Doctoral Dissertations and Master's Theses

Spring 4-4-2017

Application of Neural Networks to Acoustic Localization

Stephen Cronin
Embry-Riddle Aeronautical University

Follow this and additional works at: https://commons.erau.edu/edt

Part of the Electrical and Computer Engineering Commons

EMBRY-RIDDLE AERONAUTICAL UNIVERSITY

MASTERS THESIS

APPLICATION OF NEURAL NETWORKS TO ACOUSTIC LOCALIZATION

Author:
Stephen Cronin

Supervisor:
Dr. Brian Butka

A thesis submitted in partial fulfillment of the requirements for the degree of

Masters of Science in Software Engineering

in the

College of Engineering

Department of Electrical, Computer, Software and Systems Engineering
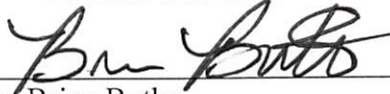
April 4th, 2017

SC

APPLICATION OF NEURAL NETWORKS TO ACOUSTIC LOCALIZATION

By

Stephen Cronin

This thesis was prepared under the direction of the candidate's thesis committee chairman, Dr. Brian Butka, Department of Electrical, Computer, Software, and Systems Engineering, and has been approved by the members of his thesis committee. It was submitted to the Electrical, Computer, Software and Systems Department and was accepted in partial fulfillment of the requirements for the Degree of Masters of Science in Software Engineering.

THESIS COMMITTEE:

Dr. Brian Butka
Committee Chairman

Dr. Eric Coyle
Committee Member

Dr. Jianhua Liu
Committee Member

Dr. Timothy Wilson
Department Chair, ECSSE

Dr. Maj Mirmirani
College Dean

Dr. Christopher Grant
Vice Chancellor

2

SC

Acknowledgments

I would like to thank my advisor, Dr. Butka, for all the help and insight he provided while

working through both the problem addressed and the writing of this thesis. To my

parents, thank you for your support through all my education to get me to this point. And

to the robotics association, thank you for allowing me to use of the hardware that was

otherwise unobtainable for the project. Finally, I would like to thank the Office of Naval

Research, whom provided partial support by ONR grant #N000141512746.

Table of Contents

SC

Figure of Figures

Figure of Tables

Definitions or Nomenclature

DAQ – Data Acquisition Device

MLP – Multi-Layer Perceptron

TDOA – Time Difference of Arrival

Abstract

The intent of the work conducted was to build a neural network for the purposes of acoustic localization. The target of this localization is a sound source underwater. For our purposes, it is an acoustic pinger, as it produces consistent sound at a fixed rate making it ideal for testing. The network was intended to ingest raw data streams and output location information based on the arrangement of sensors employed. To achieve an accurate network, a simulation factoring in the environment was to be created to produce a data set large and diverse enough to describe the unique parameters of the signals, including: frequency, environmental reflections, and range.

This problem will be approached in multiple steps. Initial models will consider simplified problem spaces, such as individual frequencies and less descriptive training sets. Through development, this will be refined and extended. Where required, simplifications will be kept managing the scope of the problem to allow for a demonstration of the technology to be made at all. Discussion of what is the root cause of the issue navigated will be presented when this occurs. Results will then be shown to demonstrate the performance of the network created as compared to the classical approach to this problem, time difference of arrival.

This paper will demonstrate the performance of a neural network as applied to the problem of acoustic localization. The network developed can accurately localize an acoustic sound source to the same order of magnitude of accuracy and execution time as the current approaches to the problem. However, the network also showed a lacking in

SC

some areas of robustness due to training factors not considered, hampering the full

potential.

Introduction

Neural Network Overview

Neural networks are a computational model based on the basic structure of how the brain processes information. They are a subset of the field of machine learning, and have applications in various fields of machine learning. In a simplified discussion of the underlying operation of the model, a model consists of an input, an output and convolutional layers. These convolutional layers are where computation takes place, and depending on the intended outcome of the model, the number and function of these layers can dramatically differ. Each individual layer is comprised of one or more neurons, or a single operation on the data input. It should be noted that a multi-neuron layer is processing the same input data across each neuron, and is simply applying different operations to the data. The intent of this is to convert the same data into unique layer outputs, leading to different unique inputs for following convolution layers.

The primary application of neural networks currently is in object classification, specifically in the space of vision processing. This is a complex space to operate in because of the numerous factors that can impact a classification, including changing environments and partial frame detection. However, even with these challenges, when there is sufficient data to train on, neural networks have shown robust to these factors and have demonstrated accuracies exceed classical approaches to this problem space [1].

Acoustic Localization Overview

Acoustic localization is a branch of signal processing focusing on determining either the planar bearing or spatial location of a sound source. It has applications in multiple areas

SC

of navigation, with the primary focus in marine environments. In water, sound waves can propagate for incredibly long distances. In subsurface navigation, vehicles cannot receive other forms of communication traditionally used in navigation, namely GPS increasing the utility of this approach. There are many approaches to acoustic localization, however they all operate on the same base principles. Devices that can measure pressure disturbances caused by sound waves are arranged in a fixed pattern and are used to sample sound data in an environment. The specifics of this pattern depend on the operating environment, dimensionality of localization output (namely 2D or 3D) and the frequency range in operation. The arrangement used for our purposes will be discussed in a later section. As the geometry of this arrangement is known, a relationship between the timing of when a sound wave reaches individual sensors in the array can be used to compute information about the originating location of the sound, more specifically, the planar bearing or the alignment angles and distance for 2D and 3D localization respectively.

Although acoustic localization can excel in certain scenarios, the use of acoustics to localize is not without flaws. Especially in subsurface environments, the noise from sources such as reflections and vehicle propulsion systems can make it quite difficult to both acquire clean data and subsequently process it to determine the location of the sound source. This is due to the much higher speed of sound in water that in air, 1480 m/s versus 343 m/s, or roughly 4.3 times faster. What this speed results in is that the data used for processing and the data originating from the sound signals reflecting off the environment arriving at the array in rapid succession. Reflections are problematic as they

SC

corrupt the data being sampled, and as such prevent the correct origination information from being computed. Figure 1 shows one of the primary impacts of reflection, phase shift. The primary data channel to be noted in this plot, is the sinusoid of lower amplitude. As can be seen across the time scale, the phase of the signal is drifting, which would result in inaccurate results under further processing as it causes the predetermined relationship of array geometry and timing to be invalidated. When talking about the high-level solution to localization, timing between sensors was discussed. As the speed of sound in water is relatively constant, especially over the span of an array, the phase difference of two signals is directly proportional to the time difference of arrival of the signals.
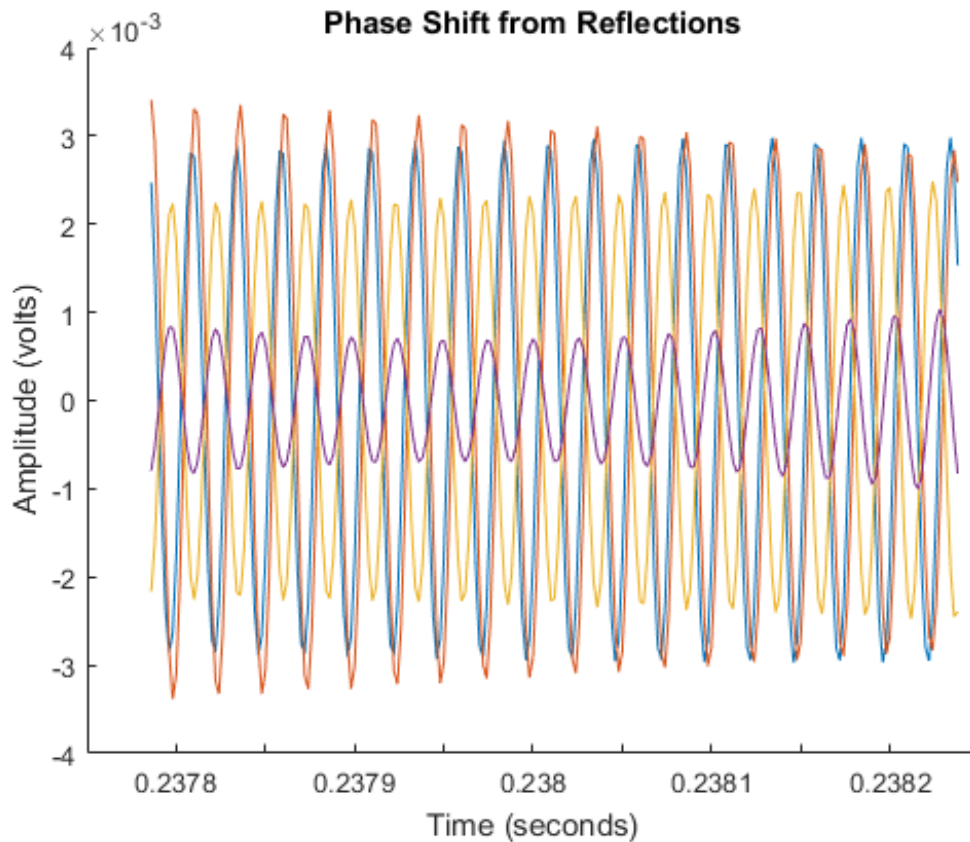

Figure 1: Phase Shifts in Reflections

SC

The primary means of attempting to address reflections is to filter them out. When processing acoustic signals, the data most likely to be free of reflections is in what is sampled when the sound waves first reach the array, the front of the waveform. This is because the path a reflection must take to reach the sensors must be longer than the path taken by the sound waves directly traveling to the sensors, thus giving a period of uncorrupted data.

