EPFL

Multichannel DOA

Student :
Antonio Morais

Supervisor :
Dalia El Badawy
Professor :
Adam Scholefield

June 12, 2020

Contents

1 Introduction

In the day-to-day life our head scatters how the sound arrives to our ears making it easier to deduce if the sound comes from our right because we would be able to hear better the sound in our right ear than in the left one and vice-versa. Simultaneously this scattering has also a different interest. Indeed, people who lost their hearing in one ear use this scattering to help them deduce where the sound they're hearing comes from, notably by using the power of the sound i.e. if the sound seems loud it probably comes from their hearing side and alternatively if it seems soft it probably comes from their non-hearing side.

This functionality to use its head as a way to deduce the direction of arrival of a sound is relative to what is called the head-related transfer function (HRTF) in humans and other animals. But the head scattering is, of course, not the only scattering possible to be able to deduce the direction of arrival with only one receiver. Indeed some recent work has shown that, while using a microphone surrounded by LEGO© bricks, it was possible to have an estimation of the direction of arrival [BD18]. However in this project we will try to have a generalization of this concept and use a scattering with several microphones and not just one to see if it helps.

The goal of this project is to compare various localization algorithms with and without the presence of scattering. The different impulse response we would use are :

- A LEGO response impulse : in this response we have 6 microphones at different positions. We can see the positions of the LEGOs and the ones of the microphones in figure 1 (2 of the microphones are not really visible because they are under the yellow truck).
- A KEMAR response impulse : a device simulating a human head and torso visible at figure 2. In this response we have 2 microphones that simulates the ears of a human.
- An omnidirectional response impulse : an analytically computed response to simulate an omnidirectional microphone receiver that has nothing around him.

Figure 1: LEGO© and microphones disposition

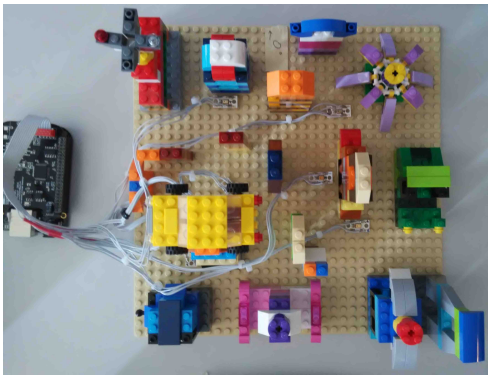


Figure 2: KEMAR head and torso simulator

2 Stacking

In this section we will talk about the white noise localization algorithm described in the paper "Direction of Arrival with One Microphone, a few LEGOs, and Non-Negative Matrix Factorization" [BD18]. As stated in the title of the paper, this algorithm is described how to be used for one microphone in the paper so in order to be able to use it in this project we have to adapt it. The adaptation is done by stacking the different impulse responses of the different mics and use it as if it was a unique microphone. It is important to notice that this algorithm is done assuming an anechoic case (i.e. no echos, only the direct sound is taken in account).

2.1 Description of the impulse response

First of all, we would like to address how the impulse responses that we have look like. For the LEGO case we have 6 different microphones each having different impulse responses. These responses have a discretization of 180 different angles, each angle being 2 degrees apart from each other. And for each angle we get an impulse response in time domain for 160 units of time. For the KEMAR case we have 2 different microphones and their responses have a discretization of 360 different angles but for the sake of this project we will only select 180 different angles and again each angle will be 2 degrees apart so that the comparison with the LEGO case is fair. For the last case, the omnidirectional one, the responses that we will have are analytically computed assuming a far-field model and using the following function for omnidirectional sound $H(w) = e^{-jw\tau}$ with $\tau = \frac{d}{c}$, d is the distance between the source and the microphone and c is the speed of sound ($c = 343 \left[\frac{m}{s}\right]$). As we are assuming far field the distance between the source and the microphone is computed using the angle so the source would be at an hypothetical position $\binom{cos(\theta)}{sin(\theta)}$ from the microphone if we want the sound coming from the θ angle. In the same way as for the other responses we will use 180 angles each 2 degrees apart for fairness.

