Source localization on solids for touch interfaces

Yap, Xuexin

2011

Yap, X. (2011). Source localization on solids for touch interfaces. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/50588>

<https://doi.org/10.32657/10356/50588>

NANYANG
TECHNOLOGICAL
UNIVERSITY

Source Localization on Solids for Touch Interfaces

YAP XUEXIN

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement of the degree of
Master of Engineering

2011

ABSTRACT

Research into human computer interface (HCI) has been very active in recent years due to the advances in software applications. Such devices are aimed at providing a more natural interface for which humans to interact with machines. In this research, we propose a new approach to the development of a touch interface through the use of a surface-mounted sensor which allows one to convert hard surfaces into touch pads. We first develop, using mechanical vibration theories, a mathematical model that simulates the output signals derived from sensors mounted on a physical surface. Utilizing this model, we determine the hardware required for this research. We then develop a source localization algorithm based on an all-pole filter model for location template matching that extracts the dominant frequencies of the tap. The performance of the proposed algorithm is compared with existing approaches and verified both in a synthetic as well as a real environment for the localization of a finger tap on a touch interface. In addition, we explore the time-difference-of-arrival based methods for source localization and implement a real-time wireless source localization prototype.

ACKNOWLEDGEMENT

I would like to acknowledge and extend my heartfelt gratitude to the persons who have made assisted and guided me thus far in this research, in particular, the members of the Speech, Touch and Acoustic Tangible Interfaces for Next-generation Applications (STATINA), for their continual support, guidance and encouragement, (in alphabetical order) Amir bin Sulaiman, Li Renshi, Liao Lei, Liu Benxu, Liu Di, Rajan Sobhana Rashobh, Ramachandran Bremananth, Vaninirappuputhenpurayil Gopalan Reju, Wang Liang, the members of the Seismic Team, Divya Venkatraman and Vinod Veera Reddy, and especially Prof. Andy Khong, for his endless patience, hardwork, and guidance. Also National Research Foundation without which this work could not have been materialized.

Special thanks to Kattukandy Rajan Arun for his patience, encouragement and advise without which some of the works found in this thesis may never have come to pass.

I would also like to thank my family for their patience and support towards me during this period as I labored through this degree. I would like to make acknowledgement to my fiancée, for her patience and labor of love towards me.

Most importantly, I would like to thank our Lord Jesus Christ for His love to me all my life, and for blessing me with a wonderful family, a great bunch of colleagues in the research teams, a loving fiancée, and a wonderful plan for my life.

CONTENTS

<i>List of Symbols</i>	12
<i>List of Abbreviations</i>	13
<i>1. Introduction</i>	14
<i>2. Hardware development for data acquisition</i>	18
2.1 Vibration Sensor	18
2.1.1 The Sensors	19
2.1.2 Testing the Sensors	21
2.2 Mechanical Model	24
2.2.1 Mathematical model of a rectangular plate	26
2.2.2 Mathematical model of accelerometer	32
2.2.3 Model between impact and sensor output	33
2.2.4 Illustrative example using the mathematical model	34
2.3 Direct Injection (DI) Box	36
2.4 Analog-Digital Converter (ADC)	42
2.4.1 Synchronization of the ADCs	44
2.5 Conclusion	47
<i>3. Template Matching</i>	48
3.1 Image Template Matching	48
3.2 Audio Template Matching	49
3.3 Location Template Matching	51
3.4 Proposed all-pole model based algorithm for LTM	54
3.4.1 Complexity analysis	57
3.4.2 Experimental setup and performance measure	58
3.4.3 Experimental results of CC-LTM and AP-LTM	61
3.5 Conclusion	63

4. <i>Localization Using Time-Differences-of-Arrival (TDOA)</i>	65
4.1 Review of the TDOA algorithm	65
4.2 TDOA Estimation	66
4.3 Estimating Source Coordinates	69
4.3.1 Formulation of Cost Function	70
4.3.2 Levenberg-Marquardt Algorithm	71
4.3.3 Generating the Jacobian Matrix	72
4.4 Implementation of a Real-time TDOA Algorithm Robust to Dis- persion	73
4.5 Conclusion	75
5. <i>Implementation of the Wireless Touch Interface Prototype</i>	77
5.1 Wireless systems	77
5.1.1 Reduction of the Synchronization Error	78
5.1.2 Tolerance in Time-Synchronization	80
5.1.3 Other Parameters to Consider When Choosing a Transmis- sion Protocol	82
5.1.4 The IEEE 802.15.4 protocol	84
5.1.5 Proposed Method	85
5.2 Wireless Prototype	87
5.2.1 Flowchart of the Coordinator Node	87
5.2.2 Flowchart of the End-Device Node	92
5.3 Proposed weighted average algorithm	95
5.3.1 Proposed algorithm to Determine the Source to Sensors Proximity	96
5.3.2 Experimental Results Using the Proposed Algorithm . . .	100
5.4 Conclusion	104
6. <i>Summary and Future Works</i>	107
6.1 Summary	107
6.2 Future Works	109
6.3 Publications Arising from this Thesis	111
<i>Appendix</i>	121
A. <i>Flowcharts of the Functions Used in the Wireless Prototype</i>	122

<i>B. UART Protocol for PC and SHARC (DSP)</i>	138
--	-----

LIST OF FIGURES

2.1	Hardware block diagram.	18
2.2	The Analog Devices ADXL001 [after [1]].	20
2.3	A Murata PKS1-4A1 accelerometer.	20
2.4	The Knowles BU21771 mounted on aluminium with its wires held down using Blu-Tack.	21
2.5	Time-amplitude response of an Analog Devices ADXL001 to a tap made on a piece of acrylic 28.28 cm from it.	22
2.6	Time-amplitude response of a Murata PKS1-4A1 to a tap made on a piece of acrylic 28.28 cm from it.	22
2.7	Time-amplitude response of a Knowles BU-21771 to a tap made on a piece of acrylic 28.28 cm from it.	23
2.8	Close up view of the response of an Analog Devices ADXL001 to a tap made on a piece of acrylic 28.28 cm from it.	23
2.9	Magnitude response of a Analog Devices ADXL001 to a tap made on a piece of acrylic 28.28 cm from it.	24
2.10	Magnitude response of a Murata PKS1-4A1 to a tap made on a piece of acrylic 28.28 cm from it.	25
2.11	Magnitude response of a Knowles BU-21771 to a tap made on a piece of acrylic 28.28 cm from it.	25
2.12	Block diagram of a touch interface.	26
2.13	Block diagram of the mathematical model of a physical object.	26
2.14	Diagram of TI with a matrix of predefined points.	28
2.15	Shaping function $W_{5,5}$ for mode $l = 5$, $b = 5$	31
2.16	Magnitude response of a plate for tap position (0.1,0.1) m.	31
2.17	Block diagram of the mathematical model of the sensor.	32
2.18	Block diagram of the mathematical model of the transforming block.	33
2.19	The frequency response of a Murata PKS1-4A1 accelerometer [after [2]].	36
2.20	Sensor outputs of first and second impacts.	36
2.21	Spectrum of sensor output for different impacts.	37
2.22	Schematic of the DI circuit.	38
2.23	AC analysis of the DI circuit.	39

2.24	Simplified AC analysis of the DI circuit.	39
2.25	DC analysis of the DI circuit.	39
2.26	Fabricated DI circuitry.	40
2.27	DI circuitry mounted in an aluminium chassis.	40
2.28	View of fabricated DI box.	41
2.29	Frequency response of the DI circuit.	41
2.30	The NI USB-6259 data acquisition unit [after [3]].	43
2.31	The M-Audio ProFire2626 [after [4]].	43
2.32	The PreSonus FireStudio [after [5]].	44
2.33	Final hardware block diagram.	47
2.34	Touch interface setup.	47
3.1	Diagram of TI with a matrix of predefined points.	52
3.2	Received signal of taps made at tap positions (41,20) cm and (42,20) cm. . .	53
3.3	Spectrum overlayed with AP-LTM of taps for $I = 3$, made at (a) (41, 20) cm and (b) (42, 20) cm.	55
3.4	Complexity of the CC-LTM and AP-LTM.	58
3.5	Experiment setup Acrylic board with the accelerometer.	59
3.6	Variation of accuracy for different AP filter orders.	60
3.7	Spectrum overlayed with AP-LTM of taps, for $I = 12$ made at (a) (41, 20) cm and (b) (42, 20) cm.	61
3.8	Illustration of estimated positions by CC-LTM and AP-LTM which are marked using ‘*’ and ‘∇’, respectively. The true source location is marked as ‘●’. The numbers beside each location denotes the number of taps estimated at the corresponding location.	63
4.1	Hyperbolic functions defined by TDOA between three sensors.	66
4.2	Signal $\mathbf{u}_1(n)$	67
4.3	Signal $\mathbf{u}_2(n)$	68
4.4	The cross correlation of $\mathbf{u}_1(n)$ and $\mathbf{u}_2(n)$	68
4.5	Photo of an Analog Device SHARC 21469 with its audio extender module. .	73
4.6	Block diagram of initial prototype.	74
4.7	A simplified block diagram of the TDOA algorithm robust to dispersion implemented in the DSP.	75
4.8	Screenshot of GUI.	76
5.1	Block diagram of initial prototype.	86
5.2	Picture of the coordinator of the wireless prototype.	88

5.3	Picture of the Jennic JN5148 end-device connected to the Analog Devices SHARC 21469 DSP board.	93
5.4	A 8 m × 3 m glass surface with eight sensors labeled S1-S8.	96
5.5	Signal received by S1, S2, S5 and S6 due to a tap at (1.5, 2.0) m.	97
5.6	Signal received by S3, S4, S7 and S8 due to a tap at (1.5, 2.0) m.	97
5.7	Sensors mounted on a glass plate with example of localization using different configurations.	100
5.8	Picture of the setup of the wireless prototype.	101
5.9	Results of estimation error using different configurations.	103
5.10	Testing the wireless prototype.	106
A.1	Flowchart of the main function of the coordinator of the wireless prototype. .	123
A.2	Flowchart of the baseband system initialization of the coordinator.	124
A.3	Wireless system initialization flowchart of the coordinator.	125
A.4	Flowchart of event queues of the coordinator.	125
A.5	Flowchart of the association request function of the coordinator.	126
A.6	Flowchart how incoming wireless data is processed by the coordinator. . . .	127
A.7	Flowchart of the pre-processing function of the coordinator.	128
A.8	Flowchart of the timer ISR of the coordinator.	129
A.9	Wireless UART transmission flowchart for the coordinator.	130
A.10	Flowchart of the main function of the end-device.	131
A.11	Flowchart of the baseband system initialization of the end-device.	132
A.12	Flowchart of event queues of the end-device.	133
A.13	Flowchart of the association request function of the end-device.	134
A.14	Flowchart of the Process incoming data function of the end-device.	135
A.15	Flowchart of the timer ISR of the end-device.	136
A.16	Flowchart of the wireless UART transmission of the end-device.	137

LIST OF TABLES

2.1	Modeling parameter values of the Murata PKS1-4A1 accelerometer	34
2.2	Modeling parameter values of the acrylic plate	35
2.3	Inter-channel delay test result: NI USB-6259.	45
2.4	Inter-channel delay test result: PreSonus FireStudio.	46
3.1	Computation complexity of CC-LTM and AP-LTM.	57
5.1	Sensor locations on 8 m $\times$ 3 m glass.	99

LIST OF SYMBOLS

t	Time index
$V(t)$	Signal generated by a finger tap
n	Sample iteration
m	Position index
(x_m^t, y_m^t)	Co-ordinates of tap location
(x_q^t, y_q^t)	Co-ordinates of q^{th} sensor location
$u(t)$	Signal received by sensor
$\mathbf{u}(n)$	Digitized signal received by sensor
L_u	Length of digitized signal received by sensor
$\mathbf{h}_m(n)$	Pre-recorded signal
N	Number of predefined locations
$\mathbf{c}(m)$	Vector of correlation coefficients
$\hat{m}$	Estimated tap index
$w_m(x, y, t)$	Vertical displacement of the plate
$P_m(x, y, t)$	Observed pressure
s	Laplace operator
$\mathbf{w}_m(x, y, s)$	Laplace representation of the vertical displacement of the plate
$\mathbf{W}_{m,p}(x, y, x_m^t, y_m^t, s)$	The Laplace representation of the transfer function of the physical object
$\omega_{l,b}$	Mode frequencies
$\mathcal{W}_s(s)$	Laplace transform of the transfer function of the accelerometer
$U(s)$	Laplace transform of the output of the accelerometer
$\hat{u}(n)$	Predicted signal
$e(n)$	Error function between the actual and predicted signal
i	Filter index
I	Prediction order
$a(i)$	Filter coefficient
j	Complex operator
$a^u(i)$	Filter coefficient of received signal
$a_m^h(i)$	Filter coefficient of pre-recorded signal
S_m	Summed squared distance between the filter coefficients of the received and pre-recorded signals

LIST OF ABBREVIATIONS

ADC:	Analog to digital converter
AP:	All-pole
AP-LTM:	All-pole based location template matching
CC-LTM:	Cross-correlation based location template matching
dB:	Decibels
DSP:	Digital signal processing
FFT:	Fast Fourier transform
HCI:	Human computer interface
LTM:	Location template matching
TDOA:	Time-difference-of-arrival
TI:	Touch interface
PC:	Personal computer
UART:	Universal asynchronous receiver/transmitter

1. INTRODUCTION

Existing human-computer interface (HCI) technology such as the alphanumeric keyboard and the optical pointing device (mouse) have become mainstays of computing, both on a commercial and consumer basis. Nonetheless, digital media software applications have advanced significantly in recent years and now require complicated user input operations that are beyond the capabilities of the current day keyboard and mouse. These input devices impede, to some extent, restrict the scope and the user friendliness of the personal computer (PC).

To address these issues, HCI research in recent years focuses on technologies which enhances the user's experience on the PC in a more natural manner. Some of these technologies include gyro-based sensors to track head rotation [6], vision-based cameras for the detection of both eye and hand motion [7], as well as localization of fingers on the surface of a waveguide medium via the frustrated total internal reflection principle of light-emitting diodes [8]. Another well-known HCI technology that has been utilized in portable devices such as the smart-phones and laptops is the touch screen. When a touch screen is tapped by a finger, it changes the electrostatic field and a layer of embedded capacitive sensors localizes the finger tap by detecting and locating such changes on the touch-sensitive surface. Due to its accuracy, this proven technology has gained much popularity. Nonetheless, its high implementation cost often confines the implementation to screens of small sizes, such as those of mobile phones.

Despite the commercial limitation of the touch screen technology, one technology has recently drawn interest from the research community. This HCI technology allows one to convert ordinary surfaces such as window panes and table-

tops into a touch interface (TI). This is achieved by using surface-mounted sensors and source localization algorithms for finger taps. In particular, the TI considered in [9], [10] describes impact localization methods based on well-known techniques such as time-differences-of-arrival (TDOAs) and location template matching (LTM). The effectiveness of both methods has been experimentally verified for isotropic [11] and anisotropic [12] materials. While the localization of finger taps on hard surfaces (e.g., wood and glass) is successful, sensors using vibration signals to detect such localization continue to face many challenges. These challenges include multipath, temperature variation, different types of wave propagation, as well as wave distortion due to dispersive effects of the channel between the finger tap and the sensor. These problems affect the accuracy of source localization to a large extent and hence most of the research results focus on improving the accuracy of the source localization algorithms.

By utilizing an array of sensors, the TDOA algorithm estimates the differences between the arrival times of the signals to these sensors. A hyperbolic function is first defined between two sensors, where the sensors correspond to the foci of the hyperbola [13]. Using several TDOA estimates, the intersection of these hyperbolae provide an estimate of the source location [13]. This intersection can be achieved by solving a set of non-linear equations related to the known sensor positions and the assumed source position [13]. Investigations into the TDOA method show that the feasibility of localization of finger taps on a surface [9], depends on the assumption that the speed of wave propagation is known. However, this assumption does not hold particularly well for solid surfaces due to the dispersive effects where different frequencies propagate at different speeds [9]. In addition, similar to acoustic propagation in air within an enclosed space, the presence of multipaths within a material increases the difficulty of TDOA estimation.

The use of LTM algorithm for source localization on solids has also been

explored. As opposed to TDOA-based approaches, only a single sensor is required for LTM where the algorithm localizes finger taps by comparing the received signal with a set of pre-recorded signals from known locations. To quantify this comparison, LTM computes the similarity by employing the cross-correlation measure (CC-LTM) [14]. It is important to note that, unlike the TDOA-based approaches, LTM exploits the dissimilarity between the source-sensor paths for different source locations. This dissimilarity results, to a large extent, from the presence of multipath and dispersion. It is therefore expected that LTM-based algorithms can achieve higher localization performance compared to TDOA-based approaches for anisotropic materials. Due to the complexity of the problem, results presented thus far have been focused on improving the accuracy of the source localization algorithms in a relatively ad-hoc manner.

In this thesis, we discuss both the TDOA and LTM-based source localization algorithms. We also propose a new approach to the development of source localization algorithms for a TI that employs surface-mounted sensors. This source localization algorithm allows one to convert any flat surface into a touch pad. To do so, we first develop the hardware required for the data acquisition of wave propagation in solids in Chapter 2. In that chapter we also describe a mathematical model to examine the frequency characteristics of a typical tap which we in turn exploit for the development of LTM-based localization algorithm.

Various applications and approaches to template matching are then reviewed in Chapter 3. This includes the application of template matching in image and audio signal processing, as well as LTM. In that chapter we propose and develop an all-pole filter based LTM (AP-LTM) algorithm to localize finger taps on a surface. The algorithm models the peak frequencies corresponding to each tap location. The performance of the proposed algorithm is compared with existing approaches and verified both in a synthetic as well as a real environment for the localization of a finger tap on a TI. We subsequently review the TDOA algorithm

for source localization in Chapter 4, and review a real-time implementation of a TDOA-based algorithm that is robust to dispersion.

In Chapter 5, we explore the feasibility of implementing a low-cost, consumer-focused real-time wireless prototype of the TDOA-based algorithm, and discuss some of the limitations encountered using various wireless protocols. We next propose a method that overcomes the time-synchronization limitations encountered when using the wireless protocol. Finally, Chapter 6 summarizes this report and lists the publications that arise from this work.

2. HARDWARE DEVELOPMENT FOR DATA ACQUISITION

In order to develop new algorithms and to verify the accuracy of different source localization algorithms, it is necessary to collect vibration signals generated by taps made on different surfaces. We therefore need to develop a hardware data acquisition system capable to digitizing and storing vibration signals for algorithmic development and performance evaluation. In this chapter we look at the different components that are required.

Figure 2.1 shows the basic block diagram of the components required for data collection. The sensor is used to detect vibrations generated by a tap on the surface under test while the multichannel analog-to-digital converter (ADC) will digitize the signal before storing them into the personal computer (PC). Each of the components required for the data collection process is explained in the subsequent sections. In order to make informed choices of the hardware we choose, we first analyze the mechanical model of wave propagating in solids to gain insight to the time and frequency information generated by a typical tap. This will provide information pertaining to the bandwidth requirements of the sensor.

2.1 *Vibration Sensor*

When a subject taps his finger on a surface, a vibration wave propagates over the surface. This vibration, as discussed in [10]- [15] and [16], ranges up to 2500 Hz. This frequency range is supported by the mechanical model as will be seen in

Fig. 2.1: Hardware block diagram.

Section 2.2. To detect this vibration, an accelerometer, which is capable of sensing vibration, is required. However as most accelerometers are normally designed to monitor *motion*, such as the free-fall, tilt, and shock detection, they normally have a frequency range of up to 100 Hz. As such only the *vibration* monitoring range of accelerometers can be considered for our HCI application. One of such is the piezoelectric based accelerometers.

