HW2: SPATIAL SPEECH SEPARATION

Peyman Javaheri Neyestanak

Iman Javaheri Neyestanak

Kasra Alizadeh

THEORETICAL TOOLS AND METHOD

1.1. Introduction

In the field of spatial speech separation, the task is to design algorithms that can effectively separate sound signals captured by a microphone array. This report focuses on a specific scenario with a linear array of two microphones, denoted as *N=2*, with a uniform spacing *d between* them. The sources of signals are positioned in space, and the goal is to develop an algorithm that can successfully separate these signals through appropriate processing.

1.2. Signal from Rotating Source and ToD/DToD

In the given scenario, we assume $N=2$ to represent the minimum array configuration. The microphones are positioned at $(-d/2,0)$ and $(d/2,0)$, with inter-microphone spacing $d = 18\ cm$. The source of the signal is rotating in a circle with a radius $r0 = 3m$ centered at the origin $(0,0)$ with an angular speed $\omega 0 = \pi\ rad/s$. The propagation of the signal is affected by a delay dependent on the distance between the source and the microphone, denoted as *r*, and an attenuation factor given by $\alpha(r) = \left(\frac{r_0}{r}\right)^2$.

1.3. Signal-to-Noise Ratio (SNR)

The Signal-to-Noise Ratio (SNR) is a fundamental concept in signal processing that quantifies the ratio of the power of a signal of interest to the power of unwanted noise in the same bandwidth. SNR is commonly used as a measure of the quality of a signal, indicating how well the signal can be distinguished from background noise. It is expressed in decibels (dB) and is a crucial metric in various fields, including telecommunications, audio processing, image processing, and wireless communications.

The formula for SNR in decibels is given by:

$$SNR(dB) = 10 \cdot \log_{10}(SignalPower/NoisePower$$

Where:

1- Signal Power is the power of the signal of interest.
2- Noise Power is the power of the background noise.

In a general sense, a higher SNR indicates a better-quality signal, as the signal is more distinguishable from the noise. Conversely, a lower SNR implies that the signal is more susceptible to interference and may be challenging to detect accurately.

1.4. Mean Squared Error (MSE)

In advanced digital signal processing, Mean Square Error (MSE) is a measure used to quantify the average squared difference between the estimated or processed signal and the true or desired signal. MSE is commonly employed as a performance metric to assess the accuracy and fidelity of signal processing algorithms or systems. It is especially relevant in scenarios where the goal is to minimize the discrepancy between the processed signal and the original signal.

Mathematically, the MSE for a discrete-time signal *xn* and its estimate or reconstruction *x′[n* is defined as:

$$MSE = 1/N \sum_{n=0}^{N-1} \left(x(n) - x'(n)\right)^2$$

Where:

1- *N* Is the total number of samples in the signal.
2- *X(n)* represents the true signal at the time index *n*
3- *X′(n)* is the estimated or processed signal at the time index *n*

The MSE formula computes the average squared difference between corresponding samples of the true signal and its estimate. Squaring the differences ensures that both positive and negative errors contribute positively to the overall error, preventing cancellation of errors. The average is taken over all samples to obtain a normalized measure that is independent of the signal length.

1.5. Cross-Correlation and Time of Arrival (ToA)

Cross-correlation is a measure of similarity between two signals as a function of the time lag applied to one of them. In this report, cross-correlation will be used to estimate the relative time delay τ *between* the signals received by the two microphones. ToA refers to the time at which a signal arrives at a specific location. In this context, it is used to align signals received by the microphones properly. The subsequent sections of this report will delve into the specific steps and results obtained through the proposed algorithms and analyses.

Solutions

2.1. Signal from rotating source and ToD/DToD

First, we set the sampling frequency (fs), duration of the signal (duration), radius of the circle (r0), microphone spacing (d), speed of sound (c), and angular frequency of the source signal (ω) Then we load the audio file and cut it briefly. Also, we create the time vector between the 0 and (duration - 1/fs) and set the microphone positions.

Since the source is rotating, we need to know its position to calculate delays. The X and Y coordinates of source position (x_s, y_s) as a function of time (t) is:

$$(x\_s, y\_s) = (r0 \cdot \cos(\omega \cdot t)\ , r0 \cdot \sin(\omega \cdot t))$$

Moreover, the distances between the source and the microphones (r1, r2) at each time step is:

$$r = \sqrt{(x_s - x_{mic})^2 + (y_s - y_{mic})^2}$$

Now we can compute the delay time for each microphone:

Delay1 = r1 / c

Delay 2 = r2 / c

Furthermore, propagation is affected by a delay that depends on the distance source-microphone r and the attenuation is:

$$\alpha(r) = \left(\frac{r_0}{r}\right)^2$$

And we calculate the delayed source signal for each microphone:

```
s1 = zeros(size(source_signal));

s2 = zeros(size(source_signal));

for i = 1: minLength

s1(i) = source_signal(max (1, i - round(delay1(i)*fs)));

s2(i) = source_signal(max (1, i - round(delay2(i)*fs)));

End
```

Then, we calculate the attenuation factors (att_factor1, att_factor2) for each microphone based on the source and microphone distances and then apply attenuation factors to the delayed signals.

