURE 4-9: Pictures of the final pre-amplifier board.

object. It was observed that the number of transitions for a knuckle tap over a time period of 10 ms was always less than 7, thus making it straightforward to distinguish between a knuckle and metal tap. Figure 4-10 shows a picture of the signal conditioning board. A schematic can be found in Appendix B.

FIGURE 4-10: Picture of the signal conditioning board.

In addition to using the frequency information to determine the type of impact, this information could be used to vary the content generated by the system. For instance, if the system included a musical mapping, lower frequency signals could generate lower pitched musical notes or notes with softer attack characteristics.

Chapter 5

Algorithms

This chapter discusses the evolution of the algorithms used in position determination from the initial algorithms that used time-of-flight analysis to the final algorithms that incorporated cross-correlation data.

5.1 Data Acquisition

The DAQ system used in the acoustic tap tracker consisted of a NI PCI 6024E card [3]. The four input channels were sampled at 48 kHz with 12 bits of resolution. This mode of data acquisition was chosen for the development stage because of the ease of programming in Matlab. Matlab was used in conjunction with the card to develop the algorithms for position location. Once the development was complete, the entire system was ported over to a TMS320C31 Digital Signal Processor (DSP) made by Texas Instruments (TI). The system was not directly implemented in the DSP since the algorithms would have to be programmed in either assembly or C and testing would be more difficult than with Matlab.

5.2 Algorithms using Time-of-Flight Analysis

The first algorithm that was developed involved using differential time-of-flight analysis to locate the position of the tap. A minimum of three sensors is required to locate the position

of a tap. An extra sensor was added for redundancy and error checking. The data from groups of three sensors was used to locate the position of the impact. The five data groups of sensors were 123, 124, 134, 234, and the data from all the sensors[1]. Thus, five guesses for the position are generated per tap.

The inverse mapping from differential timings to impact coordinates is nonlinear involving the closest intersection of three hyperbolas from the three sensors in a data group [10]. A linear least-squares fit was used to approximate the system. A second order fit would be adequate since a third order model might overfit the system.

The time-of-flight method requires that an arrival time be assigned to each sensor. The DAQ system triggers on the first sensor (upper right sensor), but a set of pre-trigger data is obtained so that if any of the other sensors receives a signal before the first sensor trigger, this information is included in the signal. The threshold must be large enough so that it is kept comfortably above noise while still obtaining the signal itself. Triggering on the first sensor was chosen arbitrarily. There is no record of absolute time so relative arrival times must be used. In general, the arrival time assigned to a waveform is the time of its first maximum. Since only the absolute value of the signal is used, the maximum may actually be either a maxima or a minima. Figure 5-1 shows the response of sensor 4 for a series of knuckle taps moving across the pane of glass from left to right. As the distance between the tap and the sensor increases, the time delay increases. In addition, the signal received by the sensor is more attenuated as the distance increases.

5.2.1 Calibration

The algorithm consists of two main components: calibration and forward determination. Calibration was used to obtain the coefficients for the linear-least squares model. The calibration routine consists of obtaining the timing information for a set of pre-determined calibration points over a set number of trials. For instance, the calibration routine used in

[1]The numbers in the data group represent the sensor number. The sensors are numbered starting at the top right and counting clockwise. Thus, the top right sensor is sensor 1, the bottom right sensor is sensor 2, and so on (see Figure 4-6).

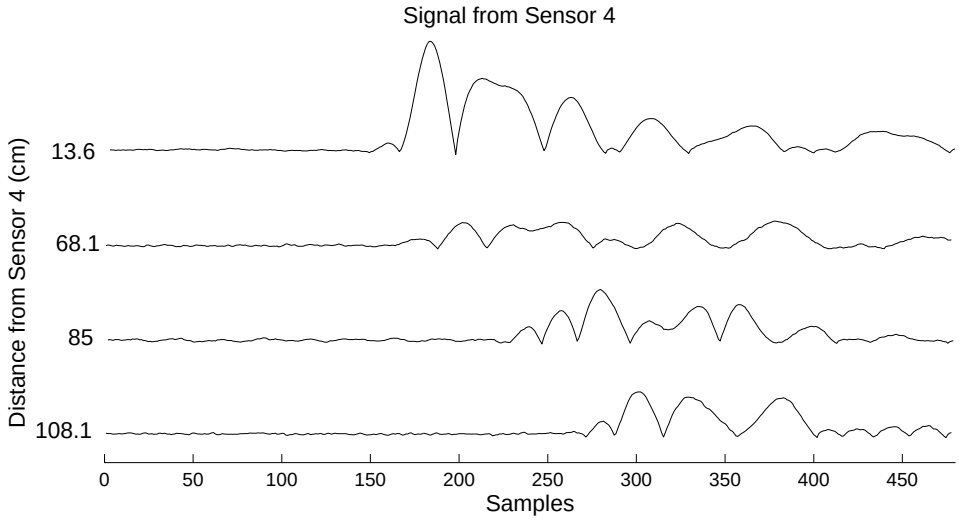


FIGURE 5-1: Signals from sensor 4 for a series of knuckle taps moving across the pane of glass from left to right.

testing consisted of 20 calibration points with five trials each[2]. The user taps five times on each calibration point. The function calculates the time differences between pairs of sensors (i.e. between 1 and 2, 1 and 3 for the data group consisting of sensors 1, 2, and 3) for each trial and each point. Any time differences that deviate significantly from the rest are neglected, and a set of time differences based on the remaining data points is calculated. This procedure is repeated for each point.

The model [11] used was of the form of Equation 5.1:

$$x(t_{12}, t_{13}) = c_1 t_{12}^2 + c_2 t_{13}^2 + c_3 t_{12} t_{13} + c_4 t_{13} + c_5 t_{12} + c_6 \tag{5.1}$$

t_{12} is the time difference between sensors 1 and 2, and t_{13} is the time difference between sensors 1 and 3. A similar equation holds for the y-coordinate. A different model is used when considering the data from all the sensors. Equation 5.2 describes the model used for

[2]The number of trials is an input to the program that can be changed.

this case.

$$x(t_{12}, t_{13}, t_{14}) = c_1 t_{12}^2 + c_2 t_{13}^2 + c_3 t_{14}^2 + c_4 t_{12} t_{13} + c_5 t_{12} t_{14} + c_6 t_{13} t_{14} + c_7 t_{12} + c_8 t_{13} + c_9 t_{14} + c_{10} \tag{5.2}$$

A separate time difference matrix is constructed for each group of sensors. For the groups that consist of three sensors, a 20 by 6 time difference matrix is constructed. There is one row for each calibration point (i.e. 20) and one column for each coefficient. For the group of all sensors, a 20 by 10 time difference matrix is constructed. For a set of calibration taps, a matrix equation of the form of Equation 5.3 is used to determine the x-coordinate.

$$X = M \times C \tag{5.3}$$

M is the matrix of time differences. C is the column of coefficients to determine the x-coordinate, and X is the column vector of x-coordinates. A similar equation holds for the y-coordinate. Using simple matrix manipulation, C can be calculated by using the following equation.

$$C = M^{-1} \times X \tag{5.4}$$

M is a non-square matrix; therefore, the inverse matrix is calculated by performing a singular value decomposition (SVD) [11] to find the least-squares solution. The coefficients are then saved to a file so that the forward determination algorithm can read the coefficients to be used in calculating the position. The entire calibration function can be found in Appendix C.

5.2.2 Foward Determination

The forward determination algorithm acquires a tap and calculates the time differences as was done in the calibration function. Using Equation 5.3 above, an estimate of the position is calculated using a data group of sensors. Since there are five data groups, each tap generates five guesses for the location. Outlying positions are rejected based on examining the averages, and the position is calculated by averaging the remaining points.

The Matlab code of the entire routine can be found in Appendix C. When the system was run, mixed results were obtained. Sometimes, the position determination was very accurate (i.e. less than 1 cm deviation); however, sometimes the position determination was extremely inaccurate resulting in guesses that were off by 20-30 cm.

By examining the guesses that were generated, one cause of the inaccurate position location was found. For some of the wildly incorrect guesses, a correct guess was among the five guesses generated. However, the other incorrect guesses were more closely spaced together resulting in the correct guess being rejected. More information would be required to extract the correct guesses.

To solve this problem, the relative attenuation of the signals between sensors was examined to provide more information to locate the position of the tap. Theoretically, there is an indirect relationship between the attenuation of a signal and the location of the source. For instance, if the signal received by sensor 1 is attenuated the most relative to the other sensors, and the signal received by sensor 3 is attenuated the least, then it can be determined that the source is located closer to sensor 3.

An algorithm based on the peak amplitude information was devised to use in conjunction with the timing information. A similar calibration routine was used to determine the coefficients for the amplitude model. The forward determination algorithm was modified so that it read both the timing coefficients and amplitude coefficients. Equation 5.3 was then solved twice for each set of coefficients so that 10 guesses of the location were generated. The additional guesses from the amplitude information did reduce some of the inaccuracies mentioned above so that correct guesses would not be rejected; however, it did not help the majority of inaccurate guesses.

One explanation is the unpredictable behavior of the signals. For example, a tap farther away from a particular sensor would not always be more attenuated than the signals received by closer sensors. If this non-ideality occurred consistently, then the model would have compensated for this, and a correct position would have been predicted. However, this did not occur consistently resulting in the inaccurate guesses. The cause of this non-ideality

is inhomogeneities in the glass surface particularly near the bottom right side of the pane. In addition, the amplitudes of latter peaks is dominated by modal oscillations of the glass pane.

The surface used for this research was part of a conference room wall. According to eye-witnesses, the right side of the glass had been stressed in order for the glass to fit into the slot. This accounts for why taps on this side of the glass were unpredictable. Figure 5-2 shows two sample waveforms demonstrating the inhomogeneities of the bottom right side of the glass. The two taps corresponding to the signals below are of roughly the same degree of impact. The tap generating the top waveform is farther from sensor 2 than the tap generating the bottom waveform; however, the closer tap resulted in greater attenuation.

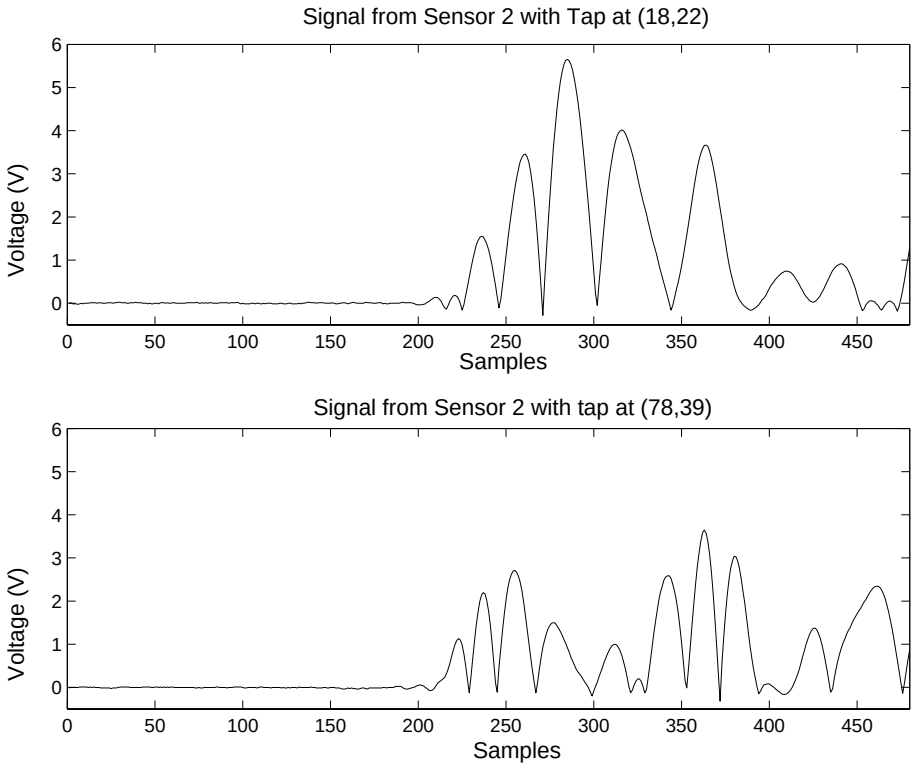


FIGURE 5-2: The two waveforms above demonstrate the inhomogeneities associated with sensor 2. Although the lower waveform was generated by a tap closer to the sensor 2, it exhibits more attenuation than a signal generated by a tap farther away.

Furthermore, the main cause of the inaccuracies of the timing algorithm was the dispersive

nature of the knuckle tap. The variable amount of low-amplitude, higher-frequency, dispersive deflection that arrives before the main wavefront complicated determination of signal arrival time. For instance, Figure 5-3 shows the signals received by each of the channels when the impact is located at (18,77) near sensor 4.

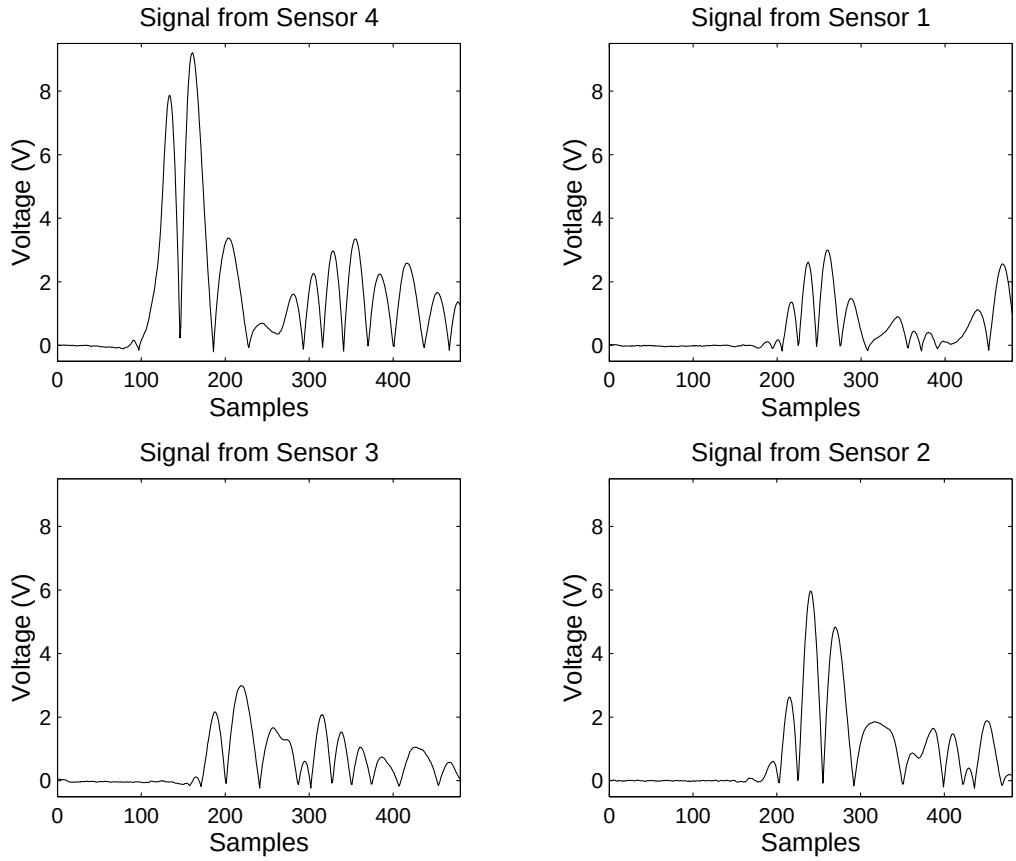


FIGURE 5-3: Waveforms generated by each sensor with an impact at (18,77) (closest to sensor 4).