One advantage of using a piezoelectric accelerometer is that the piezoelectric element in the accelerometer converts acceleration into electrical signal which, in turn, allows one to directly interface the sensor with an ADC. In addition, depending on its construction, these accelerometers have different sensitivity which is inversely proportional to measurement range that the sensor can detect. Defining $g = 9.81 \text{ m/s}^2$. An accelerometer with a sensitivity of 500 mV/g is more sensitive to minute motion changes when compared to an accelerometer with a sensitivity of 50 mV/g. However the accelerometer with a sensitivity of 50 mV/g would be able to monitor a larger range of gravitational-forces since it will not clip during data acquisition. It is therefore important to note that the sensitivity of the accelerometer is an important parameter especially when considering the input voltage range of the ADC. Several accelerometers that are considered in this work includes the Analog Devices ADXL001, the Murata PKS1-4A1, and the Knowles BU-21771.

2.1.1 The Sensors

The Analog Devices ADXL001, as seen in Fig. 2.2, is a $5 \text{ mm} \times 5 \text{ mm} \times 2 \text{ mm}$ micro-electromechanical system (MEMS) piezoelectric accelerometer packaged in a leadless ceramic carrier package [1]. As an integrated circuit, it requires powering. This accelerometer has a resonant frequency of 22 kHz and a flat response operation of 0-16 kHz, and has a sensitivity of 16 mV/g [1].

The Murata PKS1-4A1 is a piezoelectric accelerometer that detects vibra-

Fig. 2.2: The Analog Devices ADXL001 [after [1]].

tion and impact on a single axis and is able to pick up microlevel impacts and vibrations [2]. As seen in Fig. 2.3, it is packaged in a disc-like plastic casing of diameter 25 mm with a thickness of 7 mm [2]. This accelerometer has a resonant frequency of 3 kHz, a flat response of 0-2.5 kHz, and is able to generate 40 mV/g [2].

Fig. 2.3: A Murata PKS1-4A1 accelerometer.

The Knowles BU-21771 is another piezoelectric-based accelerometer and as seen in Fig. 2.4, it is encased in a metal housing of dimension 7.9 mm $\times$ 5.6 mm $\times$ 4.1 mm. This accelerometer was designed as a contact microphone that can be placed at the throat or on any bony parts of the head [17]. Its main intended application is a microphone for radio communications in high noise environments such as fire-fighting or during combat mission [17]. This accelerometer has a resonant frequency of 18 kHz and has a flat response of 0-10 kHz. It has a sensitivity of 5.62 mV/g, and requires an external power source for its built-in amplifier [17].

As discussed in [18], the bandwidth of a typical flat surface ranges from 0 to 2.5 kHz, the above sensors suit our HCI application. Although these piezoelec-

Fig. 2.4: The Knowles BU21771 mounted on aluminium with its wires held down using Blu-Tack.

tronic accelerometers considered meet the bandwidth requirement, it is noted that the Murata PKS1-4A1 has the narrowest bandwidth of 2.5 kHz while the other two accelerometers have a bandwidth of at least 10 kHz. In addition, the Murata PKS1-4A1 is the most sensitive.

2.1.2 Testing the Sensors

To test the response of the sensors, three independent experiments are conducted. In each experiment, each of the fore-mentioned sensors are mounted at (20, 20) cm on a piece of 80 cm $\times$ 80 cm $\times$ 0.52 cm acrylic and a tap is made at (40, 40) cm. The signals are analyzed and recorded using a LeCroy WaveSurfer 44Xs-A oscilloscope.

Figs. 2.5-2.7 show the time-amplitude responses of the Analog Devices ADXL001, the Murata PKS1-4A1 and the Knowles BU-21771 to a tap made by a subject. From Fig. 2.5, we note that while the Analog Devices ADXL001 has a relatively clean signal, its amplitude-time response is only in the range of ± 0.5 mV. Closer examination of the signal, as seen in Fig. 2.8, reveals that this active sensor suffers from minor quantization noise which, as described in its datasheet [1], is due to the charge amplifier in the output portion of the sensor. As this quantization noise occurs in the nanosecond range, it may contribute some unwanted high frequency components to the signal and as such this must be taken note of when using this accelerometer.

Fig. 2.5: Time-amplitude response of an Analog Devices ADXL001 to a tap made on a piece of acrylic 28.28 cm from it.

Fig. 2.6: Time-amplitude response of a Murata PKS1-4A1 to a tap made on a piece of acrylic 28.28 cm from it.

Fig. 2.6 shows the time-amplitude response of the Murata PKS1-4A1 to a tap. From the figure, we note that this accelerometer has a relatively clean signal and has a relatively large amplitude response of ± 0.1 V. Fig. 2.7 shows the time-amplitude response of the Knowles BU-21771 to a tap made on a the acrylic 28.28 cm from it. From the figure, we note that this sensor has a maximum peak-to-peak amplitude of ± 40 mV.

Figs. 2.9 - 2.11 show the magnitude response of the data acquisition system for the Analog Devices ADXL001, the Murata PKS1-4A1 and the Knowles BU-21771 respectively. For ease of comparison, these responses are normalized such that the maximum amplitude is 0 dB. From Figs. 2.9 - 2.11, we note that beyond 2 kHz, the amplitude response starts to decay significantly. However, as

Fig. 2.7: Time-amplitude response of a Knowles BU-21771 to a tap made on a piece of acrylic 28.28 cm from it.

Fig. 2.8: Close up view of the response of an Analog Devices ADXL001 to a tap made on a piece of acrylic 28.28 cm from it.

Fig. 2.9: Magnitude response of a Analog Devices ADXL001 to a tap made on a piece of acrylic 28.28 cm from it.

seen in Fig. 2.10, the frequency response of the sensor at approximately 3 kHz is significantly higher than the other sensors since the Murata PKS1-4A1 has a resonant frequency of approximately 3 kHz. In terms of mounting, since the Murata PKS1-4A1 is packaged in a plastic casing, it is the easiest to mount/remove from a surface. In addition, since it is a passive device, there is no need for external power circuitry and as such, it is the most suitable sensor for our HCI application.

Comparing the time-amplitude responses of these three sensors, it can be said that the Knowles BU-21771 accelerometer picks up the most ambient noise. This is most likely since the BU-21771 was designed as a contact microphone and is sensitive to ambience noise. Despite being a passive sensor, the Murata PKS1-4A1 has the highest amplitude response of ± 0.1 V, compared to ± 0.5 mV and ± 40 mV for the Analog Devices ADXL001 and the Knowles BU-21771 respectively.

2.2 Mechanical Model

As the aim of this research is to localize taps made on solids for HCI, it is therefore necessary to understand the mechanical effects of a piece of solid when it is tapped by a finger. We therefore explore, in this section, a model that exploits the mechanics of vibrating waves so as to gain insight into the properties of wave propagation on solids. This in turn allows us to exploit any such characteristic

Fig. 2.10: Magnitude response of a Murata PKS1-4A1 to a tap made on a piece of acrylic 28.28 cm from it.

Fig. 2.11: Magnitude response of a Knowles BU-21771 to a tap made on a piece of acrylic 28.28 cm from it.

Fig. 2.12: Block diagram of a touch interface.

Fig. 2.13: Block diagram of the mathematical model of a physical object.

of wave propagation for the development of a location template matching (LTM) algorithm as presented in Chapter 3.4. Fig. 2.12, describes a model which is comprised of models for the physical object and the sensor. The aim of these blocks is to transform a finger tap $V(t)$ made at the coordinates (x_m^t, y_m^t) into electrical signals, where we define the $m = 1, \dots, N$ as the position index while N denotes the total number of possible tap locations. Here, the superscript ‘t’ denotes explicitly for the finger tap. Since the accelerometer is used to detect wave propagation in solids, a transforming block is required to convert acceleration to vertical displacement. The subsequent subsections in this chapter will describe this transformation.

2.2.1 Mathematical model of a rectangular plate

As seen in Fig. 2.12, the touch interface is mathematically modeled for the purpose of analysis, using three blocks. The first block, as illustrated in Fig. 2.13,

describes the mathematical model of a physical object, where an input tap $V(t)$ is transformed into a vertical displacement of the plate $w_m(x^s, y^s, t)$, where the superscript ‘s’ denotes explicitly for the sensor. To begin the analysis, we consider a rectangular plate; a structure that covers a wide range of commonly seen and used objects such as tables, walls, and windows [19]. As shown in [20], the response of the plate due to an impact can be represented by the classical plate theory. With reference to Fig. 3.1, we define the coordinate of the m^{th} tap location as (x_m^t, y_m^t) . We employ the well-known Kirchoff’s hypotheses for linear, elastic small-deflection theory of thin plates. Defining $P_m(x^s, y^s, t)$ as the pressure observed at location (x^s, y^s) on the plate at time instance t due to a finger impact $V(t)$ made at location (x_m^t, y_m^t) , the vertical displacement $w_m(x^s, y^s, t)$ of the plate at (x^s, y^s) location must satisfy the following motion equality [18]

$$P_m(x^s, y^s, t) = D\nabla^4 w_m(x^s, y^s, t) + \mu \frac{dw_m(x^s, y^s, t)}{dt} + \rho L_z \frac{d^2 w_m(x^s, y^s, t)}{dt^2}, \quad (2.1)$$

where μ is the absorption coefficient, ρ is the density, L_z is the thickness of the plate, $D = EL_z^3/12(1 - \nu^2)$, E is the Young’s modulus, ν is the Poisson’s ratio while $\nabla^4 = \frac{\partial^4}{\partial x^4} + 2\frac{\partial^4}{\partial x^2 \partial y^2} + \frac{\partial^4}{\partial y^4}$.

In order to solve for (2.1), we first note that a point impact at finger location (x_m^t, y_m^t) can be expressed as

$$P_m(x^s, y^s, t) = V(t)\delta(x - x_m^t)\delta(y - y_m^t), \quad (2.2)$$

where $\delta(x)$, and $\delta(y)$ are the Dirac delta functions. The applied force can then be extended into a series given by [18]

$$P_m(x^s, y^s, t) = V(t) \sum_{l=1}^{\infty} \sum_{b=1}^{\infty} F_{lb} \sin(\alpha_l x) \sin(\beta_b y), \quad (2.3)$$

Fig. 2.14: Diagram of TI with a matrix of predefined points.

where

$$\begin{aligned}
 F_{m,lb}(x^s, y^s) &= \frac{4}{L_x L_y} \int_0^{L_x} \int_0^{L_y} \delta(x - x_m^t) \delta(y - y_m^t) \sin(\alpha_l x) \sin(\beta_b y) dx dy \quad (2.4) \\
 \alpha_l &= \pi l / L_x, \\
 \beta_b &= \pi b / L_y,
 \end{aligned}$$

and $l, b \in \mathbb{Z}^+$ are the modes of wave propagation, L_x and L_y , respectively, the length and width of the plate.

To simplify the problem, $w_m(x^s, y^s, t)$ is decomposed into time- and position-dependent functions $C_{lb}(t)$ and $W_{lb}(x^s, y^s)$ given by [18]

$$w(x^s, y^s, t) = \sum_{l=1}^{\infty} \sum_{b=1}^{\infty} C_{lb}(t) W_{lb}(x^s, y^s), \quad (2.5)$$

where

$$C_{lb}(t) = A_{lb} \cos(\omega_{lb} t) + B_{lb} \sin(\omega_{lb} t), \quad (2.6)$$

such that A_{lb} and B_{lb} are the coefficients of the cosine and sine terms respectively

and that

$$\omega_{lb} = \left(\alpha_l^2 + \beta_b^2 \right) \sqrt{\frac{D}{\rho L_z}}. \quad (2.7)$$

The position dependent function can be represented in the form of

$$W_{lb}(x^s, y^s) = \sin(\alpha_l x) \sin(\beta_b y) \sin(\alpha_l x_m^t) \sin(\beta_b y_m^t). \quad (2.8)$$

Furthermore, substituting (2.2) and (2.5) into (2.1), the solution of (2.1) is given by

$$\begin{aligned} V(t) \delta(x - x_m^t) \delta(y - y_m^t) &= D \nabla^4 w_m(x^s, y^s, t) + \\ &\quad \mu \frac{dw_m(x^s, y^s, t)}{dt} + \\ &\quad \rho L_z \frac{d^2 w_m(x^s, y^s, t)}{dt^2} \end{aligned} \quad (2.9)$$

$$\begin{aligned} V(t) F_{m,lb}(x^s, y^s) \sin(\alpha_l x) \sin(\beta_b y) &= D \nabla^4 W_{lb}(x^s, y^s) C_{lb}(t) + \\ &\quad \mu W_{lb}(x^s, y^s) \dot{C}_{lb}(t) + \\ &\quad \rho L_z W_{m,lb}(x^s, y^s) \ddot{C}_{lb}(t). \end{aligned} \quad (2.10)$$

where $\dot{C}_{lb}(t)$ and $\ddot{C}_{lb}(t)$ are, respectively, the first and second differential order of $C_{lb}(t)$.

By transforming (2.9) to the Laplace domain, we obtain

$$\begin{aligned} V(s) F_{m,lb}(x^s, y^s) \sin(\alpha_l x) \sin(\beta_b y) &= D \nabla^4 W_{lb}(x^s, y^s) C_{lb}(s) + \\ &\quad \mu W_{lb}(x^s, y^s) \left[s C_{lb}(s) - C_{lb}(0) \right] + \\ &\quad \rho L_z W_{m,lb}(x^s, y^s) \left[s^2 C_{lb}(s) - \right. \\ &\quad \left. s C_{lb}(0) - C_{lb}(0) \right], \end{aligned} \quad (2.11)$$

where s is the Laplace variable and that the Laplace transform of the time-

dependent function $C_{lb}(t)$ is given by

$$\begin{aligned} C_{lb}(s) &= \mathcal{L}\{C_{lb}(t)\} \\ &= A_{lb} \frac{s}{s^2 + \omega_{lb}^2} + B_{lb} \frac{\omega_{lb}}{s^2 + \omega_{lb}^2} \\ &= \frac{A_{lb}s + B_{lb}\omega_{lb}}{s^2 + \omega_{lb}^2}, \end{aligned} \quad (2.12)$$

such that $\mathcal{L}$ is the Laplace function. Solving for (2.1) in the Laplace domain for zero initial condition gives rise to [18]

$$\mathcal{W}_m(x^s, y^s, s) = \mathbf{W}_m(x^s, y^s, x_m^t, y_m^t, s) \mathcal{V}(s), \quad (2.13)$$

where $\mathcal{V}(s)$ is Laplace transform of the function $V(t)$. A zero initial condition is assumed as the mechanical model assumes a supported plate condition where $W_{lb}(x^s, y^s, s) = 0$ along the plate's edges, ie $x = 0$, $y = 0$, $\frac{\partial^2(W_m)}{\partial(x^2)} = 0$, and $\frac{\partial^2(W_m)}{\partial(y^2)} = 0$. The variable $\mathcal{W}_m(x^s, y^s, x_m^t, y_m^t, s)$ is the transfer function of the physical object and is in the form of

$$\mathcal{W}_m(x^s, y^s, x_m^t, y_m^t, s) = \sum_{l=1}^{\infty} \sum_{b=1}^{\infty} \frac{4}{\rho L_x L_y L_z} \frac{W_{lb}}{s^2 + \tilde{\mu}s + \omega_{lb}^2}, \quad (2.14)$$

where $\tilde{\mu} = \mu/(\rho L_z)$ is the reduced coefficient of absorption. By transforming (2.14) back to the time domain, we obtain an important expression for the vertical displacement at (x^s, y^s) due to a tap made at (x_m^t, y_m^t) given by

$$w_m(x^s, y^s, t) = \frac{4}{\rho L_x L_y L_z} \sum_{l=1}^{\infty} \sum_{b=1}^{\infty} W_{lb} \left(\frac{e^{-0.5\tilde{\mu}t} \sin(\xi_{lb}t)}{\xi_{lb}} \right), \quad (2.15)$$

where $\xi_{lb} = \sqrt{\omega_{lb}^2 - (\frac{\tilde{\mu}}{2})^2}$.

As shown in (2.15), $w_m(x^s, y^s, t)$ can be represented as the summation of mode response corresponding to each mode frequency ω_{lb} . For a given tap location and material, each mode response can be represented as the product of the shaping function $W_{lb}(x^s, y^s)$ and the time-dependent function $\frac{e^{-0.5\tilde{\mu}t} \sin(\xi_{lb}t)}{\xi_{lb}}$. It can be seen that the time-dependent function is inversely proportional to ξ_{lb} . This implies that as ω_{lb} increases with l, b , the contribution of that mode to $w_m(x^s, y^s, t)$ reduces

Fig. 2.15: Shaping function $W_{5,5}$ for mode $l = 5$, $b = 5$.

Fig. 2.16: Magnitude response of a plate for tap position (0.1, 0.1) m.

with respect to mode $l = 1, b = 1$. In addition, as illustrated by Fig. 2.15 for a tap made at location $(x_m^t, y_m^t) = (0.1, 0.1)$ m, the shaping function $W_{lb}(x^s, y^s)$ is dependent on the location (x, y) . More importantly, as shown in (2.8), $W_{lb}(x^s, y^s)$ is a function of (x_m^t, y_m^t) . This implies that the time-dependent function is weighted by a location-dependent shaping function $W_{lb}(x^s, y^s)$ and as a result, the vertical displacement $w_m(x^s, y^s, t)$ will vary with tap position due to its amplitude variation of mode frequencies with respect to time.

To illustrate the above, Fig. 2.16 shows the magnitude response of a plate when a tap is made at $(x_m^t, y_m^t) = (0.1, 0.1)$ m. The parameters used in this simulation example is tabulated in Table 2.2 for a piece of acrylic. As can be

Fig. 2.17: Block diagram of the mathematical model of the sensor.

seen, the peak frequencies are due to the modal response of the plate, which as seen in (2.15), the maximum amplitude of the modes are due to the components $W_{lb}(x^s, y^s)$ and ξ_{lb} . From this figure we note that the frequencies beyond 4 kHz contribute lesser energy than the energy of the first few modes.

2.2.2 Mathematical model of accelerometer

To understand the use of the transforming block as seen in Fig. 2.12, it is necessary to first understand the the model of the sensor shown in the rightmost block. There are different types of sensors including optical sensors and strain gauges that can be used to detect vibrational waves on hard surfaces. However one of the most cost effective solutions is the use of surface-mounted accelerometers which can, in this case, be deployed to localize impacts. The accelerometer is a simple sensor which can detect wave propagation. This sensor can be characterized by the frequency response, with its output voltage dependent on the input acceleration. Given $u(t)$ as the output of accelerometer, $g(t)$ is the input acceleration, as shown in Fig. 2.17, the behavior of such a sensor can be described by a second-order linear differential equation of the form [21]

$$M \frac{d^2 u(t)}{dt^2} + \varsigma \frac{du(t)}{dt} + ku(t) = g(t), \quad (2.16)$$

where M is the proof mass in the accelerometer, ς is the damping coefficient of the material used to construct the accelerometer, and k is the stiffness of the spring element found in the accelerometer. The corresponding transfer function is then

Fig. 2.18: Block diagram of the mathematical model of the transforming block.

given, in the Laplace domain, by

$$\begin{aligned}
 G(s) &= M[s^2U(s) + sU(0) + u(0)] + \varsigma[sU(s) + u(0)] + kU(s) \\
 &= [Ms^2U(s) + \varsigma sU(s) + kU(s)] + M[sU(0) + u(0)] + \\
 &\quad \varsigma u(0).
 \end{aligned} \tag{2.17}$$

Assuming a zero initial condition, the transfer function of the sensor is obtained as

$$\begin{aligned}
 \mathcal{W}_s(s) &= \frac{U(s)}{G(s)} \\
 &= \frac{K_s}{T^2s^2 + 2\zeta Ts + 1},
 \end{aligned} \tag{2.18}$$

where $K_s = \frac{1}{k}$ is the gain coefficient, $T = \sqrt{\frac{M}{k}}$ is the time constant and $\zeta = \frac{\varsigma}{2\sqrt{Mk}}$ is the relative damping coefficient.

