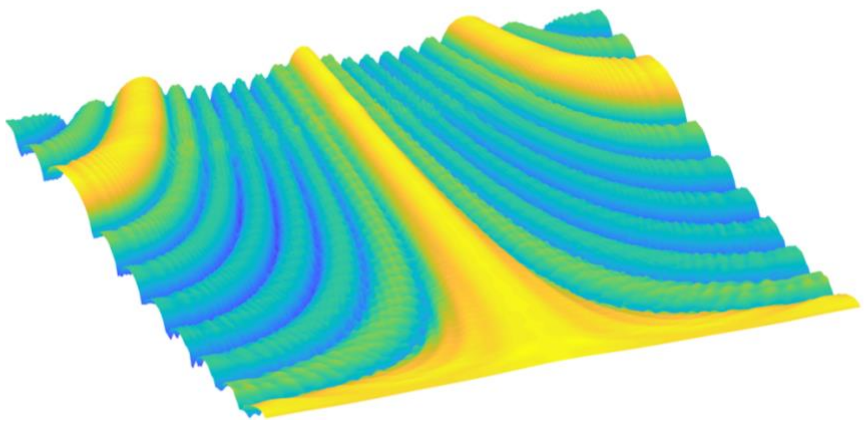

Signal Processing for Circular Microphone Array Beamforming

Interim Report

2018-19 Final Year Project

Supervisor: DR PATRICK A. NAYLOR
Student: Libang Liang
CID: 01204497

Imperial College London, EEE

Contents

1. Project Objectives and Specification

The objective of this whole project is to simulate spatial position of the source and microphone arrays, develop an algorithm in MATLAB which could extract the desired signal source from background noise, interfering sources and reverberation effects. To elaborate, the project will study microphone array beamforming and uses as a technique to achieve a directionally sensitive microphone. Smart speakers like Google Home and Alexa Echo use such technique for higher speech recognition accuracy.

To accomplish the project aim mentioned above, different microphone array beamforming techniques will be implemented on MATLAB, and their performance will be studied. Practical example like ceiling-mounted microphone array will also be simulated and investigated. The microphone array with its corresponding beamforming technique that achieves the best replication of the source signal will be selected. As extensions of this project, I will also attempt to study and simulate how to estimate source direction if time allows. A rigorous estimation of source coming direction is as crucial as beamforming techniques. Table 1 presents the project deliverables

Deliverable 1	Literature Review of most popular microphone array beamforming techniques
Deliverable 2	Linear Array Related Simulation
Deliverable 3	Spatial Aliasing Analysis
Deliverable 4	Circular Array Related Simulation
Deliverable 5	Minimum Variance Distortionless Response Beamformer Simulation
Deliverable 6	Ceiling-mounted Circular Array Simulation

Table 1 Deliverables

2. Background

2.1 Sound Travels In a Room

In a typical room environment, the desired speech signal originates from a talker's mouth, and is corrupted by interfering signals such as reverberation and other talkers [1]. When a single omni-directional microphone is used to record the sound, it collects the sound in all direction, this includes both speaker's direction, namely look direction, and all other unwanted direction. Unwanted noise such as coffee machine operating sound, other speakers' voice are all included in the output of the microphone. Speech enhancement could of course implemented in single microphone, but single microphone speech enhancement techniques only use the time-frequency information present in the signals and can therefore be considered a frequency filtering of noisy speech signal [2]. However, if an array of microphones is used, where microphones are placed in disparate positions spatially, multiple signals which contains spatial characteristics are recorded. Different outputs are collected due to microphones' geographical position. After processing using technique like beamforming, microphone array could achieve characteristics which single omni-directional microphone does not have such as directivity.

2.2 Microphone Array Industry

Microphone array is defined as a system consists of number of microphones distributed in space in certain pattern [3]. This system is widely used in many fields such as extracting voice input from ambient noise in electronic devices, locating objects, high fidelity original recordings and environmental noise monitoring [4]. With the fast development of the smart speaker industries in recent years, the demand for microphone array and beamforming technology is in a great escalation

[5]. For example, the following table(Table 2) shows a list of smart speaker products. All of them contain microphone arrays to filter out the ambient noise and enhance signals from desired direction.

Product name	Microphone array installed	Technology
Apple Homepod [6]	Six-microphone array for far-field Siri	Advanced signal processing, together with echo and noise cancellation
Amazon Echo(2nd Gen) [7]	Array of seven microphones	Beam-forming technology and enhanced noise cancellation
Google Home [8]	Dual microphones	Far-field voice recognition

Table 2 Smart Speakers and Microphone Arrays

Compare to single microphone with speech enhancement technique, the use of microphone array beamforming gives one the opportunity to exploit the fact that the source of the desired signal and the noise sources are physically separated in space. And there are many common scenarios where the desired signal and unwanted noise are spatially separated. So investigating microphone array beamforming technique is worthwhile.

2.3 Software Tools

The model simulation in this project is carried out by MATLAB Simulink together with a MEX-file(MATLAB executable file) named Room Impulse Response Generator by Audio Labs [9]. This script is able to generate multichannel Room Impulse Responses using the image method. This function enables the user to control the reflection order, room dimension and microphone directivity. By simply indicating the source and receiver positions in the room, transfer function could be obtained. The reason for using a MEX-file instead of standard MATLAB function is due to computation time, using MEX-functions is around 100 times faster than the standard MATLAB code [10]. In addition this file also considered fractional sampling issue so there is no need to concern about fractional sampling error in the remaining project.

2.4 Background information/ Literature survey

2.4.1 Array Geometry and Interpretation

The most common geometry for array processing applications is typically an equally-spaced array (Figure 1 and Figure 2), usually with a spacing of one half-wavelength at the highest frequency of operation [11]. The reason for this spacing selection is to avoid spatial aliasing. To elaborate, take linear array as example, microphones evenly placed in space sample the sound field in both space and time. The spatial sampling must occur at a rate greater than half of the wavelength of the highest frequency sinusoid in order to form an unambiguous beampattern [12], this is known as spatial Nyquist criterion.

$$l \leq \frac{\lambda}{2} \tag{2.1}$$

Where l is the spacing between each elements and λ is the wavelength of the recording signal.

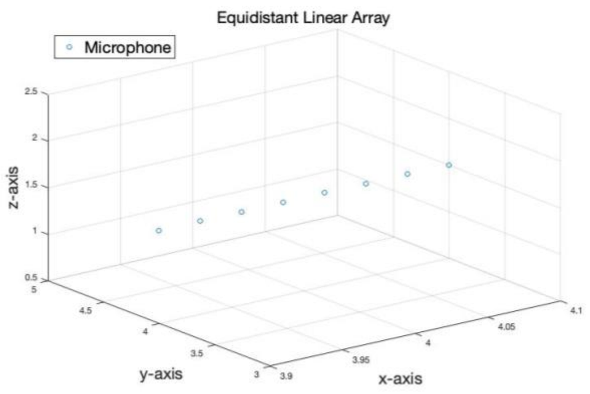

Figure 1 Linear Array

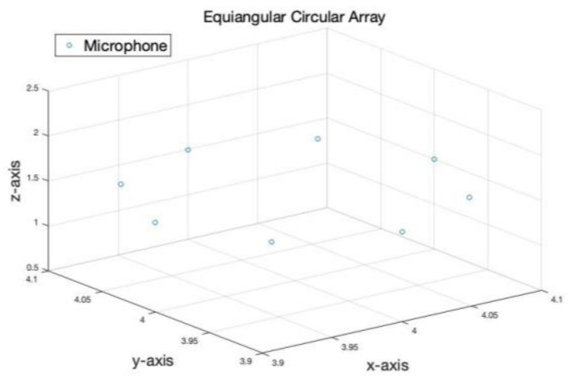

Figure 2 Circular Array

2.4.2 Evaluation of Beamformers

To get a better understanding of the features of the different designs of optimal beamformers, different measures are used to analyze beamformers' performances.