The signal received by channel 1 exhibits a significant amount of dispersion. The arrival time determination algorithm locates the first peak that is above a certain threshold. It is assumed that anything below that threshold is simply noise. Sometimes the dispersion is below this threshold and does not factor into the arrival time. However, for another tap at the same location, this same dispersion will have a larger amplitude and thus will be used as the leading edge. Figure 5-4 shows two signals generated by two taps both at location (18,77). Based on these signals, the exact start of the signal is unclear.

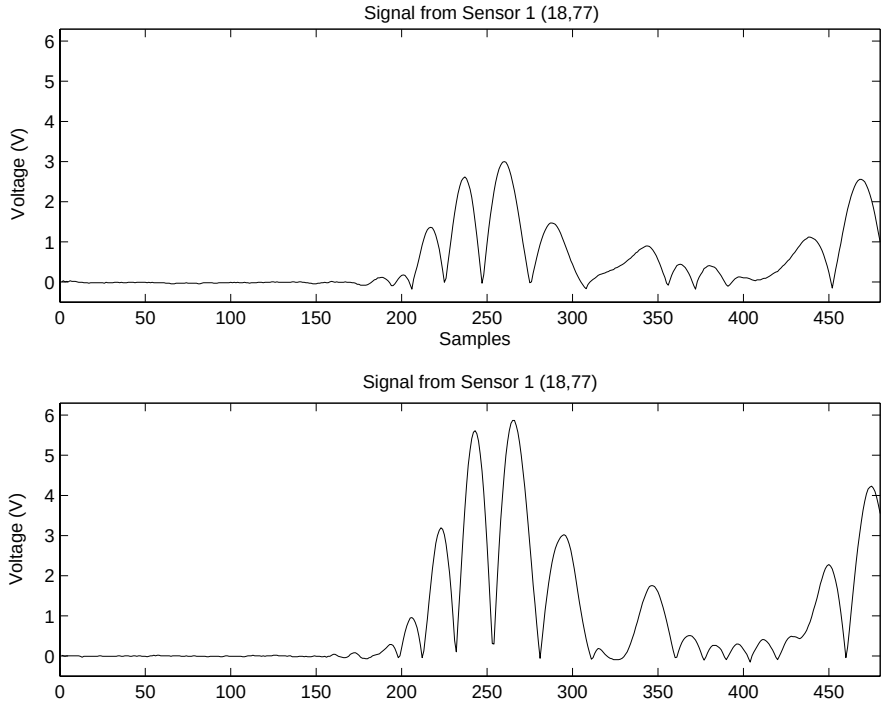


FIGURE 5-4: Dispersion comparison between two taps both at location (18,77).

These variations cause the inaccuracies of the algorithms. If the time of arrival for just one sensor is inconsistent with the model, then the guesses using information from the data groups that do not include data from the "bad" sensor will be correct. The incorrect guesses could then be rejected. However, it is not known a priori whether or not the data is inconsistent or a valid set of data. For instance, suppose the user tapped at location (80,20). The signal from sensor 4 displays a lot of dispersion so a time of arrival is used that is not consistent with the model. All the guesses that use the timing information from sensor 4 will be invalid. Thus, there is only one correct guess from data group 123 since it is the only group that does not use the incorrect data from sensor 4. If all five guesses point to different areas of the glass, rejecting the bad points is not straightforward. The situation is even more complicated when several incorrect guesses are closely spaced while the correct guess deviates significantly from these points. Because several points are clustered together,

it would appear that the correct point is located in that cluster.

In addition, more problems occur if the times-of-arrivals for more than one sensor is inconsistent with the model because then all the guesses generated will be incorrect. Even if a complicated algorithm could be devised that solved the problem with inconsistent data from one sensor, no algorithm could solve the aforementioned problem. It was found that the more frequent case here was when more than one sensor yielded inconsistent time of arrivals simultaneously.

Since the system was not reliable using the time-of-arrival and amplitude information, other techniques were investigated to improve the accuracy of the system. The best technique would yield consistent results among a series of taps at one location. A technique that met this criterion would improve the accuracy significantly.

5.3 Algorithms incoporating Normalized Integrated Amplitude

The problem with using the attenuation information was the non-ideal effects due to the properties of the glass surface used. Instead of analyzing the amplitude of the signals, normalized integrated amplitude should provide more useful information since attenuation due to inhomogeneities in the glass would not be a significant factor. By normalizing the integrated amplitude to the maximum, the non-ideal effects due to location dependent attenuation should be factored out. Thus, among a series of taps at one location, the signals generated by the taps that are received by one sensor will look approximately identical when normalized.

It was assumed that the location-dependent attenuation caused by the inhomogeneities in the glass attenuated the entire signal by the same amount. However, this assumption was incorrect, which led to the unsuitability of the normalized integrated amplitude method. For example, Figure 5-5 illustrates the problem.

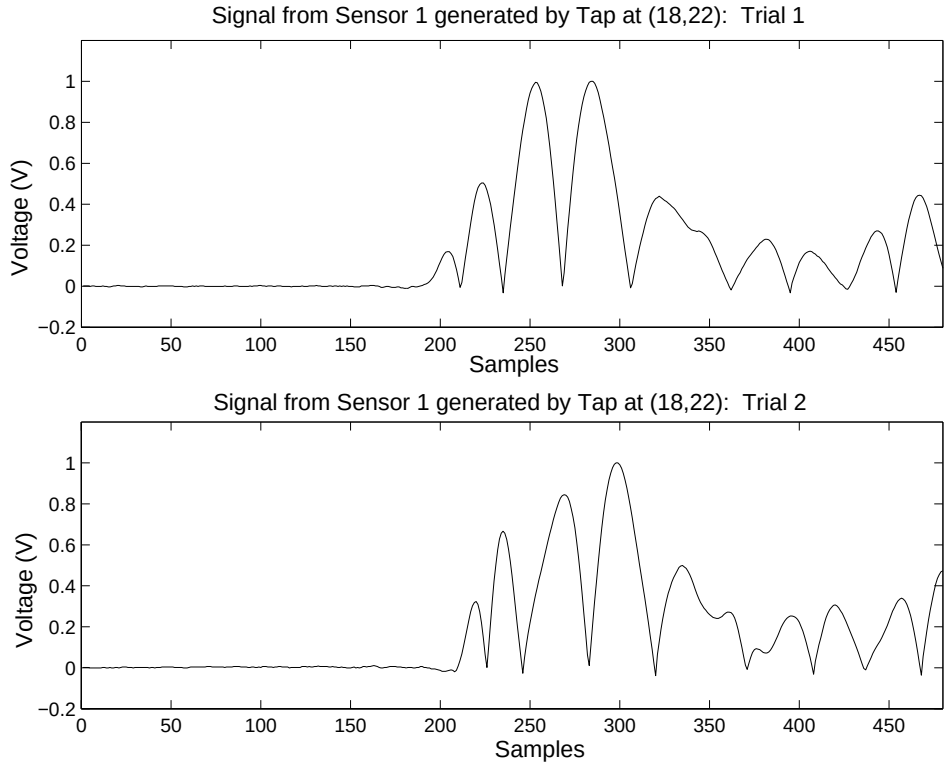


FIGURE 5-5: Waveforms showing normalized amplitudes of taps generated at location (18,22).

Two taps located at position (18,22) generated the two signals shown at sensor 1. The normalized amplitudes of the waveforms vary greatly. Because different components of the signal are attenuated differently, the integrated amplitude calculated for the two signals is very different, resulting in a guess that the tap was located further away than it really was. This happened quite frequently thus rendering the technique unsuitable.

All of the above techniques involved direct analysis on the signal itself. They all depended on analyzing one component of the signal at a time. For instance, the timing analysis examined the first arrival time, which varied due to the non-idealities of the glass. Both of the amplitude techniques relied on the maximum peak. To yield accurate results, the technique would need to analyze the entire signal so that these variations would be smoothed out. The final technique used involved examining the cross-correlation data between signals.

5.4 Algorithms incorporating Cross-Correlation Data

The cross-correlation method resembles the differential time-of-flight analysis except cross-correlations are used instead of time differences between signals. Figure 5-6 shows typical cross-correlation waveforms.

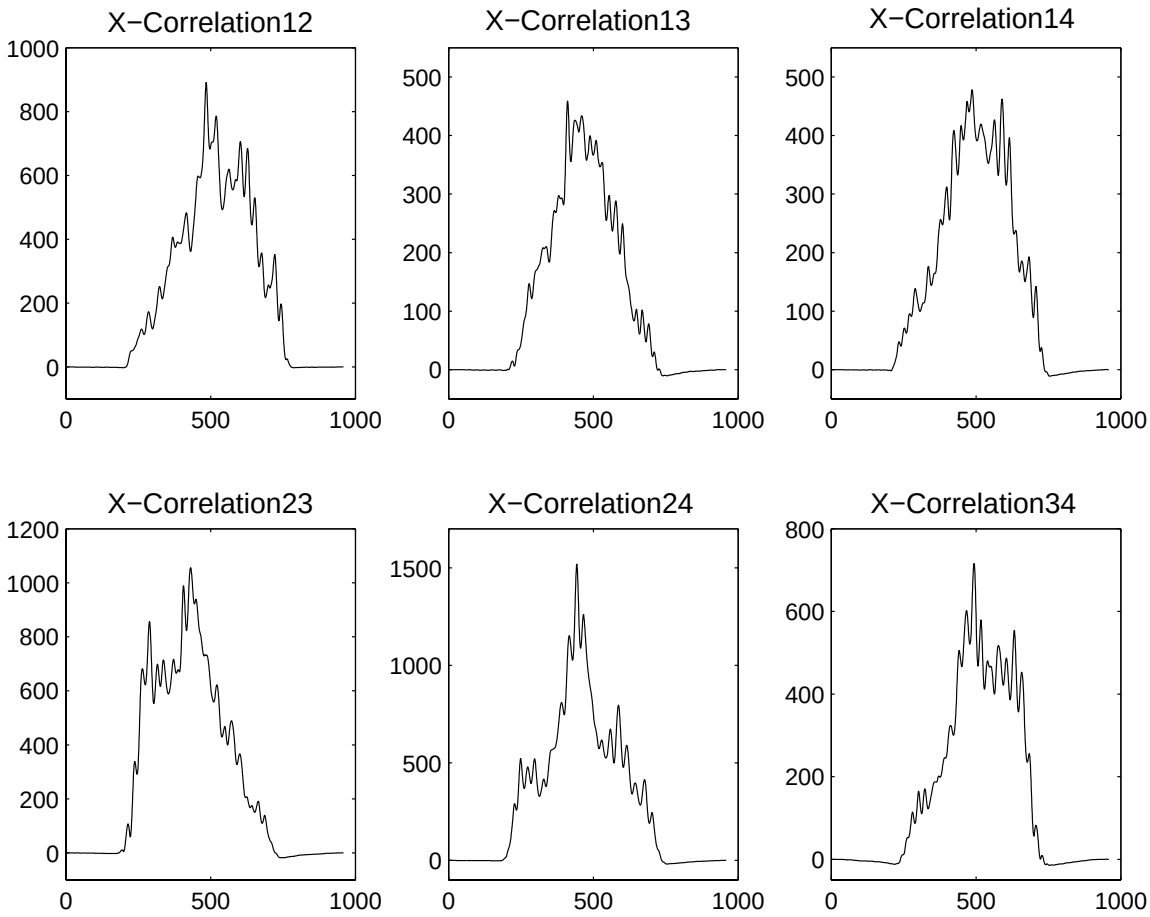


FIGURE 5-6: Cross-correlations between sets of signals for a tap at (55,65)

The cross-correlation between sensors 2 and 4 in Figure 5-6 has a large peak centered at 444 signifying the location of maximum correlation between signals 2 and 4. Among a series of taps at the same location, each generates a very similar cross-correlation waveform that has a maximum at 444. Consistent behavior among a series of taps at the same location forms the foundation for an accurate position determination algorithm. Figure 5-7 shows

the cross correlation for a tap at the same location as Figure 5-6. One can see that the waveforms are virtually identical.

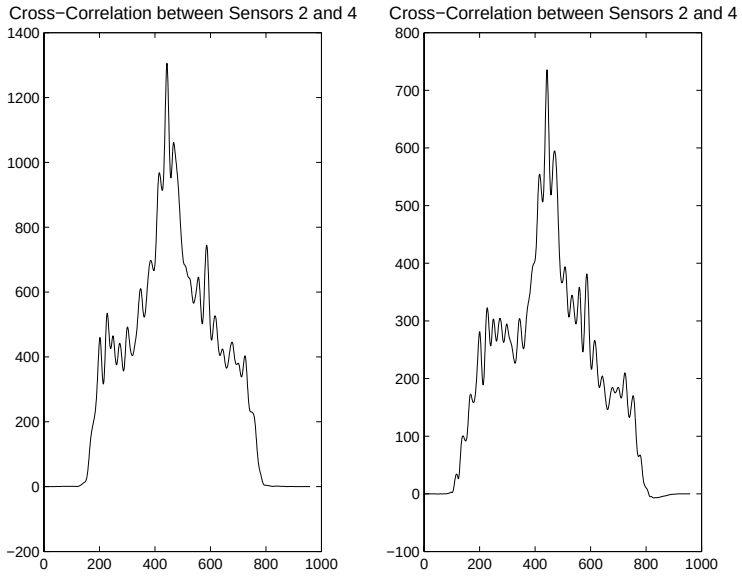


FIGURE 5-7: Both waveforms are the cross-correlations between sensors 2 and 4 for a tap located at (78,59). The waveform on the left was generated by one trial, and the waveform on the right was generated by a different trial. The waveforms are virtually identical showing the consistency of the cross-correlation. The left waveform peaks at location 444 while the right waveform peaks at 443.