2.2 Description of the algorithm

2.2.1 Idea of the algorithm

As explained in section 2.1, we will use 180 different angles. Meaning that the azimuth is discretized into 180 candidate source locations. In this description we will use D different angles to illustrate the algorithm, while keeping in mind that we have 180 angles in our case, so the locations could be $\Omega = \{\theta_1, \theta_2, \theta_3 ... \theta_{\mathrm{D}}\}$. We consider now the standard mixing model in time domain. Having L sources coming from the direction $\Theta = \{\theta_{\mathrm{j}}\}_{j \in J}$, the standard mixing model is

$$y(t) = \sum_{j \in J} s_{\mathrm{j}}(t) * h_{\mathrm{j}}(t) + e(t) \tag{1}$$

where the parameters are; J the different sources, each source j $\in \{1, 2, 3 ... D\}$, |J| = L, * is the convolution, s_j is the signal sent by the j^{th} source, $h_j(t)$ is the impulse response corresponding to the angle θ_j, y is the observed signal at the microphone corresponding to the impulse response h and e(t) is additive noise [BD18][FS16]. So, using the observed signal y, we want to find where the sources are, meaning we want to find the set Θ. We can also compute the short-time Fourier transform to approximate the equation 1 in frequency domain.

$$Y(n, f) = \sum_{j \in J} S_j(n, f) H_j(f) + E(n, f), \tag{2}$$

where f and n denotes the frequency and time indices [BD18][FS16].

As stated in the introduction of this section, this algorithm is done by stacking the different impulse responses and observed signals so to be able to consider these stackings as a unique microphone and signal. For monaural localization if the source is always the same (for example a white noise source) then the localization is simple because each direction gives a different spectral signature, if the impulse responses uses scattering, and we can detect it by correlation. Of course, in reality, the source is never fixed but this idea helps for the algorithm and the approximation that we do.

So if the sources are white noise and we have the different transfer function $\{H_d\}_{d=1}^{D}$ (i.e. the impulse responses) we can use the power spectral density (PSD) to approximate the direction(s) of arrival of the source(s) [BD18]. For a white source the PSD is flat and scaled by its power meaning that

$$\mathbb{E}[|S_j|^2] = \sigma_j^2. \tag{3}$$

And, when assuming that the noise has a zero mean, the PSD of the observed signal is

$$\mathbb{E}[|Y|^2] = \sum_{j \in J} \sigma_j^2 |H_j|^2. \tag{4}$$

So $\mathbb{E}[|Y|^2]$ belongs to a cone defined as

$$C_J = \{x : x = \sum_{j \in J} c_j |H_j|^2, c_j > 0\}, \tag{5}$$

each cone C_J corresponds to a different combination of the sources meaning that there is $\binom{D}{L}$ different cones as we have D directions and L sources [BD18]. Knowing all that, the localization only becomes a simple approximation problem where we have to identify the closest cone

$$\hat{J} = argmin_J dist(\hat{\mathbb{E}}[|Y|^2], C_J), \tag{6}$$

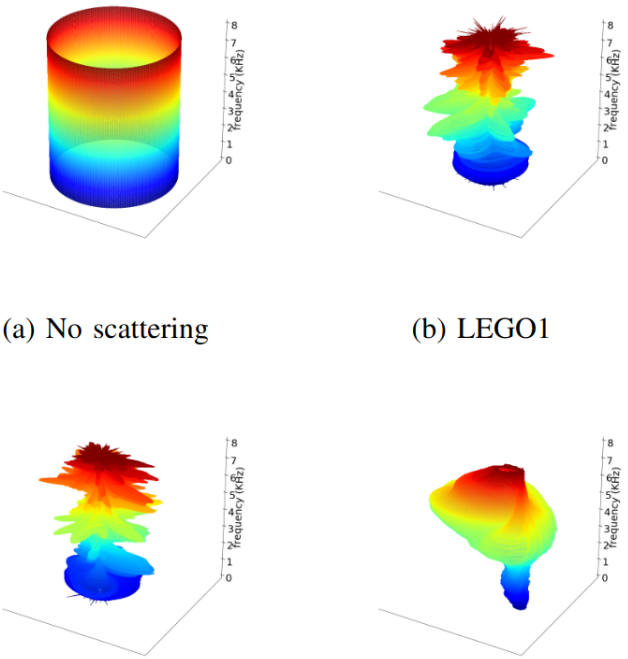
where $\hat{\mathbb{E}}[|Y|^2]$ corresponds to the expectation $\mathbb{E}[|Y|^2]$ computed empirically using the observed measurements. This estimation of the angles works when the different cones are distinct meaning that if $C_{J_1} = C_{J_2}$ then $J_1 = J_2$ [BD18]. And this will more likely hold when the different H_j are diverse, this is the reason why scattering helps with monaural localization.

