

# SONAR Imaging using Compressive Sensing

A Thesis

Presented in Partial Fulfillment of the Requirements for  
the Degree Bachelor of Science with Honors Research Distinction in  
Electrical and Computer Engineering

By

Taylor Williams

\* \* \* \* \*

Electrical and Computer Engineering  
The Ohio State University

2011

Examination Committee:

Prof. Lee Potter, Adviser

Prof. Phillip Schniter

© Copyright by  
Taylor Williams  
2011

## ABSTRACT

Sonar imaging is a technique that can locate objects in a scene from their acoustic reflections. It utilizes multiple measurements from different angles to create a 2D image. Conventional imaging algorithms such as backprojection (BP) require a large number of measurements to produce accurate images. Theory indicates that an alternative approach called compressive sensing (CS) can create accurate images using just a fraction of the measurements. CS requires that the scene is compressible and that each individual measurement provides information about the scene, not just a particular location. The sonar imaging experiment must be designed to meet these requirements. The purpose of this study is to show that the compressive sensing framework can be used in sonar imaging with results comparable to conventional methods. A sonar test stand was built that can measure acoustic reflections from a scene. A pseudo-random noise (PN) code was used for transmission. Four speakers and 16 microphones were mounted randomly perturbed from a circle surrounding the scene. Software was developed to create an image from this data through FBP. Also, a CS algorithm was developed to reconstruct an image from limited measurements. This algorithm uses a random subset of samples from each measurement. Initial results show that FBP can effectively be used to image a scene using acoustic waves. The CS algorithm yields a similar quality image using less than 10% of the measurements. These results show that CS can be used in sonar imaging to greatly reduce the

number of measurements that need to be collected. The findings are directly portable to radar imaging, a field with a high level of research for both military and civilian uses.



This is dedicated to my family.

## ACKNOWLEDGMENTS

I wish to thank my adviser, Lee Potter, for his support during the entire year. From the very beginning, he has been willing to devote time and energy to the project and it has made all the difference.

I thank Jason Parker and Josh Ash for their willingness to help and provide pointers in the right direction.

I would also like to thank Danish Siddiqui, Laura Tufts and David LaVergne, all collaborators in different ways on the project. David patiently brought me up to speed on audio hardware while Laura, Danish & I all shared a common base of signal processing and research in our individual efforts. It was great having other students to work together with.

This research was funded through the Air Force Research Lab in Dayton, OH. This funding was a great support to the research and allowed me to build a test stand and explore acoustic imaging.

# TABLE OF CONTENTS

	Page
Abstract . . . . .	ii
Dedication . . . . .	iv
Acknowledgments . . . . .	v
List of Tables . . . . .	ix
List of Figures . . . . .	x
Chapters:	
1. Introduction . . . . .	1
2. Background . . . . .	2
2.1 SONAR & Related Imaging Modes . . . . .	2
2.2 Problem Statement . . . . .	3
2.3 Imaging Techniques . . . . .	4
2.3.1 Backprojection Imaging . . . . .	4
2.3.2 Compressive Sensing Imaging . . . . .	5
2.4 Objectives & Design Goals . . . . .	6
2.5 Economic & Social Considerations . . . . .	6
3. Hardware & Test Stand Design . . . . .	8
3.1 Data Acquisition . . . . .	8
3.2 Speakers & Microphones . . . . .	9
3.2.1 Hardware Operating Range . . . . .	11
3.2.2 System Frequency Testing . . . . .	12

3.3	Acoustic Imaging Test Stand . . . . .	14
3.3.1	Equipment Mounting . . . . .	14
3.4	Data Collection using the Acoustic Imaging Test Stand . . . . .	15
4.	Waveform Design & Echo Processing . . . . .	17
4.1	Design Considerations . . . . .	17
4.2	PN Code: Description & Correlation Properties . . . . .	18
4.3	Echo Processing . . . . .	19
4.3.1	Echoes from Bistatic Geometries . . . . .	20
4.3.2	Range Profile Signal Processing . . . . .	21
4.3.3	Background Subtraction . . . . .	22
5.	Backprojection Imaging . . . . .	25
5.1	Bistatic Projections . . . . .	25
5.2	Backprojection Algorithm . . . . .	26
5.3	Backprojection Inefficiencies . . . . .	27
5.3.1	Point Spread Function . . . . .	28
5.3.2	Channel Imbalances . . . . .	29
5.3.3	Redundant Information . . . . .	30
6.	Compressive Sensing Approach . . . . .	32
6.1	Intuition & Motivation . . . . .	32
6.2	Linear Algebra Problem Description . . . . .	34
6.3	Compressive Sensing Theory . . . . .	35
6.3.1	Sparsity . . . . .	35
6.3.2	Restricted Isometry Property . . . . .	36
6.3.3	Phase Transition Plot . . . . .	37
6.3.4	Solution as an Optimization Problem . . . . .	38
6.4	Compressive Sensing Approach to Acoustic Imaging . . . . .	38
6.4.1	Sparsity in Image Domain . . . . .	38
6.4.2	Alternatives to Restricted Isometry . . . . .	39
6.5	Design Restrictions . . . . .	39
6.6	Effective Measurement Formulation . . . . .	40
6.7	Measurement Model Construction . . . . .	41
6.8	Image Reconstruction . . . . .	42
7.	Results . . . . .	44
7.1	Experimental Scene . . . . .	44
7.2	Backprojection Results . . . . .	47

7.3	Compressive Sensing Results . . . . .	49
7.4	Analysis of Results . . . . .	50
8.	Future Work . . . . .	54
8.1	Channel Balancing with Backprojection . . . . .	54
8.2	Compressive Sensing using LFM Chirps . . . . .	55
9.	Conclusion . . . . .	57
Appendices:		
A.	Hardware Details and Specifications . . . . .	59
A.1	Bill of Materials . . . . .	59
A.2	TASCAM A/D-D/A Converter Specifications . . . . .	61
A.3	Tang Band Speaker Specifications . . . . .	63
A.4	CUI Inc. Microphone Specifications . . . . .	64
B.	Source Code . . . . .	68
B.1	GenerateS.m . . . . .	72
B.2	MakePNWaveform.m . . . . .	74
B.3	SRRC.m . . . . .	76
B.4	SimulateEchos.m . . . . .	77
B.5	ProcessEchos.m . . . . .	79
B.6	Backproject.m . . . . .	81
B.7	ShowImage.m . . . . .	84
B.8	Show2DImage.m . . . . .	85
B.9	FormEffectiveMeasurement.m . . . . .	86
B.10	MakeCSParameters.m . . . . .	87
B.11	FormatCSImage.m . . . . .	91
	Bibliography . . . . .	92

## LIST OF TABLES

Table	Page
4.1 PN Code Characteristics . . . . .	19
7.1 Measured Positions for Experiment Scene . . . . .	45
7.2 Comparison of Backprojection and Compressive Sensing . . . . .	51
A.1 Bill of Materials for Acoustic Imaging Test Stand . . . . .	60

## LIST OF FIGURES

Figure	Page
2.1 Backprojection Imaging: Linear Processing . . . . .	4
2.2 Compressive Sensing Imaging: Iterative Optimization . . . . .	5
3.1 TASCAM US-1641 A/D-D/A Converter . . . . .	9
3.2 Tang Band 25-302SH 1” Shielded Neodymium Dome Tweeter . . . . .	10
3.3 Electret Condenser Microphone, CUI Inc. . . . .	11
3.4 Microphone Wiring Diagram . . . . .	12
3.5 System Frequency Magnitude Response for Speaker/Microphone Pair	14
3.6 Acoustic Imaging Test Stand . . . . .	15
3.7 Microphone Mounting System . . . . .	16
4.1 PN Code Frequency Spectrum . . . . .	20
4.2 Correlated PN Code . . . . .	21
4.3 Example Range Profile . . . . .	22
4.4 Bistatic Imaging Geometry . . . . .	23
4.5 Block Diagram for Processing on Receive . . . . .	24
4.6 Block Diagram for Processing on Receive . . . . .	24

5.1	Elliptical Projections in Bistatic Imaging . . . . .	26
5.2	Backprojection Pseudocode . . . . .	27
5.3	Simulated Point Spread Function . . . . .	28
5.4	Measured Point Spread Function . . . . .	29
5.5	PN Code Sidelobe Structure . . . . .	30
5.6	Measured Range Profile for Two Channels . . . . .	31
6.1	Camera Analogy for Compressive Sensing . . . . .	33
6.2	Acoustic Imaging Linear Algebra Formulation . . . . .	34
6.3	Phase Transition Plot (Donoho and Tanner) . . . . .	37
6.4	Effective Measurement Processing . . . . .	40
6.5	Point Response ( <i>left</i> : Simulated Data, <i>right</i> : Measured Data) . . . . .	42
6.6	A Matrix Construction . . . . .	43
7.1	Transmitter and Receiver Labels . . . . .	46
7.2	Measured "Point Reflectors" - Copper Pipes . . . . .	47
7.3	Experimental Scene for Imaging Tests . . . . .	48
7.4	Backprojection Image of Three Reflectors (Simulated Data) . . . . .	49
7.5	Backprojection Image of Three Reflectors (Measured Data) . . . . .	50
7.6	Compressive Sensing Image of Three Reflectors (Simulated Data) . . . . .	51
7.7	Compressive Sensing Image of Three Reflectors (Measured Data) . . . . .	52
7.8	Compilation of Results . . . . .	53
B.1	Software Flow Diagram: Measurement Collection . . . . .	68



B.2	Software Flow Diagram: Backprojection Imaging . . . . .	69
B.3	Software Flow Diagram: Compressive Sensing Imaging . . . . .	70

# CHAPTER 1

## INTRODUCTION

This thesis describes the research I conducted during the 2010-2011 school year on acoustic imaging. Chapter 2 overviews SONAR and other imaging modes and describes the importance of the research. Chapter 3 describes the design and construction of the hardware related to the acoustic imaging test stand. This test stand was used to measure different scenes as they respond to a transmitted waveform. The design of these transmitted waveforms is found in Chapter 4. Chapter 5 discusses backprojection imaging, a conventional approach to image estimation.

Following the discussion of backprojection, Chapter 6 delves into the theory and approach behind compressive sensing (CS). This is the approach which was used to reduce the amount of data needed to reconstruct acoustic images. Chapter 7 summarizes the results comparing CS to backprojection and analyzes the images. Finally, Chapter 8 describes two avenues in which this research would extend easily and Chapter 9 summarizes the thesis and concludes the report.

## CHAPTER 2

### BACKGROUND

#### 2.1 SONAR & Related Imaging Modes

SONAR is a technique where audible acoustic waves are used to measure the environment by their reflective behavior. It is typically used for underwater imaging, but can be used effectively with air as the medium as well.

Specifically, acoustic waves can be used to measure distance by timing delays between echos. These echos vary in strength based on the material, shape and position of the object they reflected off of. If enough echo measurements are made from different angles, they can be used to create a 2-dimensional map of reflectivity.

This process is very similar to other types of tomographic imaging such as RADAR and MRI. RADAR shares the same processing techniques and physical principles with SONAR, but uses much higher frequency waveforms in the electro-magnetic spectrum. MRI can be thought of from a tomographic perspective even though many aspects of it are different.

SONAR was chosen early on as an ideal platform to experiment with imaging algorithms. Audio equipment is very cheap and easy to use without the need for

expensive hardware and data acquisition systems. Still though, any results gained from a SONAR test have relevance to the other imaging modes.

Data Acquisition is often a bottleneck in imaging systems. In MRI, the throughput of the machine is directly linked to how long a patient must stay in the machine for a given test. In RADAR systems, often the most costly part of the system is both the A/D converter and the communication system responsible for sending the measured information to a processing system. In both of these examples, if the amount of data that needed to be collected could be reduced, significant gains could be seen in efficiency.

## **2.2 Problem Statement**

Can image quality be preserved with a substantial reduction in measured data? If so, at what cost? Specifically, can the theory of compressive sensing (CS) be used to achieve this goal?

Compressive sensing is a recently formalized notion that many signals and images are compressible. Therefore, there should be a way to sense them in this compressed state and not be required to collect nearly as much data. While this may seem impossible, it is very possible with the right framework for measurement and reconstruction. There is a fundamental tradeoff between the ability to reduce the data that is needed with added complexity and uncertainty. This project explores that tradeoff and examines the effectiveness of compressive sensing in reducing the required data for acoustic imaging.

## 2.3 Imaging Techniques

Computed tomography is an established field and there are many different algorithms for forming an image from a set of measurements. This research focused on comparing compressive sensing to a very simple imaging technique called backprojection.

### 2.3.1 Backprojection Imaging

Backprojection is a common imaging technique where measurements are used directly to compute an estimate to the image. Each individual measurement has some ambiguity to it since it is one dimensional. Backprojection assumes that when there is ambiguity, all possibilities are equally likely. Using this idea, data points are projected across regions of possibility and the projections are summed together from all different measurements. Figure 2.1 shows a diagram describing the approach of backprojection.

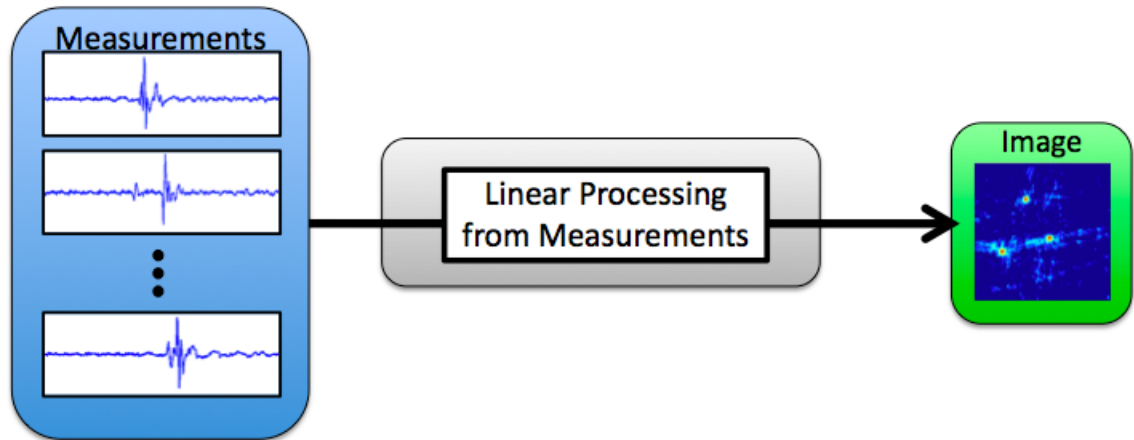


Figure 2.1: Backprojection Imaging: Linear Processing

The illustration shows that the imaging process is completely deterministic and simply performs some linear processing on the measurements to estimate the image. This is a quick and effective way to image.

### 2.3.2 Compressive Sensing Imaging

By contrast, compressive sensing imaging is a type of imaging which relies on an iterative optimization process to estimate the final image. Figure 2.2 shows this concept.

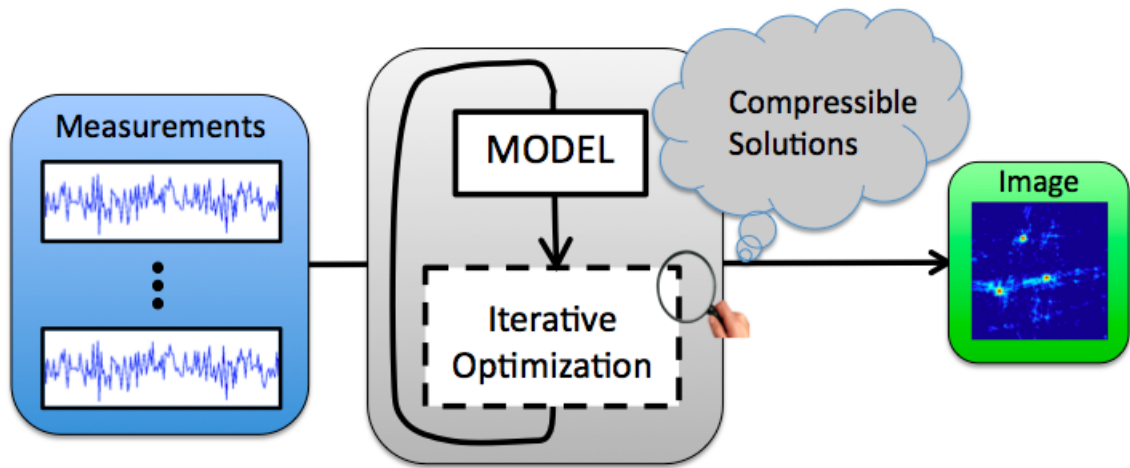


Figure 2.2: Compressive Sensing Imaging: Iterative Optimization

Compressive sensing uses a measurement model to understand how elements of the unknown image affect the measurements themselves. It uses this to iterate through possible solutions and limits itself to searching only possible compressed solutions. In this way, an exhaustive search is turned into a solvable optimization through the use of Linear Programming.

## 2.4 Objectives & Design Goals

In order to understand if compressive sensing can provide a benefit to the imaging process, it was tested and implemented on both simulated and measured data. A primary goal was to design, construct and test a physical test stand for acoustic imaging. This involved many steps from hardware selection to physical construction to early testing and software development.

Another element of design was the transmit waveform used for experiments. Section 4.1 goes into detail about the affect of the transmit waveform on testing and imaging. With the acoustic imaging test stand complete, the next goal was to implement a backprojection imaging algorithm in order to establish a baseline for image comparison.

The final objective was to design an approach for compressive sensing and implement it so that imaging could be demonstrated using the built test stand. Later chapters elaborate more on these different steps.

## 2.5 Economic & Social Considerations

Since acoustic imaging is relevant to many other imaging technologies, findings must be carefully examined against the economic and social context of the technologies that they affect. The approach of compressive sensing imaging has potentially large benefits for the economics of different imaging modes. Since data acquisition is often the limiting factor in systems, cost could be reduced greatly if a new technique reduced data collection by a large amount. This has an impact first on the users and owners of that technology. Secondly, society as a whole can benefit from this. If imaging technologies get cheaper, there will be an increased accessibility to them.

This means that more people can experience the benefits of the imaging technologies. In this way, there are social gains to be seen from humble improvements in efficiency for acoustic imaging.



## CHAPTER 3

### HARDWARE & TEST STAND DESIGN

An emphasis was placed throughout the project on measured results along side of simulated results. Therefore, significant work was done to design and build an acoustic imaging test stand. This test stand allowed for real tests to be conducted and for both imaging algorithms to be applied to actual images. The following sections describe the hardware components of the test stand. See Appendix A on page 59 for a final bill of materials. Design goals for the test stand were:

- Tabletop configuration – 4’x4’ or smaller
- 4 speakers
- 16 microphones
- Mounting system that provides stable positioning and easy adjustment

#### 3.1 Data Acquisition

Initial ideas for the project involved preserving the use of ultrasonic frequencies (typically those above 20kHz). A data acquisition (DAQ) system was found by David LaVergne to support sampling at 96ksps, providing up to 48kHz of frequency support.

The TASCAM US-1641 was the DAQ box purchased for the project. It is home audio recording equipment that provides for simultaneous recording and transmission for eight (8) microphones and four (4) speakers. Figure 3.1 shows the front and back panels of the TASCAM US-1641. The US-1641 has a USB 2.0 interface to a computer and included drivers. In addition, *Cubase LE 4.0* was used to control the box.