However, not all of the cross-correlations were as clean as those in Figure 5-7. Because of the stress on the lower right portion of the glass surface, taps in this section almost always generated cross-correlations like those in Figure 5-8.

This cross-correlation differs from that of Figure 5-7 in that there is significantly more ripple around the maximum. Thus, for a series of taps at one location, the maximum actually oscillates between a set of values. This is depicted in Figure 5-9. Two taps at location (101,29) generated the cross-correlations. The maximum for the cross-correlation on the left occurs at 396 while the second largest peak occurs at 468. The situation is reversed for the cross-correlation on the right; the maximum occurs at 467 with the second peak at 397.

This ambiguity could cause some problems. For example, suppose during calibration, the model uses the values corresponding to left cross-correlation. Since the model is fit around these data points, the model would predict an incorrect guess for a tap at location (101,29)

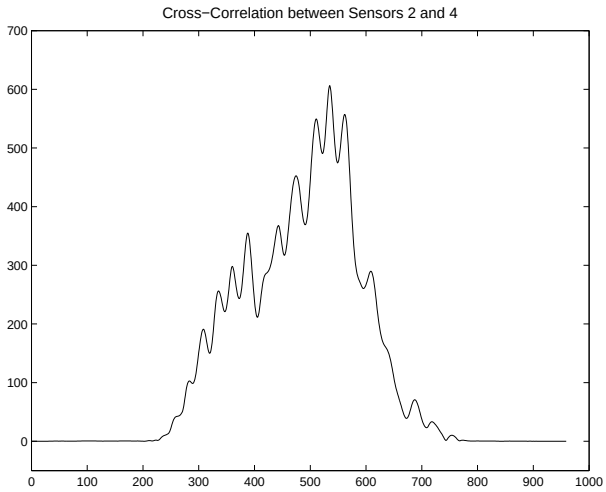


FIGURE 5-8: Cross-correlation between the signals from sensors 2 and 4 for a tap at location (98,22).

that yields the correlation data corresponding to the right cross-correlation. This problem is amplified further when an ambiguity exists between more than two peaks.

Since the "correct"[3] data is present in the cross-correlation, techniques were explored to try to consistently extract the same "maximum" for these signals with ambiguous maxima.

5.4.1 Peak Extraction Using Averaging

The first technique explored was an averaging method in which the two largest peaks are extracted, and the average of the two locations is taken as the point of maximum correlation. This worked reasonably well when deciding which of two peaks to use as the maximum. However, the algorithm did not work well when deciding between more than two peaks such as with the cross-correlation shown in Figure 5-8. Various other averaging techniques such as averaging more than two peaks were also implemented but also did not yield reliable results.

[3]Correct data refers to the data that was used in creating the model

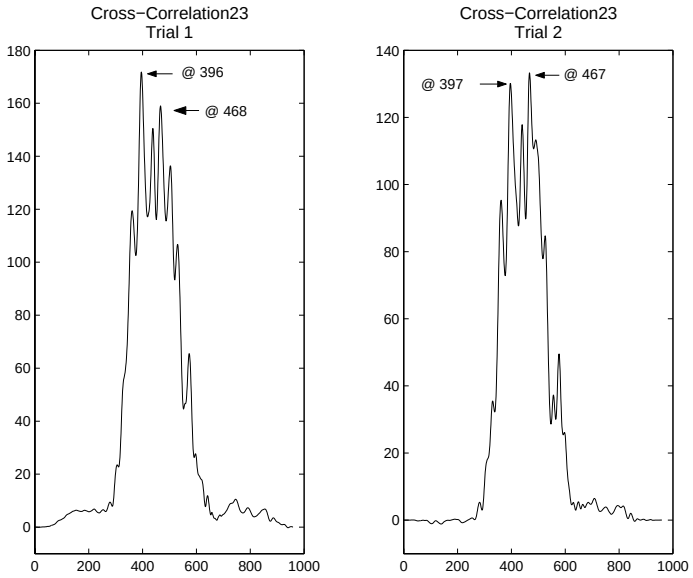


FIGURE 5-9: The cross-correlation on the left was generated by a tap at location (101,29). The cross-correlation of the right was generated by a similar tap at the same location. These two figures show the ambiguity in resolving the maximum peak.

5.4.2 Peak Extraction Using Polynomial Fitting

The second technique explored involved fitting a polynomial to the cross-correlation curve and finding the maximum of the fitted polynomial. The fit of the polynomial sometimes skewed the data. For instance, if there were more peaks located at lower sample values, the polynomial was fit well for these points but fit badly for the higher sample values, which is where the real maximum occurs. Thus, the point of maximum correlation returned as a result of the polynomial fit was often skewed.

5.4.3 Peak Extraction Using the Ratio of Peaks

The final technique was based on the notion that since the "correct" information was located within the cross-correlation signal itself, different combinations of maxima were calculated. For instance, suppose the cross-correlation data was ambiguous[4] for one set of

[4]Ambiguous is used to designate data that does not have a clear maximum for a series of taps at the same location. For instance, the cross-correlation in Figure 5-9 is ambiguous since the the location of the maximum varies for different taps at the same location. However, Figure 5-7 is not ambiguous since the location of the maximum is always the same.

cross-correlations but completely unambiguous for the other sets. For the ambiguous set, the two largest peaks were extracted since one of the two peaks was most likely to be a "correct" value. Thus, instead of one vector of cross correlations being constructed, two are used. The forward determination algorithm is then run with both cross-correlation vectors resulting in twice the number of guesses. In principle, one set of guesses corresponding to the "incorrect" cross-correlation would mispredict the location of the tap while the second "correct" set would accurately predict the location. Figure 5-10 details what the distribution of guesses was anticipated to look like using this method.

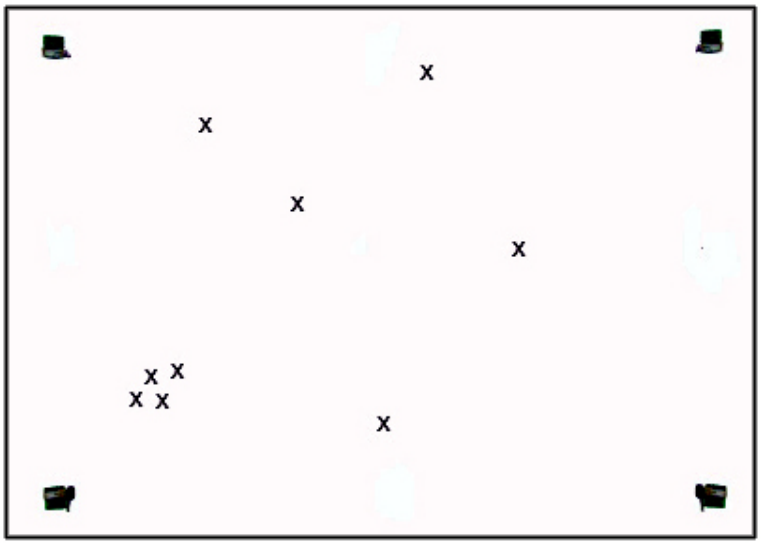


FIGURE 5-10: Predicted distribution of guesses.

The calibration routine for this strategy is similar to that of the other techniques described above. The calibration routine uses five taps at each location to determine the cross-correlation characteristics of each location. No additional functionality in the calibration routine is needed to compensate for the multiple peaks since the calibration function is already determining the cross-correlation data that is representative of each location by rejecting deviant points and averaging the rest.

The forward determination algorithm is slightly different in that now the cross-correlation

between pairs of sensors is calculated instead of time differences. In addition, as mentioned above, multiple vectors of cross-correlation data are generated for ambiguous peaks so the forward determination algorithm must be executed for each of these vectors. An example of how the algorithm works to resolve the ambiguity is described the next section.

If the ambiguity resolution function is executed for all sets of cross-correlation data, then unnecessary combinations of data would be generated for cross-correlations that have no ambiguity. For instance, the cross-correlation in Figure 5-7 always has its maximum at location 444. Finding the second largest peak for this case is unnecessary since it is known that it is incorrect. Furthermore, six cross-correlations compose the correlation vector. By performing the ambiguity resolution function on each of the cross-correlations, 64 combinations would be generated with four guesses of the positions each. The majority of the 256 guesses generated are incorrect and complicate the extraction of the correct position.

Thus, another algorithm was implemented to determine when an ambiguity exists, and when there is no ambiguity. By visually looking at the graphs of cross-correlations, one can tell when an ambiguity might exist if peaks are close together in magnitude. For instance, for the cross-correlation on the right of Figure 5-9 above, the ratio of the maximum peak to the second peak is approximately 1.023. Figure 5-7 displays no ambiguity, and the ratio of peaks in that case is approximately 1.23. By examining the peak ratios of different signals, it was determined that when the ratio of peaks was under 1.05, an ambiguity exists; thus the ambiguity resolution is performed. If the ratio is larger than this, then there is no ambiguity so ambiguity resolution is not necessary.

5.5 Example

Figure 5-6 shows the correlation data as a result of a tap at location (55,65). The cross-correlation between sensors 1 and 4 exhibits an ambiguity resulting in the following cross-correlation matrix:

$$\begin{pmatrix} xc12 & xc13 & xc14 & xc23 & xc24 & xc34 \\ 485 & 411 & 486 & 431 & 443 & 493 \\ 485 & 411 & 590 & 431 & 443 & 493 \end{pmatrix}$$

Two vectors were generated because of the ambiguous maximum in the cross-correlation between sensors 1 and 4. The forward determination algorithm is then executed using both vectors of the above matrix resulting in four guesses for each vector for a total of eight guesses. These guesses are listed below. The guesses are plotted in Figure 5-11.

$$x_{position} = \{95.72, 120.2, 100.01, 48.04, 95.72, 56.84, 212.45, 48.04\}$$

$$y_{position} = \{43.92, 1.86, 143.58, 65.7, 43.92, 1.32, 2.62, 65.7\}$$

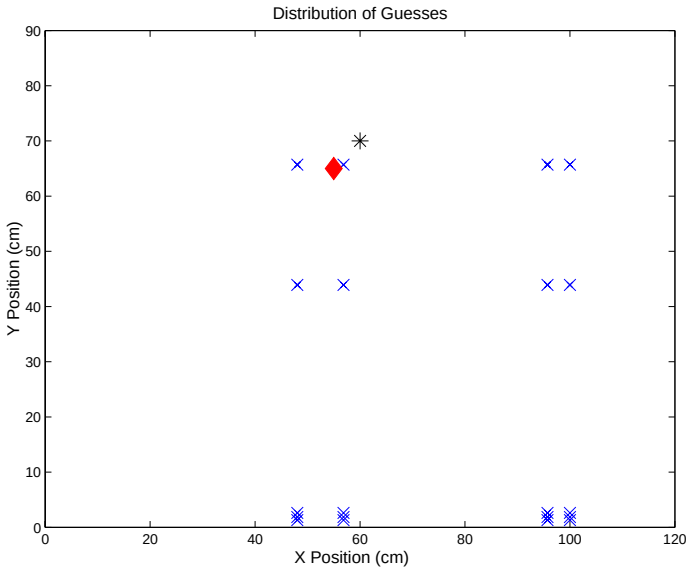
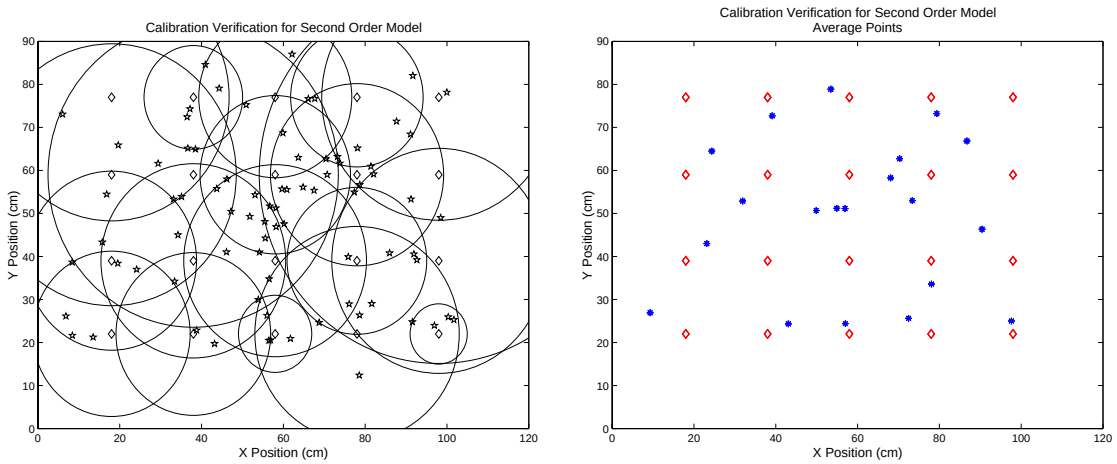


FIGURE 5-11: Distribution of guesses for a tap at location (55,65). The crosses represent guesses. The diamond represents the actual location of the tap, and the star represents the closest calibration point.

The diamond represents the actual location of the tap, the crosses represent the guesses, and the star represents the closest calibration point. Determining the closest calibration

point is discussed later. It can be seen that the smallest spacing between guesses is around (98,68) and (98,45). The algorithm would then decide that all points not in this area were mispredicts and therefore neglect them. The remaining guesses are then averaged. For this example, the algorithm would decide that the tap occurred at the incorrect location (98,56.5). This was the same problem encountered before with the differential time-of-flight analysis. One vector out of the two in the matrix fits with the model and thus should predict the correct point. All of the guesses that correspond to this vector should cluster around the same area; however, this clustering does not occur. It was hypothesized that the model was most likely responsible for this behavior.

To test the accuracy of the model, the cross-correlation data used to generate the model was run with the forward determination algorithm. The guesses were then plotted against the reference locations. This is shown in Figure 5-12(a) and Figure 5-12(b).



(a) Plot of calibration guesses (blue stars) against reference locations (red diamonds). The radius of the circles centered at a reference location represents the maximum deviation of a guess from that reference location.