Acquisition of this clean data has traditionally been a multi-step process of filtering signals. These signals are the sampling of the voltage output of the sensors in the array taken over time, and is the raw data comprised of everything in the environment producing or reflecting sound. As seen in Figure 2, we can see a sample plot of raw data captured by an array. The spike in amplitude represents where a pulse of sound from the source of interest occurs, but it is obfuscated by the little separation from the noise floor and by multiple additional frequencies clouding the data. In a specific case of filtering for a target of known frequency, a bandpass is applied around the target frequency, removing the frequencies we are not concerned with from the data. When the frequency is not known, approaches such as applying a Fast Fourier Transforms (FFT) to the raw data to determine an outlier in frequency, typically the source. This processing can be done in conjunction with other knowledge about the system to improve source identification results. With the frequency identified, a bandpass filter can again be applied. It is important to note about the use of bandpass filters overall however, that this does not remove reflections from the data, as they are still at the same or very similar frequency,

SC

but have phase shifts applied to them. The following steps taken will serve to rectify their presence in the data.
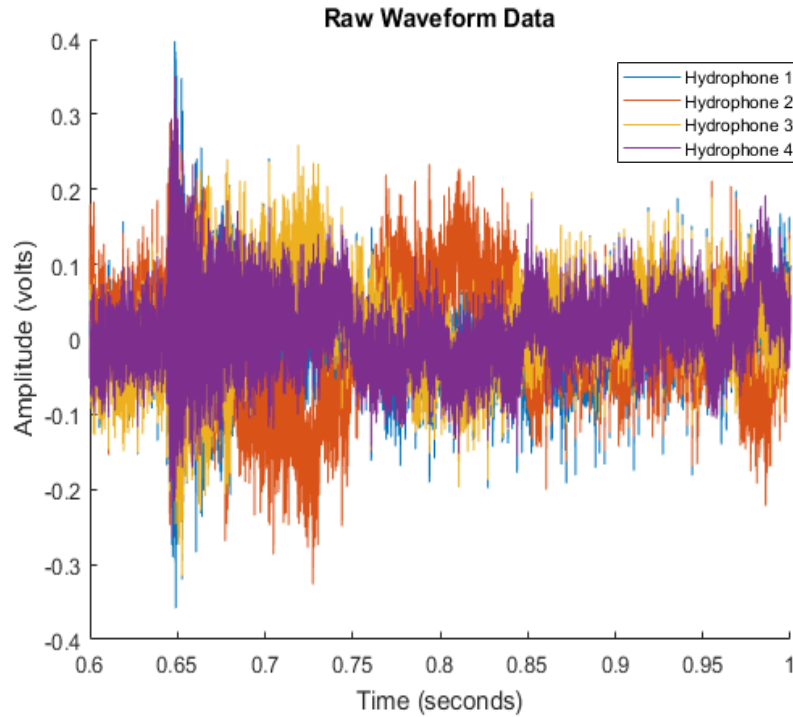
Figure 2: Raw Waveform Data. Four Channels, 40 kHz

The next step of processing begins with Figure 3, which shows the effect of the bandpass on the data. As can be seen, the random noise has been reduced, but the data shown after the initial rising edge cannot be guaranteed to not be comprised of data from reflections, and this data is treated as corrupted as it is highly likely there is a phase offset in at least one of the signals, which would yield incorrect calculations later in the processing. As a note to prove the presence of phase shifts, Figure 1, discussed above, is a small segment of the data trailing the front of the waveform in Figure 3. To avoid reflections, the last step in filtering is to determine the front of the waveform. This can be accomplished in a variety of ways, with the simplest implementation being a rising edge detector and thresholding to a minimum amplitude. The first value that meets this criterion in a time

SC

period is considered to be the front of the waveform. This approach works best when the array is further away from sources of reflection as it provides more clean signals to process. As can be seen in the filtered data, there are still small remnants of reflections from previous pulses prior to the start of the waveform. This necessitates the rising edge detector, as a threshold may falsely identify old noise as the start. The rising edge detector requires a minimum slope of the rise to counteract this, a feature dissipating noise does not match. This rising edge detectors allows for a more aggressive threshold to be employed, ensuring that the signal portion to be extracted is as close to the exact front of the waveform as possible.
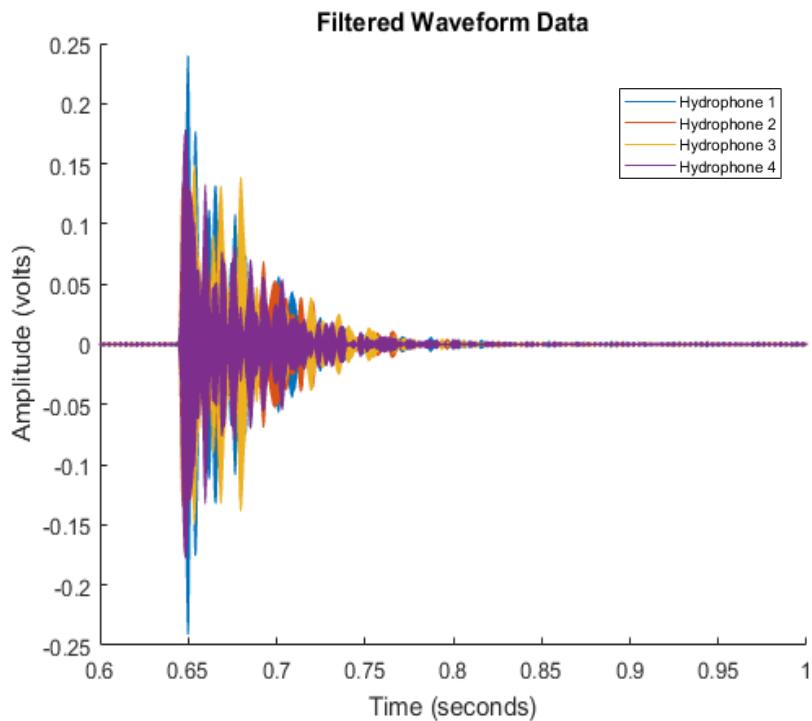
Figure 3: Filtered Data. Four Channels, 40 kHz.

With the front of the waveform determined, the data will be extracted for processing, illustrated in Figure 4. As can be seen in the signals, a feature indicative of the start of the waveform is a growth rate in amplitude, useful in the previously discussed detection

SC

method. When processing this data, the mathematics of two and three-dimensional localization differ, however they operate on the same core premise. As the geometry of the hydrophone array is known and fixed, the time difference of arrival between sensors in the array can be used to calculate bearings. For two dimensions, this result would be a single bearing in the plane of the array arrangement pointing to the source. In three dimensions, the output is a vector with three reference angles to the source and the distance to the source. These bearings are then placed in a system of equations. To fully solve, there is a fourth equation relating these bearings to the distance of the target. This system then outputs the x, y, z and distance to the target within a local reference frame. These equations are excluded as a full analysis of this form of localization will not be included in this paper.

Figure 4: Waveform Front. Four Channels, 40 kHz.

SC

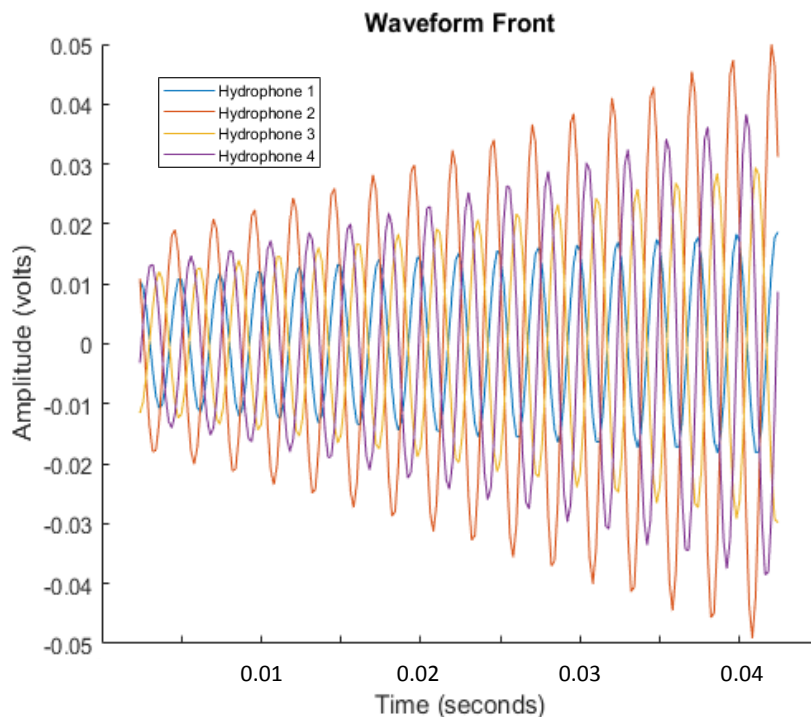The processing discussion above highlights some of the fundamental problems with the current approach to acoustic localization, the largest of which being the complexity of locating the noncorrupted data in the waveforms. From these issues comes the method investigated in the rest of this paper, the application of neural networks to signal processing. Neural networks have demonstrated much success in varying areas, especially in the image processing field as of late, and have demonstrated robustness in adapting to changing environments. Rudimentary work in the application of machine learning techniques to signal processing has already happened, and will be discussed in the Literature Review. This work however differs fundamentally from the problems they addressed as their focus was on localization problems dramatically different operating speeds and frequencies, as they were looking at wireless network signals. This work also addressed the problem without consideration of any noise in the system, a crucial factor in producing a general solution. The problem this work seeks to answer is whether a neural network can be created for localization of a single source, that is both functional in two dimensions as well as being scalable to three dimensions and capable of handling noise in the system.

Problem Overview

As discussed in the abstract, the question that arises from looking at acoustic localization and the prior successes in neural networks, to be discussed in the Literature Review, is whether the technology can be adapted a new problem space. Just as in the field of image processing, signal processing for localization requires large quantities of data to be processed rather quickly. It also shares in the fact that operational environments are not

static and robustness to this change is necessary to maintain accuracy. Just as image processing needs to account for variances such as changes in lighting, sound signals can vary dramatically due to external noise sources or reflections, two problems that will be discussed in greater detail further on in this paper. Robustness to these factors comes from the ability to accurately model the signals accounting for these factors, as it is infeasible to collect enough descriptive data from the actual recordings. From this simulation, a neural network can be trained with data that is truly descriptive of the inputs it will be receiving in an operational environment.

It is not to say that the problem spaces are identical, however. Although both problem spaces deal with high data requirements, the nature of data acquisition is fundamentally different. Images produce large amounts of data at a fixed, slower rate, whereas signal sampling relies on sampling incredibly small amounts of data at incredibly high rates to be able to resolve high frequency waveforms at resolutions that allow for processing. The other major deviation is the nature of how data can change. Throughout the paper, these differences will become apparent in the differing approach to the design and implementation of neural networks.