Figure 3.1: TASCAM US-1641 A/D-D/A Converter

## 3.2 Speakers & Microphones

The speakers and microphones are critical components of the test stand. It is essential that as a system, they have a sufficient response to transmit and receive the signals used in testing.

Several speakers were purchased initially to use for testing. The frequency response of each speaker was compared and any non-linear behavior was noted. A speaker was chosen made by Tang Band for the test stand. This speaker had the best

test results. Figure 3.2 shows the speaker as seen on the test stand. The speaker uses a neodymium design to provide a very flat response over a rated frequency range of 1.4kHz-20kHz. See Appendix A for detailed specifications.



Figure 3.2: Tang Band 25-302SH 1" Shielded Neodymium Dome Tweeter

Electret condenser microphones are an inexpensive family of microphones that have good responses over a wide frequency range. They are very small microphones and can be used with very simple circuits. The TASCAM US-1641 comes with eight (8) XLR ports which provide 48V of power supply from the box. Also, the US-1641 has built in pre-amplifiers for all of the XLR ports.

A microphone was chosen from CUI Inc. and is shown in Figure 3.3. The microphone is rated for frequencies up to 20kHz. It was wired according to the wiring diagram in Figure 3.4. See Appendix A for detailed specifications.

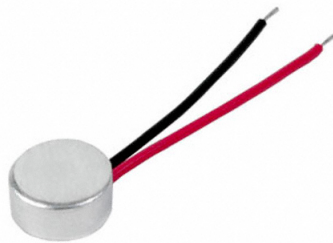


Figure 3.3: Electret Condenser Microphone, CUI Inc.

### 3.2.1 Hardware Operating Range

The speaker was tested by playing a linear frequency modulated (LFM) chirp which swept frequencies from 20Hz to 48kHz. This signal was used because it has a high bandwidth but maintains roughly constant power output. Typically, human hearing is limited to roughly 20kHz. These tests resulted in audible noise coming from the speaker when the signal was above 30kHz. This led to the conclusion that something must not be behaving properly. This could be due to non-linear behavior of the speaker above 30kHz. Because of this, all future testing and chosen waveforms were restricted to 30kHz or less.

On the lower end of the frequency spectrum, a limit was established at 2.5kHz. This was chosen to be within the rated operating range of the Tang Band speaker and also to try and limit interference from the environment. Early tests showed that there was substantial noise at very low frequencies (10-100Hz) in the test room. Also, much

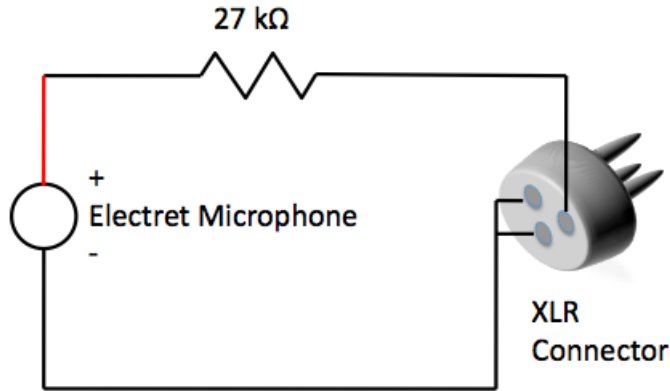


Figure 3.4: Microphone Wiring Diagram

of human speech is below 2.5kHz, so this limit allowed for accidental interference to not affect testing.

In summary, the signals used with the test stand were limited to 2.5kHz-30kHz.

### 3.2.2 System Frequency Testing

With the speaker and microphone purchased, testing was conducted to measure the system frequency response of both together. This measurement was important in validating the usable frequency range. Also, this provided a tangible way to perform equalization if needed.

Two approaches were taken for testing: (1) an LFM chirp spanning 2.5kHz-30kHz over 20 seconds and (2) a pseudo-random noise (PN) code with bandwidth from 2.5kHz-30kHz. During the test, the speaker and microphone were placed close to each other in a spatially isolated environment to eliminate echoes. The first approach relies on the long duration of the frequency sweep to accurately test each frequency. Any echoes present will not substantially change the results since they are very short

in relation to the signal time. The second approach uses a code which has very nice autocorrelation properties such that

$$s(t) * s^*(-t) \approx \delta(t) \quad (3.1)$$

where  $s(t)$  is the PN code and  $\delta(t)$  is the Kronecker delta function. Since an ideal  $\delta(t)$  contains an equal distribution of all frequencies, the correlated PN code contains a wide spectrum of frequency information. See section 4.2 on page 18 for more details about the waveform.

One speaker and microphone pair was tested using both methods. The magnitude response of the frequency spectrum was determined and compared. For the chirp test, the chirp was transmitted and recorded. Next, the Fourier transform of the recording was taken and a smoothing filter was applied to the magnitude response. For the PN code test, the PN code was transmitted and recorded. Then, the recording was match filtered against the original code. The Fourier transform of the resulting signal was taken and a similar smoothing operation was performed on the magnitude response. Figure 3.5 shows the normalized response in dB from both methods over the operating range defined in section 3.2.1.

The test shows peak response around 5kHz and roughly -10dB response at 20kHz with no lower than -15dB response at 30kHz. This verified that the speaker and microphone could be used up to 30kHz even though the ratings did not necessarily extend to that point. This information could also be used to perform equalization on each individual channel.

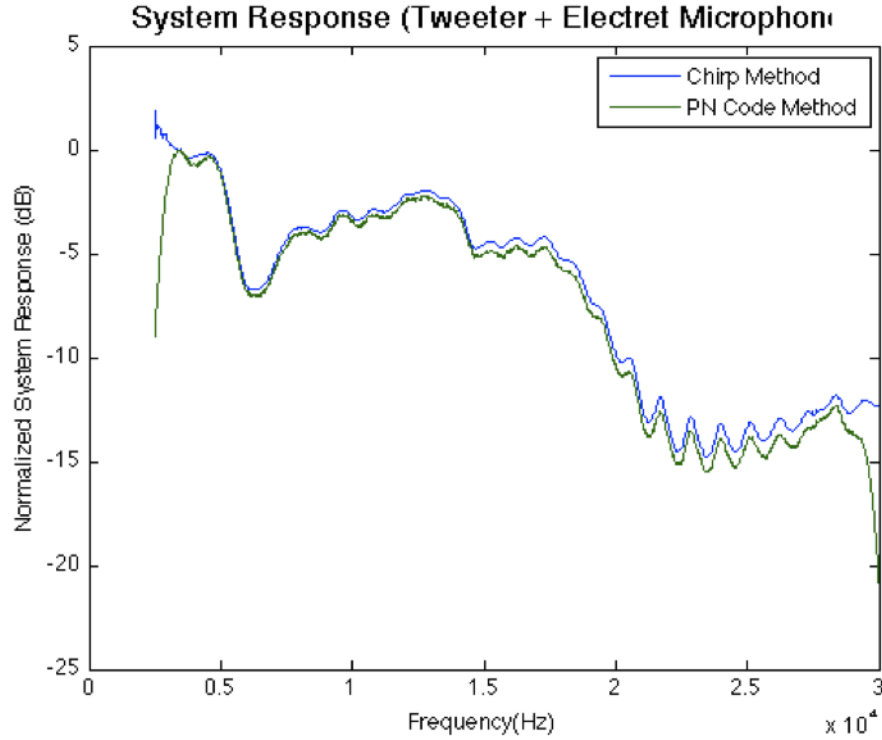


Figure 3.5: System Frequency Magnitude Response for Speaker/Microphone Pair

### 3.3 Acoustic Imaging Test Stand

With the speakers and microphones purchased, assembled and tested, the construction of the test stand began. A 4'x4' wooden base platform was used. Figure 3.6 shows the finished test stand.

#### 3.3.1 Equipment Mounting

Each of the four (4) speakers was individually mounted on a small wooden frame perpendicular to the platform emitting sound across it. These were mounted at the center of each edge of the platform as shown in Figure 3.6. Microphones were built and mounted on vertical bolts from underneath the platform. The design was easily

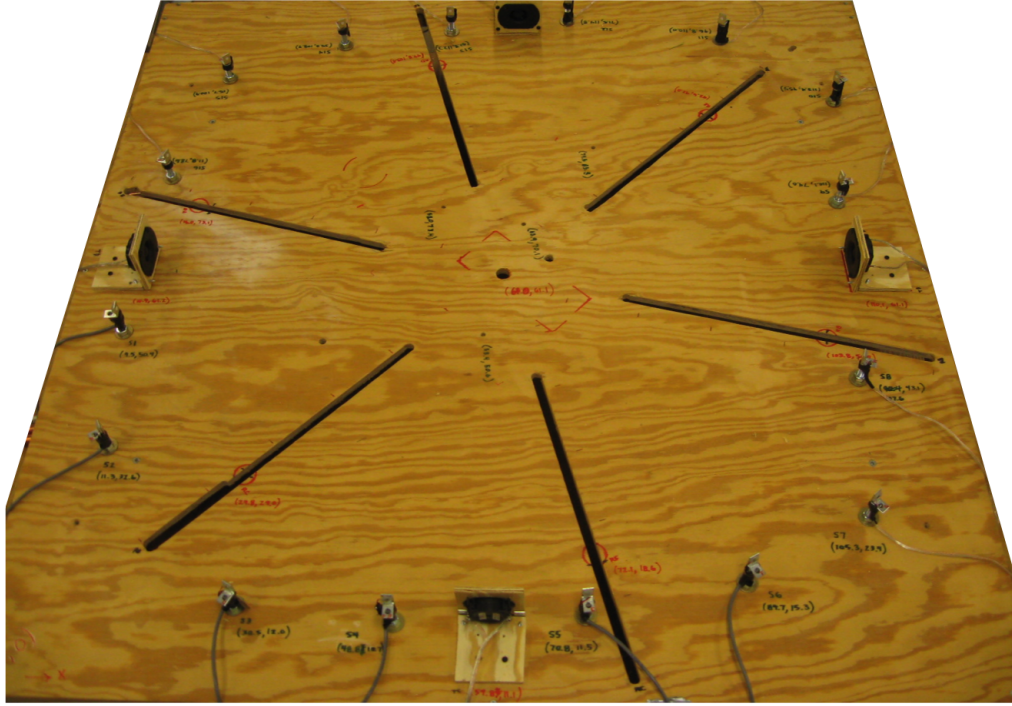


Figure 3.6: Acoustic Imaging Test Stand

adjustable allowing for quick movement of the microphones. The microphone mount included a lock-nut which attached to the top of the bolt but could easily be removed. Figure 3.7 shows the mounting of the microphones.

See Appendix A for a Bill of Materials.

### 3.4 Data Collection using the Acoustic Imaging Test Stand

A protocol was developed for collecting data from a scene for imaging. A template file was developed for the *Cubase* software which included settings for all eight (8) channels in and four (4) channels out.





Figure 3.7: Microphone Mounting System

The microphones were found to have very low responses across all frequencies. To increase the signal to noise ratio, the gains on all input channels were set to the maximum. This was done using the knobs on the front of the TASCAM box. Also, the output levels for the speakers were maximized to +6.02dB. This was done through software control using the *Cubase* program.

Data for 64 different speaker/microphone pairs must be collected for each experiment. This is done by transmitting on each speaker at different times, then segmenting the recording on each microphone into four files. Also, data from 16 microphones is collected by doing two rounds of data collection with eight each time.

## CHAPTER 4

### WAVEFORM DESIGN & ECHO PROCESSING

Careful consideration was put into the design of the transmit waveform,  $s(t)$ , for the imaging system. A pulse compression scheme was chosen whereby the received signal is matched filtered against the transmit waveform to generate a narrower compressed peak.

#### 4.1 Design Considerations

There are two desirable signal properties that were designed for:

1. Resolution
2. Signal to Noise Ratio (SNR)

Resolution in a pulse compression system is lower bounded by a function of the bandwidth of the transmit signal.

$$d_{res} \geq \frac{c}{2 \cdot BW} \quad (4.1)$$

where  $c$  is the speed of sound,  $BW$  is the transmit signal bandwidth in Hz and  $d_{res}$  is the achieved resolution in meters.

From section 3.2.2, the usable range of frequency for the speakers and microphones is from 2.5kHz up to 30 kHz. To achieve the best resolution, the full bandwidth was used. Therefore, the best resolution is

$$d_{res} \geq \frac{c}{2 \cdot BW} \geq \frac{343 \frac{m}{s}}{2 \cdot 27.5 kHz} \geq 6.2mm \quad (4.2)$$

SNR improves through match filtering. Since the filtering is essentially a pattern match operation, any noise will be reduced substantially. Therefore, any effect from noise in the match filtered signal will be reduced. Also, the effect of match filtering can bring out the signal into a very strong peak response. The combination of both of these increases the signal to noise ratio.

## 4.2 PN Code: Description & Correlation Properties

Based on the design requirements above, a pseudo-random noise (PN) code was chosen as the transmit waveform. The PN Code is a discrete sequence of symbols in the set  $\{-1,1\}$ . It is designed in such a way that the integral of  $s(t) * s(t - \tau)$  is maximal when  $\tau = 0$  and as close to zero as possible otherwise. The PN code has  $2^{16} - 1$  symbols placed at consecutive samples in the 96kHz digital waveform. This means that it contains all frequencies up to 48kHz.

The resulting PN code is bandpass filtered to only contain frequencies from 2.5kHz to 30kHz. Table 4.1 lists the properties of the transmit waveform. Figure 4.1 shows the frequency spectrum of the generated signal.

Choosing  $s(t)$  as a PN code described above, the properties can be studied. The signal is 0.68 seconds long and has a roughly constant envelope. This means that it can be transmitted effectively without requiring power fluctuations to the speaker.

Symbol Set	$\{-1,1\}$
Sequence Length	$2^{16} - 1$
Symbol Frequency	96ksps
Analog Signal	sinc interpolation
Filtering	Bandpass (2.5-30kHz)

Table 4.1: PN Code Characteristics

In the absence of noise, the filtered PN code produces a narrow peak with some sidelobe structure. Figure 4.2 shows the correlated waveform and a close-up of the peak structure. The resulting waveform has a waveform roughly 50 samples wide, corresponding to  $500\mu s$ . The main peak is 2 samples wide at the base. This corresponds to an effective resolution of  $7.1mm$ .

### 4.3 Echo Processing

The PN code described above was used with the imaging test stand to measure distances using echoes. The recorded signals from each microphone were processed to create what is called a range profile in RADAR terminology. This range profile is a signal that contains narrow peaks that describe the location of the reflective objects in the scene. Figure 4.3 shows an example range profile where three separate point reflectors were in the scene.

Three separate echoes can clearly be seen in the figure. These types of measurements were used to construct a 2-dimensional image of scenes.

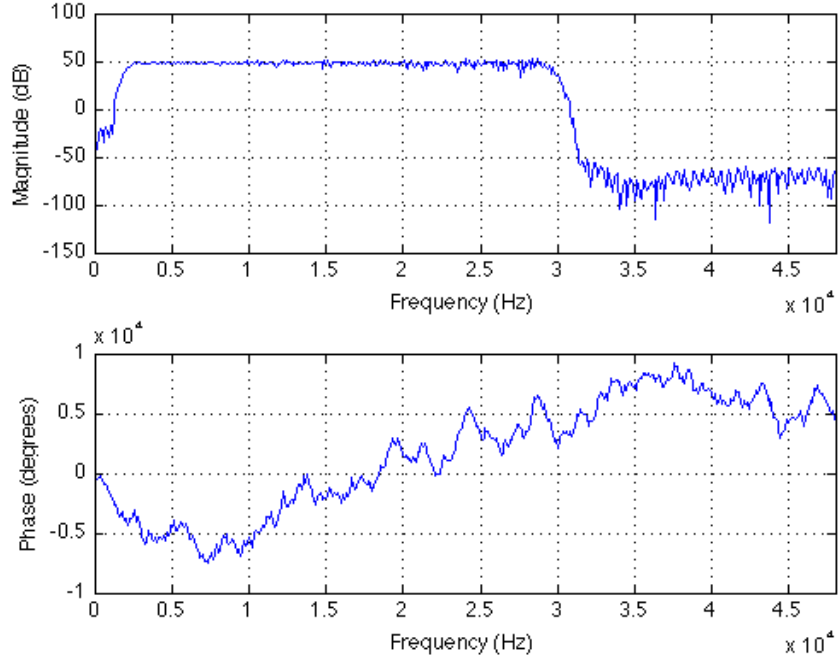


Figure 4.1: PN Code Frequency Spectrum

### 4.3.1 Echoes from Bistatic Geometries

The acoustic imaging test stand is an example of bistatic imaging geometry, where the transmitter and receiver are placed at different locations. The range profile contains echoes which are placed based on the total path length of the echo. Figure 4.4 illustrates a bistatic geometry.

$$t_{total} = t_{Tx-Rf} + t_{Rf-Rx} \quad (4.3)$$

where  $t_{total}$  is the total echo time,  $t_{Tx-Rf}$  is the time from the transmitter to the reflector and  $t_{Rf-Rx}$  is the time from the reflector to the receiver.

The figure shows that there is an inherent ambiguity with each individual measurement. Different reflectors that are both located along an ellipse with foci as the

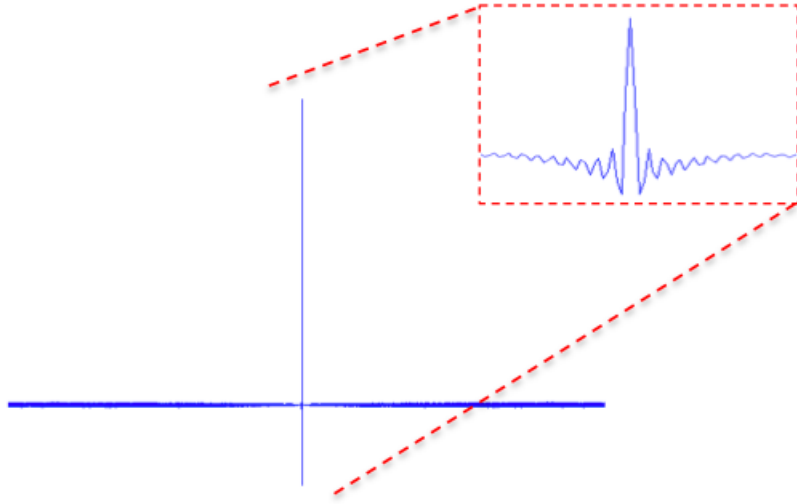


Figure 4.2: Correlated PN Code

transmitter and receiver all have the same total path length, causing their echos to add constructively in the final range profile. Therefore, it is impossible to distinguish the echoes from these different reflectors. It is this ambiguity that drives the need for different angles and more data. Chapter 5 details the process of taking range profiles and constructing an image directly.

