Acoustic Measurements Using a Microphone Array

Nils Smit-Anseeuw
Aaron Zimmer

Project Sponsor:
Prof. Chris Waltham, P. Eng.

Applied Science 459
Engineering Physics
The University of British Columbia
December 5, 2011
Group Number 1213
Executive Summary

A microphone array is designed, built and used for analyzing the radiation of a single test instrument, an oud. Two analysis algorithms are implemented for this purpose, radiation mapping and delay and sum beamforming. Radiation mapping is used to determine the spatial radiation intensity of the instrument, while beamforming is used to map the source locations on the instrument's soundboard.

In this report, the mechanical design history of the microphone array is presented, as well as a detailed overview of the finalized design. The development and testing of the beamforming algorithm is then summarized, followed by a detailed description and analysis of the final algorithm.

Radiation maps for the 5 largest vibration modes of the oud are produced and presented, as well as beamforming results for various frequencies. Some investigations are subsequently made into the validity of the results, and recommendations are given for their further verification and improvement.

It is concluded that beamforming can consistently and successfully locate single monopole sound sources. While the results of beamforming for the more complicated sources found in the oud soundboard remain far from conclusive, their consistency and intuitive placement suggest that they hold some valid results. In order to fully support this hypothesis, it is concluded that modal analysis should be performed on the oud, comparing the results with those produced by beamforming.
# Table of Contents

Letter of Transmittal ........................................................................................................ 1

Executive Summary ....................................................................................................... 3

Table of Contents .......................................................................................................... 4

1 Introduction .................................................................................................................. 6

1.1 Background and significance .................................................................................. 6

1.2 Project objectives ................................................................................................. 6

1.3 Scope and limitations ............................................................................................ 7

1.4 Organization ......................................................................................................... 7

2 Discussion ................................................................................................................... 9

2.1 Theory .................................................................................................................... 9

2.1.1 String instruments ......................................................................................... 9

2.1.2 Acoustic simulations ..................................................................................... 9

2.2 Microphone array .................................................................................................. 10

2.2.1 Array requirements ....................................................................................... 10

2.2.2 Initial designs ................................................................................................ 11

2.2.3 Final design ..................................................................................................... 14

2.2.3.1 Array .......................................................................................................... 15

2.2.3.2 Instrument frame ....................................................................................... 16

2.3 Data analysis algorithms ....................................................................................... 20

2.3.1 Microphone locations and sensitivity calibration ........................................... 20

2.3.2 Radiation mapping .......................................................................................... 22

2.3.3 Beamforming .................................................................................................. 22

2.4 Experimental overview ......................................................................................... 25

2.4.1 Experimental equipment ................................................................................ 25

2.4.2 Setup procedure .............................................................................................. 26

2.4.3 Experimental procedure ................................................................................ 27

2.5 Results ..................................................................................................................... 28

2.5.1 Radiation maps ................................................................................................ 28

2.5.2 Beamforming ................................................................................................... 34

2.5.2.1 Testing Beamforming Algorithm on Real Data ......................................... 35

2.5.2.2 Beamforming Sound Signal Transient Affects ......................................... 35

2.5.2.3 Consistency Check for Instrument Beamforming ...................................... 40
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conclusions</td>
<td>45</td>
</tr>
<tr>
<td>Project deliverables</td>
<td>46</td>
</tr>
<tr>
<td>4.1 List of deliverables</td>
<td>46</td>
</tr>
<tr>
<td>4.2 Financial summary</td>
<td>47</td>
</tr>
<tr>
<td>4.3 Ongoing commitments by team members</td>
<td>47</td>
</tr>
<tr>
<td>Recommendations</td>
<td>48</td>
</tr>
<tr>
<td>Appendix: Data Analysis Code</td>
<td>49</td>
</tr>
<tr>
<td>6.1 PlotData.m</td>
<td>49</td>
</tr>
<tr>
<td>6.2 PlotFFT.m</td>
<td>51</td>
</tr>
<tr>
<td>6.3 DelayAndSumBF.m</td>
<td>52</td>
</tr>
<tr>
<td>6.4 RadiationMap.m</td>
<td>57</td>
</tr>
<tr>
<td>References</td>
<td>60</td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Background and significance
String instruments have been around for thousands of years. The most common string instruments at present are guitars and violins, but the oldest known string instrument, the long lute, dates back to at least 2000 BC (ref. 2). The guitar originated in Italy and Spain in the early 17th century and has since become the most popular instrument in the Western world, while variations of the zither and lute have taken on larger roles in Eastern music.

Despite their prevalence and popularity, much research remains to be done in order to fully understand and characterize the sound produced by these complex instruments. One key aspect of an instrument’s output radiation that has just recently begun to be investigated is the spatial distribution of sound intensity. With a full spatial characterization of the instrument’s radiation, acoustic engineers can go on to design instrument specific recording microphones and performance chambers. Another important instrument characteristic that has started to gain some interest in the research community is the distribution of radiation sources over the soundboard of the instrument. With such a characterization, instrument designers can better understand the intricate relation between string and soundboard and can go on to optimize the design and placement of the bridge between them.

In order to quickly and accurately map the spatial distribution of an instrument’s radiation, it is necessary to place multiple microphones around the instrument to simultaneously record the sound at all angles. With such an array, we can get the full radial amplitude distribution of an instrument in a matter of seconds. Conveniently, using a computational technique known as delay-and-sum beamforming, this same array can be used to characterize the radiation sources of an instrument’s soundboard. The purpose of the current paper is to document our group’s work in building such an array and using it to perform both of the above-mentioned characterizations for a North African oud (see Figure 1).

![Figure 1](image)

**Figure 1:** A North African oud. An oud is a lute style instrument that is commonly used in Middle Eastern music. This will be the main instrument used in our analysis

1.2 Project objectives
The objectives of our project are threefold. The first objective is to design and build a circular microphone array with microphones evenly spaced around the perimeter that can support and excite an
instrument. Secondly, we aim to use the array to produce a directional radiation map of a stringed instrument. Our final objective is to use delay and sum beamforming to produce a source map of the instrument’s soundboard.

1.3 Scope and limitations

In this report, we will describe the work completed during the project timeline, present an analysis of the results obtained in the project, and outline some introductory acoustic theory to underscore the analysis. We will go on to make recommendations for future work that can be done to improve and verify our results.

In presenting our project work, we aim to not only highlight what was accomplished over the project duration, but also provide a document that will enable someone to fully replicate our research. To accomplish this, we will try to keep the focus on the steps that directly influenced our final results. Some comments will be made on faulty early designs and algorithms, but will mainly highlight the reasons for their failure. Three main aspects of the project will be outlined in full detail: the final design and structure of the microphone array, the full experimental procedure for analysis of a stringed instrument, and the workings and use of the analysis software.

The presentation and analysis of our results in this paper will be mainly focused on the radiation maps and source maps of the oud at various frequencies, as well as our characterizations and verifications of the beamforming algorithm. When discussing the oud characterizations, much of the analysis will be subjective, based on a rudimentary understanding of the instrument’s mechanisms. As no similar analysis of this instrument could be found in the literature, verification of our results will mainly come from previous work done on the instrument by the UBC acoustics group. The beamforming algorithm characterization will be more rigorous, and will consist of a proof of concept for single source location as well as an investigation into the limitations of the algorithm as more sources are introduced. In order to keep the results accessible to an engineer unfamiliar with acoustics, the introductory acoustic theory needed to follow the analysis will also be presented.

The recommendations provided at the end of the report will be targeted towards future researchers working with our equipment and software. These recommendations will include methods for improving the characterizations of the instrument, work that can be done to verify the measurements we have made, as well as other applications for our array and software.

1.4 Organization

The report consists of four main sections: the discussion, conclusion, project deliverables, and recommendations. The discussion section contains the majority of the documentation and analysis of the work completed during the project. This section presents the background theory, a description of the array design and structure, a description of the algorithms used in the analysis software, an overview of the testing procedure, a summary of the results, and an analysis of the results. The conclusion section presents the conclusions drawn about the effectiveness of the presented method, as well as any conclusions drawn from the results analysis. In the project deliverables section, we evaluate the completion of the deliverables set out in our project proposal and we provide a summary of the
finances used over the duration of the project. The recommendations section provides any future work that our group sees as valuable to the continuation of the project.