2.2.3 Model between impact and sensor output

Having described the physical object and sensor model, we integrate these models to obtain the transfer function between the impact and the output of the accelerometer. As seen in Fig. 2.18, a transforming block is required to interface the physical object model and the sensor model together. Since the output model of the physical object is that of a vertical displacement, this output must be converted into acceleration by means of s^2 . The transfer function between the impact

Tab. 2.1: Modeling parameter values of the Murata PKS1-4A1 accelerometer

Parameter	Parameter value
T	$7.1 \times 10^{-7} \text{ s}$
ζ	0.1
K_s	$3.5 \times 10^{-3} \text{ V}/(\text{ms}^{-2})$

and the sensor output is therefore given by

$$\begin{aligned}
\frac{\mathcal{U}(s)}{\mathcal{V}(s)} &= \mathcal{W}_s(s) \times s^2 \times \mathcal{W}(x^s, y^s, x_m^t, y_m^t, s) \\
&= \frac{K_s}{T^2 s^2 + 2\zeta T s + 1} \times s^2 \times \\
&\quad \sum_{l=1}^{\infty} \sum_{b=1}^{\infty} \frac{4}{\rho L_x L_y L_z} \frac{W_{lb}(x^s, y^s)}{s^2 + \tilde{\mu}s + \omega_{lb}^2}, \tag{2.19}
\end{aligned}$$

where (x^s, y^s) are the coordinates of the accelerometer.

It is therefore important to see that (2.19) is not just a function of the finger tap location but also a function of frequency through the Laplace variable s .

2.2.4 Illustrative example using the mathematical model

We now provide an illustrative example of how the model allows one to examine the properties of the received signals. By defining for the Murata PKS1-4A1 accelerometer, a time constant $T = 7.1 \times 10^{-7} \text{ s}$, a relative damping coefficient of $\zeta = 0.1$ and a gain coefficient of $K_s = 3.5 \times 10^{-3} \text{ V}/(\text{ms}^{-2})$. These values, as seen in Table 2.1, have been selected to match the frequency response of the sensor found in its datasheet, shown in Fig. 2.19. Fig. 2.19 shows the frequency response of a Murata PKS1-4A1 accelerometer, where the sensor, has a flat frequency response of 10 to $\sim 1500 \text{ Hz}$, 10 to $\sim 1000 \text{ Hz}$, and 10 to $\sim 5000 \text{ Hz}$ for impacts of

Tab. 2.2: Modeling parameter values of the acrylic plate

Parameter	Parameter value
$L_x \times L_y \times L_z$	80 cm $\times$ 80 cm $\times$ 0.52 cm
E	69×10^9 Pa
ν	0.35
ρ	2710 kg/m ³

1 g, 5 g, and 10 g respectively. As seen in the figure, the sensor has a resonant frequency of approximately 2 kHz. As such the parameters seen in Table 2.1 have been chosen specifically to match this response. Next, the properties of an acrylic surface, with dimensions 80 cm $\times$ 80 cm $\times$ 0.52 cm is defined. As seen in Table 2.2, the piece of acrylic has the following properties: $E = 69 \times 10^9$ Pa, $\nu = 0.35$, and $\rho = 2710$ kg/m³. The sensor parameters were chosen to match that of the resonant frequency while the parameters for the material are chosen so that they match that of [22] and [23]. In this synthetic simulation, a Murata accelerometer is placed at (0.3, 0.2) m while the coordinates of four impacts are (0.12, 0.12) m, (0.12, 0.68) m, (0.68, 0.68) m and (0.68, 0.12) m. For illustrative purposes, $l = b = 3$ has been used.

Figure 2.20 shows the signals received by the sensor from the first two impacts. For clarity of presentation, we only show the outputs of these two impacts. As can be seen, the signals exhibit some form of damping particularly after 5 ms. This is due to the frequency dependency nature of the signal given by (2.19). The magnitude spectra of these signals are illustrated in Fig. 2.21. As can be seen, these output signals include a high frequency component at 2 kHz that is due to

Fig. 2.19: The frequency response of a Murata PKS1-4A1 accelerometer [after [2]].

Fig. 2.20: Sensor outputs of first and second impacts.

the resonant frequency of the accelerometer. While it is important to note that the frequencies vary across different locations, we note that, for each tap location, dominant frequencies occur. We therefore propose to model these dominant frequencies using the all-pole (AP) model as described in Section 3.4.

2.3 Direct Injection (DI) Box

The direct injection (DI) box is commonly used in the professional audio arena. Its main purpose is to convert an unbalanced signal, into a balanced signal for the purpose of long distance transmission. The need for balanced line transmission is justified by the fact that beyond a transmission length of 5 m, electrical noise

Fig. 2.21: Spectrum of sensor output for different impacts.

can be significant [24]. Balanced lines on the other hand can be drawn over long distances and remains relatively immune to the additive noise. While most DI circuits use audio transformers, the implemented DI circuitry uses passive discrete components [25]. The passive component circuit was chosen over circuits that use transformers as most transformers have a limited bandwidth between 35 Hz to 18 kHz. Moreover, the cost of the audio transformers with wider bandwidths were found to be significantly more expensive. On the other hand, passive components had a flat frequency response from 5 Hz to over 10 MHz.

Fig. 2.22 shows the schematic of the DI circuit implemented, where the terminals VIN-1 and VIN-2 correspond to the positive and negative terminals of the signal input respectively. To gain a better understanding of the circuit, Fig. 2.23 shows the AC analysis of the DI circuit, where the capacitors are short-circuited. In addition, for this AC analysis, we note that both terminals of the resistor R5 are connected to the ground and hence, this resistor can be omitted. From the analysis seen in Fig 2.24, we note that the positive terminal of the input is connected to the cold pin of the output, while the negative terminal of the input is connected to the hot pin of the output. The ground pin, on the other hand, is

Fig. 2.22: Schematic of the DI circuit.

used as the reference voltage level.

Fig. 2.25 shows the simplified schematic of the DC analysis of the DI circuit after the capacitors have been treated as open-circuit. From the figure, we can see that the Zener diode limits the voltage at the intersecting node of R3 and R5 to $12 V_{DC}$. This is required since most audio devices such as the PreSonus FireStudio provides $48 V_{DC}$ from the input port of the ADC to the input port of the DI box in order to excite the diaphragm of the condenser or ribbon microphones. Under normal operation, this $48 V_{DC}$ phantom power is transmitted through the hot pin of a balanced connector. However as the cold pin is often mixed up with the hot pin, the cold pin is also equipped with the $22 k\Omega$ current limiting resistor. From this analysis we can see that AC coupling capacitors, C3 and C4 play the important role of passing phantom power to the input source without allowing the DC power from being transmitted back to the ADC, while resistors R1, R2 and R5 limit the current from the phantom power source and preventing it from damaging the DI box and the input transducer.

Each DI box consists of a chassis, connectors, and a circuit board, as seen in Fig. 2.26 and Fig. 2.27. The completed view of the DI box can be seen in

Fig. 2.23: AC analysis of the DI circuit.

Fig. 2.24: Simplified AC analysis of the DI circuit.

Fig. 2.25: DC analysis of the DI circuit.

Fig. 2.26: Fabricated DI circuitry.

Fig. 2.27: DI circuitry mounted in an aluminium chassis.

Fig. 2.28.

The magnitude response of the DI circuitry was subsequently verified. In this test, we used a function generator to generate sine waves of various frequencies at $0.5 V_{p-p}$ at the DI's input terminals. The output signals are then analyzed using the LeCroy WaveSurfer 44Xs-A oscilloscope. The result of this magnitude response test is shown in Fig. 2.29. As seen in the figure, the magnitude response increases from 1 Hz to 10 Hz, after which from 100 Hz to 1 MHz the magnitude response remains almost constant at approximately -8 dB. It is also seen that beyond 4.5 Hz, the magnitude response varies within the -3 dB range below the resonant frequency. This test suggests that the implemented device is able to provide a flat response within the audio frequency range.

Fig. 2.28: View of fabricated DI box.

Fig. 2.29: Frequency response of the DI circuit.

2.4 Analog-Digital Converter (ADC)

The ADC is another crucial component of the hardware setup as it is required to digitize the analog signals. As will be described in Chapter 4, since we will be localizing finger taps using time-differences-of-arrival (TDOAs), an ADC that can support *at least* eight channels is required. Although the discussion of sub-array algorithms is outside the scope of Chapter 2, ADC having at least eight channels will allow the research team to, for example, assess the possibility of utilizing wireless protocol and sub-array processing as described in Chapter 5. One of the requirements of the ADC is that it needs to have either no inter-channel delay, or a deterministic delay since, if any non-deterministic delay occurs, the time-differences would not be accurately reflected, giving the wrong tap location. Another requirement is that the ADC needs to have a sampling frequency of at least 96 kHz. Such high sampling rate is required due to the high velocity of waves propagating through the solids. A few of the possible ADCs are the National Instruments (NI) USB-6259, the PreSonus FireStudio and the M-Audio ProFire2626.

The NI USB-6259, as shown in Fig. 2.30, is capable of thirty-two single-ended inputs. However, the unit uses a round-robin method to switch between the inputs signal [3]. To achieve the 96 kHz sampling frequency preferred by the TDOA algorithm, the 1.25 MS/s of the ADC will have to be equally divided between the channels, such that the required eight inputs can be supported. Each channel's gain is digitally controlled using LabView when the ADC is connected to the PC via a USB cable.

The M-Audio ProFire2626 shown in Fig. 2.31, is a professional audio multi-channel recording system. It comes with eight XLR and 1/4" combination inputs, and can be daisy chained with up to two other ProFire2626s to achieve a total of twenty-four single-ended and differential inputs. This ADC is capable of sampling at 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, 176.4 kHz and 192 kHz. However at

Fig. 2.30: The NI USB-6259 data acquisition unit [after [3]].

Fig. 2.31: The M-Audio ProFire2626 [after [4]].

88.2 kHz and 96 kHz, the ADC can only support up to sixteen channels, and at 176.4 kHz and 192 kHz, it can only support up to twelve channels. As its name suggests, the ProFire2626 uses the Firewire input which, in this case, the 1394a or Firewire 400. The input ports for this ADC are located at the rear panel, while its gain control knobs are located on the front panel [4].

Fig. 2.32 shows a photo of the PreSonus FireStudio. Similar to the ProFire2626, the FireStudio is a professional audio multi-channel recording system, and has eight XLR or 1/4" inputs. Each FireStudio can be daisy chained with up to two other FireStudios to achieve a total of twenty-four single-ended and differential inputs. This ADC is capable of sampling at 44.1 kHz, 48 kHz, 88.2 kHz and 96 kHz. Similar to the Profire2626, at 88.2 kHz and 96 kHz, the FireStudio can only support up to sixteen channels. Like the ProFire2626, the FireStudio uses the 1394a or Firewire 400 input. The input ports of this ADC and

Fig. 2.32: The PreSonus FireStudio [after [5]].

its amplitude control knobs are also located at the front panel. One of the key advantages of the FireStudio over the ProFire2626 is that the word clock port, which is used to synchronize the clocks of the daisy-chained ADCs, is directly built into the system, using a BNC connector whereas the ProFire2626 integrated this port in its DB9 expansion port [5].

2.4.1 Synchronization of the ADCs

As each of the ADCs costs more than S\$1000, only the NI USB-6259 and the PreSonus FireStudio were acquired and tested. This section of the thesis documents some of the tests performed on the ADCs to ensure that the acquired equipment meet the specification requirements, more specifically the inter-channel delay. Non-deterministic inter-channel delay alters the instance the signal arrives at the ADC and as a result degrades the performance of the TDOA-based source localization algorithm, described in Chapter 4. Therefore to compute the inter-channel delay, we first connect all inputs of the ADC together to a single audio source. Next a white Gaussian noise (WGN) signal is fed into the inputs, and the signals are recorded using a PC. These signals are then cross-correlated and the time lag that achieves the highest cross correlation will correspond to the inherent time-delay between the two channels [26]. By defining $u_q(n)$ as the q^{th} received signal, and n as the sample index, the inherent delay between Channels q and r

Tab. 2.3: Inter-channel delay test result: NI USB-6259.

Channel r	Channel q								
		1	2	3	4	5	6	7	8
	1	0	0	0	-20	-20	-10	-10	-17
	2	0	0	0	-20	-20	-10	-10	-17
	3	0	0	0	-20	-20	-10	-10	-17
	4	20	20	20	0	0	10	10	3
	5	20	20	20	0	0	10	10	3
	6	10	10	10	-10	-10	0	0	7
	7	10	10	10	-10	-10	0	0	7
	8	17	17	17	-3	-3	-7	-7	0

is given by

$$\tau_{q,r} = \arg \max_n \sum_{n'=-\infty}^{\infty} u_q(n')u_r(n+n'), \quad (2.20)$$

where n' is the lag sample. Although the time-domain approach is presented here, one may choose to use the Fourier domain to compute the cross-correlation [26].

Ideally the inter-channel delay should be zero if the channels are sampled synchronously. However, in practical implementation, small amounts of delay is acceptable. This delay however needs to be consistent so that the channel delay can be compensated before TDOAs are estimated. For these tests only eight channels are used since the maximum number of sensors used is eight. The results of the inter-channel delay test, computed using (2.20), for the NI USB-6259 and PreSonus FireStudio are tabulated in Tables 2.3 and 2.4 respectively.

Table 2.3 shows the cross-channel delay (in samples) of the WGN signals recorded by the input channels of the NI USB-6259 ADC. As seen in the table, the diagonal elements of this table are zero since there is no lag time when a channel's recorded signal is correlated with itself. In addition, the upper triangular elements of this matrix is the negative of the lower triangular elements as expected. It is important to note that the inter-channel delay presented here is just one of the test results performed on the NI USB-6259. Subsequent tests revealed that the inter-channel delay varied inconsistently with the tests that have been performed

Tab. 2.4: Inter-channel delay test result: PreSonus FireStudio.

Channel r		Channel q							
		1	2	3	4	5	6	7	8
	1	0	0	0	0	0	0	0	0
	2	0	0	0	0	0	0	0	0
	3	0	0	0	0	0	0	0	0
	4	0	0	0	0	0	0	0	0
	5	0	0	0	0	0	0	0	0
	6	0	0	0	0	0	0	0	0
	7	0	0	0	0	0	0	0	0
	8	0	0	0	0	0	0	0	0

on it.

Table 2.4 shows the inter-channel delay extracted using the WGN signals recorded by the input channels of the PreSonus FireStudio ADC. As seen in the table, there is no inter-channel delay and these results are consistent across several independent tests. From this, we can conclude that the PreSonus FireStudio, which uses individual ADCs for each channel, is able to synchronize the clocks of the individual ADCs. On the other hand, the NI USB-6259 ADC which uses the round-robin method to cycle between the different input channels, has some limitations when cycling through the inputs. As a result inconsistency in the inter-channel delay is encountered. Thus the PreSonus FireStudio is chosen as the ADC to be used for this research.

With the Murata PKS1-4A1, the DI-box, and the PreSonus FireStudio in place, the data acquisition system can be utilized to record and analyze wave propagation on solids. The effective block diagram of this setup is shown in Fig. 2.33, where the sensor detects vibration on the surface, and instead of directly digitizing the signal using the ADC as shown in Fig. 2.1, the signal is conditioned using the DI box before being digitized by the ADC. The setup of the hardware, as seen in Fig. 2.34, shows sixteen Murata PKS1-4A1 accelerometers mounted on a piece of wood. These accelerometers are connected to the DI boxes which are

Fig. 2.33: Final hardware block diagram.

Fig. 2.34: Touch interface setup.

then connected to two PreSonus FireStudios.

2.5 Conclusion

In this chapter we described the data acquisition hardware that has been implemented for the development of source localization algorithms in solids. To do so, we analyzed a mathematical model for wave propagation in solids. This mathematical model gave us insights to the frequency characteristics of the received signals by the sensors. More specifically, there exists dominant frequencies which, as shown in Chapter 3, will be exploited for the development of the LTM-based localization algorithm. Next, we analyzed and implemented a DI circuitry that conditions the signal acquired from the sensor for signal transmission of longer than 5 m. Finally we look into an ADC which has a deterministic inter-channel delay which allows us to acquire signals generated by a tap.

3. TEMPLATE MATCHING

Template matching is a recognition technique used to determine an unknown phenomenon by comparing it with a set of known phenomena, whilst assuming that the unknown phenomenon has sufficient content to be used in the comparison process. When employed in signal processing, particularly in the domain of image and speech processing, template matching has been found to be useful in scenarios where automated digital systems are employed to identify analog signals. Such applications include the use of automated fingerprint readers [27] [28] [29], voice commands on mobile phones and computers [30] [31], and even sound recognition for different musical instruments [32].

In order for template matching to be successful, it is necessary to obtain sufficient pre-recorded data. If insufficient data is present in the pre-recorded data set, the recognition system may indicate a no match, or incorrect identification of the source. Therefore, in order to achieve good performance in template matching, it is necessary to have sufficient corpus size. However, in so doing, the computational complexity is increased. To address this issue, the authors of [33] proposed to use sub-templates in a multi-stage approach, whereas the authors of [34] proposed a feature extraction method that tracks the changes of an image. In addition, a Cauchy-Schwarz inequality bounded full search algorithm [35] has been proposed to reduce the complexity of template matching for image processing.

3.1 *Image Template Matching*

Template matching for image processing is an established area of research and has been deployed for fraud identification [27] [28] [29], image identification [27] [36],

as well as in the manufacturing process [37]. Some of the key challenges in this area of research arise when the image to be identified is rotated in two- or three-dimensions. Template matching for image processing is achieved by using either feature extraction-based or template-based matching. For feature extraction-based template matching, certain key features of the image, such as its edges or lines [27], as well as the motion [34] of the image are extracted and used for comparison.

However in situations where the template lacks compelling features or when a large portion of the template comprise of the image to be matched, the use of the whole template, as opposed to feature extraction-based template matching, may be preferred. The setback, however, of using the template-based method is that a large number of points of the template are required to be sampled. As such down-sizing algorithms such as the multi-resolution or pyramid [38] algorithms have been proposed to reduce the complexity.

Template matching algorithms have also been employed for the compression of video files. This is achieved by taking two consecutive frames of images in a video and comparing blocks of (typically) 8×8 pixels across these frames. Therefore if these two blocks are similar, the latter block is discarded [34]. This lossy compression typically results in significant compression and reduction in computational complexity which was particularly useful in the 1990s when expensive high capacity hard disks and expensive graphic processing units were required to play back the uncompressed videos [34].

3.2 *Audio Template Matching*

Template matching has also been applied in the area of audio signal processing for automatic speech recognition. Although automatic speech recognition is a well-established area of research, in its most primitive stage, only digits were used to determine the feasibility of speech recognition by means of template

matching [39]. To illustrate the concept of template matching, we first define $\mathbf{u}(n) = [u(n), \dots, u(n + L_u - 1)]^T$ as the input signal, where L_u is the length of $\mathbf{u}(n)$ and the set of pre-recorded data as $\mathbf{h}_m(n) = [h_m(n), \dots, h_m(n + L_h - 1)]^T$, where m is the index of the pre-defined digits and L_h is the length of $\mathbf{h}_m(n)$. By assuming $L_h = L_u$, the dissimilarities of the signals are determined by computing the correlation coefficients, with a time lag of 0

$$c(m) = \frac{1}{L_u \sigma_u \sigma_h} \sum_{i=0}^{L_u-1} (u(i) - \mu_u)(h_m(i) - \mu_h), \quad (3.1)$$

where μ_u and μ_h are the mean of $\mathbf{u}(n)$ and $\mathbf{h}_m(n)$ respectively, while σ_u and σ_h denote the standard deviation of $\mathbf{u}(n)$ and $\mathbf{h}_m(n)$ respectively.