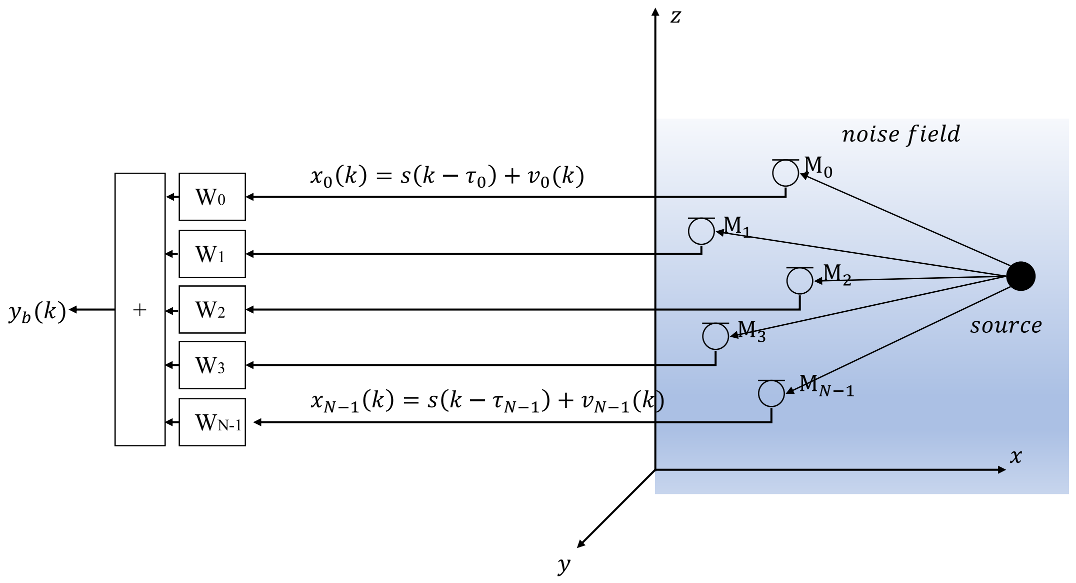

Figure 3 Signal model consisting of noise field and desired source signal

Signal model shown in Figure 3 illustrates that one sample of the discrete input sequence $x(k)$ at each sensor n is the sum of the delayed and attenuated version of the source signal $a_i s(k - \tau)$ and noise component $v_i(k)$ with arbitrary spatial statistics.

$$\begin{pmatrix} x_0(k) \\ x_1(k) \\ \vdots \\ x_{N-1}(k) \end{pmatrix} = \begin{pmatrix} a_0 s(k-\tau_0) \\ a_1 s(k-\tau_1) \\ \vdots \\ a_{N-1} s(k-\tau_{N-1}) \end{pmatrix} + \begin{pmatrix} v_0(k) \\ v_1(k) \\ \vdots \\ v_{N-1}(k) \end{pmatrix} \tag{2.2}$$

Writing in matrix form gives:

$$\boldsymbol{x}(k) = \boldsymbol{\alpha s}(k-\boldsymbol{\tau}) + \boldsymbol{v}(k) \tag{2.3}$$

Transforming this in to discrete frequency domain using Fourier Transform leads to

$$\boldsymbol{X}(e^{j\Omega}) = \boldsymbol{S}(e^{j\Omega})\boldsymbol{d} + \boldsymbol{V}(e^{j\Omega}) \tag{2.4}$$

Where $\boldsymbol{d}$ represents the attenuation and delays in frequency domain, which is $\boldsymbol{a}$ and $\boldsymbol{\tau}$ in time domain. This parameter depends on spatial location of the sensors as well as the direction of the desired signal. So $\boldsymbol{d}$ could be expressed as

$$\boldsymbol{d^T} = [a_0 e^{-j\Omega\tau_0}, a_1 e^{-j\Omega\tau_1}, a_2 e^{-j\Omega\tau_2} \dots a_N e^{-j\Omega\tau_N}] \tag{2.5}$$

According to the model in Figure 3, the output signal could be expressed as

$$Y_b(e^{j\Omega}) = \sum_0^{N-1} W_n{}^*(e^{j\Omega}) X_n(e^{j\Omega}) = \boldsymbol{W}^H \boldsymbol{X} \tag{2.6}$$

Where $\boldsymbol{W}$ is the frequency-domain coefficients of the beamformer of the sensor *n* at the frequency Ω. The inverse Fourier transform gives the time domain output $y_b(k)$.

Array-Gain

Array-Gain (AG) evaluates the enhancement of the signal-to-noise ratio(SNR) of the output(whole array) when one sensor's SNR changes. It is defined as

$$G = \frac{SNR_{Array}}{SNR_{Sensor}} \tag{2.7}$$

The SNR of one sensor is the ratio of the power spectral densities(PSD) of the signal Φ_{SS} and the average noise $\Phi_{V_aV_a}$:

$$SNR_{Sensor} = \frac{\Phi_{SS}}{\Phi_{V_aV_a}} \tag{2.8}$$

Since the input-output relationship is given by $\boldsymbol{Y} = \boldsymbol{W}^H\boldsymbol{X}$, the PSD relationship is then:

$$\Phi_{YY} = \boldsymbol{W}^H \boldsymbol{\Phi}_{\boldsymbol{XX}} \boldsymbol{W} \tag{2.9}$$

When there is only desired signal is present, the output is:

$$\Phi_{YY\ Signal} = \Phi_{SS}|\boldsymbol{Wd}|^2 \tag{2.10}$$

Assuming only noise is present, the output is:

$$\Phi_{YY\ Noise} = \Phi_{V_aV_a}\boldsymbol{W}^H\boldsymbol{\Phi}_{\boldsymbol{VV}}\boldsymbol{W} \tag{2.11}$$

Where $\boldsymbol{\Phi}_{\boldsymbol{VV}}$ is the normalized cross power spectral density matrix of the noise, where the normalization factor is set so the trace of the matrix is equal to N.

So

$$SNR_{Array} = \frac{\Phi_{SS}|\boldsymbol{Wd}|^2}{\Phi_{V_aV_a}\boldsymbol{W}^H\boldsymbol{\Phi}_{\boldsymbol{VV}}\boldsymbol{W}} \tag{2.12}$$

Rearrange (2.7) gives:

$$G = \frac{|\boldsymbol{W}^H\boldsymbol{d}|^2}{\boldsymbol{W}^H\boldsymbol{\Phi}_{\boldsymbol{VV}}\boldsymbol{W}} \tag{2.13}$$

Assuming that the noise filed is homogeneous, $\boldsymbol{\Phi}_{\boldsymbol{VV}}$ can be replaced by the coherence matrix:

$$\boldsymbol{\Gamma_{VV}} = \begin{pmatrix} 1 & \Gamma_{V_0V_1} & \Gamma_{V_0V_2} & \cdots & \Gamma_{V_0V_{N-1}} \\ \Gamma_{V_1V_0} & 1 & \Gamma_{V_1V_2} & \cdots & \Gamma_{V_1V_{N-1}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \Gamma_{V_{N-1}V_0} & \Gamma_{V_{N-1}V_1} & \Gamma_{V_{N-1}V_2} & \cdots & 1 \end{pmatrix} \tag{2.14}$$

$\Gamma_{V_nV_m}(e^{j\Omega})$ is the coherence function of noise in different sensors:

$$\Gamma_{V_nV_m}(e^{j\Omega}) = \frac{\Phi_{V_nV_m}(e^{j\Omega})}{\sqrt{\Phi_{V_nV_n}(e^{j\Omega})\Phi_{V_mV_m}(e^{j\Omega})}} \tag{2.15}$$

Thus,

$$G = \frac{|\boldsymbol{W}^H\boldsymbol{d}|^2}{\boldsymbol{W}^H\boldsymbol{\Gamma_{VV}}\boldsymbol{W}} \tag{2.16}$$

Transforming the PSD matrix of the noise to coherence matrix allows a better comparison between different noise field.

Directivity Index

Directivity Index(DI) describes the ability of the array to suppress a diffuse noise field and enhance the signal from the desired direction. It is defined as

$$DI(e^{j\Omega}) = 10log_{10}\left(\frac{\left|H(e^{j\Omega},\varphi_0,\theta_0)\right|^2}{\frac{1}{4\pi}\int_0^{\pi}\int_0^{2\pi}|H(e^{j\Omega},\varphi,\theta)|^2\sin(\theta)\,d\varphi d\theta}\right) \tag{2.17}$$

This definition can be interpreted as the ratio of the transfer function at desired direction(look-direction) φ_0 θ_0 of the array with respect to the average transfer function over all directions(using spatial integration).

Front-to-Back Ratio

For the applications, the look-direction is always the same(conference or orchestras). For these cases, calculating front-to-back ratio(FBR) is better than DI. This is because the wanted signal is always at the same direction, all other directions could be considered unwanted noise.