In figure 3 we can see different scatterings and the directional frequency magnitude response that is obtained with the respective scattering. Figure 3(a) illustrates the usual omnidirectional case and this image shows that localization without scattering is impossible when using only one microphone (i.e. monaural localization) as it has no proper asperities. Meaning that the condition $C_{J_1} = C_{J_2}$ implies $J_1 = J_2$ doesn't hold at all. Figures 3(b) and 3(c) are magnitudes produced from LEGO scatterings that are comparable to the one that we have. Finally figure 3(d) is produced by a KEMAR head and is exactly corresponding to what we have for our KEMAR impulse responses meaning that it corresponds to a usual HRTF. It is easy to notice that the KEMAR scattering is smoother than the LEGO ones, probably because the head produces a weaker scattering than a lot of different LEGOs. Meaning that the LEGO scatterings will probably be the better ones here.

2.2.2 Algorithm

We can see the pseudo-code in the algorithm 1. In the algorithm we compute the empirical PSD of y which is a sufficiently good approximation [BD18][Led01]. Then, instead of comparing to the cones we compare to its smallest enclosing subspace $S_J = span\{|H_j|^2\}_{j \in J}$ represented by the following matrix

$$B_J = [|H_{j_1}|^2, |H_{j_2}|^2 ... |H_{j_L}|^2], j_k \in J. \tag{7}$$



(a) No scattering (b) LEGO1

(c) LEGO2 (d) KEMAR

Figure 3: Directional frequency magnitude response for different devices. Each horizontal slice corresponds to a frequency between 0-8000 Hz from bottom to top. These magnitudes doesn't correspond to our scatterings but can be used as illustrations for information. They correspond to the impulse responses of the paper [BD18] from which the image comes from. These scatterings are comparable to ours.

Meaning that selecting the closest cone can now be approximately determined by selecting the subspace projection of $J \subseteq D$ that has the smallest error. Of course this algorithm is only robust if there is enough difference in these projections meaning that if the angles are too close or if the scattering gives transfer functions that only vary smoothly across directions then the algorithm will not give as good results as it could because the subspaces would be too similar [BD18].

Algorithm 1: White noise localization using scattering and stacking

Input: Number of sources L, magnitudes of transfer functions $\{|H_j|^2\}_{j\in D}$, N audio frames $Y \in \mathbb{C}^{FxN}$

Output: Directions of arrival $\hat{\Theta} = \{\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3 ... \hat{\theta}_L\}$

Stack the transfer functions of the different microphones by appending them (HStack)

Stack the observed signal of the different microphones by appending them (YStack)

Compute the empirical PSD y = $\frac{1}{N}\sum_{n=1}^{N}|Y_n|^2$

for *every* $J \subseteq D$, $|J| = L$ **do**

  $B_J \leftarrow [|H_j|^2]_{j\in J}$

  $P_J \leftarrow B_J B_J^{\dagger}$

end

$\hat{J} \leftarrow argmin_J||(I - P_J)y||$

$\hat{\Theta} \leftarrow \{\theta_j | j \in \hat{J}\}$

3 Music

In this section we will talk about the Multiple Signal Classification algorithm, mainly known as music. This algorithm is already designed to detect different sources by using several microphones so we don't need to adapt it. The only small adaptation that we will do is that music works for one frequency only so we will adapt it to use several frequencies. What we're interested in learning here is if having a scattering around our microphones actually helps to detect more precisely the direction of arrival of the different sources signal. That is why, as before, we will compare the results of the impulse responses from the LEGO and KEMAR scatterings with the omnidirectional one. Also music has a constraint that is important to note, indeed the number of sources has to be strictly smaller than the number of microphones used to detect them.