In an experiment conducted and results presented in [39], a pre-defined data set of ten signals, the spoken digits of 0-9 from a single speaker, was collected. The subjects were then asked to repeat the digits to evaluate the accuracy of the recognition. The performance of the algorithm was evaluated using [39]

$$\text{results} = \begin{cases} \text{correct,} & c(m) = 1.0, \\ \text{wrong,} & \text{otherwise,} \end{cases} \quad (3.2)$$

and it was found that the recognition accuracy when tested by the same subject was, on the average, 98%.

Subsequently, research in speech recognition using template matching progressed from speaker-dependent single words to speaker-independent single words. The authors of [30] proposed a method which breaks up a received word into several segments. Template matching for speech recognition is achieved by building a phoneme template inventory comprising the forty-one phonemes of the American English [30]. In this phoneme template, a single frame reference template was used for each of the phonemes with the exception of the diphthongs and plosives, which were represented by two frames reference templates [30]. Each of the diphthongs were represented by two single-frame reference templates of vowels which most accurately represented the initial and final states of the diphthong [30]. The plosives, which were also represented by two-single frame reference templates, were repre-

sented by the silence reference frame followed by a reference frame representing the burst release [30].

Although not covered in detail in this thesis, it is worthwhile noting that significant contribution to the automatic speech recognition was made when the hidden Markov model was implemented [40], [41]. This enabled the recognition system to be able to recognize speaker-independent single words as well as a string of words that is speaker-independent. Even so, existing speech recognition systems still encountered several challenges, particularly in the area of continuous speech recognition. As such the authors of [31] proposed a technique that employs a template matching method integrated with the filler models of keyword spotting. This is achieved by assuming that the dialogue system is able to anticipate what the user intends to say based on the contextual information as well as the background knowledge of the dialogue system.

3.3 *Location Template Matching*

Location template matching (LTM) is a source localization algorithm that is able to localize in an environment where dispersion and multipath are present; a prominent phenomenon in wave propagation in solids. In this section, we review LTM by considering a touch interface (TI) system which can be represented as a single-input, single-output system as shown in Fig. 3.1. This TI system, consists of an interactive physical surface, a surface-mounted sensor, and a PC which processes signals received from the sensor. A human interacts with this interface by tapping on the interactive physical surface with a finger, which generates a signal $V(t)$. To estimate the location of the finger tap, we first define a coordinate frame as shown in Fig. 3.1 where the origin is located at the top-left corner of the plate. The coordinates of a finger tap is denoted by (x_m^t, y_m^t) , where m is the position index for $m = 1, \dots, N$. The tap excites vibrational waves which are picked up by the surface-mounted sensor at coordinates (x^s, y^s) . The sensor then converts the

Fig. 3.1: Diagram of TI with a matrix of predefined points.

vibrational waves into electrical signals which are subsequently processed by the source localization algorithm on the PC for the control of software applications.

Next, we assume that the vibration caused by a tap generates a signal received by the sensor as $\mathbf{u}(n) = [u(n), \dots, u(n + L_u - 1)]^T$ where L_u is the length of $\mathbf{u}(n)$. The aim of LTM is therefore to estimate the source location using $\mathbf{u}(n)$. To achieve this, LTM requires prior knowledge of the source signature at a known location. This signature can be obtained during the calibration phase where we define

$$\mathbf{h}_m(n) = [h_m(n), \dots, h_m(n + L_h - 1)]^T, \quad (3.3)$$

as the pre-recorded signal corresponding to a known source location at position index m . As described in Section 3.2, we assume that $L_h = L_u$ and that there is one such pre-recorded signal for each of the N number of pre-defined locations. LTM-based algorithms compute a similarity measure between $\mathbf{u}(n)$ with that of $\mathbf{h}_m(n)$. The estimated location is then given by the location corresponding to the pre-recorded signal which is closest to the received signal.

Fig. 3.2: Received signal of taps made at tap positions (41,20) cm and (42,20) cm.

To compute this similarity, the authors of [14] [42] proposed the use of the cross-correlation function, where the cross-correlation is computed between $\mathbf{u}(n)$ and $\mathbf{h}_m(n)$, for $m = 1, \dots, N$. This approach exploits the fact that the received signal is unique with respect to each of the source location due to variations in reflections and scattering of the wave along the source-sensor path. Defining $\mathbf{c}(m) = [c(1), \dots, c(N)]^T$ as a $N \times 1$ vector containing correlation coefficients between $\mathbf{u}(n)$ and $\mathbf{h}_m(n)$, each element in $\mathbf{c}(m)$ is given by (3.1) [43]. This cross-correlation based LTM (CC-LTM) algorithm estimates the location of a tap based on the estimated tap index $\hat{m}$ that corresponds to the maximum $c(m)$ over all position index $m = 1, \dots, N$, i.e.,

$$\hat{m} = \max_m \mathbf{c}(m). \quad (3.4)$$

The performance of CC-LTM [14] relies on the disparity between impulse responses from each tap location to each sensor. Due to this disparity, neighboring tap locations will generate similar yet different responses. To illustrate this, Fig. 3.2 shows the response of a tap made at (41,20) cm and (42,20) cm which, as seen in Fig. 3.1, are spaced 1 cm apart. It has been shown in [14] that location estimation using CC-LTM allows one to achieve a localization accuracy of ± 1.5 cm. However it is important to note that signals $\mathbf{u}(n)$ and $\mathbf{h}_m(n)$ are generated on the same surface, under environmental conditions that are not vastly different. In addition, as this CC-LTM algorithm uses the correlation-coefficient, it is essential

for $\mathbf{u}(n)$ and $\mathbf{h}_m(n)$ to be time-aligned. To do so, the received signals are first processed using a peak-detection algorithm. Using the peak as a reference position, the received signal, as seen in Fig. 3.2, is truncated from 5.3 ms before the peak to 15 ms after the peak.

3.4 Proposed all-pole model based algorithm for LTM

We now propose a LTM approach which employs an all-pole (AP) filter model, to perform source localization (AP-LTM). This approach is chosen over other popular sound recognition techniques such as [44], as other techniques usually model the way the human ear respond to sounds, and therefore might not be applicable for a touch interface. In addition, as described by the spectrum of (2.19) and as shown in Fig. 2.21, the magnitude response of a typical tap is characterized by dominant frequency peaks which can be modeled by the all-pole model. In view of this, the AP filter model is exploited to extract the dominant frequencies of the received signal by minimizing the squared error difference of the received signal $u(n)$ and predicted signal $\hat{u}(n)$. This prediction error is given by

$$\begin{aligned} e(n) &= u(n) - \hat{u}(n) \\ &= u(n) - \sum_{i=1}^I a(i)u(n-i), \end{aligned} \quad (3.5)$$

where I is the prediction order and $a(i)$ is the i^{th} filter coefficient. To illustrate the motivation behind this approach, $a(i)$ is evaluated for two received signals $\mathbf{u}(n)$ with the taps made at (41, 20) cm and (42, 20) cm and the sensor mounted at (5, 25) cm of an acrylic plate. In this experiment, we have used the Murata PKS1-4A1 sensor as described in Section 2. We then modeled the signal using an illustrative AP filter order example of $I = 3$. Fig. 3.3 shows the magnitude spectrum plot of the coefficients $a(i)$ for each of these positions when they are evaluated around the unit circle. The vertical lines are plotted by evaluating the pole positions on the unit circle. For comparison, the spectrum of the received

Fig. 3.3: Spectrum overlayed with AP-LTM of taps for $I = 3$, made at (a) (41, 20) cm and (b) (42, 20) cm.

signals evaluated using the fast-Fourier transform is illustrated in this figure.

As can be seen, the magnitude spectrum of $a(i)$ peaks at the dominant frequencies of the received signal. It is also interesting to note that the dominant frequency at approximately 2 kHz for both tap locations correspond to the resonant frequency of the Murata PKS1-4A1 sensor as discussed in Section 2.

To illustrate the important property where the all-pole model can extract dominant frequencies, we first define $\omega = 2\pi f/f_s$ with f_s being the sampling frequency and f being the frequency of the signal. The frequency-domain representation of (3.5) is given by

$$E(e^{j\omega}) = U(e^{j\omega}) \left[1 - \sum_{i=1}^I a(i)e^{-j\omega i} \right], \quad (3.6)$$

where $j = \sqrt{-1}$ and $a(i)$, $i = 1, \dots, I$ can be estimated from its past samples

$u(n - i)$. The power spectrum denoted as $|U(e^{j\omega})|^2$ of the system input $u(n)$ is given as

$$\sigma_e^2 = \left| U(e^{j\omega}) \right|^2 \left| 1 - \sum_{i=1}^I a(i) e^{-j\omega i} \right|^2, \quad (3.7)$$

with σ_e^2 being the variance of $e(n)$. This implies that the power spectrum of $u(n)$ is given as

$$\left| U(e^{j\omega}) \right|^2 = \frac{\sigma_e^2}{\left| 1 - a(1)e^{-j\omega} - a(2)e^{-j2\omega} - \dots - a(I)e^{-jI\omega} \right|^2}. \quad (3.8)$$

By factorizing the denominator of the function, we can obtain

$$\left| U(e^{j\omega}) \right|^2 = \frac{\sigma_e^2}{\left| (e^{j\omega} - e^{j\omega_1})(e^{j\omega} - e^{j\omega_2}) \dots (e^{j\omega} - e^{j\omega_I}) \right|^2}. \quad (3.9)$$

Hence, we see that $U(e^{j\omega})$ peaks at frequencies $\omega_1, \dots, \omega_I$. Since $\omega_i = 2\pi f_i/f_s$, the dominant frequencies can be obtained. As can be seen from Fig. 3.3, the dominant frequencies are identified by the AP filter coefficients $a(i)$ for both locations (41, 20) cm and (42, 20) cm. More importantly, it can be noted that these dominant frequencies differ from each other and thus this diversity is employed to estimate the source position.

To exploit this diversity, the sum of the squared differences between the filter coefficients of the received signals and those from our pre-recorded signals $\mathbf{h}_m(n)$ are computed. Defining $a^u(i)$ and $a_m^h(i)$ as the filter coefficients computed from the received signal $\mathbf{u}(n)$ and the pre-recorded signal $\mathbf{h}_m(n)$ respectively, this difference is obtained using

$$S_m = \sum_{i=1}^I \left(a^u(i) - a_m^h(i) \right)^2. \quad (3.10)$$

In a similar manner to the CC-LTM described in Section 3.3, the AP-LTM-based approach then estimates the location of the source given by

$$\hat{m} = \min_m S_m, \quad (3.11)$$

Tab. 3.1: Computation complexity of CC-LTM and AP-LTM.

Algorithm	No. of real multiplications	Illustration for $L_u = 1000$ $N = 25$ and $I = 3$
CC-LTM	$\mathcal{O}(N(3L_u))$	75 000
AP-LTM	$\mathcal{O}(I(\frac{I}{2} + L_u + N))$	3080

implying that the same tap location is expected to give the same or very close filter coefficients, such that the minimum distance will correspond to the tap location.

3.4.1 Complexity analysis

The derivation of the CC-LTM algorithm is divided into two main steps, namely the cross-correlation coefficient and the decision step. As has been shown in (3.1), the computation of the cross-correlation coefficient requires the computation of the mean and variances of both the input as well as the pre-recorded signals. In the decision step of this algorithm, the maximum value of the cross-correlated signal between the input and the whole database is required. This can be achieved by, for example, employing the quantum maximum finding algorithm [45] to solve (3.4). Excluding this maximum finding process, the total complexity for the cross-correlation based LTM algorithm, as shown in Table 3.1, is $\mathcal{O}(3NL_u)$ [46], where $\mathcal{O}$ is the big-O notation, of a quadratic order.

The derivation of the computational complexity for AP-LTM is comprised of the derivation of the AP filter coefficients and the decision step. The AP-LTM requires the computation of the autocorrelation vector and the solving of the Hermitian Toeplitz system using the Levinson Durbin algorithm. The complexity of Levinson Durbin algorithm is $\mathcal{O}(I^2)$ [47], [48]. In this illustrative example, we have used a filter corresponding to the order of $I = 3$. In the decision step of

Fig. 3.4: Complexity of the CC-LTM and AP-LTM.

this algorithm, the AP filter coefficients are first compared using (3.10) which comprises of $2I$ real multiplications. Similar to that for the CC-LTM, (3.11) can be solved using the quantum minimum finding algorithm [45]. Excluding this process, the total complexity for the AP-LTM algorithm, as shown in Table 3.1, is thus $\mathcal{O}(I(\frac{I}{2} + L_u + N))$ [46].

Fig. 3.4 shows the computational complexity of CC-LTM and the proposed AP-LTM. As seen in the figure, the complexity of CC-LTM and AP-LTM varies linearly with L_u . More importantly the complexity of the AP-LTM method is lower than that of the CC-LTM. As an illustrative example, as seen in Fig. 3.4 and Table 3.1, when the input signal is $L_u = 1000$ samples long and for the case of $N = 25$ tap locations, the proposed AP-LTM requires approximately 72 000 less real multiplications than the CC-LTM.

3.4.2 Experimental setup and performance measure

To verify the algorithm's performance, an experiment is performed. In this experiment as illustrated in Fig. 3.5, a Murata PKS1-4A1 accelerometer is mounted at (5, 25) cm on a piece of acrylic of dimension 60 cm $\times$ 30 cm $\times$ 0.7 cm. We

Fig. 3.5: Experiment setup Acrylic board with the accelerometer.

have positioned the sensor away from the axis of symmetry so as to avoid similarities of the spectrum brought about by this symmetry. Twenty-five tap points, each spaced 1 cm apart, were selected for the calibration phase where signals of known positions are first stored in a database.. The first tap point is positioned at (41, 20) cm on the acrylic sheet. As seen in Fig. 3.1, the sensor is denoted by a ‘o’ and each tap point is denoted by a ‘•’. The ‘×’ is used to denote the input $V(t)$ that was made on each of the tap points. The tap points are sequentially tapped and digitized using a PreSonus FireStudio; the signals were sampled at 24 kHz with 16-bits resolution. These signals are first saved into a database denoted by $\mathbf{h}_m$ defined in (3.3).

Fig. 3.6: Variation of accuracy for different AP filter orders.

We propose to evaluate the performance of the algorithms by means of a weighted function defined as [46], [49], [50]

$$J(Q) = \begin{cases} 1, & \hat{m} = m, \\ 0.5, & \hat{m} = m \pm 1 \text{ cm}, \\ 0.1, & \hat{m} = m \pm 1.5 \text{ cm}, \\ 0, & \text{otherwise,} \end{cases} \quad (3.12)$$

where $J(Q)$ is the score of the Q^{th} test tap for $Q = 1, \dots, Q'$, with Q' being the total number of tap points used for testing the algorithms. By averaging the score

$$J_T = \frac{1}{Q'} \sum_{Q=1}^{Q'} J(Q), \quad (3.13)$$

the accuracy of the algorithm is obtained.

In an earlier experimentation phase, the CC-LTM was found to yield an accuracy of about 68% [49], [50]. This is because the pre-recorded signals are in the first place very similar. Hence the CC-LTM is unable to distinguish minor amplitude changes in the frequency domain especially when the changes are not of the fundamental frequency. An example of the signals that CC-LTM is not able to distinguish are the taps made at (41, 20) cm and (42, 20) cm, which gave us the motivation to explore a frequency domain approach to the problem. Using the above weighting function, twenty taps were made by a human subject using a stylus at each of the twenty-five points in the pre-defined tap region, giving $Q' = 500$, as seen in Fig. 3.1.

Fig. 3.7: Spectrum overlayed with AP-LTM of taps, for $I = 12$ made at (a) (41, 20) cm and (b) (42, 20) cm.

We first evaluate how performance of the proposed AP-LTM algorithm varies with I . The result of the accuracy when different AP filter orders were used are plotted in Fig. 3.6, where it is seen that the ideal AP-filter order is $I = 3$ and that the accuracy reduces with increasing I . Such degradation is expected due to the effect of over-fitting which can be seen in Fig. 3.7, where an AP filter order of $I = 12$ is used. For this illustrative example of $I = 12$, we note that higher filter orders tend to model the curvature of the spectrum instead of modeling the peak frequencies. As such, unlike the case of $I = 3$ illustrated in Fig. 3.3, Fig. 3.7 shows that the AP filter model is unable to model the peak frequencies which, as a consequence, cannot differentiate between the two tap indices.

3.4.3 Experimental results of CC-LTM and AP-LTM

Using $I = 3$ and $Q' = 25$, the accuracy of using CC-LTM and AP-LTM are 68.00% and 96.48% respectively [49], [50]. From this, it is noted that even with reduced

complexity, the AP-LTM is able to localize approximately 25% more accurately than the cross-correlation approach. This improvement can be achieved since the AP-LTM models the peak frequencies of for the localization of the taps.

More importantly, the improvement in accuracy for the proposed AP-LTM approach is brought about by the different dominant frequencies that exist for different locations on the acrylic surface as illustrated by the magnitude spectrum plots as shown in Fig. 3.3. As a result of this inconsistency across different positions, we model the signal using an AP model and estimate the location of the source by comparing the similarities of the filter coefficients. This has the same effect, to a large extent, of comparing the pole locations around the unit circle.

To further illustrate the localization performance of the algorithms, we chose a tap location where AP-LTM performs worst compared to other test locations on the piece of acrylic. Similar to the above, twenty taps are made at each of the twenty-five locations and the taps made at location denoted by ‘●’ shown in Fig. 3.8 are used for localization. The estimated tap locations for the CC-LTM and AP-LTM are denoted by ‘*’ and ‘▽’, respectively. As seen in Fig. 3.8, of the twenty taps, CC-LTM is able to localize ten taps accurately, and another five taps being located within 1 cm from the actual location whereas the remaining five taps were localized more than 1.5 cm from the source. AP-LTM, on the other hand, localized eighteen taps while two taps were located 1.5 cm from the source. For this illustrative example, using (3.12), CC-LTM achieved 62.5% accuracy while AP-LTM is able to achieve a 91.0% accuracy. As seen in Fig. 3.8, AP-LTM is able to localize all twenty taps within 1.5 cm of the actual tap location, whereas CC-LTM can only localize fifteen taps within 1.5 cm from the actual tap location.

Although it is foreseeable that a subject will tap the surface differently across different trials, results presented in Fig. 3.8 show that AP-LTM outperforms CC-LTM. Throughout the experiment, it was found that while the frequency

Fig. 3.8: Illustration of estimated positions by CC-LTM and AP-LTM which are marked using ‘*’ and ‘▽’, respectively. The true source location is marked as ‘•’. The numbers beside each location denote the number of taps estimated at the corresponding location.

content of the signal changes across each tap, its peak frequencies do not vary significantly and as a result, AP-LTM which employs the peak frequency content to localize the taps is able to localize more accurately than CC-LTM.

Apart from the improved performance, one of the advantages of using AP-LTM to perform the localization is that the number of terms used for comparison is significantly reduced. In the case where the sampling frequency is 24 kHz, each of the recorded signals is $L_u = 1000$ samples long. The calculation of the cross-correlated coefficients require a 1000 samples long summation, which is not accounted for in this method of the computational complexity. However in the case of AP-LTM using the same sampling frequency, the tap signal is now represented by $I = 3$ coefficients.

3.5 Conclusion

In this chapter we first reviewed the use of template matching, namely in image, speech and music recognition. We next review the use of template matching for lo-

calization in solids, where the authors of [14] introduced the use of cross-correlation as a means to compare the similarity between the test signal and the pre-recorded signals. We then propose to model the dominant peaks of the taps using an AP filter model, and show that the AP-LTM is able to localize more accurately than CC-LTM, whilst ensuring a significantly lower computational complexity [49], [50]. One of the key advantages of LTM based algorithms is that LTM is able to accurately localize taps in materials where the speed of wave propagation is not constant. However its key disadvantage is that the algorithm requires that the region to be localized be known and trained [46]-[51].