Literature Review

As previously mentioned, the prior work on this subject focused on two-dimensional localization utilizing a time difference of arrival (TDOA) solutions and neural networks. A visualization of the time difference of arrival is shown in Figure 5, describing the approach taken by a TDOA analysis in ultra-wide band sensor network [2]. In this

SC

approach, multiple sensors are used to compute the location of an object based upon the signal reflections on the object being tracked. These sensors are arranged in a manner not unlike that employed by acoustic localization, the design of which will be discussed later. The limitation of this comes in networks with unknown or constantly changing numbers of objects in the environment. As such, this algorithm does not provide a generic solution to localization in an unknown environment. However, even with its lackings, it does provide insight into training the network neural network. The network model they employed was that of a multi-layer perceptron (MLP). The algorithm was tuned by utilizing a least squares error estimate. This error estimate compared the actual location of the tracked object and the output produced by the model. Through back-propagation, the model can be tuned to reduce error. This network modeling approach is not unique to this one implementation, numerous of the researched papers utilized MLP networks for training the networks. Each of these papers determined this network as the optimal choice after considering multiple network models [3] [4] [5].
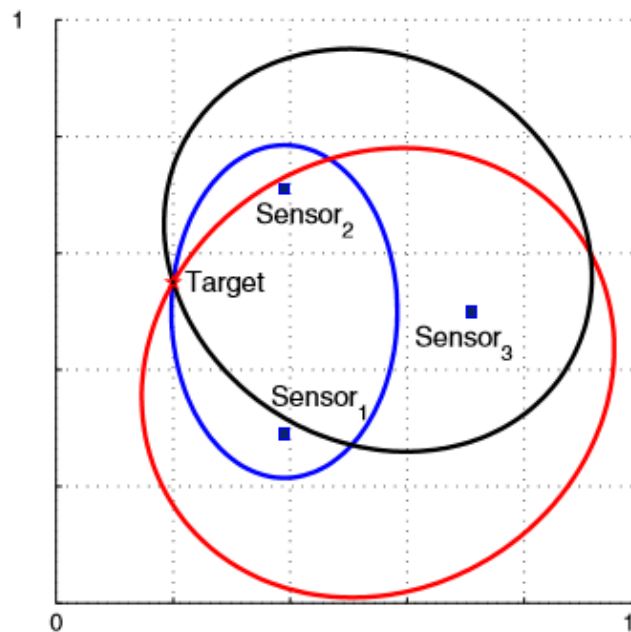

Figure 5: Sensor arrangement for TDOA calculations.

SC

Regarding the simulation of data sets, the current simplified approach to generate data of the time difference at each sensor is through calculating the distance between target and sensor and dividing it by the speed of the signal [6]. This type of simulation serves to illustrate at a rudimentary level that it is possible to generate data of simulated waveforms in large enough quantities to be used to accurately train a neural network, as they were able to achieve accurate results in the models tuned for their specific cases. The issue that arises from this simulation technique is that it oversimplifies the model and leads to inaccuracies in the model when compared to the real world. Moving beyond simplified data set generation, more advanced models account for many more problem specific factors, such as the creation and refinement of meshes defining areas where individual calculations about the properties of the signal are computed [4]. Mesh refinement techniques serves to both more accurately describe the behavior of the signal while allowing for the computational requirements of the system. Shown below in Figure 6 is a hexagonal mesh definition that was deemed more optimal for larger outdoor spaces. Based on the success found in utilizing meshes, a reduction of error of approximately 20

SC

percent was seen in experimental testing, this technique was selected and its intended

implementation is discussed in the methodology section below [4].
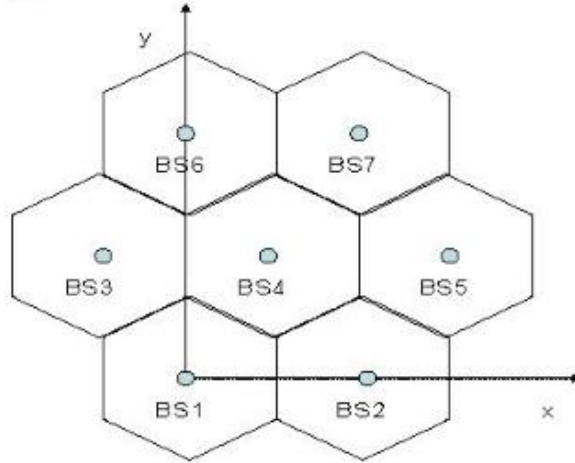
Figure 6: Hexagonal mesh refinement.

Additional work was reviewed for this thesis but not presented in this section for

purposes of brevity or as it provided information of the problem space not needed by the

reader. [7] – [11].

Hardware

Three main components are used in the process of data acquisition. The sensor array is

comprised of four Teledyne TC-4013 hydrophones. For two-dimensional localization,

they are arranged in a diamond shape of equal side lengths, with the fourth hydrophone

serving as a cross check to validate answers and can be seen in Figure 7. For three

dimensional arrays, a triangular pyramid of equal side lengths is employed. The specific

sizing of array design will be discussed later in this paper. Data from the hydrophone

array is first passed through an op-amp circuit, which in our cases provides a 19-dB gain

SC

to the signals being measured. Power for this circuit is provided through a pair of 9 volt

batteries, which provide an isolated power source free of noise. This circuit serves to

bring the output voltage range of the hydrophones to be in line with the +/- 10 volt input
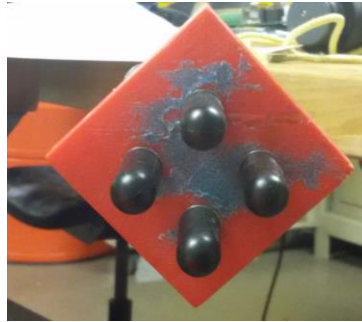
Figure 7: Hydrophone Array

range of the data acquisition (DAQ) device, the final major hardware component. This is

the device that samples the voltage of the hydrophones and sends the data to a computer

for processing. For our purposes, this DAQ is an NI-9222, providing 500

kSamples/s/Channel with each channel being synchronized.

Simulation

Before a neural network can be constructed, a large and diverse dataset for training must

be created. As this is infeasible or impossible to do on a network describing a large

enough diversity of cases, a simulation was created to generate the necessary data. This

simulation was created in MATLAB and employed an object-oriented style. Each of the

hydrophones and the sound source were generated as discrete objects, allowing for

specific parameters to be easily and dynamically changed on them. The sound source was

the primary object whose properties were changed, with the position and frequency being

updated to produce data over the range of headings.

SC

The design decisions behind the simulation partially stemmed from the next step in the localization process, that being three-dimensional localization. When first constructed, the type of array to be used for that case was not known. Multiple arrangements can be employed based on the intended localization problem. In the application of the simulation, the arrangement positioning for the hydrophone objects were not dynamically changed, this design allowed for the array to easily be changed if multiple arrangements were desired in a single dataset. Although this was not used in the implementation discussed, it is a further capability of the system that would allow for possible extensions such as an algorithm agnostic of the array that it is receiving data on, provided it operates in some bounds.

From these design considerations, a high-level design was first created to describe the simulation in totality. As can be seen in Figure 8, a UML diagram was created. Shown is the two dimensional case only, to prevent the visualization from being too cluttered. The main body of the simulation is the Simulator object. This is responsible for aggregating data and passing it between the User Interface and the environment object. As previously discussed, the hydrophones are treated individual sink objects to allow for different array designs to be validated. The source was again treated as an object to allow for multiple sources to be included, thus allowing for multiple frequency sources to be easily added to an environment to simulate random noise sources. Both the sink and source objects are implemented by the environment object. The environment takes in initial configuration from the simulator object, from which it generates the space which sound propagates through. This is accomplished differently for a three-dimensional or two-dimensional

SC

space, but the concept between the two is the same. Object factories are employed to create the different cube or grid objects, respective to the dimensional order of the simulation. There are two main implemented objects of these, the obstructed or open element. The environment will be filled with these objects based on a configuration of the space and can be refined in size based on the desired accuracy to speed tradeoff of a run. From an initial configuration of the space, obstructed objects will be assigned appropriate parameters such as absorption characteristics and the speed of sound propagation. To effectively use these grid parameters, each grid object implements an array of waveform objects, which describe the sound waves propagating through the environment. Initially, these objects are comprised of empty waveforms as they have not interacted with other grid objects or the source object. When the simulation executes, grid objects containing sources populate a grid object with data about the waveform. This grid object passes information to neighboring grid objects in the direction of travel and so on. When a grid object contains a similar waveform, in frequency, signal confliction is processed and the result becomes the new object stored. Waveforms of different frequency are stored together in an array and the data is combined when a grid object containing a sink is met. The sink then begins to store the data as a waveform over time, to allow for delayed reflections to be described.