3.1 Description of the impulse response

The different impulse responses that we will use are the same that we used in the stacking problem, that is why we will not describe them again here as a good enough description is done in section 2.1.

3.2 Description of the algorithm

3.2.1 Setup of the algorithm

Let's assume S sources at different locations that we will consider in the polar coordinates $\boldsymbol{r}_s^S = (r_s, \theta_s)$ with s = 1,...,S, we have the same for the N different microphones $\boldsymbol{r}_n^M$ with n = 1,...,N. Note that in our algorithm we assume a far-field situation meaning that the exact position of the sources is not important and only the angle is used. Furthermore the microphones positions will only be used in the omnidirectional case in order to compute the different transfer functions necessary there. So the observed signal in frequency domain is the following

$$M_n(k) = \sum_{s=1}^{S} V_n(\boldsymbol{r}_s^S, k) S_s(k) + B_n(k), \tag{8}$$

where $M_n(k)$ is the observed signal at the n microphone, $V_n(\boldsymbol{r}_s^S, k)$ is the n^{th} entry of the steering vector $\boldsymbol{V}(\boldsymbol{r}, k) = (V_1(\boldsymbol{r}, k), ..., V_N(\boldsymbol{r}, k))^T$ that, in our implementation, is equivalent to the transfer functions, $S_s(k)$ is the signal emitted by the source s, $B_n(k)$ is some added noise and k is the frequency [Syl15][Sch86]. If we define vectors for the observed signal, source signal and the noise as follows $\boldsymbol{M}(k) = (M_1(k), M_2(k)..., M_N(k))^T$, $\boldsymbol{S}(k) = (S_1(k), S_2(k)..., S_S(k))^T$ and $\boldsymbol{B}(k) = (B_1(k), B_2(k)..., B_N(k))^T$ and a matrix for the steering vector $\mathrm{V}(\boldsymbol{r}_1^S, ..., \boldsymbol{r}_S^S, k) = (\boldsymbol{V}(\boldsymbol{r}_1^S, k)|...|\boldsymbol{V}(\boldsymbol{r}_S^S, k))$ we can redefine equation 8 as a matrix multiplication [Syl15]

$$\boldsymbol{M}(k) = V(\boldsymbol{r}_1^S, ..., \boldsymbol{r}_S^S, k)\boldsymbol{S}(k) + \boldsymbol{B}(k), \tag{9}$$

3.2.2 Algorithm

The sources are assumed independent, zero-mean stationary and of a frequency k_0. Let's assume $\mathcal{I}$ and $\mathcal{O}$ to be the identity and the zero matrices then

$$\Gamma_B = \mathbb{E}[\boldsymbol{B}\boldsymbol{B}^H] = \sigma_N^2 \mathcal{I}_{N\times N} \text{ and } \mathbb{E}[\boldsymbol{S}\boldsymbol{B}^H] = \mathcal{O}_{S\times N} \tag{10}$$

Music uses the eigendecomposition of the covariance to determine the number of sources and the direction of arrival of each source. The covariance, also called interspectral, matrix $\Gamma_M = \mathbb{E}[\boldsymbol{M}\boldsymbol{M}^H]$ is

$$\Gamma_M = (\mathcal{U}_S|\mathcal{U}_N)\begin{pmatrix} \lambda_1+\sigma_N^2 & & \mathcal{O} & | & \\ & \ddots & & | & \mathcal{O} \\ \mathcal{O} & & \lambda_S+\sigma_N^2 & | & \\ -- & -- & -- & -- & -- \\ & \mathcal{O} & & | & \sigma_N^2\mathcal{I}_{N-S} \end{pmatrix}(\mathcal{U}_S|\mathcal{U}_N)^H \tag{11}$$

the λ are in descending order $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_S > 0$. $\mathcal{U}_S$ are the S first eigenvectors and are related to the S greatest eigenvalues ($\lambda_1+\sigma_N^2$ to $\lambda_S+\sigma_N^2$), $\mathcal{U}_N$ are the remaining N-S eigenvectors (the eigenvalues equal to σ_N^2) and are related to the noise space. With that we can deduce the number of sources (N minus the number of repetitions of σ_N^2) and their locations because their associated steering vector are orthogonal to $\mathcal{U}_N$.