4. LOCALIZATION USING TIME-DIFFERENCES-OF-ARRIVAL (TDOA)

4.1 *Review of the TDOA algorithm*

In the recent years, array signal processing, along with the advancement of hardware development and implementation, has become an active area of research. These advancements include applications in seismology [52], maritime [53], medical [54] as well as consumer oriented applications such as touch interfaces [55]. One of the main reasons why array signal processing techniques are popular for source localization is its ability to delay signals arriving at each sensor of the array so as to compensate for the time delay of arrival between each sensor for direction-of-arrival estimation.

One of the most popular approaches for source localization, using array signal processing, is by means of multilateration or sometimes known as hyperbolic positioning. Multilateration uses the TDOAs of the received signals between any sensor pairs for source localization. When a signal is transmitted to two spatially separated sensors, due to the difference in distances, the time taken for it to travel to the sensors will be different. The TDOA between the two sensors will therefore define a hyperbolic function with the sensors corresponding to the foci of the hyperbola [13]. Using several TDOA estimates and known sensor locations, the intersection of these hyperbolae thus corresponds to the location of the source [13] [56] [57] [58].

Fig. 4.1 illustrates how these hyperbolae can be used to localize a source. In this figure, the source is denoted by a ‘●’ and each of the three sensors, S1-S3, are denoted by ‘○’. The hyperbolic function defined by the TDOA between signals received by sensors S1 and S2 is represented by a solid line and that of S1-S3 and

Fig. 4.1: Hyperbolic functions defined by TDOA between three sensors.

S2-S3 are represented by dashed and dotted lines, respectively. As seen from the figure, in order to localize a source, at least two hyperbolae are required, which, as a result, three sensors are needed.

In practice, this intersection can be achieved by solving a set of non-linear equations relating to the source and known sensor positions [13]. As such, accurate TDOA estimates is required for accurate source localization.

4.2 TDOA Estimation

We first assume two channels of data received from a multichannel acoustic ADC,

$$\mathbf{u}_q(n) = [u_q(n), u_q(n+1), \dots, u_q(n+L_u-1)]^T, \quad (4.1)$$

$$\mathbf{u}_r(n) = [u_r(n), u_r(n+1), \dots, u_r(n+L_u-1)]^T, \quad (4.2)$$

where q and r are the channel indices. The cross-correlation, at lag n' , between these two signals are defined as

$$\gamma_{q,r}(n') \stackrel{\text{def}}{=} \sum_{n=-\infty}^{\infty} u_q(n)u_r(n'+n). \quad (4.3)$$

Defining

$$\Phi_{q,r} = [\dots, \gamma_{q,r}(-1), \gamma_{q,r}(0), \gamma_{q,r}(1), \gamma_{q,r}(2), \dots]^T, \quad (4.4)$$

the TDOA between these two signals u_q and u_r is then given by

$$\tau_{q,r} = \max_{n'} \{ \Phi_{q,r} \}. \quad (4.5)$$

As an illustrative example, consider two signals

$$\mathbf{u}_1(n) = [3 \ 7 \ 7 \ 2 \ 1 \ 5 \ 10 \ 3 \ 6 \ 2 \ 0 \ 0],$$

$$\mathbf{u}_2(n) = [0 \ 0 \ 3 \ 7 \ 7 \ 2 \ 1 \ 5 \ 10 \ 3 \ 6 \ 2],$$

which are shown in Fig. 4.2 and Fig. 4.3, where we observe that $\mathbf{u}_1(n)$ leads $\mathbf{u}_2(n)$ by two samples.

Fig. 4.2: Signal $\mathbf{u}_1(n)$

To find the lag between these two signals, we first cross-correlate the signals using (4.3) giving us

$$\Phi_{1,2} = \begin{bmatrix} 6 & 32 & 65 & 97 & 120 & 130 & 121 & 143 & 201 & 286 & 201 & 143 \\ 121 & 130 & 120 & 97 & 65 & 32 & 6 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

which is plotted as shown in Fig. 4.4.

As can be seen in Fig. 4.4, the peak of the cross-correlation occurs at lag $n' = -2$. This implies that the signal $\mathbf{u}_1(n)$ leads the signal $\mathbf{u}_2(n)$ by two samples,

Fig. 4.3: Signal $\mathbf{u}_2(n)$

Fig. 4.4: The cross correlation of $\mathbf{u}_1(n)$ and $\mathbf{u}_2(n)$

i.e.,

$$\tau_{12} = -2.$$

Computing the cross-correlation of two signals in the time domain is usually computationally expensive and as such, it can also be calculated in the frequency domain [59].

Given that there are Q sensors, the TDOAs of the received signals between sensor pairs are often tabulated in a $Q \times Q$ matrix [51]

$$\tau = \begin{bmatrix} \tau_{1,1} & \cdots & \tau_{1,Q} \\ \vdots & \ddots & \vdots \\ \tau_{Q,1} & \cdots & \tau_{Q,Q} \end{bmatrix}, \quad (4.6)$$

where the diagonal values of the TDOA matrix are zeros since there is no lag time when a channel's recorded signal is correlated with itself. In addition, the upper triangular elements of this matrix is the negative of the lower triangular elements.

4.3 Estimating Source Coordinates

The TDOA algorithm assumes that the speed of wave propagation is the same across all frequencies, which is not the case for wave propagation in solids. As such, signals received by the sensors need to be adapted prior to the TDOA algorithm. One such adaptation profiles the speed of wave propagation across frequencies [60]. Using this speed profile, the algorithm then determines the distance of the source with respect to a pre-defined reference sensor. Apart from being robust to dispersion, one of the key advantages of this adaptation is its relatively low computational complexity. In addition, with the speed profile of the plate, it is relatively simple to implement an algorithm that tracks an object that is dragged on the surface [60]. The key process of generating the profile of

the plate is however a delicate procedure, which requires an actuator that is capable of generating single tone frequencies in solids [60], as well as at least two Knowles BU-21771 accelerometers, which is significantly more expensive than the Murata PKS1-4A1 accelerometer. To prevent edge reflections from interfering with the profiling, profiling of the plate has to be achieved with the actuator and accelerometers mounted away from the edge of the plate [60].

In this section, we describe a TDOA estimation algorithm that can be used to estimate the source location in the presence of dispersion. As opposed to [60], this technique does not require the use of an actuator and employs time-frequency domain.

4.3.1 Formulation of Cost Function

In a practical situation, the sensors are mounted by the user and hence the sensor positions are assumed to be known. As such, we thus define the position of the q^{th} sensor as (x_q^s, y_q^s) . By defining the estimated tap position as $\hat{x}^t$ and $\hat{y}^t$, the difference in coordinates between the estimated tap and the q^{th} sensor is

$$\hat{\mathbf{d}}_q = \begin{bmatrix} \hat{x}^t \\ \hat{y}^t \end{bmatrix} - \begin{bmatrix} x_q^s \\ y_q^s \end{bmatrix}. \quad (4.7)$$

Defining $\tau_{q,r}$ as the TDOA determined by cross-correlation, as described in Section 4.2, c_s is the speed of propagation, the sum of square errors between $\tau_{q,r}$ and the TDOA obtained using the estimated positions $\hat{x}^t$ and $\hat{y}^t$ in (4.7) can be represented by

$$\sum_{q,r} \left[\frac{\sqrt{\hat{\mathbf{d}}_q^T \hat{\mathbf{d}}_q} - \sqrt{\hat{\mathbf{d}}_r^T \hat{\mathbf{d}}_r}}{c_s} - \tau_{q,r} \right]^2. \quad (4.8)$$

This cost function can be minimized iteratively to estimate the source location.

It is however important to note that this algorithm assumes that the speed of propagation c_s is constant throughout the whole area of wave propagation.

4.3.2 Levenberg-Marquardt Algorithm

We have chosen the Levenberg-Marquardt [61], [62] algorithm to minimize the squared error given by

$$\epsilon_{q,r}^2 = \left(\frac{\sqrt{\hat{\mathbf{d}}_q^T \hat{\mathbf{d}}_q} - \sqrt{\hat{\mathbf{d}}_r^T \hat{\mathbf{d}}_r}}{c_s} - \tau_{q,r} \right)^2. \quad (4.9)$$

The Levenberg-Marquardt algorithm is an iterative algorithm where the innovation is given by [61], [62]

$$\delta = \mathbf{J}^T \epsilon_{q,r} [(\mathbf{J}^T \mathbf{J} + \lambda \text{diag}(\mathbf{J}^T \mathbf{J}))^{-1}], \quad (4.10)$$

where $\mathbf{J}$ is the Jacobian matrix of $\epsilon_{q,r}$, δ is the increment to the estimated coordinates and λ is a non-negative damping factor.

As can be seen, the Levenberg-Marquardt algorithm is a least square minimization algorithm where (4.10) is to be minimized. The search direction of this minimization process is determined in the direction of gradient J . As such, the convergence is determined by the damping factor, λ . For example, if there is a reduction of $\epsilon_{q,r}^2$ in the previous iteration, λ is scaled down to reduce the final error value. Likewise, λ is scaled up when $\epsilon_{q,r}^2$ increases in the previous iteration. In the case where J is small, the damping factor λ is multiplied by a factor, $\text{diag}(\mathbf{J}^T \mathbf{J})$, which increases the rate of convergence.

The Levenberg-Marquardt is an optimization algorithm which converges to the nearest minimum in the vicinity of its initialization point. In this work, we initialize the Levenberg-Marquardt algorithm at the centre of the plate. To obtain the speed of wave propagation, c_s , a tap is made at a known location during the calibration process. Using the estimated value of $\tau_{q,r}$, (4.9) can then be used to solve for c_s .

4.3.3 Generating the Jacobian Matrix

Assuming that the estimated coordinates of the source are $[\hat{x}^t \ \hat{y}^t]^T$, the coordinates of sensor q are $[x_q^s \ y_q^s]^T$, and the coordinates of sensor r are $[x_r^s \ y_r^s]^T$, the coordinate difference between the estimated source and sensor q is given in (4.7) as

$$\hat{\mathbf{d}}_q = \begin{bmatrix} \hat{x}^t \\ \hat{y}^t \end{bmatrix} - \begin{bmatrix} x_q^s \\ y_q^s \end{bmatrix}. \quad (4.11)$$

The coordinate difference between estimated source and sensor r can be defined as

$$\hat{\mathbf{d}}_r = \begin{bmatrix} \hat{x}^t \\ \hat{y}^t \end{bmatrix} - \begin{bmatrix} x_r^s \\ y_r^s \end{bmatrix}. \quad (4.12)$$

From (4.11) and (4.12), the error between estimated TDOA and received TDOA is [51]

$$\epsilon_{q,r} = \frac{\sqrt{\hat{\mathbf{d}}_q^T \hat{\mathbf{d}}_q} - \sqrt{\hat{\mathbf{d}}_r^T \hat{\mathbf{d}}_r}}{c} - \tau_{q,r}. \quad (4.13)$$

whose partial differential is [51]

$$\begin{aligned} \frac{\partial^2 \epsilon_{q,r}}{\partial \mathbf{d}_q \partial \mathbf{d}_r} &= \frac{\frac{1}{2}(\hat{\mathbf{d}}_q^T \hat{\mathbf{d}}_q)^{-\frac{1}{2}} \times 2\hat{\mathbf{d}}_q - \frac{1}{2}(\hat{\mathbf{d}}_r^T \hat{\mathbf{d}}_r)^{-\frac{1}{2}} \times 2\hat{\mathbf{d}}_r}{c_s} \\ &= \frac{\hat{\mathbf{d}}_q(\hat{\mathbf{d}}_q^T \hat{\mathbf{d}}_q)^{-\frac{1}{2}} - \hat{\mathbf{d}}_r(\hat{\mathbf{d}}_r^T \hat{\mathbf{d}}_r)^{-\frac{1}{2}}}{c_s} \\ &= \begin{bmatrix} \frac{\partial \epsilon_{q,r}}{\partial x} \\ \frac{\partial \epsilon_{q,r}}{\partial y} \end{bmatrix}. \end{aligned} \quad (4.14)$$

The Jacobian matrix can then be defined as [51] [63]

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \epsilon_{1,Q}}{\partial x} & \frac{\partial \epsilon_{1,Q}}{\partial y} \\ \vdots & \vdots \\ \frac{\partial \epsilon_{Q,1}}{\partial x} & \frac{\partial \epsilon_{Q,1}}{\partial y} \end{bmatrix}. \quad (4.15)$$

Fig. 4.5: Photo of an Analog Device SHARC 21469 with its audio extender module.

4.4 Implementation of a Real-time TDOA Algorithm Robust to Dispersion

With the Levenberg-Marquardt algorithm in place to minimize the cost function, we are able to estimate the tap location. We now discuss the hardware used for this implementation. The real-time implementation of the TDOA prototype consists of eight Murata PKS1-4A1 accelerometers, an Analog Device SHARC 21469 digital signal processor (DSP), seen in Fig. 4.5, and a PC. Its block diagram can be seen in Fig. 4.6, where the accelerometers are used to pick up the vibration caused by an impact made by the user, the DSP, using the data collected by the accelerometer, then computes the estimated coordinates of the tap using the TDOA-based algorithm. Once this is computed, it then passes the estimated coordinates through its built-in RS232 port to the PC.

As seen in Section 4.3, the TDOA algorithm is only able to accurately localize a source when the speed of wave propagation in the region is constant. This is however not true for wave propagation on solids where dispersion is present. As such, a TDOA-based algorithm that is robust to dispersive effects has been proposed [64]. As this algorithm is pending patent, we provide a brief review

Fig. 4.6: Block diagram of initial prototype.

of the algorithm that has been implemented in the DSP board. In this real-time implementation of the algorithm, the signals received by each channel of the DSP board need to be processed before the Levenberg-Marquardt algorithm is employed to localize the source.

Fig. 4.7 shows a simplified block diagram of the TDOA estimation algorithm that is robust to dispersion [64]. As can be seen, the signals are first analyzed in the time-frequency domain, from which the frequency bins in the neighborhood of the Murata PKS1-4A1's resonant frequency are extracted [64]. Since the objective here is to accurately detect the instance the signal arrives at each of the sensors, the frequency bins close the resonant frequency of the accelerometer, which are most sensitive to impacts, are used in this detection. Once the signals from each of the frequency bins are extracted, they are processed using an onset detection algorithm. For each of the received signal, the onset detection algorithm utilizes the Hermitian angle between an arbitrarily defined reference vector and a vector constructed using the short-time Fourier transform. Each of these estimated signal onset is then used to accurately detect the time-of-arrival of the signal at the sensor. Finally, the Levenberg-Marquardt based TDOA algorithm, described in Section 4.3, is used to localize the source.

As described in Section 4.3 we note that for the TDOA algorithm to localize

Fig. 4.7: A simplified block diagram of the TDOA algorithm robust to dispersion implemented in the DSP.

a tap, the location of the sensors and the speed of wave-propagation is assumed to be known. As such it is necessary for a two-way communication between the PC and the DSP board. To ensure that there is no miscommunication, every data is transmitted along with a command. The transmitted data from the PC is mainly used to initialize the DSP board for detection, while the transmitted data from the DSP board to the PC is mainly to acknowledge the transmission as well as the transmission of the estimated tap location. Each data is packaged in a three-byte format, where the first byte is a command. A list of the commands are attached in the appendix of this thesis. To ease the initialization process of the touch interface, a graphical user interface (GUI) has been implemented and used to communicate between the PC and the DSP board. A screenshot of the GUI can be seen in Fig. 4.8. By using eight sensors to perform source localization, we are able to achieve up to an accuracy of 1.48 cm. More details pertaining to the accuracy using this algorithm can be found in Section 5.3.2 of this thesis.

4.5 Conclusion

In this chapter we first review the TDOA algorithm which is used for source localization. The algorithm assumes that the speed of wave propagating is known and that the wave propagates at a constant speed throughout the material. The key advantage of the TDOA-based algorithms over that of the LTM-based algorithms, described in Section 4.3, source localization on solids is that it requires only minimal calibration before it can be operated by a user [46], [49], [50], [51].

Fig. 4.8: Screenshot of GUI.

We next reviewed a TDOA algorithm which is robust to dispersion and we look at the hardware used to implement the dispersion-robust TDOA algorithm. Since a TDOA-based source localization algorithm requires more than three sensors, we note that a large number of wires are involved in the setup of this prototype. As such in the next chapter we reduce the amount of wires involved by means of a wireless approach.

5. IMPLEMENTATION OF THE WIRELESS TOUCH INTERFACE PROTOTYPE

5.1 *Wireless systems*

The prototype described in Section 4.4 is connected using wires and as such the setup is tedious. In this section, we explore various wireless solutions that can be adopted to enhance the features of the wire prototype described in Section 4.4. In addition, we also describe the challenges encountered during the selection of a suitable wireless protocol.

Wireless solutions are particularly useful in a number of scenarios. For example, having too many exposed wires give an appearance of a cluttered and messy place. To address this aesthetic issue, wires are often tied together or hidden in trunkings. However these solutions may not be particularly appealing in situations where a short setup or tear down time is preferred. In such situations, using current-day technology, one might adopt wireless solutions in such situations.

While wireless solutions provide such advantages, apart from power consumption, it too poses some challenges in terms of digital wireless communication. These challenges include frequency hogging, security and clock synchronization. In this thesis, our focus is on addressing issues pertaining to clock synchronization encountered by low-cost digital wireless sensors.

Apart from code division multiple access (CDMA), wideband CDMA (WCDMA) and global positioning system (GPS), high accuracy clock synchronization is an issue experienced by all low-cost digital wireless devices [65], [66], [67]. As such, most of the low-cost digital wireless sensors are deployed in situations

where timing accuracy is not crucial. This could include applications that track slow changes in the environment such as the temperature or humidity of a room, the chemical content of soil, tide monitoring. For such applications, the data could be sampled once every few minutes or hours. Since it is known that most of the IEEE 802.15.x protocols have a typical synchronization error of about 0.2 ms [65], [66], [67], [68], these low-cost wireless units can also be deployed in situations where the demand of the synchronization is lower such as monitoring the vibration of a bridge [69].

As with most digital systems, each wireless node has its own clock. Since it is not practical to power up all the nodes at exactly the same instance, it is expected that the clock counters have a certain degree of time difference; this difference is known as offset [65], [67]. Synchronization error therefore refers to the degree, in terms of time, of which the clocks differ after being corrected by the offset, i.e., a system having an inconsistent or unpredictable offset. Time-synchronization here refers to a low synchronization error [65], [67].

While time-synchronization is a crucial requirement of most applications across wireless networks, the degree of its accuracy is largely dependent on the intended application. For example in the case where one measures the temperature of a room once every five minutes, a synchronization error of 1 s is easily acceptable. However in the case of monitoring the vibration of a bridge [69], where the data could be sampled at 6 Hz, 1 s synchronization error equates to a somewhat acceptable 6 sample error. However, depending on application, if the bridge was to be monitored at 600 Hz, a 1 s synchronization error would equate to 600 sample error which may not be acceptable [69].

5.1.1 *Reduction of the Synchronization Error*

Details pertaining to the tolerance of the synchronization error, on its own, can be considered a study [67]. However several basic guidelines can be accepted. The two

main causes of synchronization error is drift and power-up time. Drift typically occurs with the age of the device as well as temperature differences between the various nodes [65], [66], [67], [68]. As drift normally causes a synchronization error that varies linearly with time, it can be easily resolved by linear estimation [66]. In this section we look at the cause of synchronization error that is due to time differences during the power-up process.

Synchronization error is in general avoided in wired systems. In the case of a multichannel ADC, where each input channel has its own ADC, all the ADCs' clock inputs are synchronized by the same system clock which, as a consequence, will trigger these ADCs to sample at the same instance. In so doing, synchronization error is avoided.

In inter-device wired communication protocols such as the universal asynchronous receiver/transmitter (UART), the communicating devices are considered as peers, implying that both devices are of equal status. As such, each device will operate using its own clock and synchronize the transmission. To achieve synchronized transmission, the transmission baud rate is several times slower than the slower system clock, and start and stop bits are used. As a result, this form of transmission protocol is often performed at low baud rates.