The formal definition of FBR uses again the transfer function

$$FBR(e^{j\Omega}) = \frac{\int_{\theta_0-\frac{\pi}{2}}^{\theta_0+\frac{\pi}{2}}\int_{\varphi_0-\frac{\pi}{2}}^{\varphi_0+\frac{\pi}{2}}\left|H(e^{j\Omega},\varphi_0,\theta_0)\right|^2\sin(\theta)\,d\varphi d\theta}{\int_{\theta_0+\frac{\pi}{2}}^{\theta_0+\frac{3\pi}{2}}\int_{\varphi_0+\frac{\pi}{2}}^{\varphi_0+\frac{3\pi}{2}}|H(e^{j\Omega},\varphi,\theta)|^2\sin(\theta)\,d\varphi d\theta} \tag{2.18}$$

White Noise Gain

White noise gain measures the ability of the array to suppress spatially uncorrelated noise.

For spatially uncorrelated noise, the coherence function $\boldsymbol{\Gamma_{VV\,|uncorr}} = \boldsymbol{I}$, insert in the AG, we get:

$$WNG(e^{j\Omega}) = \frac{|\boldsymbol{W}^H\boldsymbol{d}|^2}{\boldsymbol{W}^H\boldsymbol{W}} \tag{2.19}$$

On a logarithmic scale, positive values states an attenuation of uncorrelated noise, negative shows an amplification.

2.4.3 Continuous Array Spatial Filtering

This section introduces some basic ideas for how microphone array achieves directivity characteristics and some simple derivation on outputs behaviors where outputs are obtained from linear or circular array when the signal are coming from different direction. Although in simulation it is not possible to have a continuous microphone array, the following derivation gives basic understanding on microphone arrays and their corresponding algorithm.

Consider a sound wave originates from a far field source perpendicular to a continuous linear array, when it reaches the receivers, it could be considered as a plane wave, i.e. a wave whose value, at any moment, is constant over any plan that is perpendicular to a fixed direction in space [13]. As shown in Figure 4, when the plane wave is perpendicular to a linear array, the array output is maximized. Therefore far field directivity can be achieved using a continuous linear array.

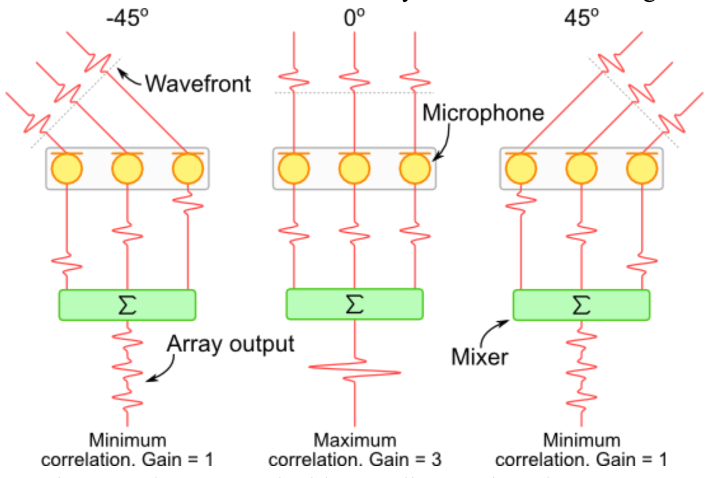
-45°
0°
45°
Wavefront
Microphone
Σ
Array output
Mixer
Minimum correlation. Gain = 1
Maximum correlation. Gain = 3
Minimum correlation. Gain = 1

Figure 4 Plane waves incident on linear microphone arrays

However, if the continuous array has a circular geometry the signals will no longer be in-phase when summed, therefore the continuous circular array will not show the directivity obtained when using a linear array. Mathematical proof is given below.

2.4.3.1 Linear Arrays [14]

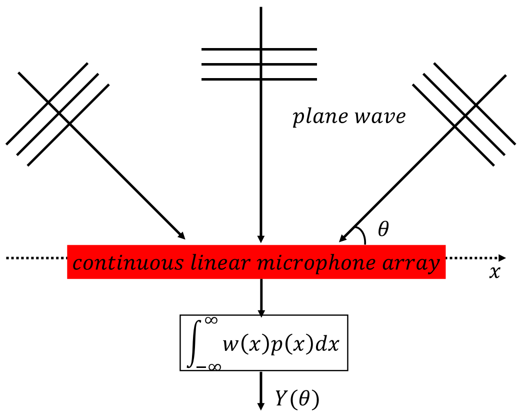
plane wave
θ
continuous linear microphone array
x
$\int_{-\infty}^{\infty} w(x)p(x)dx$
$Y(\theta)$

Figure 5 Linear Continuous Array Model

Shown in Figure 5, a plane wave with an angle θ and wavelength λ, passes through a continuous linear array, with an assumed infinite length, position on the x-axis. The output of the system is given by

$$\int_{-\infty}^{\infty} w(x)p(x)dx \tag{2.20}$$

Where $w(x)$ is the weighting function that represent the contribution of each element of the array to the output of the system and $p(x)$ is the acoustic pressure.

$$p(x) = e^{-ikx\sin\theta} = e^{-ik_x x} \tag{2.21}$$

Where k is the wavenumber of the plane wave and it is defined as $k = \frac{2\pi}{\lambda}$, λ is the wavelength of the acoustic plane wave.

The wavenumber component in the x-direction is given by $k_x = k\sin\theta$.

Assuming in reality the linear array has finite length L. Then the weighting function could be write as a rectangular aperture function

$$w(x) = \begin{cases} 1/L\,, -L/2 < x < L/2 \\ \quad 0, otherwise \end{cases} \tag{2.22}$$

This is the simplest form of weighting functions where output can be interpreted as taking the average of each element in the array. Substituting the (2.38) and (2.37) into (2.36) gives

$$Y(k_x) = \frac{1}{L} = \int_{-L/2}^{L/2} e^{-ik_x x}\, dx = \frac{1}{ik_x L}\left(e^{ik_x L/2} - e^{-ik_x L/2}\right) \tag{2.23}$$

Which gives an relationship between output and wavenumber k_x. Using the fact that $e^{ia} - e^{-ia} = 2i \sin a$, the output becomes

$$Y(k_x) = \frac{\sin[(kL/2)\sin\theta]}{(kL/2)\sin\theta} \tag{2.24}$$

This is a “sinc” function, its zero crossings closest to the origin are given by

$$\left(\frac{kL}{2}\right)\sin\theta = \pm\pi \tag{2.25}$$

When the plane wave passes the linear microphone array at angle θ. Using the definition of wavenumber $k = \frac{2\pi}{\lambda}$,

$$\theta = \pm\sin^{-1}(\lambda/L) \approx \lambda/L \quad (\lambda \ll L) \tag{2.26}$$

This equation indicates that when the plane wave come at angle $\pm\sin^{-1}(\lambda/L)$, the output $Y(k_x)$ is zero, which means the directivity of the main lobe of the linear array is situated within $-\lambda/L$ and λ/L. Thus it can be conclude that the width of the main lobe is inversely proportional to array length. Longer the array, better the directivity(when frequency of the plane wave keep constant).

2.3.6.2 Circular Array [14]

A similar arrangement is setup except for the linear array is replaced by a circular array. The array again receives a plane wave from angle θ with wavelength λ.

