NANYANG
TECHNOLOGICAL
UNIVERSITY
SINGAPORE

PROJECT TITLE GOES HERE

YOUR NAME

YOUR SCHOOL

2019

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Keywords: Blind source separation, Independent component analysis, Independent vector analysis, Sparse component analysis, Automatic speech recognition, Android application development.

Acknowledgement

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

This project would not be possible without the support and funding of the CN Yang Scholars Programme.

Table of Contents

Nomenclature

$\boldsymbol{A}$	Mixing matrix
$\boldsymbol{A}_k$	kth mixing filter in time domain
$\boldsymbol{S}$	Frequency-domain representation of the source signals
$\boldsymbol{s}$	Source signals matrix
f_s	Sampling rate
j	Number of channels in an arbitrary audio data
K	Mixing filter order
M	Number of sensors or microphones
m	Sensor or channel index
N	Number of sources
n	Source index
s_i	Source signal vector of source i
x_i	Observed signal vector of sensor i
GUI	Graphic user interface
IDE	Integrated development environment
PCM	Pulse code modulation
UI	User interface
XML	Extensible Markup Language
ZCA	Zero-phase component analysis

Lists of Figures and Listings

List of Tables

Chapter 1 Introduction

1.1 Motivation

Blind source separation is the process of estimating individual source signals from their mixture with little to no knowledge about the sources and the mixing process. Combined with automatic speech recognition, blind source separation is a powerful tool that has potential as an assistive technology for people with hearing difficulties or as a productivity technology.

However, most blind source separation algorithms can only be carried out with two or more sensors, and demand substantial processing power and memory. As such, in the past, blind source separation implementation was not possible on mobile devices which were equipped with only one microphone and had a relatively low processing power. In recent years, however, mobile devices have seen a dramatic improvement in memory capacity and processing power, approaching that of a laptop or desktop computer. Most mobile devices today are also equipped with at least two microphones, albeit mainly for noise cancellation purposes. These recent improvements have opened up the opportunity for blind source implementation on mobile devices, which would make blind source separation much more accessible as a technology.

1.2 Objectives

This project aims to develop an Android application, implementing blind source separation to improve the performance of automatic speech recognition for recordings with multiple speech sources.

1.3 Scope of Work

The scope of this project includes the development of the Android application with an implementation of blind source separation algorithms using independent component analysis, independent vector analysis, and sparse component analysis for Android mobile devices equipped with at least two built-in microphones. The scope further includes the integration of an automatic speech recognition interface with the blind source separation output and the related user interface.

1.4 Organisation of Report

This report is organised into six chapters. Chapter 2 will briefly review audio signal encoding then discuss the blind source separation problem, existing blind source separation algorithms, required preprocessing, and automatic speech recognition. Chapter 3 will discuss the development of the Android application. Chapter 4 will discuss the simulations and metrics needed for performance evaluation, whose results are presented in Chapter 5. Finally, Chapter 6 will conclude the report as well as discuss possible future works for the project.

Chapter 2 Literature Review

This chapter discusses the digital representation of an audio signal, the preprocessing required, the blind source separation (BSS) problem, the BSS algorithms used in this project, and the automatic speech recognition (ASR).

2.1 Digital Representation of Audio Signal

Digital audio signal is the digital representation of a sound. When a microphone picks up a sound, a continuous-time continuous-amplitude analog electrical signal which represents the sound is generated. The analog signal is then converted into a discrete-time discrete-amplitude digital signal using a method known as pulse code modulation (PCM).

$$\begin{aligned}
\boldsymbol{x} &= \begin{bmatrix} — & x_0 & — \\ — & x_1 & — \\ & \vdots & \\ — & x_{j-1} & — \end{bmatrix} = \begin{bmatrix} | & | & & | \\ \boldsymbol{x}(0) & \boldsymbol{x}(1) & \cdots & \boldsymbol{x}(T-1) \\ | & | & & | \end{bmatrix} \\
&= \begin{bmatrix} x_0(0) & x_0(1) & x_0(2) & \cdots & x_0(T-1) \\ x_1(0) & x_1(1) & x_1(2) & \cdots & x_0(T-1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{j-1}(0) & x_{j-1}(1) & x_{j-1}(2) & \cdots & x_{j-1}(T-1) \end{bmatrix}
\end{aligned}$$

is stored in the LPCM format as

$$\begin{bmatrix} \boldsymbol{x}(0)^T & \boldsymbol{x}(1)^T & \cdots & \boldsymbol{x}(T-1)^T \end{bmatrix}$$

where $\boldsymbol{x}(\tau)^T = \begin{bmatrix} x_0(\tau) & x_1(\tau) & \cdots & x_{j-1}(\tau) \end{bmatrix}$.