In the appendix at the end of the document, we present some reference material to support certain sections of the report. Included in this section is a full copy of the software used for data analysis, the mechanical design drawings used in the construction of the array, a bill of materials for the array, as well as some extraneous data we found useful for running the analysis.
2 Discussion

2.1 Theory

2.1.1 String instruments

String instruments are extremely complex and difficult to model precisely with mathematics: subtle differences in the way the instrument is held can even alter its normal modes and lead to the production of different sound radiation (ref. 2). Despite these limitations due to complexity, valuable information can be obtained about instrument design through experimentation, computer simulations, and simplified mathematical models.

In general, the sound production of a string instrument results from the coupling between the vibrational motion of the strings and the soundbox, as well as the motion of the air inside of the soundbox. It is possible to construct simple yet informative mathematical models from this basic abstraction, but it is more convenient to use finite element analysis to model such interactions in a more precise way.

Some important concepts in the study of string instruments are the following: modal shapes, normal modes, resonance, and the instrument’s frequency response function (FRF). The strings supply a steady pulsation of a particular frequency that drives the motion of the instrument’s soundbox, which radiates the sound more efficiently than the strings can alone. String mechanics are quite interesting in their own right and can be described most intuitively with d’Alembert’s formula (the general solution to the one-dimensional wave equation).

The sound radiation of a musical instrument is not necessarily uniform with direction. Our main research objective is to determine the factors that influence the anisotropy of an instrument’s sound radiation. By measuring the air pressure and frequency, using microphones mounted all around the instrument at equidistance, we will be able to determine how the different geometries of the instrument soundboards as well as the internal air resonance behavior affect the 3D sound radiation of the instruments.

2.1.2 Acoustic simulations

Our simulations were based on the approximate solution to the sound wave equation in spherical coordinates for a monopole. The exact solution can be written as,

\[ p(r,t) = \frac{Q(t) \cdot ik \rho_o c}{4 \pi r} \cdot e^{-ik(r-a)} \]

where

\[ Q(t) = -4 \pi a^2 U_a e^{-i\omega t}. \]

\( p \) is the gauge pressure, \( \rho_o \) is the density of the air when no wave is traveling through it, and \( a \) is the radius of the sphere whose volume is oscillating and producing the wave. For \( ak<<1 \) we can simplify the equation,
Here we equate all the multiplying coefficients to a single amplitude constant $A$. Throughout our experiments all of our amplitude values are considered relative and therefore we are not concerned with knowing the exact value of this constant. The right side of the previous equation is what we used to generate our simulated data.

For the simulation we calculate the distance between the source and each microphone and generate a data set for each microphone based on equation 1 and using a constant sampling rate to generate the time steps. An example of the waveforms generated by this simulation for a single point source at $x=0.5$ m, $y=0.5$ m and with frequency, $f=50$Hz is demonstrated in the figure below. It is color-coded by microphone number so that as you go around the array the data changes color from green to magenta, which makes it easy to observe how the amplitude changes with the phase.

More complex signals can be built up by adding in multiple monopoles. Also, the simulated noise can be added to the signals. This simulation method is used to generate data for testing our beamforming algorithm, which is described in section 2.3.3.

2.2 Microphone array

2.2.1 Array requirements

The design requirements for the microphone array are as follows:

1. 32 microphones are to be evenly spaced over a ~2m diameter circle around the instrument
2. A frame must be built to support an instrument in the center of the array
3. Some means must exist for changing the relative orientation of the instrument and array

\[
p(r,t) = \frac{i\omega \rho_o Q}{4\pi r} e^{i(\omega t - kr)} = \frac{A}{r} e^{i(\omega t - kr + \frac{\pi}{2})}.
\]
4. The array and frame must be supported inside a chamber with mesh walls, floor, and ceiling. The diameter of the array was chosen as the largest size that can fit through the door of the anechoic chamber in which measurements are to be made. A large diameter ensures the negligibility of evanescent sound sources (noise sources that decay exponentially with distance). At this diameter, we obtain a microphone spacing of 19cm, giving us a maximum measurable frequency of ~900Hz by the Nyquist sampling theorem.

The instrument support frame specified by requirement 2 must not contain any rigid couplings between the instrument and the frame. This will ensure that the frame does not resonate with the instrument and distort our data.

Two orientations are required to satisfy requirement 3. The first orientation puts the plane of the instrument soundboard in the plane of the microphone array. The second orientation places the axis of the instrument’s neck normal to the plane of the microphone array. See Figures _ and _ in section 2.2.3 for a visualization of these orientations.

Inside the anechoic chamber, the only possible supports for the array are the flexible steel mesh comprising the floor, walls, and ceiling of the chamber, and a single 1” diameter pipe anchored into the ceiling.

2.2.2 Initial designs
Two initial designs for the microphone array were proposed and rejected before the design reached its final iteration. The first design was outlined in the project proposal and consisted of a 64-sided collapsible polygon made of square steel. This design was rejected before the project start date due to manufacturing feasibility. The second design was created during the first few weeks of the project and consisted of a semicircle made of 3D-printed brackets and aluminum rods. This design lasted until the beginning of February, when it was discarded due to excessive flexibility and a long manufacture time.

As outlined in the project proposal, our initial design was intended to be a 64-sided polygon supported from the ground that could be broken into 8 easily stored pieces (See Figure 2 below). The proposed vertical configuration (pictured on the right of Figure 2) was intended as a way to change the instrument-array orientation without having to move the instrument. This design had several large problems, including difficult manufacturing and poor support.
The first iteration of the array design. The array consists of 8 curved arms connected by support brackets connected to the 8 legs. All pieces are made from hollow square steel stock. The array is pictured in lowered and raised positions on the left and right respectively.

The planned manufacturing process for this array relied on very high precision cutting and welding of square steel stock. In order for the design to work, it needed tolerances of ~.1° on welded connections between square stock. As this is a far tighter tolerance than is feasible with any available machining resources, it presented a major obstacle for the design.

The other big issue in this design lies in its support system: the proposed legs would simply fall through the floor mesh. Larger feet could be used, but the resulting support would likely be uneven, as the floor mesh is quite variable. This issue was not realized until the after the second design iteration had been completed however, and was not a factor in the rejection of this design.

Due to these concerns, a new design was completed in the first week of January to take advantage of the high tolerance machining provided by the in-house 3D printer. This design consisted of a mesh of 3D printed brackets connected by aluminum rods to form a single semicircle (see Figures 3 and 4). The semicircle was to be supported in a similar way to the first design iteration, with both a horizontal and vertical configuration.
Figure 3: The second iteration of the array design. The circular component consists of 3D printed brackets joined by aluminum rods. The support frame is made from t-slotted Aluminum extrusions. The array is pictured in its lowered and raised positions on the left and right respectively.

Figure 4: A detailed look at the second design iteration. Note the slots for microphones on the internal brackets.
This design progressed further than the last: a prototype bracket was successfully printed (see Figure 5) and plans were being made to print the rest of the array. However, it quickly became apparent that the total printing time for the array was too long to be practical. In order to print the 64 brackets, approximately 50 hours of total printing time was required.

This in itself could have been overcome by a few weekends of work, but doubts began to crop up about the stiffness and stability of such a structure, especially in its vertical configuration. As 50 hours of work is too much to spend on a design that has a good chance of being wholly insufficient, this design was rejected.

**Figure 5**: A printed prototype bracket for the second iteration of the array design. The good fit of both microphone and aluminum rod into this bracket looked promising. Total print time of this bracket was ~45min.

### 2.2.3 Final design

When designing the final iteration of the microphone array, three key changes were made in our approach to the design problem. The first key change was the decision to make the circle holding the microphones out of a single piece of material. This allowed us to sidestep some of the key problems faced in earlier designs, namely it avoided the tight tolerances needed for constructing a 32 piece circle and it avoided the structural weakness created by an excessive number of joints. The second key change was deciding to support everything from the 1” rod in the roof instead of from the floor. With the ability to fasten things to a solid anchor point, support became a much easier problem. The final change was
the decision to perform the manipulation of the instrument-array orientation on the instrument instead of the array. This gave us a stationary array, greatly simplifying the overall design problem.