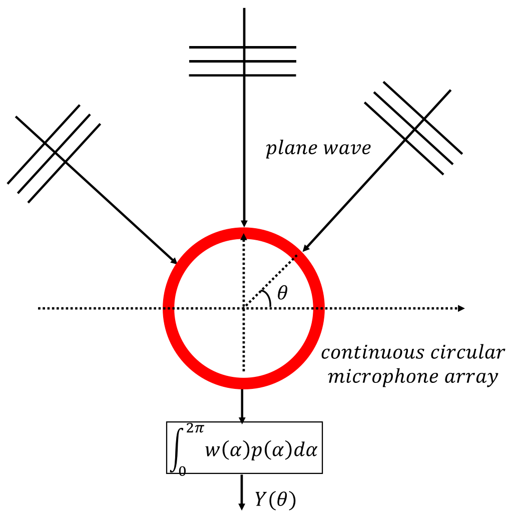
plane wave
θ
continuous circular microphone array
$\int_0^{2\pi} w(\alpha)p(\alpha)d\alpha$
$Y(\theta)$

Figure 6 Circular Continuous Array Model

The output of the array is given by integrating the product of the weighting function and received signal at all element from 0 to 2π

$$\int_0^{2\pi} w(\alpha)p(\alpha)d\alpha \tag{2.27}$$

The complex pressure of a plane wave travelling in two-dimensions can be written as [15]

$$p_p(\boldsymbol{x}) = p_p(\boldsymbol{y})e^{-i\boldsymbol{k}(\boldsymbol{x}-\boldsymbol{y})} \tag{2.28}$$

Where $p_p(\boldsymbol{y})$ is also a complex pressure at position $\boldsymbol{y}$ with wavenumber vector $\boldsymbol{k}$ with modulus k.
Now assuming $p_p(\boldsymbol{y}) = A$ and $\boldsymbol{y}$ is at origin $(0,0)$, $\boldsymbol{x}$ is on circular array which radius is r. Then the coordinate of $\boldsymbol{x}$ is $(r\cos a, r\sin a)$, when a plane wave incidents on the circular array with angle θ, then,

$$p_p(\boldsymbol{x}) = Ae^{-ikr(\cos a\sin\theta+\sin a\cos\theta)} \tag{2.29}$$

Using the trigonometric property, $\cos(a-\theta) = (\cos a\sin\theta + \sin a\cos\theta)$, (2.29) becomes

$$p_p(\boldsymbol{x}) = Ae^{-ikr\cos(a-\theta)} \tag{2.30}$$

Uniform weighting function for a circular array can be written as

$$w = \frac{1}{2\pi r} \quad 0 < a < 2\pi \tag{2.31}$$

Substitute above derivation into the output equation gives

$$Y(a) = \frac{A}{2\pi r}\int_0^{2\pi} e^{[-ikr\cos(a-\theta)]}\,da \tag{2.32}$$

From the mathematical property

$$J_n(z) = \frac{1}{2\pi i^n}\int_0^{2\pi} e^{iz\cos a}e^{ina}\,da \tag{2.33}$$

Assuming $n = 0$ and θ is constant, the output becomes

$$Y(kr) = \frac{AJ_0(-kr)}{r} \tag{2.34}$$

Where $J_0(-kr)$ is the Bessel function of the first kind of order zero [16], see Figure 7.

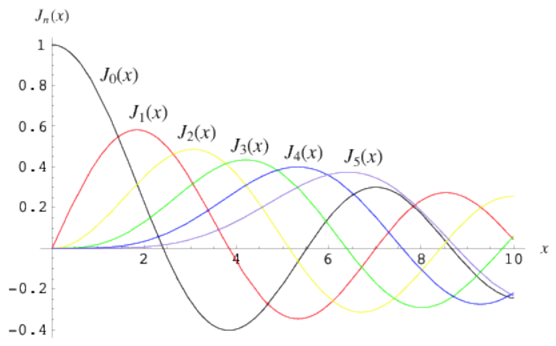

Figure 7 Bessel Function

This is an important result showing that the output does not depends on the angle of the coming plane wave, in other words, if a continuous circular array has uniform filter weighting the array is omni-directional and has no directivity.

2.4.4 Introduction to some Beamformers

Beamforming is one of the simplest and most robust means of spatial filtering [17], this technique is widely used in discriminating between signals based on physical locations of the signal sources. Beamformers like Delay-and-Sum Beamformer, Superdirective Beamformers will be introduced in this section.

2.4.4.1 Delay-and-Sum Beamformer

The Delay-and-Sum Beamformer(DSB) delays the output of each elements to co-phase the sound arriving from a given. This technique has proved very effective for hands-free sound capture in moderately reverberant auditoria rooms [18]. And the weighting function of the DSB is

$$\boldsymbol{W} = \frac{1}{N}\boldsymbol{d} \tag{2.35}$$

Where $\boldsymbol{d}$ is the direction vector in (2.5).

2.4.4.2 Superdirective Beamformer

Superdirective Beamformer(SDB) aims to minimize the power of the output signal of the array, given that the gain at desired direction is 1, i.e.

$$\boldsymbol{W}^H\boldsymbol{d} = 1 \tag{2.36}$$

The signal from all direction except for the look direction is considered as noise, so this method could be interpreted as noise minimization process. Then the problem can be formalized as:

$$\min_{\boldsymbol{W}} \boldsymbol{W}^H\boldsymbol{\Phi}_{\boldsymbol{XX}}\boldsymbol{W} \quad subject\ to\ \boldsymbol{W}^H\boldsymbol{d} = 1. \tag{2.37}$$

Using the Lagrange-multiplier or gradient computation, the solution could be obtained:

$$\boldsymbol{W} = \frac{\boldsymbol{\Phi}_{\boldsymbol{VV}}{}^{-1}\boldsymbol{d}}{\boldsymbol{d}^H\boldsymbol{\Phi}_{\boldsymbol{VV}}{}^{-1}\boldsymbol{d}} \tag{2.38}$$

where $\boldsymbol{d}$ is the direction vector in (2.5), $\boldsymbol{\Phi}_{\boldsymbol{VV}}$ is the normalized cross power spectral density matrix of the noise.
Figure 8 shows performance comparison between DSB and SDB, it could be noticed that the DSB shows no directivity at low frequency range, while the SDB demonstrates relatively constant directivity throughout whole frequency range of interest.

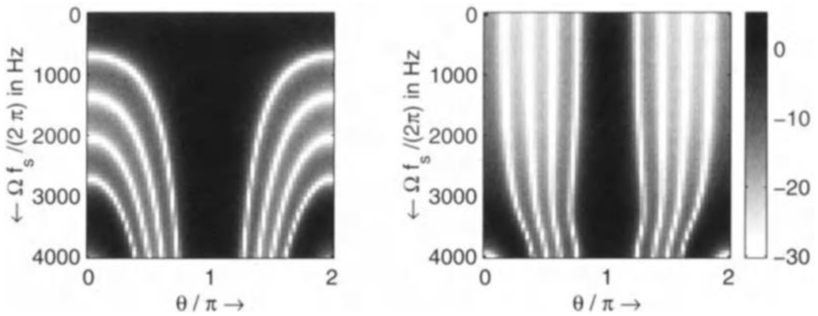

Figure 8 DSB vs. SDB

3. Implementation Plan

Microphone Array Beamforming			DURATION (days)
START DATE	END DATE	DESCRIPTION	
2019/3/24	2019/5/4	Literature Review	40
2019/5/1	2019/5/10	Linear Array Simulation	9
2019/5/11	2019/5/15	Spatial Aliasing Analysis	4
2019/5/16	2019/5/23	Circular Array Simulation	7
2019/5/24	2019/6/3	MVDR Beamformer Simulation	9
2019/6/3	2019/6/15	Ceiling-mounted Circular Array Simulation	12
2019/6/16	2019/6/20	Saved For Finishing Report	4
2019/4/27	2019/5/10	Interim Report	13
2019/5/11	2019/6/20	Final Report	39
2019/6/21	2019/6/27	Presentation	6