While an LPCM data can be stored as a raw audio file with extension `.pcm` or `.raw`, the raw audio file lacks additional information needed to play back the audio file correctly such as the bit depth, the endianness, the sample rate and the number of channels. As such, digital audio is more commonly stored as a waveform audio file format, much more commonly known as WAV or WAVE, with the extension `.wav`. The WAV file is a type of Resource Interchange File Format (RIFF) which stores data in chunks and contains the information regarding the bit depth, the sampling rate, and the number of channels. The summary of the structure of a WAV file is shown in Fig 2-1. Except for the chunk and format identifiers, the rest of a WAV file is in little endian [1]. Note that some details that are not directly relevant to this project may be omitted for brevity.

Offset (bytes)	Field	Size (bytes)	Endian	Content
`RIFF` header				
0	`ChunkID`	4	big	`"RIFF"`
4	`ChunkSize`	4	little	The size in bytes of the entire file excluding `ChunkID` and `ChunkSize`, i.e. `36 + SubChunk2Size`
8	`Format`	4	big	`"WAVE"`
`fmt` subchunk				
12	`Subchunk1ID`	4	big	`"fmt "`
16	`Subchunk1Size`	4	little	The size in bytes of the remaining of this subchunk i.e. `16` for PCM
20	`AudioFormat`	2	little	`1` for PCM
22	`NumChannels`	2	little	Number of channels, e.g. `2` for stereo
24	`SampleRate`	4	little	Sample rate in Hz, e.g. `16000` for 16 000 Hz
28	`ByteRate`	4	little	Number of bytes per sample per channel, i.e. `SampleRate * NumChannels * BitsPerSample / 8`
32	`BlockAlign`	2	little	Number of bytes per sample of all channels combined, i.e. `NumChannels * BitsPerSample / 8`
34	`BitsPerSample`	2	little	Audio bit depth, e.g. `16` for 16-bit PCM
`data` subchunk				
36	`Subchunk2ID`	4	big	`"data"`
40	`Subchunk2Size`	4	little	The size in bytes of the remaining of this subchunk, i.e. the size in bytes of `data`.
44	`data`	*	little	The actual LPCM data

Figure 2-1: Structure of a WAV file containing LPCM audio data

Adapted from [2]

2.2 Frequency-domain Preprocessing

Before we enter the discussion of blind source separation (BSS), it is necessary that we first discuss the tools required to preprocess the audio signal before attempting signal separation.

2.2.1 Short-time Fourier Transform

As discussed in §2.1, raw audio signals are recorded in the time domain. However, the BSS problem is typically tackled in the frequency domain given its relative simplicity (see §2.3). As such, a tool to transform an audio signal from the time domain to the frequency domain is needed. This is commonly done via short-time Fourier transform.

2.2.1.1 Spectrum analysis and synthesis window

A window function, loosely defined, is a function that is maximum around the middle of its domain, decreases smoothly in value as it approaches the extreme ends of the domain, and is zero outside of its domain. By windowing a frame, that is multiplying each element in a frame by the value of the window function at that point, the discontinuity thus spectral leakage is reduced. The various types of window functions and their properties are reviewed in [3]. For this project, we use the periodic Hamming window defined by

$$w(\tau) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi\tau}{R}\right) & 0 \le \tau < R, \\ 0, & \text{otherwise}, \end{cases} \tag{2.1}$$

with 75% overlap.

When a window is applied in the forward STFT, the window is called an analysis window. When it is applied in the inverse STFT, the window is called a synthesis window. Similar to the analysis window, a synthesis window helps to suppress any audible discontinuities due to the processing in the frequency domain.

2.2.2 Whitening Transformation

To find $\boldsymbol{Q}_D$, first, the covariance matrix $\boldsymbol{C}$ of X is calculated using

$$\boldsymbol{C} = \operatorname{cov}(\boldsymbol{X}) = \operatorname{E}\left[\boldsymbol{X}\boldsymbol{X}^H\right] - \operatorname{E}[\boldsymbol{X}]\operatorname{E}\left[\boldsymbol{X}^H\right] = \operatorname{E}\left[\boldsymbol{X}\boldsymbol{X}^H\right], \tag{2.2}$$

where H denotes the Hermitian transpose operator. In practice, the covariance matrix is approximated by

$$\boldsymbol{C} \approx \frac{\boldsymbol{X}\boldsymbol{X}^H}{T}. \tag{2.3}$$

A singular value decomposition (SVD) of $\boldsymbol{C}$ is then calculated such that

$$\boldsymbol{C} = \boldsymbol{U}\Sigma\boldsymbol{V}^H. \tag{2.4}$$

However, since a covariance matrix is a symmetric positive semi-definite matrix, we have $\boldsymbol{V} = \boldsymbol{U}$ thus $\boldsymbol{C} = \boldsymbol{U}\Sigma\boldsymbol{U}^H$. The decorrelation transform is given by $\boldsymbol{Q}_D = \boldsymbol{U}^H$ since

