Friedrich-Alexander-Universität Erlangen-Nürnberg

Chair of Multimedia Communications and Signal Processing

Prof. Dr.-Ing. André Kaup

Bachelor Thesis

Evaluation of optimized microphone array configurations for robot audition

from Paulo André Martinez Oliveira

August 2015

Advisors: Prof. Dr.-Ing. Walter Kellermann
M.Sc. Hendrik Barfuss

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe, und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Ort, Datum	Unterschrift

Contents

Abstract

Humanoid robots have found many applications on services and entertainment. Their auditory system plays a crucial role on verbal communication with humans, by being able to understand the human communication partner even in reverberant and noisy environments. Is it already known that the microphone positions greatly influence its effectiveness. Therefore in this thesis, the relation between effective rank and microphone array performance is investigated. The effective rank is a measure used for quantifying the amount of information the array is capable to perceive. Optimum microphone positions are obtained by maximizing the effective rank with respect to a specific set of frequencies and source positions. Simulations in MATLAB show that arrays with higher effective rank possess better performance in terms of beampattern, beamformer response and white noise gain. Regarding signal-dependent metrics, the optimized array configurations have a better signal-to-noise ratio and automatic speech recognition score.

Acronyms

GHRTF	Generalized head-related transfer function
HRTF	Head-related transfer function
RIR	Room impulse response
SVD	Singular value decomposition
MVDR	Minimum variance distortionless response
SNR	Signal-to-noise ratio
ASR	Automatic speech recognition
DTFT	Discrete-time Fourier transform
STFT	Short-time Fourier transform
WGN	White noise gain
FIR	Finite impulse response
FSB	Filter-and-sum beamformer
RLSFI	Robust least-squares frequency-invariant
EARO	Embodied audition of robots object
MFCC	Mel-frequency cepstral coefficients
DOA	Direction of arrival
GSC	Generalized sidelobe canceler
PSD	Power spectral density
ADM	Adaptive directional microphone

Chapter 1

Introduction

Automated services are becoming more and more common in daily life. The next step in this evolution can be the usage of humanoid robots. Their application is being developed on a variety of fields, such as services and entertainment.

There are many aspects responsible for their human-like aspect and behaviour, like aritificial intelligence and mechanical ability. One of these aspectsis the verbal communication with humans. For instance, the largest bank in Japan will start employing the humanoid robot NAO for attending costumers in branches [1]. NAO is a 58cm tall humanoid robot which is developed by Aldebaran robotics, a subsidiary of Japanese telecommunications group Softbank [2]. The robot has to fully comprehend the desired listener, even if they are few meters apart and several other sources of interference or background noise are present. One also has to account for reverberation due to the acoustic environment. Especially environments like the lobby of a bank may produce a lot of reverberation due to the presence of hard surfaces like glass or a stone floor.

Hence, the robot's audition system is a very important component in order to establish a successful communication. Some of the challenges in this task includes localizing the sources, reducing the influence of the interfering ones and suppress noise as well as reverberation. Nonetheless many aspects need further developed, in order to improve its capability.

A solution for solving most of these issues is called beamforming. This is a spatial filtering technique where signals arriving from a certain direction are extracted while signals from other directions are attenuated. Because of that it is said that this algorithm forms a beam pointing to the desired direction. It is well known in array processing literature that increasing the number of microphones yields a better beamforming performance. Moreover, It is also strongly dependent on the positioning of the microphones [3]. An optimal array geometry is able to effectively acquire the signal from the environment, given a variety of source positions in a desired frequency range, which in this context is the one of the human voice. This leads to a higher speech recognition rates, which is the final objetive of the robot auditory system.

Regarding the microphone array, it was common to use binaural configurations due to its human-like appearance and simplicity. With them, it is already possible to perform source localization, tracking, and separation [4]. Nevertheless, robots with more than two microphones distributed on the robot's head are being used more often. For instance, the current version of NAO and Pepper from Aldebaran uses 4 microphones. The HRP-2 [5] and Hearbo [6] robots use 8-microphone arrays. Moreover, the upcoming new version of NAO and current ROMEO [7] robots, also from Aldebaran, will employ 12 and 16 microphones in its arrays.

In this thesis, the focus is on the microphone positions on the robot head. An optimum microphone array configuration is pursued, such that the signal enhancement performance of spatial filtering algorithms is improved. Few was done until now in the literature regarding a method obtaining an optimum microphone array configuration.

With respect to the optimality of microphone positions, there are a number of publications on linear arrays [8, 9, 10]. However they should not be applied in this context, since placing such a constraint for the array on the robot head will greatly reduce its performance.

A narrow-band method for positioning arbitrary arrays is proposed in [11], but it can be only used for waveform estimation when the look directions are already known. Also a wide-band optimization procedure for binaural configurations is described in [12], which is based on feature extraction, like interaural time and level differences (ITD/ILD).

In [13], a method for obtaining optimum microphone array configurations with arbitrary geometry and number of microphones is presented. An optimum array configuration is obtained by maximizing the effective rank of the steering matrix wich contains the head-related transfer functions (HRTFs) of the robot head for which the array design is carried out. The article shows that this optimum array geometry generally results in reduced variance for direction of arrival (DOA) estimation and increased beamforming robustness.

The main objective in this thesis is to further investigate the relation between effective rank and actual performance improvement in the context of robot audition.

In contrast to [13], the optimization will be carried out for the head of the humanoid robot NAO, instead of a dummy human head. In the assessment part, optimized array configurations are compared with themicrophone positions of the current and future NAO robot's version. Signal-independent and signal-dependent measures are used for comparison.

This thesis unfolds as follows: In Chapter 2 the measurement model which is employed in this thesis is discussed. The theoretical background of the algorithms which are used for the performance assessment is also presented. Chapter 3 deals with the optimization process for the microphone array positions. In Chapter 4 the performance assessment of the optimum array configurations are carried out. Chapter 5 specifies how the computational codes are used in this thesis. Finally, Chapter 6 includes the conclusion of the thesis and also an outlook to future work.

Chapter 2

Robot Audition Framework

2.1 Measurement Model

Let a sound field be generated by D spatially separated sources in the far-field with the presence of noise. Now consider a robot head with L microphones in the presence of this field. The complex sound pressure amplitudes measured by the microphones at frequency ω_i and time frame l can be written as

$$\mathbf{p}(\omega_i, l) = [p_1(\omega_1)\ p_2(\omega_1)\ \cdots\ p_L(\omega_K)]^\top, \tag{2.1}$$

where the $(\cdot)^T$ operator denotes matrix transposition. We consider that the measured amplitudes are a linear combination of the source signals plus additive noise. This can be described in matrix form by

$$\mathbf{p}(\omega_i, l) = \mathbf{A}(\omega_i)\mathbf{s}(\omega_i, l) + \mathbf{n}(\omega_i, l). \tag{2.2}$$

The D-point vector

$$\mathbf{s}(\omega_i, l) = [s_1(\omega_i, l)\ s_2(\omega_i, l)\ \cdots\ s_D(\omega_i, l)]^\top \tag{2.3}$$

are the amplitudes of the D sources. The $L \times D$ steering matrix $\mathbf{A}$ relates each source signal with each microphone signal. It is defined as

$$\mathbf{A}(\omega_i) = \begin{bmatrix} a_{1,1}(\omega_i) & a_{1,2}(\omega_i) & \cdots & a_{1,L}(\omega_i) \\ a_{2,1}(\omega_i) & a_{2,2}(\omega_i) & \cdots & a_{2,L}(\omega_i) \\ \vdots & \vdots & \ddots & \vdots \\ a_{D,1}(\omega_i) & a_{D,2}(\omega_i) & \cdots & a_{D,L}(\omega_i) \end{bmatrix}. \tag{2.4}$$

The second subscript index account for the microphones. Also note that the frame indexes is removed, because the response is considered to not change over time. It may also be defined as a concatenation of steering vectors:

$$\mathbf{A}(\omega_i) = [\mathbf{a}_1(\omega_1)\ \mathbf{a}_2(\omega_1)\ \cdots\ \mathbf{a}_D(\omega_1)]^\top, \tag{2.5}$$

where $\mathbf{a}_j(\omega_i)$ is the $L \times 1$ steering vector which relates the j-th source signal to the sound pressure measured by all microphones at frequency ω_i [13]. Finally, $\mathbf{n}$ is the L-point noise vector, which accounts for the sensor imperfections and present interference in the sound field:

$$\mathbf{n}(\omega_i, l) = [n(\omega_i, l)\ n(\omega_i, l)\ \cdots\ n(\omega_i, l)]^\top. \tag{2.6}$$

2.2 Wide-band Generalization

Now the model is extended to the wide-band scenario, By this it is possible to better characterize the recording environment. The sound pressure measured at the microphones at all K frequencies of interest is concatenated in one vector as follows:

$$\mathbf{p}(l) = \left[\mathbf{p}(\omega_1, l)^\top\ \mathbf{p}(\omega_2, l)^\top\ \cdots\ \mathbf{p}(\omega_K, l)^\top\right]^\top. \tag{2.7}$$

Thus, the sound pressure for one microphone can be seen as the short-time Fourier transform (STFT) of the recorded signal for the time frame of interest. The source and noise vectors are also extended in a similar fashion:

$$\mathbf{s}(l) = \left[\mathbf{s}(\omega_1, l)^\top\ \mathbf{s}(\omega_2, l)^\top\ \cdots\ \mathbf{s}(\omega_K, l)^\top\right]^\top, \tag{2.8}$$

$$\mathbf{n}(l) = \left[\mathbf{n}(\omega_1, l)^\top\ \mathbf{n}(\omega_2, l)^\top\ \cdots\ \mathbf{n}(\omega_K, l)^\top\right]^\top. \tag{2.9}$$

According to the way the sound pressure vector is extended in (2.7), the steering matrix generalization leads to the $LK \times D$ matrix $\mathbf{H}$. In this thesis it accounts not only the direct path for the influence (reflections) of the head on the sound field, Therefore it is called the generalized head-related transfer function (GHRTF) matrix. In (2.10), an example of the matrix structure for $L = K = D = 3$ is seen:

$$\mathbf{H} = \begin{bmatrix} | & | & | \\ \mathbf{a}_1(\omega_1) & \mathbf{a}_2(\omega_1) & \mathbf{a}_3(\omega_1) \\ | & | & | \\ | & | & | \\ \mathbf{a}_1(\omega_2) & \mathbf{a}_2(\omega_2) & \mathbf{a}_3(\omega_3) \\ | & | & | \\ | & | & | \\ \mathbf{a}_1(\omega_1) & \mathbf{a}_2(\omega_2) & \mathbf{a}_3(\omega_3) \\ | & | & | \end{bmatrix}, \tag{2.10}$$

with this, the wide-band model is defined as

$$\mathbf{p}(l) = \mathbf{H}\mathbf{s}(l) + \mathbf{n}(l). \tag{2.11}$$

2.3 Spatial Filtering

One of the most popular techniques in array processing is spatial filtering. By this emphasis is applied on the desired signal, which comes from a certain direction, while suppressing the interfering signals coming from others. It can be seen as a beam is formed at the desired look direction, thus the algorithms are usually termed beamformers. In Figure 2.1, an example is illustrated, where the beam is formed to mphasize the front source in detriment of the side source.
Here two beamforming algorithms, namely the filter-and-sum (FSB) and minimum variance distortionless response (MVDR) beamformer, are presented and it is illustrated how the microphone positions affects its performance.