Alternative communication protocols such as the inter-integrated circuit (I2C) and the serial peripheral interface (SPI) use a master-slave concept, where the master synchronizes the communication by transmitting the clock throughout the transmission process so as to decide when the slave transmits and receives data.

To achieve highly accurate time-synchronization across the clocks of digital wireless systems, it is necessary to understand the basic structures of these IEEE transmission protocols. These low-cost transmission protocols such as the IEEE 802.11 (WiFi), the IEEE 802.15.4 and IEEE 802.15.1 (Bluetooth), all have at their

lowest layer a physical layer, and a medium access control (MAC) layer. On top of these two layers are the middle layers which are dependent on the protocols following which at the topmost layer, the application layer resides [67], [68], [70].

The physical layer determines radio characteristics, such as the signalling method, signal strength and signal sensitivity [70]. The MAC layer, on the other hand, packages the data prior to transmission, handles the beacon and serves as the management interface between the physical layer and the upper layers [70]. The application layer is where the users can access to program [65]-[70].

In terms of process priority, the upper layers, such as the application layer, have a lower priority than the lower layers such as the MAC and physical layers [65]-[70]. This implies that in terms of time determinism, the lower layers are more deterministic than the upper layers. As such to ensure low synchronization error, time-synchronization algorithms are often implemented in the MAC layer [65]-[70].

Time-synchronization algorithms typically take a timestamp of the instance just before the data is transmitted by the transmitter, and the instance the data is received by the receiver. Following which, the offset is computed and either of the clocks are corrected. Most of such algorithms however are tested to work well only in setups with two nodes [65], [66]. If the same program is downloaded to multiple nodes of a system, one of the problems that arises when the clocks of all the transmitting nodes are calibrated to the same reference is that they will all transmit at the same time, causing a jam in the network. As such most of these algorithms attach a timestamp to the transmitted data and then offset the time difference at the receiver end [65]-[70].

5.1.2 Tolerance in Time-Synchronization

For a real-time system that incorporates wireless sensors that processes the data before transmitting the computed information such as the time-of-arrival of the

signal at each sensor node, the tolerance of the synchronization error can be determined by first analyzing parameters including the sampling frequency and block length. In this thesis, we only consider the sampling rate. Therefore it is assumed that the data is sampled periodically as opposed to block sampling where a frame of data is sampled at regular time intervals. The sampling frequency is defined as the rate that corresponds to the time between two samples and depending on application, this parameter may determine the tolerance of the synchronization error.

As an illustrative example, we first assume that the data is sampled at $f_s = 100$ kHz giving a sampling period of $T_s = 10 \mu s$. We further assume that there are two wireless nodes, and the clock counter on Node A is $20 \mu s$ faster than Node B. The ADCs of the two nodes are connected together to a function generator generating an impulse which Node A picks up at its $60 \mu s$ mark while Node B would detect the impulse at $40 \mu s$. If the signals are correlated, a $20 \mu s$ offset should be observed, when in fact it should have been zero since these two ADCs are connected to the same function generator. Depending on applications, this intrinsic delay caused by the counter offset of $20 \mu s$ may be acceptable.

As with most real-time DSP, data is processed in a block-wised manner to ensure that real-time performance can be achieved. In cases where the synchronization error is larger than a block length, it would imply that programs without synchronization capability would not be able to detect the data. For example in the case of two wireless nodes where the clock of Node A is 15 ms faster than Node B and a block the block is of length of $T_B = 10$ ms. Assuming that a signal change is detected at the 19 ms mark on Node A, this signal change will be observed at the 9 ms on the second block index, whereas the same change will be seen at 4 ms on the first block of Node B. Thus programs without synchronization capability, which only processes data of the same block, may not be able to make sense of this information and treat it as a glitch in the system. Of course one may argue that

current processors have the capability to cope with the processing demands to compensate such errors, however such algorithms may be deemed as unnecessary loads for the processor.

Using the Analog Devices SHARC 21469 DSP board for the touch interface, we process a block of 2048 samples, sampled at 96 kHz. This corresponds to a sampling period $T_s = 10.42 \mu\text{s}$ and a block length $T_B = 21.33 \text{ ms}$. Since it is found that one sample of TDOA error can give rise to an error of about 2 cm [55], this error is aggravated when multiple sensors each have their own synchronization errors. Thus given that the IEEE 802.15.x protocols have a synchronization error (unpredictable offset) of 0.2 ms [67], [68], this corresponds to approximately twenty samples error per node with respect to a reference node. This implies that in the worst case scenario, there could be an error of approximately forty samples giving a large localization error of the finger tap.

In an ideal situation, a synchronization error that corresponds to less than half the sampling rate is preferred. This is because if the synchronization error is more than half the sampling rate, the sampled data could belong to the next sample. However in a practical situation, where the device might be processing other tasks it is preferred that the synchronization error is less than a tenth of the sampling rate. Therefore for our case, although a synchronization error of $5 \mu\text{s}$ is acceptable, a $1 \mu\text{s}$ or less synchronization error is preferred [69].

5.1.3 Other Parameters to Consider When Choosing a Transmission Protocol

Apart from time-synchronization, one key parameter to consider before adopting a transmission protocol is the transmission bandwidth $\mathcal{B}$. Assuming data is to be transmitted real-time,

$$\begin{aligned} \mathcal{B} &= f_s \times B_r \times \text{Channels} \\ &= 96 \text{ kHz} \times 16 \text{ bits} \times \text{Channels} \\ &= 1.536 \text{ Mbps/Channel}, \end{aligned}$$

where B_r is the sampling resolution of the data. Therefore if eight channels were to be used, the required bandwidth of the selected wireless protocol will need to be at least $\mathcal{B} = 12.288$ Mbps.

Assuming some amount of loss in in the data during transmission, only the IEEE 802.11g and IEEE 802.11n protocols with transmission bandwidths of 54 Mbps and 300 Mbps respectively, can cope with its data rate. Since the IEEE 802.11 protocols cannot provide real-time quality of service (QoS), it is not possible to transmit raw data. As a consequence, it is therefore necessary to compute the time-of-arrival for each sensor and transmit these time-of-arrivals rather than the raw data itself. In this context, the bandwidth would be reduced to

$$\begin{aligned}\mathcal{B} &= \frac{1}{T_B} \times [B_d + B_A + B_t] \times \text{Channels} \\ &= \frac{1}{21.3 \text{ ms}} \times [32 \text{ bits} + 32 \text{ bits} + 16 \text{ bits}] \times \text{Channels} \\ &= 3.75 \text{ kbps/Channel},\end{aligned}$$

where T_B is the sampling block length, B_t is the number of bits used to represent timestamp during which the data is processed, B_A is the number of bits required to represent the amplitude of the signal peak and B_d is the number of bits used to represent the timestamp where the onset is detected. Hence if eight channels were used the required bandwidth will be $\mathcal{B} = 30$ kbps.

With such a data rate requirement, even the low bandwidth IEEE 802.15.4 based protocols would be able to transmit and also deal with possible packet losses.

One other parameter to consider is the transmission distance. As most wireless protocols have a range of at least 10 m, this is not a crucial factor since the sensors are normally mounted on a HCI that is less than 10 m. However this parameter needs to be considered if the prototype is implemented using off-the-shelf Bluetooth devices that have a line-of-sight transmission range of only

10 m [68].

5.1.4 *The IEEE 802.15.4 protocol*

The IEEE 802.15.4 is a wireless protocol standard that is designed to fulfill the low-power, low-bandwidth, low-cost and real-time transmission requirements [71]. In the IEEE 802.15.4 protocol, every node is identified using a 64-bit medium access control (MAC) address, and requires a node to coordinate the network. This node is known as the coordinator and must be present in any IEEE 802.15.4 network [71]. The protocol also has end-devices which, as its name suggests, are the extremities of the wireless network [71]. In this protocol, only one coordinator can be present, and can be connected to as many as two hundred and fifty-six end-devices to form a star network. However, in cases where the end-devices are beyond the transmission distance of the coordinator, a router can be used to relay the message whilst transmitting its own data, forming a tree network [71].

As discussed in [71], it is important to note that a coordinator can have children but not parent. In addition, a router can have as children, other routers and end-devices, but only one parent that is either another router or a coordinator. Finally, an end-device cannot have any children and can only have a parent that is either a router or a coordinator.

Some of the communication rules of this protocol includes, a node being able to communicate with its immediate parent or child nodes. This implies that nodes from different branches cannot communicate with each other and hence, no end-device can communicate directly with another end-device which simplifies the programming. In our wireless prototype, we use a star network with one coordinator and multiple end-devices. As a proof-of-concept, we only use two end-devices.

As the IEEE 802.15.4 forms a very basic structure for wireless communication, many IEEE 802.15.4 compliant variants of the protocol have since stemmed

from it. This includes SimpliciTI by Texas Instruments, JenNet by Jennic, as well as the ZigBee and ZigBee Pro by the ZigBee Alliance. These protocols differ from each other in terms of their security coding, data packaging, channel hopping as well as transmission features.

5.1.5 Proposed Method

Since most of the advanced time-synchronization techniques using wireless transmission have an average synchronization error of 0.2 ms [67], [68], a possible solution would be to synchronize the nodes using wired protocol before being deployed wirelessly. While this simple solution is likely to achieve very low synchronization errors, it gives rise to other practical problems such as when one of the nodes is powered down, the nodes subsequently need to be re-synchronized. Moreover, this method can be said to defeat the whole purpose of a wireless solution.

In view of the above, we implemented a method that employs a wireless solution whilst working around the boundaries of the 0.2 ms synchronization limits. As a proof-of-concept, we illustrate this method as shown in Fig. 5.1, where four sensors are connected to a Analog Devices SHARC 21469 DSP board that, in turn, is connected to a wireless node which transmits and receives messages to and from the wireless node connected to the PC. In this method, each Analog Devices SHARC 21469 DSP board computes the estimated tap location based on the received signals from the sub-array of four sensors. The two estimated coordinates (from the two DSP boards) are then transmitted to the PC wirelessly after which the final estimated tap location is computed.

To ensure optimal processing, and to prevent any hierarchy issues, both the wireless Nodes A and B are setup as the end-device, and will only communicate with wireless Node C, which assumes the role of the coordinator. As such, only the coordinator is aware of the presence of the end-devices. Likewise, neither of the two Analog Devices SHARC 21469 DSP boards are aware of the presence of

Fig. 5.1: Block diagram of initial prototype.

the other board.

An advantage of the above method is that minimal changes need to be made to both the software in the Analog Devices SHARC 21469 DSP board as well as the GUI operating in the PC. In addition, in cases where more Analog Devices SHARC 21469 DSP boards are required to be deployed, they can be added into the system seamlessly. In the next section of this chapter, we describe the implementation of the above as well as issues encountered in the implementation of this wireless system.

5.2 Wireless Prototype

The Jennic JN5148 wireless evaluation kit, which uses the IEEE 802.15.4 wireless protocol, is chosen for the implementation of the wireless prototype as it meets the bandwidth as well as the time-synchronization requirements. Moreover, it is currently one of the fastest and most flexible IEEE 802.15.4 evaluation kits available in the market. We first present in Section 5.2.1 the flowchart of the algorithm programmed into the coordinator and in Section 5.2.2, we present the flowchart of the end-devices of the wireless systems. We will then discuss some of the issues encountered during the implementation and how each of them were overcome.

5.2.1 Flowchart of the Coordinator Node

Fig. 5.2 shows a photo of the Jennic JN1548 coordinator board which is connected to the PC. From the picture we can see that the coordinator board is operated using two AAA batteries, has an liquid crystal display (LCD) screen, four push buttons and four light emitting diodes (LEDs). The LCD screen, push buttons and LEDs are configurable.

Fig. A.1 shows the flowchart of the main function, where upon booting up of the baseband components, both the baseband and wireless systems are initialized. Once the initialization process is completed, the the event queues are then processed.

Fig. A.2 shows the process of baseband system initialization where the LEDs which are connected to the digital inputs-outputs (DIOs) of the coordinator are initialized. The process of initialization begins by performing a functional check by turning them on and off successively. Next the global variables and the wireless functions are initialized. Some of the wireless functions include of these functions include, a “I have no child” status, a “I am not ready to accept children” status. In addition, two circular buffers (one for the input and output)

Fig. 5.2: Picture of the coordinator of the wireless prototype.

are initialized for the wireless transmission. The serial port is next initialized after which the tick timer of the coordinator is initialized. This timer runs at 16 MHz, and is configured to interrupt the system every 10 ms. The process of the interrupt service routine (ISR) of the tick timer will be described later.

Fig. A.3 illustrates the process of the wireless system initialization, where the coordinator will scan within the frequency range 2400 - 2483.5 MHz in steps of 250 kHz to find the least used bandwidth [71]. The coordinator will then establish the least used bandwidth as its transmission channel, and to update its status to “I am ready to accept children” status. The flowchart of the interrupt service routine (ISR) of the timer can be seen in Fig. A.8.

The “Process Event Queues” function illustrated in the last block of Fig. A.1 is called upon by an infinite loop in the main function of the program. As seen in Fig. A.4, this function handles all incoming wireless events such as incoming data.

When the coordinator receives an association requests, as seen in Fig. A.5,

it first determines if the node making the request has the same application number, the same security codes (if any), and is using the short address format. Next, the coordinator determines if it can accept any more nodes. If it is able to accept additional nodes to join, it assigns a short address to the node and stores the address in its buffer. The coordinator then indicates on its second and third LED a successful association with the first and second end-device node respectively.

Fig. A.6 illustrates what happens when the coordinator receives an incoming data. As can be seen, the coordinator first ensures that the data originated from its child. If the data is from its child, it will copy the data to the buffer allocated to the child. The pre-process function, as will be explained in the next paragraph, will then process the data. Finally, the coordinator will send an acknowledgement message to the child, informing the child of the data size it received.

The pre-processing function of the coordinator, which is shown in the second penultimate block of Fig. A.6, is illustrated in detail in Fig. A.7. As can be seen, the function which systematically compiles and processes the data received from the children prior to transmitting them to the PC via its RS232 port. To do so, the pre-processing function first ensures that the first byte of the received data is that of an expected command and that the received data is of the correct packet size.

In the event that the received command is a sensor location command, 0x80 - 0x83, where the least significant four bits are the sensor index, the commands are modified according to the node the command was received from. As an illustrative example, if the command 0x81 was received from Node A, the output command would be unchanged. However if the received command 0x81 was received from Node B, the output command would be changed to 0x85. In this way, only the coordinator needs to manage the presence of multiple end-devices. This implies that only the coordinator is aware of the presence of multiple wireless nodes, and

that the software running in the end-device can be the same.

However, as shown in Fig A.7, in the event that the received command is an expected commands other than the sensor location command, the pre-processing function will check the state of the received data. For example, if there are two end-device nodes, if neither nodes have transmitted to the coordinator, the received data state will be *zero*. However once the coordinator receives data from Node A, it would change its received data state to *one*, reset the data timeout counter and waits for data to arrive from Node B. This state is reset to *zero* either a data timeout is encountered or every time a data is transmitted from the RS232 port. Given that the received data state is *one*, the coordinator will have to ensure that the newly received command is from a different node and is the same command as the command received from the previous node. This ensures that the coordinator only transmits to the PC when it receives the same command from two different nodes. In so doing, we prevent an unusual situation where a node sends the same command to the coordinator twice.

Once the coordinator receives the same command from two different nodes, it then verifies if the data is received within the expected time. For example, if a user taps twice on the touch interface with a 5 s interval, we will expect to receive two sets of data from each node, with an approximate 5 s interval. However if we only receive one set of data from Node A and another set of data from Node B which arrived approximately 5 s later, we can assume that the data are due to two instances of tapping. As such the timeout feature of this function will ensure that the two sets of data that are received beyond the timeout threshold correspond to two separate taps.

Given that the node, command and timing conditions are satisfied and that the received command is that of the “detected coordinates”, we then compute a new set of coordinates. Although it is possible to compute the new set of coordinates by averaging the detected coordinates. However for a more scientif-

ically based approach to this estimation, we propose, in Section 5.3, a weighted averaging method to estimate the new set of coordinates for tap location.

The timer initialization process shown in the penultimate block of Fig. A.2 is described in detail in Fig. A.8. As can be seen, the timer ISR is triggered every 10 ms by the tick timer of the coordinator and for every 1.28 s, the ISR toggles the first LED of the coordinator. The ISR then runs the wireless UART transmission function, which is described in Fig. A.9, and checks if any wirelessly received data is ready to be timed out. If received data is ready to be timed out, the data will be transmitted through the RS232 port to the PC. This ensures that in a situation where one of the end-devices malfunctions, the whole system will still be able to function.

The flowchart of the wireless UART transmission function, as seen in Fig. A.9, aims to transmit the data received in the first-in-first-out (FIFO) buffer as efficiently as possible. This is necessary since the wireless UART transmission function is triggered only once every 10 ms, during which the FIFO buffer may have only received part of the data. Since the IEEE 802.15.4 protocol transmits with a header and footer, that is typically twenty-five bytes long [72] and hence, compared to the three-byte data, the header and footer constitute a more significant portion of the total packet size. Since the Analog Devices SHARC 21469 DSP cannot do anything with incomplete packets of data, it makes sense to transmit wirelessly only when the expected data size is met in order not to congest the network.

The wireless UART transmission function, which is triggered only once every 10 ms, by the tick timer ISR, begins by reading from the FIFO circular buffer. In the event if there are any unread data in the buffer, the wireless UART transmission function checks if there it has already any stored commands in its transmission buffer. If there is a command in its transmission buffer, the new data will be treated as data bytes and will be stored in the transmission buffer.

If, however, there is new data and there is no command byte stored, the function will check that the first byte of the data is a command byte and assign the node(s) to which the data is to be sent.

As with the wirelessly received data, if the command received is a “sensor location” command, the indices of the commands will be modified accordingly so that the commands will match the sensors available on the Analog Devices SHARC 21469 DSP board connected to the end-device. Once the wireless UART transmission function has ensured that the packet of data has a command byte and the expected data size, it will transmit the data to the respective nodes.

5.2.2 Flowchart of the End-Device Node

Fig. 5.3 shows a photo of the Jennic JN5148 end-device board connected to the Analog Devices SHARC 21469 DSP board. From the picture we can see that, similar to the coordinator board, the end-device board is operated using two AAA batteries. However unlike the coordinator board, it does not have an LCD screen, and has only two push buttons and two LEDs, which, like the coordinator board, are configurable.

Figs. A.10 - A.14 show the flowcharts of the software operating in each of the end-devices. Inline with the IEEE 802.15.4 protocol, the end-devices do not communicate with each other, but rather, they operate independently. Fig. A.10 shows the flowchart of the main function, where upon booting up, similar to the coordinator, the baseband components are initialized. After the baseband systems have been initialized, the end-device will scan for the presence of a coordinator. Even if it doesn't find a coordinator the end-device will proceed to process the event queues, the flowchart of the event queues.

Similar to the coordinator, in the initialization flowchart of the baseband systems, as seen in Fig. A.11, the LEDs which are connected to the DIOs of the end-device are initialized. This is done by performing a functional check of turning

Fig. 5.3: Picture of the Jennic JN5148 end-device connected to the Analog Devices SHARC 21469 DSP board.

them on and off. After which, the global variables used in the program and wireless functions and statuses are initialized. Some of these statuses include the “I have no parent” status and the “I will not accept any children” status. The input and output buffers for the wireless transmission and the serial port of the end-device are then initialized. Lastly, the tick timer of the end-device is initialized. As with the coordinator, this timer runs at 16 MHz, and is configured to interrupt the system every 10 ms. The flowchart of the interrupt service routine (ISR) of the timer can be seen in Fig. A.15.