2.2.3.1 Array
With this new approach to the problem, the main challenge in the array design was to find a way to make a 2m diameter circle out of a single piece of material. Several options were initially explored, including waterjet-cut plywood, bent pipe stock, laminated wood strips, and a large hula hoop. Plywood was dismissed as it would be a messy solution due to splintering on the edges. Bent pipe stock was rejected as the logistics of bending the pipe proved too difficult. Laminated wood strips were also rejected, as the manufacturing process would have been too involved. After failing to find a suitable self-made option, the final decision was to purchase a large hula hoop from a retailer in the US. The hula hoop looked like a good option, as it would be lightweight, collapsible and easy to work with.

To fasten the microphones to a hula hoop, we passed a zap strap through a hose clamp fastened to the hoop, and closed the zap strap around the microphone (see Figure 6). This provides a secure fixture for the microphones from which they can be easily popped in and out.

![Microphone holder detail. The microphone is held in place with a zap strap threaded through a hose clamp. A ridge on the microphone body keeps it in place and allows it to snap in and out.](image)

To ensure even spacing of the microphones around the hoop, an iterative spacing process was used. An initial guess was made for the correct distance between adjacent microphones, and the microphones were spaced at this interval around the hoop. The remaining gap between the last and first microphones was then measured, and the difference between this value and our initial guess was divided by the number of microphones and added to the initial guess. The process is now repeated with this value as the initial guess, and keeps repeating until the guesses converge.

FIGURE 6: Microphone holder detail. The microphone is held in place with a zap strap threaded through a hose clamp. A ridge on the microphone body keeps it in place and allows it to snap in and out.
Now, with the 32 microphones evenly spaced around the hoop, we began to look at supporting the hoop. However, we realized that hoop would not hold its shape when hung vertically and would sag into an oval. To add support, we built a square frame out of shelving supports and fastened it to the exterior of the hula hoop with zap straps (see Figure 7). With this frame in place, the hoop remained circular, thereby completing our objective for the microphone array.

![Completed microphone array, with shelving supports framing the hula hoop.](image)

**FIGURE 7**: Completed microphone array, with shelving supports framing the hula hoop.

### 2.2.3.2 Instrument frame

Under the new approach, the instrument frame had to satisfy two main tasks. Firstly, it needed to support both itself and the microphone array from the ceiling anchor. Secondly, it needed to support an instrument under multiple orientations in the center of the microphone array. 80/20 t-slotted aluminum extrusions were chosen as the construction material for the frame, as they are strong, easy to assemble, and adjustable.
The final frame design consists of two main components: a top fixture designed to fasten to the roof anchor and support the microphone array (Figure 8), and a square frame for supporting the instrument and impact hammer (Figure 9). The two pieces are joined with a revolute joint to allow for more flexibility in the orientation of the instrument.

![Figure 8: Top fixture of the instrument frame](image)

![Figure 9: Square frame component of the instrument frame.](image)
The top fixture of the instrument frame needed to be able to fasten to the roof and microphone array with fixed joints and fasten to the instrument frame with a revolute joint. The roof joint was accomplished using a stock 80/20 piece made for clamping to 1” pipe (see Figure 10). The hula hoop was bolted to a plate which was in turn bolted to the fixture (see Figure 11). To create the rotating joint to the instrument frame, a “lazy suzan” type bearing was used (see Figure 12). To facilitate alignment between the fixture and frame, an alignment mask was cut out of Plexiglas on the laser cutter (see Figure 13).

**Figure 10:** 80/20 pipe clamp for fastening to the roof support.

**Figure 11:** The joint between the array and the top fixture. A simple plate bolted to both pieces.
The square frame consists of a very simple rectangle made of 80/20. The instrument is fastened to this frame by means of bungee cords and elastic bands. This ensures a non-solid coupling between the instrument and frame, thereby keeping the instrument vibrations isolated. There are two configurations...
for supporting the instrument in the frame: upright and horizontal. The impact hammer is attached to
the frame by a single piece of 80/20, which can be adjusted to place the hammer in the desired position.
See Figure 14 below for details of the frame fastenings.

FIGURE 14: The instrument supported in the upright (left) and horizontal (right) positions. In the upright
position, the instrument is fastened by two bungee cords around the neck. In the horizontal position,
the instrument is supported by two elastics around the base of the strings as well as a bungee cord
around the neck.

2.3 Data analysis algorithms

2.3.1 Microphone locations and sensitivity calibration
In order to make sense of the results coming from our microphone array, it is first necessary to fully
characterize a couple of its key properties. The two most important properties of the array are the
depth locations of the microphones, as well as the relative pressure sensitivity of the microphones.

The microphone locations were determined by taking a photograph of the front of the array and using
measurement software to locate the microphones on the picture. The photograph that was used to
localize our microphones is given in Figure 15 below. By determining the pixel locations of the
microphones and scaling them using the reference meter stick in the picture, we can determine the
relative locations of the microphones in meters. For convenience, the reference origin of the
microphones was placed at the centroid of their locations.
Figure 15: The photograph used to determine the relative microphone locations on the array.

The microphone sensitivity calibration was performed by using a microphone calibration tool to generate a reference pressure level at each microphone. The measured voltage at this pressure level is then recorded and the voltages of all the microphones are expressed relative to the first microphone. This gives us a relative value for the sensitivities of each microphone.

The resulting locations and sensitivities for each microphone are given in table 1 below.

Table 1: The relative microphone locations and sensitivities of our array

<table>
<thead>
<tr>
<th>Microphone</th>
<th>X (mm)</th>
<th>Y (mm)</th>
<th>Relative Sensitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>274.6614</td>
<td>857.0348</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>446.0786</td>
<td>775.3127</td>
<td>1.033263</td>
</tr>
<tr>
<td>3</td>
<td>600.2212</td>
<td>660.3701</td>
<td>1.038803</td>
</tr>
<tr>
<td>4</td>
<td>726.4586</td>
<td>515.5293</td>
<td>1.04298</td>
</tr>
<tr>
<td>5</td>
<td>815.4892</td>
<td>360.0579</td>
<td>1.079108</td>
</tr>
<tr>
<td>6</td>
<td>873.2927</td>
<td>181.3322</td>
<td>1.079903</td>
</tr>
<tr>
<td>7</td>
<td>893.2249</td>
<td>-9.35282</td>
<td>1.113835</td>
</tr>
</tbody>
</table>
2.3.2 Radiation mapping

The radiation mapping algorithm is quite simple. The microphone data is initially read in and cut down to its relevant portions (i.e. a short period of time after the hammer strike). A fast Fourier transform is then performed on the microphone signals to get the frequency response. Using Matlab's built-in peak finder, we find the first $n$ peaks of the frequency response to get the frequencies of the largest contributing vibration modes.

We then iterate through the microphones, calculating the amplitude of their response at each of these modes, by taking the absolute value of the FFT at those points. These amplitudes are scaled by the calibration sensitivities of each microphone. This gives us the relative sound amplitudes at each microphone location for each vibration mode, which will form the radius of our polar radiation map plots. The microphone angles are taken from the microphone location calculations of the previous section.

With microphone angle and amplitude values at discrete points around the circle, we then use cubic splines to interpolate between these points, giving us a smooth function of amplitude versus angle. This smooth function is then plotted using Matlab's polar plot function.

2.3.3 Beamforming

Our beamforming algorithm allows us to generate source maps that depict the approximate locations of the regions within cross section of the array that give rise to the sound radiation at a particular
frequency. The design of the algorithm went through multiple iterations to reduce the time required to perform the computations. To give an idea of the magnitude of the time reduction, the original algorithm took up to 10 minutes complete and the final algorithm takes approximately 1 sec to complete. This reduction of computation was achieved by performing all calculations in the complex frequency domain. A complete description of the algorithm follows.

In the complex frequency domain the waves at each microphone can be thought of as phasors pointing in different directions. When you decrease the phase of the phasors based on the time difference required for the wave to travel the distance from a potential source location to that microphone location then all phasors will be aligned in the complex plane when one of two situations occur, the potential source location is the actual source location or in rare incidences when the potential source location happens to be at a distance that corresponds to an integer multiple of the sound’s wavelength added to the actual distance between each particular microphone and the sound source. The second situation is very rare, but can cause problems at high frequencies. However a third unwanted artifact arises when one applies the general sum and delay beamforming concept to multiple source locations, but we will discuss that later on. The following are the list of steps involved in beamforming.