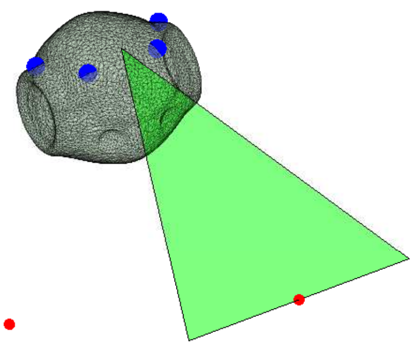


Figure 2.1: Beamform Functionality

2.3.1 Filter-and-Sum Beamformer

This is one of the most fundamental beamform algorithms, which is effectively depicted by Figure 2.2. Each microphone signal is filtered by its respective filter. The time-domain output is given by the summation of all filtered microphone signals:

$$\mathbf{y}(l) = \mathcal{F}^{-1}\left\{\sum_{i=0}^{L-1} \mathbf{p}_i(l) \odot \mathbf{w}_i, \right\} \tag{2.12}$$

where $\mathbf{w}_i = [W_i(\omega_1)\ W_i(\omega_2)\ \cdots\ W_i(\omega_K)]^\top$ are vectors which contain the frequency-domain coefficients of the i-th FIR filter and $\mathbf{p}_i(l)$ is the sound pressure measured by the associated microphone and. $\mathcal{F}^{-1}\{\cdot\}$ and $\odot$ are the inverse STFT and point-wise multiplication operators, respectively. The FSB has the advantage of a somewhat low complexity, although its performance is generally lower as compared to other state-of-the-art methods.

The procedure to obtain the filter coefficients employed in this thesis follows the one proposed in [14, 15], which designs a robust least-squares frequency-invariant (RLSFI)

FSB. The beamformer response is defined as [3, 16]

$$B(\omega, \Omega) = \sum_{i=0}^{L-1} W_i(\omega) g_i(\omega, \Omega), \tag{2.13}$$

where $g_i(\omega, \Omega)$ is the response of the i-th sensor with respect to a plane wave coming from direction $\Omega = \{\theta, \phi\}$ in polar coordinates at frequency ω. The angles follow the same convention from [3, Chap.2]

If free-field propagation is assumed, so the design is invariant to reverberation, the sensor response can be written as $g_i(\omega, \Omega) = e^{-j\mathbf{k}^T\mathbf{p}_i}$, where $\mathbf{k} = \frac{\omega}{c}$ is the wave vector, with c being the speed of sound.

The design of the RLSFI FSB aims to approximate a desired response $\hat{B}(\omega, \mathbf{\Omega})$ in the Least-Squares sense. In order to obtain a numerical solution, the design is done for a set of K discrete frequencies $\omega = \{\omega_0, \cdots, \omega_{K-1}\}$ and S look directions $\mathbf{\Omega} = \{\Omega_0, \cdots, \Omega_{S-1}\}$ [15]. Thus the cost function may be formulated as

$$\min_{\mathbf{w}_f(\omega)} \|\mathbf{G}(\omega)\mathbf{w}_f(\omega) - \hat{\mathbf{b}}\|_2^2, \tag{2.14}$$

where $\hat{\mathbf{b}} = [\hat{B}(\Omega_0), \cdots, \hat{B}(\Omega_{S-1})]^\top$ contains the desired responses for all look directions, $\mathbf{w}_f(\omega) = [W_0(\omega), \cdots, W_{L-1}(\omega)]^\top$ comprises the DTFT-transformed coefficients of each filter at frequency ω and matrix $\{\mathbf{G}(\omega)\}_{[m,n]} = e^{-j\mathbf{k}_m^\top\mathbf{p}_n}$ contains the free-field steering vectors [15].

Moreover two constraints are considered. The first is a distortionless response in the desired look direction:

$$\mathbf{w}_f^T(\omega)\mathbf{d}(\omega) = 1, \tag{2.15}$$

where $\mathbf{d}(\omega) = [e^{-j\mathbf{k}_d^\top\mathbf{p}_0}, \cdots, e^{-j\mathbf{k}_d^\top\mathbf{p}_{N-1}}]^T$ is the steering vector which corresponds to the desired look direction Ω_d.

The other is regarding the white noise gain (WNG) [17], which is defined as

$$\frac{|\mathbf{w}_f^T(\omega)\mathbf{d}(\omega)|^2}{\mathbf{w}_f^H(\omega)\mathbf{w}_f(\omega)} \tag{2.16}$$

and $(\cdot)^H$ denotes the conjugate transpose of a matrix or vector. This quantity is able to quantify the array's capacity of suppressing spatially uncorrelated noise. Therefore

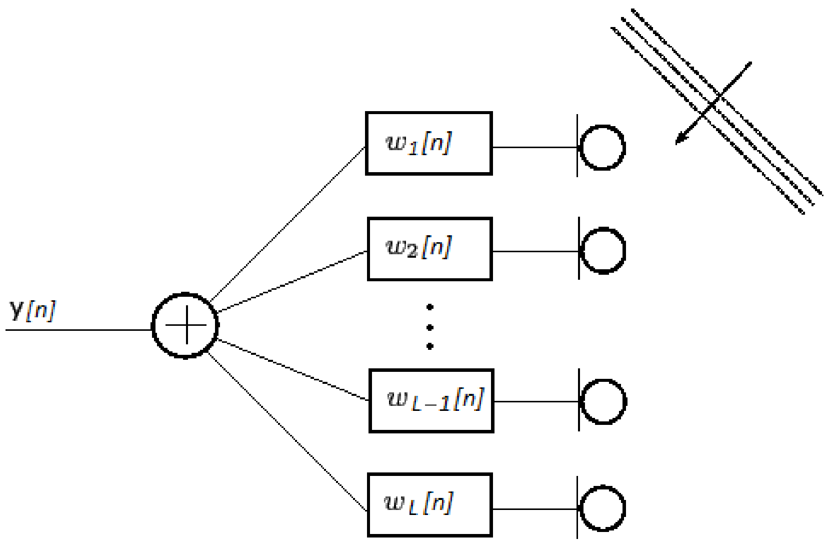


Figure 2.2: FSB block-diagram

positive value on a logarithmic scale indicates attenuation of uncorrelated noise, while negative values signs for amplification.

This is important because small random errors such as sensor mispositioning or deviations in the sensor characteristics can be modeled as white noise. Therefore, the WNG can be used as a measure of the beamformer robustness against these kind of errors. It is imposed that $WNG \geq \gamma > 0$.

The optimization problem (2.14) is convex [14] and has to be solved for each frequency ω_i. A toolbox for solving convex problems in MATLAB called CVX [18] is used for such task. In a final step, the FIR filters of length L are obtained by FIR approximation of the optimum filter weights [15].

Throughout the whole investigation, the design is made for $300 \leq f \leq 5000$ Hz, because the intention is to only capture the speech, which lies almost entirely in this range. Regarding the WNG constraint, $\gamma = -15, -30$dB are used for showing its effect on the design. Nonetheless, only $\gamma = -15$dB is used for comparison between different arrays.

2.3.2 MVDR Beamformer

The second beamformer which is employed in this thesis is the minimum variance distorless response beamformer (MVDR). It is a specific case of the linearly constrained minimum variance (LCMV) beamformer [19]. The idea is to minimize the output signal's power given a unique constraint which guarantees an undistorted response for the desired source.
The noise output power of the FSB beamformer can be written as

$$E\left\{\|y\|^2\right\} = E\{\mathbf{w}^H\mathbf{p}\mathbf{p}^H\mathbf{w}\} = \mathbf{w}^H\mathbf{\Phi}_{pp}\mathbf{w}, \tag{2.17}$$

where $\mathbf{\Phi}$ is the crosspower spectral density matrix and $\mathbf{w} = \left[\mathbf{w}_0^\top \; \mathbf{w}_1^\top \; \cdots \; \mathbf{w}_{L-1}^\top\right]^\top$ is the concatenation vector of all filter coefficients.
A distortionless response constraint for the desired signal in the output is placed, such as

$$\mathbf{y}_{s_d} = \mathbf{w}_d^H\mathbf{d}\mathbf{s}_d \overset{!}{=} \mathbf{s}_d, \tag{2.18}$$

where $\mathbf{s}_d$ is the desired signal component. The MVDR design consists in minimizing the output signal power subject to a distortionless response constraint in the desired look direction:

$$\min_{\mathbf{w}} \mathbf{w}^H\mathbf{\Phi}_{pp}\mathbf{w}; \quad \mathbf{w}^H\mathbf{d} = 1. \tag{2.19}$$

Therefore, the MVDR filter coefficients are obtained by [19]

$$\mathbf{w}_{MVDR} = \frac{\mathbf{\Phi}_{pp}^{-1}\mathbf{d}}{\mathbf{d}^H\mathbf{\Phi}_{pp}^{-1}\mathbf{d}}. \tag{2.20}$$

The MVDR beamformer has generally a notably higher performance than the FSB design, since it is directly based on the signal and environment characteristics. However, it has a high computational cost and the estimation of the PSD is already a problem to be issued.
Many MVDR design algorithms were proposed, like the generalized sidelobe canceler (GSC) [20] and adaptive directional microphone (ADM) [21]. In this thesis, it is assumed that the signal components at the beamformer input are known, such that

spectral covariance matrix $\mathbf{\Phi}_{\text{pp}}$ can be estimated easily. Although it is not a realistic method, it will serve to benchmark an upper bound for the microphone array beamforming performance.

2.3.3 HRTF-based beamforming

Traditionally, the beamformer design uses free-field steering vectors. However in the context of robot audition, this approach does not account for the head's influence, like attenuations and reflections. This effects reduce the algorithm performance. With this in consideration, this thesis incorporates the HRTFs, following a method proposed in [15]. Instead of employing the steering vectors matrix $\mathbf{G}$, the GHRTF matrix $\mathbf{H}$ is used. Accordingly, the transfer function for the desired look direction $\mathbf{h}_d$ is employed rather than $\mathbf{d}$. By accounting this additional information, the separation capability is increased.

Since the FSB design is made separately for each frequency, the narrow-band steering matrix $\mathbf{A}$ and response form desired direction $\mathbf{a}_d$ are employed. The problem is now defined as

$$\min_{\mathbf{w}_f(\omega)} \|\mathbf{A}(\omega)\mathbf{w}_f(\omega) - \hat{\mathbf{b}}\|_2^2, \tag{2.21}$$

$$\mathbf{w}_f^T(\omega)\mathbf{d}(\omega) = 1, \quad \frac{|\mathbf{w}_f^T(\omega)\mathbf{a}_d(\omega)|^2}{\mathbf{w}_f^H(\omega)\mathbf{w}_f(\omega)}. \tag{2.22}$$

For the MVDR design, the filter coefficients are given by:

$$\mathbf{w}_{MVDR} = \frac{\mathbf{\Phi}_{pp}^{-1}\mathbf{h}_d}{\mathbf{h}_d^H\mathbf{\Phi}_{pp}^{-1}\mathbf{h}_d}. \tag{2.23}$$

Chapter 3

Optimization of Microphone Array Configurations

The main task is finding the optimal microphone positions. By optimal, it is meant an array which is able to retrieve the maximal possible information from its surrounding sound field, given a desired range of frequencies and a set of source possible positions. In the last chapter, it was shownthat the GHRTF matrix $\mathbf{H}$ is capable to effectively characterize the considered scenario. Therefore a measure is needed to assess how much information a certain configuration can obtain.