SC
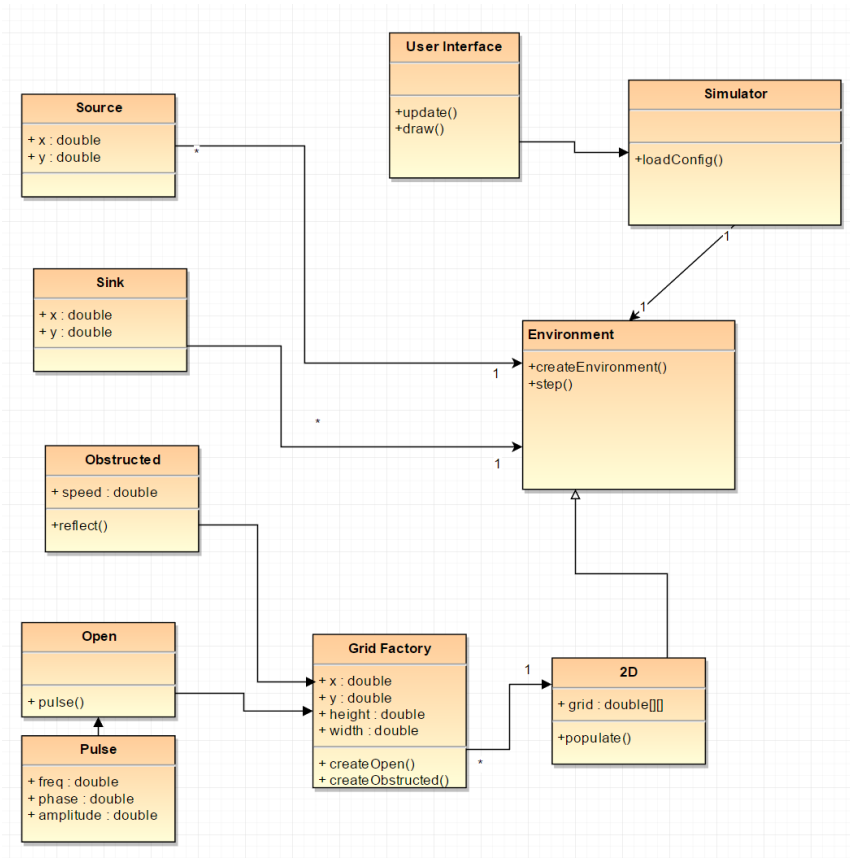
Figure 8: Simulation UML – 2D

The initially proposed simulation was not fully implemented. Throughout the initial development of the simulation and network, the intended data to be processed changed from a raw data input to a signal representative of the front of the waveform. A discussion of reasoning behind this will take place in the following network section. Modeling this portion of the signal requires greatly reduced simulation capabilities. Instead of describing diverse amounts of noise in a system and the full 256,000 samples taken by a DAQ to guarantee the waveform is in the data, at the maximum pulse rate of the physical sound source available, instead only 20 samples were required, the samples required to capture a full phase of a 25-kHz signal, the lowest frequency in consideration

as it is the lowest frequency able to be produced in the hardware available. This dramatically changed what was required of the simulation. The major two simulation that could be removed from this were environmental noise and reflections. As the function of the simulation was now to only describe signals at the point where these factors are not affecting the data, they would only serve to be unnecessary computations rather than important descriptive factors.

What these changes in the simulation intent resulted in was the implementation of the simulator, source and sink objects, with the environment object being converted into a struct to effectively store the source and sink data. Finally, the user interface was removed as little of the full functionality was required, and was deemed unnecessary to implement.

Planar Localization

Initial work on the subject was conducted looking at planar, or two dimensional, localization. The result of this computation gives a bearing to a sound source in the plane of the array, however not distance. The array used to for this localizing is comprised of a right triangle with each hydrophone placed on a vertex. Figure 9 shows the arrangement of the array, where theta is the bearing to be determined.
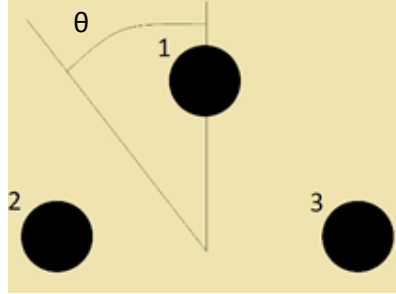
SC

Figure 9: Planar Array Arrangement

To size the array, it is necessary to know the upper limit of the frequency you will be processing. The maximum side length of the array is half the phase length of the highest frequency, which is the shortest phase. This is because when more than a half of a wavelength is received, the determination of phase difference can be rendered incorrect by the common approaches to calculating, such as zero crossing determinations. As our test data was collected in a freshwater environment, and the velocity used to size the array reflects the speed of sound in fresh water accordingly. As calculated below, this size length is 1.855 cm.

$$\lambda = \frac{v}{2f} = \frac{1484 \ m/s}{2 * 40000 \ s^{-1}} = 1.855 \ cm$$

From this array, a pure geometry can be used to relate the difference of phase, ϕ, of the arriving signals to the bearing, θ, that the sound source originated the pulse from. A full derivation of the mathematics of this is referenced in the appendices, for brevity only the final equation will be shown below. It shall be noted that this approach will be hence referred to as the classical approach in the following sections.

$$\theta = \tan^{-1} \frac{\phi_{32}}{2\phi_{31} - \phi_{32}} , where \ \phi_{32} = \phi_2 - \phi_2$$

SC

Network

The proposed network design was to consist of multiple layers of various signal processing and mathematical operations. This network would have an input of 256,000 samples, the number of samples to guarantee a pulse has been captured based on the sampling rate of the DAQ and fastest pulse rate of the sound source. To train the model, simulation data would be used to meet the diverse number of cases. In attempting to accomplish this design, it was determined that to be to computationally costly to pursue this avenue. This cost goes beyond the difficulties in simulating discussed prior to more of the hardware requirements that would be needed to train a network on the data. From the revised model discussed, typical training sets are around 1,000,00 signals. This quantity of signals serves to cover a diverse range of frequencies, 25 kHz – 40 kHz, while being able to achieve accuracy at these levels. This quantity of signals was determined through experimental testing of the different signal configurations, initially on a single target frequency. This was found to require approximately 30,000 signals for to converge in a manner that accurately describes the data as opposed to an overfitting of the data. This was then extended to testing multiple frequencies, in which again through experimentation it was found a frequency step of at least 500 Hz was required to prevent large variances in error when interpolating between frequencies. The resulting signal count is the combination of these two parameters. Assuming the full implementation would require approximately the same amount of data and considering that each pulse is comprised of 3 channels of 16-bit data, this means that to train the full network model, 1430.5 Gigabytes of data would need to be generated. This is not feasible to work with

SC

for a variety of reasons, and as such lead to a fundamental change in approach in the network to be constructed.

Differing from the proposed design, it was found that the proposed layer structure was deemed unnecessary to achieve the desired result. This determination was based upon the performance achieved from a more simplified model. This simplification stemmed from the changes in the simulation. This resulted in a fundamental reworking of the core layers of the model. The core layers were changed to operate under a curve fitting premise, in which any arbitrary curve can be described as a summation of sinusoids. The goal of a neural network is to find a general way to relate an input and output to each other. Through the training of the network, these sinusoids are computed to best fit the data provided to the output. The output layer takes the processed result from the intermediate steps and maps it to the output of the data.

The design process of implementing the network was based on the sprint methodology of SCRUM. The reasoning behind this approach stemmed from the necessity of testing multiple varying network designs as well as the fine tuning required for individual network structures. Many parameters went into the design of the model, including number of internal layers, as well as the numerous different variables in the training set. Following the SCRUM methodology, the sprint process started off with a rather small model and training set looking at just one frequency of data. This served to judge the performance over all. From then on sprints focused on redesigning and fine tuning parameters based on two key metrics, accuracy and input data coverage, that is the scope

SC

of what the model covers. Deviating slightly from the SCRUM methodology, documentation of each model was taken, recording all parameters used as well as saving a copy of the model to validate future improvements off.

At the end of the design process, a model was constructed which took in 20 samples per channel and output the final bearing. The network was comprised of 32 layers (1 input, 1 output, 30 sinusoid convolutions) to accomplish its calculations. In comparison to the infeasible proposed model, the signal data to train this network was only 0.11 Gigabytes of data. This allowed for rapid development and validation, ensuring the performance of the final model produced. Table 1 below are the parameters used for the training of the final network whose performance will be shown in the results section.

Table 1: Final Network Training Set Parameters

Parameter	Value
Frequency Range Lower Bound	25 kHz
Frequency Range Upper Bound	40 kHz
Frequency Range Step Size	500 Hz
Bearing Range Lower Bound	-90 Degrees
Bearing Range Upper Bound	90 Degrees
Bearing Range Step Size	0.00125 Degrees
Sampling Rate	500000 Samples/second/Channel

Of note in the training parameters, we can see the range of bearings to be bounded between -90 and 90 degrees. This range can easily be extended to the full 360 degree range, requiring just the use of the atan2() function in the classical method for comparison. As these bearings provide no unique features as compared to restricted range, they were excluded so as to allow for faster training times at the desired accuracy level.

SC

Results

Overview

Two primary metrics are employed to validate the performance of the network, that of

accuracy and execution time. As a reference to compare against, the classical approach is

employed. For each of the synthetic tests discussed in this section, a quantity of test

cases, 144001 per frequency, was deemed statistically significant. For the real-world test

data, this was limited by the geometry of the pool itself, and this restriction will be

reflected in the bearings chosen. In execution time tests, 144001 test cases were run for

individual frequency trials, a balance between being statistically significant and

maintaining a realistic total runtime for acquiring results.

Accuracy is measured in two different manners, synthetically creating signals with

known bearings, looking at error distribution and the standard deviation of multiple pings

in a fixed location. Standard deviation is employed in this case as it was not possible to

accurately enough measure the location of the source and the array in the university pool,

where real-world data was collected. When looking at the data presented from this

performance measurement, it should be noted that both approaches received the same

input of data, stemming from a filtering algorithm tailored specifically to the cases being

reviewed, ensuring that the data is at the front of the waveform.