(b) Plot of the average of calibration guesses (blue stars) against reference locations (red diamonds).

FIGURE 5-12: Figures generated by running the forward algorithm with the correlation data of the calibration points for the second order model.

Some of the guesses deviate significantly from the reference locations. In fact, if the user were to tap at location (18, 77) and generated correlation data identical to that of data used

in the calculation of the coefficients, the model would predict a location that deviated by approximately 13 cm. A fitted model will not pass directly through the reference locations, but the guesses generated should not deviate from the references significantly. The fit should not be constrained to actually pass through each of the calibration points so that points located between calibration points can be extrapolated.

To improve the accuracy of the model, the order was increased from second order to third order. Equation 5.5 describes the new third order model used.

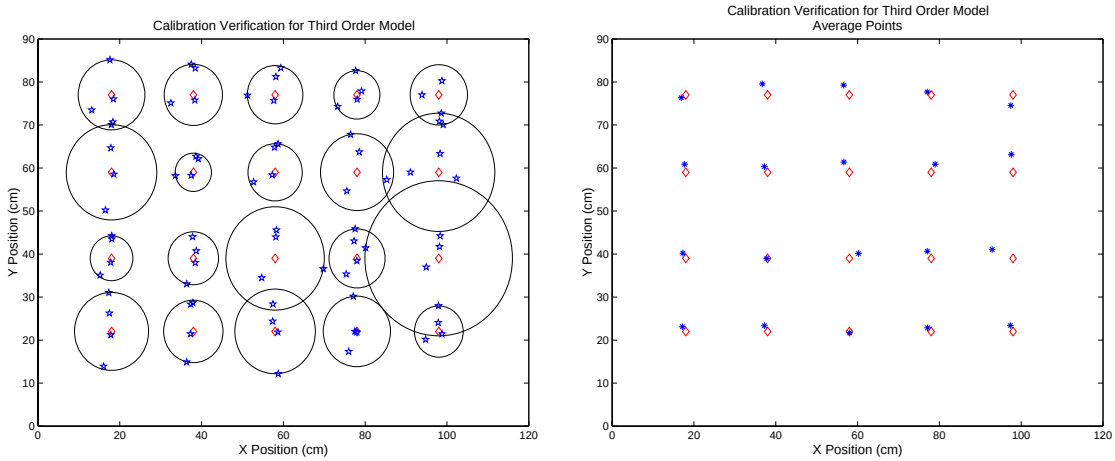
$$\begin{aligned} x = \quad & c_1 t_{23}^3 + c_2 t_{24}^3 + c_3 t_{34}^3 + c_4 t_{23}^2 t_{24} + c_5 t_{23}^2 t_{34} + c_6 t_{24}^2 t_{23} + \\ & c_7 t_{24}^2 t_{34} + c_8 t_{34}^2 t_{23} + c_9 t_{34}^2 t_{24} + c_{10} t_{23}^2 + c_{11} t_{24}^2 + c_{12} t_{34}^2 + \\ & c_{13} t_{23} t_{24} + c_{14} t_{23} t_{34} + c_{15} t_{24} t_{34} + c_{16} t_{23} + c_{17} t_{24} + c_{18} t_{34} + c_{19} \end{aligned} \tag{5.5}$$

The subscripts indicate the sensors used in the cross-correlation vector. A calibration check was then performed on the new model yielding the plots in Figure 5-13(a) and Figure 5-13(b).

The model does not pass exactly through the calibration points, but the average of all the points fits very nicely. This is shown in Figure 5-13(b). Making sure that the model accurately reflects the calibration data while not passing exactly through the points ensures that the model is not overfitting to these points. When the acoustic tap tracker was run incorporating these new algorithms, the accuracy of the position determination increased significantly from the other techniques but still resulted in some inaccurate guesses.

The correct location was among all the guesses that were generated from the algorithm. The algorithm rejected these points because there were more bad guesses that cluster together than good guesses. A new algorithm was devised to try and extract the good guesses.

By analyzing the cross-correlation matrix, it can be determined which calibration points are closest to the tap. Thus, the section of the glass can also be determined. By narrowing down the section of the glass, more points can be rejected from the list of all the guesses.



(a) Plot of calibration guesses (blue stars) against reference locations (red diamonds). The radius of the circles centered at a reference location represents the maximum deviation of a guess from that reference location.

(b) Plot of the average of calibration guesses (blue stars) against reference locations (red diamonds).

FIGURE 5-13: Figures generated by running the forward algorithm with the correlation data of the calibration points for the third order model.

Determining the calibration point that is nearest to the tap is not so straightforward. Two algorithms were devised, and a combination of the two was used in the final algorithm.

The first algorithm compared the cross-correlation vector (for ambiguous points, all cross-correlation vectors are compared) to each of the cross-correlation vectors for the calibration points. The sum of the absolute value of all deviations between the calibration and candidate points for a cross-correlation vector were summed together, and the calibration point with the minimum sum was deemed as the closest point to the tap. This was not always accurate. For example, a set of cross-correlation data was declared as ambiguous if the ratio of the main peak to the second peak was below the threshold. A problem occurs if the point's correlation data is ambiguous, but the peak is above the threshold. In this case, at least one of the cross-correlations is incorrect. When taking the deviations as described above, this deviation could be quite large, making the overall sum larger than the deviation sum of another point. Thus, the closest calibration point is incorrectly determined.

The second algorithm compares the cross-correlation vectors by taking the difference be-

tween a point's cross-correlation and each row of the calibration cross-correlation matrix. Instead of summing all the deviations, the number of individual cross-correlations that are below a certain threshold are counted. The point with the maximum number of correlations below this threshold is deemed as the calibration point closest to the tap. Occasionally multiple calibration points have the same number of deviations below the threshold (i.e. when a tap occurs at the midpoint of those calibration points). When this situation occurs, the information provided by the first algorithm is incorporated to make a decision as to which calibration point the tap is closest to. The procedure described above is repeated for all cross-correlation vectors generated by the tap.

When only one cross-correlation vector is generated (i.e. the cross-correlation is unambiguous), the calibration point that is determined as a result of the algorithms is used to reject deviant points and pinpoint the location of the tap. However, when more than one cross-correlation vector is generated (i.e. for an ambiguous set of cross-correlation data), all the calibration points that are returned must be examined to decide which point the tap is closet to. The algorithm combines all the predicted nearest points into a vector and determines the mode of the vector. This point is then used for the rejection and pinpoint analysis. When several calibration points occur with the same frequency, then the information from the first algorithm is incorporated to make a better decision. A flowchart of the algorithm decision process is shown in Figure 5-14.

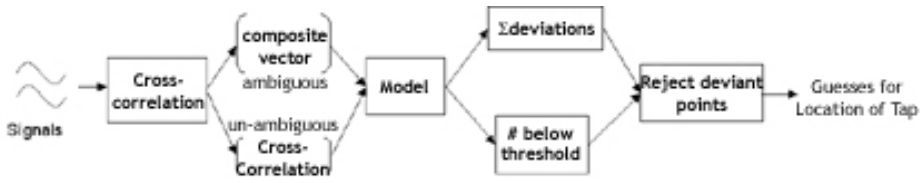


FIGURE 5-14: Flowchart of algorithm decision process.

This additional information is not useful if it does not predict a point that is among the ones being considered by the second algorithm. In this situation, the algorithm has no further information and arbitrarily picks a point. This does not result in very many inaccuracies since the two points being considered are in the same section and would both reject the

same values and pinpoint the location to the same place. The only time errors occur is when the two points being considered are in different sections, and the algorithms picks the incorrect point resulting in the wrong guess being rejected. This situation does not occur frequently (approximately 1/30 taps).

The situation that occurs most frequently is when the algorithm unanimously picks one calibration point as being closest to the tap. The additional functionality of the algorithm described above was added so that the correct location would be determined during the infrequent cases. Taps along the right side of the glass generated ambiguous cross-correlation data and thus used the additional analysis of the algorithm. All of this complex analysis is required to help in rejecting bad points while narrowing down the possible locations for the tap.

As a result of all of this analysis, each tap results in a deviation from the actual location of less than 5 cm. For the example discussed above, the new algorithm guesses that the location is (56,65). The actual location is (55,65). All the Matlab code can be found in Appendix C.

The DAQ system is in the process of being ported to a DSP. The DSP is particularly adept at performing cross-correlations at a high speed and, thus switching the DAQ system would not result in any performance loss.

5.6 Hard Impacts

As described in Chapter 3, hard impacts generated by a metal tap propagate via a different mode than knuckle taps. The waves generated by knuckle taps are flexural waves while the waves generated by hard impacts are most likely acoustic in nature. Acoustic waves do not suffer from such large dispersion, and the sharp leading edge of the signals permits the use of differential time-of-flight analysis. Furthermore, position determination should be more accurate than that of the knuckle taps because of the consistency of the arrival time. A signal generated by a tap at location (40,50) is shown in Figure 5-15.

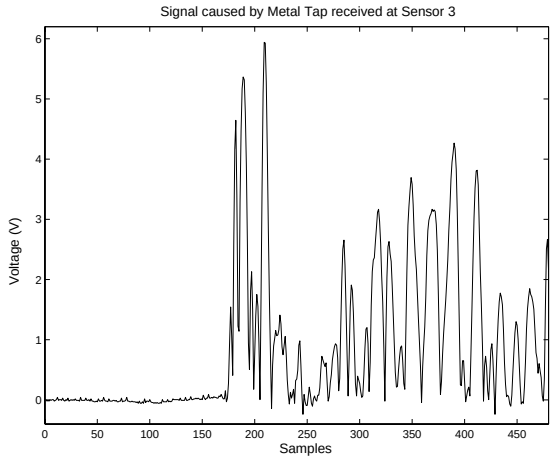


FIGURE 5-15: Signal generated by a metal tap at location (40,50).

When the system is executed using time-of-flight analysis, it was observed that the resolution of the system was approximately 10 cm. In other words, the system was not able to differentiate between taps that are located less than 10 cm apart. The cause of this inaccuracy was of a different nature than that of the knuckle taps. The problem here was in the sampling rate. The maximum sampling rate of each channel is 48 kHz while the waves generated by the hard impact propagate at a speed of approximately 3500 m/s. For a surface that is approximately one meter long and a sampling rate of 48 kHz, this only allows for approximately 10 cm of resolution.

Cross-correlation data was analyzed but was not useful because of the coarse sampling. Solutions to this problem are proposed in Chapter 7.

Chapter 6

Further Testing of the Portable Acoustic Tap Tracker

This chapter presents results from testing the acoustic tap tracker with another glass surface to determine the dependency of the algorithms to the surface.

All of the analysis described thus far was performed using a tempered, shatter-proof glass pane that was part of a glass wall enclosing a conference room. In general, all versions of the acoustic tap tracker have only been implemented using this surface. It is believed that certain properties of this particular pane of glass affected the robustness of the system. For instance, because of the stress on the lower right part of the glass, impacts at this location were sometimes unpredictable complicating the position determination. Additional functionality had to be added to the algorithms to compensate for these effects. Furthermore, the amplitude and normalized integrated amplitude analyses did not work well because of the inhomogeneities in the glass and inconsistent location dependent attenuation.

If the properties described above are unique to this pane of glass, then some of the analysis that was not usable with the system using the conference room wall might be usable with a new pane of glass. For instance, if the system is ported to a new pane that does not suffer from the problems of the conference room wall, then the amplitude and normalized

integrated amplitude analyses might work well. Using the guesses for position from that analysis in conjunction with the current position determination algorithm would better resolve the actual location by eliminating deviant guesses. For these reasons, a different pane of glass was chosen to see the effects of porting the system.

The new surface chosen was a glass wall in the atrium of the MIT Media Lab. The glass wall is significantly thicker than the glass in the conference room and is mounted differently. The conference room wall (used in testing) was supported by approximately half a centimeter of heavy rubber grout connecting it to adjacent panes. The glass wall in the atrium is approximately 1 cm thick and is supported by the ceiling and wall. Figure 6-1 shows a picture of the setup used.

FIGURE 6-1: Picture of the acoustic tap tracker setup in the atrium of the MIT Media Lab. The effective area of enclosed by the sensors is approximately 35 inches by 47 inches. The computer was used for DAQ.

The signals received by the sensors when tapping on the glass wall of the atrium are very similar to those generated by the glass wall used in testing. Signals generated by a knuckle tap at location (18,77) are shown in Figure 6-2.