In practice, we don't know Γ_M so we compute an approximation on W time snapshots

$$\hat{\Gamma}_M = \frac{1}{W}\sum_{w=0}^{W-1}\hat{\boldsymbol{M}}_w(k)\hat{\boldsymbol{M}}_w^H(k) \tag{12}$$

$\hat{\boldsymbol{M}}_w(k)$ is a Discrete Fourier Transform approximation of $\boldsymbol{M}(k)$. And we can find the directions of arrivals of the sound sources by isolating the maximum values of the pseudo-spectrum

$$h(r,\theta) = \frac{1}{\boldsymbol{V}^H(r,\theta)\hat{\Pi}_{\mathcal{N}}\boldsymbol{V}(r,\theta)} \tag{13}$$

All this is just for one frequency so in our algorithm, that we can see in algorithm 2, we compute that for each frequency that we have. We add all the pseudo-spectrum together and isolate the maximum value in the sum to be more precise.

Algorithm 2: Music Localization with scattering

Input: Transfer functions $\{H_j\}_{j\in D}$
Output: Directions of arrival $\hat{\Theta} = \{\hat{\theta}_1, \hat{\theta}_2, ...\hat{\theta}_L\}$
$Totalspectrum \leftarrow \mathbf{0}$
for *every frequency* **do**
  Compute the empirical covariance matrix $\hat{\Gamma}_M = \frac{1}{W}\sum_{w=0}^{W-1}\hat{\boldsymbol{M}}_w(k)\hat{\boldsymbol{M}}_w^H(k)$
  Compute the eigenvectors and the eigenvalues of the covariance matrix
  Compute the spectrum using $h(r,\theta) = \frac{1}{\boldsymbol{V}^H(r,\theta)\hat{\Pi}_{\mathcal{N}}\boldsymbol{V}(r,\theta)}$
  $Totalspectrum \leftarrow Totalspectrum + spectrum$
end
$\hat{J} \leftarrow argmax(TotalSpectrum)$
$\hat{\Theta} \leftarrow \{\theta_j | j \in \hat{J}\}$

The algorithm 2 is the pseudo code for the music algorithm that we wrote and as we can see it follows all the steps described before.

4 Results

In this section we will talk about the different results we got. Each table represent a different number of microphones and number of sources pair. In each of them there's a different number of frequencies used. Each line in a table is a compilation of the results of 50 runs done in the evoked algorithm. The values used to illustrate the error are the error average, max error and min error. The error average is the mean of the difference of the true angle and the estimate angle for the 50 runs. The max error is the maximal difference done in the 50 runs and the min error is the minimal one. The error varies from 0 to 180 as angles are periodical meaning that a 300° angle only has a difference of 60 with the 0° angle. So if a max error has the value 180 it means that the worst error happened at least once and if the min error has the value 0 it means that we were at least correct once. The best result of each table is in **bold**.

Using	N° Freqeuncies	Error Average	Max Error (Min Error)
Stacking Lego	65	0.1	2 (0)
Stacking Lego	129	0.04	4 (0)
Stacking Lego	257	0.1	6 (0)
Stacking Lego	**513**	**0**	**0 (0)**
Stacking Omnidirectional	65	84.18	172 (16)
Stacking Omnidirectional	129	93.78	178 (24)
Stacking Omnidirectional	257	93.72	176 (0)
Stacking Omnidirectional	513	105.2	176 (0)
Music Lego	**65**	**0**	**0 (0)**
Music Lego	129	2.76	84 (0)
Music Lego	**257**	**0**	**0 (0)**
Music Lego	**513**	**0**	**0 (0)**
Music Omnidirectional	**65**	**0**	**0 (0)**
Music Omnidirectional	129	1.8	180 (0)
Music Omnidirectional	257	1.8	180 (0)
Music Omnidirectional	**513**	**0**	**0 (0)**

Table 1: Summary of the algorithms using 6 Microphones and for one source, we're using the Lego responses and omnidirectional one (analytically computed). The different algorithms used are the stacking one and music. Every line is a compilation of 50 runs each done with a noise of 20 decibel.