$$\operatorname{cov}(\boldsymbol{Q}_D\boldsymbol{X}) = \mathrm{E}\left[(\boldsymbol{U}^H\boldsymbol{X})(\boldsymbol{U}^H\boldsymbol{X})^H\right] \approx \frac{\boldsymbol{U}^H\boldsymbol{X}\boldsymbol{X}^H\boldsymbol{U}}{T} = \boldsymbol{U}^H\boldsymbol{C}\boldsymbol{U} = \Sigma \tag{2.5}$$

which is a diagonal matrix. The scaling matrix is then given by $\boldsymbol{Q}_S = \Sigma^{-\frac{1}{2}}$ since

$$\begin{aligned}\operatorname{cov}(\boldsymbol{Q}_S\boldsymbol{Q}_D\boldsymbol{X}) &= \mathrm{E}\left[(\Sigma^{-\frac{1}{2}}\boldsymbol{U}^H\boldsymbol{X})(\Sigma^{-\frac{1}{2}}\boldsymbol{U}^H\boldsymbol{X})^H\right] \approx \frac{\Sigma^{-\frac{1}{2}}\boldsymbol{U}^H\boldsymbol{X}\boldsymbol{X}^H\boldsymbol{U}\Sigma^{-\frac{1}{2}}}{T} \\ &= \Sigma^{-\frac{1}{2}}\boldsymbol{U}^H\boldsymbol{C}\boldsymbol{U}\Sigma^{-\frac{1}{2}} = \Sigma^{-\frac{1}{2}}\Sigma\Sigma^{-\frac{1}{2}} = \boldsymbol{I}.\end{aligned} \tag{2.6}$$

2.3 Blind Source Separation

Blind source separation is the process of retrieving estimates of individual source signals from a mixed signal with little to no information about the sources and the mixing process. The BSS problem is formally stated as follows.

A set of N source signals $\boldsymbol{s}(\tau) = [s_0(\tau), \ldots, s_{N-1}(\tau)]^T \in \mathbb{R}^{N\times T_t}$ over the samples $0 \le \tau < T_t$, such that $s_n = [s_n(\tau = 0), s_n(1), \ldots, s_n(T_t - 1)]$ for $0 \le n < N$, undergoes mixing whose result is observed by an array of M sensors (in this project's context, microphones). The observed signals are represented by $\boldsymbol{x}(\tau) = [x_0(\tau), \ldots, x_{M-1}(\tau)]^T$, such that $x_m = [x_m(\tau = 0), x_m(1), \ldots, x_m(T_t - 1)] \in \mathbb{R}^{M\times T}$ for $0 \le m < M$. In real acoustic environments, the sources are mixed convolutively such that

$$\boldsymbol{x}(\tau) = \sum_{k=0}^{K-1} \boldsymbol{A}_k \boldsymbol{s}(\tau - k) \tag{2.7}$$

where $\boldsymbol{A}_k \in \mathbb{R}^{M\times N}$ with K often assumed, in practice, to be some finite positive integer. This means that $\boldsymbol{x}(\tau)$ can be thought of as the output signal of a finite impulse response filter with input $\boldsymbol{s}(\tau)$ and $\boldsymbol{A}_k$ representing the coefficients of the kth-order filter.

However, solving a convolutive system in the time domain is difficult and requires significant computational power. To simplify the problem, BSS problems are often transformed into the frequency domain such that

$$\boldsymbol{X}(\omega, t) = \boldsymbol{A}(\omega)\boldsymbol{S}(\omega, t) \tag{2.8}$$

where $\boldsymbol{X}(\omega, t) \in \mathbb{C}^{M\times T_f}$ and $\boldsymbol{S}(\omega, t) \in \mathbb{C}^{N\times T_f}$ are the frequency-domain representation of the observed signals and the source signals respectively [4].

Chapter 3 Application Development

This chapter discusses the implementation of the mobile application implementing BSS and ASR for an Android device.

A crucial part of an Android application comprises of ‘activities’. An activity is the point of interaction between the user and the application. An activity represents a single screen and its GUI. The GUI of each activity is called the ‘layout’ and each element in a layout is called a ‘view’. The interaction between the user and a view invokes a specific method corresponding to the user’s action. An activity may be created when the user launches the application from the home screen or when called by another activity. Upon being created, the `onCreate()` method is invoked. The `onCreate()` method initializes all essential components of the activity, such as setting the layout using `setContentView()`, and initializing views. Once `onCreate()` finishes, `onStart()` is automatically called and the activity becomes visible to the user to interact [5].