### 4.3.2 Range Profile Signal Processing

As mentioned earlier, the range profiles are the result of signal processing on the recorded signal. Figure 4.5 shows the block diagram of the process. First, each of the 64 recordings are match filtered against the PN Code,  $s(t)$ . Then, they are shifted in time to compensate for the delay from filtering and the hardware delays of the system. The hardware delay through the TASCAM box was found experimentally to be 412 samples, corresponding to  $4.29ms$ . The result is a waveform where the

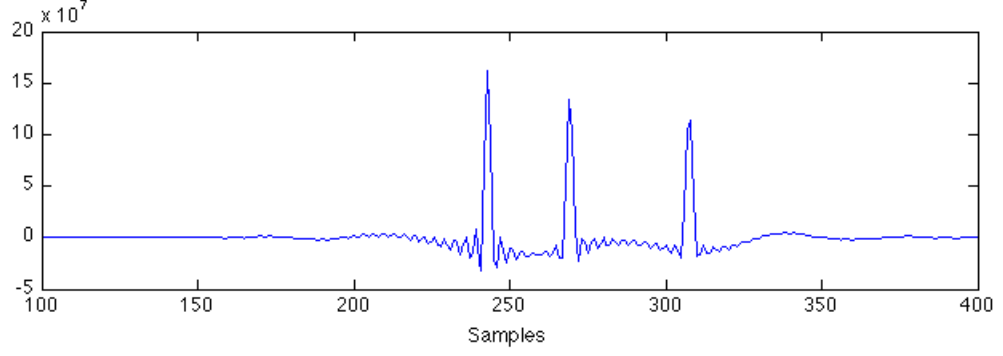


Figure 4.3: Example Range Profile

very first data point corresponds to the location of the speaker. Any echos present are measured from this point. All acoustic waves were assumed to propagate at a constant speed of  $c = 343 \frac{m}{s}$ .

Then, each signal is gated in time to remove echos from objects outside of the image scene. Using the DAQ sampling rate of 96kHz and the speed of sound, this limits the final measurements to a few hundred samples.

### 4.3.3 Background Subtraction

The basic processing described in the prior section is effective and produces range profiles which describe where the reflective objects in the scene are. However, the static components of the test stand itself produce many unwanted echoes. Also, the direct transmission of the signal to the microphone is seen in the range profile.

In order to remove unwanted echos and keep just those that are related to the scene of interest, a method of coherent background subtraction was used. Since the scene and test stand is static, the echos measured from the test stand should be

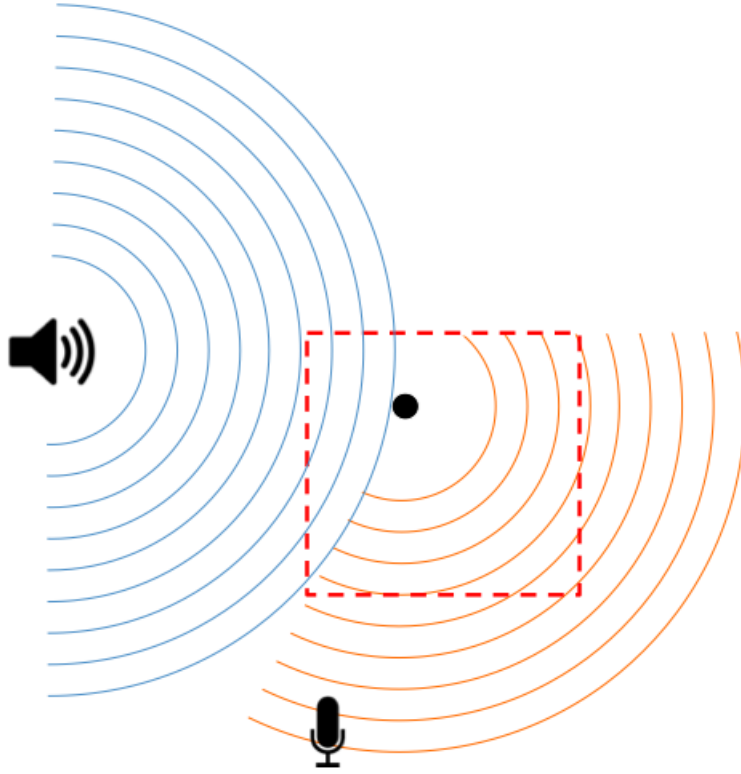


Figure 4.4: Bistatic Imaging Geometry

repeatable and can be used to selectively remove the unwanted echoes from future recordings.

As described in section 3.4, for each of the 64 geometries, a recording is made with no reflective objects in the scene (the background) and also with the reflectors present. This is done without unplugging the microphones or changing any other configuration. Both signals are processed to generate a range profile and then they are simply subtracted from one another. The resulting signal is a range profile that contains just the echos from the reflectors introduced to the scene. Figure 4.6 shows this process for a sample data set.



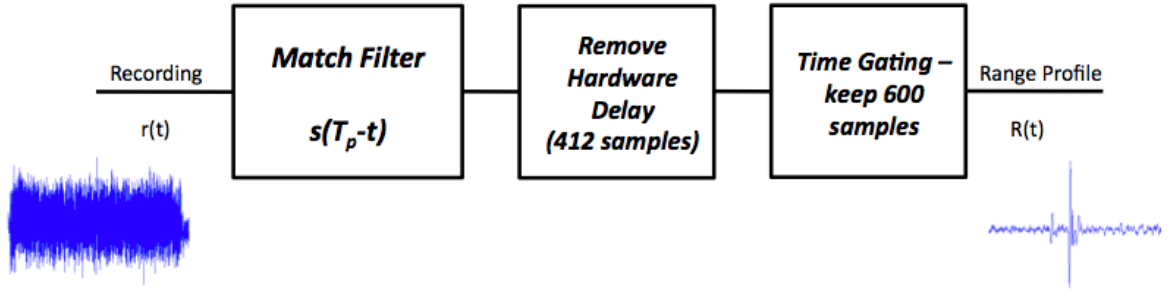


Figure 4.5: Block Diagram for Processing on Receive

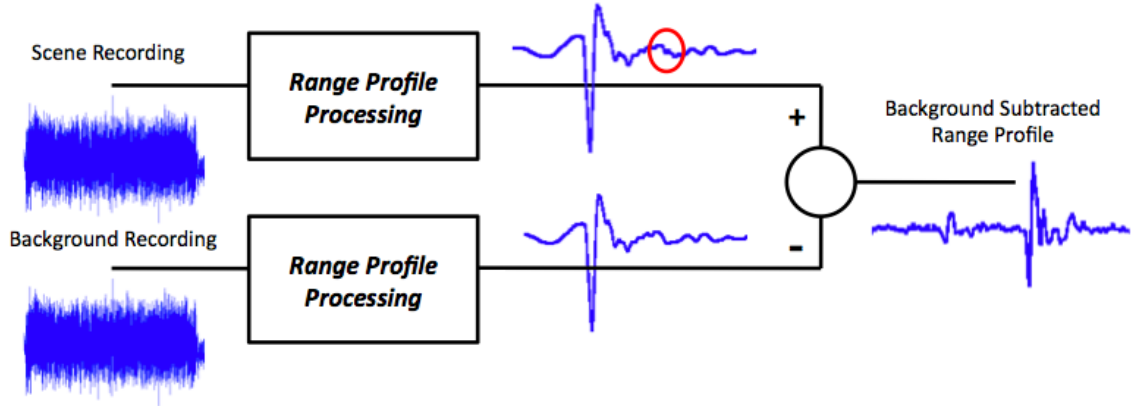


Figure 4.6: Block Diagram for Processing on Receive

The key benefit of this approach is that the affect of the background can be removed. This is a much easier approach than having to use physical isolation to remove unwanted echos.

These range profiles are the data provided to the imaging algorithms detailed in the next chapters. Chapter 5 details a backprojection algorithm while chapter 6 discusses the use of compressive sensing.

## CHAPTER 5

### BACKPROJECTION IMAGING

Backprojection is a traditional imaging technique used for many different imaging modes. It is based on the principle that in the presence of ambiguous locations, the data will simply be "smeared" across the entire space of possible locations. These smears are known as projections. The projections are added up from all the different geometries and the final image contains constructive interference in the location of reflectors and destructive interference everywhere else. While the principles of backprojection are the same for all modes of imaging, the form of the projections depends on the data collection.

#### 5.1 Bistatic Projections

For bistatic imaging, the projections take the shape of ellipses with foci at the transmitter and receiver. Figure 5.1 shows a sample projection over the image space from a simulated range profile.

When the projections are added from all 64 different speaker/microphone pairs, the result is an image which approximates the scene. Reflective objects are shown as strong responses because of the constructive sum of all the signals. The other

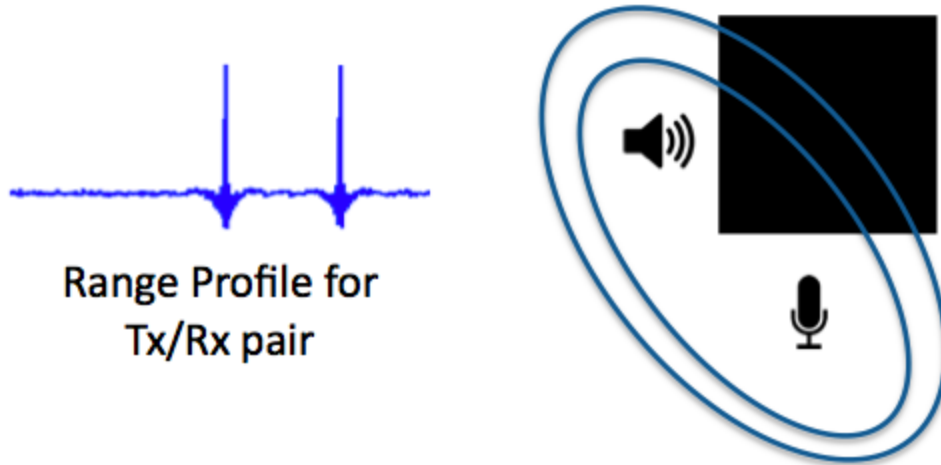


Figure 5.1: Elliptical Projections in Bistatic Imaging

locations effectively get canceled out because no more than a few projections pass through any given point.

## 5.2 Backprojection Algorithm

The implementation of the backprojection algorithm is fairly straight forward. Figure 5.2 describes the pseudocode for the algorithm. Essentially, a loop goes through each pixel location and examines all of the range profiles. The final value of each pixel is the sum of the appropriate values from each range profile. The appropriate value is chosen by calculating the total echo time for the given transmitter and receiver to that pixel.

Each range profile is interpolated to the pixel grid before beginning. This guarantees the most accurate image for a given pixel grid. See *backproject.m* in Appendix B for implementation.

```

for each pixel do
  for each Tx/Rx do
    pixel = pixel + range profile(pixel location)
  end for
end for

```

Figure 5.2: Backprojection Pseudocode

The backprojection algorithm was used on several data sets in order to demonstrate the quality of the algorithm. First, a single centered point reflector was simulated in *MATLAB*. The designed PN code was used to create range profiles for each speaker/microphone pair. The generated range profiles were formed into an image using the backprojection algorithm. Figure 5.3 shows the resulting image on a dB scale. This image is known as the Point Spread Function (PSF) and describes how the imaging algorithm represents a single point reflector in the scene.

Figure 5.4 shows the same test but using measured data and background subtraction described above. To approximate a point reflector, a single  $\frac{3}{8}$ " copper pipe was used as a reflector.

There is a substantial difference in the simulated PSF and the measured PSF. Section 5.3.2 discusses this problem in more detail.

### 5.3 Backprojection Inefficiencies

The figures above show a good approximation to the correct scenes but have some notable flaws. Several of these problems are detailed below.

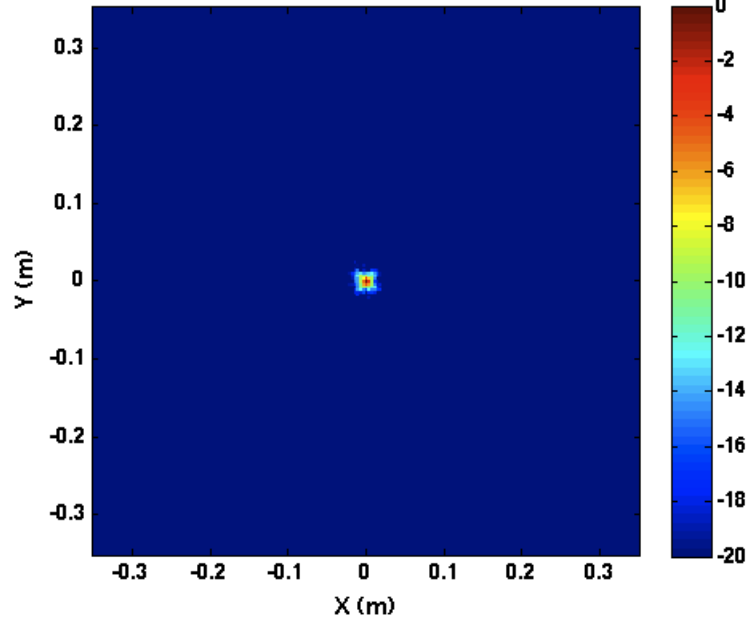


Figure 5.3: Simulated Point Spread Function

### 5.3.1 Point Spread Function

The PSF is a function of both the transmitted waveform and the number of angles that were measured. If the transmit waveform could generate a perfect  $\delta(t)$  after match filtering and all angles were observed, the PSF would be ideal. In practice, the transmit waveform contains a certain shape. Seen again here in Figure 5.5, the PN code has a sidelobe structure that is not ideal. The PSF is a 2-dimensional extension of this sidelobe structure surrounding each point reflector circularly.

The PSF is an inherent part of the resulting backprojection image unless it is intentionally removed through some deconvolution process.

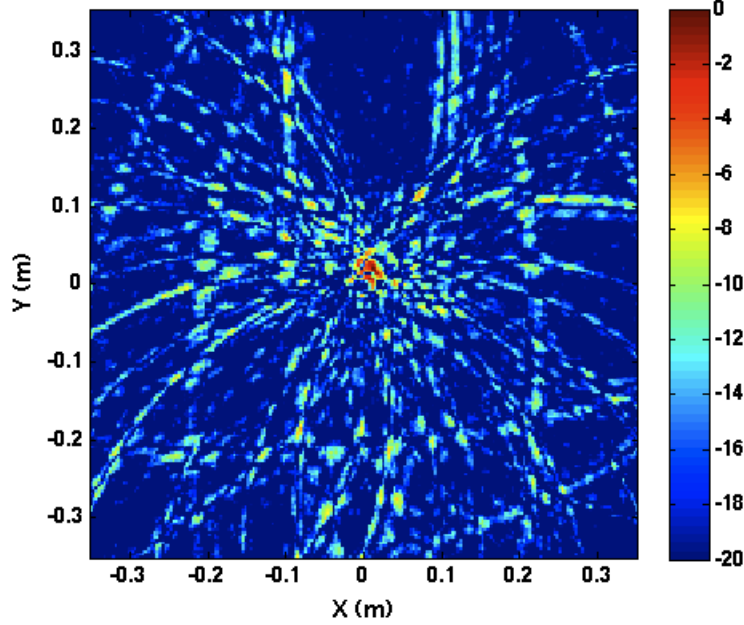


Figure 5.4: Measured Point Spread Function

### 5.3.2 Channel Imbalances

One of the largest differences between the simulated and measured results shown in Figures 5.3 and 5.4 is due to channel imbalances. The backprojection algorithm is carefully balanced based on the fact that the response from each channel will be similar. It is this balance that produces the nice destructive interference at non-reflective points. If the gains or SNR are different for each channel, this balance is upset and the resulting image shows the affects of the imbalance.

Figure 5.6 shows the measured range profile from two different channels. The first from speaker #1 and microphone #1 and the second from speaker #1 and microphone #9. Two things are apparent from the images. First, the strength of the echo is different in the two images. Second, the SNR is very different as well. The

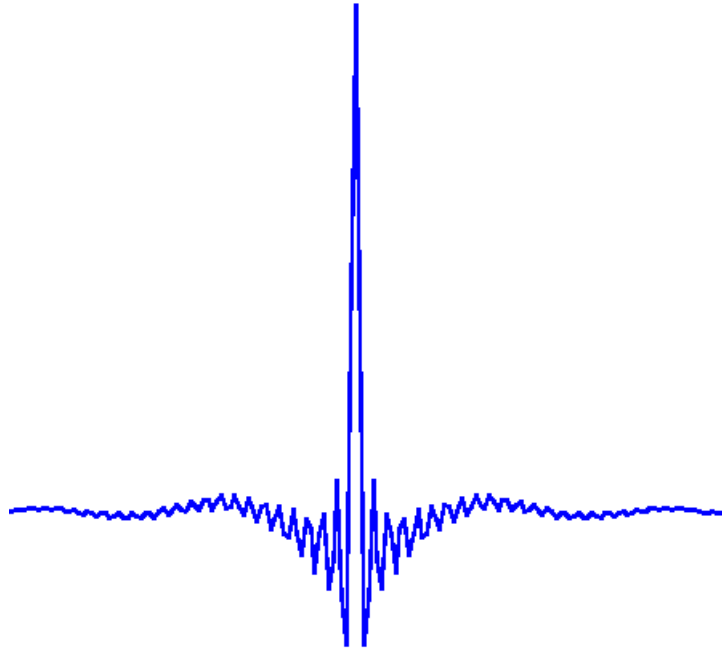


Figure 5.5: PN Code Sidelobe Structure

first graph shows a strong SNR of 30.2dB, while the second has a much lower SNR of only 15.2dB.

These inequalities were not adjusted for in the backprojection algorithm and therefore affected the final images substantially.

See Chapter 8 for an idea to compensate for both Channel Imbalances and PSF for standard imaging.

### 5.3.3 Redundant Information

One final criticism of backprojection is that it is wasteful of data and uses many redundant measurements to produce an accurate image. The measurements are inherently redundant to some extent because they describe the same scene but from

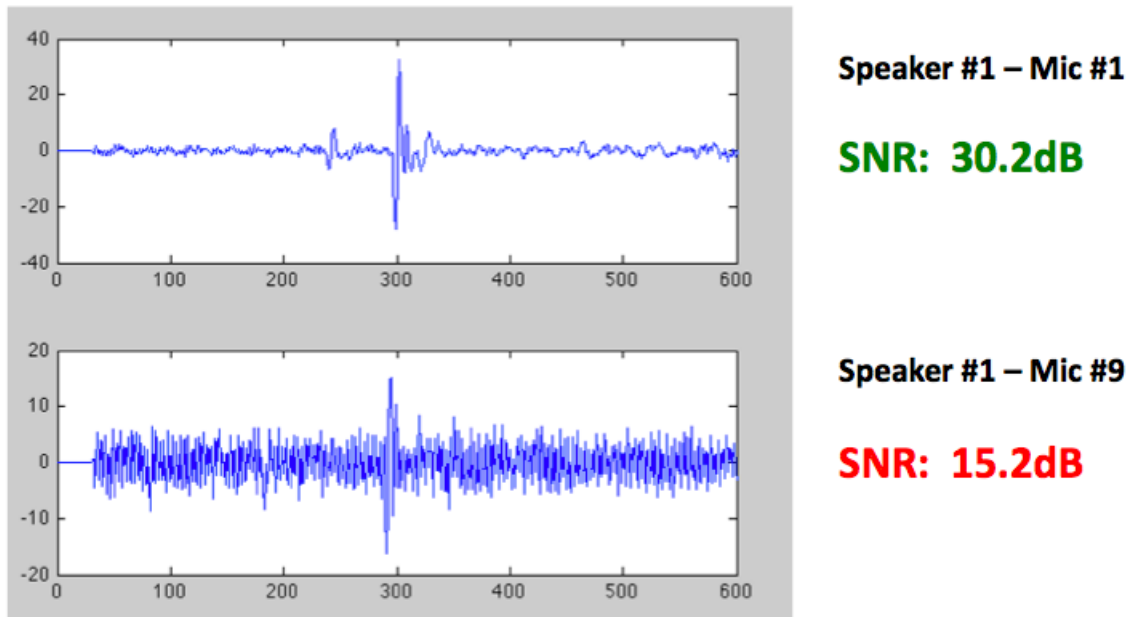


Figure 5.6: Measured Range Profile for Two Channels

different angles. Also, the amount of data used for backprojection is not dependent at all on the complexity of the image itself. For example, a scene with a single point reflector needs the same amount of data as a complicated scene with many reflectors of different shapes.