1. Determine the coefficient in the Discrete Fourier Transform for all microphone inputs corresponding to the desired frequency whose source location you would like to find.
2. In general this coefficient is a complex number whose magnitude represents the amplitude of the wave of the desired frequency that contributes to the signal received by the microphone and whose angle in the complex plane represents the relative phase shift of the signal received by the microphone. Using the amplitude and the phase angle we can represent the signal at each microphone as a phasor.
3. A grid is constructed that represents the domain of the microphone array and for each location in this grid the phasors are delayed through multiplication with a complex exponential whose phase angle is the angular displacement of a wave of the desired frequency traveling from that grid location to the microphone.
4. All the phasors are added up and their summed amplitude is the value plotted at that location in the grid. This process is performed for all grid locations and the resulting matrix is the source map.

Note that because our microphone array surrounds the source from all sides we cannot use the far field approximation, which is a common method used in sum and delay beamforming to represent the computations for each grid location as matrix multiplication, which allows a pathway to more advanced beamforming algorithms that can be used to reduce the affects that arises for multiple sound sources. The following set of graphs shows the affect on the beamforming output when the frequency of the simulated monopole sound source is increased. Note the x and y position for the monopole is the same in each graph, x=y=0.5m.
Figure 16: Single source beamforming plotted at increasing frequency from 200Hz to 3000Hz at 200Hz steps. Source is located at (0.5,0.5)

The frequency increase offers a benefit up to a certain point as the width of the peak decreases, after about 1000Hz the results are not useful because it is not possible to distinguish that main peak from the rest. Now I will show a progression from f=100Hz to f=1600Hz progression of frequency, but this time for 2 sources at $x_1=-0.1, x_2=0.1, y_1=-0.1$, and $y_2=0.1$.

Figure 17: Dual source beamforming plotted at increasing frequency from 100Hz to 1600Hz at 100Hz steps. Sources are located at (-.1,-.1) and (1,1) respectively.

Here we see again that increasing frequency up to less than 1700Hz results in better source location. However, before f= 900 it is impossible to locate the actual positions of the sound sources. This may be a result of the method used for simulating the source data and it remains to be seen how this inability to locate multiple sources transfer over to real data. One possible solution to this problem would be to have more microphones around the array.
Currently there are 30 microphones in the simulations that I have shown. The following graphs will show how the output of the beamforming changes for a 400Hz 5-sound-source configuration for an increase in the number of microphones that are evenly spaced around the array.

![Simulation of 5 monopoles using 300 microphones instead of 30](image)

**Figure 19:** Multiple (5) monopole simulation using a greatly increased number of microphones (300).

The above graph was generated with 300 microphones instead of 30 and there is no noticeable enhancement in the ability to distinguish between multiple sources.

### 2.4 Experimental overview

#### 2.4.1 Experimental equipment

The equipment and software used for taking and analyzing a set of measurements is as follows:

- **Instrument:** The instrument being characterized.
- **Microphone array:** A microphone array as described in Section 2.23
- **Automated impact hammer:** Built by a previous co-op student, this device consists of a small hammer connected to a DC motor (see Figure 20). When the hammer controller is given a signal, a pulse is sent to the motor, causing it to give a little “kick”. This hammer is used to excite the instrument in a precise, repeatable fashion.
• *Data acquisition system:* (See Figure 21) Built by a previous PhD student, the data acquisition system is designed to measure the voltages of 30 microphones and send those measurements to a computer. The DAQ can sample at a rate of 40 kHz over each input channel. Currently, 3 of the 30 pins are not functioning.

• *LabView data acquisition program:* This program is located on the desktop of the computer in the anechoic chamber. The program will take in the data from the DAQ, filter it for a select range of frequencies and write to file.

• *MATLAB code:* Beamforming and radiation mapping code as detailed in Section 2.3

2.4.2 **Setup procedure**

Before the data collection and analysis can be carried out, the following setup procedure must be completed (not necessarily in order).
1. **Hang up microphone array:** To hang up the array, securely fasten the clamp at the top of the frame fixture to the anchor post hanging from the ceiling of the chamber.

2. **Plug microphones into DAQ:** Plug the microphones into the DAQ at the locations specified in Figure 22.

3. **Turn on electronics:** Plug DAQ, hammer signal conditioner, and hammer power source into the wall.

4. **Hang instrument in frame:** If beamforming data is to be taken, hang the instrument vertically, as shown on the left in Figure 14 above. If radiation map data is to be taken, hang the instrument as shown on the right of Figure 14.

5. **Align impact hammer:** Adjust the position of the impact hammer until its impact point on the instrument corresponds with the desired point. In the case of the oud, this point is on the small rubber block just above the center of the bridge. To verify your placement, trigger the hammer, observe the impact point, and listen for a clean hit.

6. **Test data collection:** Start the LabView program using a long data collection time and observe the microphone signals. Whistle/clap/make noise and see if the microphones respond.

7. **Close door:** Close the door to the chamber and switch off the lights. The data collection is now ready to be run.

![Figure 22: Pin diagram for plugging the microphones into the DAQ. Microphone 1 goes into the slot labeled 1, etc. Red labels indicate broken pins.](image)

### 2.4.3 Experimental procedure

The procedure for recording and analyzing the microphone data is as follows:

1. **Set up the LabView program:** Under Base Path, input the desired file location, select “Abbrev.” from the drop down date format list, set the data collection time (generally ~.5s), set the sampling rate (generally ~10kHz), set the number of samples to [collection time]*[sampling rate].
rate], and set the desired frequency range (generally 20Hz-2kHz). Click on each green box except for the last three such that they turn bright green.

2. **Record the data**: Click the run button on LabVIEW and quickly trigger the impact hammer. The timing on this step is very important as the hammer impact needs to occur while LabVIEW is collecting data. Note that LabVIEW will often lag and only begin collecting data ~1 second after clicking run, so this might take a few attempts.

3. **Verify the data**: Open the resultant data file in excel. Select one of the microphone data columns (any column except for 1 and 6 in our setup) and plot it. If the data run is successful, you should see a single peak followed by a steep decay. If there is no visible peak or the highest amplitudes occur at the beginning of the file, the hammer strike was likely missed by LabVIEW. Repeat step 2. If multiple peaks appear in rapid succession, the hammer is likely not set up properly and is bouncing against the instrument. If this is the case, adjust the hammer position until it is only striking the instrument once and repeat step 2.

4. **Beamform the data**
   a. Run the function PlotData, using the values 0 and 1 for fraction1 and fraction2, and any working microphone numbers for inputs. This will return a plot of the microphone voltages for the selected microphones over the entire time range.
   b. Estimate the fraction of the time range at which you would like to start beamforming (generally just after the hammer strikes) and run the function PlotData again using this estimate for fraction1. You should see the same plot as before, this time beginning from after the hammer strike.
   c. Plot the FFT of the signal data using the function PlotFFT using the first two values returned from the previous function call for the arguments start_index and end_index. This will return the FFT of the microphone data as well as the frequencies of the peaks in the FFT.
   d. Call the function DelayAndSumBF using the arguments specified in the function file. This will return the beam forming plot of the microphone data.

5. **Radiation map the data**: Call the function RadiationMap using the arguments specified in the function file. This will return the radiation maps of the instrument at the n largest frequency peaks, where n is specified in the function arguments.

### 2.5 Results

#### 2.5.1 Radiation maps
With the oud in a lengthwise orientation through the array (see Figure 23) radiation maps were generated over the 5 largest vibration modes. The results are given in Figures 24-28 below.
**FIGURE 23:** Instrument position for radiation mapping.
**Figure 24:** Radiation map of the largest mode of vibration of the oud. This is a largely symmetric mode, with a slightly larger radiation at the top than the bottom.
FIGURE 25: Radiation map of the second largest mode of vibration of the oud. This mode is much more asymmetric than the previous one, again with higher amplitudes above than below the oud.
FIGURE 26: Radiation map of the third largest mode of vibration of the oud. This mode is much more asymmetric again compared to the previous two; it is also beginning to lose the smoothness of the first two modes.


**FIGURE 27:** Radiation map of the fourth largest mode of vibration of the oud. This mode is much more regains some of the symmetry of the previous low frequency modes.
Figure 28: Radiation map of the fifth largest mode of vibration. This graph displays similar asymmetries to the other high frequency mode, although it has a much higher radiation amplitude emerging from the bottom of the instrument.