One significant difference between the signals generated using this pane of glass is that the

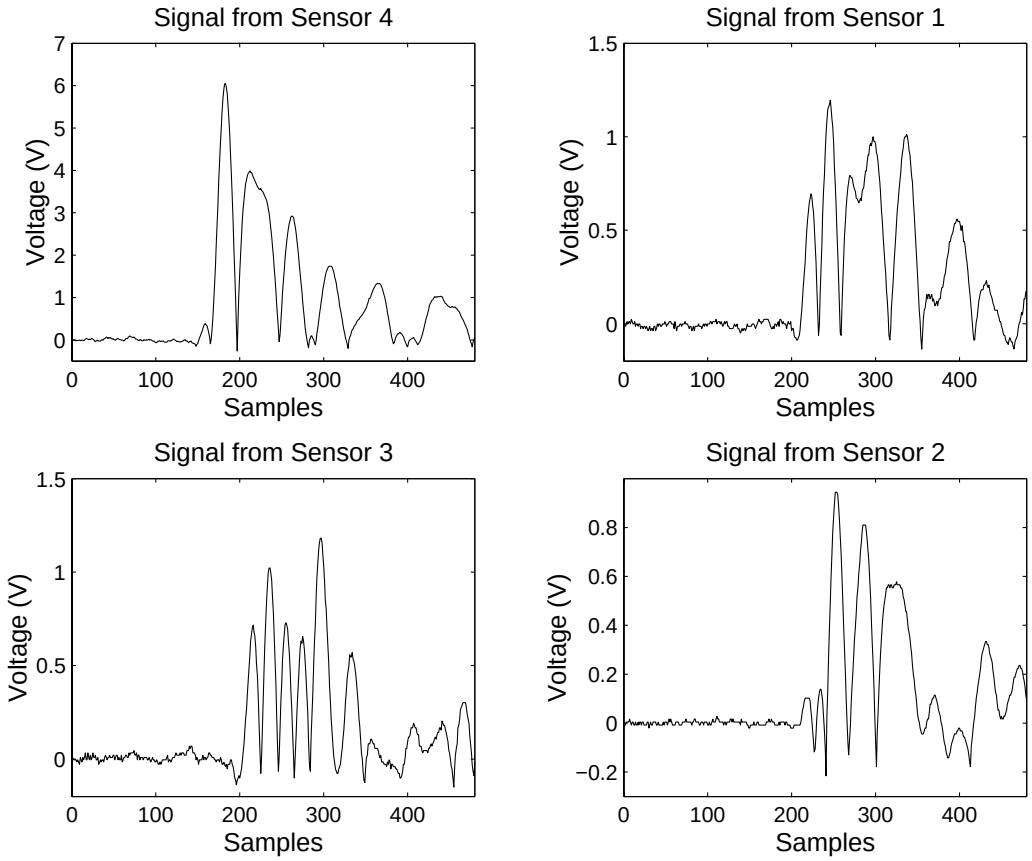
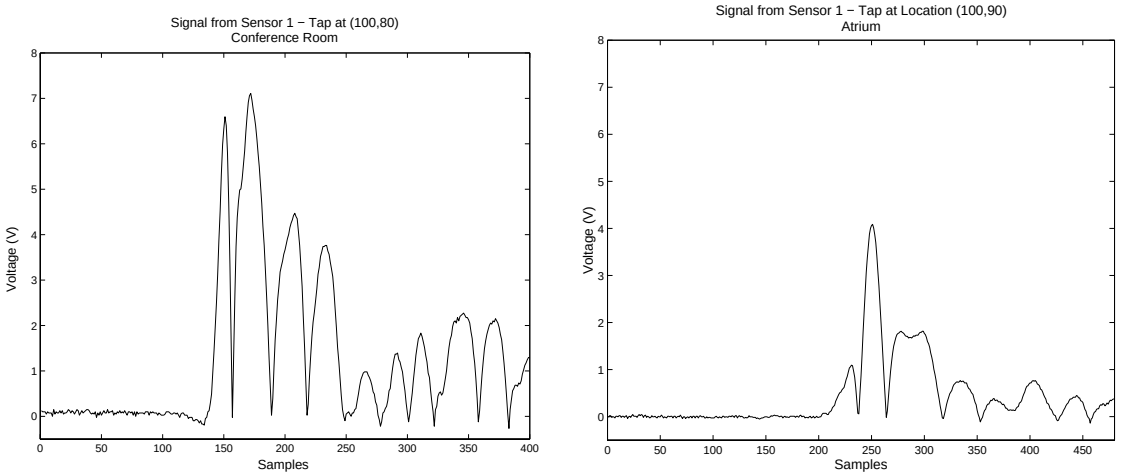


FIGURE 6-2: Waveforms generated by a tap at location (18,77) using the pane of glass in the atrium.

signals are significantly more attenuated due to the thickness of the glass. Figure 6-3(a) and Figure 6-3(b) compare signals generated by a knuckle tap at location (100,80) using the pane of glass in the conference room and the pane of glass in the atrium. Since the waves generated by knuckle taps are bending waves, they will change depending on the type of glass used.

Because of the variable amount of dispersion, time-of-arrival analysis would not work very well. The cross-correlations were examined to determine whether the algorithm used for position determination would still be accurate. A sample cross-correlation is shown in Figure 6-4.

The cross-correlation data shows large peaks; however, sometimes the cross-correlation data



(a) Waveform using the glass pane in the conference room.

(b) Waveform using the glass pane in the atrium.

FIGURE 6-3: Signals from sensor 1 generated by a tap at location (100,80) using two different panes of glass.

displays ambiguities in the largest peak as described in Chapter 5. However, the position determination algorithm accounts for this by running the algorithm on all combinations of peaks if an ambiguity exists.

All of the above analysis was on a surface approximately 35 inches by 47 inches. The size of the surface was increased to 55 inches by 60 inches to determine if the position determination would be accurate on a larger surface. Figure 6-5 shows waveforms from sensors 1 and 3 that are generated by a tap near sensor 1. These signals are not noticeably different than the signals on the smaller surface. The sensors were placed close to the edges of the glass to determine the effect of reflections. Any effect that the reflections cause is not noticeable in the signal.

Finally, metal taps were examined to determine if the pane of glass effected the signals generated. Again, no noticeable change was detected. This was expected since the mode of propagation of these signals is acoustic and should not depend on the pane of glass used. Figure 6-6 shows a sample waveform.

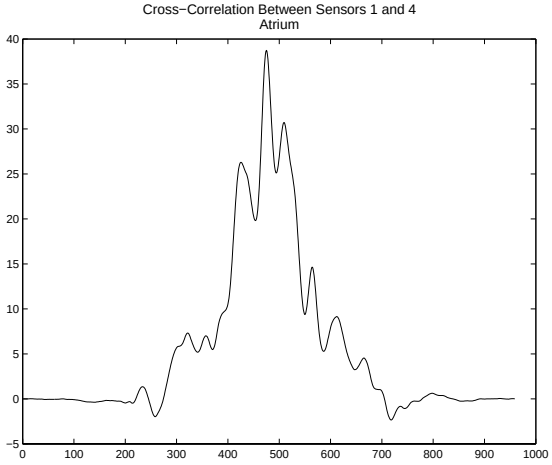


FIGURE 6-4: Cross-correlation between sensors 1 and 4. Generated by a tap at location (58,52).

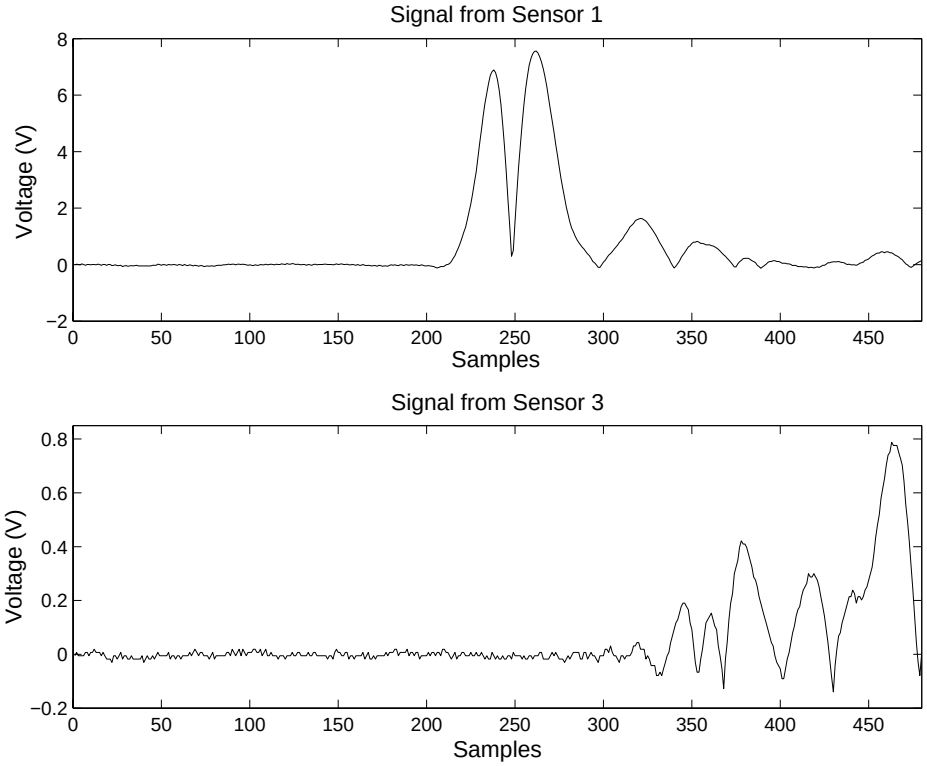


FIGURE 6-5: Signals from sensors 1 and 3. The size of the effective surface is 55 inches by 60 inches.

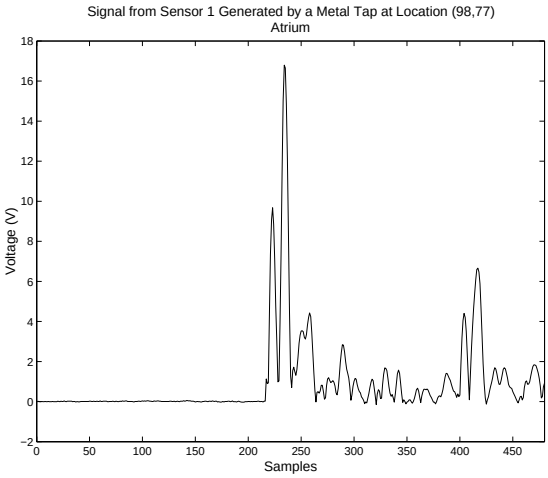


FIGURE 6-6: Signal from sensor 1 generated by a metal tap at location (98,77). The pane of glass is in the atrium.

Chapter 7

Conclusions and Future Research

The research presented in this thesis described the design of an acoustic tap tracker that is used to track impacts on large interactive surfaces. The system was portable, inexpensive, and simple to setup.

In this chapter, applications of the acoustic tap tracker are discussed. Possible improvements to the acoustic tap tracker are also proposed.

7.1 Applications

Natural applications for the acoustic tap tracker involve turning a large surface into an interactive surface to run "point and click" based applications. Tapping on a surface is much like the notion of a mouse click. In a computer-based application, the acoustic tap tracker cannot easily map the notion of dragging a mouse. A full system that includes the acoustic tap tracker that is able to implement these types of applications is described in the section.

The acoustic tap tracker was inspired by the Tangible Media Group's Ping-Pong Plus installation. Please refer to Chapter 2 for more information on this system. The acoustic tap tracker could be used as the front-end of the PP+ application in lieu of the electret

microphone based system.

Another interesting application is to use the system as part of an interactive store window. Shoppers can tap on the glass to find out more information about the products of the store as well as specials and sales. An application based on a similar idea was implemented.

This application was a showcase of the projects of the Responsive Environments Group of the MIT Media Lab. People who wanted to find out more about the projects of the group would simply walk up to the showcase wall and tap on the icon that represents the project of their choice. Figure 7-1 shows a user viewing the menu of projects [1].

FIGURE 7-1: Picture of the showcase wall demo for the acoustic tap tracker.

Currently, eight projects are being showcased. When a user taps on the icon, they are presented with some background information on the project as well as some pictures of the designs. Completed projects also include video of the project being demonstrated. If the user chooses to view more information on the acoustic tap tracker, he or she can choose to run a different tap tracker demo known as Magic Circles. In this demo, circles centered at

[1] To view a video of the system in action please refer to http://www.media.mit.edu/resenv/Tapper/

the user's tap are displayed (guesses from the other data groups are also displayed). The radius of the circle represents the uncertainty in the location of the tap as determined by the redundant sensor information. For instance, if the system is able to pinpoint the location of the tap to a region of approximately 5 cm in length and width, then a circle at the location of the tap with a diameter of 5 cm is displayed. Figure 7-2 shows a picture of the Magic Circles demo.

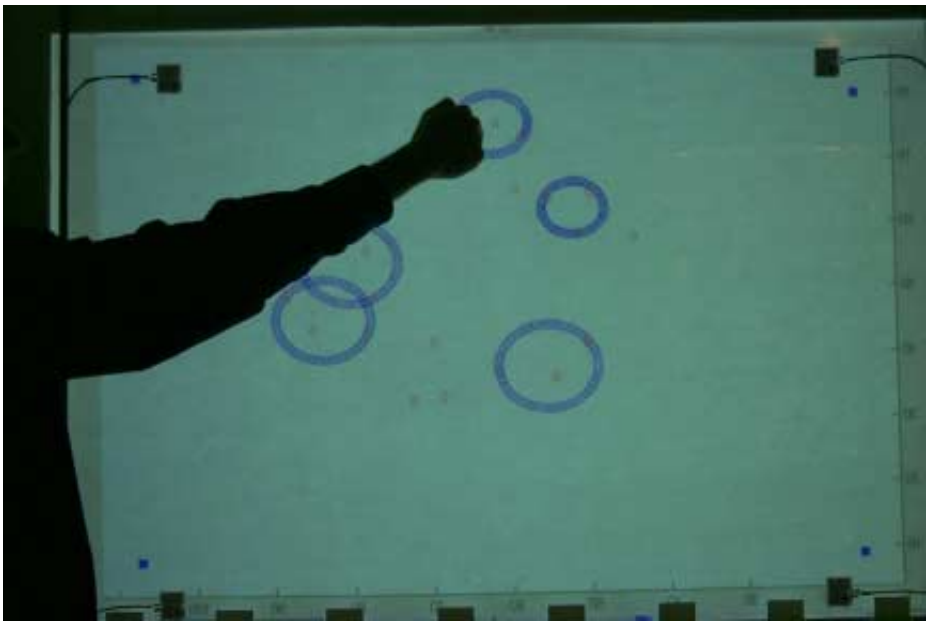


FIGURE 7-2: Picture of the Magic Circles demo. The circles drawn are centered at the tap location, and the radius represents the uncertainty region.

The user can also choose to exit the demo and return to the Showcase wall.

The system was implemented in C and Matlab. Matlab was used for position determination and wrote the guess for the position to a file that was read by a C program. The C program then generated a mouse event based on the tap. This mouse event was then used to control a series of web pages that showcased all the projects. Essentially, a normal web browser was executed, and a tap on the showcase wall was used to navigate instead of a mouse. Appendix D includes all of the C code required to implement the application.

The PC running the system crashed often if too many applications were running on it.

To reduce the number of applications, the system was networked using sockets. The web application is run on a different computer than the data acquisition and is connected to a projector that projects the image on the wall. The C program running on the DAQ PC sends the location of the tap to the receiving computer to generate the mouse event.

7.2 Future Applications

The human-computer interface (HCI) applications that are suitable for the acoustic tap tracker are limited to working with the notion of a mouse click. Another interactive surface known as the LaserWall[22], developed by the Responsive Environments Group, can be used in conjunction with the Acoustic Tap Tracker to enhance its functionality. The LaserWall uses an inexpensive scanning laser rangefinder to track bare hands near very large display surfaces.

A scanning laser rangefinder placed at the corner of the surface determines the coordinates of the hands above the surface [17]. Only one unit in the corner is required because it is able to scan the entire display surface by producing two coordinates simultaneously. Figure 7-3 shows a picture of the LaserWall system.

A system employing both the laser rangefinder and the acoustic tap tracker would be able to fully encompass all the actions of a touchscreen enabling any standard application to be developed. The tap tracker would be used to handle discrete events such as clicking while LaserWall would be used to handle continuous tracking events such as dragging.