Compressive sensing reduces the amount of redundant measurements by including *a priori* information about the solution. This is described in more detail in the next chapter.



## CHAPTER 6

### COMPRESSIVE SENSING APPROACH

Compressive sensing is an alternate approach to image reconstruction and promises no loss of quality but uses substantially less data. Section 6.1 details the intuition behind this approach followed by a Linear Algebra framework for SONAR imaging in section 6.2. Section 6.3 goes through the formal theory surrounding compressive sensing and the requirements for success. Sections 6.4, 6.5 and 6.6 describe how this theory was adapted for the specific acoustic imaging problem. Sections 6.7 and 6.8 describe the process of modeling and solving for images.

#### 6.1 Intuition & Motivation

Compressive sensing is the formalization of ideas and techniques that have been seen in different settings for many decades. Fundamentally, it says that if a signal or image can be compressed in some way, there might be a way to measure it directly in its compressed state. Figure 6.1 shows a common analogy of a digital camera.

It is well known through the success of compression techniques such as JPEG and JPEG2000 that most images are relatively compressible. In the right basis, many images can be represented with very few non-zero coefficients without a loss of visual

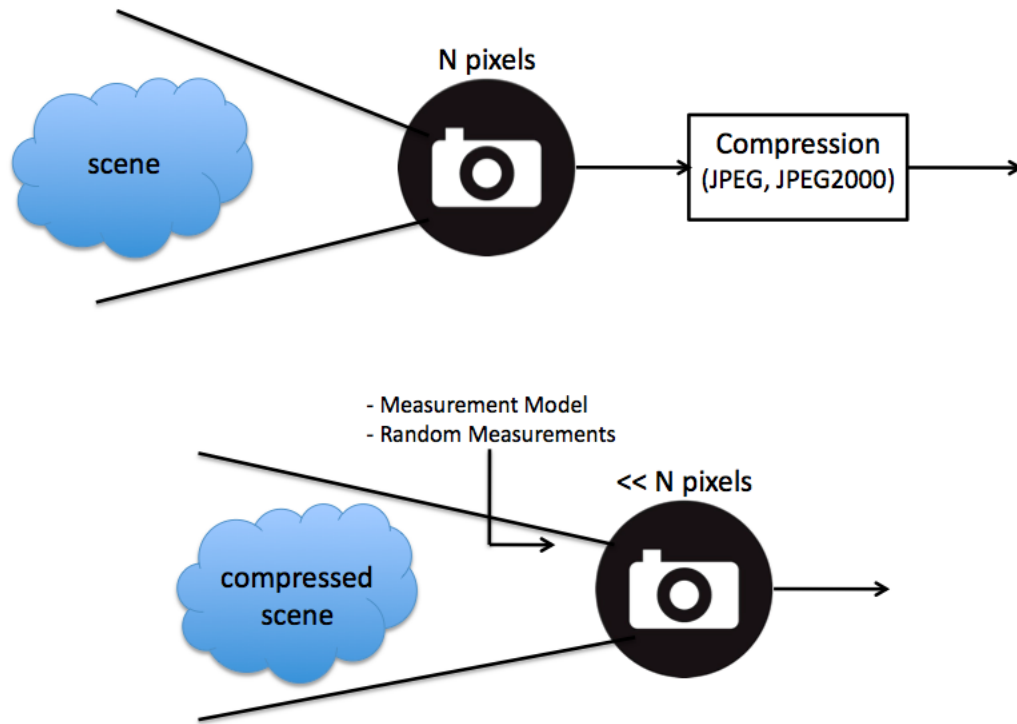


Figure 6.1: Camera Analogy for Compressive Sensing

quality. Therefore, the idea of compressive sensing is to sense this compressed state directly rather than taking in the full image and applying compression after.

If this can be done, then less data can be collected. This is a very important result for many applications including SONAR, RADAR and MRI. However, there are certain conditions that must be met to guarantee success. Section 6.3 describes these conditions. First, the problem of acoustic imaging must be described in a linear algebra framework.

## 6.2 Linear Algebra Problem Description

Acoustic imaging, like all imaging modes, can be described as a system of linear equations. Backprojection can be thought of as a particular estimation to the solution of this system and compressive sensing is an alternate approach.

Let  $\tilde{\mathbf{x}}$  be an  $N \times 1$  column vector containing a list of all the  $N$  pixels in the unknown image.  $\tilde{\mathbf{y}}$  is an  $M \times 1$  column vector containing the measurements collected from the image. Then,  $\mathbf{A}$  is an  $M \times N$  matrix which describes how each pixel in the image is measured. Specifically, the  $n$ th column of  $\mathbf{A}$  is the measured response due to an object in the  $n$ th pixel of  $\tilde{\mathbf{x}}$ . Figure 6.2 shows the formulation of the imaging problem. The objective is to solve for  $\tilde{\mathbf{x}}$  such that  $\tilde{\mathbf{y}} = \mathbf{A}\tilde{\mathbf{x}}$ .

$$\begin{bmatrix} \tilde{\mathbf{y}} \end{bmatrix} = \begin{bmatrix} \mathbf{A} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}} \end{bmatrix}$$

Figure 6.2: Acoustic Imaging Linear Algebra Formulation

For the backprojection imaging described in chapter 5, the measured data included 64 different speaker/microphone pairs each with 600 sample range profiles.

The measurement vector is defined by stacking all of these 64 measurements on top of each other into a single column containing all the information.

Compressive sensing offers an approach to solve this system of linear equations when  $M \ll N$  making the system very underdetermined. A solution can only be

guaranteed if two conditions are met: compressibility of the image and a sufficiently incoherent measurement model.

## 6.3 Compressive Sensing Theory

The mathematical framework for compressive sensing was formalized in [1], [2] and [3]. They provide mathematical conditions which are sufficient to guarantee an exact solution. The first condition is surrounding the compressibility of sparsity of the solution. The second surrounds the incoherence of the  $\mathbf{A}$  matrix.

### 6.3.1 Sparsity

An unknown  $\tilde{\mathbf{x}}$  of size  $N \times 1$  is defined to be **S – sparse** if it contains only  $S$  non-zero elements where  $S < N$ . In applications, many signals are not sparse in the standard basis, but can be described in another basis where they are sparse. Assume that there exists some linear transformation  $\Psi$  such that

$$\tilde{\mathbf{u}} = \Psi \tilde{\mathbf{x}} \tag{6.1}$$

In other words, the  $i^{th}$  element of  $\tilde{\mathbf{u}}$ ,

$$u_i = \sum_{j=1}^N \psi_{i,j} \cdot x_j \tag{6.2}$$

where  $\psi_{i,j}$  is the  $(i, j)^{th}$  element of  $\Psi$ . In many cases, it is sufficient to threshold the coefficients in this sparse basis, without a loss of information. A common example is a truncated Fourier Series. Most of the energy is in the first few terms of a Fourier series, so an approximation can be made by only keeping the first few terms.

A similar approximation is often made in image compression. Some transformation such as the Discrete Cosine transform (DCT) or discrete Wavelet transform

(DWT) is applied, then only coefficients above a certain threshold are kept. This is the scheme used by compression schemes such as JPEG-2000.

Once the unknown  $\tilde{\mathbf{x}}$  can be described so that it is  $S$ -sparse, the second condition must be examined.

### 6.3.2 Restricted Isometry Property

If two possible solutions for  $S$ -sparse  $\tilde{\mathbf{u}}$  are chosen arbitrarily,  $\tilde{\mathbf{u}}_1$  and  $\tilde{\mathbf{u}}_2$ , their difference is defined as  $\tilde{\mathbf{u}}_\Delta = \tilde{\mathbf{u}}_1 - \tilde{\mathbf{u}}_2$ .

It follows that since both  $\tilde{\mathbf{u}}_1$  and  $\tilde{\mathbf{u}}_2$  are  $S$ -sparse,  $\tilde{\mathbf{u}}_\Delta$  is at most  $2S$ -sparse. The Restricted Isometry Property (RIP) is a restriction on the  $\mathbf{A}$  matrix which says

$$\exists \delta \text{ s.t. } (1 - \delta)\|\tilde{\mathbf{u}}_\Delta\|_2^2 \leq \|\mathbf{A}\tilde{\mathbf{u}}_\Delta\|_2^2 \leq (1 + \delta)\|\tilde{\mathbf{u}}_\Delta\|_2^2 \quad (6.3)$$

If the RIP for  $\mathbf{A}$  is met, then an exact solution for an  $S$ -sparse  $\vec{x}$  can be found by solving the optimization equation

$$x = \arg \min_x \|\vec{u}\|_1 \text{ s.t. } y = \Phi x \quad (6.4)$$

where  $\Phi = A \cdot \Psi$ . Unfortunately, the RIP cannot be tested since infinitely many combinations must be tried. A looser criteria states that the columns of  $\mathbf{A}$  must be as uncorrelated as possible. In this case, reconstruction is highly likely. If a Gaussian distribution is used for the  $\mathbf{A}$  matrix, then the RIP is approximately met when the number of measurements,  $M$

$$M \approx S \log_2 \left( \frac{N}{S} \right) \quad (6.5)$$

### 6.3.3 Phase Transition Plot

If an  $A$  matrix is found to be RIP, there is a set of design parameters which can be chosen: The number of measurements to be taken and the sparsity of the unknown solution. Work done by [4] produced a plot called a Phase Transition Plot which shows which parameters will lead to a solvable problem using compressive sensing. Figure 6.3 shows the phase transition plot as seen in [4].

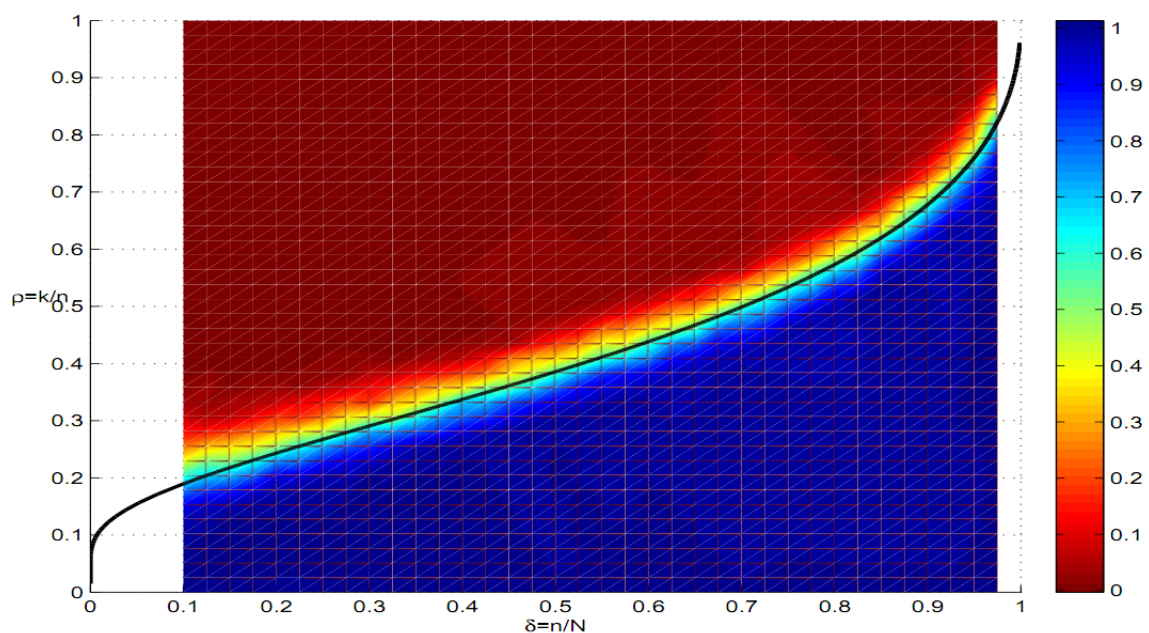


Figure 6.3: Phase Transition Plot (Donoho and Tanner)

The plot shows  $\rho$  vs.  $\delta$  where  $\rho = \frac{S}{M}$  and  $\delta = \frac{M}{N}$ . In other words,  $\delta$  represents how underdetermined the system is, with a lower number meaning less equations than unknowns.  $\rho$  is the ratio of non-zero coefficients in the unknown solution to measurements.

The plot indicates that in order to be solvable, the ratio of measurements to non-zero coefficients must be sufficiently high for a given measure of how underdetermined the system is. The line indicates the boundary where all points below are solvable using CS and all points above must be solved using exhaustive search.

### 6.3.4 Solution as an Optimization Problem

Finally, assuming that  $S$ ,  $N$  and  $M$  are chosen so the problem is solvable using CS, the solution can be found using convex optimization (see equation 6.4).

In the presence of noise, if a certain error  $\epsilon$  is acceptable, a solution for  $\vec{x}$  can be found by alternately solving

$$\vec{x} = \arg \min_x \|\vec{x}\|_1 \text{ subject to } \|A\vec{x} - \vec{y}\|_2 < \epsilon \quad (6.6)$$

The theory behind convex optimization is beyond the scope of the project. An algorithm called SPGL1 described in [5] and found at [6] was used to solve these problems using *MATLAB*.

## 6.4 Compressive Sensing Approach to Acoustic Imaging

The compressive sensing theory was used and adapted to the acoustic imaging problem in order to improve the imaging technique.

### 6.4.1 Sparsity in Image Domain

To meet the sparsity condition described above, an early choice was made to just use images which are sparse in the image domain. This means they are composed of a small number of point reflectors. This was realized using the test stand by imaging small copper pipes which appear as just a few pixels in the imaging process.

### 6.4.2 Alternatives to Restricted Isometry

The RIP is a sufficient but not necessary condition for solvability. It is a difficult condition to meet with real-world systems, but close approximations can yield useful results.

As mentioned before, a looser condition on the  $A$  matrix is that every subset of the columns of  $A$  are approximately orthogonal. The closer that this condition is, the better the chances of reconstruction are.

Unfortunately, the  $A$  matrix cannot be freely chosen in the acoustic imaging problem. That is because it encapsulates the effect on measurements due to the transmit waveform and the positions of the speakers and microphones. These fixed choices restrict the columns of the  $A$  matrix. However, steps can be taken to transform these measurements using some linear transformation and improve the chances of reconstruction. Section 6.6 describes the specific approach used for testing.

## 6.5 Design Restrictions

*MATLAB* was used with the SPGL1 algorithm to solve the compressive sensing optimizations. A memory constraint in *MATLAB* limits matrices to 192 million elements. In order to stay under this limit, and to try and meet the right ratios needed to compressive sensing, the following parameters were chosen:

- $N = 40,000$  unknowns (200x200 pixel image)
- $M = 3,000$  measurements
- $S \leq 10$  point reflectors



This made sure that the A matrix which is NxM is sufficiently small to be held in memory at one time. In order to use 3,000 measurements to solve for the image, 150 data points were taken from 20 different speaker/microphone pairs. For each of the 20 pairs, the full range profile was taken. This range profile was put turned into an effective measurement which served two purposes. First, it provided a means of making the measurement more random so that the A matrix columns will be more uncorrelated. Second, the effective measurement only contains 150 data points, which meets the design requirement.

## 6.6 Effective Measurement Formulation

To meet the design requirements for the imaging system, A process was created to form an effective measurement from range profiles. Figure 6.4 shows the process in block diagram form.

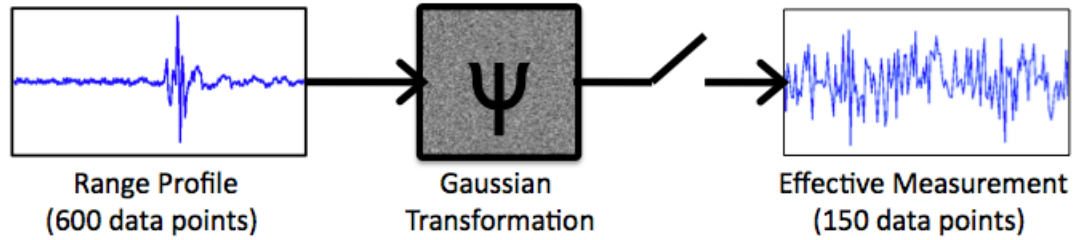


Figure 6.4: Effective Measurement Processing

First, the 600 point range profile is transformed by a pre-determined Gaussian distributed 600x600 matrix. This effectively takes each point in the original range profile and randomly disperses it among all the data points. Then, this randomized

signal is sampled by taking 150 randomly pre-selected data points. These points comprise the final effective measurement.

This method was found to produce sufficient randomization in the measurements so that compressive sensing could work reliably for the scenes that were tested.

It is important to note that this approach of processing the range profile to both reduce the number of samples and to apply a transformation is not purely in the vein of compressive sensing. If hardware existed which could do this at the time of data collection, it would be sufficient. However, this approach does not reduce the amount of data collected initially, it only selects a portion of the data to use for demonstration.

Because of this, the results put forward in this thesis can only be considered a proof of principle; imaging can be done using compressive sensing using the specified ratios and design parameters. Actual implementation would depend on being able to collect the effective measurement directly, which may or may not be possible.

See section 8.2 for more information about a different approach which could be actually implemented in hardware and would likely yield the same results.

## **6.7 Measurement Model Construction**

Once the effective measurements were determined, a measurement model matrix was created. First, measurements were made from a single centered point reflector to model the point response. Figure 6.5 show a backprojection image of a single point reflector using simulated and measured data. Effective measurements from both data sets were used to model the system.

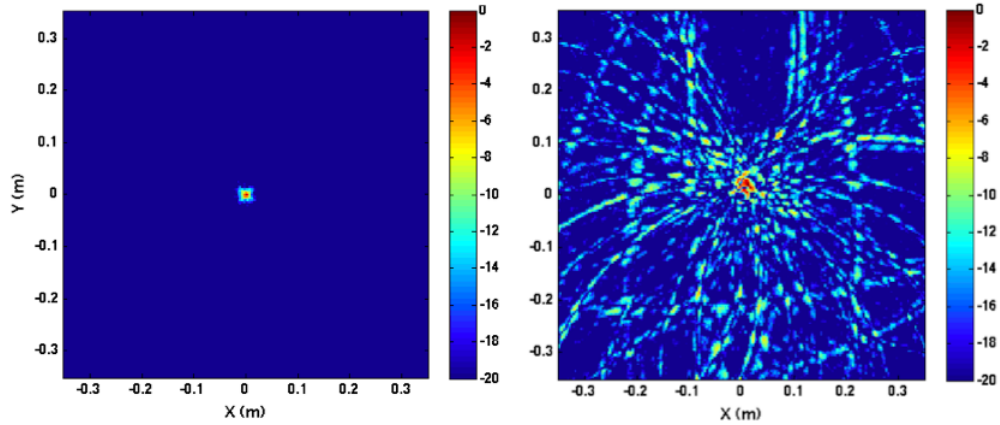


Figure 6.5: Point Response (*left*: Simulated Data, *right*: Measured Data)

Each column of  $\mathbf{A}$  was populated by stacking the effective measurements from each of the 20 geometries used in the test. In order to model the response of each individual pixel, the images above were simply shifted so that the point reflector was in each pixel and the measurements were derived from this.