After adding random noises (by randn() function in Matlab) to the attenuated signals, we can play r1 and r2 and their plot is shown in following picture.

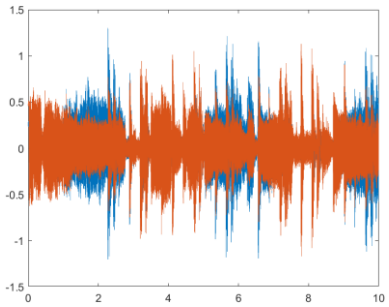

Figure 1: signals received in microphone 1 and 2

2.1.1. Section A

Difference Time of Arrival is computed by the following formula:

$$DToA = \frac{r2}{c} - \frac{r1}{c}$$

For plotting MSE versus SNR, first we define various levels of noise (from 10^{-4} to 10^{-1}) and then we multiply these amounts to the signals to create noisy signals.

The SNR is computed by dividing the power of signal by the power of noise, and the power of signal or noise is the average of the squared signal amplitude. So, we calculate the SNR for diverse levels of noise. After which, we can compute the amount of MSE for each microphone for all the noisy signals and plot MSE versus SNR.

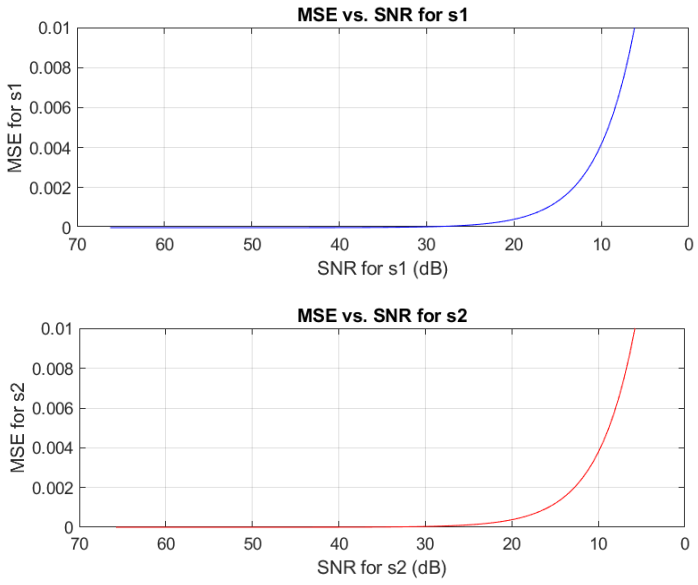

Figure 2: MSE vs. SNR

These plots show the detrimental effect of noise on the MSE calculation, as MSE is increasing with lower SNRs.

2.1.2. Section B

In this part, we want to remove the effect of rotation by realigning the signal. First, we average the DToA for all time steps. Then we calculate the time offsets (α1, α2) for individual signals to align them with the average DToA.

So, we assign $\alpha 1 = -\frac{average\ of\ DToD}{2}$ and $\alpha 2 = +\frac{average\ of\ DToD}{2}$.

Moreover, we apply inverse attention factors to equalize the amplitudes of signals.

After that, we create time vectors (t_ α 1, t_ α 2) accounting for the time offsets and interpolate the signals using the shifted time vectors to align them in time and we plot them:

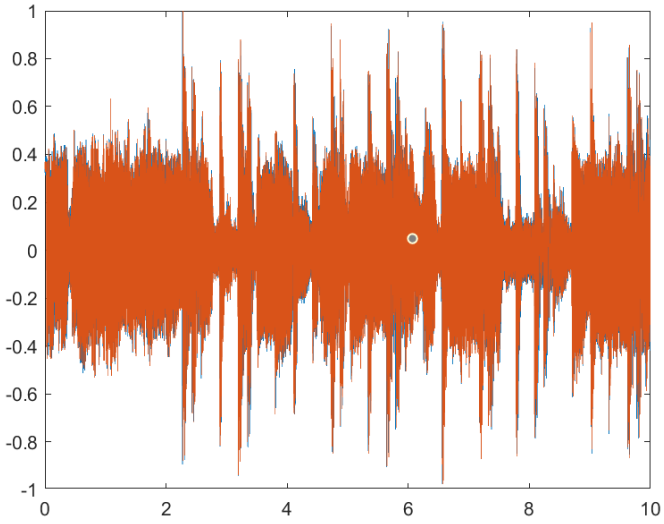

Figure 3: signals received in microphone 1 and 2 after removing the effects of rotation

2.1.3. Section C

In this section we assume R(t) = 0.5 * R1(t) + 0.5 * R2(t).