In this thesis the effective rank [22] of $\mathbf{H}$ is considered. It is a real-valued extension to the classic rank measure [23]. This metric, proposed by Roy *et al.*, offers better insight than its integer-valued counterpart in a number of applications.

Given the singular value decomposition (SVD) of $\mathbf{H}$ has Q non-zero singular values arranged in descending magnitude, its effective rank is given by

$$\mathcal{R}(\mathbf{H}) = \exp\left\{ -\sum_{i=1}^{Q} \bar{\sigma}_i \log \bar{\sigma}_i \right\}, \tag{3.1}$$

where $\bar{\sigma}_i$ are the energy-normalized singular values of $\mathbf{H}$, which are obtained by

$$\bar{\boldsymbol{\sigma}} = [\bar{\sigma}_1 \cdots \bar{\sigma}_Q] = \frac{\boldsymbol{\sigma}}{\|\boldsymbol{\sigma}\|_1}, \tag{3.2}$$

and $\|\cdot\|_1$ is the l_1 norm for matrices [23].

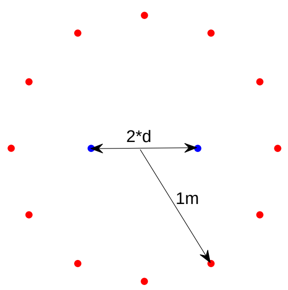
In other words, the effective rank evaluates the singular values uniformity, by considering the spectral entropy [24] of $\bar{\boldsymbol{\sigma}}$. Thus it is minimized when $\sigma_1 \gg \sigma_i, \forall i \neq 1, (\mathcal{R}(\mathbf{H}) = 1)$ and maximized when all singular value are equal $(\mathcal{R}(\mathbf{H}) = Q)$. This can be also interpreted as the importance of each singular matrix components in the SVD.

In this context, it means a microphone array with a low effective rank only has few microphones retrieving meaningful information, while the other ones are redundant or highly correlated. When the effective rank is maximized, all microphone retrieve meaningful information and the array in overall is capable of better sound acquisition. This can be shown in the following experiment: Let be 12 sources in a circle with $r = 1m$ and a pair of microphones with spacing $2d$, as illustrated in Figure 3.1a. The microphones are opposite to each other with respect to the center of the source circle. Figure 3.1b shows how the effective rank of the corresponding steering matrix $\mathbf{H}$ behaves, as spacing between the microphones is changed. The effective rank is very low when the microphones are close to each other in the center and when they get closer to the circle of sources. The highest effective rank is obtained when the sources are approximately 0.4 meters from the center. Another fact is that the effective rank is bounded by 7, and not 12. This happens because the 5 pairs of broadside sources opposite to each other in respect with the center have identical responses because of symmetry. Thus, the effective rank is reduced.

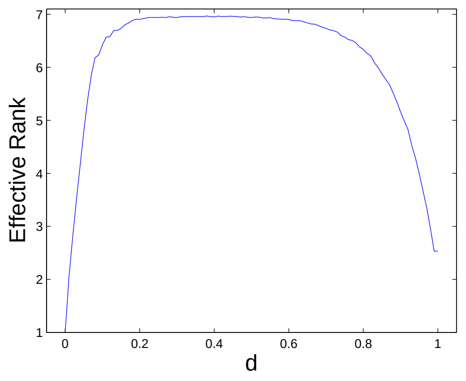
3.1 Cost Function Setup

Having now defined our cost function, we can formulate the optimization problem. Let L be the desired number of microphones in the array and M the number of possible positions on the robot head, whose set is denoted by $\mathcal{M}$. Also define the GHRTF matrix corresponding to a certain microphone position $m \in \mathcal{M}$ by $\mathbf{H}_m$. The optimization problem is defined in the following way:

$$\mathcal{L}^* = \max_{\mathcal{L}} \mathcal{R}(\mathbf{H}_{\mathcal{L}}), \tag{3.3}$$



(a) Setup



(b) Result

Figure 3.1: Effective Rank Experiment

where $\mathcal{L} = \{l_1, l_2, \cdots, l_L\} \subset \mathcal{M}$ is a subset of L microphone positions and $\mathbf{H}_{\mathcal{L}} = [\mathbf{H}_{l_1}^\top, \mathbf{H}_{l_2}^\top, \cdots, \mathbf{H}_{l_L}^\top,]^\top$ is the column concatenation of the GHRTF matrices for a given array [13].

3.2 Parameters Setup

Regardless of the somewhat simple definition of (3.3), there are a number of parameters and configurations to be defined. In this section, further details on the optimization process are provided.

3.2.1 Possible Microphone Positions

The initial procedure in this study is to simulate the head-related transfer functions (HRTFs) between each possible position on the surface of the robot head and each considered source position.

First, the set of possible microphone positions is discussed. In contrary to[13], where the study was carried out for the KU-100 dummy human head surface, the transfer functions are calculated in respect with the head surface of the robot NAO, which is depicted in Figure 3.2. In this model there are $M = 9996$ possible microphone

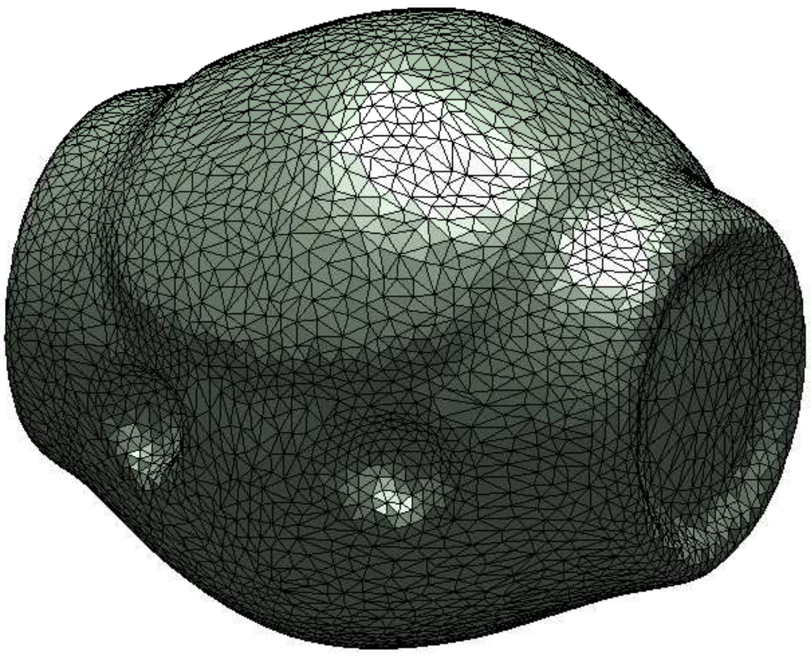


Figure 3.2: Geometrical model of the NAO robot head.

positions. Each position is a vertex on the geometry shown in Figure 3.2.

3.2.2 Number of Sources and its Configuration.

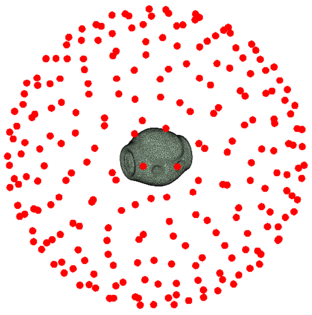
The transfer functions also depend on the source configuration, which is considered during the optimization. Therefore a set which reflects realistic scenarios has to be chosen for the optimization process. In [13] three scenarios are considered: (a) 240 uniformly distributed sources in a sphere, (b) 36 uniformly distributed sources as a horizontal belt and (d) 36 uniformly distributed sources as a vertical belt. While the configurations (a) and (b) closely resemble practical possibilities for source location, (d) is seldom seen in a practical scenario and was utilized to better illustrate the properties of the effective rank.

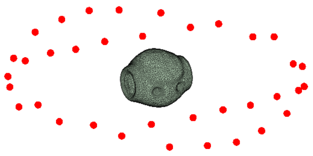
Usually the robot desired sources are in its 'line of sight' or above it. Another fact is that moving parts of the robot cause severe amount of noise, which is also known as self-noise. In order to tackle both issues, the following source setup is proposed: a set of uniformly distributed sources in a sphere with no sources placed near the south pole. For instance, no sources are placed for $\theta > \theta_l = \frac{2\pi}{3}$ as the third source configuration scenario in this thesis. Figure 3.3 displays where the sources are places in all three configurations.

With this, it is expected that the optimized microphone positions will be in average dislocated to the top part of the robot head, leading to better beamforming performances for sources above or in the same level of the robot's line of sight, which is more common. Furthermore, less self-noise retrieval may be also an advantage of this source configuration, but is not analyzed in this thesis..

Note that both (b) and (c) may be obtained by selecting a subset of sources used on (a), which dismisses the need of recalculating the HRTFs.

The maximum effective rank which the optimal array can have is of $\min[LK, D]$.


(a) All Sources


(b) Horizontal Belt

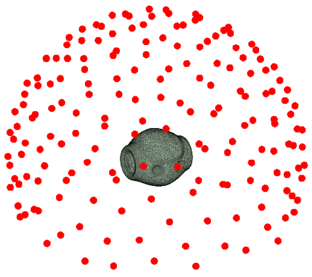

(c) Only Top

Figure 3.3: Source Configurations

3.2.3 Number and Type of Microphones

Arrays with different number of microphones are used in state-of-the-art robot auditory systems, as already discussed in the Chapter 1. Since the study is carried out for the NAO robot head, $L = \{4, 12\}$ is employed, so a fair comparison with the two available arrays is possible, since arrays with the same number of microphones are compared.

Another aspect is the type of the microphones which are used. Directional microphone usually have better response towards certain look directions, but are more costly and difficult to implement. On the other side, omnidirectional microphone are cheap and easy to be employed. Another advantage is they are better suited for beamforming algorithms. I this thesis omnidirectional microphones are considered, because it is the type which will be used in the next version of the NAO robot.

3.2.4 Frequency Range and Number of Frequency Bins

The GHRTF matrix coefficients are calculated for each source and possible microphone at a certain frequency range. This range must comprise the spectrum of the human voice. With this aim, 51 frequency bins are considered between $0 - 5kHz$, with steps of $100Hz$.

3.2.5 Optimization Algorithm

The cost function (3.3) is a non-convex problem and it is not possible to evaluate all possible microphone array configurations, because this yields the effective rank calculation of $\frac{M!}{L!(M-L)!}$ possible configurations. Hence, there are approximately 417 trillion arrays by using 4 microphones and 10^{39} for 12 microphones.

In [13], the Genetic Algorithm (GA) [25] is employed to solve the optimisation problem (3.3). It is a heuristic method based on a natural selection process. Although optimal solutions are not guaranteed, it has a advantage of spanning a great and diverse number of solutions in its search.

