Universiteit Leiden
The Netherlands

Opleiding Informatica

Can the Waves of the Sea Speak to Us?

Casper de Jong

Supervisors:
Edwin van der Heide
Maarten Lamers

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

17/07/2024

Abstract

This study investigates the effectiveness of concatenative speech synthesis using various noisy corpus diversities and configurations. By systematically incrementing spectral diversity through a curated 5-stage noise corpus, we aimed to determine the minimal diversity required for a source corpus to encompass the spectral characteristics of the target corpus for proper resynthesis. This goal was notably achieved at the fifth level of source corpus diversity, where the target speech was whispered. The results highlight the complexity of speech as a phenomenon and the challenges of recreating it from noisy sources. Yet, the results show that intelligible resynthesis is certainly approachable using noisy source corpora. Further, this research paper also explores the possibilities within the realm of noisy speech resynthesis that will allow for ample improvement in future endeavours.

Contents

1 Introduction

Speech synthesis, the artificial production of human speech, has been a subject of research for decades as it has wide-ranging applications. Significant progress has been made within this field, however, gaps exist in research about the minimal diversity of the sounds required for effective speech re-synthesis, particularly from noisy or pitchless sources.

With this research project, we aim to address this by investigating the spectral envelope's role in speech re-synthesis and identifying other essential features. Advanced signal processing and machine learning techniques will be leveraged in order to contribute new insights into speech processing. These new insights are not only interesting for academia, but have the potential to be efficiently applied and lead to improvements within communication systems, interactive applications and multimedia. This leads us to the following research question:

> *Considering noisy sounds as a starting point, what is the minimal diversity of the corpus required to successfully re-synthesise speech?*

To evaluate the effectiveness of our synthesis techniques, a series of experiments were designed. These experiments systematically vary the diversity of the noise corpus to assess its impact on speech re-synthesis quality. Starting with a basic noise corpus—a simple audio clip of ocean waves—we incrementally introduce more diverse noise corpora. Each variation is rigorously evaluated using subjective listening tests, PCA analysis, and spectral analysis to determine its suitability for synthesizing a target Dutch sentence that attempts to encompass all vowels and phonetic complexities while still remaining natural and clear.

1.1 Thesis overview

This chapter contains the introduction; Section 2 contains Definitions and Related Work; Section 3 describes the implementation of the concatenative speech synthesizer along with the design choices and possibilities for expansion within the program; Section 4 explains the configuration of the program, the curation of the corpora, the process of resynthesis, the manner of evaluating the synthesis quality and the procedure of incremental experimentation; Section 5 show the results of the experiments for every corpus level; Section 6 concludes the research; and Section 7 gives suggestions for improvement and future possibilities.

2 Definitions & Related Work

2.1 Speech Synthesis Methods

Since its infancy, speech synthesis systems have relied on methods such as concatenative speech synthesis, which uses a database (corpus) typically consisting of short samples of recorded speech in order to synthesise a given vocal target[Sch05]. Examples of these concatenative sound synthesis system are Caterpillar[Sch00] and CataRT[SBVB06]. Further, there is formant synthesis, which aims to manipulate resonant frequencies (formants) by a set of rules in order to produce intelligible

speech [Kla80].

However, the limitations of the previously mentioned methods have spurred the development of more sophisticated methods of speech synthesis. Leveraging advancements within machine learning and signal processing, statistical parametric synthesis (SPS) has been developed, which statistically models the average characteristics of several speech segments in order to synthesise speech. These models tend to be based upon hidden Markov models (HMMs) or deep neural networks (DNNs). The latest methods of statistical parametric synthesis offer wide improvement in terms of the flexibility and intelligibility of the synthesised speech[BZT09].

2.2 Speech Processing

Speech processing plays a very important role in transforming raw audio signals into meaningful data. The concept of spectral envelope is a very important aspect within speech processing, as it represents the energy distribution across different frequency bands over time and plays a crucial role in speech perception and recognition. Each speech sound, in particular vowels, is characterised by a unique spectral envelope which allows for consistent and robust recognition even when the pitch is varied. This also applies to pitch-less sounds called noise[WJG05], or within the realm of speech—whispers. Thus, the spectral envelope information can be extracted and manipulated in order to synthesise speech that maintains intelligibility and a natural sound [STW00].

2.2.1 Mel-Frequency Cepstral Coefficients

Analysis of the Mel-Frequency Cepstral Coefficients (MFCC) is a fundamental technique in speech processing, offering insights into the characteristics of speech signals that are not readily apparent in the time or frequency domains. Cepstral analysis involves taking the inverse Fourier transform of the logarithm of the magnitude spectrum of a signal, revealing the quefrency domain representation of the signal[Ran17]. These cepstral coefficients capture important aspects of the parametric representation of speech[Ima05], and are thus very effective tools for speech synthesis methods. Other methods that are effective at extracting speech features are Auditory Image Model (AIM)[PAG95] and Linear Predictive Coding (LPC)[Mak75], however, neither of them have proven to be more effective over using MFCC[TK02].

Therefore, MFCC has seen widespread use within the realm of speech processing. For example, by using MFCC extracts from a speech signal the spoken words were able to be recognised[SY12]. Further, by using MFCC extracts, it's possible to detect the emotion of the speech signal[LGHR17]

2.3 Sound Classification

Central to the focus of this research project is analysing and classifying sound from different sources and matching them. Machine learning algorithms, such as support vector machines or neural networks, are very powerful tools capable of automatically extracting meaningful features from sound data and classifying these sounds based on their properties [MZX+15].

However, for the purpose of this research, a different approach was taken. The focus shifted towards employing the k-nearest neighbors (k-NN) algorithm coupled with a k-d tree for efficient nearest neighbour search. This approach offers advantages such as simplicity, ease of implementation, and effectiveness within our research.

3 Implementation of Concatenative Speech Synthesizer

In order to answer the research question, we had to create a concatenative speech synthesis program. To achieve this, we have used Max[max], a visual programming language for sound synthesis, audio processing and other multimedia manipulation. Within Max, FluCoMa[flu] was used, which is a library providing advanced audio analysis, processing, and synthesis capabilities powered by machine learning among other advanced techniques. Our program works as follows:

3.1 Program Introduction

The purpose of this implementation chapter is to not only explain the choices that were made while developing the speech synthesis program, but also to show other implementation possibilities that are beyond the scope of this research project along with reasoning as to why this might be a good alternative implementation, or why not. Figure 1 is a graphical abstract that summarises all of the possible choices mentioned within this chapter, while also highlighting the specific ones that were explored within this research project.

3.2 Loading in the Corpora

To start off, load in a source file and target file. The source file is the sound corpus which will be used to try and synthesis the target file. For the purpose of our specific research, the source file is a noisy corpus (e.g. the sound of ocean waves) and the target file is a person speaking.

3.3 Slicing

Then, we have to slice both the source and the target into segments. We do this by analysing the Mel Frequency Cepstrum Coefficients (MFCC) with a threshold, such that when the coefficients exceed this threshold, a new slice will be made. There are several methods within FluCoMa to achieve this with:

1. **Novelty:** Slicing based on novelty is a very broad concept. The basic idea behind this is to put a slice where significant changes occur within the audio. Within FluCoMa this is done using the function `fluid.noveltyslice`. There are several features on which can be sliced, such as the spectrum or chroma, but for the purpose of our research we slice based on the MFCC. This approach is very flexible and allows for a more general way of distinguishing between segments, and it offers a less fine-grained method of slicing than based on onsets or transients.