In table 1 we have the results of the algorithms using 6 microphones and 1 source. We can see that we have several perfect results notably in the music algorithm that seems to work really well when there is several microphones. The 2 algorithms seem to be pretty good except when using the stacking algorithm with omnidirectional impulse responses which was expected.

In table 2 we have the results of the algorithms using 6 microphones and 2 sources. We can see that it becomes harder and the algorithms don't find as often the good angles yet the music algorithm managed to do a perfect run and is, on average, better for this configuration.

Using	N° Freqeuncies	EA source 1	EA source 2	Max Error (Min Error)
Stacking Lego	65	12.7	5.26	69 (0)
Stacking Lego	129	0.22	1.88	83 (0)
Stacking Lego*	257	0.0	0.25	5 (0)
Stacking Omnidirectional	65	82.28	100.48	158 (18)
Music Lego	**65**	**0**	**0**	**0 (0)**
Music Lego	129	0.12	0.5	25 (0)
Music Lego	257	0.46	4.24	42 (0)
Music Lego	513	0	3.5	42 (0)
Music Omnidirectional	65	0	3.22	35 (0)
Music Omnidirectional	129	2.64	1.68	90 (0)
Music Omnidirectional	257	0	3.34	13 (0)
Music Omnidirectional	513	0.02	5.32	88 (0)

Table 2: Summary of the algorithms using 6 Microphones and for 2 sources, we're using the Lego responses and omnidirectional one (analytically computed). The different algorithms used are the stacking one and music. Every line is a compilation of 50 runs each done with a noise of 20 decibel (* is an exception and has only 20 runs because the running time was too long, EA stands for error average)

Using	N° Freqeuncies	Error Average	Max Error (Min Error)
Stacking Lego	65	1.26	64 (0)
Stacking Lego	129	0.34	8 (0)
Stacking Lego	257	0.16	4 (0)
Stacking Lego	513	0.06	2 (0)
Stacking Lego	**1025**	**0**	**0 (0)**
Stacking Kemar	65	19.22	170 (0)
Stacking Kemar	129	20.28	171 (0)
Stacking Kemar	257	10.64	174 (0)
Stacking Kemar	513	2.48	13 (0)
Stacking Kemar	1025	2	7 (0)
Music Lego	65	3.86	116 (0)
Music Lego	129	1.3	82 (0)
Music Lego	257	2.92	82 (0)
Music Lego	513	3.9	150 (0)
Music Kemar	513	1.9	92.0 (0)
Music Omnidirectional	65	2.86	128 (0)
Music Omnidirectional	129	3.28	164 (0)
Music Omnidirectional	257	1.8	166 (0)
Music Omnidirectional	513	0.76	74 (0)

Table 3: Summary of the algorithms using 2 Microphones and searching for one source, we're using the Lego responses, kemar responses and omnidirectional one (analytically computed). The different algorithms used are the stacking one and music. Every line is a compilation of 50 runs each done with a noise of 20 decibel.

In table 3 we have the results for 2 microphones and 1 source. In the music algorithm we only used 180 degrees and not 360 because if the microphones are on a line (always the case for 2 microphones) the music algorithm is not able to precisely deduce if the sound is coming from the front or the back; a seemingly symmetry problem. So we decided to use only half of the data to be fairer. However we still can see that stacking seems to work better for 2 microphones. It's not really surprising as the more

microphones there is the better music works. There is no line for stacking omnidirectional because it would give results similar to a random one as in the other tables. The stacking with LEGO managed to score a perfect run which shows that the scattering done with LEGOs is really helpful and better than the one with KEMAR.