3.1 `MainActivity` Activity

The `MainActivity` activity is the launcher activity. This activity is responsible for the recording of the observed mixture as well as other auxiliary functions such as playback and saving. The GUI of this activity is shown in Figure 3-1.

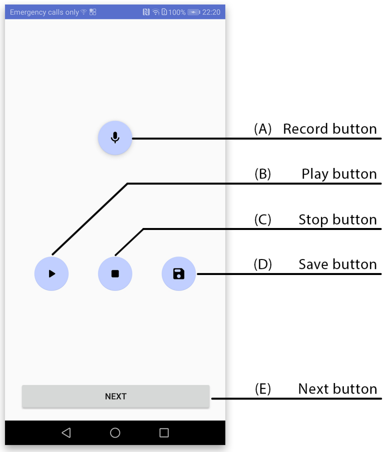


Figure 3-1: GUI of `MainActivity` activity

3.1.1 Requesting Permissions

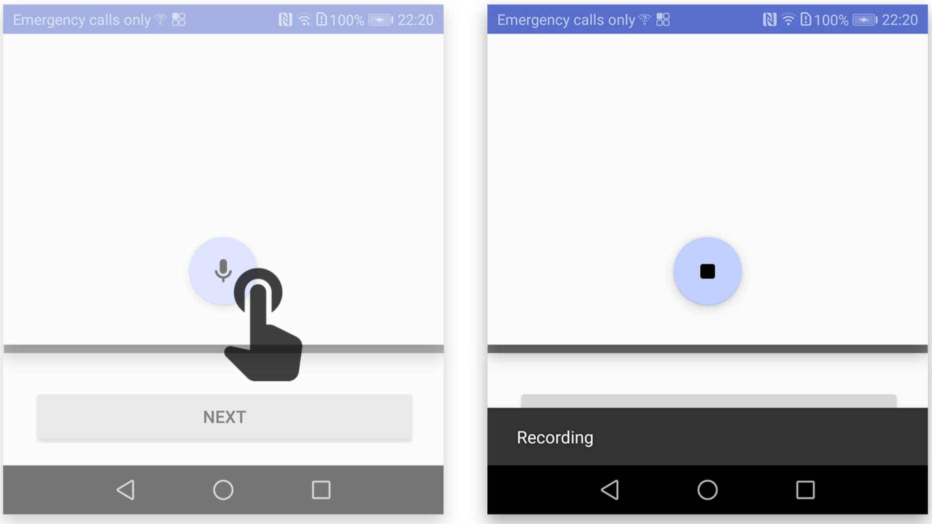
This application requires the use of audio recording, reading and writing files from and to external storage, and internet access. These functionalities are considered sensitive. Thus the permissions to perform any process with such functionalities must be declared in the manifest as shown in Figure 3-2.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    <!-- Package declaration...  -->

    <uses-permission android:name="android.permission.RECORD_AUDIO" />
    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />

    <!-- The rest of the manifest... -->
</manifest>
```

Figure 3-2: Declaration of the permissions required in the Android manifest

3.1.2 Recording of the Observed Mixture

As the user presses the record button (A), the method `startRecording()` is invoked and the record button's icon changes to a stop symbol as shown in Figure 3-3.



Recording

Figure 3-3: Record button icon changes from a microphone to a stop symbol

When `startRecording()` is invoked, an `AudioRecord` object is initialized and a background thread `RecordingRunnable()` is created so that the recording process does not block the main thread (also known as the UI thread). The `AudioSource` is set to `DEFAULT` to access the main microphones. The sampling rate `SAMPLING_RATE_IN_HZ` is set to `16000` which is the minimum sampling rate recommended by the Google Speech-to-text API [6].

```
private void startRecording() {
    recorder = new AudioRecord(
        MediaRecorder.AudioSource.DEFAULT,
        SAMPLING_RATE_IN_HZ,
        CHANNEL_CONFIG,
        AUDIO_FORMAT,
        BUFFER_SIZE);

    nChannels = recorder.getChannelCount();