The process is illustrated in Figure 6.6. This approach depends on the response from the copper pipe being very similar regardless of its location.

An  $\mathbf{A}$  matrix was constructed for both simulated and measured data and these were provided as measurement models to the optimization algorithm.

## 6.8 Image Reconstruction

With the system models constructed, the image was constructed by providing the SPGL1 algorithm with three parameters: the model  $\mathbf{A}$ , the measurement  $\vec{\mathbf{y}}$  and an error bound  $\sigma$ . Solutions for a 200x200 image generally took a few thousand iterations taking roughly 10 minutes to compute on a laptop.

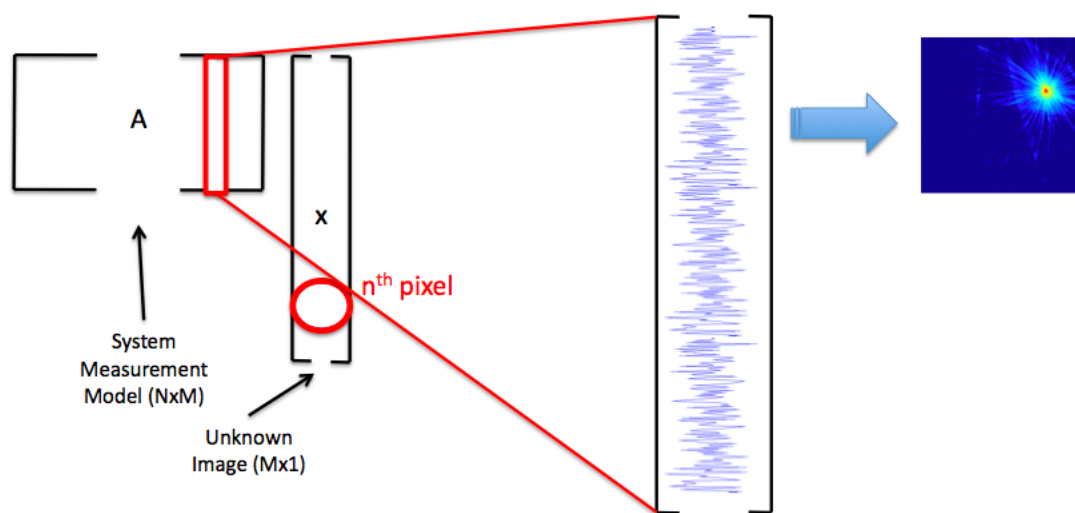


Figure 6.6: A Matrix Construction

## CHAPTER 7

### RESULTS

With the software and hardware configured and designed, a simple demonstrative test was performed to examine both backprojection and compressive sensing for reconstructing images from acoustic data. In both cases, a simulated and measured result are compared.

The simulated results should mirror the measured results since the same configuration was used. In both cases, the measured position of each speaker and microphone was used. Also, the transmit waveform described in Chapter 4 was used. In a way, the simulated images represent the ideal, noiseless images that could be gathered from the actual test stand. They serve as a good benchmark for comparison.

#### 7.1 Experimental Scene

A common scene was used for all experiments. The scene contains 4 transmitters, 16 receivers and three point reflectors positioned. Table 7.1 lists the measured positions of all objects. Figure 7.1 shows labeled transmitters and receivers in reference to the scene.

<i>Object</i>	<i>x</i>	<i>y</i>
Transmitter #1	2.1cm	51.0cm
Transmitter #2	49.1cm	0.0cm
Transmitter #3	-1.2cm	-50.0cm
Transmitter #4	-50.1cm	0.1cm
Receiver #1	-51.5cm	-10.7cm
Receiver #2	-49.7cm	-28.5cm
Receiver #3	-30.5cm	-49.1cm
Receiver #4	-12.2cm	-50.4cm
Receiver #5	9.8cm	-49.6cm
Receiver #6	28.7cm	-45.8cm
Receiver #7	44.3cm	-37.2cm
Receiver #8	46.6cm	-18cm
Receiver #9	49.1cm	12.9cm
Receiver #10	52.4cm	34.4cm
Receiver #11	35.8cm	48.9cm
Receiver #12	10.5cm	53.7cm
Receiver #13	-9.2cm	51.1cm
Receiver #14	-27.0cm	47.8cm
Receiver #15	-45.3	39.8cm
Receiver #16	-49.2cm	17.5cm
Reflector #1	4.0cm	-3.0cm
Reflector #2	-20.0cm	-10.0cm
Reflector #3	-8.0cm	17.0cm

Table 7.1: Measured Positions for Experiment Scene

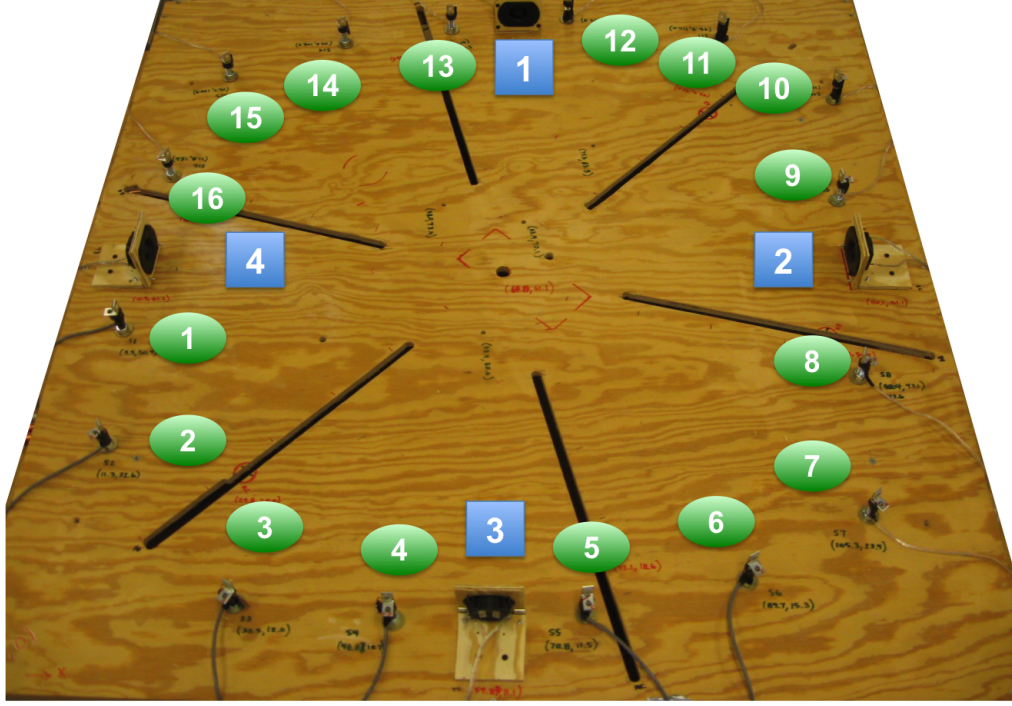


Figure 7.1: Transmitter and Receiver Labels

Simulated point reflectors are ideal points in space, producing one single echo. Measured point reflectors were 8" copper pipes of  $\frac{3}{8}$ " diameter placed perpendicular to the test stand base. Figures 7.2 and 7.3 show the experimental setup.

All images have 199 x 199 pixels and represent a scene of 0.7m x 0.7m. This was chosen for several reasons. The image size limits the number of unknowns in the image. This was a design constraint for the compressive sensing approach. By choosing the scene size to be 0.7m x 0.7m, each pixel represents very close to one sample of the range profile at 96kHz.

$$R_{pixel} = \frac{0.7m}{199pixels} = 3.52mm \quad (7.1)$$

$$R_{data} = \frac{343^m/s}{96,000^{samples}/s} = 3.57mm \quad (7.2)$$

where  $R_{pixel}$  is the width of each pixel and  $R_{data}$  is the range difference between each data sample. Since  $R_{pixel} \approx R_{data}$ , interpolation has a negligible impact on the imaging process.

All images are plotted on a dB scale using a colormap in *MATLAB*.



Figure 7.2: Measured "Point Reflectors" - Copper Pipes

This scene was used throughout all of the following tests.

## 7.2 Backprojection Results

Data was collected from all 64 different speaker/microphone pairs and the back-projection algorithm described in section 5.2 was used. A total of  $64 \times 600 = 38,400$



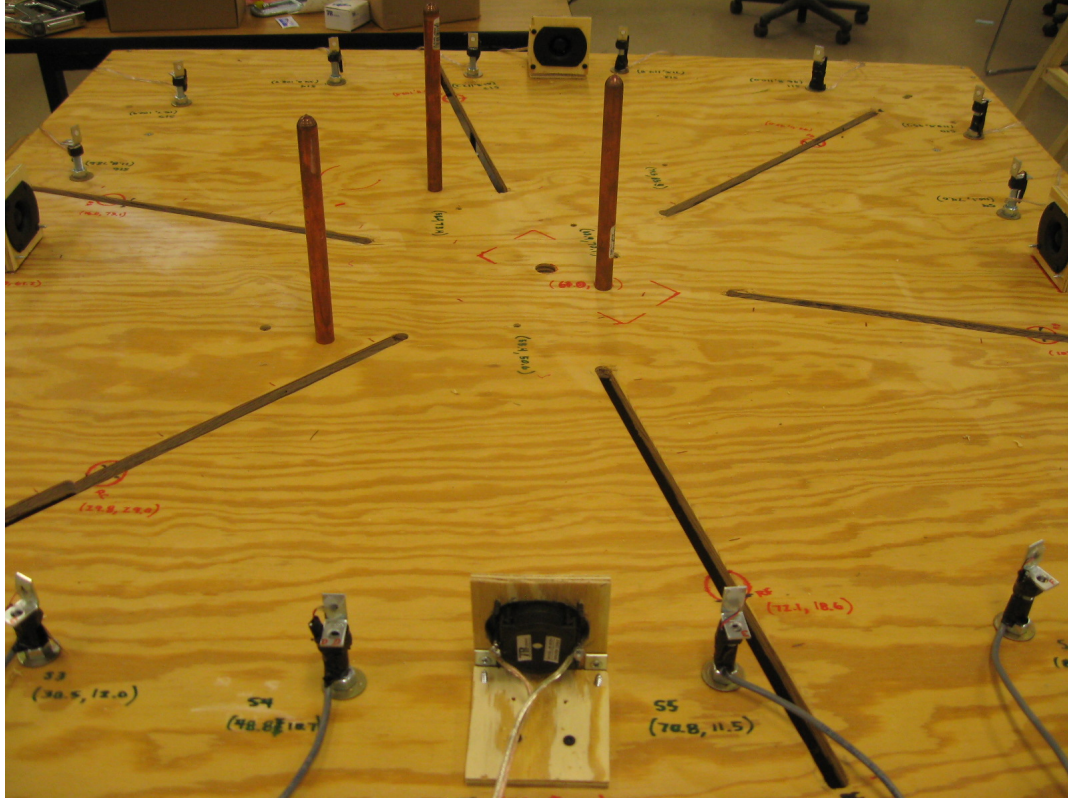


Figure 7.3: Experimental Scene for Imaging Tests

data points were used in the imaging process. Figure 7.4 shows the result of the simulated test. The three reflectors are very clearly seen in the image.

Figure 7.5 shows the result using measured data from the acoustic imaging test stand. The three reflectors are visible as the red dots, but are obscured by other artifacts in the image.

The differences between the simulated and measured backprojection images are due to the three factors described in section 5.3. The channel imbalances in the system along with the unequalized channel responses are the most likely factors. Smaller

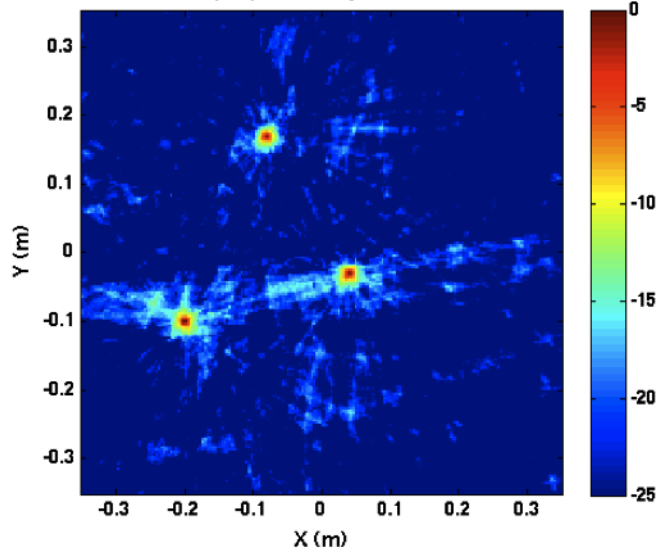


Figure 7.4: Backprojection Image of Three Reflectors (Simulated Data)

contributors are noise in the scene and multiple reflections which are not modeled in the simulation.

### 7.3 Compressive Sensing Results

The data collected from the three pipes was taken and reduced down by two methods:

1. only data from 20 of the 64 geometries was used
2. for each range profile, an effective measurement of 150 samples was used

This process left a total of  $20 \times 150 = 3,000$  measurements for the compressive sensing imaging approach. Figure 7.6 shows the simulated result using the compressive sensing approach described in chapter 6.

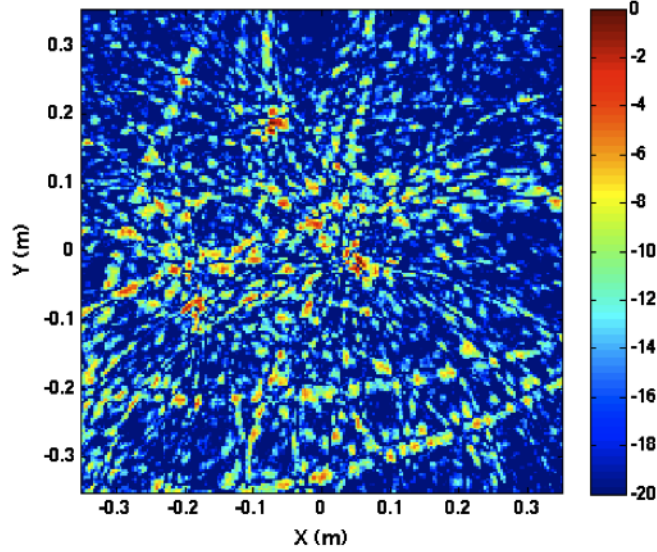


Figure 7.5: Backprojection Image of Three Reflectors (Measured Data)

The three point reflectors are clearly shown in the image without much else. Figure 7.7 shows the image using measured data through the compressive sensing approach.

Once again, the image shows the three reflectors clearly. Strikingly, the positions of the reflectors are recovered from the same data that produced such a busy image using the backprojection algorithm.

## 7.4 Analysis of Results

The results very clearly show that the compressive sensing framework can be used to create very accurate images with less than 10% of the data used by the backprojection algorithm. Since some improvements could be made to the backprojection imaging approach (already discussed) the results show that compressive sensing produce images of at least the same quality as backprojection. Figure 7.8 shows all four

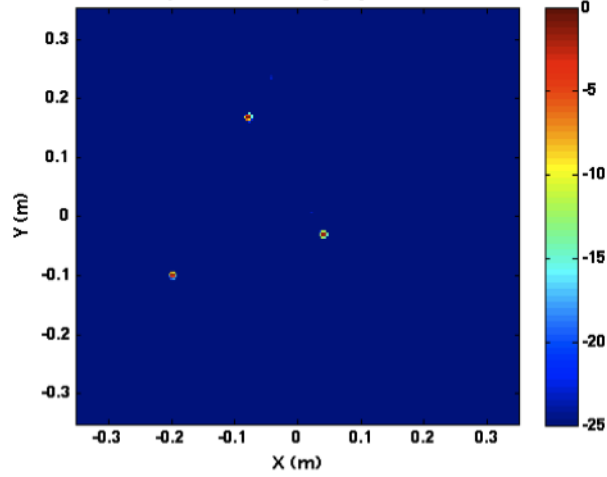


Figure 7.6: Compressive Sensing Image of Three Reflectors (Simulated Data)

	Backprojection	Compressive Sensing
Measurements Needed	38,400	3,000
Reconstruction Time	$\approx 10$ seconds	$\approx 10$ minutes
Image Quality	=	$\geq$
Complexity	Always Works	Requires CS Framework

Table 7.2: Comparison of Backprojection and Compressive Sensing

result images next to each other. The left column is simulated data while the right column is measured data. The top row is backprojection while the bottom row is compressive sensing.

Based on these results, Table 7.4 summarizes the qualities of each imaging approach.

Based on the comparison, for applications where data acquisition can be traded for processing time, compressive sensing is a very good option. If processing time is essential, backprojection or other standard imaging techniques may be more desirable.

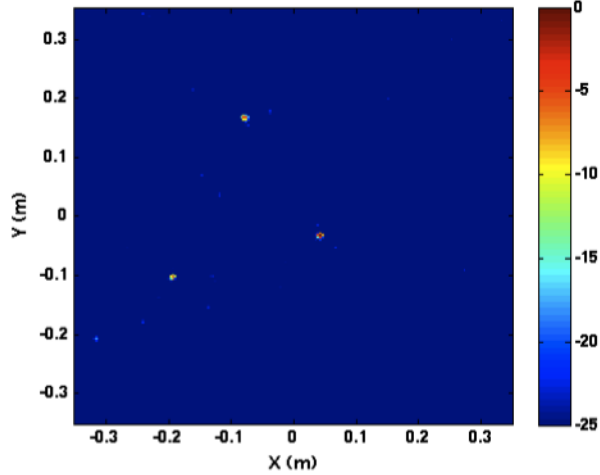


Figure 7.7: Compressive Sensing Image of Three Reflectors (Measured Data)

Many important applications tend to favor less data acquisition. For example, many radar systems have expensive A/D-D/A equipment. If cheaper equipment could be purchased, the entire system cost would go down at least linearly. Also, in many cases, the bottleneck for radar systems is the data transmission to the processing station. With compressive sensing, there is 10 times less data to transmit which means that with sufficient processing power, throughput is increased by 10 fold.

Also, in MRI applications, patient time in the machine is the limiting factor for throughput. The data is processed by computers separate from the imaging device and can be done in parallel while other patients are being imaged. This means that a reduction in data collection has a direct impact on throughput of the MRI machine. These are just two of many examples that are convincing for the benefit of compressive sensing.