While this is an advantage, especially using it as an initial method of exploring the slicing methods, it can also be regarded as a disadvantage. The novelty slicing function only returns a slice at the start of an FFT window which means that there might be contradictory information at the start of the slice. This can be problematic when looking for parts of the source that fit this slice, as it may be looking for wrong information. This is especially true when analysing only the start of the slice. Say, the change in novelty happens at the very end of an FFT window, then the start of the slice will contain very little relevant information. However, this disadvantage can be remedied by using different ways of analysing the slices, such as taking the MFCC mean of each slice, or get the MFCC points from the start, middle and end of the slice. Combining these methods with the MFCC will result in a great slicing method for our purpose.

2. **Onset:** As mentioned before, slicing based on onset might offer more fine-grained results. FluCoMa offers the function `fluid.onsetslice`, which provides this method of slicing. Similar to the novelty slicer, the onset slicer also uses MFCC as a metric. Additionally, like the novelty slicer, it does not provide sample-accurate results but slices at the start of an FFT window. Consequently, the issue remains where analyzing only the start of the slice may give incorrect results. However, onset and novelty slicers differ in terms of *where* they choose to start their slices. The onset slicer focuses more on abrupt changes or transients within the audio signal and examines the spectrum to detect changes indicating a new onset of sound, such as a piano note or a speech syllable. Given this understanding, it appears that onset slicing is not well-suited for our source. This is because our source is noisy, and noisy sounds typically do not produce distinct onsets. Therefore, slicing based on onset is not effective for our project.

Both of these slicing algorithms (`fluid.noveltyslice` and `fluid.onsetslice` have one important parameter in common: `@threshold`. This parameter controls the threshold which essentially determines the sensitivity of the slicing process. It sets the minimum value that must be exceeded by the respective metric (novelty or onset) for the slice to be initiated. This parameter is especially important within our experiment as it directly influences the granularity and appropriateness of the slicing process for our purpose. Lower threshold leads to more slices being created, capturing finer details within the audio but also increasing computational load and perhaps creating segments so small that any important audio qualities are lost. On the other hand, higher thresholds reduces the number of slices, potentially overlooking subtle variations in the audio but also minimising the risk of over-segmentation and computational overhead. Thus, finding a balance when choosing an appropriate threshold is imperative to getting good results within this project.

Another aspect to consider when slicing is the creation of slices with uniform sizes. This entails slices that have the same time duration. This approach can remedy the issue where the source slice chosen is much shorter than the target slice. However, determining an appropriate slice size can be challenging as it depends on the characteristics of both audio files. Additionally, imposing a fixed slice size might cause the slicing algorithms to miss significant segments within the audio, as they may be restricted from slicing until the maximum slice size is reached.

Moreover, it is unnecessary to create slices within the source audio that are very quiet. Slicing algorithms may detect novelty or transients in low-amplitude regions, but it is not practical to use a quiet source slice to match a louder target slice. Therefore, we could consider filtering out slices

with very low amplitudes.

The last parameter that we will take a look at is `@minslicelength`. This parameter sets a minimum for the duration of a slice in hops. In audio processing, a hop refers to the number of samples by which the analysis window advances, such as 512 samples. By setting `@minslicelength` to 2 hops, each slice is ensured to be at least 1024 samples long. If we adjust this to 1 hop, slices can be as short as 512 samples, capturing finer details and increasing the number of slices. This adjustment allows us to experiment with the granularity of the slicing process, which is certainly a worthwhile parameter to experiment with. This is because within our project, keeping the slice lengths small is imperative. A `@minslicelength` of about 1, 2 or 3 would be a good fit, as it is big enough to capture cepstral data about the audio file while also being small enough where it remains flexible to catch fluctuations within the target audio. For the next section note that this is an important aspect as to why the choice to prefer analysing the cepstral mean versus the cepstral start was made.

3.4 Analysis

Now that we have a list of slices using either of the previous methods, we can start analysing them. This involves iterating over each slice and putting them through `fluid.bufmfcc`. This will give each slice its corresponding 12 cepstral coefficients. We have different methods of determining which slice-data to use:

1. **Start of Slice:** The first and most simple method involves only taking the 12 coefficients at the start of a slice. The `fluid.bufmfcc` function has 2 relevant parameters, namely `@numframes` and `@startframes`. The start frame will be the start of the slice. However, determining the number of frames to analyse is not as simple as it seems. The question arises, what is exactly the "start" of a slice? Taking `@numframes 1` means analyzing literally 1 sample of audio. A single sample of audio means 1/44100th of a second. This is a very small time frame, it is much too short to contain any audio data that could be analyzed. The next logical value that would represent the start of a slice would be the FFT size, which consists of 1024 frames. As mentioned before, the slices made by the novelty and onset slicers take the start of the FFT window. Therefore, the most sensible parameter to take would be `@numframes 1024` to give data about the start of the slice. However, the issues previously mentioned may arise: What if the novelty/onset detects something at the very end of the FFT window? Then the data that is being analysed is not representative of the sonic values of the whole slice at all. Therefore, it is rather difficult to use this method properly without running the risk of assigning incorrect values to each slice.

   To conclude, slicing based on the start might not always provide results that represent the cepstral values of a slice. Deciding the parameter for `@numframes` is thus a very precise matter. Accidentally taking in more frames than the length of the slice would skew the results, while taking too little frames will provide unsubstantial information.

2. **Mean of the Entire Slice:** The second method comes up to fix the issues that the previous method may bring. This method keeps `@startframes` the same. However, the `@numframes` variable dynamically changes based on the start of the next slice. The formula to determine

the number of frames is:

$$\text{numframes} = \text{next\_slice} - \text{current\_slice} \tag{1}$$

This way, the number of frames being analysed is the total length of the slice. This already eliminates the issues that the previous method brings. After analysing the whole slice, the results are put through the `fluid.bufstats` component which takes the mean of all the 12 cepstral coefficients over the duration of the slice, and creates a single representative data point which holds the mean of each of the 12 coefficients. An argument can be made that averaging out these values will misrepresent the cepstral qualities of the slice. While this is a valid thought, we can prevent any such issues by taking slices that are very small (as is explained in the slicing section above). Therefore, this analysis method combined with using very small slices will ensure certainty within the representation of the cepstral values without having to constantly experiment with precise parameters.

3. **Start, Middle and End of Slice:** Another method is to take in the cepstral values of the start, middle and end of the slice, and adding these to a single data point. Thus, each slice will have 36 coefficients (3 sets of 12). The advantage of this method is that it can provide a more detailed and accurate depiction of the slice's characteristics, making it easier to match slices between the source and target. By examining the start, middle, and end, we can account for changes and variations within a slice that might be missed if only a single point is analyzed. It can also make sure that less details are lost compared to simply averaging out the cepstral coefficients across the entire slice.

   However, this method also has its downsides. Increasing the number of coefficients for each slice can lead to higher computational complexity and may require more sophisticated algorithms for matching slices due to the increased dimensionality. Additionally, the choice of frames for the middle and end points needs careful consideration to ensure they are representative and do not overlap too closely with each other or with the boundaries of adjacent slices.

Every time a datapoint is generated, it is added to a `fluid.dataset` component, which simply stores a table that contains all the slices and their representative cepstral coefficients.

3.5 Standardization

Once we have generated the dataset containing representative cepstral coefficients for each slice from both the source and target corpora, we proceed with the standardization process. Standardization plays a crucial role in ensuring that the data is prepared and scaled appropriately for the concatenative speech synthesis task, although it may also have drawbacks, such as the potential loss of absolute values.

Standardization involves transforming the data such that each variable or dimension has a mean of zero and a standard deviation of one. This transformation is achieved by subtracting the mean and dividing by the standard deviation of each variable or dimension separately, ensuring that each feature is centered and scaled appropriately. While alternative methods like normalization, which scales the data to a fixed range, might seem plausible, they are not suitable for our specific project

for several reasons.