Each so-called individual is considered to be a vector which stores the indices of the microphone positions to be used in the GHRTF matrix.
These indices are randomly assigned to the possible microphone positions at the robot head. Hence, there is no correlation between index value and microphone position. Because of this, there is no guarantee that individuals with high effective rank will produce better solutions.
This happens because the genetic algorithm lies on the premise that good solutions can be found from mixing up previous good solutions (crossover) or changing its values by small amounts (mutation). Although it is not the case here. For instance, if a light mutation changes one index of one microphone of an array configuration by a small value, like 1, it places the microphone in a completely different position on the head, which may lead to a much worse new microphone array configuration. If the indices values were at least loosely tied with their respective microphone position, the GA algorithm would be much more suitable.
In contrast to [13], which used default settings in MATLAB's 'ga' function, the crossover and migration ratio were increased, while reducing retained elite individuals. This yields a higher dynamic range of the search.

3.3 Optimum Microphone Array Configurations

Table 3.1 shows the NAO and optimum arrays' effective rank of the **H** matrix and the respective maximum theoretical value. It is important to mention that there are six optimal arrays and two NAO arrays, whose effective rank is calculated for the three different source configurations. The indices of the microphone positions in each **H** matrix of each array can be found in Appendix A.
The effective rank of the optimal arrays is higher than the current and the constrained optimzed array in all considered cases. It is also noticeable that the maximum theoretical effective rank is much higher compared to the ones of the microphone arrays. This effect occurs because of the symmetry present in the source configuration and

Table 3.1: Optimum microphone arrays

L	Source config.	$\mathcal{R}(\mathbf{H}_{NAO_L})$	$\mathcal{R}(\mathbf{H}_{\mathcal{L}^*})$	$\max \mathcal{R}(\mathbf{H})$
	Belt	9.45070	12.1735	36
4	All sources	13.3562	14.2533	208
	Only top	13.6730	14.2004	180
	Belt	14.4608	16.4867	36
12	All sources	26.1757	28.5415	240
	Only top	24.7887	26.8822	180

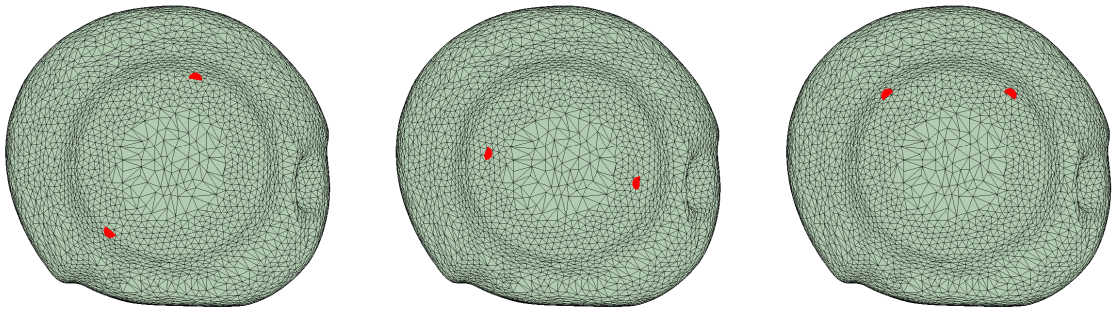

(a) All Sources (b) Horizontal Belt (c) Only Top

Figure 3.4: Optimal microphone positions for $L = 4$.

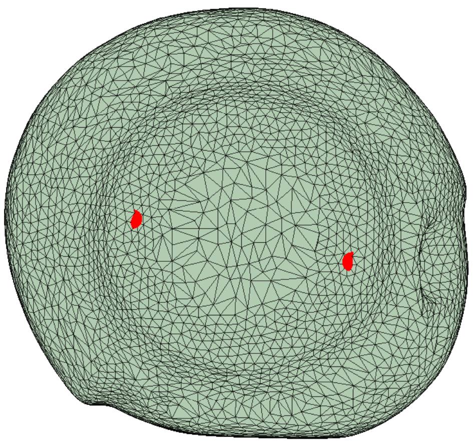
also the robot head geometry, which leads to very similar or equal responses for many positions.

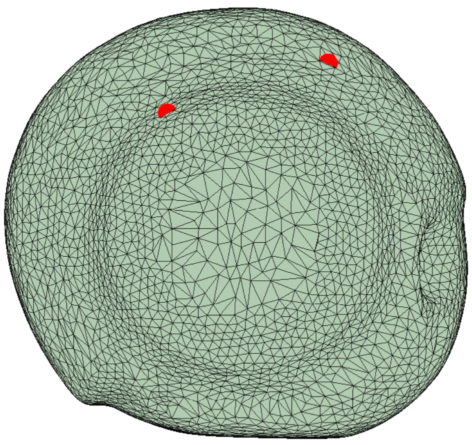
In Figure 3.4, the optimal configurations for $L = 4$ are compared. Only the right view of the robot head is displayed, because the arrays are symmetrical with respect to the vertical plane passing through the center of the head. The optimal microphone positions are always at the circular edge of the head. For the 'all sources' configuration, there is one microphone above and one down, while in the array optimized for the 'horizontal belt', both microphones lie in the middle. At last, the microphones for the 'only top' source configuration are on the upper region of the head, just as the average. These positions are in accordance with what was expected, because they reflect the positioning of the source configurations used in the optimization process (Figure 3.3). Although the optimization was done for three different source configurations, further performance assessment is done only for arrays which were optimized for the 'horizontal belt' source setup. This is because the current code for beamform design and evaluation is suitable for source configurations with constant azimuth angle.

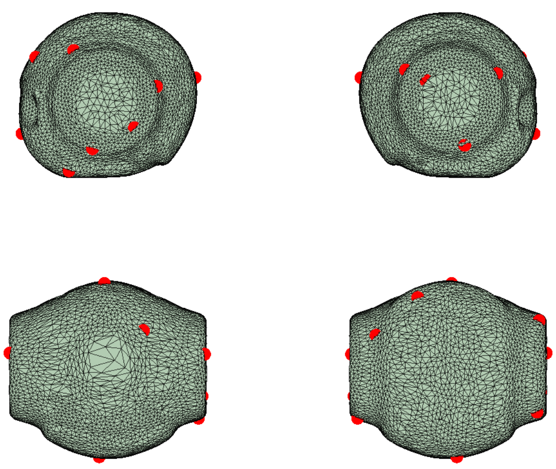
In Figure 3.5, the optimal configurations for the horizontal belt are put side-by-side with the NAO arrays. Because of symmetry, only the right view of the head is used again for $L = 4$. However the 12-microphone arrays do not have this property and the microphones are much more spread over the head surface. Therefore, 4 different views are used for displaying them: left (upper left), right (upper right), bottom (bottom left) and top (bottom right).

Regarding the arrays with 4 microphones, the optimal one has its pairs of microphones more far apart. This larger spacing may lead to better beamforming design performance.

The 12-microphone configuration from NAO will be used in the future version and was obtained by using the method in [13], subject to constraints on the possible microphone ositions due to manufacturing limitations.


(a) Proposed, $L = 4$


(b) NAO, $L = 4$


(c) Proposed, $L = 12$

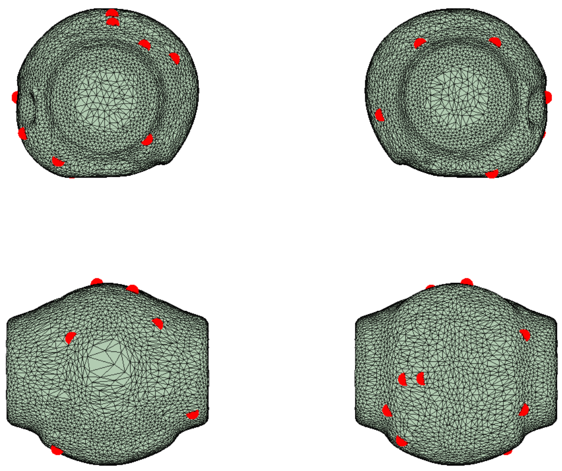

(d) NAO, $L = 12$

Figure 3.5: Signal independent metrics for front look direction.

Chapter 4

Performance Assessment

4.1 Simulation Setup

In order to evaluate which array configuration yields a better signal enhancement performance, the simulation places the robot head in a reverberant room of dimensions $5m \times 5m \times 3m$. Different scenarios with one desired and one interfering source are considered. As in the optimization procedure, only sources in the horizontal belt around the head are considered ($\phi = \pi/2$).

12 uniformly separated sources positions in a circle with $r = 2m$ are considered with no environment noise. Moreover, for each desired source position, the interferer is placed at one position to the left and to the right of the desired speaker, i.e.

$$\theta_i = \theta_d \pm 30, \tag{4.1}$$

where θ_i, θ_d are the azimuth angles of interferer and desired source, respectively. Figure 4.1 displays the described scheme. Therefore 24 different configurations are possible, because of the 12 look directions with two possible interferer positions. For each desired look direction, the beamformer has to be redesigned.

The HRTFs, which considers only the direct path (no reflections on the walls), are used for the FSB beamformer design, while the RIRs are used for convolution with the source signals, thus obtaining the respective microphone signals.

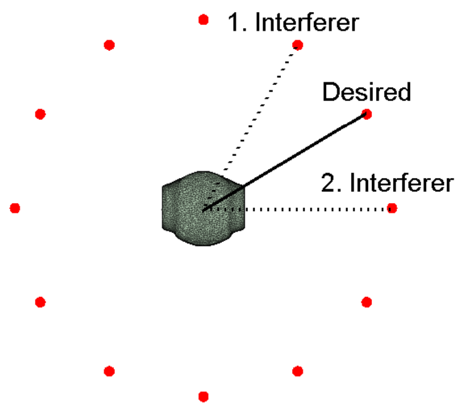


Figure 4.1: Proposed Setup

The HRTFs need to have a higher spatial (angular) resolution for the beamform design than the considered desired source positions, which uses a step of 30 degrees. Hence, 72 uniformly distributed sources in the horizontal belt are used, giving a resolution of 5 degrees.

For each scenario, signal-independent and -dependent measures are considered.

Regarding the signal-independent metrics, only the FSB is analyzed with respect to the beampattern, the beamformer response in the desired look direction, and the WNG.

Regarding signal-dependent performance, (i) signal-to-noise ratio (SNR) and (ii) automatic speech recognition (ASR) score are used for both FSB and MVDR beamformers.

The SNR is a widely known metric, which calculates the ratio between the power of desired signal components and the power of the present noise. Is is calculated in this context by

$$\mathrm{SNR} = \log_{10} \frac{\sigma^2_{\mathbf{s}_d}}{\sigma^2_{\mathbf{n}}}, \tag{4.2}$$

where $\sigma^2_{\mathbf{s}_d}, \sigma^2_{\mathbf{n}}$ denote the variance of the desired and interferer signal, which in this case is considered to be the signal of the interfering speaker.