Table 3 Project Planned Schedule

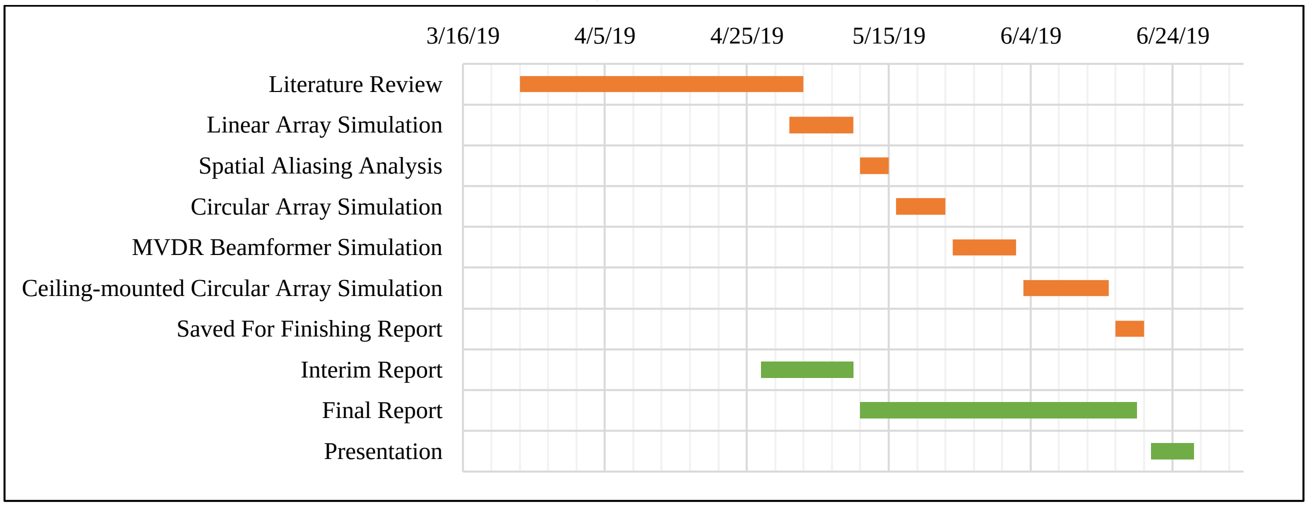

Figure 9 Gantt Chart

The table and Gantt chart shown above(Table 3 and Figure 9) list out an initial plan for the tasks. After background study, several practical tasks are selected. First, Linear Array will be investigated by simulation in MATLAB, its characteristics such as directivity when sources and receivers are in the same plane and the case where it is not will be inspected. The same approach will be done for Circular Array after an brief analysis of spatial aliasing. After the analysis of the array geometry, beamformers like Minimum Variance Distortionless Response Beamformer will be reproduced. Variation could also be taken here since variety of beamformers are available and worth-studying. Then ceiling-mounted circular/spherical array will be simulated and its corresponding beamforming algorithms will be studied. Finally, investigation on how to discriminate spatially separated sources such as different people talking in a room will begins. Note that a gap is left between completion of final task and submission of the final report(2019/6/20), the gap can also be used for project extensions and provides flexibility in occasions where tasks take longer to complete.

4. Evaluation Plan

There are several measures to evaluate the performance of different beamformers. Parameters like Array-Gain, Directivity Index, Front-to-Back Ratio and White Noise Gain [19] are used to measure different performance aspects. The details of these parameters is mentioned in 2.4.2.
Except for parameters stated above, graphs could be plotted to better visualize the performance of different arrays. A directivity illustration could be plotted, in either polar or Cartesian plane. This is a plot of the pattern in which the output level of the microphone or array changes when the sound source changes position in an anechoic space [20]. Even further, two dimensional plots can visualized the changes in directivity when frequency varies, an example is shown in Figure 10. Figure 10 shows that at low frequency, there this specific array is omnidirectional. As frequency increases, a better directivity is observed.

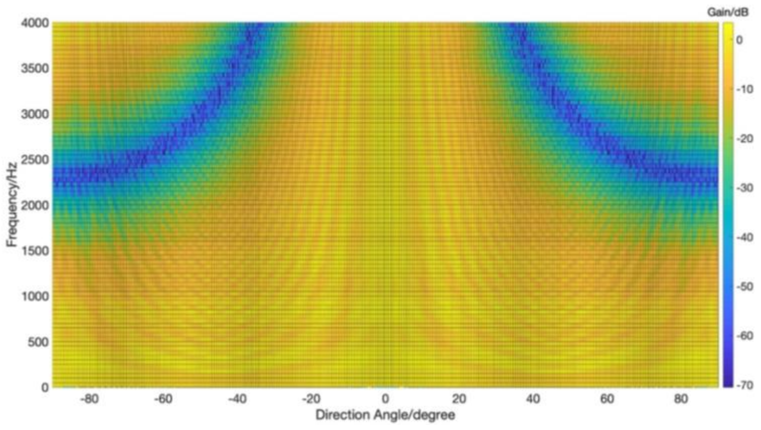

Figure 10 DSB 2 Mic 7.5cm Spacing Beampattern

5. Ethical, Legal and Safety Plan

Safety, ethical and legal issues have been examined for this project, and it has been concluded that these are not major concerns for this project. Since the project is based on computer simulation and is implemented on numerical processing software such as MATLAB, there are minimal safety concerns. Similarly, ethical and legal issues are hardly applicable while this project mainly focuses on theoretical model and algorithm comparison. One point to notice for legal issue is the use of the script Room Impulse Response Generator. According to the tutorial attached, the program is a free software it can be redistributed and modified under the terms of the GNU General Public License as published by the Free Software Foundation [10].

6. First Experiment

In this section, delay and sum beamforming(DSB) is simulated in MATLAB. Firstly, an equi-spaced microphone array is set up, showing in Figure 11. The microphone array consisting 8 omni-directional microphone elements is placed in the center of a room with dimension $10m \times 10m \times 20m$. The space between each microphone is set to be 0.075m, so the array has a total length of $0.075\text{m} * 7 = 0.525\text{m}$. The far field distance between the source and the center of the array is chosen to be $5m$. The directivity is calculated with sources(in blue dots) spaced at $0.5°$ interval, for 80 frequencies distributed up to 6kHz.

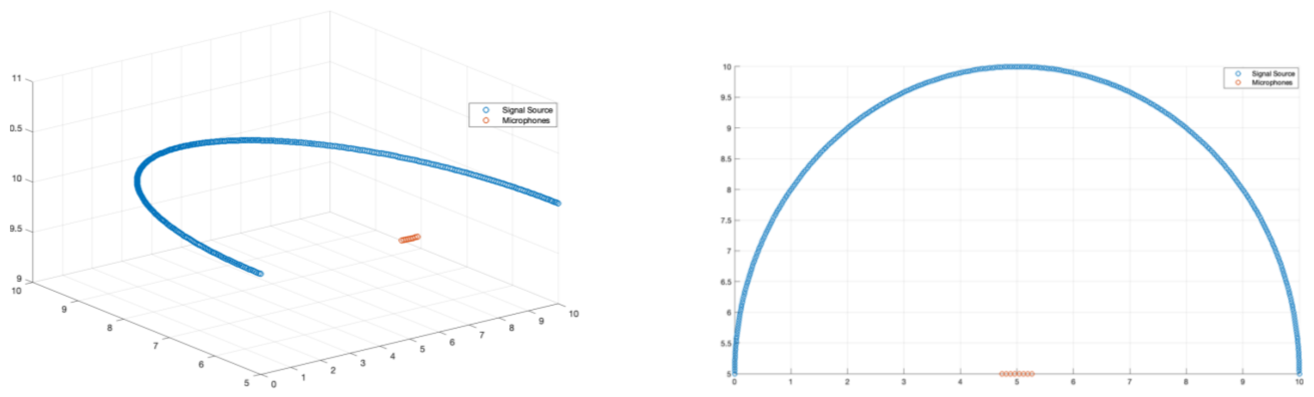

Figure 11 Source and Microphone Array Arrangement

Firstly impulse responses are calculated using RIR Generator function. In the RIR function, sampling frequency f_s is set to be $16000Hz$, number of samples generated is set to be $n = 1024$, reverberation time $\beta = 0.33s$, the speed of sound $c = 340m/s$. The test signal and microphone array's coordinates are listed in Table 4.

Element	Coordinates(m)
Microphone 1	(4.7375, 5, 10)
Microphone 2	(4.8125, 5, 10)
Microphone 3	(4.8875, 5, 10)
Microphone 4	(4.9625, 5, 10)
Microphone 5	(5.0375, 5, 10)
Microphone 6	(5.1125, 5, 10)
Microphone 7	(5.1875, 5, 10)
Microphone 8	(5.2625, 5, 10)
Signal	(10, 5, 10)

Table 4 Element Position

The impulse responses of eight microphones are shown in Figure 12. It could be noticed the delay between each element and a gradual decrease in amplitude when distance between microphone and source becomes larger.

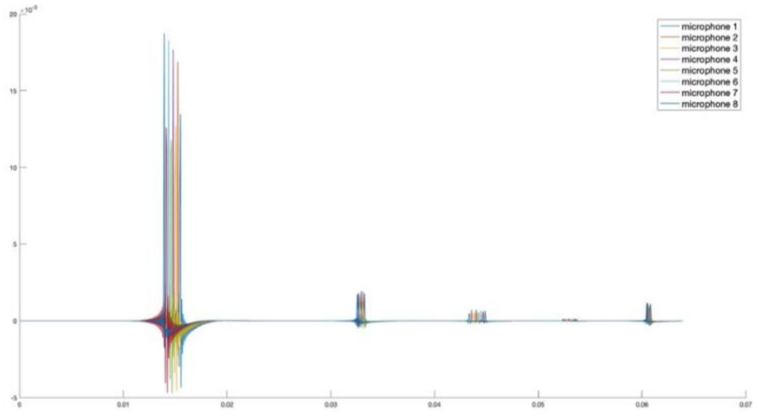

Figure 12 Impulse Responses of Microphones

Then $filter$ function is used with input impulse response, 1 and source signal. In Figure 13 shown below, source signal is set to be $\sin(2\pi \times 200t)$. It can be noticed that there is a large delay between source signal and microphone outputs and small delay between each microphone elements, also the amplitude of the output decreases as microphone is further to the source.