Firstly, normalization, by scaling the data based on its minimum and maximum values, can distort the original distribution of the data. In concatenative speech synthesis, where the characteristics of each segment are crucial for accurate synthesis, preserving the relative differences in cepstral coefficients is very important. Standardization maintains the distribution of the data, thus retaining the relative relationships between features, making it more suitable for our task.

Moreover, normalization is sensitive to outliers, as it scales the data based on the range of values. In speech synthesis, outliers could represent unique or important characteristics of certain segments that should not be disregarded. Standardization, by centering the data around its mean and scaling by its standard deviation, is less sensitive to outliers, ensuring that they do not disproportionately influence the transformation process. Therefore, we chose not to implement normalization nor use it in any of the experiments.

Lastly, within the implementation it's imperative that the data from both the source and target corpora undergo compatible transformations. This compatibility is achieved through standardization, where the same scaling parameters are applied to both datasets, thereby preserving the relationships between features. We do this by applying the `fittransform` message to our **fluid.standardize** function, which will firstly fit the model to the source dataset and then standardize the source dataset. Then the `transform` message is relayed within the standardization function, which standardizes the target dataset using the learned statistics from the previous call to the function.

3.6 Matching

Now that we have two standardized datasets—one for the source corpus and one for the target corpus—we are ready to start matching the target slices to the appropriate corpus slices. We want to have a target slice number as an input, which then outputs a source slice number whose 12 coefficients are as close to the target as possible in order to ensure proper resynthesis. There are several ways to do this:

1. **K-d Tree & K-Nearest Neighbours:** In this stage, we utilize a k-d tree to facilitate the matching process between the slices from the target corpus and those from the source corpus. The k-d tree, short for k-dimensional tree, is a data structure that partitions space into regions to enable efficient multidimensional queries. Considering our data has 12 different dimensions, this method of indexing the data is a very good choice. By fitting the data into a k-d tree, we organize the slices from the source corpus in a structured manner that enables rapid search and retrieval based on similarity metrics.

   Once the data is fitted into the k-d tree, we iterate through all the slices of the target corpus. For each slice in the target corpus, we extract the MFCC values, which are stored within our dataset.

Next, we relay the `knearest` message along with this MFCC datapoint into the k-d tree. The `knearest` message prompts the k-d tree to search for the nearest neighbor within the source corpus that closely matches the MFCC coefficients of the current target slice.

2. **K-Means:** KMeans clusters points by iteratively adjusting centroids to minimize inertia, which measures clustering quality. Each point is assigned to the nearest centroid, forming clusters. This method can reduce search space for KNN by narrowing down to specific clusters before matching individual slices. However, it will still give the nearest neighbour. Therefore it will not affect synthesis results over the k-dtree with KNN and thus it seem unnecessary to experiment with unless the computational load changes in such a way where it will give a computational advantage. Another disadvantage that K-means has is that the amount of clusters must be defined beforehand, which is not always an obvious choice.

3. **MLPClassifier:** The MLPClassifier is a type of neural network used for classification tasks. It consists of multiple layers of nodes (neurons) connected in a directed graph, with each layer fully connected to the next layer. This approach can capture more complex relationships between the input and output features. It seems like a good alternative to the k-dtree with KNN. However, it's important to note that the MLPClassifier involves more parameters compared to simpler methods like KNN and takes longer to train. Additionally, it may require more computational resources. While the MLPClassifier can be advantageous in cases where the relationships between slices are highly nonlinear and cannot be captured well by simpler methods, that doesn't seem to be the case for this project. Therefore we will make the assumption that relationships between the data, as captured by simpler methods like KNN, appear to be sufficiently mapped without the need for a more complex method like the MLPClassifier. Introducing a more complex model in this context may not provide any significant advantage for the re-synthesis task.

3.7 Synthesis

Once the corresponding slice from the source corpus has been matched to a target slice using one of the previous methods, we play the sound of this particular slice. However, since we are not taking the amplitude coefficient into account when selecting corresponding slices, an issue might occur. While the slices are a great match considering the MFCC coefficients, the source slice might have a much higher amplitude than the target slice, thus creating inaccurate synthesis results. This can be solved by multiplying the source slice signal by the ratio of the target slice amplitude to the source slice amplitude. The formula is as follows:

$$\text{source\_adjusted\_amplitude} = \text{source\_slice\_signal} \times \left| \frac{\text{target\_slice\_amplitude}}{\text{source\_slice\_amplitude}} \right|$$

Another possible quality-of-synthesis adjustment is to smooth/crossfade the source slices. This can help because abrupt transitions between slices can result in audible artifacts, such as clicks or pops, that degrade the naturalness of the synthesized speech. By applying a crossfade, where the end of one slice gradually fades out while the beginning of the next slice gradually fades in, we can create smoother transitions.

Unfortunately, neither of these adjustments have been implemented in the current version of the program due to time and scope constraints.

3.8 Final Implementation

Here follows a summary of what has ultimately been implemented within the concatenative synthesis program:

- **Slicing Methods:** The novelty slicer along with its `threshold` and `minslice` parameter have been implemented within the program and used within the experiments. The onset slicer has been implemented but is not deemed to be useful within the experiments and has therefore been excluded. The options to create uniform slices or to filter slices with low amplitudes have not been implemented.
- **Analysis Methods:** The implemented MFCC analysis takes the mean of the entire slice. The other options have not been implemented.
- **Matching the Slices:** The K-D Tree along with Nearest Neighbour search are the only ones that have been implemented, as the other options do not seem to create better or differing results as explained in their respective sections.

4 Method

To properly answer the research question, *"Considering noisy sounds as a starting point, what is the minimal diversity of the corpus required to successfully re-synthesise speech?"*, a series of experiments were designed that systematically vary the diversity of the noise corpus to evaluate its impact on speech re-synthesis quality. This section outlines the program configuration along with a notation format that will be used throughout the rest of this section. Further, the selected corpora will be explained along with the process of incrementally increasing diversity, the re-synthesis process, the evaluation metrics and finally the experimental procedure.

4.1 Program Configuration

Before we start with any experiments, we must decide what program configuration the synthesis will be ran on. The options are explained with great detail in the previous section. However, in order to maintain an overview, a graphical abstraction of these options can be found in Figure 1. For example, one of the configurations through which this program can run would be: Novelty with a threshold of 0.3 and minslice of 2, analyze the mean of the entire slice, then fit into k-d tree and find slices using nearest neighbour. This way of conveying the configurations is very wordy and inefficient. However, if we consider the fact that within the current implementation the options are not actually that many, we can slim this down into a simple notation. The only relevant parameters within this particular research are the `threshold` and `minslice` length. Thus, to clearly and swiftly convey the configuration used for running our program, we adopt the following concise notation system that displays the key components of each configuration.