    recorder.startRecording();
    isRecording.set(true);
    recordingThread = new Thread(new RecordingRunnable(), "Recording
        Thread");
    recordingThread.start();
}
```

Figure 3-4: Code for `MainActivity.startRecording()`

Chapter 4 Performance Evaluation

To evaluate the application, we assessed the performance of the BSS algorithms, and the performance of the ASR algorithm on the extracted sources. The details of the experiments and the performance metrics used are discussed in this chapter. The results and discussion are presented in Chapter 5.

4.1 Source Images Preparation

The speech recordings used for the evaluation are provided by the TIMIT Acoustic-Phonetic Continuous Speech Corpus [7]. We prepared a total of six sources, each source consisting of only one speaker. Three of the sources are from female speakers and the other three from male speakers. All of the speakers have different speaker dialect region numbers.

For each source, we concatenated three to four recordings from the same speaker and trimmed the recording to exactly 10 seconds, or 160000 samples at the sampling rate of 16000 Hz. Each source was prepared manually to ensure that no word is cut off prematurely.

To create source images in real environments, each source was played by a loudspeaker and recorded individually by a smartphone. Both the loudspeaker and the phone are placed on level ground. For each of the source images, the loudspeaker was positioned at an angle measured counterclockwise with respect to the top of the phone and centred at the centre of the phone. The summary of the positions of the loudspeaker with respect to the phone for each speaker is shown in Figure 4-1. All source images were recorded at 50 cm away from the centre of the phone. Each of the sources at each angle was recorded in three different environments, which, in order of decreasing reverberation, were, a lecture theatre, an office, and an outdoor space. The source images were recorded in stereo at the sampling rate of 16000 Hz with 16-bit LPCM encoding and stored in WAV files.

The full details of the sources, the speakers, and the recording locations can be found in Appendix A.

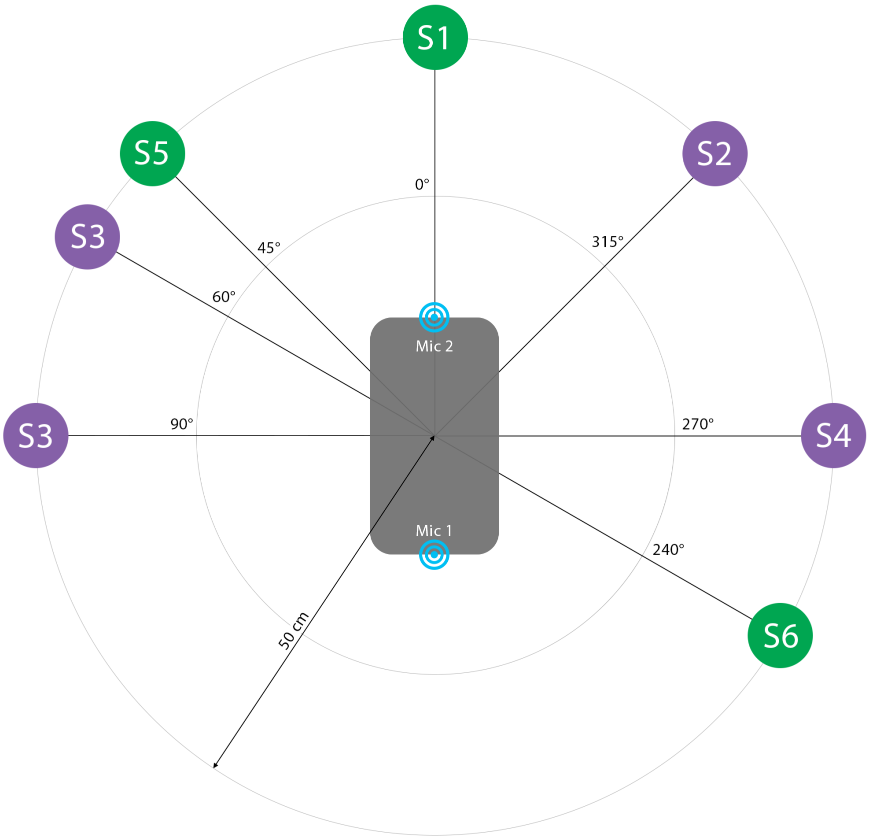


Figure 4-1: Positions of the loudspeaker for each source speaker
A green circle denotes a male speaker. A purple circle denotes a female speaker. The angle is measured with respect to the line passing through the microphones in a counterclockwise manner, starting from the top of the phone where Microphone 2 is located. Note that Speaker 3 was recorded at two different positions. The figure is not to scale.

4.2 Mixture Preparation

We generated three mixtures each for the two-source and the three-source scenarios. The first mixture contains only female sources, the second mixture contains only male sources, and the last mixture contains a mix of male and female sources. The mixtures were artificially created in MATLAB by adding the source images together, then normalized to unit amplitude. The mixtures are stored in LPCM files with a sampling rate of 16000 Hz and a resolution of 16 bits.

The configurations of the mixtures are shown, not to scale, in Figures 4-2 to 4-7. The source configuration for each mixture is the same across all acoustic environments.

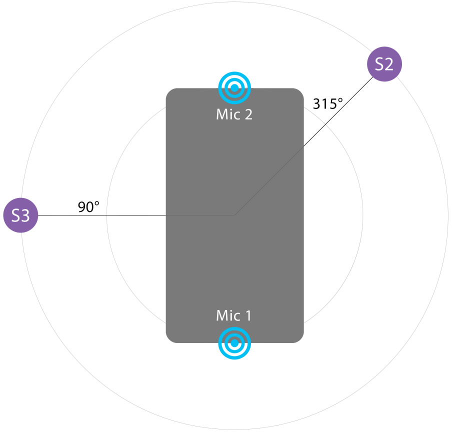