Although we have too little information to draw any objective conclusions about these maps, there are a couple of subjective trends that can be seen. One overall trend that is clear from looking at the radiation maps of these 5 largest modes is that as the frequency of the mode increases, the asymmetry of the radiation emitted increases as well. This could possibly indicate a shift from low frequency modes spread over the entire instrument to more localized, higher frequency nodes. For example, where a low frequency node coming from the first “breathing” mode of the instrument would likely be relatively symmetric, a high frequency mode consisting of vibrations in only the top soundboard would result in a highly unsymmetrical mode. In order to test this hypothesis and better understand the results coming out of the radiation maps, a useful future computation would be to perform a modal analysis on the oud. This would give use the shapes of the modes at these mapped frequency, and would let us correlate changes in the mode shape to changes in the radiation distribution.

2.5.2 Beamforming
2.5.2.1  Testing Beamforming Algorithm on Real Data
By using the point sound source available in the lab we can test out the calibration of our entire set up by checking to see how close our beamforming algorithm is to being able to calculate the actual location of the sound source. The steps that were taken to calibrate the array included obtaining accurate measurements of all microphone locations with respect to the centre of the microphone array and finding the relative sensitivities of all of the microphones. The values are now hard coded into the beamforming algorithm that is included in the appendix.

The following graph shows the calculated peak location values using MATLAB. The measured position is $x=0.449\text{m}, y=-0.197\text{m}$, which is $0.0134\text{m}$ off from the calculated value. Notice that the majority of the error is in $x$ component.

![Location of the peak for the point source calibration test at $f=350\text{Hz}$](image)

**FIGURE 29:** Point source location using beamforming algorithms.

2.5.2.2  Beamforming Sound Signal Transient Affects
For the instrument data sets the data processing is less straightforward. The oud is initially tapped on the bridge by the impact hammer to produce the sound radiation. The amplitude of this sound radiation decays over time, which means that when we take the FFT of the oud data set we are actually taking the Fourier transform of the oud data set convolved with a decaying exponential and this will alter the spectrum that we are analyzing with our beam-forming algorithm. Therefore we must choose a period of time for which our beamforming algorithm will work on the data.

There are several choices for possible regions in the data set to use for beamforming and each gives different results. By looking at the figure below it seems the best options are to either use a small region at the peak of the oscillations or a long region after the initial transients has died down. A third option would be to just use the entire region and ignore the transients. A slightly better alternative to option 3 would be to fit the peaks with a decaying exponential function and then do the beamforming on the microphone input data divided through by the fitted exponential. In this report only the first three options are explored and the fourth is left as a follow-up project. We will show that the effect of the chosen range does not significantly affect the resulting source map for instrument analysis with the impact hammer.

**FIGURE 30**: Single microphone signal for oud vibration.

Before beamforming the oud data set for a particular time range we have to decide what frequency we want to analyze. To do this we plot the FFT over the desired time range and compute the frequencies of the amplitude peaks. The FFT also changes depending on the range that we decide on. The first pair of FFT and beamforming graphs are generated for the long steady state region at the end: 0.3 seconds to 0.45 seconds.
FIGURE 31: Oud FFT over steady state vibration (0.3s-0.4s)

The main peaks in the FFT for the [0.3,0.45] sec range are 0.3131 kHz, 0.3997 kHz, 0.5063 kHz, 0.7328 kHz, 0.8328 kHz and 1.1659 kHz. The source map for f=0.3131 kHz is shown in the following graph.

FIGURE 32: Beamformed oud data under steady state vibration
The source map generated using the steady state region of the oud sound signal results in sound source locations that lay across the full sound board of the instrument.

Next we will perform the same steps using the time range that includes the full transient range, 0.1643 sec to 0.3 sec. The FFT for that range is shown below.

![Oud FFT over steady decaying vibration (0.1643s-0.3s)](image)

**FIGURE 33: Oud FFT over steady decaying vibration (0.1643s-0.3s)**

Notice that the signal looks much cleaner now. This is a combined result of the resolution of the FFT decreasing with sample range and the decrease of the signal to noise ratio. The peak frequencies are now at 170.6 Hz, 304.2 Hz, 497.0 Hz, 838.3 Hz, 749.3 Hz, and 994.1 Hz. Now let’s observe the source map for the frequency corresponding to frequency for the source map of the steady state time range from before, f=304.2 Hz.
The results are nearly identical, which indicates that the region we choose does not affect the beamforming as long as we choose either the transient region or the steady state region, but not both. What happens when we choose the full region? I will now perform the same steps as for the above two regions to generate the beam forming result when the time range is extended to between 0.1653 sec and 0.5 seconds.

**Figure 34:** Beamformed oud data under decaying vibration

**Figure 35:** Oud FFT over entire vibration range (0.1643s-0.5s)
The main peak is at \( f = 304.6595 \) Hz so we will beamform at that frequency.

**FIGURE 36:** Beamformed oud data over entire vibration range (0.1643s-0.5s)

Once again the resulting source map looks the same. This suggests that we do not need to be concerned with the region of the data that we use for generating the source maps for instruments.

### 2.5.2.3 Consistency Check for Instrument Beamforming

Three separate data sets were obtained for the oud on three separate days. Here we will show how the beamforming changes between data sets. This time we will plot a higher frequency peak so that we can see what aspects of the image are due to unwanted artifacts in the beamforming algorithm. We will plot data on the periphery of the viable frequency range where the results seem to be more corrupted.
The main lobes in this first instance appear to be resonating from the bottom edges of the oud. Overlaying this graph with the photo of the oud arrangement should verify this, but first let's look at the same frequency peak on the 2\textsuperscript{nd} set of data, which is plotted below.
FIGURE 38: Second oud data set beamformed at 1142.91Hz

The basic arrangement looks the same as before. Some additional analysis is required to understand exactly what this is telling us about how the instrument radiates at high frequencies.
FIGURE 39: Second oud data set beamformed at 1149.1Hz and overlaid on the instrument position.
As can plainly be seen what looked to be a random adverse result is actually repeatable, showing that the array may have some potential use in modal analysis.
3 Conclusions

Four main conclusions can be drawn from the results of this project: the validity of the array design, the success of single-source beamforming, the plausible success of instrument beamforming, and the ambiguity of radiation mapping.

One of the primary objectives of this project was to construct a microphone array that could measure the radiation of an instrument in all directions. We presented a design, consisting of a hula hoop array and an aluminum instrument frame in an attempt to meet these goals. The validity of the data produced by the array and the structural integrity of the assembly indicate that our presented design was successful in meeting the stated goals.

Over the course of the project, a delay and sum beamforming algorithm was developed to localize sound sources in a circular array. This algorithm was tested using both simulated and physical point sound sources. For a simulated point source, the algorithm was able to perfectly return the location of the source. It was found that when locating a single physical monopole source, the algorithm could predict the location to within 1.33cm of its actual location. When attempting multiple simulated point source localization, however, the algorithm experienced difficulty locating the sources, and often returned erroneous source locations. From this we can conclude that the developed beamforming algorithm is very successful in locating point sources, but that more work must be done to move beyond that.

The next application of the developed algorithm was to perform beamforming over the soundboard of the oud. At the low frequency resonance, the beamforming routine performed as expected, returning a large source over the soundboard of the instrument. However, as the analysis frequency increased, it became increasingly difficult to determine the validity of the resultant source maps. To try and shed some light on these maps, they were tested across multiple data sets for consistency. The consistency check came out positive, with all three data sets returning similar source maps. This indicated that there is some merit to the results from the algorithm, although further investigation is required to verify this.

One of the main stated goals of the project was to produce radiation maps for the instrument at various modal frequencies. While this was indeed accomplished, the lack of any external data made limited our ability to verify the plots. After a subjective analysis of the plot shapes, it was concluded that the plots were plausibly accurate. However, again more work will be required to verify this.
4 Project deliverables

4.1 List of deliverables

Our original intended deliverables for this project included a completed microphone array frame, which was fully accomplished and met all the design criteria at a very reasonable cost; the frame was also intended to contain a fully functional microphone array with verified measurement accuracy. The completed array meets all of the electrical design goals summarized in the following list: the spacing between measurement points was to be 0.19m, it was to be suited for microphones that have a cylindrical plastic housing with a 9' cord and a 3.5mm plug, which must be fitted onto the array at equal intervals, and in order to fit the microphone plugs directly into the DAQ system a modification had to be made. The modified outputs were verified to be accurate.

We determined the sensitivities of the individual microphones and accounted for these discrepancies by hard coding them as attenuation parameters. The impact hammer was integrated with the measurement setup to allow for automated instrument excitations. The impact hammer can be triggered from outside of the anechoic chamber with a simple switching circuit (note: this must be grounded to the wire mesh floor).