Using	N° Freqeuncies	EA source 1	EA source 2	Max Error (Min Error)
Stacking Lego	65	16	20.32	82 (0)
Stacking Lego	129	6.42	14.16	84 (0)
Stacking Lego	257	5.42	4.6	31 (0)
Stacking Lego	**513**	**0.36**	**0.22**	**10 (0)**
Stacking Kemar	65	31.32	45.48	154 (3)
Stacking Kemar	129	31.76	35.28	103 (1)
Stacking Kemar	257	32.52	25.52	132.5 (1)

Table 4: Summary of the algorithms using 2 Microphones and 2 sources, we're using the Lego responses, kemar responses and omnidirectional one (analytically computed). The algorithm used is the stacking one because the music one has the boundary [number of sources $<$ number of microphones]. Every line is a compilation of 50 runs each done with a noise of 20 decibel. EA stands for error average

In table 4 we can see the results for 2 microphones and for 2 sources. In this table the music algorithm is not present because music only works when there is less sources than microphones available. The stacking with LEGO is still doing pretty good but the stacking with KEMAR is pretty bad when searching for 2 sources. This result shows again that the LEGO scattering seems pretty good and helpful.

5 Pyroomacoustics

In this section we will talk about pyroomacoustics. Pyroomacoustics is a python library developed by EPFL that simulates sound in a room generating the proper echos and decays of sound using omnidirectional sound.

Figure 4: A room with a source at position (1,1) and a circle of microphones centered at (2,2)

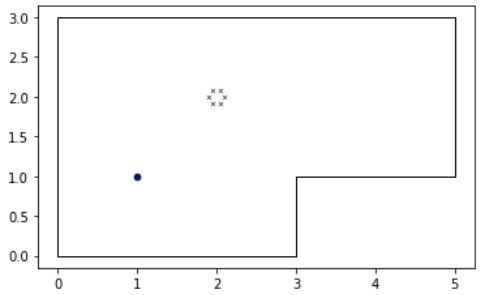
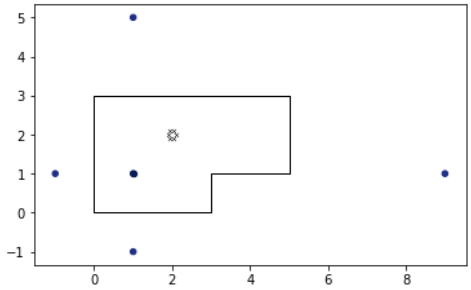
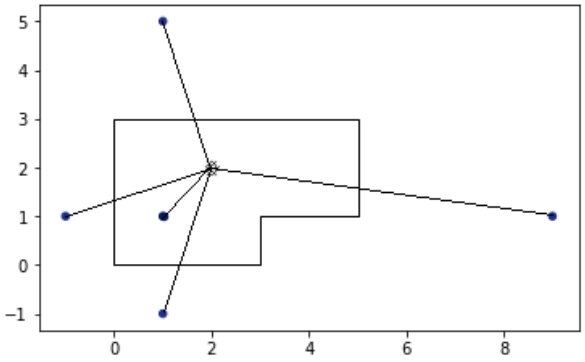


Figure 5: The same room but with images of the source to simulate echo



What pyroomacoustics do to simulate echo is creating images of the sources out of the room as we can see in figure 5. These images are created by symmetry with the walls of the room. In order to simulate echo, these images will send the same signal as the source but with a delay and a decay computed with the dirac function and a fractional delay filter [Väl95]. These signals act as an echo. In the figure 5 we have a first order echo but we could have a second order or even more meaning that it could simulate an echo of an echo etc. We can see the path of these signals sent to the microphones in the figure 6

Figure 6: The room with the path of the signals



What we wanted to do with pyroomacoustics was to use it and modify the code in order to adapt it to be able to use different impulse responses than the omnidirectional one. So we would be able to use our LEGO and KEMAR responses and see if scattering could help in the direction of arrival for a room with echo. So the microphone class was modified to keep in memory the different impulse responses. And we also modified how the room impulse response was computed in order to take in account the given impulse responses. Another thing we had to think about was which angles to choose when using the impulse responses. Because, as we can see in figure 6, the signals come from different directions and the angles from where it comes from probably are not integer values and, even if they are, they might not correspond to an available angle in the impulse responses. Indeed, our impulse response is a discrete response and doesn't include all integer angles also. Meaning that one of the problems we were facing

was to interpolate our values so what we decided to do was to take the closest available angle to the computed angle from the positions of the microphones.