Figure 4-2: Configuration 1 for two-source mixture

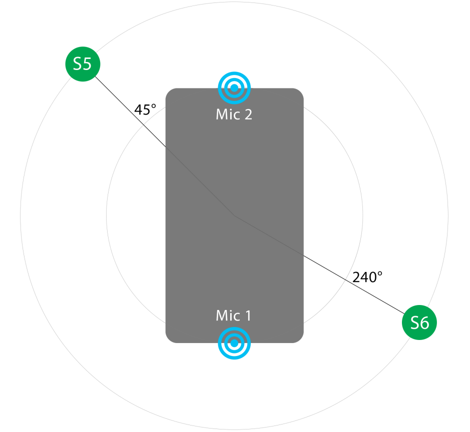


Figure 4-3: Configuration 2 for two-source mixture

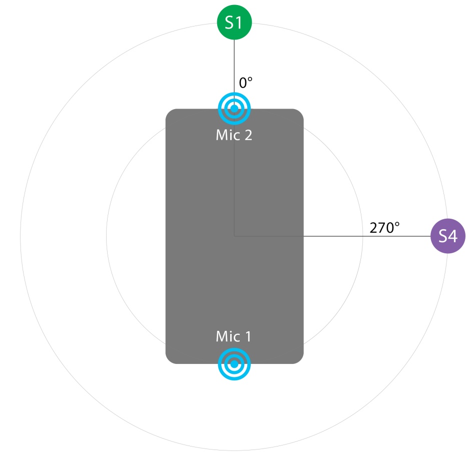


Figure 4-4: Configuration 3 for two-source mixture

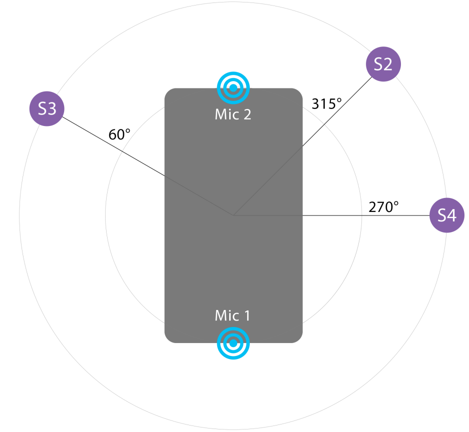


Figure 4-5: Configuration 1 for three-source mixture

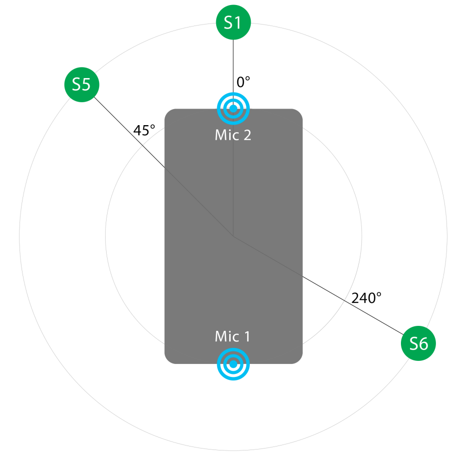


Figure 4-6: Configuration 2 for three-source mixture

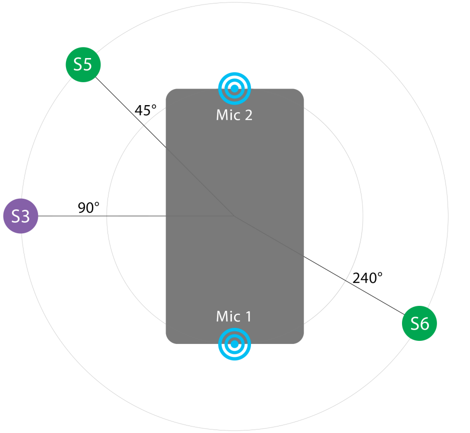


Figure 4-7: Configuration 3 for three-source mixture

4.3 Source Separation

The parameters of the relevant algorithms are listed in §4.3.1. Relevant technical specifications of the Android device used can be found in §4.3.2.

4.3.1 Parameters

For all processes, division by zero is prevented by a constant `EPSILON` of value 10^{-8}, where appropriate. Whitening transform for FastICA and SCA-DSF are done via principal component analysis.

4.3.1.1 STFT and inverse STFT parameters

The following parameters apply to all separations.

- Window function: Periodic Hamming window
- Frame length: 2048 samples
- Hop size: 512 samples
- STFT normalization: enabled
- Inverse STFT normalization: disabled

4.3.2 Device Specifications

The testing device is a Huawei Mate 10 Pro phone running on Android 8.0 Oreo (API Level 26). The device runs on a Kirin 970 chipset with octa-core CPU (4 x Cortex A73 2.36 GHz + 4 x Cortex A53 1.80 GHz), i7 co-processor, and Mali-G72 MP12 GPU. The device is equipped with 6GB of RAM and 128GB of ROM.

4.4 Performance Metrics

We further predicted the intelligibility of the source estimates with respect to the corresponding source images using the short-time objective intelligibility measure (STOI) [8] and the extended short-time objective intelligibility measure (ESTOI) [9]. For stereo source estimates, the average between the two channels was taken for each source estimate. Higher STOI and ESTOI indicate better intelligibility. The calculations were done using the authors' original MATLAB codes.

Chapter 5 Results and Discussion

This chapter discusses the results obtained from the performance evaluation.

5.1 Two-source separation

Table 5-1 shows the summary of the performance for each algorithm by acoustic environment (Env.) and the average runtimes (Rt.) for two-source separation. Except the runtimes, the reported values are the averages of across the sources and the mixtures in each acoustic environment.

	Algorithm	Rt. (s)	Env.	SDR	SIR	SAR	STOI	ESTOI	WER
Raw	AuxIVA	358.2	LTH	7.218	8.883	**13.310**	0.675	0.503	0.392
			OFC	**8.240**	10.302	**13.316**	0.705	0.544	0.310
			OTD	**14.357**	17.246	**18.263**	0.780	0.611	0.161
	FastICA	135.2	LTH	**9.564**	**15.960**	10.961	0.778	**0.609**	**0.155**
			OFC	7.873	**13.206**	10.001	**0.760**	**0.599**	**0.207**
			OTD	10.421	**18.700**	11.281	0.781	0.615	**0.146**
	SCA-DSF	**96.1**	LTH	9.425	15.675	10.917	**0.779**	0.606	0.167
			OFC	7.300	12.221	9.805	0.754	0.595	0.224
			OTD	9.705	17.472	10.852	**0.785**	**0.622**	0.162
Improvement	AuxIVA		LTH	16.959	8.838	19.726	0.423	0.330	-0.587
			OFC	16.471	10.258	16.212	0.394	0.315	-0.690
			OTD	23.727	17.041	24.701	0.483	0.440	-0.820
	FastICA		LTH	19.304	15.915	17.377	0.527	0.436	-0.824
			OFC	16.104	13.161	12.897	0.448	0.370	-0.793
			OTD	19.791	18.495	17.719	0.484	0.444	-0.836