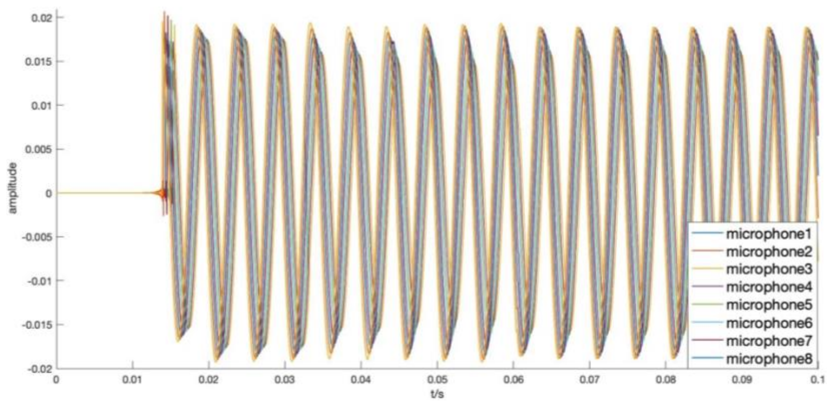

Figure 13 Outputs of microphones

The next step is to calculate the direction vector $\boldsymbol{d}$ in (2.5). A standard far field model can be estimated for linear arrays with equidistant sensors [11]:

$$\boldsymbol{d}^T = [1, \exp(-i\Omega f_s c^{-1} l \cos(\theta_0)), \exp(-i\Omega f_s c^{-1} 2l \cos(\theta_0)), \dots, \exp(-i\Omega f_s c^{-1}(N-1) l \cos(\theta_0))] \tag{6.1}$$

Where $f_s = 16000Hz$ is the sampling frequency, $c = 340m/s$ is the speed of sound, l is the inter-microphone spacing and θ_0 is the coming angle of the source signal shown in Figure 5.

After calculating the direction vector, The weighting function of Delay-and-Sum Beamformer can be calculated using (2.35). Then output is obtained using equation (2.6). Gain in dB is then calculated using $G = 10\log(|output|^2/|input|^2)$. The approach stated above is then calculated for sources(in blue dots) spaced at $0.5°$ interval from $-90°$ to $+90°$ and for 80 source signal frequencies evenly distributed from 0Hz up to 6kHz. Obtaining all the gain data from different angles with different source frequencies, two dimensional plot stated in Evaluation Plan can be plotted, shown in Figure 14.

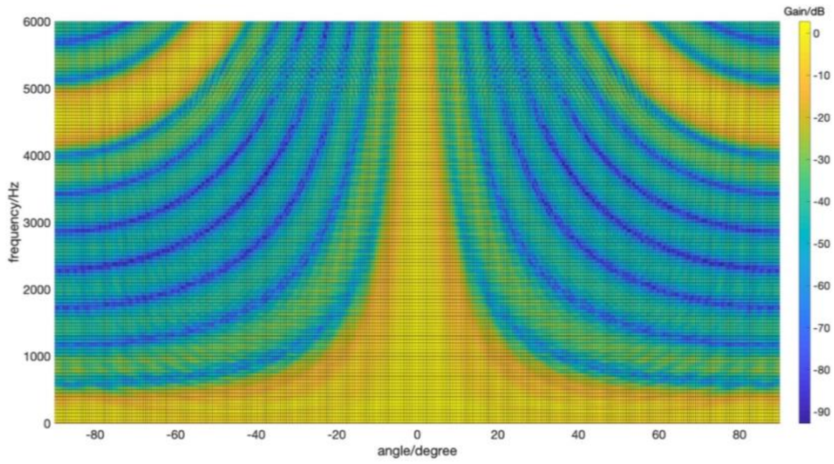

Figure 14 Simulated Beampattern of DSB

It can be clearly seen that the gain at look direction is much higher than other direction in most frequency range. But at low frequency range(0-500Hz), the array acts as a omnidirectional microphone.

More experiments are done on changing the spacing and quantity of the array's elements. Figure 15 illustrates how changing the quantity of the microphones affects the beampattern. And Figure 16 shows how varying the spacing would influence the beampattern.

It could be noticed that if we increase the quantity of microphones with the same spacing, the main lobe become narrower and thus a better directivity. The same conclusion applies to the side lobes, they also become narrower as number of elements increases.

If we vary the spacing and keep number of element unchanged instead, the number of side lobes increases. This could be explained using (2.26). Actually elements output are still in phase for $n\lambda$

$$\theta = \pm \sin^{-1}(n\lambda/L) \tag{6.2}$$

So for example the right bottom plot in Figure 16 has eight microphones with each spacing 20cm, the main lobe and side lobes' angles when source frequency is $4000Hz$ is

$$\theta = \pm\sin^{-1}(n \times 340/(4000 \times 0.2)) \tag{6.3}$$

$$when\, n = 0, \theta = 0°$$

$$when\, n = 1, \theta = \pm 25.15°$$

$$when\, n = 2, \theta = \pm 58.21°$$

To conclude, more elements, longer array, narrower the main lobe. Larger spacing between elements, more side lobes. In practical environment, spacing and quantity of microphones are two crucial parameters to consider if delay and sum beamformer is used.