In addition, the applications of a low-power radar are currently being investigated. One application would be to detect people as they approach the interactive surface causing the projected content to vary accordingly. This system would involve the use of a Saab tank-measuring radar [4].

In addition, the acoustic tap tracker can also be used with non-HCI applications. Depending on the type of knock, amplitude, frequency, and other characteristics of the waveform, the projected content of the system cab be varied.

FIGURE 7-3: LaserWall.

7.3 Future Research

The system was able to quite accurately track low-frequency knuckle taps; however, when hard impacts such as that generated by a metal tap were tracked, the system was not as accurate. The metal taps do not suffer from the problems associated with knuckle taps such as dispersion. In addition, the leading edge is sufficiently abrupt so timing analysis on the metal taps should be quite accurate. The problem with locating the hard impacts were of a completely different nature than the challenges of the knuckle tap. The maximum sampling rate of each channel is 48 kHz. The waves generated by the hard impact are acoustic and propagate at a speed of approximately 3500 m/s. For a surface that is approximately one meter long and a sampling rate of 48 kHz, this only allows for approximately 10 cm of resolution.

When the system was implemented using differential time-of-arrival analysis, it was observed that the system was not able to resolve sharp taps that were spaced less than 10 cm apart. One possible solution is to implement the leading edge analysis on a separate chip that is able to sample must faster such as a Senix micontroller [5]. The Senix would sample the

leading edge and return the arrival time of the four sensor waveforms. This information would then be sent to the existing system for position determination. Furthermore, when the system is ported to a DSP, the DSP could rapidly sample one of its digital input pins during the DAQ loop to determine the arrival time. An interrupt-driven solution could also be used where the number of transitions is counted using a timer interrupt. Another solution is to use the existing coarsely sampled waveforms and to try and fit some curve to the sampled points to extract the arrival time. Once a solution is implemented, the resolution of the hard impacts should be much smaller than that of knuckle taps.

Appendix A

Abbreviations and Symbols

PP+	Ping-Pong+
LED	Light-Emitting Diode
PVDF	Polyvinylidene Fluoride
DAQ	Data Acquisition
NI	National Instruments, Corp.
ADC	Analog to Digital Converter
SH	Super-H (Hitachi SH Microprocessor)
A/D	Analog to Digital
TI	Texas Instruments, Inc.
DSP	Digital Signal Processor
PCB	Printed Circuit Board
IR	Infrared
OP AMP	Operational Amplifier
HCI	Human-Computer Interface

Appendix B

Schematics and PCB Layouts

B.1 Pre-Amplifier Schematic

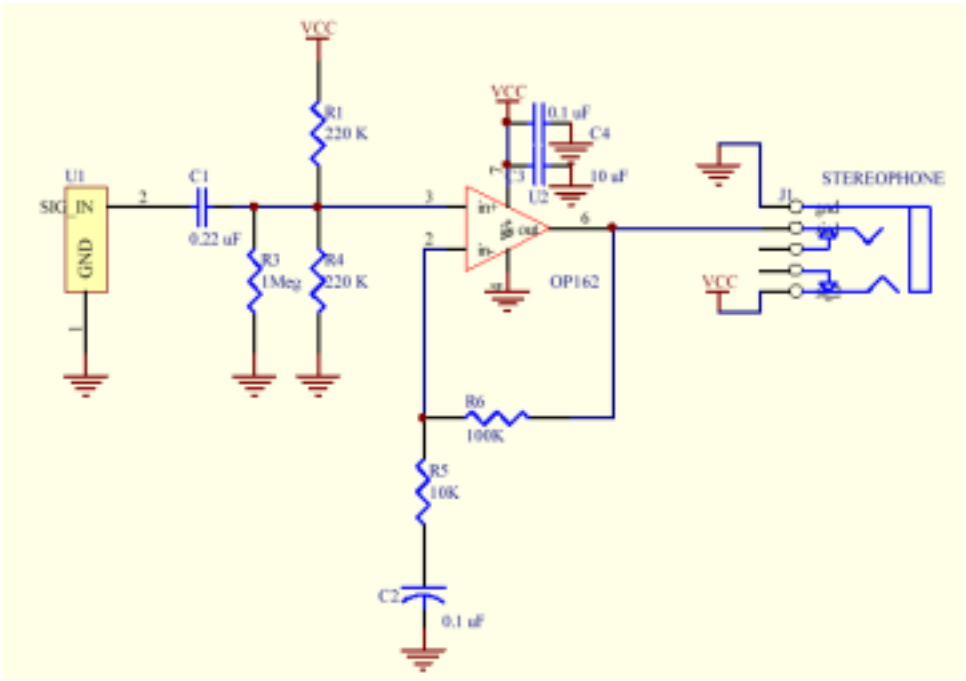


FIGURE B-1: Schematic of the pre-amplifier board.

B.2 Pre-Amplifier PCB Layout

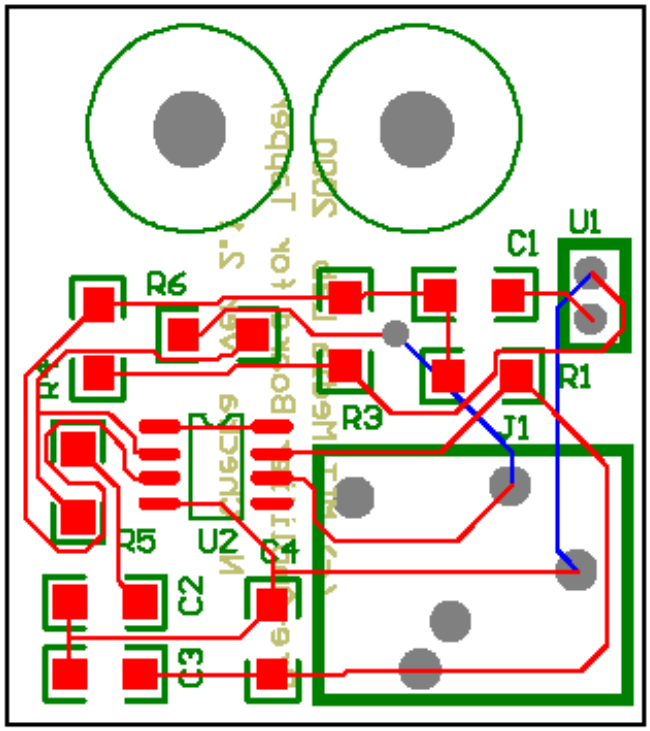


FIGURE B-2: PCB Layout of the pre-amplifier board (top and bottom layers).

B.3 Signal Conditioning Board Schematic

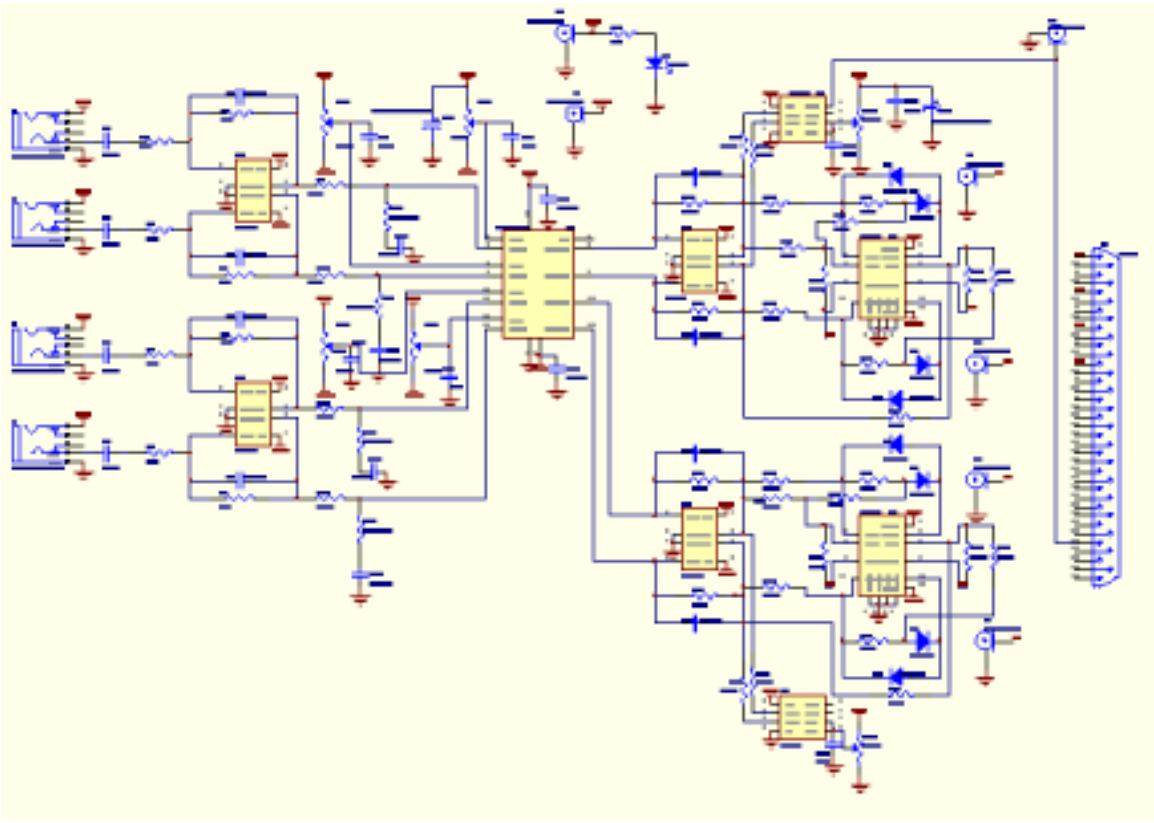


FIGURE B-3: Schematic of the signal conditioning board.

B.4 Signal Conditioning Board PCB Layout

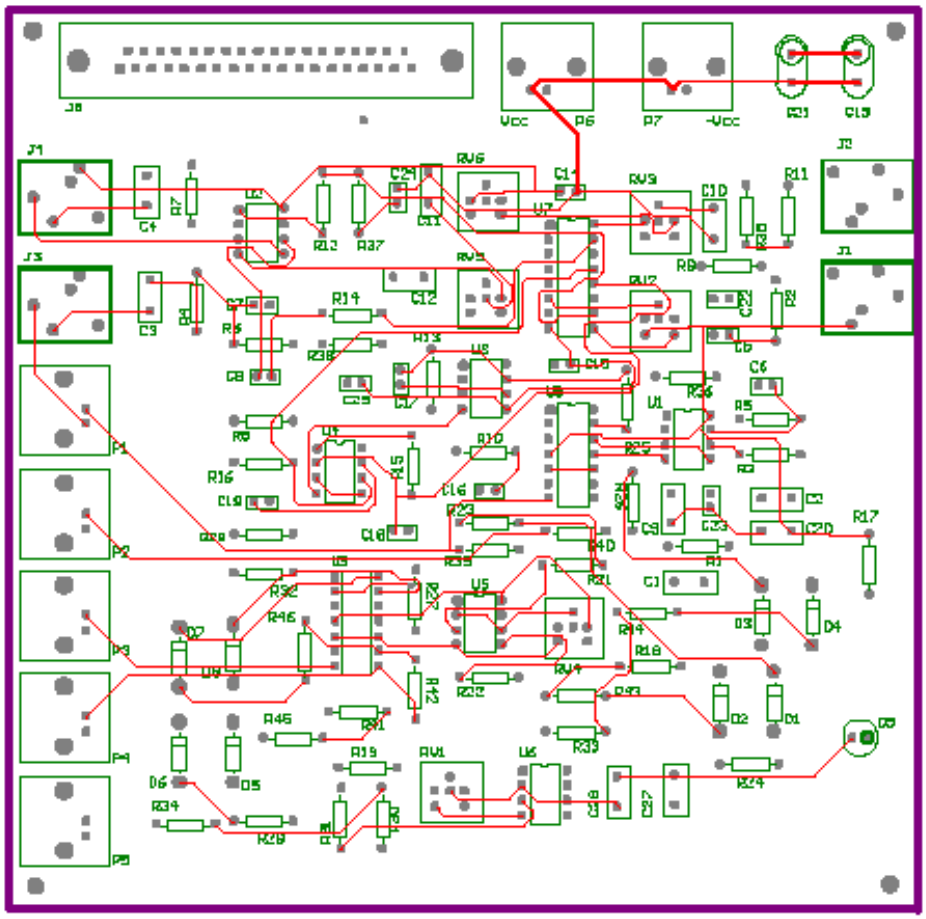


FIGURE B-4: PCB Layout of the signal conditioning board (top layer).

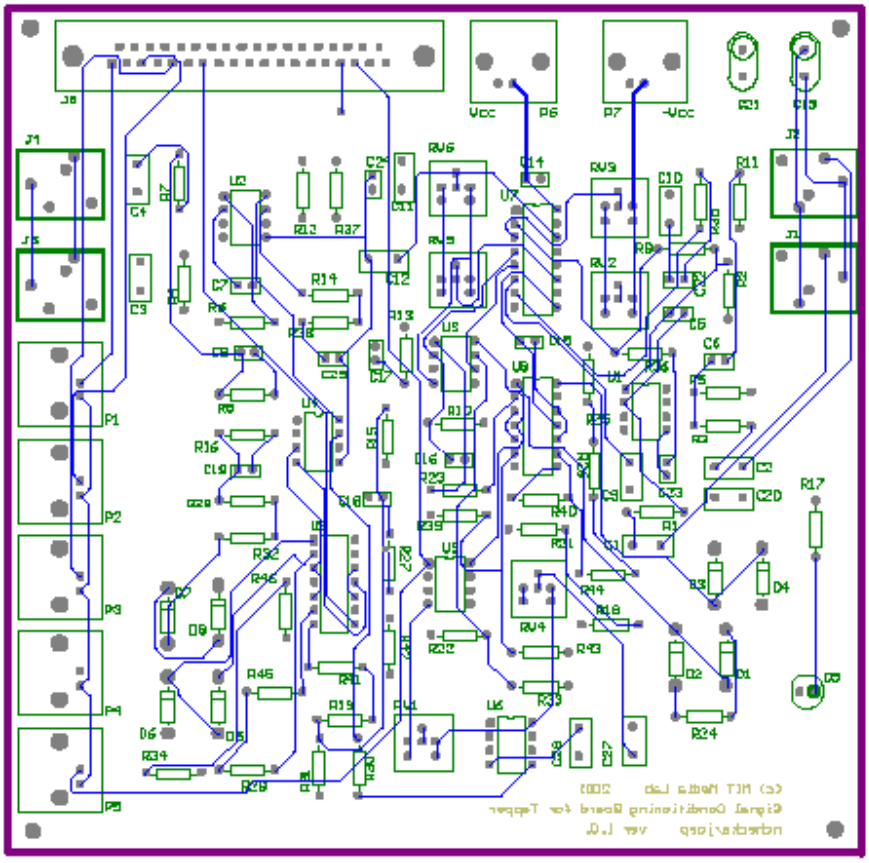


FIGURE B-5: PCB Layout of the signal conditioning board (bottom layer).