Sadly the adaptation done was not working as, when comparing the anechoic case of pyroomacoustics with the music algorithm explained earlier, the results were not similar at all but they should have been. We tried different solutions, for example computing the angle for the group of microphones instead of each microphone because that's how the angle was computed when obtaining the impulse responses. But none of the tried solutions worked and there's still a problem. We think that it might come from the fact that we're using different frames of reference or it might come from an unfound bug in the code. Unfortunately we had no time left to work on that.

The tables 5 and 6 illustrates the bad results that we were talking about. And even in the anechoic case (order 0 or absorption 1) the results are not good as they should be because they should normally look like the music results that we got before. This fact shows that there is something done wrong either in the data or in the algorithm.

Order	absorption	N° Freqeuncies	Error Average	Median	Max Error (Min Error)
0	1	129	31.3	5.5	172 (0)
0	1	257	39.1	7	165 (0)
0	1	513	34.94	9.5	143 (0)
1	0.8	513	40.72	7	163 (0)
1	0.5	513	29.52	6.5	147 (0)
1	0.2	513	31.	7	169 (0)
2	0.5	513	27.28	6.5	164 (0)

Table 5: Summary of pyroomacoustics with a shoe box room with dimensions 10x10. Every line is a compilation of 50 runs each done with a noise of 20 decibel.

Order	absorption	N° Freqeuncies	Error Average	Median	Max Error (Min Error)
0	1	513	29.34	8	149 (0)
1	0.8	513	38.46	5.5	172 (0)
1	0.5	513	48.34	10.5	177 (0)
1	0.2	513	35.76	6.5	170 (0)
2	0.5	513	35.34	7.5	168 (0)

Table 6: Summary of pyroomacoustics with a shoe box room with dimensions 20x20. Every line is a compilation of 50 runs each done with a noise of 20 decibel.

6 Conclusion

In conclusion of this work we saw that the scattering indeed helps to find the direction of arrival when using multiple microphones. But we can also see that this help is not as necessary as for only one microphone. Of course for the stacking algorithm the scattering helps a lot but it's because it uses the scattering to find the direction of arrival so if there's no scattering, which is the case in the omnidirectional case, the results will be evidently bad. And if we look into the music algorithm that already takes in account the different microphones, we can see that the scattering doesn't change much in the one source case but actually helps when having more than one source as we can see in the 2 sources table. We also saw that both algorithms are pretty good but the music is probably more consistent in general. Yet the stacking algorithm works better when using only 2 microphones[1].

As a final word, I would like to thank personally Dalia El Badawy for all the help she provided as she contributed a lot to this work by explaining some concepts that were completely new to me.

[1] All the written code and the results data can be found in the following git: `https://github.com/anmorais/MultichannelDoA`

References

[Sch86] R. Schmidt. "Multiple emitter location and signal parameter estimation". In: *IEEE Transactions on Antennas and Propagation* 34.3 (1986), pp. 276–280. DOI: 10.1109/TAP.1986.1143830.

[Väl95] Vesa Välimäki. "Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters". In: Helsinki University of Technology, Faculty of Electrical Engineering, Laboratory of Acoustics and Audio Signal Processing, Dec. 1995. Chap. 3. ISBN: 951–22–2880–7.

[Led01] M. Ledoux. "The Concentration of Measure Phenomenon". In: *AMS Surveys and Monographs* 89 (Jan. 2001).

[Syl15] Philippe Souères Sylvain Argentieri Patrick Danès. "A Survey on Sound Source Localization in Robotics: from Binaural to Array Processing Methods". In: *Computer Speech and Language, Elsevier* 34 (2015), pp. 87–112.

[FS16] Christof Faller and Dirk Schröder. "Course notes book : Audio Signal Processing and Virtual Acoustics". In: École polytechnique fédérale de Lausanne, Sept. 2016.

[BD18] Dalia El Badawy and Ivan Dokmanic. "Direction of Arrival with One Microphone, a few LEGOs, and Non-Negative Matrix Factorization". In: *IEEE/ACM Transactions on Audio, Speech and Language Processing* (2018). DOI: 10.1109/TASLP.2018.2867081.