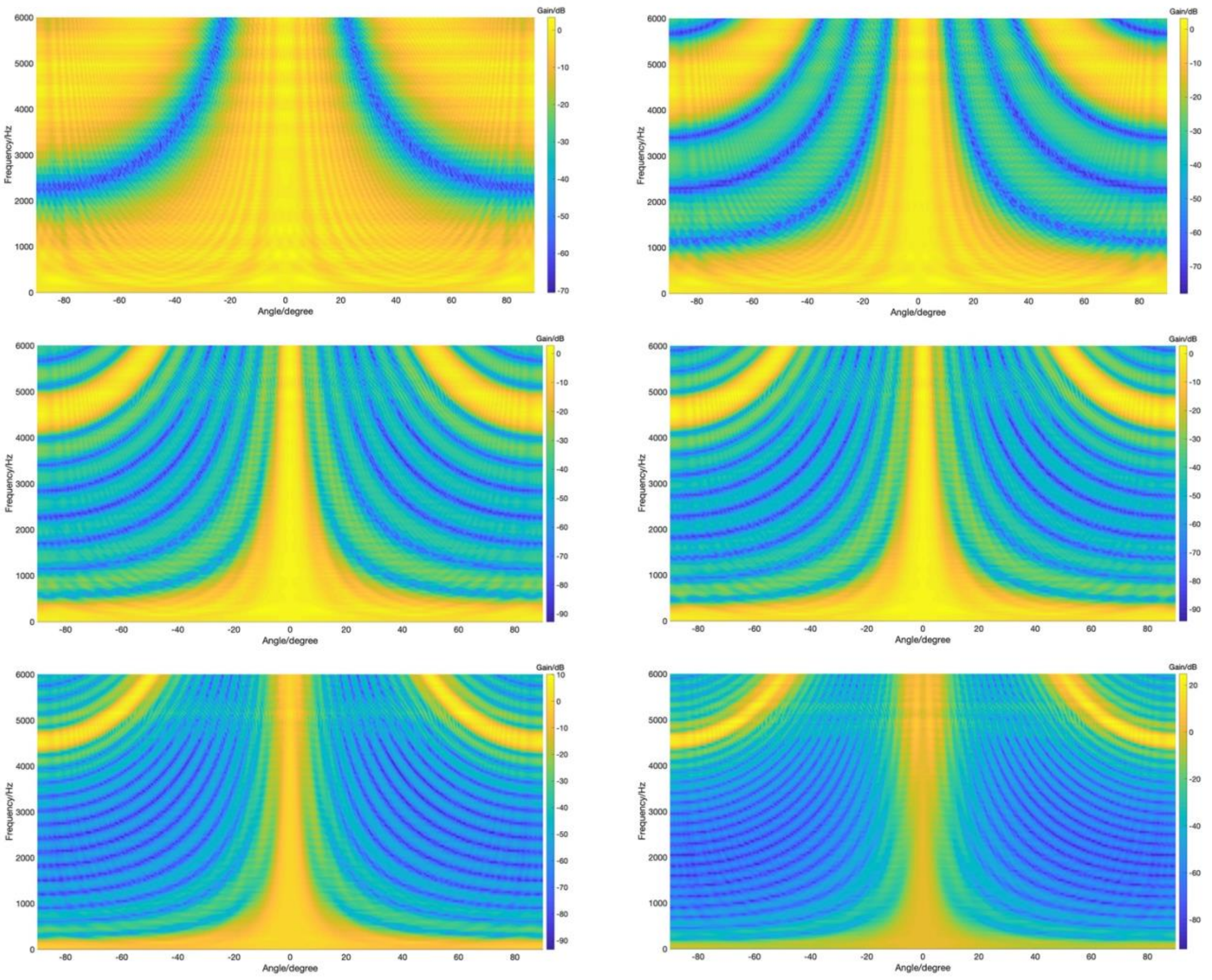

Figure 15 Beampattern varies with number of microphones(Spacing:7.5cm Number of microphones, from top to bottom, left to right: 2, 4, 8, 10, 15, 20)

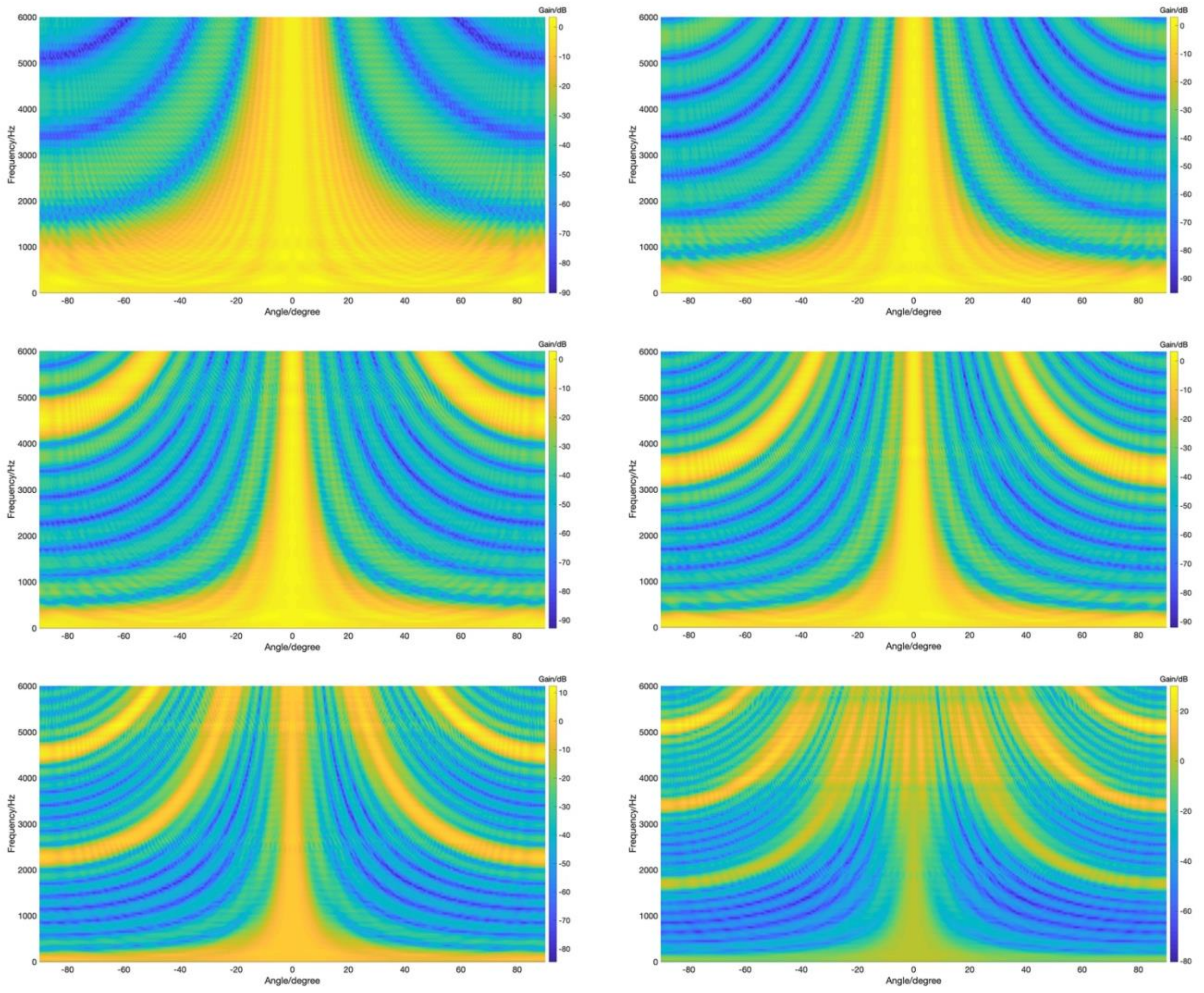

Figure 16 Beampattern varies with microphone spacing(Number of microphones: 8 Spacing, from top to bottom, left to right: 2.5cm, 5cm, 7.5cm,10cm, 15cm, 20cm)

References

[1] M. Masato and K. Y, "Inverse filtering of room acoustics," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 36, no. 2, pp. 145-152, 1988.

[2] S. Doclo and M. Moonen, "GSVD-based optimal filtering for single and multimicrophone speech enhancement," *IEEE Transactions on Signal Processing,* vol. 50, no. 9, pp. 2230-2244, 2002.

[3] E.-E. Jan and J. Flanagan, "Microphone Arrays for Speeclh Processing," *Proceedings of ISSE'95 - International Symposium on Signals, Systems and Electronics,* 1995.

[4] "Microphone array - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Microphone_array. [Accessed 2 May 2019].

[5] J. Koetsier, "Smart Speaker Users Growing 48% Annually, To Hit 90M In USA This Year," Forbes, [Online]. Available: https://www.forbes.com/sites/johnkoetsier/2018/05/29/smart-speaker-users-growing-48-annually-will-outnumber-wearable-tech-users-this-year/#6f88bddb5dde. [Accessed 2 May 2019].

[6] "HomePod - Technical Specifications - Apple," Apple, [Online]. Available: https://www.apple.com/homepod/specs/. [Accessed 9 May 2019].

[7] "Amazon Echo - Alexa-enabled Bluetooth speaker," Amazon, [Online]. Available: https://www.amazon.co.uk/d/Amazon-Echo-Devices/Amazon-Echo-2nd-Generation-Charcoal-Fabric/B06Y5ZW72J. [Accessed 9 May 2019].

[8] "Technical Specs of Google Home – Google Store," Google, [Online]. Available: https://store.google.com/gb/product/google_home_specs. [Accessed 9 May 2019].

[9] E. Habets, "AudioLabs - RIR Generator," Audio Labs, [Online]. Available: https://www.audiolabs-erlangen.de/fau/professor/habets/software/rir-generator. [Accessed 2 May 2019].

[10] E.̈ Habets, "Room Impulse Response Generator," 20 September 2010.

[11] D. B. Ward and R. A. Kennedy, "Constant Directivity Beamforming," in *Digital Signal Processing*, Berlin, Springer, 2001, pp. 12-37.

[12] J. Dmochowski, J. Benesty and S. Affes, "On Spatial Aliasing in Microphone Arrays," *IEEE Transactions on Signal Processing,* vol. 57, no. 4, pp. 1383-1395, 2009.

[13] "Plane wave - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Plane_wave. [Accessed 5 May 2019].

[14] C. F. Cardoso, "Signal Processing for Circular Microphone Arrays," 2007.

[15] S. Elliott and P. A. Nelson, "Active noise control," *IEEE Signal Processing Magazine,* pp. 12-35, 1993.

[16] "Bessel Function of the First Kind," Wolfram MathWorld, [Online]. Available: http://mathworld.wolfram.com/BesselFunctionoftheFirstKind.html. [Accessed 6 May 2019].

[17] J. Bitzer and K. U. Simmer, "Superdirective Microphone Arrays," in *Digital Signal Processing*, Berlin, Springer, 2001.