Appendix C

Matlab Code

C.1 calibrate.m

```
function = calibrate(trials)
% This is the calibration function for the acoustic tap tracker.
% The argument 'trials' is used to specify the number of trials per
% calibration point.  The function computes the coefficients of the
% model and write them to a file.

% set of 20 pre-defined calibration points
positions = [18 77
             38 77
             58 77
             78 77
             98 77
             18 59
             38 59
             58 59
             78 59
             98 59
             18 39
             38 39
             58 39
             78 39
             98 39
             18 22
             38 22
             58 22
             78 22
             98 22];

x = positions(:,1);
y = positions(:,2);

% variable declarations
numberOfPoints = length(x);
numberOfCoeffs = 6;
numberOfTrials = trials;
timeBin = 10;
amplitudeBin = 0.01;
times = [];
maxima = [];
meanRelativeTimes = [];
meanRelativeMaxima = [];
chan0data = [];
chan1data = [];
chan2data = [];
chan3data = [];

%Each combination of sensors has its own coefficients matrix.
cM123 = zeros(numberOfPoints, 20);
cM124 = zeros(numberOfPoints, 20);
cM134 = zeros(numberOfPoints, 20);
cM234 = zeros(numberOfPoints, 20);


col_counter=0;
for i=1:numberOfPoints,
   i
   times = [];
   col_counter = col_counter + 1;
   for trial=1:numberOfTrials,
      trial
      once  % take data from a tap
      samples = length(corr12);

      trial_locations(trial,:) = max_locations;

      % False Trigger.  Throw out data and retry.
      while ((max_locations(1) < 100) | (max_locations(2) < 100) |
             (max_locations(3) < 100) | (max_locations(4) < 100) |
             (max_locations(5) < 100) | (max_locations(6) < 100)),
```

```
            'Please Tap Again'
             once  % take data from a tap
             samples = length(tapperData);
        end
    end

    % for loop to calculate representative cross-correlation data
    for s=1:6,
        timeEdges =  -samples:timeBin:samples;
        timeSpread = histc(trial_locations(:,s),timeEdges);
        [count, index] = max(timeSpread);
        point_locations(s)=mean(trial_locations((trial_locations(:,s) >= timeEdges(index))
                                      & (trial_locations(:,s) <= timeEdges(index+1)),s));
    end

    t12 = point_locations(1);
    t13 = point_locations(2);
    t14 = point_locations(3);
    t23 = point_locations(4);
    t24 = point_locations(5);
    t34 = point_locations(6);

    % Calculate cross-correlation matrices
    cM123(i,:) = [t12^3, t13^3, t23^3, t12^2*t13, t12^2*t23, t13^2*t12, t13^2*t23,
                  t23^2*t12, t23^2*t13, t12^2, t13^2, t23^2, t12*t13, t12*t23,
                  t13*t23, t12, t13, t23, 1, 1];
    cM124(i,:) = [t12^3, t14^3, t24^3, t12^2*t14, t12^2*t24, t14^2*t12, t14^2*t24,
                  t24^2*t12, t24^2*t14,, t12^2, t14^2, t24^2, t12*t14, t12*t24,
                  t14*t24, t12, t14, t24, 1, 1];
    cM134(i,:) = [t13^3, t14^3, t34^3, t13^2*t14, t13^2*t34, t14^2*t13, t14^2*t34,
                  t34^2*t13, t34^2*t14, t13^2, t14^2, t34^2, t13*t14, t13*t34,
                  t14*t34, t13, t14, t34, 1, 1];
    cM234(i,:) = [t23^3, t24^3, t34^3, t23^2*t24, t23^2*t34, t24^2*t23, t24^2*t34,
                  t34^2*t23, t34^2*t24, t23^2, t24^2, t34^2, t23*t24, t23*t34,
                  t24*t34, t23, t24, t34, 1, 1];

end

% Find inverses of cross-correlation matrices.
cW123 = pinv(cM123);
cW124 = pinv(cM124);
cW134 = pinv(cM134);
cW234 = pinv(cM234);

% Find coefficients.
cCoeff123 = [cW123*x cW123*y]
cCoeff124 = [cW124*x cW124*y]
cCoeff134 = [cW134*x cW134*y]
cCoeff234 = [cW234*x cW234*y]

% Write to file.
dlmwrite('cCoeff123.dat',cCoeff123,'\t');
dlmwrite('cCoeff124.dat',cCoeff124,'\t');
dlmwrite('cCoeff134.dat',cCoeff134,'\t');
dlmwrite('cCoeff234.dat',cCoeff234,'\t');
```

C.2 tapper.m

```
% Forward determination algorithm for the acoustic tap tracker
ai=initializeDAQ();  % initialize data acquisition object

% Import coefficients matrices obtained from calibration (calibrate.m).
cCoeff123 = dlmread('cCoeff123.dat','\t');
cCoeff124 = dlmread('cCoeff124.dat','\t');
cCoeff134 = dlmread('cCoeff134.dat','\t');
cCoeff234 = dlmread('cCoeff234.dat','\t');

loop = 1;   offset = 0;   c=[];
while (loop)
   once  % take data

   % false trigger.  Throw out data and retry.
   while ((max_locations(1) < 100) | (max_locations(2) < 100) |
          (max_locations(3) < 100) | (max_locations(4) < 100) |
          (max_locations(5) < 100) | (max_locations(6) < 100)),
       'Please Tap Again'
       once  % take data from a tap
       samples = length(tapperData);
   end

   % extract coefficients
   xcCoeff123 = cCoeff123(:,1);
   ycCoeff123 = cCoeff123(:,2);
   xcCoeff124 = cCoeff124(:,1);
   ycCoeff124 = cCoeff124(:,2);
   xcCoeff134 = cCoeff134(:,1);
   ycCoeff134 = cCoeff134(:,2);
   xcCoeff234 = cCoeff234(:,1);
   ycCoeff234 = cCoeff234(:,2);

   for n=1:length(composite(1,:)),
       t12 = composite(1,n)';       t13 = composite(2,n)';
       t14 = composite(3,n)';       t23 = composite(4,n)';
       t24 = composite(5,n)';       t34 = composite(6,n)';

       % Calculate cross-correlation matrix
       cPoly123 = [t12^3, t13^3, t23^3, t12^2*t13, t12^2*t23, t13^2*t12, t13^2*t23,
                   t23^2*t12, t23^2*t13, t12^2, t13^2, t23^2, t12*t13, t12*t23, t13*t23,
                   t12, t13, t23, 1, 1];
       cPoly124 = [t12^3, t14^3, t24^3, t12^2*t14, t12^2*t24, t14^2*t12, t14^2*t24,
                   t24^2*t12, t24^2*t14, t12^2, t14^2, t24^2, t12*t14, t12*t24, t14*t24,
                   t12, t14, t24, 1, 1];
       cPoly134 = [t13^3, t14^3, t34^3, t13^2*t14, t13^2*t34, t14^2*t13, t14^2*t34,
                   t34^2*t13, t34^2*t14, t13^2, t14^2, t34^2, t13*t14, t13*t34, t14*t34,
                   t13, t14, t34, 1, 1];
       cPoly234 = [t23^3, t24^3, t34^3, t23^2*t24, t23^2*t34, t24^2*t23, t24^2*t34,
                   t34^2*t23, t34^2*t24, t23^2, t24^2, t34^2, t23*t24, t23*t34, t24*t34,
                   t23, t24, t34, 1, 1];

       % calculate position
       c(n+offset,1) = cPoly234*xcCoeff234;
       c(n+offset,2) = cPoly234*ycCoeff234;
       c(n+1+offset,1) = cPoly134*xcCoeff134;
       c(n+1+offset,2) = cPoly134*ycCoeff134;
       c(n+2+offset,1) = cPoly124*xcCoeff124;
       c(n+2+offset,2) = cPoly124*ycCoeff124;
       c(n+3+offset,1) = cPoly123*xcCoeff123;
       c(n+3+offset,2) = cPoly123*ycCoeff123;
       offset = offset+3;
   end

   pinpoint     % Call pinpoint function to eliminate deviant guesses
   a=[sx sy];   % a contains the guess for the location
end
```

C.3 finalize.m

```
% Function used to eliminate deviant guesses
% initialize variables
goodx = [];  goody = [];
finalx = [];  finaly = [];  inty = [];

xloc = abs(c(:,1));     yloc = abs(c(:,2));
xpos = [];  ypos = [];  counter=1;

for i=1:length(xloc),
    if ((xloc(i)< 120)),
        xpos(counter) = xloc(i);
        counter=counter+1;
    end
end

counter=1;   avgx=mean(xpos);
for i=1:length(xpos),
    if( ((xpos(i)/avgx) <= 1.5) & ((xpos(i)/avgx) >= .5) ),
        goodx(counter) = xpos(i);
        counter=counter+1;
    end
end

counter=1;
for i=1:length(goodx),
    if( abs(xpos(i)-avgx) < 10),
        finalx(counter) = goodx(i);
        counter=counter+1;
    end
end

counter=1;
for i=1:length(yloc),
    if ((yloc(i)< 95) & (yloc(i) > 0)),
        ypos(counter) = yloc(i);
        counter=counter+1;
    end
end

counter=1;   avgy=mean(ypos);
for i=1:length(ypos),
    if(abs(ypos(i)-avgy) < 21),
        goody(counter) = ypos(i);
        counter=counter+1;
    end
end

counter = 1;
for i=1:length(goody),
    if( abs(ypos(i)-mean(goody)) < 10),
        inty(counter) = goody(i);
        counter=counter+1;
    end
end

counter = 1;
for i=1:length(goody),
    if( abs(ypos(i)-mean(inty)) < 10),
        finaly(counter) = goody(i);
        counter=counter+1;
    end
end
```

C.4 initializeDAQ.m

```
function ai = initializeDAQ()
% function used to initialize the data acquisition hardware.

% Constants.
sampleRate = 48000;
samples = floor(sampleRate/100);
pretrigger = floor(0.45*samples);

% Check for hardware and adaptors.
hwinfo = daqhwinfo;
InstalledAdaptors = hwinfo.InstalledAdaptors;
hwinfo = daqhwinfo('nidaq');
ID = 1;

% Initializing data acquisition object.
ai = analoginput('nidaq', ID);
triggerChannel = addchannel(ai, 0, 'mic1');
addchannel(ai, 1, 'mic2');
addchannel(ai, 2, 'mic3');
addchannel(ai, 3, 'mic4');

set(ai, 'SampleRate', sampleRate);
set(ai, 'SamplesPerTrigger', samples);
set(ai, 'TriggerChannel', triggerChannel);
set(ai, 'TriggerType', 'Software');
set(ai, 'TriggerCondition', 'Rising');
set(ai, 'TriggerConditionValue', 0.1);
set(ai, 'TriggerDelayUnits', 'Samples');
set(ai, 'TriggerDelay', -pretrigger);
hwinfo = daqhwinfo(ai);
```

C.5 once.m

```
% Function used to take data from the data acquisition hardware.  It calls takeOneTap to
% take the data and calculates the cross-correlations between all the signals

% take data
tapperData = takeOneTap(ai);
data0 = tapperData(:,1);
data1 = tapperData(:,2);
data2 = tapperData(:,3);
data3 = tapperData(:,4);

% Software Gain
gain = 5;
gainData = tapperData;
gainData=gain*tapperData;

corr12=xcorr(gainData(:,1),gainData(:,2));
corr13=xcorr(gainData(:,1),gainData(:,3));
corr14=xcorr(gainData(:,1),gainData(:,4));
corr23=xcorr(gainData(:,2),gainData(:,3));
corr24=xcorr(gainData(:,2),gainData(:,4));
corr34=xcorr(gainData(:,3),gainData(:,4));

[fourpts12, weighted12] = determinePeaks(corr12);
[fourpts13, weighted13] = determinePeaks(corr13);
[fourpts14, weighted14] = determinePeaks(corr14);
[fourpts23, weighted23] = determinePeaks(corr23);
[fourpts24, weighted24] = determinePeaks(corr24);
[fourpts34, weighted24] = determinePeaks(corr34);

prloc12 = peakRatio(corr12,fourpts12);
prloc13 = peakRatio(corr13,fourpts13);
prloc14 = peakRatio(corr14,fourpts14);
prloc23 = peakRatio(corr23,fourpts23);
prloc24 = peakRatio(corr24,fourpts24);
prloc34 = peakRatio(corr34,fourpts34);

len12 = length(prloc12);
len13 = length(prloc13);
len14 = length(prloc14);
len23 = length(prloc23);
len24 = length(prloc24);
len34 = length(prloc34);

composite = [];

% determine all combinations of peaks
composite=combvec(prloc12,combvec(prloc13,combvec(prloc14,combvec(prloc23,
          combvec(prloc24,prloc34)))))

max_locations = [loc12 loc13 loc14 loc23 loc24 loc34];
```