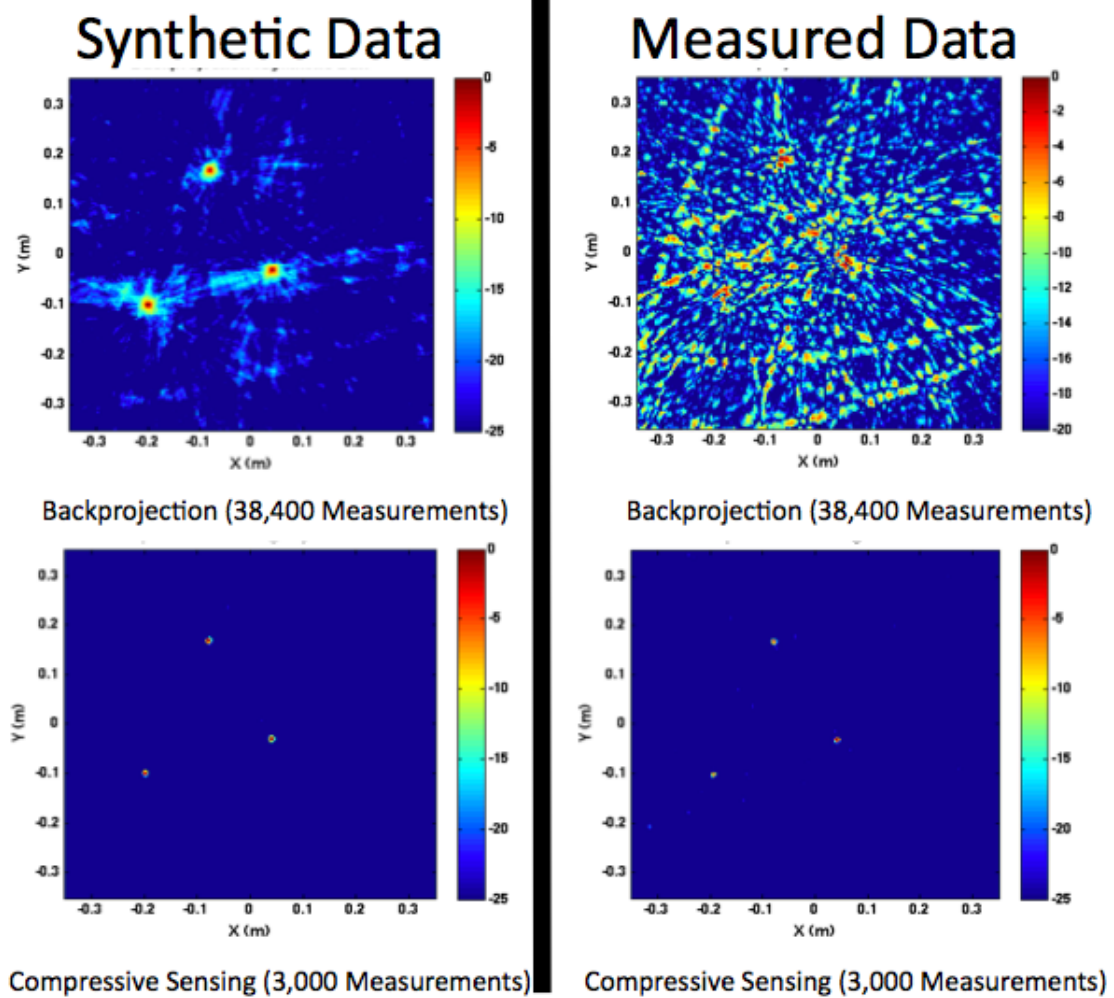


Figure 7.8: Compilation of Results

## CHAPTER 8

### FUTURE WORK

As in any project, there are items which can be done to extend the work as presented. Attention is given to two items based on the ease of implementation and the impact of the modification.

#### 8.1 Channel Balancing with Backprojection

The channel imbalance issue has been discussed several times throughout the report (see 5.3.2, 7.2). It is attributed to the primary reason for differences between simulated and measured images using the backprojection algorithm. Since the basic backprojection algorithm does not use any equalization, the image suffers.

An easy modification to the traditional imaging approach would be to construct a model from measured data to automatically apply the equalization on all channels. Similar to section 6.2, the acoustic imaging problem can be described as a linear system of equations where

$$\begin{bmatrix} \tilde{\mathbf{y}} \end{bmatrix} = \begin{bmatrix} \mathbf{A} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}} \end{bmatrix} \quad (8.1)$$

once again defining  $\tilde{\mathbf{y}}$  as the measurement vector,  $\mathbf{A}$  as the system model and  $\tilde{\mathbf{x}}$  as the unknown image.

Using the full data set, the system is roughly determined, and an approximation for  $\mathbf{x}$  can be found using the adjoint operator of the  $\mathbf{A}$  matrix.

$$x \approx A^\dagger y = A^T y \quad (8.2)$$

If the  $\mathbf{A}$  matrix is constructed in the same way described in section 6.7 except the full range profiles from 64 geometries are included, the model will fully describe the behavior due to the actual system.

Then, to recover the image, the transpose of  $\mathbf{A}$  is simply multiplied by the measurement vector  $\mathbf{y}$  to recover  $\mathbf{x}$ . Since this  $\mathbf{A}$  matrix will be larger than what can be stored in memory at once, the multiplication will be performed in a piecewise fashion, reading chunks of the matrix from memory at a time and performing the multiplication iteratively.

This method should remove the affects of channel and gain imbalances that plagued the results from the backprojection image. This will also give an important comparison to compressive sensing that removes all differences between the imaging algorithms. There can be a pure comparison of the image quality based on the measurement and processing time tradeoff.

## 8.2 Compressive Sensing using LFM Chirps

As mentioned in section 6.6, there is a major step that still needs to be taken: come up with an effective measurement that can actually be implemented in hardware



so the only the data used for the CS model is collected by the system. Right now, the full data set is collected and a subset is chosen from this set.

Existing radar systems use transmit waveforms called linear frequency modulated (LFM) chirps. These chirps are of the form

$$s(t) = \Re\{e^{j2\pi(f_c + (\frac{t}{2T} - \frac{1}{2})f_{BW})t}\} \quad (8.3)$$

where  $f_c$  is the carrier frequency in Hz,  $T$  is the duration of the pulse in seconds and  $f_{BW}$  is the total frequency range that is swept by the chirp in Hz.  $s(t)$  sweeps linearly from  $f_c - \frac{1}{2}f_{BW}$  to  $f_c + \frac{1}{2}f_{BW}$  over  $T$  seconds.

Ongoing research by [7] suggests that a signal composed of several chirps at different carrier frequencies with random starting phases could provide enough randomness to be used directly by a compressive sensing algorithm. Also, the same research suggests that the signal could be intentionally sampled at a lower rate, leading to spatial aliasing. The design of the waveform would allow for post-processing to recover the original signal from a severely undersampled signal.

This approach would bring together the work that has been demonstrated here and use it for a real application. If the idea can be shown to work, then compressive sensing could be used on existing radar equipment with minimal modifications.

## CHAPTER 9

### CONCLUSION

Compressive sensing has been demonstrated to be able to reduce the necessary data significantly for imaging problems. This work serves as a proof of principle that the theory can be made to fit the acoustic imaging framework. The relevance of decreasing data acquisition is important not only to SONAR imaging but to RADAR and potentially MRI as well.

An acoustic imaging test stand was designed and built in order to provide a platform for testing. Audio equipment was purchased, installed and tested for use on the test stand. Also, a transmit waveform was designed to give good resolution to the imaging results. Then, software was developed to take the data from the test stand and from simulated experiments and reconstruct an image using backprojecion. This proved to be an effective way to image reliably and quickly with a given data set. However, the images were subject to clarity issues. Compressive sensing theory was explained and the design and approach for acoustic imaging was explored. The software approach and implementation was detailed and the imaging results were shared.

The results showed that with the designed compressive sensing framework, images of equal or greater quality were reconstructed with only 10% of the data used for

backprojection. However, there is still work to be done to make this method implementable on a real system. Nevertheless, this work shows that compressive sensing is a promising technique that offers to greatly decrease the need for data collection in common imaging systems.

## **APPENDIX A**

### **HARDWARE DETAILS AND SPECIFICATIONS**

#### **A.1 Bill of Materials**

Table A.1 details the cost of all components of the acoustic imaging test stand. The total cost of the equipment was \$735.12.

Sections A.2, A.3 and A.4 include the manufacturer specification sheets for the A/D-D/A converter, the speakers and the microphones respectively.

<i>Item</i>	<i>Quantity</i>	<i>Unit Cost</i>	<i>Total Cost</i>
<b>Data Acquisition</b>			
TASCAM US-1641 DAQ System	1	\$219.99	\$219.99
<b>Speakers</b>			
Tang Band 25-302SH 1" Shielded Neodymium Dome Tweeter	4	\$20.90	\$83.60
Speaker Wire	80 ft.	\$0.60/ <i>ft.</i>	\$48.00
$1/4$ " mono speaker plug	4	\$3.99	\$15.96
<b>Microphones</b>			
Electret Microphone (Digikey PN: 102-1732-ND)	16	\$1.89	\$30.24
27 k $\Omega$ Resistor (Digikey PN: 27H-ND)	16	\$0.06	\$0.96
2-conductor Wire	320 ft.	\$0.60/ <i>ft.</i>	\$192.00
XLR Connector (Digikey PN: SC1003-ND)	16	\$4.47	\$71.52
<b>Mounting Equipment</b>			
$3/8$ " $\times$ 16x6" Bolt	16	\$1.91	\$30.56
$3/8$ " Washer	32	\$0.20	\$6.40
$3/8$ " Nut	32	\$0.20	\$6.40
$3/8$ " Lock Nut	16	\$0.37	\$5.92
L-Bracket	16	\$0.49	\$7.84
Plywood ( $4' \times 4' \times 1/4$ " )	1	\$7.85	\$7.85
Miscellaneous Adhesives	1	\$7.88	\$7.88

**Total:** \$735.12

Table A.1: Bill of Materials for Acoustic Imaging Test Stand

## A.2 TASCAM A/D-D/A Converter Specifications



### US 1641 Technical Documentation

#### PRODUCT OVERVIEW

Utilizing the latest high-speed USB 2.0 technology, the TASCAM US-1641 packs the interfacing power of a big console into a single rack space.



The eight mic/line inputs utilize TASCAM's Professional Mic Preamps providing 60 dB of gain with phantom power. Two, front panel, balanced TRS balanced line inputs; four, rear panel, line inputs; and a S/PDIF pair (switchable to AES/EBU) complete the US-1641's 16-channel inputs. Rear panel outputs consist of four, balanced (line level) and a left/right channel (balanced) monitor output.

**MSRP \$ 499.00**

**1-U Rack-mountable**

**Eight, TASCAM Pro Mic Pre-Amps w/phantom power.**

**Six, balanced 1/4 TRS line inputs (two on front panel)**

**Four, balanced line outputs**

**Stereo S/PDIF digital input and output**

**Headphone Out w/volume control**

**High Speed USB 2.0**

**Built in Power Supply (No wall wart!)**

**96kHz/24-bit support on all inputs and outputs**

USB 1.1: 12mbps

USB 2.0: 480mbps

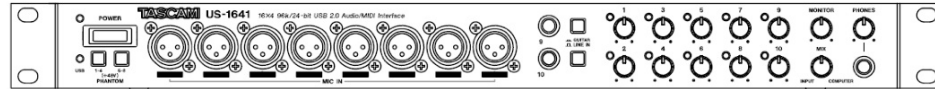
FireWire: 400mbps

TASCAM 7733 Telegraph Road Montebello, CA 90640 (323) 726-0303 <http://www.tascamcontractor.com>  
TASCAM, a division of TEAC Corporation. All features and specifications are subject to change without notice.

1

TASCAM US 1641

## PANELS



## DRIVER COMPATIBILITY

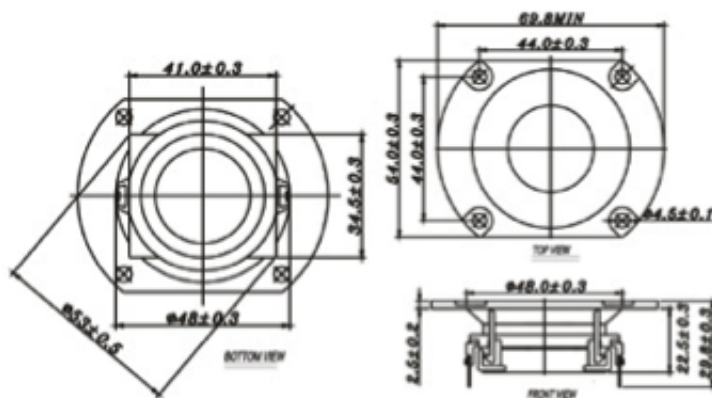
Driver Compatibility Chart		
	Windows XP	
	Windows XP 64-bit	
	Windows Vista	
	Windows Vista 64	
	Mac OS X 10.4 Tiger	
	Mac OS X 10.5 Leopard	
	Apple Universal Binary (Intel processor)	
= Compatible      = Currently Incompatible		

TASCAM 7733 Telegraph Road Montebello, CA 90640 (323) 726-0303 <http://www.tascamcontractor.com>  
 TASCAM, a division of TEAC Corporation. All features and specifications are subject to change without notice.

## A.3 Tang Band Speaker Specifications

**NS** SERIES

**25-302SH**



- SOFT FABRIC DOME DIAPHRAGM COATING WITH HIGH LOSS MATERIAL
- FERROFLUID COOLED
- DAMP WOOL IN CAVITY OF POLE PIECE
- ABS FRONT PANEL
- NEODYMIUM DESIGN
- CCAW VOICE COIL

1" FABRIC DOME / NEODYMIUM



DIAPHRAGM MTL	Fabric
SURROUND MTL	N/A
NOMINAL IMPEDANCE	8 W
DCR IMPEDANCE	6 W
SENSITIVITY 1W/1m	91 dB
FREQUENCY RESPONSE	1400-20K Hz
FREE AIR RESONANCE	1400 Hz
VOICE COIL DIAMETER	25.4 mm
AIR GAP HEIGHT	2 mm
RATED POWER INPUT	8 W
MAXIMUM POWER INPUT	80 W
FORCE FACTOR, BL	N/A
MAGNET WEIGHT ( oz)	Neodymium
MOVING MASS	N/A
FERRO FLUID ENHANCED	Yes
SUSPENSION COMPL.	N/A
EFFEC.PISTON AREA	N/A
Levc	N/A
Zo	N/A
X-max	N/A
Vas	N/A
Qts	N/A
Qms	N/A
Qes	N/A

**TB** SPEAKERS

VOICE: 886.2.26570282 FAX: 886.2.26580166  
E-MAIL: info@tb-speaker.com



## A.4 CUI Inc. Microphone Specifications



page 1 of 4

date 06/2008

**PART NUMBER:** CMC-2742WBL-25L

**DESCRIPTION:** electret condenser microphone

### SPECIFICATIONS

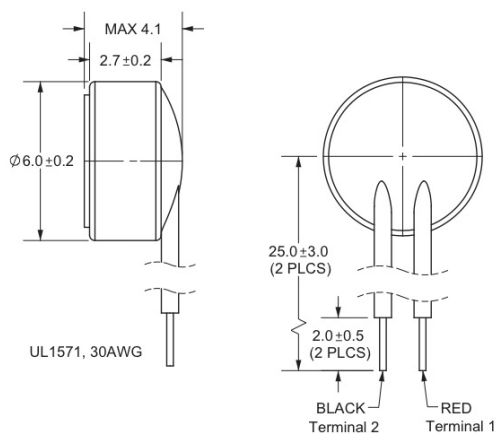
directivity	omnidirectional	
sensitivity (S)	-42 $\pm$ 3 dB	f = 1KHz, 1Pa 0dB = 1V/Pa
sensitivity reduction ( $\Delta$ S-Vs)	-3 dB	f = 1KHz, 1Pa Vs = 2.0 ~ 1.5 V dc
operating voltage	2 V dc (standard), 10 V dc (max.)	
output impedance (Zout)	2.2 K $\Omega$	f = 1KHz, 1Pa
operating frequency (f)	100 ~ 20,000 Hz	
current consumption (I <sub>DSS</sub> )	0.5 mA max.	Vs = 2.0 V dc RL = 2.2K $\Omega$
signal to noise ratio (S/N)	57 dBA	f = 1KHz, 1Pa A-weighted
operating temperature	-20 ~ +70° C	
storage temperature	-20 ~ +70° C	
dimensions	$\phi$ 6.0 x 2.7 mm	
weight	0.22 g max.	
material	Al	
terminal	wire type (hand soldering only)	
RoHS	yes	
dustproof and waterproof level	IP57	

note:

We use the "Pascal (Pa)" indication of sensitivity as per the recommendation of I.E.C. (International Electrotechnical Commission). The sensitivity of "Pa" will increase 20dB compared to the "ubar" indication. Example: -60dB (0dB = 1V/ubar) = -40dB (1V/Pa)

### APPEARANCE DRAWING

tolerances not shown:  $\pm$ 0.3mm

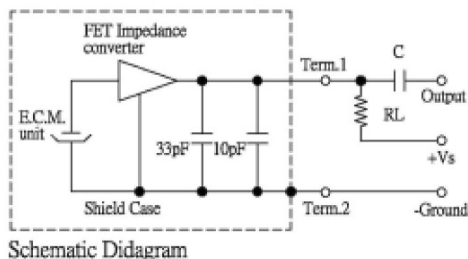


**PART NUMBER:** CMC-2742WBL-25L

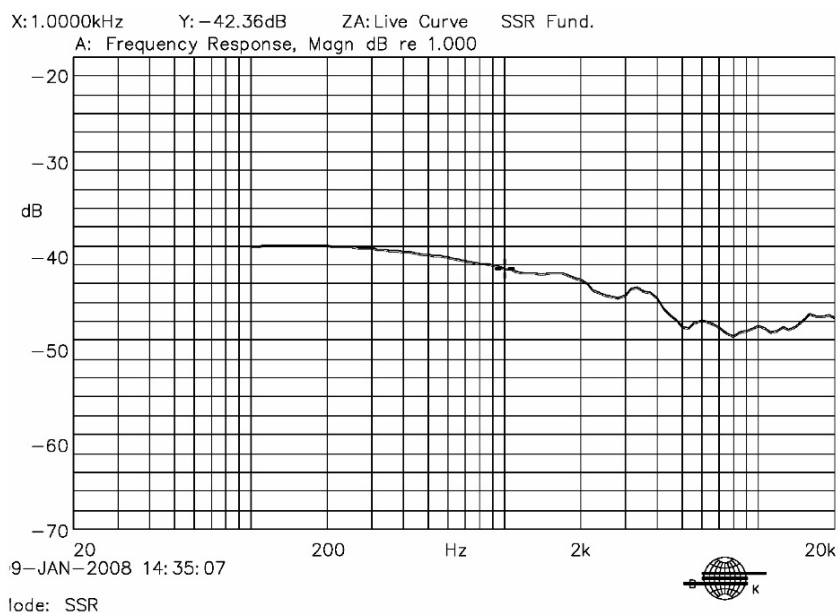
**DESCRIPTION:** electret condenser microphone

### MEASUREMENT CIRCUIT

$R_L = 2.2K\Omega$



### FREQUENCY RESPONSE CURVE



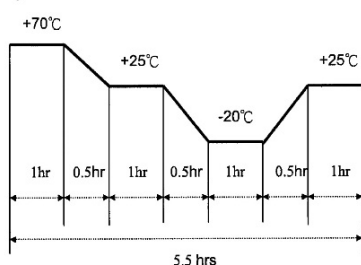
**PART NUMBER:** CMC-2742WBL-25L

**DESCRIPTION:** electret condenser microphone

### MECHANICAL CHARACTERISTICS

item	test condition	evaluation standard
solderability (connector excepted)	Stripped wires of lead wires are immersed in rosin for 5 seconds and then immersed in a solder bath of $270 \pm 5^\circ\text{C}$ for $3 \pm 0.5$ seconds.	90% min. stripped wires will be wet with solder. (except the edge of the terminal)
lead wire pull strength	The pull force will be applied to double lead wire: Horizontal 4.9N (0.5kg) for 30 seconds	No damage or cutting off.
vibration	The part will be measured after applying a vibration amplitude of 1.5 mm with 10 to 55 Hz band of vibration frequency to each of the 3 perpendicular directions for 2 hours.	After any tests, the sensitivity should be within $\pm 3\text{dB}$ compared to the initial measurement.
drop test	The part will be dropped from a height of 1 m onto a 20 mm thick wooden board 3 times in 3 axes (X, Y, Z) for a total of 9 drops.	