	SCA-DSF		LTH	19.166	15.630	17.333	0.528	0.433	-0.812
			OFC	15.530	12.176	12.700	0.442	0.365	-0.776
			OTD	19.075	17.267	17.290	0.488	0.451	-0.819
	Input Mixture		LTH	-9.741	0.045	-6.416	0.251	0.173	0.979
			OFC	-8.231	0.045	-2.895	0.311	0.229	1.000
			OTD	-9.370	0.205	-6.438	0.297	0.171	0.982

Table 5-1: Performance summary for two-source separation
LTH - lecture theatre; OFC - office; OTD - outdoor.

From the results, all algorithms produce a good separation as reflected by the SDR of at least 7.2 dB. For SDR, while AuxIVA performs better in low- and moderate-reverberation (outdoor and office, respectively), it performs the worst out of the three algorithms in a high-reverberation environment (lecture theatre). FastICA and SCA-DSF have similar energy ratio metrics, performing best outdoor and in the lecture theatre but significantly worse in the office environment. In terms of the predicted intelligibility, FastICA and SCA-DSF perform significantly better than AuxIVA. This is supported by the low WER for the two algorithms.

5.2 Three-source separation

Table 5-2 shows the summary of the performance by acoustic environment (Env.) and the average runtimes (Rt.) for three-source separation. Except the runtimes, the reported values are the averages of across the sources and the mixtures in each acoustic environment.

	Algorithm	Rt. (s)	Env.	SDR	SIR	SAR	STOI	ESTOI	WER
Raw	SCA-DSF	132.0	LTH	7.043	13.204	8.475	0.653	0.470	0.325
			OFC	4.455	8.760	7.252	0.599	0.441	0.566
			OTD	7.404	13.885	8.940	0.661	0.476	0.272
Improv.	SCA-DSF		LTH	15.345	16.332	-66.128	0.200	0.127	-0.648
			OFC	10.788	12.356	-66.787	0.132	0.080	-0.427
			OTD	15.122	17.503	-65.503	0.181	0.137	-0.707
	Input Mixture		LTH	-8.302	-3.128	74.603	0.454	0.343	0.973
			OFC	-6.333	-3.596	74.039	0.467	0.361	0.993
			OTD	-7.718	-3.619	74.443	0.480	0.340	0.979

Table 5-2: Performance summary for three-source separation
LTH - lecture theatre; OFC - office; OTD - outdoor.

5.3 Summary

Overall, the application shows promising performance for source separation in real acoustic environments, particularly for SCA-DSF and FastICA. The integration of BSS with ASR significantly reduces WER by between 0.59 to 0.82 for two-source separation, and between 0.42 to 0.71 for three-source separation.

Chapter 6 Conclusion

6.1 Project Summary

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

6.2 Areas of Improvement

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

6.3 Future Works

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

References

[1] Library of Congress, *WAVE audio file format*, Library of Congress, Oct. 2012. [Online]. Available: `https://www.loc.gov/preservation/digital/formats/fdd/fdd000001.shtml`.

[2] C. S. Sapp, *WAVE PCM soundfile format*, Aug. 2004. [Online]. Available: `http://soundfile.sapp.org/doc/WaveFormat/`.

[3] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978.

[4] P. Smaragdis, "Blind separation of convolved mixtures in the frequency domain," *Neurocomputing*, vol. 22, no. 1-3, pp. 21–34, 1998.

[5] Google, *Application fundamentals*, Google Developers. [Online]. Available: `https://developer.android.com/guide/components/fundamentals`.

[6] ——, *Selecting a transcription model*, Google Cloud, Feb. 2019. [Online]. Available: `https://cloud.google.com/speech-to-text/docs/transcription-model`.