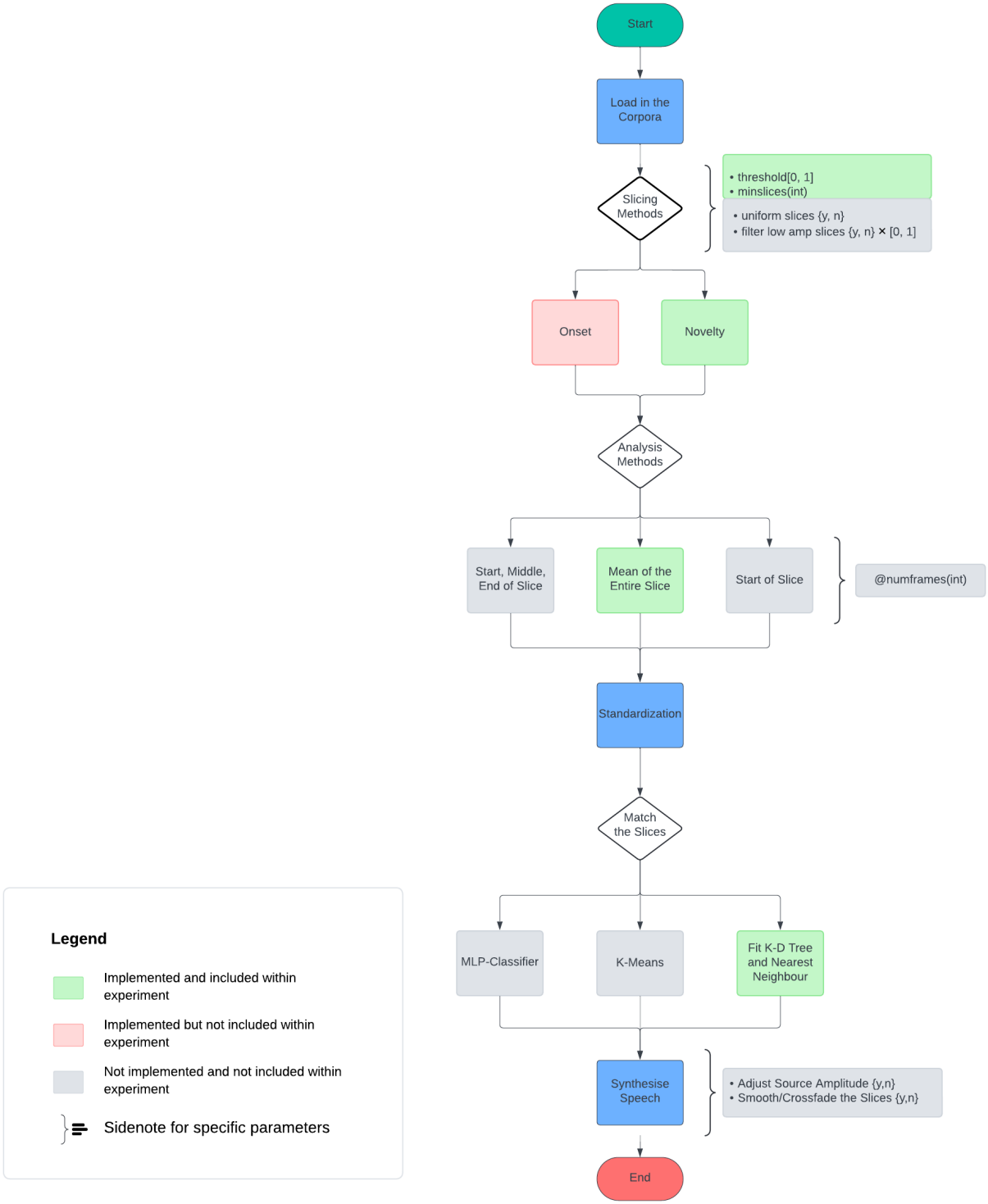


Figure 1: Program configuration overview.

4.1.1 Notation Format

Source Corpus - Target Corpus - [(threshold)(minslice)]

This notation assumes that novelty slicer is being used — the threshold and minslice value affect that slicer. Here is an example of this notation in use:

Level 1 - Target 1 - (0.3)(2): This notation means that it uses a source corpus called *Level 1*, a target corpus called *Target 1*, a threshold of 0.3 and a minslice length of 2. As a file name, it can be represented as `Level1_Target1_(0.3)(2).mp3`.

4.2 Corpus Selection

To determine the minimal diversity required within the corpus for successful speech resynthesis, we will systematically increase the diversity of the source corpus and evaluate the results at each step. We will begin with a basic noise corpus—a short audio clip of sea waves—chosen for its minimal variability to serve as a baseline. Then we will increment upon this corpus by swapping this corpus out for more diverse ones. The descriptions of these corpora are provided in the following sections.

4.2.1 Corpus Level 1: Simple Ocean

The first level is a short and simple audio clip of ocean waves. It should serve as a baseline.

4.2.2 Corpus Level 2: Extended Ocean with Equalizer

At the second level, the length of the corpus is extended with a more diverse sounding ocean. Further, the audio is processed with an equalizer to remove high frequencies that are not typically produced by human speech. Additionally, another equalizer is applied that sweeps through the audio to introduce more diversity and cover aspects that the previous corpus could not.

4.2.3 Corpus Level 3: Formant Filtered Pink Noise

For the third level, simple pink noise is used. To better mimic human speech, the pink noise is passed through a formant filter that boosts specific frequencies to resemble vowels. This is a more targeted approach compared to the previous level, aiming to improve the corpus's ability to reproduce human speech. The reason pink noise was chosen over white noise is because the high frequencies that white noise produces do not typically appear within human speech, therefore, we hypothesise that pink noise will yield better synthesis results.

4.2.4 Corpus Level 4: Whispered Alphabet

The fourth level consists of the Dutch alphabet being whispered. This corpus encompasses all of the letters within the Dutch alphabet and should be able to closely reproduce them, however, in practice, the words are not pronounced like concatenated alphabet levels. Therefore, it will still lack on certain fronts.

4.2.5 Corpus Level 5: Target Whispered

The final corpus consists of the target speech being whispered. This should encompass all the necessary sounds to effectively resynthesize the original target sentence using a noisy source.

4.2.6 Target Corpus

The target corpus we decided to resynthesise during these experiments is the following Dutch sentence: *Ik zoek een groot gevaar met mijn dure auto. (English: I am looking for a great danger with my expensive car.)* We chose this sentence because it encompasses every vowel within the Dutch language, included common words and letters within the alphabet, and sounds like a natural sentence. The sentence is pronounced in a slow manner, with every word distinctly pronounced. This will allow us to better tell what words or which parts of the words are resynthesized better compared to other parts when doing the listening tests.

4.3 Re-synthesis Process

For each level of corpus diversity:

1. **Corpus Loading:** Load the noise corpus and the target speech file into the concatenative speech synthesizer.
2. **Slicing:** Slice both the source and target files using the methods outlined in in the previous sections.
3. **Feature Extraction:** Extract MFCC features from each slice of the noise corpus and target speech.
4. **Standardize:** Standardize the features.
5. **Match:** Match the slices using one of the methods outlined in the previous sections (e.g. K-d tree and KNN, KMeans)
6. **Synthesise:** Play the target corpus and the corresponding slices that were chosen from the source.

The results are then recorded within an `.aif` file and can be re-listened.

4.4 Evaluation Metrics

To evaluate the success of the speech resynthesis, the following metrics will be employed:

1. **Subjective Listening Tests** The quality of the re-synthesis will be subjectively rated for each experiment performed on a scale of 1-5. This will be done ourselves. The aspects that will be taken into account for the quality of the synthesis are:
    - Is the result **intelligible**? Can any words or part of the sentence be understood or made out?

- Is it **consistent** with the target? Does the resynthesized audio play when it is supposed to? Is it quiet when the target audio is quiet, and does it play for the correct duration when the target audio plays?
- Is it **accurate**? Apart from intelligibility, do the sounds chosen from the source match the target in terms of length and energy? Are specific parts of the word recognizable, such as consonants or vowels?

Along with this rating, the quality of the synthesis will be described with words.