To compute MSE and SNR, we use the method explained above. Then we apply cross-correlation between noisy signal and the source signal to estimate time of arrival.

First, we identify the maximum peak in the cross-correlation. This peak corresponds to the time when the source signal and the noisy signal are most aligned, which is the estimated ToA.

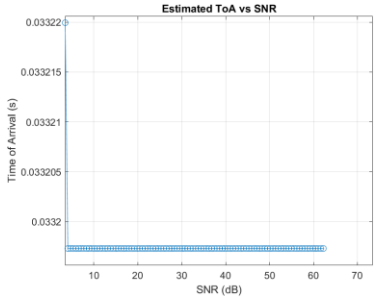

Figure 5: Estimated ToA vs. SNR

In this case, the ToA is constant as expected. Since the combined signal Rh is constructed by aligning R1h and R2h based on their respective times of arrival. So, it shows that alignment is perfect.

2.2. signals from static source with array

In this section, it is assumed that there are 15 microphones, located along the x-axis with distance defined by d, as well as a rotating source just like the previous section. For simulating the position of microphones, we used linspace function in MATLAB to divide the x-axis equally.

Following which, we calculated delay based on the position of each microphone and we implemented it to the received signal, before averaging the combined signal using for loop. As it is shown below.

```
for l = 1: N
    alphal = x_positions(l) / c;
    Rlh(l, :) = simulate_microphone_signal(t, alphal, source_signal);
    Rh = Rh + Rlh(l, :);
End
```

For doing so we have implemented below function, which introduce the time shift to the simulated_signal:

```
function simulated_signal = simulate_microphone_signal(signal, delay, fs)
    sample_shift = round (delay * fs);
    simulated_signal((sample_shift+1): end) = signal(1:(end-sample_shift));
end
```

Next, we applied different levels of noise to the variable named Rh which was followed by estimation of Time of Delay, all, in order to plot ToA versus SNR.

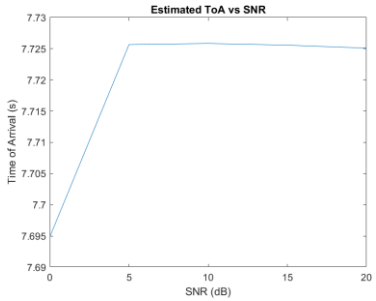

Figure 6: Estimated ToA vs SNR

According to the figure, at higher SNRs we have almost constant as well as more accurate estimations than lower ones, which means that we have clear noise that makes it more convenient to estimate ToA.

In the next step, the values of MSE are calculated versus various SNRs and then it is plotted.

Calculation of MSE:

mse_values_Rlh(i, l) = mean (noisy_signal - clean_signal). ^2);