The “Process Event Queues” function of end-device, as seen in Fig. A.12, handles all incoming wireless events such as incoming data, and association requests. Their flowcharts can be seen in Figs. A.14 and Fig. A.13 respectively.

Fig. A.13 shows the process when the end-device receives an association

request. As can be seen, the end-device first checks if it is associated to a parent, which in this case is a coordinator. If it is not associated, it will scan the 2400 - 2483.5 MHz bandwidth until it finds a coordinator, and request to associate with the coordinator. Once it is associated to a coordinator, it will store the short address of the coordinator, and the short address assigned by the coordinator to it in a structure. Lastly, the end-device will update its association status and will also light up its second LED.

When the end-device receives an incoming data, as seen in Fig. A.14, the end-device first checks if the data came from its parent. If the data is from its parent, it will check if the data is an acknowledgement from the coordinator regarding the data it has previously sent. In this acknowledgement, the coordinator indicates to the end-device the package size it receives from the coordinator. If the size of the package differs from what had been sent, the end-device will check if the maximum retries have been reached. If it has not, the data will be resent. The purpose of the data resend feature in this function differs from the data resend feature incorporated in tick timer ISR.

In the case where the received data from the coordinator is not an acknowledgement packet from the coordinator, the end-device will assume that the received data is properly packaged by the coordinator and will transmit the data to the Analog Device SHARC 21469 DSP board via its RS232 port.

The flowchart of the end-device's tick timer ISR, is illustrated in Fig. A.15. Similar to the tick timer ISR of the coordinator, this ISR is triggered every 10ms by the tick timer of the end-device. Using a counter, the first LED of the end-device is toggled every 1.28 s after which, the ISR runs the wireless UART transmission function, whose flowchart can be seen in Fig. A.16.

The flowchart of the wireless UART transmission function for the end-device, as seen in Fig. A.16, begins by checking if the previously sent data has

been acknowledged by the coordinator. If the data has not been acknowledged, the end-device will check the timeout counter to see if enough time has passed before the last retry. On the other hand, if the data has been acknowledged, the end-device will reset the timeout counter and check if the maximum resend tries has been reached. Otherwise it will extract and resend the old data. This strategy is particularly useful as it allows the coordinator enough time to process its own data whilst handling new input data from the nodes.

Unlike the coordinator, a data resend component is required for the end-device. This is required since, as discovered in earlier experiments, the data sent by an end-device is often lost, which is not so for the coordinator. As such the data resend feature became a required component of the end-device. The maximum retry is another crucial component that ensures that the end-device does not choke the coordinator's wireless input buffer. In our wireless prototype, we have set the maximum retry as 5.

Assuming the previous data has been cleared, the wireless UART transmission function will, similar to the coordinator, check if there is any unread data in the serial FIFO buffer. It then ensures that the received UART data is packaged in the correct three-byte data format, where the first byte is an expected command, followed by the expected data sizes. As with the coordinator, this data packaging method, by ensuring that no incomplete data is sent, is very crucial in keeping the transmission channel is kept as clear as possible.

5.3 *Proposed weighted average algorithm*

The aim of this thesis is to convert surface of typically large areas into a touch interface. We therefore convert a 8 m $\times$ 3 m glass surface into a touch surface using eight sensors by deploying four of the sensors, S1-S4, to the top of the glass, and the other four, S5-S8, at the bottom of the glass. As seen in Fig. 5.4, the sensors are denoted by 'o'. In order to reduce the length of wires running along

Fig. 5.4: A $8\text{ m} \times 3\text{ m}$ glass surface with eight sensors labeled S1-S8.

the glass, we use the wireless nodes to group the sensors into sub-arrays. To avoid the ambiguity of the angle of arrival, which an uniform linear array suffers from, the grouping of S1, S2, S5 and S6 in the first sub-array and S3, S4, S7 and S8 in the second sub-array is adopted.

If a tap is made in the left half of the glass, the first sub-array would be able to localize the tap more accurately than the second sub-array. This is because localization within the boundary of the sensor array is often more accurate than outside its boundary [73]. A wave propagating from a tap made on solids to different parts of the solid is analogous to paths taken by the tap. As such, the paths taken by the wave propagating towards each of the sensors in the second sub-array is more similar than that of the paths of the wave propagating towards the sensors. As such, in this section, we propose a cross-correlation based algorithm that uses the signals received by each of the sensors within each sub-array to determine the reliability of the estimated coordinates.

5.3.1 Proposed algorithm to Determine the Source to Sensors Proximity

In this section we simulate a piece of $8\text{ m} \times 3\text{ m}$ glass, with eight sensors mounted on it, where the coordinates of each of the sensors are listed in Table 5.1. A tap is made at $(1.5, 2.0)\text{ m}$, and the signals received by each of the sensors can be seen in Figs. 5.5 and 5.6. Therefore, the tap is located nearer to the first sub-array.

Fig. 5.5: Signal received by S1, S2, S5 and S6 due to a tap at (1.5, 2.0) m.

Fig. 5.6: Signal received by S3, S4, S7 and S8 due to a tap at (1.5, 2.0) m.

Denoting $g = 1, 2$ as the sub-array index, the correlation coefficient of the signals received at the different sensors are given as

$$C_{q,r}(g) = \frac{1}{L_u \sigma_q \sigma_r} \sum_{i=1}^{L_u} (u_q(i) - \mu_q)(u_r(i) - \mu_r), \quad (5.1)$$

where q and r are the sensor indices within a particular sub-array, L_u is the length of the signal $\mathbf{u}_q(n)$ and $\mathbf{u}_r(n)$, μ_q and μ_r are the mean of $\mathbf{u}_q(n)$ and $\mathbf{u}_r(n)$ respectively, while σ_q and σ_r denote the standard deviation of $\mathbf{u}_q(n)$ and $\mathbf{u}_r(n)$ respectively. Operating (5.1) on our received signals shown in Fig. 5.5, the correlation matrix is given by

$$\mathbf{C}(1) = \begin{bmatrix} 1 & 0.0475 & 0.2118 & 0.0704 \\ 0.0475 & 1 & 0.0510 & -0.0970 \\ 0.2118 & 0.0510 & 1 & -0.0580 \\ 0.0704 & -0.0970 & -0.0580 & 1 \end{bmatrix} \quad (5.2)$$

for the first sub-array while for the second sub-array, we have

$$\mathbf{C}(2) = \begin{bmatrix} 1 & 0.3460 & 0.2651 & 0.1756 \\ 0.3460 & 1 & 0.2375 & 0.2772 \\ 0.2651 & 0.2375 & 1 & 0.4106 \\ 0.1756 & 0.2772 & 0.4106 & 1 \end{bmatrix}. \quad (5.3)$$

From the results, we see that the diagonals of both matrices are one, indicating that the signals are perfectly matched. This is so as the signal is correlated with itself. We can also see that the signals received by the sensors in both sub-arrays have a low correlation coefficient, unlike the results obtained in Section 3.3, implying that they are in general rather different from each other. This is expected since the sensors are located a distance from each other unlike the case where the points in LTM are 1 cm from each other. More importantly, we observe that the correlation of the received signals for the second sub-array, as seen in (5.2), varies significantly less than the first sub-array, seen in (5.3). This result is expected since the closer the source is to the sub-array, the lower the correlation, and the

Tab. 5.1: Sensor locations on 8 m $\times$ 3 m glass.

Sensor	Coordinates (m)
S1	(0.25, 3.75)
S2	(2.75, 3.75)
S3	(5.25, 3.75)
S4	(7.75, 3.75)
S5	(0.25, 0.25)
S6	(2.75, 0.25)
S7	(5.25, 0.25)
S8	(7.75, 0.25)

further the sub-array is to the source, the higher the correlation [18], [73].

In view of the above, the computation of the weights for each sub-array is given by

$$W(g) = 1 - \frac{\min_{q,r} C_{q,r}(g)}{\sum_g \min_{q,r} C_{q,r}(g)}, \quad (5.4)$$

where the minimum correlation coefficient for each sub-array is normalized against the sum of all the minimum correlation coefficient of all the sub-arrays. As such the sum of the weights,

$$\sum_g W(g) = 1.$$

Given that the estimated coordinates for each sub-array is given by $[\hat{x}_g^t \ \hat{y}_g^t]^T$, and the coordinates of the weighted average algorithm is given by

$$\begin{bmatrix} \hat{x}^t \\ \hat{y}^t \end{bmatrix} = \sum_g W(g) \begin{bmatrix} \hat{x}_g^t \\ \hat{y}_g^t \end{bmatrix}. \quad (5.5)$$

When the TDOA algorithm robust to dispersion was used to estimate the coordinates of the source using the data generated in the synthetic environment, both sensor groups accurately estimated the coordinates as the source. As such, regardless of the weights, the new estimated position is the same as the estimates from the two arrays, and the same as actual location.

Fig. 5.7: Sensors mounted on a glass plate with example of localization using different configurations.

5.3.2 Experimental Results Using the Proposed Algorithm

In this subsection we verify the accuracy of the proposed algorithm. Fig. 5.7 shows eight Murata PKS1-4A1 accelerometers, S1-S8, denoted by ‘o’, mounted on a piece of glass of dimension 108 cm $\times$ 60 cm. The sensors, S1-S8 are mounted at the coordinates (0, 0) cm, (46, 0) cm, (91, 0) cm, (91, 27) cm, (91, 52) cm, (46, 52) cm, (0, 52) cm, and (0, 27) cm respectively. Therefore, sensors S1-S4 form the first sub-array while sensors S5-S8 form the second sub-array.

Fig. 5.8 shows a photo of the setup of the wireless enabled touch interface where the objective is to convert an ordinary flat panel screen to a touch screen. This is achieved by mounting vibration sensors on a piece of glass which, in turn, is mounted on a flat panel screen using velcro. In this experiment, similar to what has been described in the previous paragraph, eight sensors are mounted on a piece of glass of dimensions $L_x = 108$ cm and $L_y = 60$ cm. Sensors S1-S4 are connected to the first Analog Devices SHARC 21469 DSP board, seen at the left of screen while sensors, while S5-S8 are connected to the second DSP board, seen at the right of the flat panel screen. Each DSP board is connected to a Jennic

Fig. 5.8: Picture of the setup of the wireless prototype.

JN5148 wireless node, configured as an end-device, which will transmit data to another Jennic JN5148 wireless node. This wireless node, as seen below the flat panel screen, is configured as the coordinator, and is connected to the computer. For real-time feedback of the localization, the glass plate which has been converted into a touch interface, the mouse cursor has been configured to respond as a left click at the estimated tap location using the GUI described in Section 4.4.

To verify the accuracy of the taps, we select fifty random points on the surface and estimate the source using TDOA localization algorithm described in Section 4.3. The proposed weighted average technique is compared against the following algorithms

1. a wired solution without sub-arrays, i.e., localization performed using all eight sensors,
2. localization performed using sensors S1-S4, i.e., on the first sub-array only,
3. localization performed using sensors S5-S8, i.e., on the second sub-array only.

We present in Fig. 5.7 an example of the results obtained using the proposed weighted average method where the '×' seen in the figure denote the position that the tap was made. The position estimated by the first sub-array is denoted by '*', while the '●' denote the position estimated by the second sub-array. The weighted average estimate of the two sensor groups is denoted with '◇', and the estimate obtained from using the eight sensors is denoted with '△'. From this figure, we note that by using eight sensors, we are able achieve a fairly high degree of localization accuracy. In addition, unlike the second sub-array, the first sub-array is unable to as accurately localize the source. This is expected since the source is within the boundary second sub-array. It is also observed that in this case, the proposed weighted average method localized the source most accurately.

Fig. 5.9: Results of estimation error using different configurations.

Fig. 5.9 shows a plot of the error when each of the four configuration were used to localize the taps. The horizontal axis denotes each of the method used to perform the localization, while the vertical axis represents the estimation error of the localization with respect to the actual location. This implies that the lower a point is, the better the estimation of the source, and a more compact clustering of the points of each method indicates that the method has a higher consistency, and a lower clustering of the points indicates a better overall estimation of the source. This implies that in an ideal situation will yield a plot with a lower and more compact clustering of the points.

Fig. 5.9, shows results obtained over the fifty tap locations for the different configurations. For each of the configuration, the estimation error for each tap is measured and tabulated. From this figure, we see that by using eight sensors to localize a tap we achieve low mean error of 1.48 cm with a standard deviation of 0.16 cm across the fifty taps. This implies that it has a very consistent performance and low error. The second and third columns of Fig. 5.9, show the result of using the first and second sub-array, respectively. The mean error for the first sub-array is found to be 4.04 cm while the mean error for the second sub-array is 4.13 cm. The standard deviation of these configurations are 0.42 cm and 0.43 cm for the

first and second sub-array respectively. These results imply that the first sub-array performed better than the second sub-array. This is so as in the fifty random taps made on the glass, more taps were made in the top and right regions of the glass than the lower and left regions.

In comparison, the mean error and standard deviation of the errors were found to be 2.09 cm and 0.24 cm respectively for the proposed weighted average algorithm. This implies that the proposed algorithm is able to localize more accurately and most consistently than the individual groups of four sensors, whose results can be seen in the second and third columns of Fig. 5.9. However when compared to the localization performed by the eight sensors, seen in the first column of Fig. 5.9, the proposed algorithm is modestly less accurate and less consistent. It is also seen that the proposed algorithm compared to the rest of the methods have three more outliers. This is caused by a scenario where the weights were placed on the wrong sub-array, and a scenario where both sub-arrays had wrongly estimated the source. While the proposed algorithm provides modest reduction in localization performance, its sub-array configuration provides more flexibility in terms of sensor placement as well as the ability to reduce the amount of wiring by placing a wireless unit near to each of the sub-array.

5.4 Conclusion

In order to ease the process of setting up the touch interface, we explore a low-cost wireless solution in this chapter. In doing so, we reduce the amount of wires involved. The aim of adopting a low-cost based wireless solution is to improve the aesthetics of the touch interface, specifically in reducing the amount of wires used, and to enhance the consumers' setup experience. To determine the method for the implementation of our wireless solution, we examined some of the issues encountered in low-cost wireless solutions. One of such issues include the synchronization error between two wireless nodes, in which the time difference between the sys-

tem clocks of two or more wireless nodes cannot be accounted for. We explained how this unaccounted time difference would affect the TDOA source localization algorithm. We then examined other parameters that need to be considered when determining the transmission protocol for our wireless prototype. Some of these parameters include transmission bandwidth and transmission distance. Based on these factors, we proposed a wireless implementation that overcomes the synchronization limits. The process flow for the coordinator and end-device nodes of the wireless solution have been presented in this chapter. In addition, we described some of the issues and their solutions encountered when implementing the solution. We also proposed, in this chapter, a weighted average algorithm that uses the coordinates estimated by two sub-arrays that are connected wirelessly to the PC which acts as a coordinator node. The proposed method has shown to be able to localize the source more accurate than the individual sub-arrays. Fig. 5.10 shows the wireless prototype being tested. As seen in the figure, there are the circular rings on the screen around around the finger. These rings are in fact a visualization tool which helps enhance the user experience of identifying the estimated location. The figure shows the wireless prototype successfully localizing the tap.

Fig. 5.10: Testing the wireless prototype.

6. SUMMARY AND FUTURE WORKS

6.1 *Summary*

In this thesis, we presented the hardware required for the acquisition of the data to facilitate research in source localization on solids. In addition, we reviewed two methods for source localization in solids, namely the location template matching (LTM) and the time-difference-of-arrival (TDOA) algorithms. Although the LTM requires only a single sensor and that it provides relatively high accuracy, it requires that the region to be localized to be known and trained. While the TDOA is a relatively less accurate source localizing algorithm it assumes the wave propagation throughout the material is constant. This implies that the algorithm works well on isotropic materials. Finally, we present a low-cost, low-power and low-bandwidth wireless source localization prototype.

In Chapter 3, we reviewed the CC-LTM algorithm, which cross-correlates the input tap with a set of pre-recorded signals. By corresponding the location of the index of the pre-recorded signals with the highest correlation coefficient, the CC-LTM algorithm is able to localize the input tap. We then proposed the AP-LTM algorithm which models the peak frequencies of the input and pre-recorded taps. We showed that by comparing the modeled peak frequencies of the input tap with modeled peak frequencies of the pre-recorded signals, the AP-LTM localizes more accurately than the CC-LTM. In so doing, the complexity cost of the localization is also reduced. The final result of the proposed AP-LTM was verified with real experimental data. The author, in a separate collaborative work, has proven that the time-frequency approach further enhances the distinct features of the tap, and thus allows more accurate localization [46]. Some of the advantages of using the LTM based algorithms include high localization accuracy in materials

where the speed of wave propagation is unknown or non-constant.

In Chapter 4, we also reviewed a source localization method using TDOA where we note that the TDOA algorithm requires a number of parameters such as the speed of wave propagation in the material, the location of sensors and accurate estimates of the instance the signal arrives at the individual sensors. However as dispersion causes traditional TDOA algorithms to be unable to localize taps on solids, thus the TDOA prototype was implemented using a patent pending TDOA algorithm which is robust to dispersion. One of the key advantages of the TDOA algorithm for source localization is that unlike the LTM based algorithms, it requires minimal calibration for source localization.

As shown in Chapter 5, it is crucial that the clocks of the individual ADCs connected to the sensors are synchronized. However, since current day low-cost digital wireless technologies are not able to achieve a low enough time-synchronization error, we did not attach a wireless transceiver to each sensor. Instead, we implement a low-cost wireless network, where each wireless sub-array is made up of four sensors connected to a Jennic JN5148 end-device node via an Analog Devices SHARC 21469 DSP board. In so doing we avoid the high time-synchronization accuracy requirement.

For each of the sub-array, the Analog Devices SHARC 21469 DSP board computes the estimated location of a tap, using the TDOA algorithm robust to dispersion [64]. The estimated coordinate is then transmitted to the Jennic JN5148 end-device node connected to the DSP board. Once the Jennic JN5148 end-device receives the expected data size, it then transmits the estimate to the coordinator. From the estimated coordinates computed using two end-device nodes, the Jennic JN5148 coordinator node computes the estimated tap location using the proposed weighted average algorithm.

The proposed weighted average algorithm determines the reliability of the

estimated location by means of cross-correlation computed in each sub-array. The proposed algorithm then assigns weights to each of the received estimated location from which compute a weighted average of the estimated location is computed.

In conclusion we show that both the LTM and TDOA based algorithms are able to accurately perform source localization, and both algorithms have their advantages and disadvantages. Lastly, we implement a TDOA-based wireless touch interface solution, and compared its capabilities against its wired counterpart.

6.2 Future Works

Future works into source localization in solids includes source localization for multi-touch as well as localization of various movements such as dragging. One possible algorithm that can be used to localize multi-touch on solids is the time-reversal signal processing (TRSP) algorithm. TRSP is an algorithm that is able to localize sources in non-homogeneous materials, where the number of sources TRSP is able to localize is less than the number of sensors [74]. One possible way of utilizing this algorithm could be to first create a virtual wave-propagation environment, which can be accomplished by modeling the material on which we want to perform source localization. Next, using the signals received by the surface-mounted sensors, we perform TRSP in the virtual environment to estimate the source. The key challenge in using this method for source localization is in the creation of an accurate virtual environment.

One of the disadvantages of the LTM-based algorithm is that it requires that every position that is wanted for localization be trained. As such, the training for a large area with many positions wanted for localization becomes a tedious process. To overcome this, a possible future work could be to reduce the number of training points required for LTM. One possible way of achieving this could be to plot the correlation coefficients with respect to their coordinates. Next by interpolating a plane using the correlation coefficients, we could possibly obtain a

peak which could possibly correspond to the coordinates of the source, which may not be the coordinates of any of the training points. In so doing, we could possibly increase the interval between the training points and could possibly localize the points in between the intervals.