2. **PCA Analysis and Overlap Assessment** A method of evaluating the corpus's ability to resynthesise the target before any synthesis takes place is by plotting each slice data point, which consists of 12 cepstral coefficients, on a 2D graph using Principal Component Analysis (PCA). The source and the target corpora will both have their separate distinct graph. Despite an error margin of about $0.75 - 0.80$, this approach is sufficiently accurate and provides valuable insights. We will fit and transform the 12 dimensional target corpus into a 2 dimensional PCA plot. Then we give the reference of this transformation to the PCA transformation of the source corpus. This will ensure consistency within the target PCA for each corpus level and allow us to see how the corpora get more diverse. This allows for visual inspection of overlapping points as well; if the source corpus contains points similar to those in the target corpus, it indicates that resynthesis is feasible. Additionally, this will allow us to investigate whether there are regions in the PCA plot that the noisy corpus is unable to reach, which will provide further insights into the limitations and capabilities of the noisy corpus in general.

3. **Spectral Analysis** Furthermore, we will employ the SpectrumDraw object from the HISStools[HIS] library to analyze the spectral characteristics and energy distribution of individual slice points in both the source and target corpora. SpectrumDraw displays spectra from real-time buffers and inputs. By comparing the spectral displays and energy levels of corresponding slices, we can gain a deeper understanding of how well the frequency content and energy distribution of the source matches the target. This analysis will help identify any discrepancies or areas where the source corpus may fall short in capturing the spectral and energetic nuances of the target speech.

   This is a crucial component within the analysis as merely the presence of frequencies is not enough to properly resynthesise intelligible and natural sounding speech. What distinguishes natural speech is the distribution and intensity of these frequencies, known as the energy of the audio. Natural speech has specific frequency bands (formants) with varying energy levels that change dynamically over time. These energy patterns are essential for capturing the nuances of speech sounds. Therefore, in our resynthesis process, matching the spectral content alone is not enough; we must also ensure that the energy distribution of the source slices closely aligns with the target slices. This is why tools like SpectrumDraw are valuable, as they allow us to analyze and compare the spectral energy of slices from both the source and target corpora.

4.5 Experimental Procedure

4.5.1 Experimental Configuration

In order to test the corpus on each level, several sets of configurations must be performed on them to find the ideal configuration. In the Implementation section we can find all the available options. Some of these options are more important to experiment with compared to others, and the reasoning for this has been given. In Figure 1 can be found a graphical abstract which shows what has actually been implemented and is available to be experimented with. This boils down to the `threshold` and `minslice` parameter.

While testing the program and experimenting with several configurations, we encountered an unexpected problem: the outcomes of the `threshold` parameter are not evenly distributed across its range. Specifically, the sensitivity of the `fluid.noveltyslice` function's `threshold` parameter is extreme and inconsistent. For instance, setting the threshold around 0.35 results in no slice points, while slightly lowering it to 0.33 suddenly generates approximately 400 slice points.

Due to this issue, experimenting with the `threshold` parameter is not practical or meaningful. Therefore, our focus shifted to the `minslice` parameter as the primary variable for experimentation. The `minslice` parameter influences the granularity of the slices and is critical in determining the efficacy of the re-synthesis process. We do not deem this very problematic for our research. While the novelty slicer does slice on MFCC novelty within the audio file, the actual MFCC analysis happens separately to that. It is an unfortunate limitation within our experimentation, but we believe it will not gravely affect the results.

We have thus decided on the following configuration for each level. Note that possible alterations might be made if deemed necessary for specific corpus levels. If that is the case, it will be noted in the results section for that level. The configurations are as follows:

- OnsetSlice(0.0)(1)
- OnsetSlice(0.0)(2)
- OnsetSlice(0.0)(3)

We kept the threshold the same for every level due to the reasons mentioned above. Note that having the threshold be 0.3 will give the same slice amount as 0.0.

For the minimal slice length, we use values of 1, 2 and 3, as larger values tend to produce slices too large for natural-sounding resynthesis in our project.

4.5.2 Incremental Experiments

The incremental experiments entail that we will repeatedly perform re-synthesis on a noisy source corpus, which incrementally increases in complexity and variety. The results will be documented. For each of the 5 corpus diversity levels, the same experiments and evaluations will be conducted. First, the resynthesis of the noisy corpus will be performed for each configuration. Following this,

subjective listening tests will be conducted, where each resynthesis is rated on a scale of 1-5, with 1 being completely unintelligible and 5 being fully intelligible. We will also provide descriptions of the resynthesis. The best resynthesis will be selected for further analysis.

For resynthesis with the highest rating, PCA analysis will be conducted to determine where the source and target overlap or where the target is potentially unreachable by the source. This will be described and documented. Subsequently, specific points will be selected within the target PCA analysis for spectral analysis. By choosing a point within the target PCA analysis, the corresponding point within the source will also be identified. This allows us to determine if the point selected through the k-d tree has a spectrum similar enough to the target to be properly resynthesized. Through the results, we can identify which further additions to the noise corpus no longer result in significant improvements in re-synthesis quality.

5 Results

5.1 Corpus Level 1 - Simple Ocean

5.1.1 Subjective Listening Test

The three configurations are rated as follows:

minslice	Rating (1-5)
1	1
2	1
3	1

Table 1: Listening test synthesis quality rating for corpus level 1

As Table 1 suggests, the resynthesis quality of each configuration is very poor, therefore, we arbitrarily choose to describe and analyse the second (highlighted) resynthesis.

There is absolutely nothing intelligible about the resynthesis. No words of the target can be made out or even close to it. The resynthesis is slightly consistent, as it tends to consistently play while the words are being spoken, while getting quieter in between words. It is also very slightly accurate, for example, the sound of /t/ (as in "**t**ea") can be heard when it is required. Besides that, it would be a stretch to say it is any more consistent or accurate. This result was expected, as the corpus serves as a baseline.

5.1.2 PCA Analysis

The poor results from the subjective listening tests are clearly reflected within the PCA analysis as seen in Figure 2. The source corpus has a big cluster in the middle, while the target merely has a few slices that are present there. Those few slices should not have any problem being resynthesized, but not nearly enough to compensate for all of the slices that are not able to find anything remotely close enough for successful resynthesis. The rest of the slices are present either below, to the left

or to the right of the source, and thus we expect it will have difficulty finding a slice that can resynthesise it properly.