The microphone input data is an analog waveform, which was filtered using a LabView program that is available in the lab area outside of the AC. The data could be plotted in numerous ways to observe the frequency spectrum and broken up into regions of transients and steady state in order to apply beamforming and radiation mapping to the data. Currently 26 microphones are in use as the 27th input to the DAQ is being used for hammer impact input. These are mounted on the frame with equal angular spacing such that they are equidistant to the center of the frame. They are mounted with hose clamps to the tandem hula-hoop that gives our array its round shape.

Radiation maps consisting of a continuous hemisphere showing the radiation intensity at a distance equal to the radius of the hula-hoop from the instrument for a range of frequencies are attainable using the MATLAB function included in the appendix.

A GUI was implemented in MATLAB that allows the user to simulate output from the microphone array for multiple sound sources with specified frequencies, amplitudes, and locations. Also, the relative level of noise for the simulation can be specified. The GUI can be used to plot output from the array in both the frequency and time domain and with all data combined on a single graph or have each microphone output graphed separately. The GUI also allows the user to calculate the location of a monopole sound source for a specified data file by guessing the approximate location and frequency of the sound source. Two beamforming algorithms have been implemented and are also accessible through the GUI, allowing the user to generate source maps for a specified data file, frequency, and at a specified resolution. The second beamforming algorithm makes use of the phase information in the complex Fourier transform coefficients and plots the amplitude of the superposition of multiple Fourier components over a specified frequency range for each microphone output.
The entire simulation program will be handed over to the project advisor along with a full instruction manual to be used and modified by all subsequent researchers.

### 4.2 Financial summary

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>Quantity</th>
<th>Vendor(s)</th>
<th>Cost</th>
<th>Purchased by:</th>
<th>To be funded by:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tandem Hula</td>
<td>1</td>
<td>Myhoop.com</td>
<td>60$</td>
<td>Nils Smit-Anseeuw</td>
<td>Project Sponsor</td>
</tr>
<tr>
<td>2</td>
<td>Hose Clamps</td>
<td>60</td>
<td>Home depot</td>
<td>60$</td>
<td>Nils Smit-Anseeuw</td>
<td>Project Sponsor</td>
</tr>
</tbody>
</table>

### 4.3 Ongoing commitments by team members

Nils Smit-Anseeuw will be working part time during the summer on the continuation of the project. He will be working on streamlining the analysis equipment and process, as well as performing analyses on the instruments of the UBC Chinese Orchestra. This commitment is set to end at the end of July.
5 Recommendations

In order to further the work completed in this project, we make 2 main recommendations for future research. Firstly, we suggest that a modal analysis of the oud would be useful in verifying both the beamforming and radiation mapping results. Secondly, we recommend using our developed equipment and methods to analyze further instruments.

One of the main limitations of the results stemming from the project is the lack of verification. Without some external data to compare to, the validity of our results remains questionable. In order to resolve this, we recommend performing a full modal analysis on the oud body to obtain the shapes and structures of the various vibrations modes. With these shapes, we can compare the predicted mode shapes from the beamforming algorithm to the actual values. We can also interpret the results given by radiation mapping to more depth.

Now that we have developed a method for characterizing the radiation for a single stringed instrument, the logical next step is to use the same methods to characterize other, similar instruments. The UBC Chinese Orchestra has gladly agreed to lend the UBC acoustics group the use of their instruments, and analyzing these unique instruments would be a valuable opportunity. Since so little rigorous characterization has been done with these instruments, there is great potential here for exciting results.
6 Appendix: Data Analysis Code

6.1 PlotData.m

```matlab
function [fraction1_index fraction2_index hammer_blow_index size] = PlotData(file_name,inputs,fraction1,fraction2)
% file_name: tab delimited text with column 6 the impact hammer
% inputs: a vector with the microphones whose data is to be plotted
% fraction1: fraction of the entire data set that indicates plot range
% fraction2: fraction of the entire data set that indicates plot range
% note: if all data is to be plotted inputs=1:30;

    %actual computer input index
    microphone_mask = [20 26 27 24 1 18 23 16 21 8 9 19 -999 12 10 7
                        -999 11 2 -999 14 17 13 15 3 4 22 25 5 -999];

    %some might call this the microphones relative sensitivity
    % microphone_attenuation = [1 1.033262562 1.038803023 1.042979848
                                1.079107958 1.079902589 1.113834809 1.085145431 1.09456934
                                1.09952542 1.098493581 1.09797727 1.091939302 1.085591067 0.1.086948876
                                1.096280369 0 1.071335322 1.064121287 1.063082325 1.04715426 1.058979761
                                1.0590757 1.064149007 1.061996408 1.057844062 0];

    %assigns the correct x and y coordinates to each microphone and
    %999 to indicate a microphone out of use
    % A=1/1000*[274.6613894; 446.078571; 600.2211528; 726.4586121;
                815.4892413; 873.2927095; 983.2249; 875.9503403; 823.4621335;
                729.1162429; 603.5431912; 450.7294247; -999; 283.9630969;
                -80.79671972; -999; -266.8308703; -438.9124595; -999;
                -591.0618184; -717.9636854; -815.6316144; -878.7503441;
                -904.6622436; -891.37409; -833.5706218; -737.2315081;
                -608.3364181; -999];
    % B=1/1000*[857.0347996; 775.3126549; 660.3701262; 515.5292518;
                  360.0578546; 181.3321885; -9.35281581; -195.3869664;
                  -372.194094; -536.2281065; -669.1096426; -774.7504638;
                  -999; -849.164124; -894.3438463; -894.3438463; -849.8420856;
                  -767.4419793; -999; -660.4723428; -523.6041479;
                  -366.8041479; -189.4072972; 1.277707078; 188.640673;
                  369.3595621; 534.7970746; 671.0006491; -999];

    fid = fopen(file_name);
    Input = textscan(fid,'%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f',
                      'delimiter',
                      '\t'); % read all 28 inputs including time and hammer

    %determine the index for when the hammer blow occured and
    %return value for later use
    [maxhampac hammer_blow_index] = max(Input{6});

    %select region of data to examine, also return the values for
    %explorations and fourier transforming
    size = length(Input{1});
    if (fraction1_index<1)
        fraction1_index=1;
    end
    fraction1_index = ceil(fraction1*size);
```
if(fraction2_index>size)
    fraction2_index=size;
end

%plot specified data
Input{1}=Input{1}(fraction1_index:fraction2_index);

Input{microphone_mask(inputs(1))+1}=Input{microphone_mask(inputs(1))+1}(fraction1_index:fraction2_index);
plot(Input{1},Input{microphone_mask(inputs(1))+1});
hold on;
for i=2:length(inputs)
    if(microphone_mask(inputs(i))>0)
        Input{microphone_mask(inputs(i))+1}=Input{microphone_mask(inputs(i))+1}(fraction1_index:fraction2_index);
        plot(Input{1},Input{microphone_mask(inputs(i))+1});
    end
end
6.2 PlotFFT.m

function peak_freqs = PlotFFT(file_name, inputs, number_of_peaks, sample_rate, start_index, end_index)
% file_name: tab delimited text with column 6 the impact hammer
% inputs: a vector with the microphones whose data is to be plotted (the
%     first specified microphone is used to find the peak_freqs)
% sample_rate: the sampling rate of the DAQ
% number_of_peaks: the number of peak locations to return (frequencies)
% start_index: array index of first data point for FFT range
% end_index: array index of last data point
% note: if all data is to be plotted inputs=1:30;

% actual computer input index
microphone_mask = [20 26 27 24 1 18 23 16 21 8 9 19 -999 12 10 7 -999 11 2 -999 14 17 13 15 3 4 22 25 5 -999];

% some might call this the microphones relative sensitivity
% microphone_attenuation = [1 1.033262562 1.038803023 1.042979848 1.079107958 1.079902589 1.113834809 1.085145431 1.129369026 1.09456934 1.095952542 1.098493581 0 1.095952542 1.099532542 1.099902589 1.113834809 1.085145431 1.129369026 1.09456934 1.096280369 0 1.071335322 1.064121287 1.063082325 1.094715426 1.058979761 1.0590757 1.06199007 1.06199007 1.057844062 1.057844062 0];