In the research into current-day digital wireless transmission systems, it was found that low-cost transmission protocols are not deployed in areas where time-synchronization is required. This is so as they are not designed with low time-synchronization error. However, there are countless uses of wireless nodes with time-synchronization. Thus this area of research could be looked at in the future. One of the possible ways of accomplishing this could be to include, in the association phase of the end-device, a time synchronization procedure where the end-device synchronizes its system clock to the coordinator.

6.3 Publications Arising from this Thesis

- 1, X. Yap, A. W. H. Khong, and W.-S. Gan, "Localization of acoustic source on solids: A linear predictive coding based algorithm for location template matching," in *Proc. IEEE Int. Conf. Acoust. Speech and Signal Process.*, 2010, pp. 2490-2493.
- 2, K. Poletkin, X. Yap, and A. W. H. Khong, "A touch interface exploiting the use of vibration theories and infinite impulse response filter modeling based localization algorithm," in *Proc. IEEE Int. Conf. Multimedia and Expo*, 2010, pp. 286-291.
- 3, K. R. Arun, X. Yap, and A. W. H. Khong, "A touch interface exploiting time-frequency classification using Zak transform for source localization on solids" *IEEE Trans. Multimedia*, vol. 13, pp. 487-497, 2011.

BIBLIOGRAPHY

- [1] Adxl001: High performance wide bandwidth accelerometer. Analog Devices. [Online]. Available: <http://www.analog.com/en/mems-sensors/high-g-accelerometers/adxl001/products/product.html>
- [2] *PKS1-4A1 Shock Sensor*, Murata Manuf. Co., Ltd. [Online]. Available: <http://www.murata.com/catalog/p19e.pdf>
- [3] National instruments DAQ M series. [Online]. Available: <http://sine.ni.com/nips/cds/view/p/lang/en/nid/202803>
- [4] Profire 2626 - high-definition FireWire audio interface with octane preamp technology. M-Audio. [Online]. Available: http://www.m-audio.com/products/en_us/ProFire2626.html
- [5] Firestudio recording interface. Presonus. [Online]. Available: http://www.presonus.com/media/manuals/FireStudio_2626_v3_manual-web.pdf
- [6] Y. W. Kim, "A novel development of head-set type computer mouse using gyro sensors for the handicapped," in *Proc. IEEE 2nd Annual Int. EMB Special Topic Conf. Microtechnol. in Medicine & Biology*, 2002.
- [7] G. Shin and J. Chun, "Vision-based multimodal human computer interface based on parallel tracking of eye and hand motion," in *Proc. IEEE Int. Conf. Convergence Inform. Technol.*, 2007.
- [8] J. Kim, J. Park, H. Kim, and C. Lee, "HCI (human computer interaction) us-

- ing multi-touch tabletop display,” in *Proc. IEEE Pacific Rim Conf. Commun. Comput. and Signal Process.*, 2007.
- [9] W. Rolshofen, D. T. Pham, M. Yang, Z. B. Wang, Z. Ji, and M. Al-Kutubi, “New approaches in computer-human interaction with tangible acoustic interfaces,” in *Proc. Virtual Int. Conf.*, IPROMs, Ed., May 2005.
- [10] D. T. Pham, Z. Wang, Z. Ji, M. Yang, M. Al-Kutubi, and S. Catheline, “Acoustic pattern registration for a new type of human-computer interface,” in *Proc. Virtual Conf.*, 2005.
- [11] C. Bornand, A. Camurri, G. Castellano, S. Catheline, A. Crevoisier, E. B. Roesch, K. R. Scherer, and G. Volpe, “Usability evaluation and comparison of prototypes of tangible acoustic interfaces,” in *Proc. 2nd Int. Conf. on Enactive Interfaces*, Genoa, Italy, Nov 2005.
- [12] T. Kundu, S. Das, S. A. Martin, and K. V. Jata, “Locating point of impact in anisotropic fiber reinforced composite plates,” *Ultrasonic*, vol. 48, pp. 193–201, 2008.
- [13] A. W. H. Khong and M. Brookes, “The effect of calibration errors on source localization with microphone arrays,” in *Proc. IEEE Int. Conf. Acoust. Speech and Signal Process.*, 2007.
- [14] D. T. Pham, M. Al-Kutubi, Z. Ji, M. Yang, Z. Wang, and S. Catheline, “Tangible acoustic interface approaches,” in *Proc. Virtual Conf.*, 2005.
- [15] W. Rolshofen, P. Dietz, and G. Schaefer, “Neuartige berührbare schnittstellen durch die rückprojektion akustischer wellen,” in *Proc. Jahrestagung der Deutschen Gesellschaft fuer Akustik*, 2006.
- [16] W. Rolshofen, D. T. Pham, M. Yang, Z. B. Wang, Z. Ji, and M. Al-Kutubi, “New approaches in computer-human interaction with tangible acoustic interfaces,” in *Proc. Virtual Conf.*, 2005.

- [17] Knowles bu-21771 piezoelectric accelerometer datasheet. Knowles Electronics. [Online]. Available: http://www.knowles.com/search/prods_pdf/BU-21771-000.pdf
- [18] E. Ventsel and T. Krauthammer, *Thin plates and shells: theory, analysis, and applications*. Marcel Dekker, Inc., 2001.
- [19] D. Rovetta, A. Sarti, G. D. Sanctis, and M. Fabiani, "Modelling elastic wave propagation in thin plates," in *Proc. Eur. Signal Process. Conf.*, 2006.
- [20] J. Doyle, "Determining the contact force during the transverse impact of plates," *Experimental Mechanics*, pp. 68–72, 1987.
- [21] R. Pallas-Areny and J. G. Webster, *Sensors and signal conditioning*. John Wiley & Sons, Inc., 1991.
- [22] M. LLC. Overview of materials for acrylic. [Online]. Available: <http://www.scribd.com/doc/8637812/Acrylic-Material-Data-from-PARSGLASS>
- [23] E. T. Box. Elastic properties and young modulus of materials. [Online]. Available: http://www.engineeringtoolbox.com/Elastic-Properties-d_417.html
- [24] G. Blyth, *Audio Balancing Issues*, Soundcraft, Ed. White Papers, 2010.
- [25] R. Elliott. Direct injection box for recording and pa systems. [Online]. Available: <http://www.epanorama.net/circuits/direct-injection.html>
- [26] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoust., Speech and Signal Process.*, vol. 24, pp. 320–327, 1976.
- [27] H. Deng and X. Song, "License plate characters segmentation using projection and template matching," in *Proc. Int. Conf. Inform. Technol. and Comput. Science*, 2009.

- [28] B. Zhang, Y. Jia, J. Zhao, D. Yu, and Q. Zhou, "Fingerprint template matching algorithm based on Daubechies wavelet," in *Proc. Int. Conf. Commun. Software and Networks*, 2009.
- [29] J. Leng, J. Ni, and J. Luo, "A novel fingerprint bifurcation extraction algorithm based on neural network template matching," in *Proc. Int. Symposium on Comput. Science and Computational Technol.*, 2008.
- [30] T. Svendsen and F. Soong, "On the automatic segmentation of speech signals," in *Proc. IEEE Int. Conf. Acoust. Speech and Signal Process.*, 1987.
- [31] G. Zhang, H. Sun, F. Zheng, and W. Wu, "Robust speech recognition directed by extended template matching in dialogue system," in *Proc. World Congress on Intell. Contr. and Automat.*, 2004.
- [32] K. Yoshii, M. Goto, and H. G. Okuno, "Drum sound recognition for polyphonic audio signals by adaptation and matching of spectrogram templates with harmonic structure suppression," *IEEE Trans. Audio, Speech and Language Process.*, vol. 15, pp. 333–345, 2007.
- [33] G. Vanderbrug and A. Rosenfeld, "Two-stage template matching," *IEEE Trans. Comput.*, vol. c-26, pp. 384–393, 1977.
- [34] M. Choi, N. Galatsanos, and D. Schonfeld, "On the relation of image restoration and template matching: Application to block-matching motion estimation," in *Proc. IEEE Int. Conf. Acoust. Speech and Signal Process.*, 1996.
- [35] S. Mattoccia, F. Tombari, and L. D. Stefano, "Fast full-search equivalent template matching by enhanced bounded correlation," *Trans. Image Process.*, vol. 17, pp. 528–538, 2008.
- [36] F. Jurie and M. Dhome, "Real time 3D template matching," in *Proc. Comput. Vision and Pattern Recognition*, 2001.

- [37] H.-J. Cho and T.-H. Park, "Template matching method for smd inspection using discrete wavelet transform," in *Proc. Semi Conductor Industry Eng.*, 2008.
- [38] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Trans. Commun.*, vol. COM-31, pp. 532–540, 1983.
- [39] K. Davies, R. Biddulph, and S. Balashek, "Automatic speech recognition of spoken digits," *J. Acoust. Soc. Am.*, vol. 24, pp. 637 – 642, 1952.
- [40] R. Rose and D. Paul, "A hidden Markov model based keyword recognition system," in *Proc. Int. Conf. Acoust. Speech and Signal Process.*, 1990.
- [41] J. Wilpon, L. Rabiner, C.-H. Lee, and E. Goldman, "Automatic recognition of keywords in unconstrained speech using hidden Markov models," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 38, pp. 1870 – 1878, 1990.
- [42] G. Ribay, S. Catheline, D. Clorennec, R. K. Ing, N. Quieffin, and M. Fink, "Acoustic impact localization in plates: properties and stability to temperature variation," *IEEE Trans. Ultrason., Ferroelect., Freq. Contr.*, vol. 54, pp. 378–385, 2007.
- [43] R. K. Ing, N. Quieffin, S. Catheline, and M. Fink, "In solid localization of finger impacts using acoustic time-reversal process," *Applied Physics Letters*, vol. 87, pp. 1–3, 2005.
- [44] S. Chu, S. Narayanan, and C. C. J. Kuo, "Environmental sound recognition with time-frequency audio features," *IEEE Trans. Audio, Speech and Language Process.*, vol. 17, pp. 1142–1158, 2009.
- [45] M. W. Coffey and Z. Prezkuta, "A quantum algorithm for finding the modal value," *Quantum Inform. Process.*, vol. 7, pp. 51–54, Jan 2008.

- [46] K. R. Arun, X. Yap, and A. W. H. Khong, "A touch interface exploiting time-frequency classification using Zak transform for source localization on solids," *IEEE Trans. Multimedia*, vol. 13, pp. 487–497, 2011.
- [47] D. Sundararajan, M. Ahmad, and M. Swamy, "Fast computation of the discrete Fourier transform of real data," *IEEE Trans. Signal Process.*, vol. 45, pp. 2010–2022, 1997.
- [48] C. Cheng and K. Parhi, "Low-cost fast VLSI algorithm for discrete Fourier transform," *IEEE Trans. Circuits Syst.*, vol. 54, pp. 791–806, 2007.
- [49] X. Yap, A. W. H. Khong, and W.-S. Gan, "Localization of acoustic source on solids: A linear predictive coding based algorithm for location template matching," in *Proc. IEEE Int. Conf. Acoust. Speech and Signal Process.*, 2010, pp. 2490–2493.
- [50] K. Poletkin, X. Yap, and A. W. H. Khong, "A touch interface exploiting the use of vibration theories and infinite impulse response filter modeling based localization algorithm," in *Proc. IEEE Int. Conf. Multimedia and Expo*, 2010, pp. 286–291.
- [51] A. Sulaiman, K. Poletkin, and A. W. H. Khong, "Source localization in the presence of dispersion for next generation touch interface," in *Proc. IEEE Cyberworlds*, 2010.
- [52] L. Sha and B. L. F. Daku, "A performance comparison of three localization algorithms," in *Proc. IEEE Canadian Conf. Elect. Comp. Eng.*, May 2005, pp. 328–331.
- [53] A. Silva, S. Jesus, and J. Gomes, "Probe timing optimization for time-reversal underwater communications," in *Proc. IEEE Oceans*, 2011.
- [54] B. N. Cuffin, "EEG dipole source localisation," *IEEE Eng. Med. Biol. Mag.*, vol. 17, pp. 118–122, 1998.

- [55] D. Pham, Z. Ji, M. Yang, Z. Wang, and M. Al-Kutubi, "A novel human-computer interface based on passive acoustic localisation," in *Proc. 12th Int. Conf. Human-Computer Interface*, 2007, pp. 901–909.
- [56] P. Polotti, M. Sampietro, A. Sarti, S. Tubaro, and A. Crevoisier, "Acoustic localization of tactile interactions for the development of novel tangible interfaces," in *Proc. of the 8th Int. Conf. on Digital Audio Effects*, 2005.
- [57] T. K. C. Lo and H. C. So, "Theoretical study of impact localisation limits in tangible acoustic interface applications," *IET Electron. Lett.*, vol. 45, pp. 856–858, 2009.
- [58] Y. T. Chan, T. K. C. Lo, and H. C. So, "Passive range-difference estimation in a dispersive medium," *IEEE Trans. Signal Process.*, vol. 58, pp. 1433–1439, 2010.
- [59] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing*, , ISBN 0-201-36186-8, page 4. Prentice Hall, 2000.
- [60] M. Fabiani, "Development of a tangible human-machine interface exploiting in-solid vibrational signals acquired by multiple sensors," Master's thesis, Politecnico di Milano, 2006.
- [61] K. Levenberg, "A method for the solution of certain problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, pp. 164–168, 1944.
- [62] D. Marquardt, "An algorithm for least squares estimation on nonlinear parameters," *SIAM J. on applied mathematics*, vol. 11, pp. 431–441, 1963.
- [63] G. D. Sanctis, D. Rovetta, A. Sarti, G. Scarparo, and S. Tubaro, "Localization of tactile interactions through TDOA analysis: Geometric vs. inversion-based method," in *Proc. Eur. Signal Process. Conf.*, 2006.
- [64] A. W. H. Khong, V. G. Reju, A. Sulaiman, K. Arun, and W. S. Gan, US Patent US61/507,717, Jul., 2011 (Pending).

- [65] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: A survey," *IEEE Trans. on Network*, vol. 18, pp. 45–50, 2004.
- [66] R. Casas, H. J. Gracia, A. Marco, and J. L. Falco, "Synchronization in wireless sensor networks using Bluetooth," in *Proc. IEEE 3rd Int. Workshop on Intell. Solution in Embedded Syst.*, 2005.
- [67] P. Ferrari, A. Flammini, D. Marioli, S. Rinaldi, and E. Sisinni, "On the implementation and performance assessment of a wirelessHART distributed packet analyser," *IEEE Trans. Instrum. and Meas.*, vol. 59, pp. 1342–1352, 2010.
- [68] A. Marco, R. Casas, J. L. S. Ramos, V. Coarasa, A. Asensio, and M. S. Obaidat, "Synchronization of multihop wireless sensor networks at the application layer," *IEEE Trans. Wireless Commun.*, vol. 18, pp. 82–88, 2011.
- [69] C. Townsend and S. Arms, *Wireless sensor networks: principles and application*. MicroStrain, Inc., 2004, ch. 22 on Sensor Technol. Handbook by Jon S. Wilson, pp. 575–587.
- [70] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, and M. Nixon, "WirelessHART: Applying wireless technology in real-time industrial process control," in *Proc. IEEE Real-Time and Embedded Technol. and Applicat. Symposium*, 2008.
- [71] J. Gutierrez, M. Naeve, E. Callaway, M. Bourgeois, V. Mitter, and B. Heile, "IEEE 802.15.4: a developing standard for low-power low-cost wireless personal area networks," *IEEE Network*, vol. 15, pp. 12–19, 2001.
- [72] Application note: JN-AN-1035: Calculating 802.15.4 data rates. NXP Semiconductors. [Online]. Available: http://www.jennic.com/files/support_files/JN-AN-1035_Calculating_802-15-4_Data_Rates-1v0.pdf

- [73] B. Yang and J. Scheuing, "A theoretical analysis of 2D sensor arrays for TDOA based localization," in *Proc. IEEE Int. Conf. Acoust. Speech and Signal Process.*, 2006.
- [74] D.-Q. Nguyen, X. Yap, W.-S. Gan, and Y.-K. Chong, "Selective focusing with numerical technique of time reversal operator decomposition," in *Proc. Sensor Array and Multichannel*, 2008.

APPENDIX

A. FLOWCHARTS OF THE FUNCTIONS USED IN THE WIRELESS PROTOTYPE

Fig. A.1: Flowchart of the main function of the coordinator of the wireless prototype.

Baseband System Initialization

Fig. A.2: Flowchart of the baseband system initialization of the coordinator.

Wireless System Initialization

Fig. A.3: Wireless system initialization flowchart of the coordinator.

Process Event Queues (Wireless)

Fig. A.4: Flowchart of event queues of the coordinator.

Handle Association Requests

Fig. A.5: Flowchart of the association request function of the coordinator.

Process Incoming Data (Wireless)

Fig. A.6: Flowchart how incoming wireless data is processed by the coordinator.

Pre-Process

Fig. A.7: Flowchart of the pre-processing function of the coordinator.

Fig. A.8: Flowchart of the timer ISR of the coordinator.

Fig. A.9: Wireless UART transmission flowchart for the coordinator.

Fig. A.10: Flowchart of the main function of the end-device.

Baseband System Initialization

Fig. A.11: Flowchart of the baseband system initialization of the end-device.

Process Event Queues (Wireless)

Fig. A.12: Flowchart of event queues of the end-device.

Handle Association Requests

Fig. A.13: Flowchart of the association request function of the end-device.

Process Incoming Data (Wireless)

Fig. A.14: Flowchart of the Process incoming data function of the end-device.

Fig. A.15: Flowchart of the timer ISR of the end-device.

WUART TX Data

Fig. A.16: Flowchart of the wireless UART transmission of the end-device.

B. UART PROTOCOL FOR PC AND SHARC (DSP)

There are two types of bytes transmitted in this PC to DSP communication.

- Command byte: Instructs SHARC with a specific command
 - E.g. Start detection, Stop detection, sensor coordinates
 - MSB of command byte is 1
- Data bytes : Carries data
 - E.g. x and y coordinates
 - 0x01 (co-ordinate value 1)

Commands from PC to SHARC

Command	Data1	Data2	Description
0x80	[x]	[y]	Coordinates of sensor 0, [x][y]
0x81	[x]	[y]	Coordinates of sensor 1, [x][y]
0x82	[x]	[y]	Coordinates of sensor 2, [x][y]
0x83	[x]	[y]	Coordinates of sensor 3, [x][y]
0x84	[x]	[y]	Coordinates of sensor 4, [x][y]
0x85	[x]	[y]	Coordinates of sensor 5, [x][y]
0x86	[x]	[y]	Coordinates of sensor 6, [x][y]
0x87	[x]	[y]	Coordinates of sensor 7, [x][y]
0xB0			Start detection
0xC0			Stop detection
0xF0			Keep alive: checks whether other DSP is listening
0x90			Silence calibrate for every channel
0xD0	[MSB]	[LSB]	Adjust speed 16-bit [MSB] [LSB]

Commands from SHARC to PC

Command	Data1	Data2	Description
0x80	[x]	[y]	Acknowledgement of coordinates of sensor 0, [x][y]
0x81	[x]	[y]	Acknowledgement of coordinates of sensor 1, [x][y]
0x82	[x]	[y]	Acknowledgement of coordinates of sensor 2, [x][y]
0x83	[x]	[y]	Acknowledgement of coordinates of sensor 3, [x][y]
0x84	[x]	[y]	Acknowledgement of coordinates of sensor 4, [x][y]
0x85	[x]	[y]	Acknowledgement of coordinates of sensor 5, [x][y]
0x86	[x]	[y]	Acknowledgement of coordinates of sensor 6, [x][y]
0x87	[x]	[y]	Acknowledgement of coordinates of sensor 7, [x][y]
0xB1			Start detection
0xC1			Stop detection
0xF0			Keep alive: checks whether other DSP is listening
0x91			Silence calibrate for every channel
0xD1	[MSB]	[LSB]	Adjust speed 16-bit [MSB] [LSB]
0xE0	[x]	[y]	Coordinates detected [x] [y] (unsigned integer values in cm)