C.6 pinpoint.m

```
% Function used to eliminate deviant points

% Calibration points
positions = [18 77
             38 77
             58 77
             78 77
             98 77
             18 59
             38 59
             58 59
             78 59
             98 59
             18 39
             38 39
             58 39
             78 39
             98 39
             18 22
             38 22
             58 22
             78 22
             98 22];

% Cross-correlation data of calibration points
orderpts = [468.1667  522.0000  565.3333  535.6667  577.1667  521.8333;
            481.6000  502.0000  597.8000  447.2500  525.5000  539.8333;
            594.3333  476.1667  597.1667  328.6667  484.6667  568.0000;
            551.5000  491.0000  540.5000  397.3333  449.5000  529.8000;
            511.1667  457.5000  467.5000  411.6000  414.1667  482.8333;
            487.5000  539.3333  388.0000  531.0000  381.0000  447.4000;
            485.8333  433.0000  512.6000  458.0000  484.0000  507.8000;
            486.2000  489.1667  498.8000  371.8000  487.6667  485.2000;
            478.7500  467.5000  447.2000  441.8333  452.6000  488.8333;
            520.0000  463.0000  469.6000  415.5000  421.0000  483.6667;
            472.0000  483.3333  510.0000  561.4000  368.8333  462.0000;
            469.3333  501.0000  509.3333  381.8000  393.7500  490.3333;
            467.1667  487.0000  431.0000  476.8000  431.0000  494.2000;
            316.1667  453.1667  440.3333  587.6667  424.8333  468.5000;
            475.1667  420.3333  431.6667  589.0000  434.5000  474.0000;
            457.3333  579.1667  544.3333  599.3333  561.3333  444.1667;
            407.5000  528.0000  480.0000  599.5000  472.0000  432.0000;
            371.6667  493.0000  277.3333  600.0000  454.6667  265.2500;
            326.8000  453.8333  312.3333  573.8333  466.0000  471.5000;
            318.8000  416.0000  413.8000  476.6667  472.6667  479.2000];

diffmatrix = zeros(20,6);
dummy = zeros(20,6);
maxgood = [];
locmax=[];

for n=1:length(composite(1,:)),
    for i=1:20,
        dummy(i,:) = composite(:,n)';
    end

    diffmatrix = abs(orderpts-dummy);

    for i=1:20,
        deviations(i) = sum(diffmatrix(i,:));
    end
    [mindev, locmin] = min(deviations);

    devnum = zeros(20,1);

    for i=1:20,
```

```
        for j=1:6,
            if (diffmatrix(i,j) < 10),
                devnum(i) = devnum(i) + 1;
            end
        end
    end

    [m, v]=max(devnum);
    maxgood(n) = m;
    locmax(n) = v;
end

listmode = [];
listmode(1) = locmax(1);   countmode = zeros(1,10);   match = 0;

comparex = positions(locmin,1);  comparey = positions(locmin,2);

locmax = [locmax locmin]
for i=2:length(locmax),
    for j=1:length(listmode),
        if (match == 0),
            if (locmax(i) == listmode(j)),
                countmode(j) = countmode(j) + 1;
                match = 1;
            end
        end
    end
    if ((match == 0) & (j==length(listmode))),
        listmode(length(listmode)+1) = locmax(i);
    end
    match = 0;
end

[pickmax,location] = max(countmode);  num=0;  counter=1;
for i=1:length(countmode),
    if (countmode(i) == pickmax),
        num = num + 1;
        remember(counter) = listmode(i);
        counter=counter+1;
    end
end

dist = [];
if (num > 1),
    finalize
    for i=1:length(remember),
        if (isempty(finalx) == 1),
            dist(i) = abs(mean(finaly)-positions(remember(i),2));
        else dist(i) = abs(mean(finalx)-positions(remember(i),1));
        end
    end

    [ambmin,ambloc]=min(dist);
    calibx = positions(remember(ambloc),1);
    caliby = positions(remember(ambloc),2);
else
    [pickmax,location] = max(countmode);
    pickloc = listmode(location);
    calibx = positions(pickloc,1);
    caliby = positions(pickloc,2);
end

if ((pickmax == 0) | length(locmax) == 1),
    calibx = comparex;
    caliby = comparey;
end

xloc = abs(c(:,1));  yloc = abs(c(:,2));
```

```
xpin = [];  ypin = [];  counter=1;

for i=1:length(xloc),
    if (abs(xloc(i) - calibx) < 15)
        xpin(counter) = xloc(i);
        counter=counter+1;
    end
end

counter=1;
for i=1:length(yloc),
    if (abs(yloc(i) - caliby) < 15)
        ypin(counter) = yloc(i);
        counter=counter+1;
    end
end

% code to translate point into coordinates for mouse_event
sx=.01*(bx-26)+.2;
sy=-(by-27)/73.33 + .8;
```

C.7 takeOneTap.m

```
function tapperData = takeOneTap(ai)
% TAKEONETAP Rudimentary data taking subroutine.
% Returns a matrix of data -- columns are data channels and rows are
% data samples on each channel.  Instantiates an analog input object
% and waits for a trigger.  Returns the data taken from that event
% and then deletes the analog input object.

start(ai);
'Begin tapping now'
samples = get(ai, 'SamplesPerTrigger');
loop = 1;
while (loop)
   if (get(ai, 'SamplesAvailable') >= samples)
      tapperData = getdata(ai);
      stop(ai);
      loop = 0;
   end
end
```

C.8 determinePeaks.m

```
function [fourpts, weighted] = determinePeaks(data)
% function used to determine the local maxima of the cross-correlation data

dprime = diff(data);
dprime_length = length(dprime)-1;
counter = 1;
threshold = 0.2;
firstTime = 0;
inflecpts = [];

for i=1:dprime_length,
    if ((dprime(i) >= 0) & (dprime(i+1) < 0))
        inflecpts(counter) = i+1;
        counter = counter + 1;
    end
end

inflecpts_length = length(inflecpts);
inflecpts;
max1=1;   max2=1;   max3=1;   max4=1;
index1=1;   index2=1;   index3=1;   index4=1;

for i=1:inflecpts_length,
    if (data(inflecpts(i)) > max1),
        max4 = max3;        index4 = index3;
        max3 = max2;        index3 = index2;
        max2 = max1;        index2 = index1;
        max1 = data(inflecpts(i));
        index1 = inflecpts(i);
    elseif (data(inflecpts(i)) > max2),
        max4 = max3;
        index4 = index3;
        max3 = max2;
        index3 = index2;
        max2 = data(inflecpts(i));
        index2 = inflecpts(i);
    elseif (data(inflecpts(i)) > max3),
        max4 = max3;
        index4 = index3;
        max3 = data(inflecpts(i));
        index3 = inflecpts(i);
    elseif (data(inflecpts(i)) > max4),
        max4 = data(inflecpts(i));
        index4 = inflecpts(i);

    end
end

w1 = max1*index1;   w2 = max2*index2;
w3 = max3*index3;   w4 = max4*index4;

w=[w1 w2 w3 w4];
sum_max = max1 + max2 + max3;

mean(w);
point = mean(w)/sum_max;
weighted = (w1+w2+w3) / (sum_max);
fourpts = [index1 index2 index3 index4];
fourmax = [max1 max2 max3 max4];
```

C.9 peakRatio.m

```
function loc = peakRatio(signal,fourpts)
% Function used to calculate the ratio between the two largest peaks of the
% cross-correlation

    ratio = signal(fourpts(1))/signal(fourpts(2));
    if (ratio < 1.05),
        loc = [fourpts(1) fourpts(2)];
    else loc = [fourpts(1)];
    end
```

Appendix D

C Code

D.1 client.cpp

```
// Client.cpp (for TCP)
// Compile and link with wsock32.lib
// Usage: Client ServerName PortNumber

#include <stdio.h>
#include <string.h>
#include <time.h>
#include <sys/timeb.h>
#include <winsock.h>
#include <fstream.h>

// Function prototype
void StreamClient(char *szServer, short nPort, char *szBuf);

// Helper macro for displaying errors
#define PRINTERROR(s)  \
               fprintf(stderr,"\n%: %d\n", s, WSAGetLastError())

void main(int argc, char **argv)
{
       WORD wVersionRequested = MAKEWORD(1,1);
       WSADATA wsaData;
       int nRet;
       short nPort;
       // Check for the host and port arguments
       if (argc != 3) {
              fprintf(stderr,"\nSyntax: Client ServerName PortNumber \n");
              return;
       }
       // get port no
       nPort = atoi(argv[2]);
       // Initialize WinSock and check the version
       nRet = WSAStartup(wVersionRequested, &wsaData);
       if (wsaData.wVersion != wVersionRequested) {
              fprintf(stderr,"\n Wrong version\n");
              return;
       }

       while(1) {
              double x=0;
              double y=0;

           // read coordinates of point from file
              ifstream chan0stream("point.dat",ios::in);

              if (!chan0stream) {
                     cerr << "File could not be opened" <<endl;
                     exit(1);
              }

              bool resume = 0;
              bool invalid = 0;
              if (chan0stream >> x >> y)
                     resume = 1;
              else resume = 0;

              fcloseall;
              if (resume == 1) {
                     ofstream outFile("nisha.dat",ios::trunc);
                     outFile.close();
                     cout <<"erased" <<endl;
                     cout <<"(x,y) = " <<x <<" " <<y <<endl;

                     char xpos[3];
                     char ypos[3];
                     char buf[9];
```

```
                        char temp[] ="p ";
                        char space[] = " ";
                        char end[] ="\n";

                        if (x < .1) {
                                x= .1;
                        } else invalid = 0;

                        if (!invalid) {
                                strcpy(buf,temp);
                                gcvt(x,2,xpos);
                                strcat(buf,xpos);
                                gcvt(y,2,ypos);
                                strcat(buf," ");
                                strcat(buf,ypos);
                                strcat(buf,"\n");

                                cout <<"buf is " <<buf <<endl;


                                // file transfer
                                StreamClient(argv[1], nPort, buf);
                        }
                } // end if(resume)
        } // end while

        // Release WinSock
        WSACleanup();
} // end main

void StreamClient(char *szServer, short nPort, char *szBuf)
{

        printf("Client connecting to server: \n%s on port: %d",
                                szServer, nPort);

        // Find the server
    LPHOSTENT lpHostEntry;

        lpHostEntry = gethostbyname(szServer);
    if (lpHostEntry == NULL) {
         PRINTERROR("gethostbyname()");
         return;
    }

        // Create a TCP/IP stream socket
        SOCKET  theSocket;

        theSocket = socket(AF_INET,                                     // Address family
                                        SOCK_STREAM,                    // Socket type
                                        IPPROTO_TCP);                   // Protocol
        if (theSocket == INVALID_SOCKET) {
                PRINTERROR("socket()");
                return;
        }

        // Fill in the address structure
        SOCKADDR_IN saServer;

        saServer.sin_family = AF_INET;
        saServer.sin_addr = *((LPIN_ADDR)*lpHostEntry->h_addr_list);
                                                                        // Server's address
        saServer.sin_port = htons(nPort);               // Port number from command line

        // connect to the server
        int nRet;

        nRet = connect(theSocket,                               // Socket
                       (LPSOCKADDR)&saServer,                   // Server address
```

```
                    sizeof(struct sockaddr));  // Length of server address structure

        printf("nRet :%i",nRet);
        if (nRet == SOCKET_ERROR)
        {
               printf ("ERROR");
               PRINTERROR("socket()");
               closesocket(theSocket);
               return;
        }

        long totbyte=0;
        //char szBuf[] = "p 0 0\n";
        int c, i;
        printf ("\nSending data: %s\n Length: %i\n",szBuf,strlen(szBuf));

        nRet = send(theSocket,                        // Connected socket
                    szBuf,                           // Data buffer
                    strlen(szBuf),                   // Length of data
                    0);
        szBuf = "l 1\n";
        nRet = send(theSocket,                        // Connected socket
                     szBuf,                           // Data buffer
                     strlen(szBuf),                   // Length of data
                     0);

        if (nRet == SOCKET_ERROR)
        {
               printf ("ERROR");
               PRINTERROR("send()");
               closesocket(theSocket);
               return;
        }

        closesocket(theSocket);
        return;
}
```

Bibliography

[1] For more information on Intrepid touchscreens, refer to http://www.intrepidtouch.com.

[2] For more information on the Initial Prototype of the Acoustic Tap Tracker, refer to http://www.media.mit.edu/ lifton/proj/tapper.

[3] For more information on the PCI-6024E Data Acquisition Board, refer to http://www.ni.com.

[4] For more information on the Saab Tank Radar, refer to http://www.saabradar.com/html/pro.html.

[5] For more information on the Senix Microcontroller, refer to http://www.sxlist.com/techref/ubicom/index.htm.

[6] For more information on the SH1WH embedded controller and peripherals, refer to http://web.mit.edu/bunnie/www/proj/sh1wh/sh1whquick.html.

[7] For more information on the SH7032, refer to http://www.hitachi.co.jp/sicd/english/products/micom_all/l016e.htm.

[8] http://www.kettering.edu/ drussell/demos/dispersion/flexural.html.

[9] Measurement Specialities (i)nc., refer to http://www.msiusa.com/piezo/vibration_dynamic_sensors.htm.

[10] Y.T. Chan and K.C. Ho. A Simple and Efficient Estimator for Hyperbolic Location. *IEEE Transactions on Signal Processing*, 42(8):1905–15, 1994.

[11] N. Gershenfeld. *The Nature of Mathematical Modeling.* Cambridge University Press, 1999.

[12] H. Ishii and B. Ullmer. Tangible Bits: Towards Seamless Interfaces Between People, Bits, and Atoms. In *Proceedings of the 1991 Conference on Human Factors in Computing Systems*, 1997.

[13] H. Ishii, C. Wisneski, J. Orbanes, B. Chun, and J. Paradiso. PingPongPlus: Design of an Athletic-Tangible Interface for Computer-Supported Cooperative Play. In *Proceedings of CHI 1999*, 1999.

[14] D.R. Lide. *CRC Handbook of Chemistry and Physics*. CRC Press, 1999.

[15] O. Omojola, E.R. Post, M.D. Hancher, Y. Maguire, R. Pappu, B. Schoner, P.R. Russo, R. Fletcher, and N. Gershenfeld. An Installation of Interactive Furniture. *IBM Systems Journal*, 39(3&4):861–879, 2000.

[16] J.A. Paradiso. The Interactive Balloon: Sensing, Actuation, and Behavior in a Common Object. *IBM Systems Journal*, 35(3&4):473–487, 1996.

[17] J.A. Paradiso, K. Hsiao, J. Strickon, J. Lifton, and A. Adler. Sensor Systems for Interactive Surfaces. *IBM Systems Journal*, 39(3&4):892–914, 2000.

[18] G.M. Sessler. Piezoelectricity in Polyvinylidenefluoride. *Journal of the Acoustical Society of America*, 70(6):1596–1668, 1981.

[19] J.R. Smith, T. White, C. Dodge, J. Paradiso, and N. Gershenfeld. Electric Field Sensing for Graphical Interfaces. *IEEE Computer Graphics and Applications*, 18(3):54–60, 1998.

[20] B.T. Turko and R.C. Smith. A Precision Timing Discriminator for High Density Detector Systems. *IEEE Transactions on Nuclear Science*, 39(5):1311–15, 1992.

[21] C. Wisneski, J. Orbanes, and H. Ishii. PingPongPlus: Augmentation and Transformation of Athletic Interpersonal Interaction. In *Proceedings of CHI 1998*, 1998.

[22] C.T. Yang. New Laser Range Finder Designs for Tracking Hands atop Large Interactive Surfaces. M.Eng thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2001.