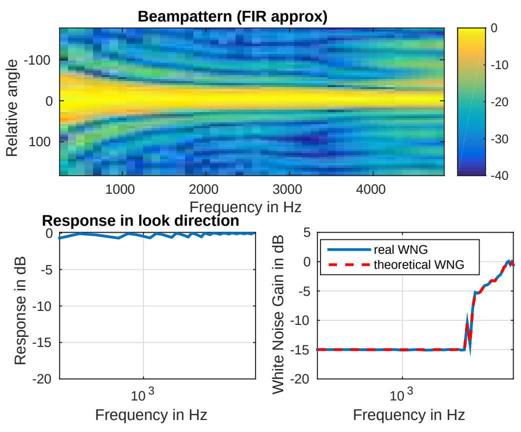
The ASR score is calculated by an automatic speech recognizer. Methodology is the same one of [15]. An ASR engine called PocketSphinx [26] is employed with an acoustic model, which is trained on clean speech from the GRID corpus [27]. The utilized features are MFCC+Δ + $\Delta\Delta$ and cepstral mean normalization. As in the CHiME challenge [28], ASR score computation is done only by evaluating the letter and number in the utterances. In this evaluation, the test signal contained 200 utterances.

In order to assess the beamforming performance, we calculate the metrics at the beamformer output. Previously in this study, the input, which is calculated with the unprocessed microphone signals, and the gain with respect to the beamformer output were also calculated. For this, one microphone has to be regarded as the reference one. Since there is no central microphone, as in [15], each microphone has to be considered as reference and then average the results at the end. Hence, there are 96 or 288 possible values for the 4 and 12 microphone arrays, respectively. However, the output and gain were omitted in order to not make this section cumbersome, although it is worthwhile mentioning that the proposed arrays had naturally a better SNR and ASR input of approximately 0.1 dB and 0.5%, respectively.

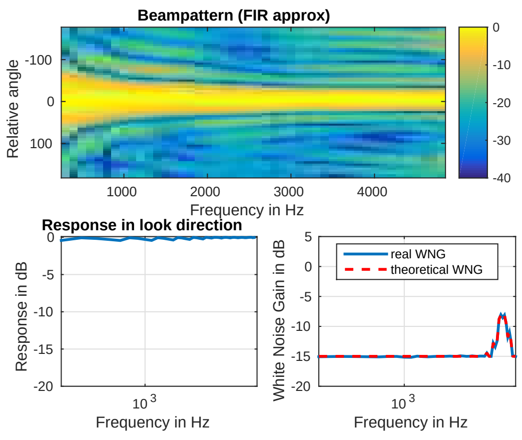
For each microphone array, different reverberation times are also considered. In order to obtain the respective reflection coefficients for matching desired reverberation time T, we used the approximation from [29, p. 172–173]:

$$T \approx 0.163 \left[\frac{s}{m}\right] \frac{V}{\bar{\alpha} A}, \tag{4.3}$$

where V, A are the volume and area, respectively, and $\bar{\alpha}$ is the pondered average of the absorption coefficient in respect with the walls' area. It is considered that all walls have the same absorption. Length of the beamforming FIR filters was set to $L = 128$. The processing is made in time domain for the FSB design, whereas the MVDR beamformer calculation takes place in the frequency domain.



(a) Proposed, $L = 12$



(b) NAO, $L = 12$

Figure 4.2: Signal-independent metrics for front look direction.

4.2 Results

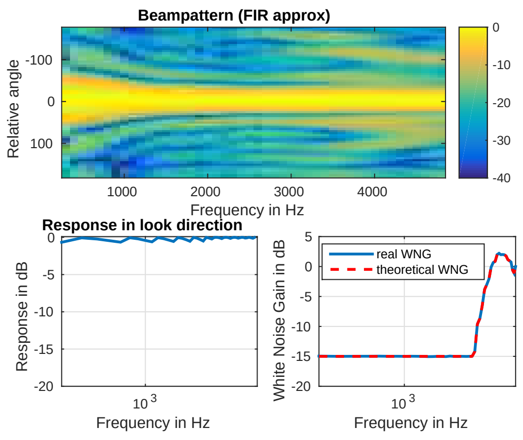
4.2.1 Signal-independent Metrics

First the comparison is made only in one specific look direction for $T = 150ms$ using the 12-microphone arrays. The 4-microphone arrays weren't used because the difference between the designs were barely noticeable. Figures 4.2, 4.3 and 4.4 shows the signal-independent metrics for the the front, right and left look directions, repectively.

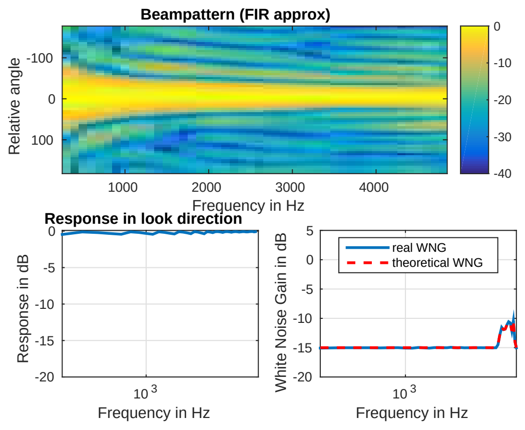
The beampattern is plotted with the azimuth angle relative to the look direction and not absolute, so the main beam is centered. The Proposed arrays have a much narrower beam, especially in lower frequencies, and weaker spatial aliasing. Regarding WNG, the proposed has better figures for the higher frequencies.

The beamformer response in both are by the same level, but the NAO's arrays have a slightly flatter response in the desired look direction.

Another evaluation is regarding different values of γ for the FSB design. In Figure 4.5 both 12-microphone arrays are used for designing beamformers with $\gamma = -15$dB and $\gamma = -30$dB. We see that designs with a lower constraint in the WGN have an even narrower beam, especially for the lower frequencies.

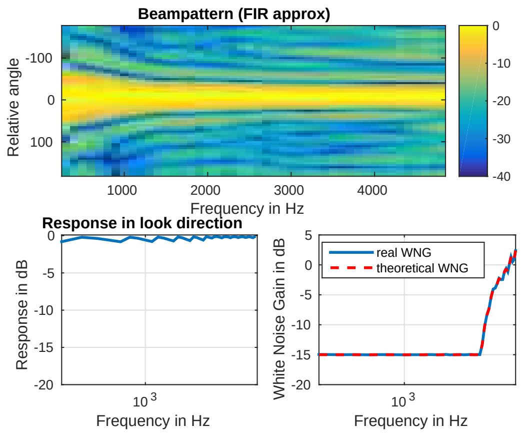


(a) Proposed

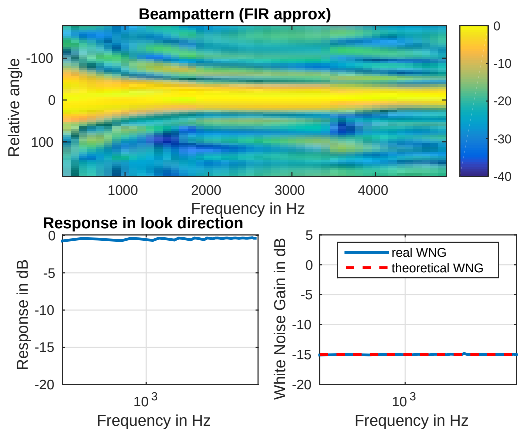


(b) NAO

Figure 4.3: Signal-independent metrics for right look direction.

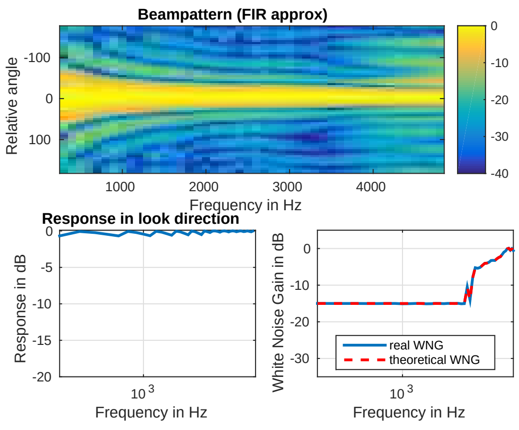


(a) Proposed

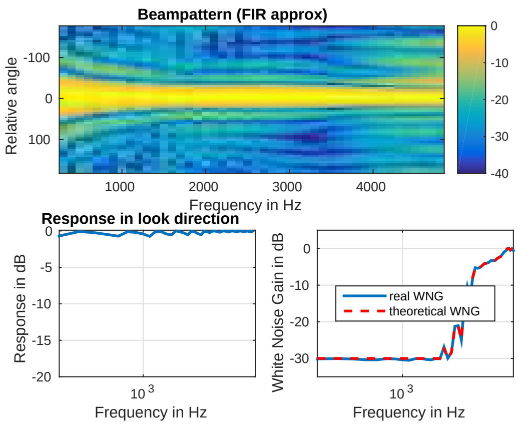


(b) NAO

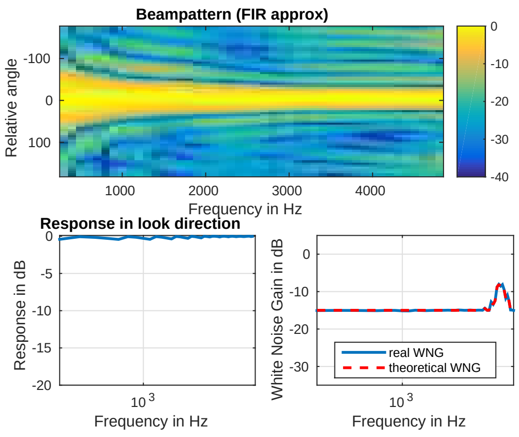
Figure 4.4: Signal independent metrics for left look direction.



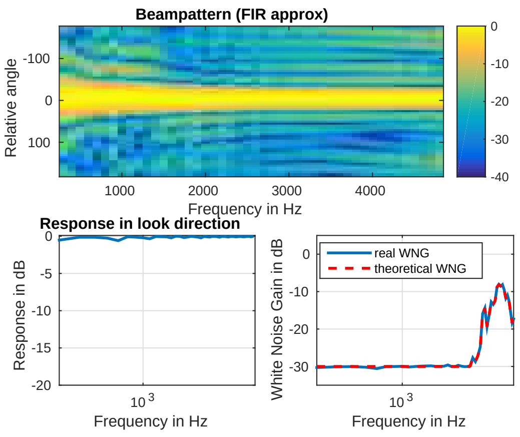
(a) Proposed, $\gamma = -15$dB



(b) Proposed, $\gamma = -30$dB



(c) NAO, $\gamma = -15$dB



(d) NAO, $\gamma = -30$dB

Figure 4.5: Signal-independent metrics for different values of γ (front look direction).

Table 4.1: Signal-dependent Metrics for Three Different Look Directions ($T = 150ms, L = 4$)

Look Direction	Metric	Proposed	NAO
Front	SNR output [dB]	1.951/9.744	1.646/6.331
	ASR output [%]	57.84/69.88	52.47/56.91
Right	SNR output [dB]	1.078/6.664	1.910/4.769
	ASR output [%]	52.59/55.88	52.14/49.94
Left	SNR output [dB]	2.531/6.753	2.053/4.360
	ASR output [%]	50.53/56.91	52.81/48.12

4.2.2 Signal-dependent Metrics

Tables 4.1 and 4.2 displays the signal-dependent measures for such configurations. In the output metrics, the FSB result is on the left and the MVDR result is on the right. All FSB designs were obtained for $\gamma = -15$dB. The best results are marked with green. Regarding the SNR, the proposed arrays perform clearly better in almost all cases, with exception of $L = 4$ in the right look direction using the FSB design.