### ENVIRONMENT TEST

item	test condition	evaluation standard
high temp. test	After being placed in a chamber at $+70^\circ\text{C}$ for 72 hours.	The part will be measured after being placed at $+25^\circ\text{C}$ for 6 hours. After any tests, the sensitivity should be within $\pm 3\text{dB}$ compared to the initial measurement.
low temp. test	After being placed in a chamber at $-20^\circ\text{C}$ for 72 hours.	
humidity test	After being placed in a chamber at $+40^\circ\text{C}$ and $90 \pm 5\%$ relative humidity for 240 hours.	
temp. cycle test	The part shall be subjected to 10 cycles. One cycle will consist of: 	

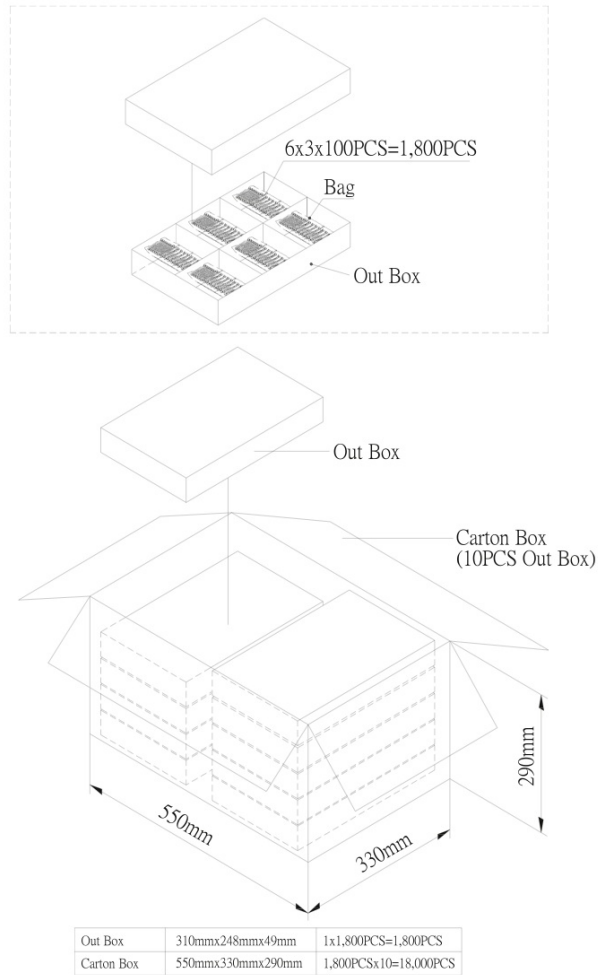
### TEST CONDITIONS

standard test condition	a) temperature: $+5 \sim +35^\circ\text{C}$	b) humidity: 45 - 85%	c) pressure: 860-1060 mbar
judgement test condition	a) temperature: $+25 \pm 2^\circ\text{C}$	b) humidity: 60 - 70%	c) pressure: 860-1060 mbar

**PART NUMBER:** CMC-2742WBL-25L

**DESCRIPTION:** electret condenser microphone

## PACKAGING



## APPENDIX B

### SOURCE CODE

The software involved in the project can be divided into three main categories: measurement collection, backprojection imaging and compressive sensing imaging. Figure B.1 shows the flow of information through the various software components from the formation of the transmit waveform all the way through the generation of the range profiles.

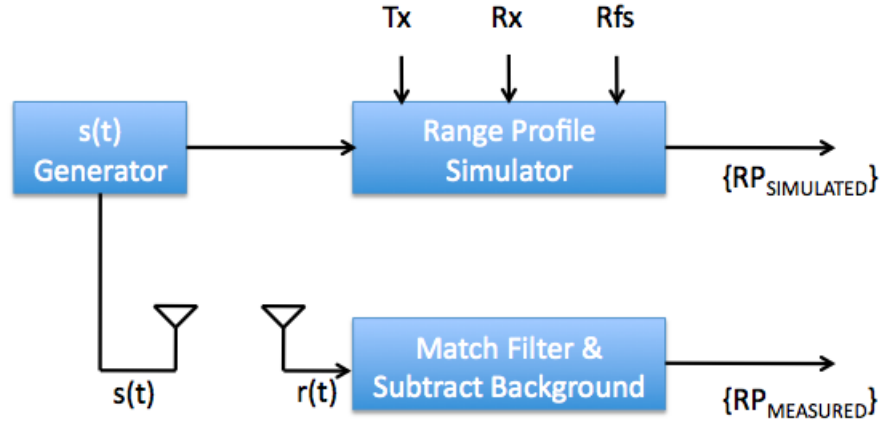


Figure B.1: Software Flow Diagram: Measurement Collection

where  $\mathbf{Tx}$ ,  $\mathbf{Rx}$  and  $\mathbf{Rfs}$  are scene parameters which describe the locations of the transmitters, receivers and reflectors respectively.  $\{RP_{MEASURED}\}$  and  $\{RP_{SIMULATED}\}$  is the set of range profiles produced for either measured or simulated data. *GenerateS.m* in section B.1 creates the  $s(t)$  waveform. *MakePNWaveform.m* and *SRRC.m* are auxiliary methods used in the process and are detailed in sections B.2 and B.3 respectively. *SimulateEchos.m* in section B.4 creates simulated range profiles using a transmit waveform and an imaging geometry. *ProcessEchos.m* in section B.5

Figure B.2 shows the flow of information through the software components involved in backprojection imaging.

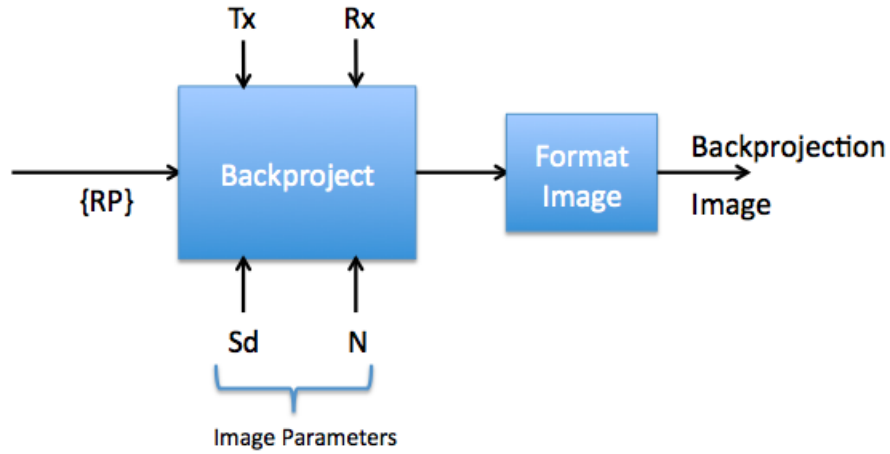


Figure B.2: Software Flow Diagram: Backprojection Imaging

where  $\mathbf{Sd}$  and  $\mathbf{N}$  are imaging parameters describing the width of the scene in meters and the number of pixels to include in the final image. *Backproject.m* in section B.6 takes the range profiles and the locations of each transmitter and receiver

and produces an image. *ShowImage.m* and *Show2DImage.m* in sections B.7 and B.8 respectively are used to format the final image.

Figure B.3 shows the flow of information through the software components involved in compressive sensing imaging.

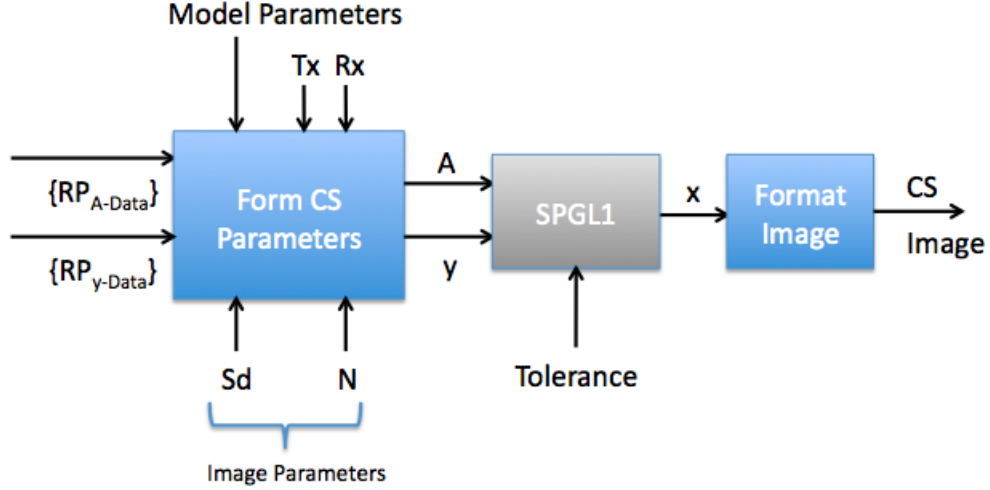


Figure B.3: Software Flow Diagram: Compressive Sensing Imaging

where *Model Parameters* are parameters describing how the effective measurement is formed from the range profiles,  $\mathbf{A}$  and  $\mathbf{y}$  are the CS model inputs representing the system measurement model and the observed measurement.  $\mathbf{x}$  is a vectorized form of the image solution. *FormEffectiveMeasurement.m* in section B.9 is a routine to transform the range profiles into the effective measurement used for modeling. *MakeCSParameters.m* in section B.10 forms the  $\mathbf{A}$  matrix and the  $\mathbf{y}$  vector from two different sets of effective measurements. The *spg\_bpdn* routine from the SPGL1

library [6] produces the image solution and ***FormatCSImage.m*** in section B.11 formats the final image.



## B.1 GenerateS.m

```
1 %%Written by Taylor Williams
2 %
3 % Code to generate the transmit waveform s(t) used in the acoustic ...
   imaging
4 % tests referenced in "SONAR Imaging using Compressive Sensing"
5 %
6 %% Output Parameters
7 %   s: A digital waveform at a 96kHz sampling rate representing a ...
   waveform
8 %       that is band-limited from 2.5kHz–30kHz and was formed from an
9 %       original PN sequence of length  $2^{16}-1$  as seen in ...
   MakePNWaveform.m
10 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
11
12 function s = GenerateS()
13
14     %Motivation: We want to create a signal that is bandlimited to ...
   roughly
15     %[2.5kHz, 30kHz] since the chosen speakers have low gains below ...
   2.4kHz and
16     %the speakers start to alias above 31 kHz. To do this, we design...
   a PN
17     %waveform using an upsampling factor of 1 and use a bandpass ...
   filter design
18
19     %generate ideal PN Code for transmission
20     [s seq sequence Ns] = MakePNWaveform(1,16,0,96000);
21
22     fs = 96000; %sample rate
23
24     L = 160;
25     fsa = 0.025;
26     fpa = 0.05; %0.05*nyquist = 2.4 kHz
27     fpb = 0.6; %0.6 *nyquist = 28.8 kHz
28     fsb = 0.65;
29     dels = 3e-5; %90dB stopband attenuation
30     delp = 0.8279; %10% passband variation allowed
31
32     %create LS FIR Bandpass Filter
33     F = [0,fsa,fpa,fpb,fsb,1];
34     A = [0,0,1,1,0,0];
35     W = [1/dels,1/delp,1/dels];
36
37     hbp = firls(L,F,A,W);
38
```

```
39     %create bandpass version of s and make sure that correlation is ...
        still
40     %sufficient. Since the original signal only has significant ...
        frequency
41     %content up to about 24 kHz, the filtering should not affect the ...
        signal
42     %very much.
43
44     s = real(ifft(fft(s).*fft(hbp,length(s))));
45
46 end
```

## B.2 MakePNWaveform.m

```
1 %%Adapted by Taylor Williams using code from Arthur C. Ludwig
2 % ...
3 % (http://www.silcom.com/~aludwig/Signal-processing/Maximum-length-sequence
4 % s.htm)
5 %
6 %%Input Parameters:
7 % T: total time for the entire symbol sequence to occur
8 % N: Symbol sequence will be of length  $2^N - 1$ . Must be in the ...
9 % range
10 % [3, 18].
11 % alpha: SRRC pulse shaping parameter. See SRRC.m
12 % fs: sampling rate of signal
13 %
14 %%Output Parameters:
15 % s: final waveform at fs sampling rate. Padded with zeros so that...
16 % the
17 % signal has a length of a power of two for FFT speed-up.
18 %
19 % seq: upsampled version of the PN code by U. Can be used to plot...
20 % s vs.
21 % seq and see exactly where the symbols match up.
22 %
23 % sequence: Maximum Length Sequence (no sampling rate). Just a ...
24 % series
25 % of symbols in  $\{-1,1\}$ .
26 %
27 % Ns: Upsampling factor in order to meet the required time. Integer
28 % value  $\geq 1$ .
29 function [s seq sequence Ns] = MakePNWaveform(T,N,alpha,fs)
30 len = T*fs;
31
32 %%generate Maximum Length Sequence of length 255 ( $2^8-1$ )
33 %code section from
34 %http://www.silcom.com/~aludwig/Signal-processing/Maximum-length-sequences.htm
35
36 %Copyright, Arthur C. Ludwig, 2001.
37 if N == 18; taps=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1]; end;
38 if N == 17; taps=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1]; end;
39 if N == 16; taps=[0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1]; end;
40 if N == 15; taps=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1]; end;
41 if N == 14; taps=[0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 1]; end;
42 if N == 13; taps=[0 0 0 0 0 0 0 0 0 1 1 0 1 1]; end;
43 if N == 12; taps=[0 0 0 0 0 1 0 1 0 0 1 1]; end;
44 if N == 11; taps=[0 0 0 0 0 0 0 0 1 0 1]; end;
45 if N == 10; taps=[0 0 0 0 0 0 1 0 0 1]; end;
```

```

42     if N == 9; taps=[0 0 0 0 1 0 0 0 1]; end;
43     if N == 8; taps=[0 0 0 1 1 1 0 1]; end;
44     if N == 7; taps=[0 0 0 1 0 0 1]; end;
45     if N == 6; taps=[0 0 0 0 1 1]; end;
46     if N == 5; taps=[0 0 1 0 1]; end;
47     if N == 4; taps=[0 0 1 1]; end;
48     if N == 3; taps=[0 1 1]; end;
49
50
51     M = 2^N-1;
52     m = ones(1,N);
53     regout = zeros(1,M);
54     for ind = 1:M
55         buf = mod(sum(taps.*m),2);
56         m(2:N) = m(1:N-1);
57         m(1)=buf;
58         regout(ind) = m(N);
59     end
60     comp = ~ regout;
61     sequence = regout - comp;
62
63     %%Create SRRC pulse for shaping
64     U = fix(len/(2^N-1)); %upsample factor
65     Ns = U;
66     g = SRRC(4,alpha,U);
67     g = g./max(g); %normalize to 1
68
69     x = upsample(sequence,U);
70     seq = [zeros(1,4*U) x];
71
72
73     %% shape pulse and create s(t) that is length of power of 2
74     s = conv(x,g);
75
76     s = [s zeros(1,2^(ceil(log2(len))-length(s)))];
77
78 end

```

## B.3 SRRC.m

```
1 %Written by Phil Schniter
2 %
3 % SRRC Creat an oversampled square-root raised cosine pulse
4 %   SRRC(N, alf, P) creates an oversampled SRRC pulse, where
5 %   N is one half the length of srrc pulse in symbol durations,
6 %   alf is the rolloff factor (between 0 and 1; alf=0 gives a sinc ...
   pulse),
7 %   P is the oversampling factor (a positive integer).
8 %   SRRC(N, alf, P, t_off) works the same way, but offsets the pulse
9 %   center by t_off fractional samples.
10
11 function g = SRRC(N, alf, P, t_off);
12
13 if nargin==3, t_off=0; end;      % if unspecified, offset is 0
14 k = -N*P+1e-8+t_off:N*P+1e-8+t_off; % sampling indices as multiples ...
   of T/P
15 if alf==0, alf=1e-8; end;      % numerical problems if alf=0
16 g = ...
   4*alf/sqrt(P)*(cos((1+alf)*pi*k/P)+sin((1-alf)*pi*k/P)./(4*alf*k/P))./...
17   (pi*(1-16*(alf*k/P).^2));
```

## B.4 SimulateEchos.m

```
1 %%Written by Taylor Williams (Last Edited: 3/23/2011)
2 %
3 % Code to Simulate ideal data from a provided geometry of transmitters,
4 % receivers and reflectors.
5 %
6 %% Input Parameters
7 %   All Distances in Meters
8 %   reflectors: row vector where each entry is a complex number
9 %               corresponding to the location of each point ...
10 %               reflector
11 %               in a 2D plane (0,0) is the center of the scene
12 %   Tx: row vector where each entry is a complex number ...
13 %       corresponding to
14 %       the location of each transmitter on a 2D plane
15 %   Rx: row vector where each entry is a complex number ...
16 %       corresponding to
17 %       the location of each receiver on a 2D plane
18 %   s: transmit waveform to be convolved to generate r
19 %
20 %   fs: sampling frequency (Hz)
21 %
22 %% Output Parameters
23 %   data: cell array where data{x,y} is the signal received by the yth
24 %         receiver from the xth transmitter. Ideal impulse response ...
25 %         assumed for
26 %         every point reflector
27 %   cdata: cell array where cdata{x,y} is the signal data{x,y} ...
28 %         correlated
29 %         against s and shifted so that the peak of the correlation is ...
30 %         at the
31 %         same location as the original impulse in data{x,y}
32 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
33 function [data cdata]=SimulateEchos(Rfs, Tx, Rx, s, fs)
34     c = 343; %speed of sound, assumed constant
35     nTx = length(Tx); nRx = length(Rx); nRfs = length(Rfs);
36
37     %calculate the distance between each reflector and Receiver, each
38     %reflector and transmitter respectively
39     for r = 1:length(Rfs) dTxR(r,:) = abs(Tx-Rfs(r)); dRxR(r,:) = ...
40         abs(Rx-Rfs(r)); end
41
42     N=nTx*nRx;
```

```

39     %Start by creating data{tx,rx} which contains an ideal impulse ...
        response
40     %at the location of each reflector. Data is positioned so that t=0
41     %corresponds to the speaker location.
42     fprintf('Simulating Data...',N);
43     for tx = 1:nTx
44         fprintf('.');
45         for rx = 1:nRx
46             %calculate the path lengths for all reflectors
47             d = (dTxR(:,tx)+dRxR(:,rx))';
48             %caculate the appropriate index corresponding to those ...
                lengths
49             n = fix(d./c.*fs);
50             %intitalize the output data array to all zeros
51             data{tx,rx} = zeros(1,max(n));
52             %add 1 for every reflector position
53             for rfs = 1:nRfs
54                 data{tx,rx}(n(rfs)) = data{tx,rx}(n(rfs)) + 1;
55             end
56         end
57     end
58     fprintf(' Complete.\n');
59
60     %%Next, pulse shape with the provided waveform correlated against ...
        itself
61     %Adjust the result so that the peak of the correlation function is ...
        at the
62     %same place as the ideal response
63     fprintf('Correlating Data...');
64
65     %generate pulse shape (s(t)*s(-t)) using ffts
66     d = real(fftshift(ifft(fft(s).*fft(fliplr(s)))));
67
68     %shape each data{tx,rx} to make the cdata{tx,rx} vector
69     for tx = 1:nTx
70         fprintf('.');
71         for rx = 1:nRx
72             %first, extend data by length(d)/2 so that we can do
73             %convolution using ffts and not have any difference in
74             %result
75             zpaddata = [data{tx,rx} zeros(1,length(d)/2)];
76             %pulse shape with d
77             cd = real(ifft(fft(zpaddata,length(d)).*fft(d)));
78             %Shift signal so that peaks line up with ideal response
79             cdata{tx,rx} = cd(length(cd)/2:length(cd));
80             %truncate arbitrarily to twice the original length
81             cdata{tx,rx} = cdata{tx,rx}(1:(2*length(data{tx,rx})));
82         end
83     end
84     fprintf(' Complete.\n');
85 end