Synthetic Testing

Before showing frequency specific plots, the error distribution across the entire frequency

range is shown in Figure 10 as a convenient reference to the performance bounds of the

SC

network model. As can be seen in individual frequencies, represented by unique colors, each there is enough variance of error in any frequency that this plot serves to illustrate the overall bounding space for the model. The classical approach was omitted as its it was deemed to be not useful as an aid for understanding the performance.
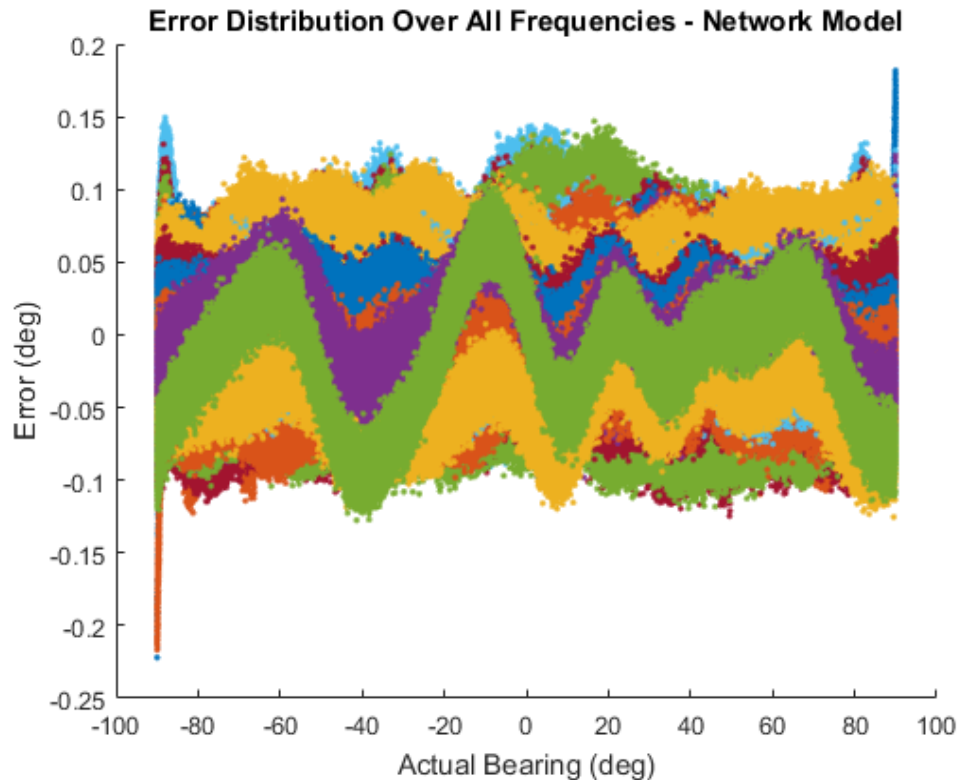

Figure 10: Error Distribution of Network Model Over All Frequencies

Moving on from the overall performance of the approaches, we now will look at the behavior at certain frequencies. In this section, the focus will be on 40 kHz, 25 kHz and 31.125 kHz. These values were selected to cover the upper and lower bounds of the operating frequencies in consideration as well as including a frequency that was not explicitly used in the training of the network model.

SC

The first frequency in consideration, 40 kHz, is of note as it is the highest frequency value in the range tested. This is of note because at this frequency, a full phase of the signal is described by the least number of points at the sampling rate of the DAQ. A signal at this frequency is comprised of 13 samples at the 500,000 Samples/s DAQ update rate, when compared to the 20 samples of a 25-kHz signal. This represents 35 percent less data to describe a single signal. It is not to say that this reduction in data creates unusable amounts of error in the system. Some of the total error is offset because the model is input with a constant number of samples, the full 20 required by 25 kHz. As such, the phase calculation employed by the bearing produces more accurate result than calculations with only the 13 of a single phase. This can be seen in Figure 11. Looking at the network model, Figure 12, at this frequency, the error of the model is slightly higher as compared to the classical approach. The error seen in the model is around the lower bounds of error in the network, stemming the fact that this was one of the frequencies the network was trained upon.
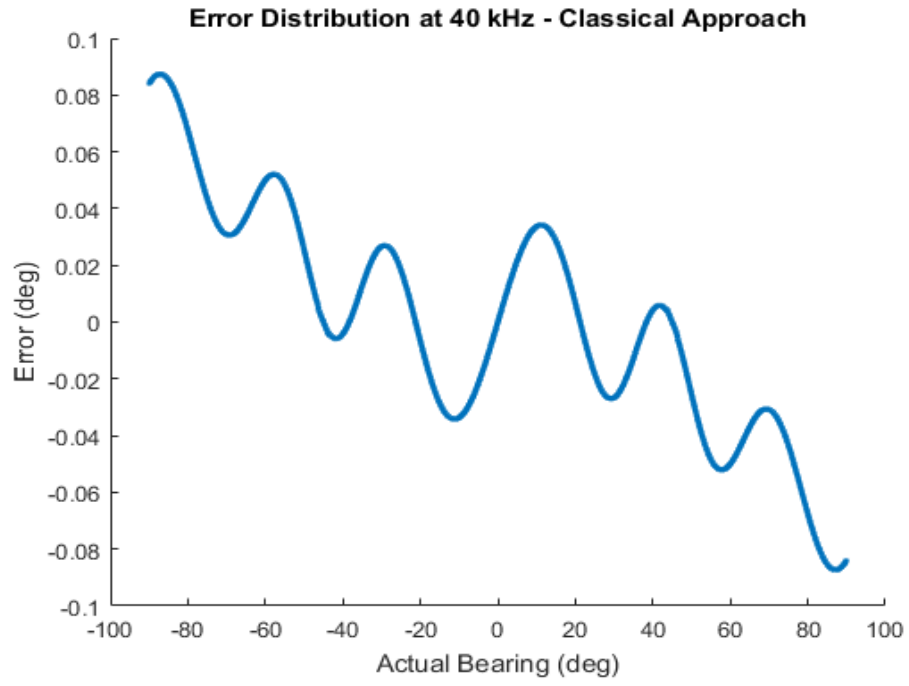
SC

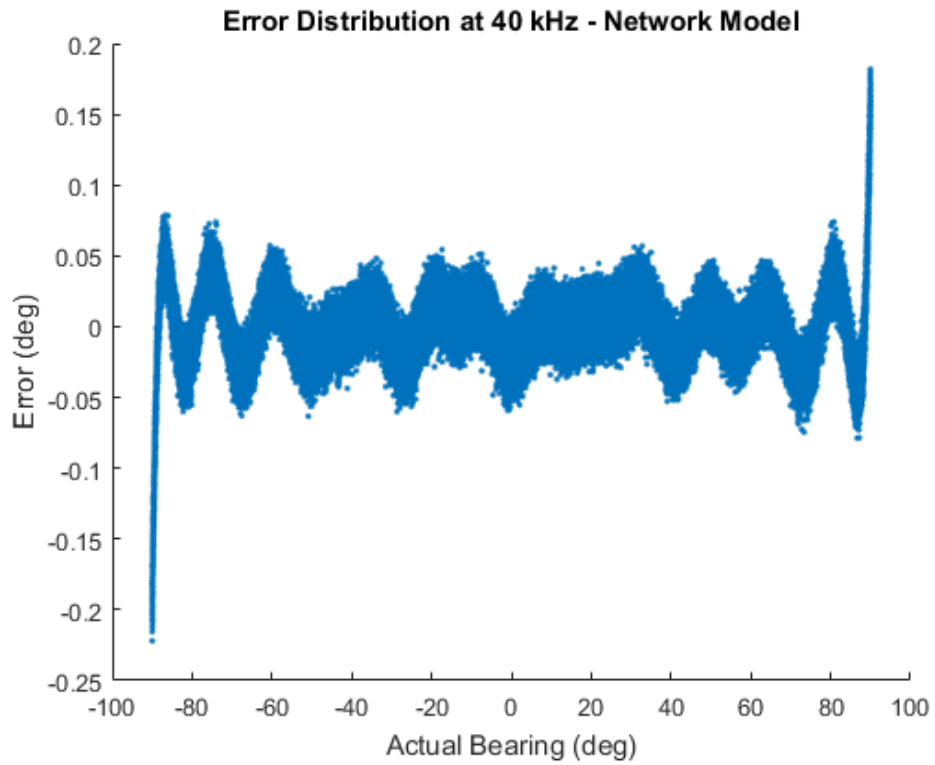*Figure 11:* Error Distribution at 40 kHz - Classical Model

Figure 12: Error Distribution Network Model 40 kHz

SC

Looking to the other end of the frequency spectrum, 25 kHz represents the most accurate frequency in the classical approach, shown in Figure 13, illustrating that more data describing a single phase of a signal is more relevant that describing more than a single phase. This trend is also illustrated in the network model, shown in Figure 14, with slight reduced error as compared to the 40 kHz signal, admits the fact that both frequencies were used in the training process.
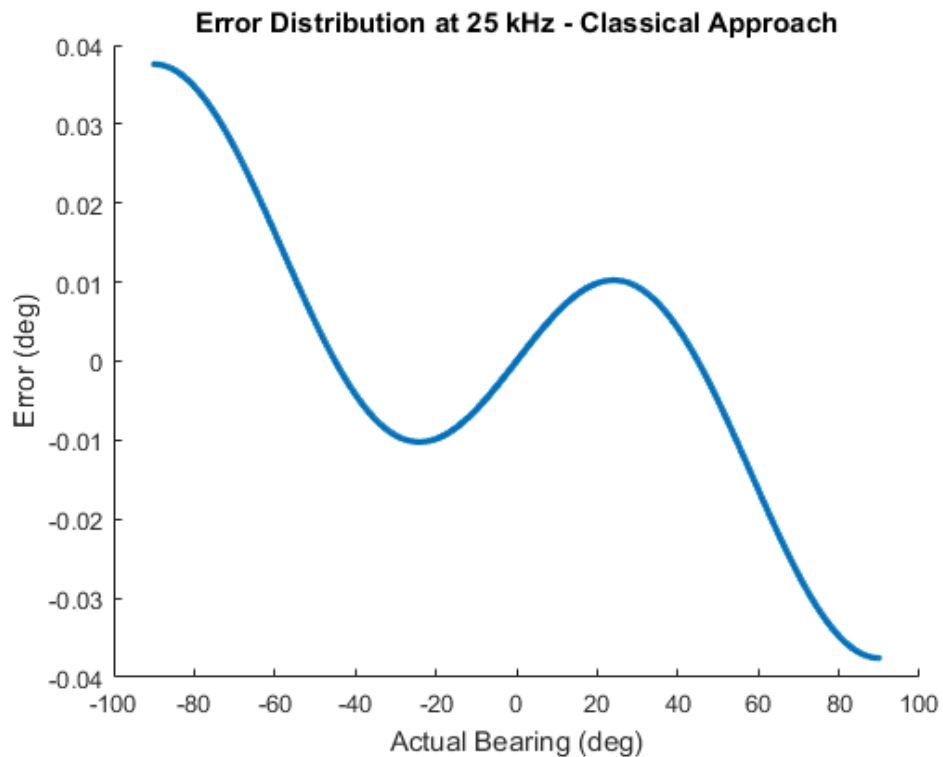