[7] J. S. Garofolo, "TIMIT acoustic phonetic continuous speech corpus," *Linguistic Data Consortium, 1993*, 1993.

[8] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2010, pp. 4214–4217.

[9] J. Jensen and C. H. Taal, "An algorithm for predicting the intelligibility of speech masked by modulated noise maskers," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 11, pp. 2009–2022, 2016.

Appendix A Source Information

Table A-1 shows the speaker initials (ID), sex, dialect region number (DR), and the sentences used for each of the sources. Sentences that are trimmed are denoted with asterisks. The concatenated TIMIT transcripts, with punctuation and sentence-case capitalisation removed, are shown in Table A-2. Table A-3 shows the WER for the clean sources and the source images.

The source images for the lecture theatre environment were recorded in Lecture Theatre 25, SS1-B2-01, Nanyang Technological University. The primary source of the background noise for this environment was the air conditioner.

The source images for the office environment were recorded in Garage@EEE Meeting Room, S1-B3c-26, Nanyang Technological University. The primary source of the background noise for this environment was the air conditioner.

The source images for the outdoor environment were recorded in an open space on Nanyang Hill, Nanyang Technological University. The sources were recorded at night when there was no foot traffic. However, there was some minor background noise from the insects, the wind, and the road traffic from a nearby highway.

Source	Speaker ID	Sex	DR	Sentences			
1	JWT0	M	1	SI1291	SI751	SI1381*	
2	AEM0	F	2	SA1	SA2	SI762	SI1392*
3	SLS0	F	3	SI1056	SI1686	SI2316	
4	BAS0	F	4	SI1387	SI1472	SI2066*	
5	DWH0	M	5	SI1168	SI1925	SX35	
6	JRK0	M	6	SI1662	SI2130	SI880	SX160*

Table A-1: Speaker information for the evaluation sources
M - male; F - female. DR1 - New England; DR2 - Northern; DR3 - North Midland; DR4 - South Midland; DR5 - Southern; DR6 - New York City.

Source	Transcript
1	they should live in modest circumstances avoiding all conspicuous consumption serve in frankfurter buns or as a meat dish but briefly the topping
2	she had your dark suit in greasy wash water all year don't ask me to carry an oily rag like that fill small hole in bowl with clay assume for
3	can thermonuclear war be set off by accident it latches when you close it so stay as long as you like Davy Mathews it's disgusting the way you're always eating
4	several factors contributed to this change she greeted her husband's colleagues with smiling politeness offering nothing He saw a pint-sized man
5	it takes a great deal of sophisticated thought to get the impact of this fact so what's this all about help celebrate your brother's success
6	did anyone see my cab See you in about an hour the revolution now under way in materials handling makes this much easier as co-authors we presented our new book

Table A-2: Transcript of the evaluation sources

Source	Clean source	Angle (°)	Source images		
			Lecture theater	Office	Outdoor
1	0.000	0	0.167	0.125	0.125
2	0.000	315	0.129	0.097	0.097
3	0.097	60	0.097	0.129	0.129
		90	0.161	0.161	0.097
4	0.045	270	0.136	0.091	0.136
5	0.000	45	0.077	0.077	0.038
6	0.100	240	0.067	0.067	0.067

Table A-3: Word error rates of the evaluation sources and source images