```

## B.5 ProcessEchos.m

```
1 %%Written by Taylor Williams
2 %
3 % Process to take measured data for the background and the scene, match
4 % filter and perform background subtraction.
5 %
6 %%Input Parameters
7 %   bdata — a NxM cell array where N is the number of transmitters ...
8 %           and M is
9 %           the number of receivers. background recording with no reflectors ...
10 %          in the
11 %          scene.
12 %   sdata — an NxM cell array where N is the number of transmitters ...
13 %           and M
14 %           is the number of receivers. recording with reflectors in the scene.
15 %   s — transmit waveform at 96kHz
16 %   dmax — maximum range for echoes. Used to determine time gating.
17 %   ndelay — hardware delay in samples at 96kHz. Is applied to each ...
18 %           data
19 %           set
20 %
21 %%Output Parameters
22 %
23 %   data — an NxM cell array where N is the number of transmitters ...
24 %           and M is
25 %           the number of receivers. Each signal in data is the result of
26 %           subtracting the match filtered background from the match filtered
27 %           scene and adjusting for all delays.
28 function data = ProcessEchos(bdata,sdata,s,dmax,ndelay)
29 {
30     [nTx nRx] = size(bdata);
31
32     for tx = 1:nTx
33         for rx = 1:nRx
34             %correlate background and scene data
35             cback{tx,rx} = processData(bdata{tx,rx}',s,0,dmax/343);
36             cdata{tx,rx} = processData(sdata{tx,rx}',s,0,dmax/343);
37             %perform background subtraction
38             data{tx,rx} = cdata{tx,rx}-cback{tx,rx};
39             %account for hardware delays
40             data{tx,rx} = data{tx,rx}(ndelay:length(data{tx,rx}));
41         end
42     end
43 }
```



```

42 }
43
44 %%cr = processData(r,s,Tstart,Tmax)
45 % takes a recorded waveform r and match filters against s, and ...
    applies a
46 % shift to guarantee that the peak from the correlation occurs at the
47 % beginning of the echo.
48 %   Tstart is the experiment time delay between the start of ...
    recording and
49 % the start of transmission. Typically 0.
50 %   Tmax is a time gating parameter. The resulting match filtered ...
    result
51 %   will be truncated in time assuming a sampling rate of 96000 Hz.
52 function cr = processData(r,s,Tstart,Tmax)
53     fs = 96000;
54     c = 343;
55
56     M = length(s);
57     N = length(r);
58
59     nstart = fix(Tstart*fs);
60     nmax = fix(Tmax*fs);
61
62     if (length(r) < length(s)) r = [r zeros(1,length(s)-length(r))]; end
63
64     %correlate and shift so that the peak occurs at the first sample ...
        of the
65     %original signal
66     cr = real(ifft(fft(r,N+M-1).*fft(fliplr(s),N+M-1)));
67     cr = cr(M:length(cr));
68
69     %ignore the first Tstart seconds – given delay from start of ...
        record to
70     %time of transmit
71     cr = cr((nstart+1):length(cr));
72
73     %gate the signal in time – limit to Tmax after the time when the ...
        signal
74     %starts
75     cr = cr(1:nmax);
76 end

```

## B.6 Backproject.m

```
1 %%Written by Taylor Williams 3/23/2011
2 %
3 % Code that takes correlated time-domain recordings and produces an ...
  image
4 % through backprojection
5 %
6 % Input Parameters:
7 %   cdata: cell structure where cdata{tx,rx} is the recording from the
8 %         specified Tx and Rx.  t=0 when the sound leaves the speaker.
9 %
10 %   Tx/Rx: vector telling the positions of each transmitter and ...
  receiver
11 %         in the x-y plane using complex numbers (x+iy) in meters
12 %
13 %   L: Width of the scene to image.  Produced image will span from
14 %     [-L/2, L/2] in both x and y directions
15 %
16 %   N: Width in pixels of image to produce.  Function produces an NxN
17 %     image.
18 %
19 %
20 % Output Parameters:
21 %   pixelgrid: NxN matrix where each entry is the location in the x-y
22 %             plane of the center of that pixel.
23 %
24 %   image: NxN matrix with the result of the backprojection in the ...
  scene.
25 %         magnitudes normalized to be from 0 to 1.
26
27 function [pixelgrid image] = Backproject(cdata, Tx, Rx, L, N)
28     %Static Variables - speed of sound and sampling rate
29     c = 343;
30     fs = 96000;
31
32     nTx = length(Tx); nRx = length(Rx);
33
34     %Resample provided data vectors so that there is one point per pixel
35     % either interpolating or decimating by a rational factor
36
37     R = (c/fs)/(L/N); %resampling factor (provided delta x / image ...
  delta x)
38
39     %use interp1 to create a new interpolated data vector to fit N ...
  pixels
40     % This is alot faster to do ahead of time all in one swoop
41     fprintf('Resampling Data...');
```

```

42     for tx = 1:nTx
43         for rx = 1:nRx
44             len = length(cdata{tx,rx});
45             cdata{tx,rx} = ...
                    interp1(1:len,cdata{tx,rx}, linspace(1,len,fix(len*R)));
46         end
47     end
48     fprintf(' Complete.\n');
49
50     %make sure N is odd (number of pixels in final image)
51     %guarantees a (0,0) pixel
52     if (mod(N,2)==0) N = N-1; end
53
54     %create 2D NxN vector that contains the positions of the center ...
        of each
55     %pixel (using complex numbers for x,y coordinates)
56     pixpos = ...
        (linspace(-L/2,L/2,N)'*ones(1,N))'+linspace(-L/2*i,L/2*i,N)'*ones(1,N);
57
58     %initialize image to zeros
59     image = zeros(N,N);
60
61     %Looping through each pixel in the final image
62     fprintf('Constructing Image: ');
63     tic;
64     mark = 0;
65     for x=1:N
66         %display status
67         t=toc;
68         if (floor(t)>mark) fprintf('%d%%, ',ceil(x/N*100)); mark = ...
            mark + 1; end
69         for y=1:N
70             %examining the contribution from each geometry
71             for tx=1:nTx
72                 for rx=1:nRx
73                     %calculate the path distance from the chosen ...
                        Tx/Rx to
74                     %the pixel being examined
75                     d = abs(Tx(tx)-pixpos(x,y))+abs(Rx(rx)-pixpos(x,y));
76                     n = fix(R*d/c*fs); %determine the index for the ...
                        distance using the resampling factor
77                     %add to the pixel the value from the range profile
78                     if (n <= length(cdata{tx,rx})) image(x,y) = ...
                        image(x,y) + cdata{tx,rx}(n); end
79                 end
80             end
81         end
82     end
83     fprintf(' Completed.\n');
84
85     %normalize image

```

```
86     image = image./max(max(image));  
87  
88     pixelgrid = pixpos;  
89  
90 end
```

## B.7 ShowImage.m

```
1 %Jason T. Parker
2
3 function ShowImage(result,bounds,spacing,f_hand,title_string,cscale)
4
5 if bounds(5) == bounds(6) %2-D image in X-Y plane
6     Show2DImage(result,bounds,f_hand,title_string,cscale,'X (m)','Y ...
7         (m)');
8
9 elseif bounds(1) == bounds(2) %2-D image in Y-Z plane
10
11     %Fix the bounds
12     bounds = bounds(1,3:end);
13     Show2DImage(squeeze(result).',bounds,f_hand,title_string,cscale,'Y...
14         (m)','Z (m)');
15
16 elseif bounds(3) == bounds(4) %2-D image in X-Z plane
17
18     %Fix the bounds
19     bounds = [bounds(1,1:2) bounds(1,5:6)];
20     Show2DImage(squeeze(result).',bounds,f_hand,title_string,cscale,'X...
21         (m)','Z (m)');
22
23 else
24
25     figure(f_hand)
26     clf
27
28     %Generate the coordinate vectors
29     x = bounds(1):spacing:bounds(2);
30     y = bounds(3):spacing:bounds(4);
31     z = bounds(5):spacing:bounds(6);
32
33     %
34     %Call plotting code option 1
35     contours = linspace(cscale(1),cscale(2),20);
36     ...
37     plot3_patch(f_hand,x,y,z,20*log10(abs(result/max(abs(result(:))))),contours,linsp
38
39     %Plotting code, option 2
40     isosurface(x,y,z,20*log10(abs(result)/max(abs(result(:)))) ,cscale(1));
41     axis(bounds);
42     title(title_string);
43 end
```

## B.8 Show2DImage.m

```
1 %Jason T. Parker
2
3 function ...
4     Show2DImage(data,bounds,f_hand,title_string,cscale,x_string,y_string)
5 %This function plots a normalized image from 2d data produced by ...
6     pcmf_image
7
8 %data is a 2d matrix of raw (i.e. complex) pixel values
9
10 %bounds is the 1 by 6 vector of axis limits x,y,z ordering
11
12 %fignum is the desired figure number
13
14 %title_string is the title string of the figure
15
16 %cscale is the desired color scale
17
18 if nargin < 6
19     x_string = 'X (m)';
20     y_string = 'Y (m)';
21 end
22
23 figure(f_hand)
24 clf
25 imagesc(bounds(1,1:2),bounds(1,3:4),20*log10(abs(data/max(abs(data(:))))),cscale);
26 %imagesc(bounds(1,1:2),bounds(1,3:4),20*log10(abs(data)),cscale);
27 set(gca,'YDir','normal','fontsize',11,'fontweight','bold');
28 grid off
29 axis square
30 xlabel(x_string,'fontsize',12,'fontweight','bold');
31 ylabel(y_string,'fontsize',12,'fontweight','bold');
32 title(title_string,'fontsize',12,'fontweight','bold');
33 colormap('jet');
34 colorbar
```

## B.9 FormEffectiveMeasurement.m

```
1 %%Written by Taylor Williams
2 %
3 %quick function to take an original measurement y0 (Nx1) and create ...
4 %an effective
5 %measurement y (Nx1) by putting y0 through the provided linear ...
6 %transformation TM
7 %and keeping only the samples in keepsamples
8 %
9 %TM must be NxN where N is the length of y0
10 %
11 %keepsamples must contain only integer values in [1,N]
12
13 function y = FormEffectiveMeasurement(y0, TM, keepsamples)
14     y0 = TM*y0;
15     y = y0(keepsamples);
16 end
```

## B.10 MakeCSParameters.m

```
1 %%Written by Taylor Williams
2 %
3 %A function that constructs an A matrix and a y vector from two separate
4 %data sets. Input parameters specify exactly how to do this.
5
6 %%Input Parameters
7 % Let T = number of transmitters in full data set
8 % R = number of receivers in full data set
9 %
10 % adata: a TxR cell array containing range profiles from a centered...
11 % point
12 % reflector. Data used to construct the a matrix
13 %
14 % ydata: a TxR cell array containing range profiles from the unknown
15 % scene. used to construct the y measurement vector.
16 %
17 % dataparams.G - number of data sets to use (must be <= T*R)
18 % dataparams.g - a 2xG matrix where g(:,i) contains [tx rx]' where...
19 % tx is
20 % the transmitter number (from 1 to T) and rx is the receiver ...
21 % number
22 % (from 1 to R). This matrix describes exactly what data to use.
23 % dataparams.gpos - a 2xG matrix where gpos(:,i)
24 % contains [ptx prx]' where ptx and prx are complex numbers ...
25 % with the
26 % location of the transmitter or receiver in the 2D plane ...
27 % (x+iy) in
28 % meters.
29 % dataparams.nmax - maximum cutoff length for range profiles. (600 ...
30 % used
31 % throughout thesis research.)
32 % dataparams.TM - a transformation matrix used to make the effective
33 % measurement. Must be nmax by nmax. (randn(600,600) used ...
34 % throughout
35 % research)
36 % dataparams.keepsamples - a 1xG cell array where each entry is a row
37 % vector of the same length (<nmax) listing the integer valued ...
38 % entries of
39 % each effective measurement to keep. (for research, these were
40 % pre-determined randomly as 150 samples out of the 600 for each
41 % geometry) - All cell elements must be vectors of the same ...
42 % length!!
43 %
44 % imageparameters.L - length in meters of the unknown scene. ...
45 % Assumed to
46 % be square (LxL meters)
```



```

37 %   imageparameters.N – number of pixels along one edge of the unknown
38 %       scene.  Final modeled image is NxN pixels square.
39 %
40 %%Output Parameters
41 %   A – MxN matrix (M = (number of geometries)(length of effective
42 %       measurement) and N = (imageparameters.N)^2
43 %       Each column of A is the stacked ideal measurement of all G
44 %       geometries based on the adata provided.  See below for more
45 %       algorithmic details.
46 %
47 %   y – an Mx1 vector representing the effective measurement of the ...
    imaged
48 %       scene.
49 %
50
51
52 function [A y] MakeCSParameters(adata, ydata, dataparams)
53     fs = 96000;
54     c = 343;
55
56     L = imageparams.L; %square image dimension (meters) centered at ...
        (0,0)
57     N = imageparams.N; %square image pixels (NxN image)
58
59     G = dataparams.G;
60     gpos = dataparams.gpos;
61     nmax = dataparams.nmax;
62
63     keepsamples = dataparams.keepsamples;
64     T = dataparams.TM;
65
66     %length of each effective measurement
67     nEff = length(dataparams.keepsamples{1});
68
69
70     %make cell vector for just the used data (maps from {tx,rx} to {g})
71     for k = 1:dataparams.G
72         Adata{k} = adata.data{dataparams.g(k,1),dataparams.g(k,2)};
73         if (length(Adata{k})<dataparams.nmax) Adata{k} = [Adata{k} ...
            zeros(1,length(dataparams.nmax-Adata{k}))]; end
74         Ydata{k} = ydata.data{dataparams.g(k,1),dataparams.g(k,2)};
75         if (length(Ydata{k})<dataparams.nmax) Ydata{k} = [Ydata{k} ...
            zeros(1,length(dataparams.nmax-Ydata{k}))]; end
76     end
77
78     %%Form the A matrix
79     %caculate the offset required for each component of the A matrix
80     %create a 2D matrix with the positions in meters of each pixel
81     pixpos = ...
        (linspace(-L/2,L/2,N)'*ones(1,N))'+linspace(-L/2*i,L/2*i,N)'*ones(1,N);
82

```

```

83         centerDistance = abs(gpos(:,1))+abs(gpos(:,2));
84     for g = 1:G
85         xydelay{g} = ...
            fs/c*((abs(gpos(g,1)-pixpos)+abs(pixpos-gpos(g,2)))-centerDistance(g)
86         pixvec = xydelay{g}(:,1);
87         for n = 2:N
88             pixvec = [pixvec; xydelay{g}(:,n)]; end
89         delay{g} = pixvec;
90     end
91
92
93     %%Apply Time shift in frequency domain and construct matrix using
94     %%sample points
95
96     A = zeros(G*nEff,N^2);
97
98     for g = 1:G
99         FreqData{g} = fft(Adata{g}(1:nmax),nmax);
100     end
101
102     indexes = (nEff*(0:G))+1;
103
104     df = 1/(nmax*1/fs);
105     f = df*(1:nmax);
106
107     for pixel=1:N^2
108         if (mod(pixel,fix(N^2/100))==0) fprintf('%d ',pixel); end
109         for g = 1:G
110             tau = round(delay{g}(pixel))/fs;
111             e = exp(j*2*pi.*f*-tau);
112             %apply shift in time in freq domain
113             FreqD = e.*FreqData{g};
114             %sample based on input parameters from user
115             datapoints = T*real(ifft(FreqD))';
116             datapoints = datapoints(round(keepsamples{g}));
117             %normalize measurement
118             datapoints = datapoints./max(datapoints);
119             %place into the empty A matrix
120             A(indexes(g):(indexes(g+1)-1),pixel) = datapoints;
121         end
122     end
123
124
125     %% create y vector of measurements
126     indexes = (nEff*(0:dataparams.G))+1;
127     for g=1:dataparams.G
128         datapoints = (dataparams.TM*Ydata{g}(1:dataparams.nmax))';
129         datapoints = datapoints(round(dataparams.keepsamples{g}));
130         datapoints = datapoints./max(datapoints);
131         y(indexes(g):(indexes(g+1)-1),1) = datapoints;
132     end

```

133

134 `end`

## B.11 FormatCSImage.m

```
1 %%Written by Taylor Williams
2 %
3 %simple method to take a vectorized image x (containing N columns of N
4 %pixels stacked consecutively) back into an NxN image
5
6 function image = FormatCSImage(x,N)
7     indices = ((0:N)*N)+1;
8     for col = 1:N
9         image(1:N,col) = flipud(x(indices(col):(indices(col+1)-1)));
10     end
11 end
```

## BIBLIOGRAPHY

- [1] D. Donoho, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 1289–1306, 2006.
- [2] E. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 489–509, 2006.
- [3] E. Candès, “The restricted isometry property and its implications for compressed sensing,” *Comptes rendus-Mathématique*, vol. 346, no. 9-10, pp. 589–592, 2008.
- [4] D. Donoho and J. Tanner, “Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing,” Department of Statistics, Stanford University, Tech. Rep., 2009.
- [5] E. van den Berg and M. P. Friedlander, “Probing the pareto frontier for basis pursuit solutions,” *SIAM Journal on Scientific Computing*, vol. 31, no. 2, pp. 890–912, 2008. [Online]. Available: <http://link.aip.org/link/?SCE/31/890>
- [6] “SPGL1: A solver for large-scale sparse reconstruction,” June 2007, <http://www.cs.ubc.ca/labs/scl/spgl1>.
- [7] E. Ertin, L. Potter, and R. Moses, “Sparse target recovery performance of multi-frequency chirp waveforms,” 2011, unpublished.