With respect to the ASR score, Again both proposed arrays clearly outperform the NAO's arrays in almost all evaluated metrics. The exception is the 12-microphone NAO array using the MVDR design.

Now all 24 desired source+interferer possible configurations are averaged for $T = \{150,\ 500ms\}$. The results are shown in Tables 4.3 and 4.4. They follow the same tendency as for one specific look direction. The NAO arrays have inferior SNR performance in all considered cases. The optimized arrays are also better in almost all cases regarding ASR score, with exception again of the 12-microphone NAO array ASR score output using the MVDR design.

Table 4.2: Signal-dependent Metrics for Three Different Look Directions ($T = 150ms, L = 12$)

Look Direction	Metric	Proposed	NAO
Front	SNR [dB]	2.438/11.09	1.973/9.888
	ASR [%]	58.71/77.92	51.46/80.00
Right	SNR [dB]	2.177/10.07	2.160/10.09
	ASR [%]	60.91/73.35	51.91/78.58
Left	SNR [dB]	3.397/11.17	2.281/10.98
	ASR [%]	59.35/74.88	51.30/78.64

Table 4.3: Average of Signal-dependent Measures for all look directions ($T = 150ms$)

L	Metric	Proposed	NAO
4	SNR [dB]	1.659/7.717	1.45/5.635
	ASR [%]	55.35/62.10	53.54/54.09
12	SNR [dB]	2.364/11.37	1.818/11.03
	ASR [%]	58.94/75.40	51.75/79.83

Table 4.4: Average of Signal-dependent Measures for all look directions ($T = 500ms$)

L	Metric	Proposed	NAO
4	SNR [dB]	1.135/4.694	1.045/3.948
	ASR [%]	24.34/22.85	24.18/19.99
12	SNR [dB]	1.330/6.945	0.706/6.609
	ASR [%]	21.31/32.46	18.07/39.00

Chapter 5

Final Remarks

In this thesis, the connection between effective rank and microphone array configuration is investigated, in the context of robot audition. Optimal arrays with 4 and 12 microphones for the 'horizontal belt' source configuration are calculated. They are optimal by maximizing the effective rank, a quantity used for estimating the amount of information the array is able to perceive.

The array's performance is compared to the current 4-microphone array of the NAO robot and the new 12-microphone array which is currently being developed. The simulation environment consists of the robot head placed in a reverberant room with two different reverberation times. Different scenarios presenting one desired and one close interfering source are considered. Signal-independent analysis shows that the optimized arrays have a narrower main beam and are more robust against small random errors than the non-optimized arrays in higher frequencies. Regarding signal-dependent measures (SNR and ASR score), the proposed arrays have clearly better performance in almost all considered cases. There are many possibilities for future work in this area. It is important to deeper investigate the effective rank meaning in the microphone array context. Furthermore, it is possible to try other optimization algorithms for maximizing it and more different evaluation scenarios, with noisy environments and more interferers. The effective rank optimization setup may also be used in a variety of other problems. One of them with direct application is the positioning of antenna arrays.

Appendix A

Microphone Arrays Indices

L	Source config.	Indices
4	Belt	6760,9276,1118,9396
	All sources	258,5404,6738,4735
	Only top	1479,8184,9974,9348
12	Belt	3382,6584,7467,9902,5666,7854,2559,2938,6556,8578,6188,3738
	All sources	1138,7592,7871,934,8498,3217,8459,8648,1107,3190,8092,1130
	Only top	9579,9439,7398,6640,6734,4793,4605,5795,7405,6176,6017,1998
NAO_4		8947, 1021, 622, 241
NAO_{12}		773, 3347, 3124, 2306, 5161, 9304, 9535, 7831, 6899, 8881, 905, 7110

Appendix B

MATLAB Codes with Guideline

All the procedures in this work were programmed in MATLAB [30]. The scripts are basically divided in three main phases:

- Obtain the optimal microphone array positions and generate the EARO data objects.
- Simulate the respective RIRs and HRTFs.
- Beamforming design and performance assessment.

The most important scripts can be found below.

B.1 Microphone Positions Optimization and EARO object

This part consists of six MATLAB scripts:

- `OptPosSolver.m` – main script, where the number of microphones, set of sources and number of iterations are determined.
- `effrank.m/computeEffectiveRank.m` – Effective rank calculation.

- `fitf.m` – defined fitness function (in this case the negative of the effective rank).
- `visualizeGeom.m` – plots the microphone positions, given its indexes.
- `create_earoObject_withGHRIRs.m` – creates the EARO object, given a microphone array.
- `earo.m` – set of functions for creating and editing the EARO object.

I ran the main script several times, considering different numbers of microphones and source configurations. The different source configurations were obtained by removing some indices of the **H** matrix. The program returns the indices and the microphone positions of the optimal array, which are used for creating the Embodied Audition of Robots Object (EARO). It is a struct file for microphone arrays which can store different kinds of information, e.g., the number and position of the microphones, .WAV files, and HRTFs. In this thesis, we are only interested in storing the array geometry and respective HRTFs.

B.2 Simulation of room and head-related impulse responses

In this part, the room impulse responses (RIRs) and head-related transfer functions (HRTFs) between the sources and microphones are generated. The `'main_naoRoomSim.m'` script is used for this task, while `'main_FreeFieldSim.m'` can be employed in free-field scenarios.

For each microphone array and source configuration case, the following inputs are given:

- Array's EARO file.
- Number of microphones.
- Source positions (for simulating RIRs) or look directions (for simulating HRTFs).

- Room dimensions.
- Absorption coefficients of the walls, when calculating RIRs.

The simulation uses MATLAB in conjunction with MCRoomSim [31]. It is a multi-channel acoustic acoustics simulator for generating reverberant impulse responses. The output responses and transfer functions are then used in the last part of the work.

B.3 Beamforming Design and Performance Assessment

In the last part of the process, the microphone signals are combined using beamforming and evaluated. The whole code is run through the `'main.m'` script, which is basically comprised in the following steps:

- Setup: number of microphones, look direction angles in degrees, interfering source index and load HRTF.
- Run `'SetAcousticScenario.m'`, which loads the RIR and defines the source positions. It is also possible to tune noise and filterbank parameters.
- Create microphone signals using `'LoadMicInputs.m'` with sample rate $f_s =$ 10 kHz.
- Beamform design by `'main_robust_FSB.m'`/`'robust_FSB.m'`. Design look directions are defined in `'BF_Array_Geometry.m'` and in `'BF_Plot_BP.m'` the beampattern and beamformer response are calculated.
- Calculate beamformer output using the microphone signals and define reference microphone.
- The output signal is resampled to 16 kHz. After this, SNR and ASR score are calculated in `'Evaluation.m'`.

Except for HRTF loading and number of microphones, all steps are iteratively repeated for different look directions, interferer positions and reference microphones. In the end all results are averaged and saved in a .mat file.

B.4 Remarks

In this section I point out some details for further usage of this code:

- The angle convention changes throughout the phases. It is commented in each script which one should be adopted.
- The RIR/HRTF Simulation in phase 2 can only be done in Windows, because of the MCRoomSim compatibility.
- The ASR and PESQ (not used in this thesis) scores in Phase 3 can be only calculated in Linux, because of its engine's executables. Nonetheless you can shut them down by setting the third and fourth variables of the Evaluation function to zero and run the other metrics in Windows.
- Since there are many source configurations and microphone arrays to be evaluated, I recommend including each one in a switch/case structure and make all variables (positions, file to be loaded) dependent on one variable at the main script.
- First Part - Optimization Function

```matlab
%% %%%%%%%% optimal microphone positioning - GAs based solution %%%%%%%%
%% =====prepare data=====
% load H and make it 'global'
clear all; close all; clc;
global H;
load('../AllDataSpaceUniform.mat');
```

```

%extract horizontal belt (sources)
%theta = [1.4416 1.7000]
% H = H(:,:,[28 30 32 34 38 40 41 42 43 44 45 46 88 90 92 94 98 101 103 105 ...
% 136 138 140 142 172 174 176 178 193 195 203 204 208 210 212 214]);

%remove bottom (sources)
%theta = [1.0274 pi]
% H(:,:,[2,7,13,15,25,27,29,33,37,39,50,52,55,56,62,67,74,78,79,82,89,93,95, ...
% 96,100,104,110,115,124,125,128,129,134,139,143,144,146,148,151,152,161,162, ...
% 165,166,170,175,181,183,185,189,196,200,206,211,227,228,230,234,235,238]) = [];

%% =====obtain a solution=====
L=12; % number of positions (microphones) to choose
% display the process (There are lot's of additional parameters and options,...
% but usually the default work fine. If you want to change something,
% just use MATLAB help.)

% pts = gaoptimset('PlotFcns',@gaplotbestfun);

% find optimal positioning using GA
% (fitf()-optimization objective function that uses eff. rank)

opt = zeros(5,L+1);

options = gaoptimset('CrossoverFraction',.9,'EliteCount',5, ...
                     'MigrationFraction',.2);

for i=1:50,
    [GAsol, fval]=ga(@fitf,L,[],[],[],[],ones(1,L), ...
                                 size(H,1)*ones(1,L),[],1:L,options);
    opt = cat(1,opt,[fval, GAsol]);
    opt = sortrows(opt);
    opt = opt(1:5,:);
end
```

```

GAsol = opt(:,2:L+1);
GAsol = GAsol(:);
GAsol = GAsol';

%% visualize results

load geometry
load('../Geometry_of_NAOs_head.mat');

figure;
visualizeGeom(geomData);
patch('vertices',geomData.vertices,'faces',geomData.faces(GAsol,:),'facecolor','red')


% get optimum microphone positions
% (dimensions L x 3: [x-ccordinate,y-coordinate, z-coordinate])
mic_opt = [mic_positions.x(GAsol,:), mic_positions.y(GAsol,:), ...
                  mic_positions.z(GAsol,:)];
```

- Second Part - Main Function

```
%-------------------------------------------------------------------------
%this script runs the room impulse response (RIR) simulation for the NAO robot
%-------------------------------------------------------------------------
clear all
close all
clc

%-------------------------------------------------------------------------
% Initialisation
%-------------------------------------------------------------------------
%add path of MCRoomSim toolbox
```

```
addpath('MCRoomSim_v214');
%add path to internal functions
addpath('internal');

% General parameters
target_fs = 10e3;  % Target sample rate for both HRIR and ROOM
ism_fs = 48e3;       % Room simulation sample rate (must be minimum 44100)
isFree = false;        % Free field (true) or room (false)
%----------------------------------------------------------------------------

%----------------------------------------------------------------------------
% Load and resample NAO HRIR data
%----------------------------------------------------------------------------
fprintf('Loading and processing NAO-HRIR data...\n');
%Here you must load the so-called EARO object which contains the set of
%HRIR (head-related impulse responses) or RIR (room impulse responses)
%corresponding to the optimal microphone positions
L = '4'; %number of microphones
src_conf = 'hor'; %source configuration in the optimization
type = 'RIR';  %Simulate RIR or HRTF
load(['data\EARO_' L '_' src_conf '.mat']); %
%load(['data\EARO_NAO_4.mat']);
nao=nao.resampleData(target_fs);
N=floor(sqrt(size(nao.data,1))-1); %SH order
%----------------------------------------------------------------------------

%----------------------------------------------------------------------------
% Set room properties and source positions
%----------------------------------------------------------------------------
disp('Performing room simulation...');
%room dimensions (x,y,z)
roomDims=[5,5,3];
%position of receiver (NAO)
recPos=roomDims/2;