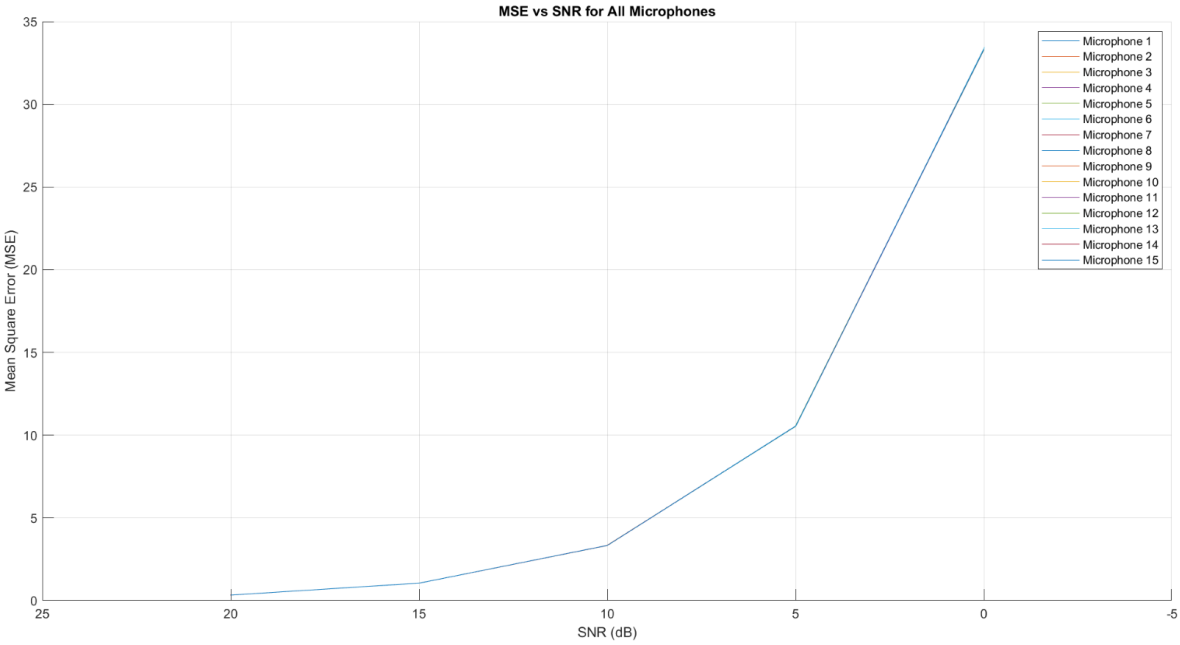

Figure 7: MSE vs SNR for all microphones

It is worth noting that the MSE for all the microphones are the same despite having statistical independent noise, while it should not be so. There might be some error in the MATLAB code that we hope we can fix it. However, when it comes to analyzing the general trend, it can be seen that as SNR increases the MSE converges to zero which signifies a better estimation as a result of diminishing of the noise influence on the signal.

Conclusion

During our project on audio signal processing, we have made some interesting discoveries about the core ideas and complex issues that we had, particularly when we trying to deal with sounds coming from a source that is moving around and a set of microphones.

This project showed us how sound moves across space and the challenges we face with moving sound sources. We found that how a spinning source can change, the timing of the sounds, Doppler effect and how they can get quieter as they spread out.

Another issue that we are dealing with is Background noise. By adding some background white noise at different levels, we found out how it affects SNR as well as how the MSE grows, as the noise increases. Having said that, the graphs related to MSE for each signal are not completely accurate, as we have statistical independent noise, but all the plots were exactly the same. This taught us quite a bit about how difficult audio signal processing has to be to cut through noise.

By using not only one microphone, but also two microphones, we saw how bringing different signals together can enhance our ability to figure out where a sound is coming from, and from our point of view this was the most interesting part, as we could relate it to human's ears and real sounds around us.

The simulation taught us step by step the process of capturing sounds, sorting out time delays and weak signals, and dealing with noise, to see how these affect the quality of the sound.

All in all, this simulation was an especially useful tool, linking theoretical knowledge with practical application. It taught us about the detailed aspects of locating sound sources, the effects of noise, and taught us essential approaches for improving and analyzing signals.

The techniques we have examined in these simulations are quite important and useful, not only for this scenario, but also for many areas in real life.

Talking about audio signal processing, the things we have covered can make music sound better by reducing unwanted noise, changing the quality of the signal, and even finding the direction of the sound that it comes from. In this point of view, such analysis is very widespread in music and film producers to introduce a high-quality product.

When it comes to telecommunications, understanding how noise affects the quality of signals is essential. Needless to say, it is about making sure the message we send through our devices is received clearly without any misunderstandings. In other words, by having the Time of Arrival of the transmitted data we can cut off other delayed signals.

Furthermore, from the security point of view, being able to locate where a sound is coming from using microphone or sensor arrays can be very vital for security professionals.

So, in general, these simulations give us a strong understanding of audio processing, opening a world of possibilities for applying these concepts in different industries.

Looking into the way we process signals, especially when we are dealing with sounds that come from moving sources and are picked up by an array of microphones, we see a spacious room for improvement. Here are some methods that we can consider getting closer to what happens in the real world.

Firstly, as mentioned before, sound spreads through many ways and channels, as it reflected, absorbed, transmitted, and internally reflected when it encounters an obstacle, for instance walls and trees in the city. However, in the simulation we just considered a direct path, so it could be more realistic if we considered other paths.

Furthermore, we could consider noise more realistic. To be more specific, instead of just having a standard background noise, it would better if we could have noise that changes depending on the channel. That would give us a better intuition for what you hear in daily life.

In addition, we should also use better ways to understand how sound gets weaker as it travels further. But we used a certain attenuation factor for doing so.

In overall, by considering these aspects of the real world, we could improve the version of our system when working with sounds that are real.