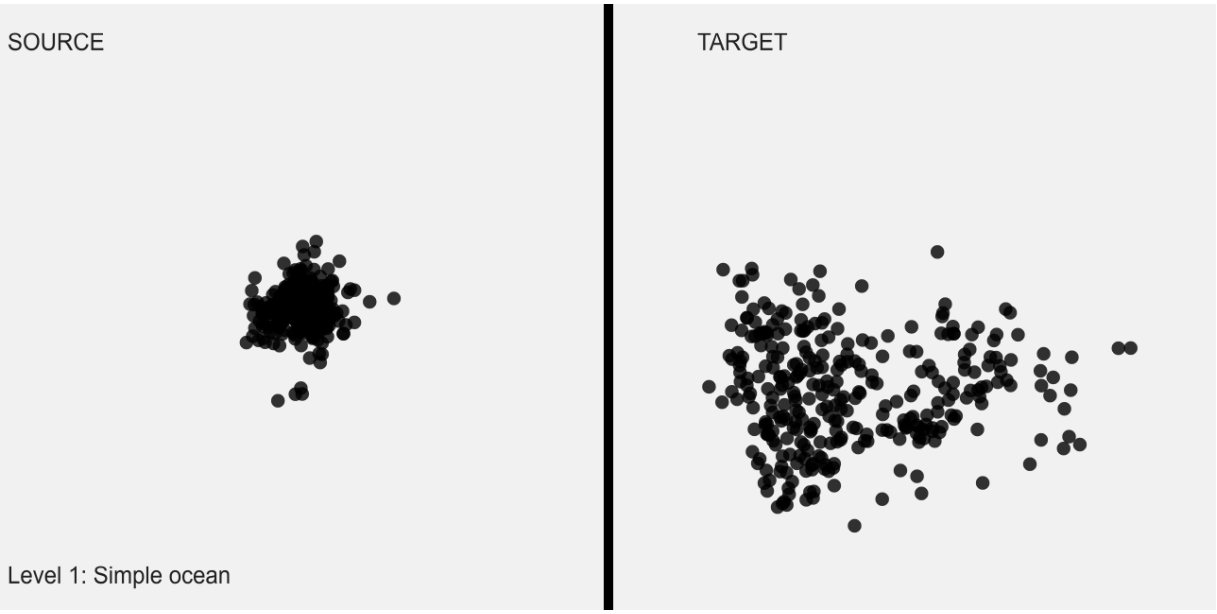


Figure 2: PCA Plot Level 1

5.1.3 Spectral Analysis

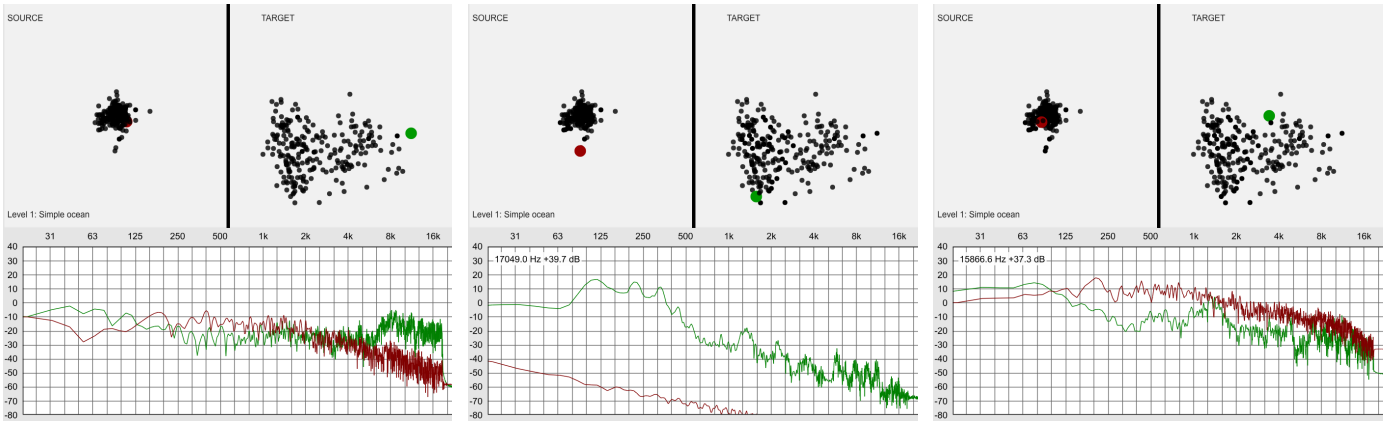
The question arises: then what are those slices that this noisy corpus is not able to encompass? We can take a look at this by looking at the spectral envelope of some of the slices in the bottom left and far right. Within figure 3 (and the spectral analysis figures in the future sections), the chosen source slice is the enlarged red dot along with the red line within the spectrum view. Green is the target.

For subfigure 3a, the target was chosen to be on the far right; very much outside the proximity of the source. In the lower end of the spectrum, both the source and target exhibit relatively flat characteristics; however, the source fails to track the general trends of the target's high-end frequencies. We can see that the target spectrum goes upwards around the 8k Hz mark, while the source slice goes downwards. Therefore, we can conclude that the matched source slice is not capable of accurately synthesising the target slice. It does not follow the targets low-end trends and it does not have the peaking high-end that the target has. This is true when looking at the sounds the slices produce. The target produces an /f/ sound, which is consistent with the high-end peak. However, the source does not even come close to this, it sounds almost like a percussive hit which is the polar opposite of the sharp /f/ sound.

Subfigure 3b target has a point in the lower left. The matching point is an outlier of the big source cluster in the middle. However, compared to the mismatch of the previous point, this

mismatch is even more jarring. The source rapidly trends downwards and barely produces any sound at all, while the target has a dynamic spectrum with varying peaks and trends. It produces an /ə/-like sound (as in "sof**a**"). The source slice is practically inaudible. It is a very bad slice mismatch.

For the last point, as seen within subfigure 3c, a point was chosen that lies more within the source cluster to see whether the matched source slice is of good quality. However, we can see that it is still not a very good match. The source target is very flat with a downwards trend towards the high-end, while the target is moving dynamically; a dip around 300 Hz, another one around 750 Hz, then remaining mostly flat with another very steep notch at about 5k Hz. The target has a much more delicate soft-g (/ɣ/) sound (as in Dutch word "**g**even"), while the source is noisy droning.



(a) (b) (c)

Figure 3: Spectral analyses corpus level 1

5.2 Corpus Level 2 - Extended Ocean with Equalizer

5.2.1 Subjective Listening Test

The three configurations are rated as follows:

minslice	Rating (1-5)
1	2
2	1
3	1

Table 2: Listening test synthesis quality rating for corpus level 2

As seen in Table 2, the first resynthesis has a higher score than the others. However, it must be noted that this difference is not big, and that it barely made the 2 rating. Though there is a difference in resynthesis quality, particularly within the accuracy aspect of the resynthesis. The second part of the target sentence, namely "gevaar met mijn dure auto" is where this configuration outshines the others. The /aː/ in "gev**aa**r" is more clearly resynthesised, and the words "met" and "mijn" are even very slightly intelligible. However, the resynthesis quality remains rather poor, and

therefore it has only barely reached the quality rating of 2.

5.2.2 PCA Analysis

While the resynthesis remains of rather poor quality, with only very slight improvements over the previous level, the PCA analysis, as seen in Figure 4, does seem to have much more overlap than the previous level. The source corpus is not just a small clump of slices in the middle, but it actually manages covers about half of the target corpus. It is mainly the right part of the target corpus that has a lot of outliers that will be difficult to resynthesise.