```

```
%source positions in spherical coordinates (azimuth, elevation, radius)
%azimuth (=src_az): 0 degrees (right side) -> 180 degrees (left side)
%elevation (=src_el): 90 degrees (up) -> 0 degrees (front) -> -90 degrees (bottom)
%radius (src_r) in meter
if strcmp(type,'RIR')

    %source positions

    if strcmp(src_conf,'hor') %horizontal belt

        src_az = (180:-30:-150)*pi/180;
        src_el = repmat(0,size(src_az))*pi/180;
        src_r = repmat(1.5,size(src_az));

    elseif strcmp(src_conf,'all')

        src_el = (45:-22.5:-45)*pi/180;
        src_az = repmat(90,size(src_el))*pi/180;
        src_r = repmat(1.5,size(src_el));

    elseif strcmp(src_conf,'top')

        src_az = (135:-22.5:45)*pi/180;
        src_el = pi/4-((src_az-pi/2)/sqrt(pi)).^2;
        src_r = repmat(1.5,size(src_az));

    else
        error('Invalid choice of the variable src_conf\n');
    end

elseif strcmp(type,'HRIR')

    %positions for beamform design

    if strcmp(src_conf,'hor') %horizontal belt
```

```

        src_az = (180:-5:-175)*pi/180;
        src_el = repmat(0,size(src_az))*pi/180;
        src_r = repmat(1.5,size(src_az));

    elseif strcmp(src_conf,'all')

        src_el = (90:-5:-90)*pi/180;
        src_az = repmat(90,size(src_el))*pi/180;
        src_r = repmat(1.5,size(src_el));

    elseif strcmp(src_conf,'top')

        src_az = (180:-5:0)*pi/180;
        src_el = pi/4-((src_az-pi/2)/sqrt(pi)).^2;
        src_r = repmat(1.5,size(src_az));

    else
        error('Invalid choice of the variable src_conf\n');
    end

else
    error('Invalid choice of the variable type\n');
end

%get source positions in cartesian coordinates
[x, y, z]=sph2cart(src_az,src_el,src_r);
%merge all sources
SrcPos = [x.', y.', z.'];
%add receiver position to all source positions
SrcPos = SrcPos + repmat(recPos,size(SrcPos,1),1);

%set source type
srcType = 'omnidirectional';
freqAbs = [125, 250, 500, 1e3, 2e3, 4e3];
```

```
%with alphaAbs (see documentation of mcroomsim tool) you can define the
%reflection coefficients of each wall (thus, the reverberation time is
%defined here implicitly)
%smaller values mean more reflections

%---------------------------------------------------------------------------
% Calculate reflection coefficients
%---------------------------------------------------------------------------

if strcmp(type,'RIR')

    T=500; %reverberation time in miliseconds
    Volume = prod(roomDims);
    Area = 2*(roomDims(1,1)*roomDims(1,2)+roomDims(1,1)*roomDims(1,3) ...
        +roomDims(1,2)*roomDims(1,3));
    alpha = 163*Volume/(Area*T);    %Reflection coefficient
    absMatrix = repmat(alpha, 6, 6);%same coefficients for all frequencies and walls
    savestr = [type '_' L '_' src_conf '_' num2str(T) 'ms']; %Output file name

elseif strcmp(type,'HRIR')

    absMatrix = ones(6,6); %->no reflection
    savestr = [type '_' L '_' src_conf]; %Output file name

else
    error('Invalid choice of the variable type\n');
end



%---------------------------------------------------------------------------

%---------------------------------------------------------------------------
% Run the ISM simulation (requires MCRoomSim)
%
```

```
% The simulated HRIRs will be contained in the variable 'earp'
%---------------------------------------------------------------------------
% absMatrix = [];
if isempty(freqAbs) || isempty(absMatrix)
    Room=SetupRoom('Dim',roomDims);
else
    Room=SetupRoom('Dim',roomDims,'Freq',freqAbs,'Absorption',absMatrix);
end
Receivers=AddReceiver('Location',recPos,'Type','sphharm', ...
            'MaxOrder',N,'NFComp',false, 'Orientation',[0,0,0]);

Sources = [];
for idx_source = 1:size(SrcPos,1)
    Sources=AddSource(Sources,'Location',SrcPos(idx_source,:),
               'Type',srcType,'Orientation',[rad2deg(pi+src_az(idx_source)),0,0]);
end
Options=MCRoomSimOptions('SimDirect',true,'SimSpec', ...
               ~isFree,'SimDiff',~isFree,'Fs',ism_fs);
PlotSimSetup(Sources,Receivers,Room);
irArray=RunMCRoomSim(Sources,Receivers,Room,Options);
P=MCRoomPerm(N)';
C=SHc2r(N)';

if ~iscell(irArray)
    irArray_save = irArray; clear irArray;
    irArray{1}=irArray_save;
end

for idx_source = 1:size(irArray,2)
    %get set of irs for current source position
    current_anm=C*P*irArray{idx_source}.';

    % Resample and fft anm, trim negative freqs
    gComDiv = gcd(ism_fs, target_fs);
    p = target_fs / gComDiv;
    q = ism_fs / gComDiv;
    current_anm = resample(current_anm.',p,q); % mind the transpose
    nFFT = 2^nextpow2(size(current_anm,1));
    current_anm = fft(current_anm, nFFT);
    fVec = linspace(0,target_fs,nFFT);
    current_anm = current_anm(1:(nFFT/2)+1,:);
    fVec = fVec(1:(nFFT/2)+1);

    anm{idx_source} = current_anm;
end

% Do the same for HRIRs
nao = nao.toFreq(nFFT);
nao.data = nao.data(:,1:(nFFT/2)+1,:);

% Send anm's back to space domain in Nao's source grid
Y=shMatrix(N, nao.sourceGrid.elevation, nao.sourceGrid.azimuth);
for idx_source = 1:size(irArray,2)
    A{idx_source}=anm{idx_source}*Y;
end

%iterate over all source positions
for idx_source = 1:size(irArray,2)
    % Iterate through Nao's ears (mics)
    for nn=1:size(nao.data,3)
        fprintf('Rendering response and audio for mic #%d ...\n',nn);
        thisEar = squeeze(nao.data(:,:,nn));
        aW = repmat(nao.sourceGrid.quadWeight,(nFFT/2)+1,1);
        % pressure at ear in freq
        pk = sum(thisEar.*A{idx_source}.'.*aW.');
        pk = [pk,fliplr(conj(pk(2:end-1)))];
        % pressure at ear in time (response only)
        earp(idx_source,:,nn) = ifft(pk,'symmetric');
    end
end
```

```
%----------------------------------------------------------------------------

%----------------------------------------------------------------------------
% Create struct for simulated RIRs and save it in folder ./saves/
%----------------------------------------------------------------------------
imp_resp = permute(earp,[2,3,1]);
fs_RIR = target_fs;

source_directions.phi = src_az/pi*180;%azimuth
source_directions.theta = src_el/pi*180 + 90;%elevation
source_directions.r = src_r;%radius

[x y z] = sph2cart(nao.micGrid.azimuth, nao.micGrid.elevation - pi/2, nao.micGrid.r);

mic_positions.x = x;
mic_positions.y = y;
mic_positions.z = z;

save(['saves\' savestr '.mat'],'imp_resp','fs_RIR','source_directions', 'mic_position
```

- Third Part - Main Function

```
clc;
clear;
close all;

%--------------------------------------------------------------------
% Initialization
%--------------------------------------------------------------------

cfg = [];
sig = [];
flt = [];
```

```


% Path of filterbank m-files (required if FSB beamformer design is applied)
addpath(genpath('filterbank'));
% Path of beamformer m-files (required if FSB beamformer design is applied)
addpath(genpath('Generalized_RLSFI_BF'));

% Parameters for FSB beamformer design
% constraint of the white noise gain (WNG) in dB
% Note that the maximum possible value
% for the WNG is 10*log10(N_transducer),
% which corresponds to delay-and-sum beamforming.
cfg.WNG_dB = -15;

%-------------------------------------------------------------------
% Set Acoustical scenario parameters (parameters are stored in cfg
% structure
%-------------------------------------------------------------------

cfg.nmic = 12; %Number of microphones
cfg.src_conf = 'hor'; %Source configuration
cfg.RIRcond = [ num2str(cfg.nmic) '_' cfg.src_conf]; %For NAO, append `NAO_'
savestr = ['saves/result_' cfg.RIRcond]; %Results output file name

% Parameters for general beamformer design
% indicates that HRTFs are used for designing both
% the robust FSB and the MVDR beamformer
cfg.design = 'hrtf';

% cfg.design = 'freefield';
% Alternatively, free-field steering vectors can be used (not required)

if strcmp(cfg.design,'hrtf')
     % Path to protoype HRTFs which are required for beamformer design
    cfg.path_hrirs = ['Data/RIRs/HRIR_' cfg.RIRcond '.mat'];
```

```
end

% Initialize final result variables

final = [];
final.snr_in = 0.0;
final.snr_out = 0.0;
final.snr_gain = 0.0;
final.fwsegsnr_mics_in = 0.0;
final.fwsegsnr_mics_out = 0.0;
final.fwsegsnr_mics_gain = 0.0;
final.fwsegsnr_in = 0.0;
final.fwsegsnr_out = 0.0;
final.fwsegsnr_gain = 0.0;
% Only in Linux
final.PESQscore_in = 0.0;
final.PESQscore_out = 0.0;
final.PESQscore_gain = 0.0;
final.ASRscore_in = 0.0;
final.ASRscore_out = 0.0;
final.ASRscore_gain = 0.0;

final.BPattern = zeros(46,72);
final.WNG_real = zeros(1,46);
final.WNG_theoretical = zeros(1,46);
final.BF_lookDir_abs_log = zeros(46,1);

% Indices of desired sources
for i=4
% Indices of interfering sources relative to the desired
for k=[-1 1]


cfg.interf_idx = i+k; % Select interfering source index