% assigns the correct x and y coordinates to each microphone and
% -999 to indicate a microphone out of use
A = 1/1000*[274.6613894; 446.078571; 600.2211528; 726.4586121; 815.4892413; 873.2927095; 893.22494; 875.9503403; 823.4621335; 729.1162429; 603.5431912; 450.7294247; -999; 283.9630969; 102.5798001; -80.79671972; -999; -266.8308703; -438.9124595; -999; -591.0618184; -717.9636854; -815.6316144; -878.7503441; -904.6622436; -891.37409; -833.5706218; -737.2315081; -608.3364181; -999];
B = 1/1000*[857.0347996; 775.3126549; 660.3701262; 515.5292518; 360.0578546; 181.3221885; -9.352815811; -195.3869664; -372.1194094; -536.2281065; -669.1096426; -774.7504638; -999; -849.164124; -894.3438463; -894.3438463; -999; -845.820856; -767.4419793; -999; -660.4723428; -523.6043606; -366.8041479; -189.4072972; 1.277707078; 188.640673; 369.3595621; 534.7970746; 671.0006491; -999];

fid = fopen(file_name);
Input = textscan(fid, '%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f', 'delimiter', '');
% read all 28 inputs including time and hammer

% crop time array and generate frequencies
Input{1} = Input{1}(start_index:end_index);
n = length(Input{1});
frequencies = (0:n-1)*(sample_rate/n);

cn = zeros(length(inputs), n);
peak_freqs = zeros(number_of_peaks, 1);
ranges = floor(n/2);

Input{microphone_mask(inputs(1))+1} = Input{microphone_mask(inputs(1))+1}(start_index:end_index);
\begin{verbatim}
\textbf{cn}(1,:) = \text{fft}(\text{Input}\{\text{microphone}\_\text{mask}(\text{inputs}(1))+1\});

\text{Signals} = \text{figure}(\text{'XVisual',...}
\text{'0x24 (TrueColor, depth 24, RGB mask 0xff0000 0xff00 0x00ff')});
\text{axes('Parent', Signals, 'Position',[0.05 0.495 0.9 0.475]);}
\text{box('on');}
\text{grid('on');}

\text{plot(frequencies(1:range), abs(cn(1,1:range)))};
\text{hold on;}
\text{for i=2:length(inputs)}
  \text{if (microphone\_mask(\text{inputs}(i))>0)}
    \text{Input}\{\text{microphone}\_\text{mask}(\text{inputs}(i))+1\}=\text{Input}\{\text{microphone}\_\text{mask}(\text{inputs}(i))+1\}(\text{start_index:end_index});
    \text{cn(i,:) = \text{fft}(\text{Input}\{\text{microphone}\_\text{mask}(\text{inputs}(i))+1\});}
    \text{plot(frequencies(1:range), abs(cn(i,1:range)))};
  \text{end}
\text{end}

\text{axes('Parent', Signals, 'Position',[0.05 0.025 0.9 0.475]);}
\text{box('on');}
\text{grid('on');}

\text{hold on;}
\text{for i=1:length(inputs)}
  \text{if (microphone\_mask(\text{inputs}(i))>0)}
    \text{plot(frequencies(1:range), 180/pi*atan2(imag(cn(i,1:range)), real(cn(i,1:range)))};
  \text{end}
\text{end}

\text{for i=1:number\_of\_peaks}
  \text{[am id] = max(abs(cn(1,1:range)))};
  \text{peak\_\text{freqs}(i) = frequencies(id);}
  \text{cn(1,id)=0;}
\text{end}
\end{verbatim}

\subsection{DelayAndSumBF.m}

\textbf{function}
\text{DelayAndSumBF(file\_name, resolution, sample\_rate, frequency, start\_index, end\_index)}
\begin{verbatim}
c=343; \% this constant is very important to have accurate!!!!!
\% actual computer input index
microphone\_mask = [20 26 27 24 1 18 23 16 21 8 9 19 -999 12 10 7
-999 11 2 -999 14 17 13 15 3 4 22 25 5 -999];
\% if assuming symmetry then...
microphone\_mask = [20 26 27 24 1 18 23 16 21 8 9 19 -999 12 10 7
-999 11 2 -999 14 17 13 15 3 4 22 25 5 26];
\% some might call this the microphones' relative sensitiv45
\end{verbatim}
microphone_attenuation = [1 1.033262562 1.038803023 1.042979848 1.079107958 1.079902589 1.113834809 1.085145431 1.129369026 1.09456934 1.099532542 1.098493581 0 1.095597727 1.091939302 1.085591067 0 1.086948876 1.096280369 1.071335322 1.064121287 1.063082325 1.094715426 1.058979761 1.05908757 1.064149007 1.061996408 1.057844062 0];

%if assuming symmetry then...

radius = 0.89804801; %average radius for all microphones

%assigns the correct x and y coordinates to each microphone and -999 to indicate a microphone out of use

A=1/1000*[274.6613894; 446.078571; 600.2211528; 726.4586121; 815.4892413; 873.2927095; 893.22494; 875.9503403; 823.4621335; 729.1162429; 603.5431912; 450.7294247; -999; 283.9630969; 102.5798001; -80.7967197; -999; -266.8308703; -438.9124595; -999; -591.0618184; -717.9636854; -815.6316144; -878.7503441; -904.6622436; -891.37409; -833.5706218; -737.2315081; -608.3364181; -450.8717978];

%if symmetry

B=1/1000*[857.0347996; 775.3126549; 660.3701262; 515.5292518; 360.0578546; 181.3321885; -9.352815811 -372.1194094; -536.2281065; -669.1096426; -774.7504638; -999; -849.1641218; -999; -660.4723428; -523.6043606; -366.8041479; -189.4072972; 1.277707078; 188.640673; 369.3595621; 534.7970746; 671.0006491; 783.2855471];

%783.2855471];  if symmetry

fid = fopen(file_name);
Input = textscan(fid, '%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f%f' ,'delimiter', 't');</pre>
%2nd lowest frequency length
microphone_input(k,5)=abs(C_n(index+1))*microphone_attenuation(k);

%2nd lowest frequency angle
microphone_input(k,6)=atan2(imag(C_n(index+1)),real(C_n(index+1))); 

%2nd highest frequency length 
microphone_input(k,7)=(index+1)*sample_rate/n; 

%2nd highest frequency angle 
microphone_input(k,9)=atan2(imag(C_n(index+2)),real(C_n(index+2))); 

%highest frequency 
end 

if microphone_mask(k)<0 % if pin is dead then that microphone does not contribute

%lowest frequency
microphone_input(k,1)=0;

%lowest frequency length
microphone_input(k,2)=0;

%lowest frequency angle
microphone_input(k,3)=0;

%2nd lowest frequency
microphone_input(k,4)=0;

%2nd lowest frequency length
microphone_input(k,5)=0;

%2nd lowest frequency angle
microphone_input(k,6)=0;

%2nd highest frequency
microphone_input(k,7)=0;

%2nd highest frequency length
microphone_input(k,8)=0;

%2nd highest frequency angle
microphone_input(k,9)=0;

%highest frequency
microphone_input(k,10)=0;

%highest frequency length
microphone_input(k,11)=0;

%highest frequency angle
microphone_input(k,12)=0;
end
end

%delay the phase angle based on the location of the grid point 
and create a vector that spans a full frequency spectrum that 
represents the sum of all the delayed sinusoids added together 
and plot the max value in this vector
m=ceil((radius*2)/resolution); 
actual_resolution = radius*2/m; 
data=zeros(m,m);

for i=1:m %iterate through the rows y
for j=1:m %and columns x
    phase_sum = 0;
    r_i = sqrt((-radius+actual_resolution*(j-1+1/2))^2+(radius-actual_resolution*(i-1+1/2))^2);
    if(r_i<=radius)
        for k=1:30
            r_ii = sqrt(((radius+actual_resolution*(j)-A(k))^2+(radius-actual_resolution*(i)-B(k))^2);
                phase_sum=phase_sum+microphone_input(k,2)*exp(sqrt(-1)*(microphone_input(k,3)-2*pi/c*microphone_input(k,1)*r_ii))+microphone_input(k,5)*exp(sqrt(-1)*(microphone_input(k,6)-2*pi/c*microphone_input(k,4)*r_ii))+microphone_input(k,8)*exp(sqrt(-1)*(microphone_input(k,9)-2*pi/c*microphone_input(k,7)*r_ii))+microphone_input(k,11)*exp(sqrt(-1)*(microphone_input(k,12)-2*pi/c*microphone_input(k,10)*r_ii));
        end
        data(i,j) = abs(phase_sum);
    else
        data(i,j) = 0;
    end
end
end

count = length(data(1,:));
X=-radius+actual_resolution*(count-1+1/2);
Y=radius-actual_resolution*(count-1+1/2);
surf(X,Y,data);

peakamp = max(max(data))
[row col] = find(data==peakamp);
peakxpos = radius-actual_resolution*(row-1+1/2)
peakypos = radius-actual_resolution*(col-1+1/2)
hold on;
scatter3(peakypos,-peakxpos,peakamp);