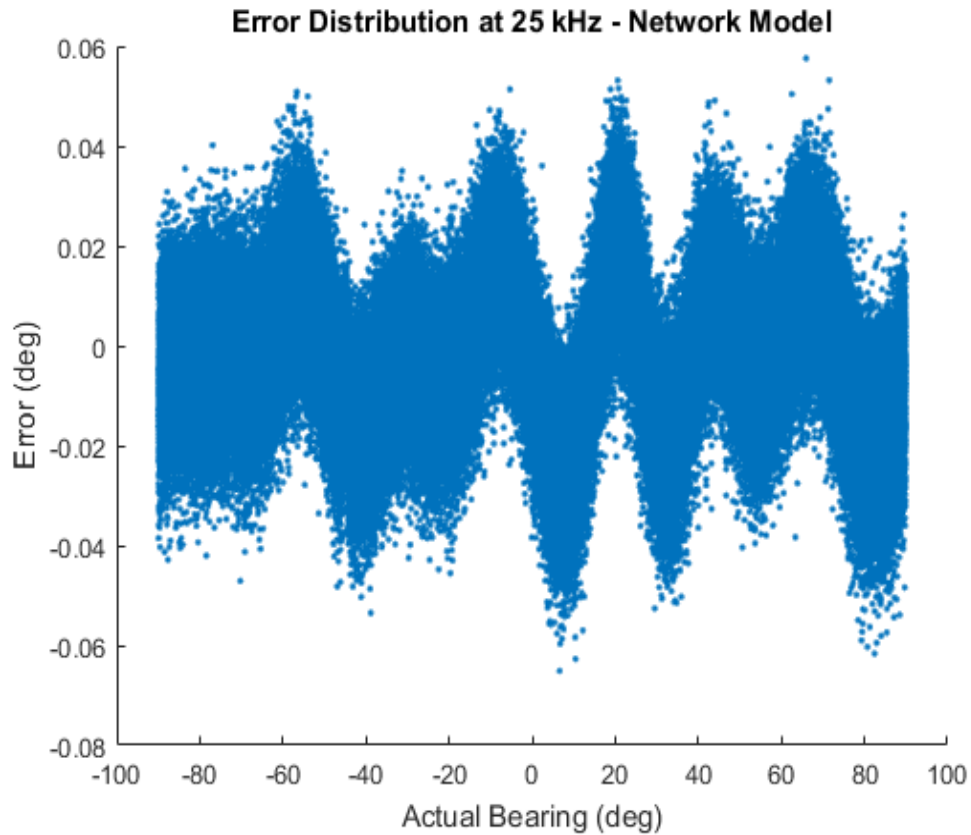*Figure 13:* Error Distribution at 25 kHz - Classical Approach

Figure 14: Error Distribution Network Model 25 kHz Full Spectrum

The final frequency in consideration is that of 31125 Hz. As can be expected from the

classical approach, the performance is between the upper and lower bounds of the

frequency range. Figure 16 shows this error performance. The error at this frequency has

peaks akin to 40 kHz, but overall has less error across the board. The more interesting

discussion comes when looking at the error in the network model, shown in Figure 15.

What can be noted in this chart is the error is not fundamentally different from either of

the other two frequencies discussed. This highlights the capabilities of the network to
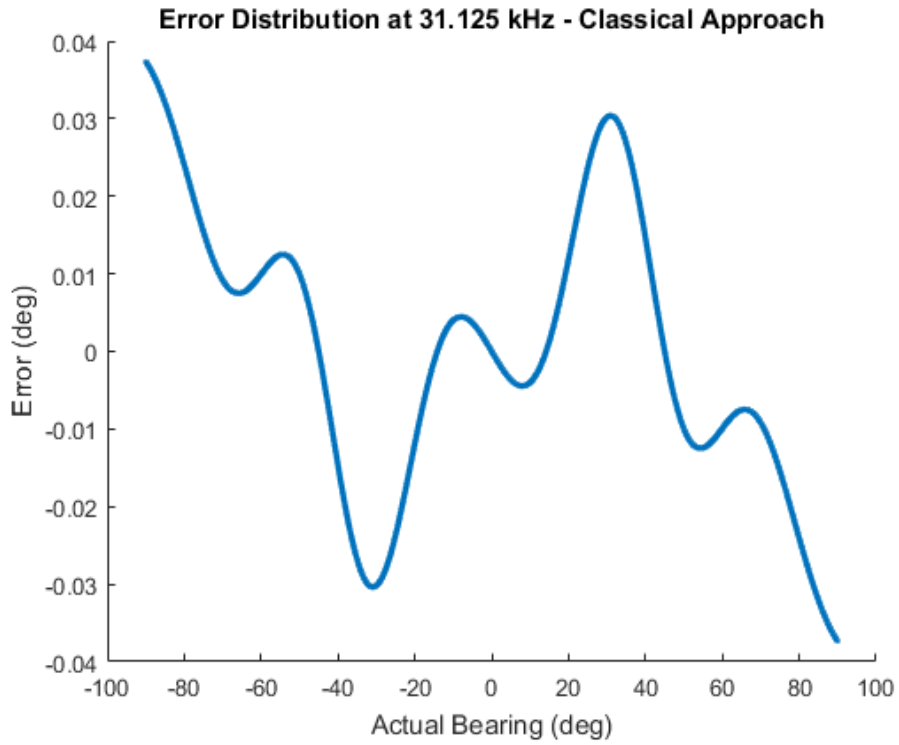
handle inputs not trained upon.

SC

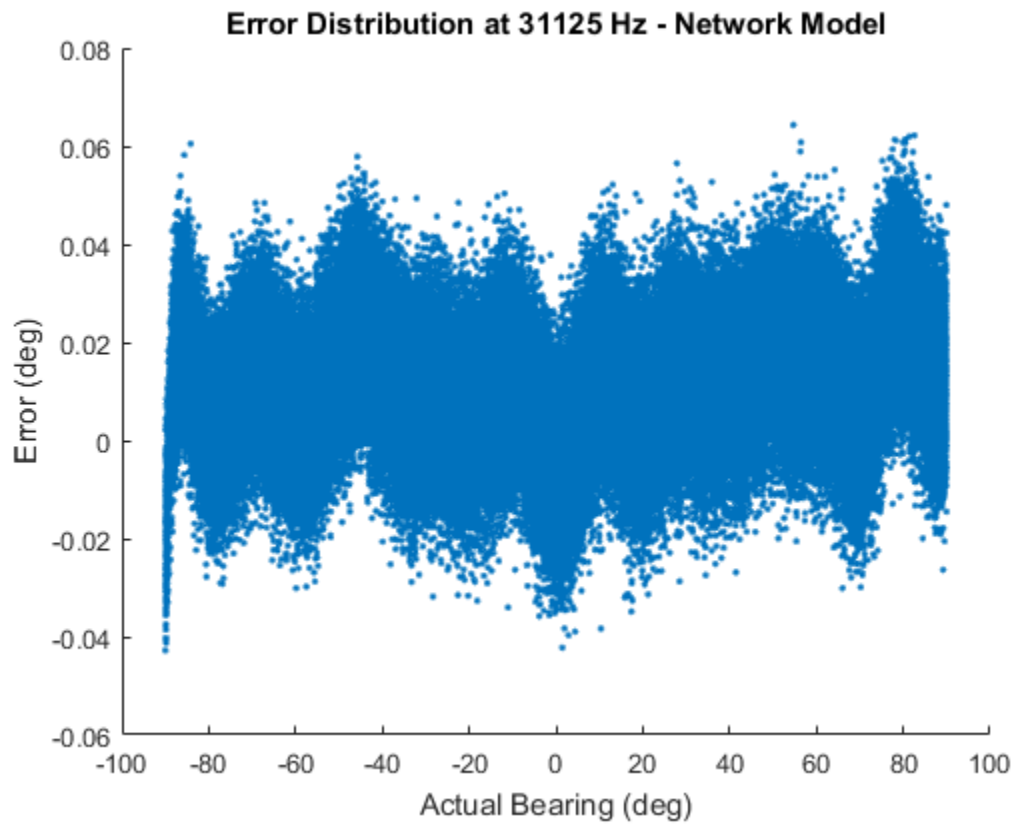*Figure 16:* Error Distribution at 31125 Hz - Classical Approach

Figure 15: Error Distribution Network Model 31125 Hz Full Spectrum

SC

Real World Data

Multiple tests were conducted on data collected in a real-world scenario. In this section, a test with the source orientated at approximately 3 degrees from the array and 33 degrees are shown as they are indicative of the behavior of all cases. For all tests the angle computed is assumed to be correct within the best possible measurements taken. To make results comparable, both approaches were given the same data at the front of the waveform to process. It should also be noted that the training set on the network model remains unchanged from the synthetic data testing. All data used in training was generated in simulation. Tests were conducted with no other individuals in the pool and both the array and the source were located as far away from a reflection source as possible, a meter in the case of the array, and an amount that changed for the source based on test conditions and pool geometry. The standard deviation and average result are collated in Table 2 below for convenience.

Table 2: Real world results collated

TEST CASE	CLASSICAL AVERAGE	CLASSICAL DEVIATION	NETWORK AVERAGE	NETWORK DEVIATION
3 DEGREES	3.592172	0.948474	5.230275	2.204653
33 DEGREES	32.93659	1.29054	35.85841	1.729078

Looking first at the 3 degree case, we can see two fundamentally different results between the network and the classical approach in terms of standard deviation error. From the testing, the source of this deviation stems from the voltage variations across channels of the waveform being processed. As shown in Figure 17 and Figure 18, the difference between one of the channels and the other two is dramatically different, however the actual phase of each channel is not. This directly correlates to the spike in

SC

error between trial 2 and 4 seen in Table 3. This resulting error of the network model is a current limitation of the model. What you see in these two trials is that the voltages be read from two of the three hydrophones are lower than the third. More specifically, the hydrophone at the front of the array, labeled 1 in Figure 9, has the voltage level higher than the other two sensors. The exact reasoning of this could not be determined, however through an investigation of recorded data at multiple bearings it is observed that the voltage of the sensors behind the first sensor to receive the sound wave were in most all cases at or below the voltage level of the initially receiving sensor. This voltage differential between channels causes an error because it is a behavior not captured by the simulation. The simulation operated under the assumption that channel voltage would be the same between channels. As such, the convolutions computed through training did not account for these voltage deviations.