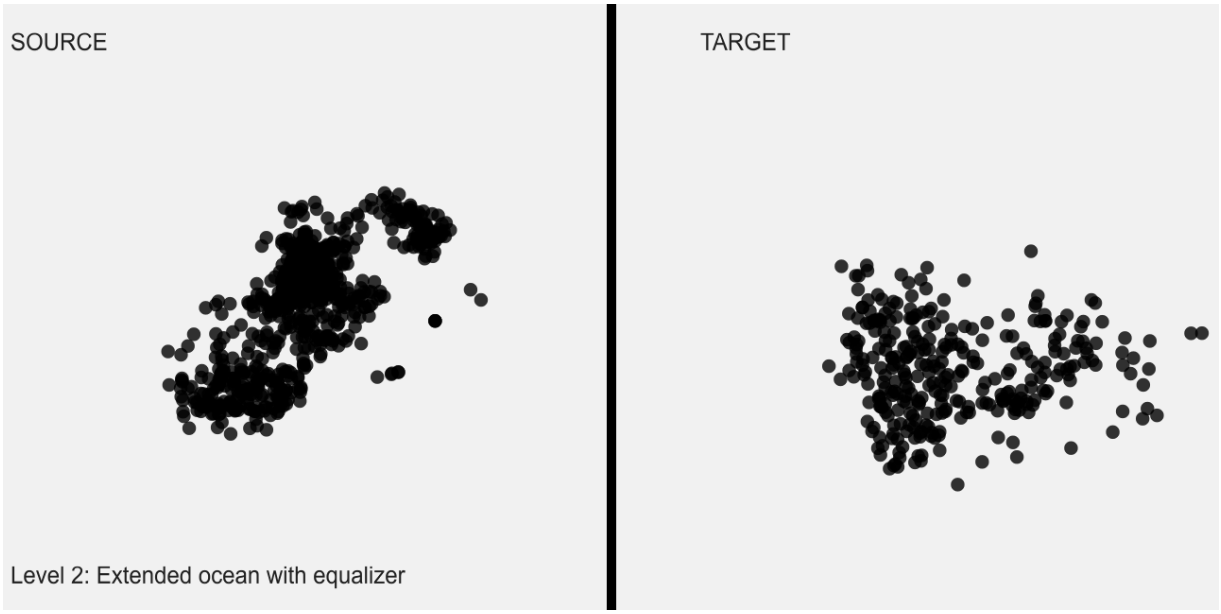


Figure 4: PCA Plot Level 2

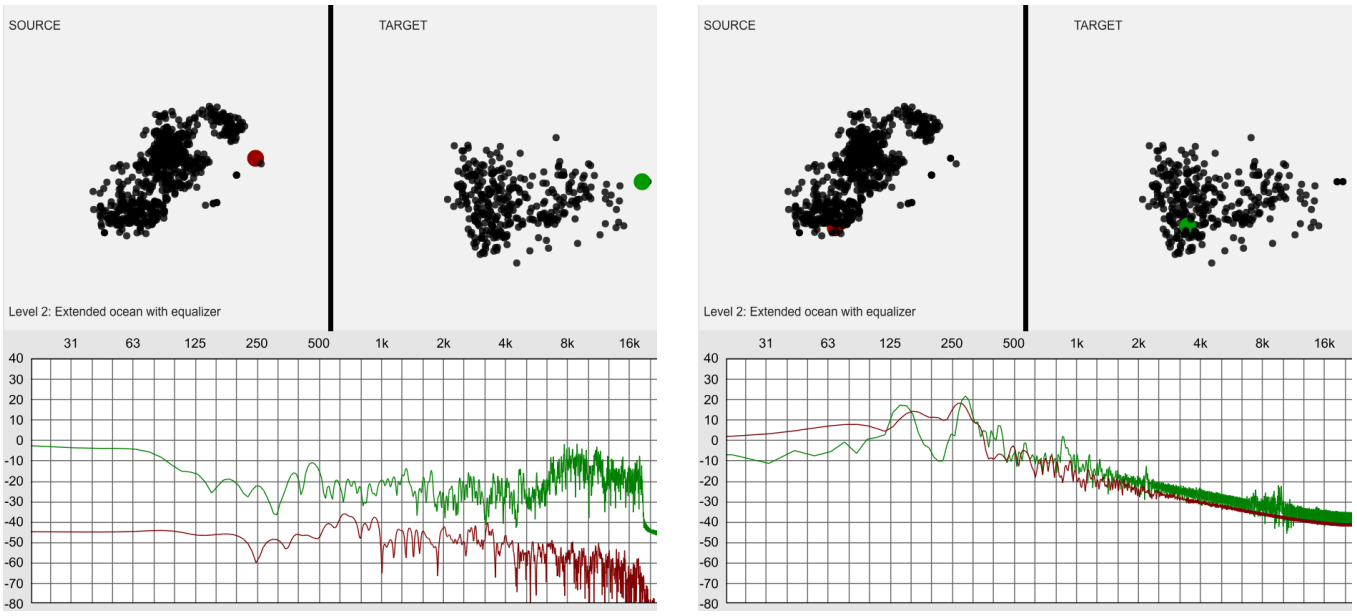
5.2.3 Spectral Analysis

This time only two different points have been taken for spectral analysis; one outlier at the far right of the target and one point that overlaps with the source corpus. The results can be seen within Figure 5.

First, we examine the furthest possible target outlier, as shown in Figure 5a. The target and the source spectra both start out flat. Some small movements here and there, but not enough to warrant a drastic change in resynthesis quality. However, there is a general upward trend around 8 kHz in the target spectrum, which the source spectrum lacks the high-end qualities to match. Due to the short length of the slices (1), it is difficult to describe the exact sound they produce. Nonetheless, it is evident that the source completely lacks the high-end necessary to reproduce this target slice.

The subfigure 5b target has a point in the middle. It is obscured by the other dots, but the found source corpus is at approximately the same spot. The source slice that it has been matched with is

almost exactly able to replicate the peak within the low-end at around 125 Hz to 250 Hz. After that, we can see that the target spectrum starts trending downwards for the rest of the spectrum, while the source spectrum takes a bit longer to catch up. The target sounds very clearly like an /uː/ sound (as in "f**oo**d"); this is where the properly matched peak in the low-end of the spectrum comes in. It is this particular peak that allows the /uː/ sound to be made. Since the source is able to match this, the resulting sound is not too far-off. Therefore, it is able to resynthesise this particular slice very well.



(a) (b)

Figure 5: Spectral analyses corpus level 2

5.3 Corpus Level 3 - Pink Noise with Formant Filter

5.3.1 Subjective Listening Test

The three configurations are rated as follows:

minslice	Rating (1-5)
1	1
2	1
3	1

Table 3: Listening test synthesis quality rating for corpus level 3

As seen in Table 3, the quality of the resynthesis is worse than at the previous level. The resynthesis with a minslice of 3 is perhaps the lowest quality we have listened to so far. The primary issue appears to be that the very last slice of the source corpus is being matched to many target slices. In every recording, the "pop" sound of this last slice repeats throughout the entire resynthesis, resulting in poor quality with no redeeming qualities in terms of intelligibility, accuracy, or consistency. Since

minslice 3 produces the worst results, and minslice 2 has worse consistency than minslice 1, we will continue the analysis with the configuration that has a minslice of 1.

5.3.2 PCA Analysis

The resynthesis was very poor, and this is partially reflected within the PCA plot as well (Figure 6). While there is some clear overlap for the slices in the middle, there is a large part of the target on the right side that has no overlap with the source, save for those three points on the right that carry the burden of representing the right part of the target. This observation is seemingly also the culprit of the problems with the resynthesis. Most of those points in that right area will match with those three outlying points that the source has on the right. This is cause for poor resynthesis as those few points cannot encompass the various sounds present in the right area of the target corpus.