[18] J. L. Flanagan and E.-E. Jan, "Microphone arrays for speech processing," in *Proceedings of ISSE'95 - International Symposium on Signals, Systems and Electronics*, San Francisco, 1995.

[19] M. Brandstein and D. Ward, Microphone Arrays: Signal Processing Techniques and Applications, Berlin: Springer, 2001, pp. 21-24.

[20] InvenSense, "Microphone Array Beamforming Application Note AN-1140," 31 12 2013. [Online]. Available:

https://www.invensense.com/wp-content/uploads/2015/02/Microphone-Array-Beamforming.pdf. [Accessed 8 May 2019].

Appendix

Code implemented in MATLAB

```
%%%%%%%%%%%%%%%%%%%% RIR Generator Setup %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
c = 340;                    % Sound velocity (m/s)
fs = 16000;                 % Sample frequency (samples/s)
s = [1 0 1.5];              % Source position [x y z] (m)
L = [10 10 20];                % Room dimensions [x y z] (m)
beta = 0.33;                 % Reverberation time (s)
n = 1024;                   % Number of samples
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
t = 0:1/fs:0.1; % 0.1 sec of signal input
%%%% Create numbers of microphones in the centre of the room with total
%%%% length and equally spaced
numberOfMic = 2;
micSpacing = 0.075;
r = createMicrophonePositionArray(numberOfMic,micSpacing*(numberOfMic-1),L(3)/2,L(1),L(2));
degreeNeeded = 180; % angle needed to plot
resolution = 0.5; % how dense the source will be placed
source_position_array = calculateSourcePositionArray(degreeNeeded,L(3)/2,5,resolution); % calculate
source position
%%%%%% scatter the spatial arrangement if needed %%%%%
% scatter3(source_position_array(:,1),source_position_array(:,2),source_position_array(:,3));
% hold on;
% scatter3(r(:,1),r(:,2),r(:,3));
% hold off;
% legend('Signal Source','Microphones');
% hold off;
set(0, 'DefaultLineLineWidth', 1); %%%set default line width
%%% angle array
angleIndex = -degreeNeeded/2:resolution:degreeNeeded/2;
%%% frequency of source signal
source_frequency_array = 0:50:6000;
%%% output gain array
patternToPolarPlot_matrix = [];
%%%% iterate source frequency and angle to plot 2D graph
for i = 1:length(source_frequency_array)
    source_input = cos(2*pi*source_frequency_array(i)*t);
    powerArrayAtDifferentAngle = calculatePowerAtDifferentAngle(source_position_array,source_input, c,
fs, r, L, beta, n);
    sourceInputPower = sum(abs(source_input).^2)/length(source_input);
    patternToPolarPlot = abs(powerArrayAtDifferentAngle)/sourceInputPower;
    normalizationFactor = (length(patternToPolarPlot)-1)/2+1;
    patternToPolarPlot = patternToPolarPlot/patternToPolarPlot(normalizationFactor);%%% force zero
degree direction gain = 1
    patternToPolarPlot = 10*log(patternToPolarPlot);
    patternToPolarPlot_matrix = [patternToPolarPlot_matrix;patternToPolarPlot];
end
%%% plot 2D graph
surf(angleIndex,source_frequency_array,patternToPolarPlot_matrix);

%%%%%%%%%%%%%% function : calculate all 7 impulse responses %%%%%%%%%%%%%%%
function h_array = calculateImpulseResponses(c, fs, r, s, L, beta, n)
h_array = [];
microphoneNumbers = size(r,1);
for microphpneIndex = 1:microphoneNumbers
    h = rir_generator(c, fs, r(microphpneIndex,:), s, L, beta, n);
    h_array = [h_array; h];
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%% this function calculate the power of input signal %%%%%
function powerOfSignal = calculatePower(inputSignal)
if ~isvector(inputSignal)
    error('Input must be a vector')
end
powerOfSignal = sum(inputSignal.^2)/length(inputSignal);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%% function to calculate all outputs using filter() function %%%%%%%%%%%
function output_array = calculateAllMicrophonesOutput(inputSource,impulseResponses)
output_array = [];
microphoneNumbers = size(impulseResponses,1);
for microphpneIndex = 1:microphoneNumbers
    output = filter(impulseResponses(microphpneIndex,:),1,inputSource);
    output_array = [output_array;output];
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%% create source position for needed degree %%%%%%%%%%%%%%
function source_position_array = calculateSourcePositionArray(degreeNeeded,z,r,resolution)
source_position_array = [];
for source_angle =
0*degreeNeeded/360:2*pi*(degreeNeeded/360)/degreeNeeded*resolution:2*pi*degreeNeeded/360
   source_position = [5+r*cos(source_angle) 5+r*sin(source_angle) z];
   source_position_array = [source_position_array;source_position];
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%% create linear array microphone array position %%%%%%%%%%%%%
function microphone_position_array =
createMicrophonePositionArray(numberOfMicrophones,arrayLength,z,roomX,roomY)
microphone_position_array = [];
spacing = arrayLength/(numberOfMicrophones-1);
for microphoneIndex = 0:numberOfMicrophones-1
   microphonePosition = [(roomX-arrayLength)/2+microphoneIndex*spacing roomY/2 z];
   microphone_position_array = [microphone_position_array;microphonePosition];
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%% calculate final output power array %%%%%%%%%%%%%%%%%%%%
function powerArrayAtDifferentAngle =
calculatePowerAtDifferentAngle(source_position_array,source_input, c, fs, r, L, beta, n )
powerArrayAtDifferentAngle = [];
sourceNumbers = size(source_position_array,1);



for sourceIndex = 1:sourceNumbers
   impulseResponsesArray = calculateImpulseResponses(c, fs, r, source_position_array(sourceIndex,:),
L, beta, n);%%% get impulse responses at this position of s and r
   output_array = calculateAllMicrophonesOutput(source_input, impulseResponsesArray);
   timeDomainLength = length(output_array(1,:));
   frequencyDomainLength = fs*(0:(timeDomainLength/2))/timeDomainLength;
   frequencyDomainLength = [fliplr(-frequencyDomainLength),frequencyDomainLength(2:end)];
   final_output_fft_array = calculateOutputFFT(output_array); %%% transform output into frequency
domain
   spacing = abs(r(1,1) - r(2,1));
   source_position = source_position_array(sourceIndex,:);
   coming_angle = atan((source_position(2)-5)/(source_position(1)-5));
```

```
    directionArray = calculateDirectionVector(r, spacing, fs, coming_angle, c, frequencyDomainLength);
    system_output = zeros(1,length(directionArray(1,:)));
    for microphoneNumber = 1:size(directionArray,1) %%%% delay and sum implementation
        tmp_array = directionArray(microphoneNumber,:) .* final_output_fft_array(microphoneNumber,:);
        system_output = system_output + tmp_array;
    end
    system_output = system_output/size(directionArray,1); %%%divided by N
    timeDomainOutput = ifft(system_output);
    final_output_power = sum(abs(timeDomainOutput).^2)/length(timeDomainOutput); %%% calculate final
output power
    powerArrayAtDifferentAngle = [powerArrayAtDifferentAngle,final_output_power];
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%% calculate output array fft %%%%%%%%%%%%%%%%%%%%%%%%%%%
function final_output_fft = calculateOutputFFT(outputArrayInTimeDomain)
final_output_fft = [];
outputArrayNumber = size(outputArrayInTimeDomain,1);
for outputIndex = 1:outputArrayNumber
    final_output_fft = [final_output_fft; fft(outputArrayInTimeDomain(outputIndex,:))];
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%% calculate d in frequency domain%%%%%%%%%%%%%%
%%% far field in the book
%%% take first microphone position in the array as the reference position
function directionArray = calculateDirectionVector(microphonePositionArray, spacing, fs,
coming_angle, c, frequencyDomainLength)
directionArray = [];
mircrophoneNumber = size(microphonePositionArray,1);
frequencyNumber = length(frequencyDomainLength);
for microphoneIndex = 1:mircrophoneNumber
    singleDirectionVector = []; %%% reset singleDirectionVector
    for frequencyIndex = 1:frequencyNumber %% for different frequcny component in one microphone
output
        singleDirectionVector = [singleDirectionVector,exp(-
1*1i*frequencyDomainLength(frequencyIndex)*2*pi*fs/c*spacing*cos(coming_angle))];
    end
    directionArray = [directionArray;singleDirectionVector]; %%% append microphones direction vector
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```