Table 3: Real world data - 3 degrees

TRIAL	CLASSICAL APPROACH	NETWORK
1	2.07319	3.212398
2	2.50552	1.579879
3	2.41081	7.509019
4	3.73535	9.618713
5	3.85782	6.689917
6	3.70094	6.173939
7	4.96728	4.275547
8	4.8643	4.649938
9	4.2948	4.906695
10	3.51171	3.686701

When looking at the results of the 33 degree case in Table 4, a similar trend in the performance arises. Both approaches are very near the actual value and the network model has higher deviation trial to trial. The same voltage behavior was seen in the

SC

network, there just was simply less voltage deviation between channels. Overall from this testing, we confirm the results of the synthetic testing. Both approaches can localize, with a bit more accuracy out of the classical approach.

Table 4: Real world data - 33 degrees

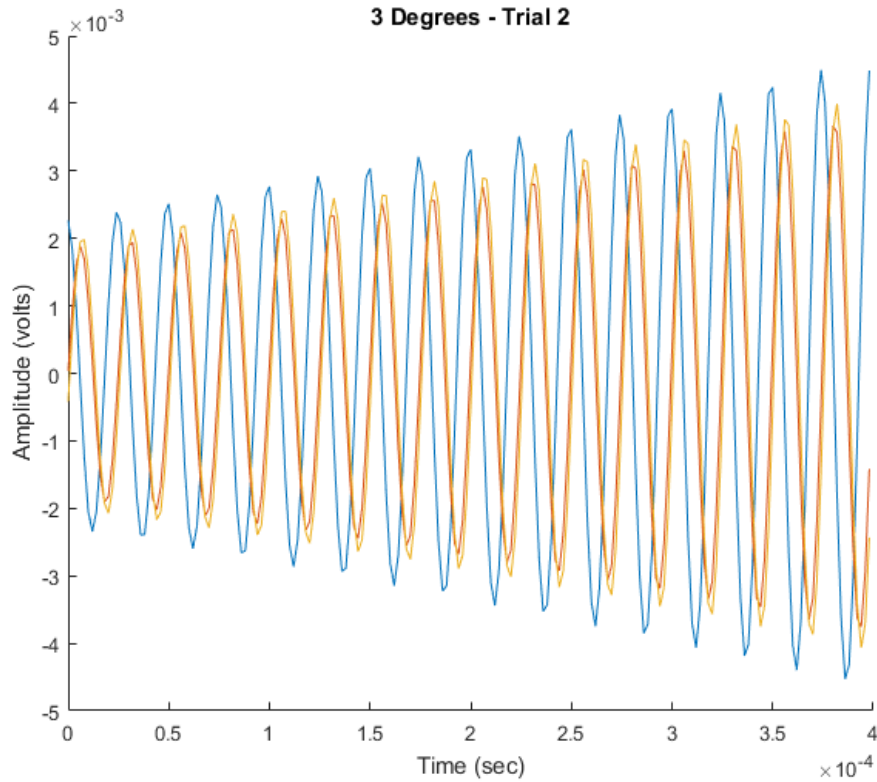| TRIAL	CLASSICAL APPROACH	NETWORK
1	34.668	35.65961
2	32.8978	33.2122
3	33.1016	36.61564
4	33.0676	36.91564
5	31.3817	37.21121
6	31.725	36.171
7	31.9448	32.0231
8	31.4983	36.13202
9	35.4146	37.03154
10	33.6665	37.61213

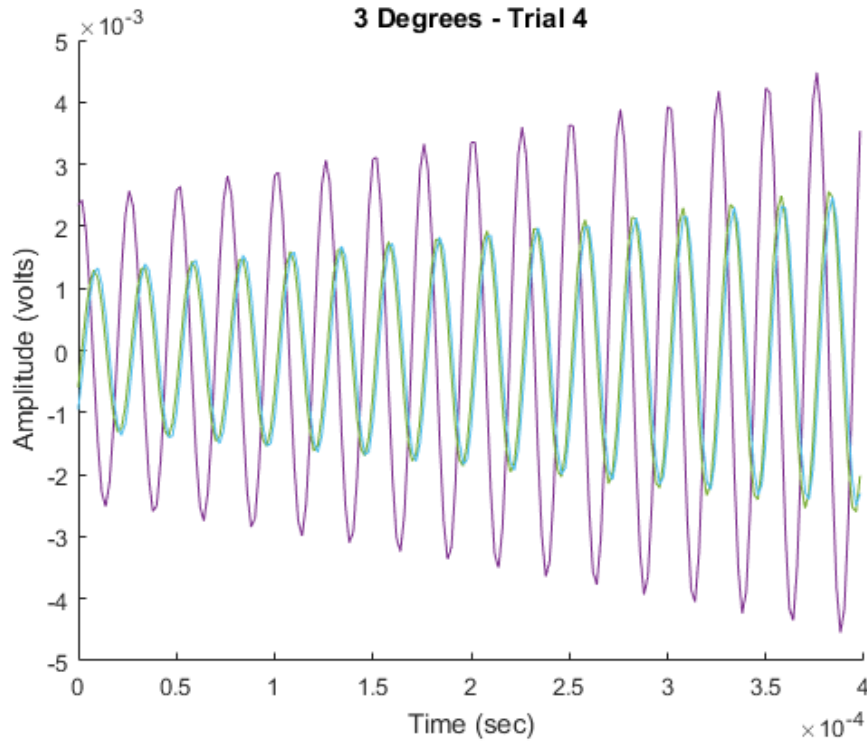*Figure 17:* Real world data - Low voltage difference

SC

Figure 18: Real world data - High voltage difference

Execution Time

To evaluate the execution time of the approaches, timing data was taken from the
synthetically generated data. For the case of the non-network approach, timing was taken
from computing the phase of the signal through the angle. This is to give the same inputs
and outputs as the model. It should be noted that timing was taken in the Windows
operating system, as opposed to a real-time operating system. As such, to best ensure the
evaluation of computational time, all processes were evaluated to the highest priority in
Windows, the real-time level. This level ensures the process core or cores the task is
running on is cleared of any other threads as well as running the cores at full speed. To
further ensure the timing of an individual execution, approximately 240,000 pulses were
used to average the run time of an individual input.

SC

Looking at the execution time distribution in Figure 19 and Figure 20, we see the average execution time between the two approaches have a rather small difference between them. To clarify the plots, each point refers to the average execution time of a particular frequency evaluated, more specifically 144001 cases per point on the plot. The average execution time for the classical approach and network model are respectively $2.88 * 10^{-5}$ s and $5.3995 * 10^{-5}$ s. Although the classical approach performs faster in operation in this set, the executions times are both quite small and in actual operation with signals are both likely not the bottleneck on any signal processing problem.
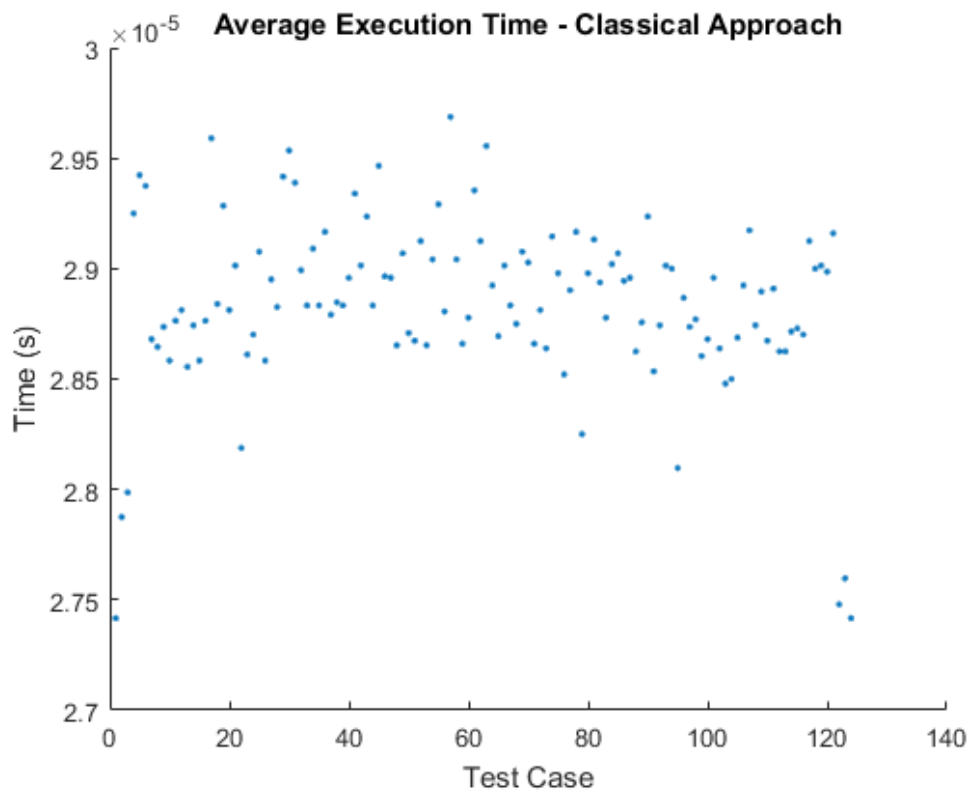