```

```
if cfg.interf_idx == 0
    cfg.interf_idx = 12;
elseif cfg.interf_idx == 13
    cfg.interf_idx = 1;
end

% Beamformer look direction (in degrees)
% Azimuth angle: 0 (right), 90 (front), 180 (left) and 270 (back)
% Elevation angle: 0 (up), 90 (middle) to 180 (down)
cfg.des_look_dir.azimuth = 180 - 30*(i-1);
cfg.des_look_dir.elevation = 90;

% Inside SetAcousticScenario: # of sources in experiment (desired+interferers),
% Sources sph. coordinates, mic channels, noise.

[cfg] = SetAcousticScenario(cfg);

%--------------------------------------------------------------------
% Create microphone signals (signals are stored in sig structure, RIRs in
% flt structure
%--------------------------------------------------------------------
% Nothing to be modified here
[cfg,sig,flt] = LoadMicInputs(cfg,sig,flt);

%--------------------------------------------------------------------
% Perform beamformer design (of robust FSB)
% If MVDR beamforming is applied, this is not necessary
%--------------------------------------------------------------------
[flt.w, BPattern, WNG_real, WNG_theoretical, BF_lookDir_abs_log] ...
= main_robust_FSB(cfg);


%For one look direction use `/2.0'
final.BPattern = final.BPattern + circshift(BPattern,36-6*(i-1),2)/24.0;
final.WNG_real = final.WNG_real + WNG_real/24.0;
final.WNG_theoretical = final.WNG_theoretical + WNG_theoretical/24.0;
final.BF_lookDir_abs_log = final.BF_lookDir_abs_log + BF_lookDir_abs_log/24.0;


%-------------------------------------------------------------------
% Create microphone signals (signals are stored in sig structure) at
% beamformer output (only use for FSB design).
%-------------------------------------------------------------------
sig.y = zeros(length(sig.x)+length(flt.w)-1,1);
sig.ySrc = zeros(length(sig.x)+length(flt.w)-1,cfg.nsrc);
for idx_channels = 1:cfg.nmic
    sig.y  = sig.y + fftfilt(flt.w(:,idx_channels), ...
    [sig.x(:,idx_channels); zeros(length(flt.w)-1,1)]);

    for idx_sources = 1:cfg.nsrc
        sig.ySrc(:,idx_sources) = sig.ySrc(:,idx_sources) + ...
        fftfilt(flt.w(:,idx_channels), ...
        [sig.xSrc(:,idx_channels,idx_sources); zeros(length(flt.w)-1,1)]);
    end
end

%-------------------------------------------------------------------
% Create microphone signals (signals are stored in sig structure) at
% MVDR beamformer output. The signals are created with the myMVDR
% function, which also contains the MVDR design.
%-------------------------------------------------------------------
% Index used to select HRTF desired component
%[sig flt.w] = myMVDR_new(cfg, sig, i);

%-------------------------------------------------------------------
% Evaluation regarding the fwsegsnr, PESQ score and ASR score
%-------------------------------------------------------------------

% Average results for all reference microphones
 for j=1:cfg.nmic

 cfg.ref = j; % Select reference microphone channel

% Signal dependent metrics evaluation
% For windows switch to 1, 1, 0, 0
[sig, cfg, res] = Evaluation(sig, cfg, 1, 1, 1, 1);

% For one look direction use `/(cfg.nmic*2.0)'
final.snr_in = final.snr_in + res.snr_in/(cfg.nmic*24.0);
final.snr_out = final.snr_out + res.snr_out/(cfg.nmic*24.0);
final.snr_gain = final.snr_gain + res.snr_gain/(cfg.nmic*24.0);

final.fwsegsnr_mics_in = final.fwsegsnr_mics_in + ...
res.fwsegsnr_mics_in/(cfg.nmic*24.0);
final.fwsegsnr_mics_out = final.fwsegsnr_mics_out + ...
res.fwsegsnr_mics_out/(cfg.nmic*24.0);
final.fwsegsnr_mics_gain = final.fwsegsnr_mics_gain + ...
res.fwsegsnr_mics_gain/(cfg.nmic*24.0);

final.fwsegsnr_in = final.fwsegsnr_in + res.fwsegsnr_in/(cfg.nmic*24.0);
final.fwsegsnr_out = final.fwsegsnr_out + res.fwsegsnr_out/(cfg.nmic*24.0);
final.fwsegsnr_gain = final.fwsegsnr_gain + res.fwsegsnr_gain/(cfg.nmic*24.0);

% Only in Linux
final.PESQscore_in = final.PESQscore_in + res.PESQscore_in/(cfg.nmic*24.0);
final.PESQscore_out = final.PESQscore_out + res.PESQscore_out/(cfg.nmic*24.0);
final.PESQscore_gain = final.PESQscore_gain + res.PESQscore_gain/(cfg.nmic*24.0);

final.ASRscore_in = final.ASRscore_in + res.ASRscore_in/(cfg.nmic*24.0);
final.ASRscore_out = final.ASRscore_out + res.ASRscore_out/(cfg.nmic*24.0);
final.ASRscore_gain = final.ASRscore_gain + res.ASRscore_gain/(cfg.nmic*24.0);

end
end
end
```

```

%-------------------------------------------------------------------
% Plot signal independent metrics
%-------------------------------------------------------------------

figure

subplot(2,2,1:2);
imagesc((300:100:4800),(180:-5:-175),final.BPattern.')
xlabel('Frequency in Hz')
ylabel('Relative angle')
title('Beampattern (FIR approx)')
colorbar

subplot(2,2,3);
semilogx((300:100:4800),final.BF_lookDir_abs_log.','LineWidth',2);
axis([10^(log10(cfg.frange(1))) 10^(log10(cfg.frange(end))) -20 0.1]);
grid on;
xlabel('Frequency in Hz');
ylabel('Response in dB')
title(['Response in look direction'])


subplot(2,2,4)
semilogx((300:100:4800),final.WNG_real,'LineWidth',2); hold on;
semilogx((300:100:4800),final.WNG_theoretical,'--r','LineWidth',2);
legend('real WNG','theoretical WNG','Location','best')
grid on
xlim([10^(log10(cfg.frange(1))) 10^(cfg.frange(end))]);
xlabel ('Frequency in Hz')
ylabel ('White Noise Gain in dB')
ylim([-20, 5])



```

```
%-----------------------------------------------------------------
% Save results (if needed)
%-----------------------------------------------------------------
save([savestr '.mat'], 'final');
```

Bibliography

[1] Justin McCurry, "Japanese bank introduces robot workers to deal with customers in branches." `http://www.theguardian.com/world/2015/feb/04/japanese-bank-introduces-robot-workers-to-deal-with-customers-in-br` 2015.

[2] Aldebaran Robotics, "NAO NEXT Gen H25 Datasheet," Dec 2011.

[3] H. L. Van Trees, *Detection, Estimation, and Modulation Theory.* Wiley, 2 ed., 2013.

[4] U.-H. Kim and H. G. Okuno, "Improved binaural sound localization and tracking for unknown time-varying number of speakers," *Advanced Robotics*, vol. 27, pp. 1161–1173, Jul 2013.

[5] T. Takahashi, K. Nakadai, K. Komatani, T. Ogata, and H. Okuno, "Improvement in listening capability for humanoid robot HRP-2," in *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 470–475, May 2010.

[6] K. Nakamura, K. Nakadai, and H. G. Okuno, "A real-time super-resolution robot audition system that improves the robustness of simultaneous speech recognition," *Advanced Robotics*, vol. 27, pp. 933–945, May 2013.

[7] M. Maazaoui, K. Abed-Meraim, and Y. Grenier, "Adaptive blind source separation with hrtfs beamforming preprocessing," in *2012 IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*, pp. 269–272, Jun 2012.

[8] V. Murino, A. Trucco, and C. S. Regazzoni, "Synthesis of unequally spaced arrays by simulated annealing," *IEEE Transactions on Signal Processing*, vol. 44, pp. 119–122, Jan 1996.

[9] D. Pearson, S. U. Pillai, and Y. Lee, "An algorithm for near-optimal placement of sensor elements," *IEEE Transactions on Information Theory*, vol. 36, pp. 1280–1284, Nov 1990.

[10] S. R. Tuladhar and J. R. Buck, "Optimum array design to maximize fisher information for bearing estimation," *The Journal of the Acoustical Society of America*, vol. 130, no. 5, pp. 2797–2806, 2011.

[11] S. Joshi and S. Boyd

[12] A. Skaf and P. Danes, "Optimal positioning of a binaural sensor on a humanoid head for sound source localization," in *2011 IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 165–170, Oct 2011.

[13] V. Tourbabin and B. Rafaely, "Theoretical framework for the optimization of microphone array configuration for humanoid robot audition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, pp. 1803–1814, Dec 2014.

[14] E. Mabande, A. Schad, and W. Kellermann, "Design of robust superdirective beamformers as a convex optimization problem," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 77–80, Apr 2009.

[15] H. Barfuss, C. Huemmer, G. Lamani, A. Schwarz, and W. Kellermann, "HRTF-based robust least-squares frequency-invariant beamforming," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Oct 2015.

[16] L. C. Parra, "Steerable frequency-invariant beamforming for arbitrary arrays," *Journal of the Acoustical Society of America*, vol. 119, no. 6, pp. 3839–3847, 2006.

[17] J. Bitzer and K. Simmer, “Superdirective microphone arrays,” in *Microphone Arrays* (M. Brandstein and D. Ward, eds.), Digital Signal Processing, pp. 19–38, Springer Berlin Heidelberg, 2001.

[18] CVX, “version 2.1,” 2015.

[19] S. Doclo, W. Kellermann, S. Makino, and S. Nordholm, “Multichannel signal enhancement algorithms for assisted listening devices: Exploiting spatial diversity using multiple microphones,” *IEEE Signal Processing Magazine*, vol. 32, pp. 18–30, Mar 2015.

[20] L. J. Griffiths and C. Jim, “An alternative approach to linearly constrained adaptive beamforming,” *IEEE Transactions on Antennas and Propagation*, vol. 30, pp. 27–34, Jan 1982.

[21]

[22] O. Roy and M. Vetterli, “The effective rank: A measure of effective dimensionality,” in *2007 European Signal Processing Conference (EUSIPCO)*, pp. 606–610, Sept 2007.

[23] G. Seber, *A Matrix Handbook for Statisticians*. Wiley Series in Probability and Mathematical Statistics, Hoboken, NJ: John Wiley and Sons, Inc., 2008.

[24] W. Yang, J. Gibson, and T. He, “Coefficient rate and lossy source coding,” *IEEE Transactions on Information Theory*, vol. 51, pp. 381–386, Jan 2005.

[25] D. S. Weile and E. Michielssen, “Genetic algorithm optimization applied to electromagnetics: a review,” *IEEE Transactions on Antennas and Propagation*, vol. 45, pp. 343–353, Mar 1997.

[26] D. Huggins-Daines, M. Kumar, A. Chan, A. Black, M. Ravishankar, and A. Rudnicky, “Pocketsphinx: A free, real-time continuous speech recognition system for

hand-held devices," in *2006 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, p. I, May 2006.

[27] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audiovisual corpus for speech perception and automatic speech recognition," *The Journal of The Acoustic Society of America (JASA)*, vol. 120, pp. 2421–2424, Nov 2006.

[28] H. Christensen, J. Barker, N. Ma, and P. Green, "The CHiME corpus: a resource and a challenge for computational hearing in multisource environments," in *2010 Spoken Language Processing for All (INTERSPEECH)*, pp. 1918–1921, Sep 2010.

[29] J. Blauert and N. Xiang, *Acoustics for Engineers.* Springer Berlin Heidelberg, 2008.

[30] MATLAB, "version 8.5 (R2015a) documentation," 2015.

[31] MCRoomSim, "version 2.14," 2015.