[row2 col2] = find(data>peakamp*19/20);
halfpeakradius = sqrt(max(((radius-actual_resolution*(row2-1+1/2)-peakxpos).^2 + (radius-actual_resolution*(col2-1+1/2)-peakypos).^2)))
for i=1:length(row3)
    if halfpeakradii(row3(i))>halfpeakradius
        halfpeakradius=halfpeakradii(row3(i));
    end
end

t=0:pi/50:2*pi;
z=(zeros(101,1)+1)*peakamp*19/20;
x=peakypos+halfpeakradius*cos(t);
y=-peakxpos+halfpeakradius*sin(t);
scatter3(x,y,z);

hold off;

\%
\%     B=1/1000*[857.0347996; 775.3126549; 660.3701262; 515.5292518; 360.0578546; 181.3321885; -9.352815811; -195.3869664; -372.1194094; -536.2281065; -669.1096426; -774.7504638; -849.164124; -894.3438463; -894.3438463; -845.8420856; -767.4419793; -660.4723428; -523.6043606; -366.8041479; -189.4072972; 1.277707078; 188.640673; 369.3595621; 534.7970746; 671.0006491];
\%
\%     A=1/1000*[274.6613894; 446.078571; 600.2211528; 726.4586121; 815.4892413; 873.2927095; 893.22494; 875.9503403; 823.4621335; 729.1162429; 603.5431912; 450.7294247; 283.9630969; 102.5798001; -80.79671972; -266.8308703; -438.9124595; -591.0618184; -717.9636854; -815.6316144; -878.7503441; -904.6622436; -891.37409; -833.5706218; -737.2315081; -608.3364181];
\%
\%     hold on
\%
\%     C= zeros(26,1)+1;
\%
\%     C=C*max(data);
\%
\%     scatter3(A,B,C);
\% end
6.4 RadiationMap.m

function [theta amplitude] = RadiationMap(file_name,n_Peaks,peak_Width,sample_rate,start_time,end_time,microphone_number)
%file_name: a file with the hammer output in row G and tab delimited text format
%n_Peaks: the number of frequencies to be plotted
%peak_Width: assumed width of a frequency peak (suggested: 10Hz) (Hz)
%sample_rate: sampling rate (duh!) (Hz)
%start_time: how long to wait after the hammer blow before accepting data (seconds)
%end_time: time range for accepting data (seconds)
%microphone_number: the number of the mic that you want to plot's FFT
%example: [t p]=RadiationMap('Good_90degRot_ForRadMap_10kHz_d5s.txt',50,10000,0.1,0.3,8);
polar(t,p); tt=1.2607:-0.1:5.0225; pp = spline(t,p,tt); polar(tt,pp);
%note: may want to take fourier transform of data to determine frequencies
%to try

%actual computer input index
microphone_mask = [20 26 27 24 1 18 23 16 21 8 9 19 -999 12 10 7
-999 11 2 -999 14 17 13 15 3 4 22 25 5 -999];

%some might call this the mirophones relative sensitivity
microphone_attenuation = [1 1.033262562 1.038803023 1.042979848
1.079107958 1.079902589 1.113834809 1.085145431 1.129369026 1.09456934
1.099532542 1.098493581 0 1.095597727 1.091939302 1.085591067 0 1.086948876
1.096280369 0 1.071335322 1.064121287 1.063082325 1.094715426 1.058979761
1.05908757 1.06419007 1.061996408 1.057844602 0];
radius = 0.89804801; %average radius for all microphones
%sample_rate =40000;

%assigns the correct x and y coordinates to each microphone and
%-999 to indicate a microphone out of use
A=1/1000*[274.6613894; 446.078571; 600.2211528; 726.4586121;
815.4892413; 873.2927095; 893.22494; 875.9503403; 823.4621335; 729.1162429;
603.541912; 450.7294247; -999; 283.9621396; 102.5798001; -80.79671972; -999;
-266.8308703; -438.9124595; -999; -591.0618184; -717.9636854; -815.6316144; -
878.7503441; -904.6622436; -891.37409; -833.5706218; -737.2315081; -
608.3364181; -450.8717978; -282.7766546];
B=1/1000*[857.0347996; 775.3126549; 660.3701262; 726.4586121;
360.0578546; 181.3321885; -9.352815811; -195.3869664; -372.1195409; -
536.2281065; -669.1096426; -774.7504638; -999; -849.164124; -894.3438463; -
894.3438463; -999; -845.8402856; -767.4419793; -999; -600.4723428; -
523.6043606; -366.8041479; -189.4072972; 1.277707078; 188.640673;
369.3595621; 534.7970746; 671.0006491; 783.2855471; 861.6856534];
%microphone Positions =[A B];

%initialize the polar points for the radiation maps
theta = atan2(B,A);
amplitude = zeros(30,n_Peaks);

fid = fopen(file_name, 'w');
Input = textscan(fid, '.getvalue', 'delimiter', ''); % read all 28 inputs including time and hammer

% determine the index for when the hammer blow occurred and
% print value to screen for later use
[maxo ind] = max(Input{6})

% select region of data to examine, also print this for
% explorations and fourier transforming
start_index = ind(1) + floor(start_time*sample_rate)
end_index = ind(1) + floor(start_time*sample_rate) + floor(end_time*sample_rate)

% plot data to ensure that the range is correct
Input{1}=Input{1}(start_index:end_index);
Input{2}=Input{2}(start_index:end_index);
plot(Input{1},Input{2});
for i=3:28
    Input{i}=Input{i}(start_index:end_index);
end

%... then plots the FFT for good measure...
 cn=fft(Input{microphone_mask(microphone_number)+1});
 n=length(Input{1}); freq = (0:n-1)*(sample_rate/n);
 plot(freq(1:floor(n/2)),abs(cn(1:floor(n/2))));

peaks = mspeaks(freq(1:floor(n/2))',abs(cn(1:floor(n/2))));
 [peakAm peakId] = sort(peaks(:,2));
 peakFreq = peaks(peakId(end+1-(1:n_Peaks)));
 for i = 1:n_Peaks
    % generate the amplitudes for the radiation maps
    index = floor(n*(peakFreq(i)-peak_Width)/sample_rate):ceil(n*(peakFreq(i)+peak_Width)/sample_rate);
    for k=1:30
       if microphone_mask(k)>0
          % use fourier transform to determine maximum
          frequency
          C_n = fft(Input{microphone_mask(k)+1});
          c=abs(C_n(index))*microphone_attenuation(k);
          amplitude(k,i) = max(c);
       end
       if microphone_mask(k)<0 % if pin is dead then that
          microphone does not contribute
       end
    end
 end
end

ind = [13 17 20]; % indices to be removed
amplitude(30,:) = amplitude(2,:);
amplitude(31,:) = amplitude(1,:);
amplitude(ind,:) = []; % remove those little tiddly winks
theta(ind) = []; % remove those little tiddly winks
theta(22:28) = theta(22:28) - 2*pi;
theta(end+1) = theta(1) - 2*pi;
amplitude(end+1,:) = amplitude(1,:);
tt = theta(1) + (1:1000)/1000*(theta(end) - theta(1));

% generate a splined radiation map plot for each frequency peak
figList = zeros(1,n_Peaks);

for i = 1:n_Peaks
    ampSpl = spline(theta,amplitude(:,i),tt);
    figList(i) = figure('Name',[num2str(peakFreq(i)) 'Hz (Peak ' num2str(i) ') Radiation Map'], 'NumberTitle', 'off','Position',[100,50,500,500]);
    polar(tt,10/log(10)*log(ampSpl))
    title({[num2str(peakFreq(i)) 'Hz (Peak ' num2str(i) ') Radiation Map']; 'Radius represents interpolated microphone voltage at 1V reference (dB)'});
end
end
7 References