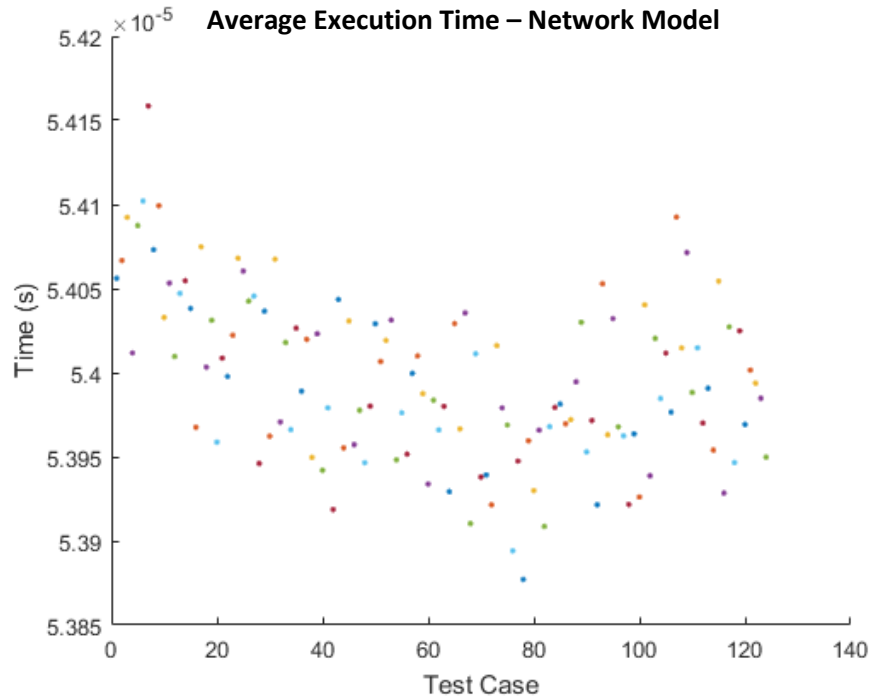*Figure 19:* Execution Time – Classical Approach

SC

Figure 20: Execution Time - Network Model

Conclusion

The work conducted to the writing of this paper produced a two-dimensional model with lower accuracy and execution time than the conventional approaches to signal processing. Although less successful in both metrics, the network model showed performance at the same order of magnitude of the classical approach, and has the potential to improve performance with another pass at optimizing layer size and the training set.

Although the end product produced showed a great deal of promise in the application of neural networks to signal processing, the work was not without its own problems. Due to the nature of the data being worked with, traditional approaches to training become infeasible just on the simple nature of data requirements alone. Massive storage arrays would need to be employed to prevent data from having to be generated every test.

SC

Beyond this, producing data of a diverse and descriptive enough nature to generate a converging model would necessitate an incredibly generic simulation to produce.

Beyond just the pure performance of the model generated, the use of a neural network provides certain functionalities that could not be had in a classical approach. The network requires no information on the frequency of the data being processed. This capability allows for far greater potential applications of the technology. The same principles that allow for acoustic localization to excel underwater, for instance the ability to communicate over long distances, is also a burden in potential use cases. A sound source cannot be effectively directionally sent, especially as the source is ignorant of the array listening. As such, any array can localize off a source, including those potentially unwanted. However, because of the robustness to frequency changes, the source can change frequencies impeding sources that cannot adapt while having a negligible impact on a network model.

Future Work

Looking to future extensions of the work produced, there are two primary directions that are recommended to be pursued is the implementation of a network for three-dimensional localization. If employing the same approaches applied to two-dimensional localization presented, there would be a relatively small level of time commitment required to restructure the network for four outputs produced, that of three orientation vector angles and vector magnitude. The simulator was designed has support for rearranging the array and moving the source or array in the third dimension, output calculations would have to be refactored however. The primary driving force of not implementing this network in

41

SC

this paper has more to do with the development time associated with a classical solution to the problem.

The other primary focus should be on the robustness of the network to the differing voltages across the channels. This had the largest impact on the success of the network when deploying in the field. To accomplish this, the likely courses of action are to add an additional initial layer to the model to scale channel data to the same value, or to train the model on varying channel voltages. Which of these is the best solution will be determined from testing on diverse sets of data.

Bibliography

[1] Pomerleau, D. A. (1996). Neural Network Vision for Robot Driving. *The Handbook of Brain Theory and Neural Networks*.

[2] Singh, P., & Agrawal, S. (2013). TDOA Based Node Localization Using Neural Networks. *Communication Systems and Network Technologies Conference*, (pp. 400-707).

[3] Ergut, S., Rao, R., Dural, O., & Sahinoglu, Z. (2008). Localization via TDOA in a UWB Sensor Network using Neural Networks. *IEEE International Conference on Communications*, 2398-2403.

[4] Chen, H., & Chou, T. (2008). Hybrid TDOA/AOA Mobile User Location with Artificial Neural Networks. *IEE Internation Conference on Networking, Sensing and Control*, (pp. 847-852).

[5] Satymurti, S., & Joshi, R. (2014). Range based node localization in WSN by using artificial neural network. *Internation Conference on Electrical, Electronics, Computer Science & Mechanical Engineering.*

[6] Johnson, M., & Waterman, A. (2006). APLS - An Acoustic Pinger Location System For Autonomous Underwater Vehicles.

[7] Hu, Y., & Hwang, J. (2002). *Handbook of neural network signal processing.* CRC Press.

[8] Malioutov, D., Cetin, M., & Willsky, S. (2005). A sparce signal reconstruction perspective for source localization with sensor arrays. *IEEE Transactions on Signal Processing*, 3010-3022.

[9] Pal, S., & Mitra, S. (1992). Multilayer perceptron, fuzzy sets, and classification. *IEEE Transactions on Neural Networks*, (pp. 683-697).

[10] Rajaww, S., Sadeghi, H., & Almodarresi, T. (2008). Energy Efficient localization in wireless ad-hoc sensor networks using probablisitc neural network and Indepent Component Analysis. *Telecommunications International Symposium*, (pp. 365-370). Tehran.

[11] Tank, D., & Hopfield, J. (1986). Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit. *IEEE Transactions on Circuits and Systems*, (pp. 533-541).

SC

Appendices
Derivation of Planar Localization

The calculation used in this algorithm looks at a grouping of three hydrophones in a right triangle. The side length of the two shorter sides is the 1.855 cm, calculated in Equation 1. For the derivation of the solution, this will be defined as d, as this solution is for the general form of this array. Figure 21: Hydrophone array geometry shows the triangle and includes references to the hydrophones that will be used throughout the rest of the calculations. The goal of the calculations is to determine θ, the heading of the sound source relative to the triangular array. [2]
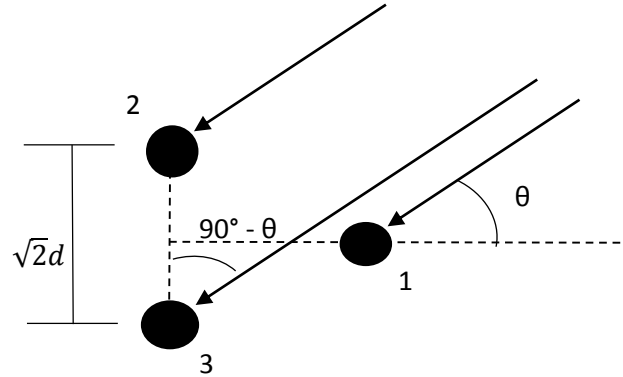

Figure 221: Hydrophone array geometry

The reference hydrophone is defined as 1. Based, on this there is a lag in the time it takes to reach hydrophone 2 and 3.

From Figure 21, a geometric relationship between θ and d can be established. Pairing this with the relationship between the speed of sound in a fluid, c_0, the time of arrival t_2 and t_3, the geometry of the array arrives at the geometric relationships of Equations 2, 3 and 4.

$$\cos(90° - \theta) = \sin(\theta) \tag{2}$$

$$\sin(\theta) = \frac{c_0(t_3 - t_2)}{\sqrt{2}d} \tag{3}$$

$$\cos(\theta - 45°) = \frac{c_0(t_3 - t_1)}{d} \tag{4}$$

Dividing Equation 4 by 3:

$$\frac{\cos(\theta - 45°)}{\sin(\theta)} = \frac{\frac{c_0(t_3 - t_1)}{d}}{\frac{c_0(t_3 - t_2)}{\sqrt{2}d}} = \sqrt{2}\frac{(t_3 - t_1)}{(t_3 - t_2)} \tag{5}$$

Using properties of cosine, we are can expand:

SC

$$\cos(\theta - 45°) = \cos(\theta)\cos(45°) + \sin(\theta)\sin(45°) \quad (6) = \cos(\theta)\frac{1}{\sqrt{2}} + \sin(\theta)\frac{1}{\sqrt{2}}$$

Dividing through by $\sin(\theta)$ and simplifying:

$$\frac{\cos(\theta - 45°)}{\sin(\theta)} = \frac{\cos(\theta)}{\sin(\theta)}\frac{1}{\sqrt{2}} + \frac{\sin(\theta)}{\sin(\theta)}\frac{1}{\sqrt{2}} \qquad (7)$$

$$\cot(\theta)\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} = \sqrt{2}\frac{(t_3 - t_1)}{(t_3 - t_2)} = \frac{\sqrt{2}t_{31}}{t_{32}} \qquad (8)$$

Simplifying the definition of time difference for the rest of the calculations:

$$t_{31} = t_3 - t_1 \qquad (9)$$

Substituting (9) into (8) and simplifying:

$$\cot(\theta) + 1 = 2\frac{t_{31}}{t_{32}} \qquad (10)$$

Solving for $\cot(\theta)$:

$$\cot(\theta) = \frac{2t_{31}}{t_{32}} - \frac{t_{32}}{t_{32}} = \frac{2t_{31} - t_{32}}{t_{32}} \qquad (11)$$

Solving for θ:

$$\theta = \tan^{-1}\frac{t_{32}}{2t_{31} - t_{32}} \qquad (12)$$

Verification checks:
  If $\theta = 0°$, $t_{32} = 0$
  If $\theta = 45°$, $t_{32} = t_{31}$

Putting t in terms of ϕ, the phase shift, where f is the frequency of the signal:

$$t_{32} = \frac{2\pi(\phi_3 - \phi_2)}{360° f} \qquad (13)$$

Substituting and simplifying:

$$\theta = \tan^{-1}\frac{\dfrac{2\pi(\phi_3 - \phi_2)}{360° f}}{2\dfrac{2\pi(\phi_3 - \phi_1)}{360° f} - \dfrac{2\pi(\phi_3 - \phi_2)}{360° f}} = \tan^{-1}\frac{\phi_{32}}{2\phi_{31} - \phi_{32}} \qquad (14)$$

SC

This final simplification is used to speed up computation time. It was determined to be faster computationally to compute phase difference than time difference and as there is a direct correlation, the ratio is the same.