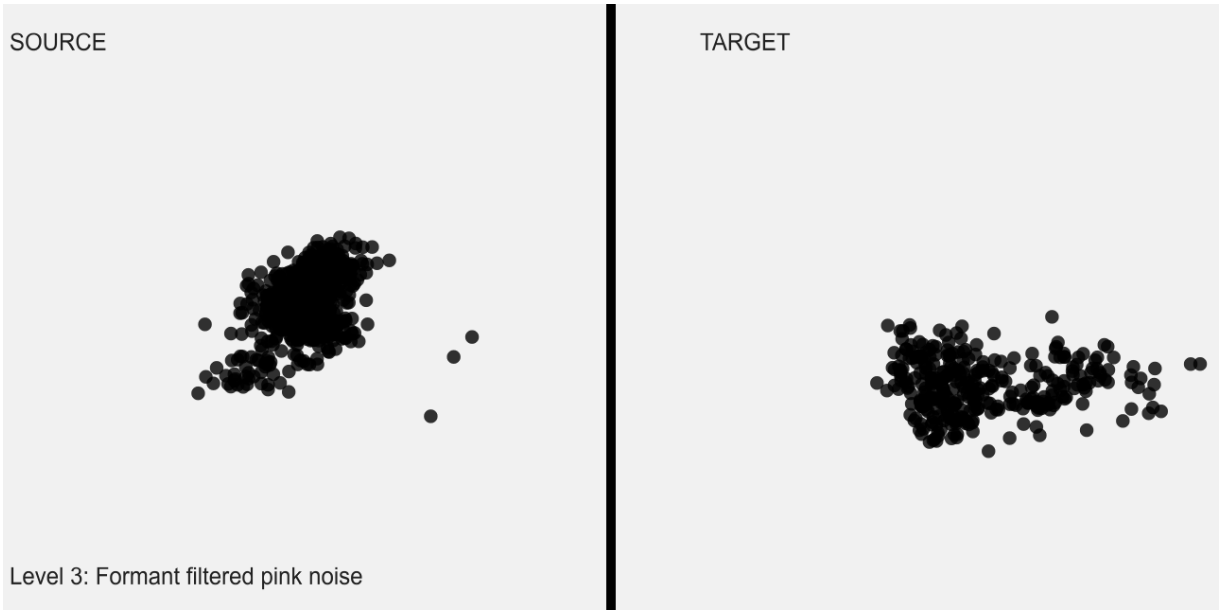


Figure 6: PCA Plot Level 3

5.3.3 Spectral Analysis

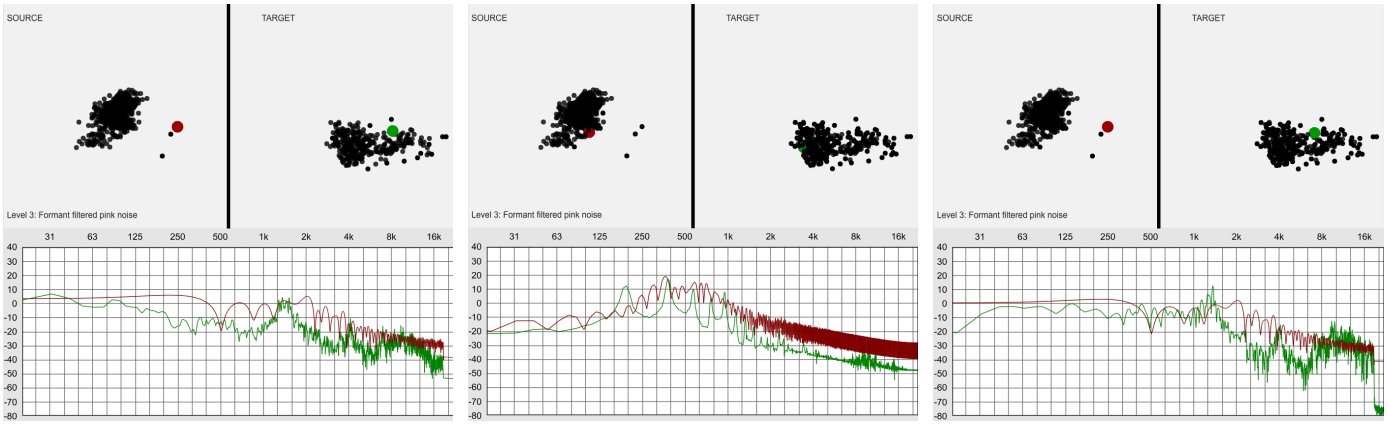
For this level 3 points are taken, as seen within figure 7.

The first point can be found within subfigure 7a. It is one of the points that make up the previously mentioned problematic right section of the target corpus that lacks spectral representation by the source. We can already see from the low-end that the source slice is a mismatch. The low-end of the source is very flat, while the target has slight downwards trends from 120 Hz to about 1k Hz. After that, it the mismatch becomes even more evident. All that the source spectrum does is curve downwards slightly from about 2k on. However, the target corpus has a peak around 1250 Hz,

another one at 4k Hz and also an upwards trend around 8k Hz. The source slice is not capable of matching this at all. These are very important peaks to follow that determine the formant qualities of the sound. In this case, the target sounds like a gentle /xk/ sound. It is difficult to describe, but imagine saying "hawk" very quickly without the vowel sound of "a". However, the source sounds completely different. It is a noisy screeching sound that will have trouble resynthesizing any kind of speech sounds. Considering that a lot of other slices in that area only have those three points to match with means that this slice will poorly represent a large part of the target corpus.

The subfigure 7b target slice is located in the middle of the target area, overlapping with the source. Therefore it should find a slice that can resynthesise it well. This holds true for the source slice that it has been matched with. There is a clear trend in both of the spectra that show a general peaking around the 500 Hz area after going back down again. Both have an /oʊ/ sound (as in "g**o**"), although the source sounds a bit more buzzy and not as natural. Considering a formant filter was applied to pink noise, we expect the source to be able to properly reproduce sounds with distinct formant features such as the /aː/ sound (as in "f**a**ther"),/iː/ (as in "s**ee**"), /uː/ (as in "t**oo**"), /eː/ (as in "s**a**y"), among others. These examples are more likely for a source corpus with these characteristics to resynthesise it correctly.

Last, the spectrum seen within subfigure 7c. This is another aspect that the formant filtered resynthesis struggles with; plosive consonants such as /t/, /k/, and /p/, among others. These sounds have distinct and abrupt acoustic characteristics that are difficult to accurately capture using formant-filtered pink noise. The mismatch can also be seen within the spectrum, as the target has a distinct peak around 1350 Hz, then a downwards decline, then two more peaks around 4k and 8k. The source high-end is mostly flat and will therefore fail to reproduce the plosive /k/ sound.



(a) (b) (c)

Figure 7: Spectral analyses corpus level 3

5.4 Corpus Level 4 - Whispered Alphabet

5.4.1 Subjective Listening Test

The three configurations are rated as follows:

minslice	Rating (1-5)
1	2
2	2
3	2

Table 4: Listening test synthesis quality rating for corpus level 4

Table 4 shows that we have reached a point in corpus diversity where the resynthesis is starting to become more sensible. Considering the poor resynthesis results from the previous levels, scoring 2 for every configuration is a decent improvement. Quite frankly, we were not able to tell much difference between either of the configurations. We will choose to further analyse configuration 2.

The first word "ik" sounds very clear; quite intelligible, and very consistent and accurate to the target. After that, it loses these qualities and becomes unintelligible again for the most part. The consistency of the resynthesis is decent, as there are distinct sections within the resynthesis that are supposed to be the resynthesized words. So, while it does sound unintelligible for the most part, there are certain parts of the words that tend to be accurately resyntesized considering the target. Most vowels seem to be in place. Further, compared to the previous level, the plosive sounds like "p" and "t" are accurately represented within the source corpus and therefore the resynthesis is able to cover these sounds consistently and accurately. However, it is still not enough to warrant a high quality resynthesis.

5.4.2 PCA Analysis

Compared to all of the previous levels, this source corpus (Figure 8) seems to be the most complete by far; great coverage of the target and no outliers that will skew the matching process. The target has a little more slices present on the right compared to the source corpus, however, the source corpus has a decent variety of slice points that could compensate for that. Overall, the PCA results look promising and better resynthesis quality would have been expected.

5.4.3 Spectral Analysis

The results of the spectral analysis can be found within Table 9.
For our first slice we examine one of the points that finds itself on the right side of the target spectrum. As mentioned in the PCA analysis, the right side of the target is more dense than the source. However, we hypothesised that the source still has enough representative slices within this area that will allow for proper resynthesis. As we can see within the subfigure 9a, the target does not have any distinct peaks and neither does the source within the low-end. Therefore any differences between them are quite negligible. The important part comes in after about 1k Hz, where the target spectrum starts deviating a bit creating upwards and downwards trends. The source case is able to follow these trends pretty accurately, except for around 8k Hz where the target spectrum

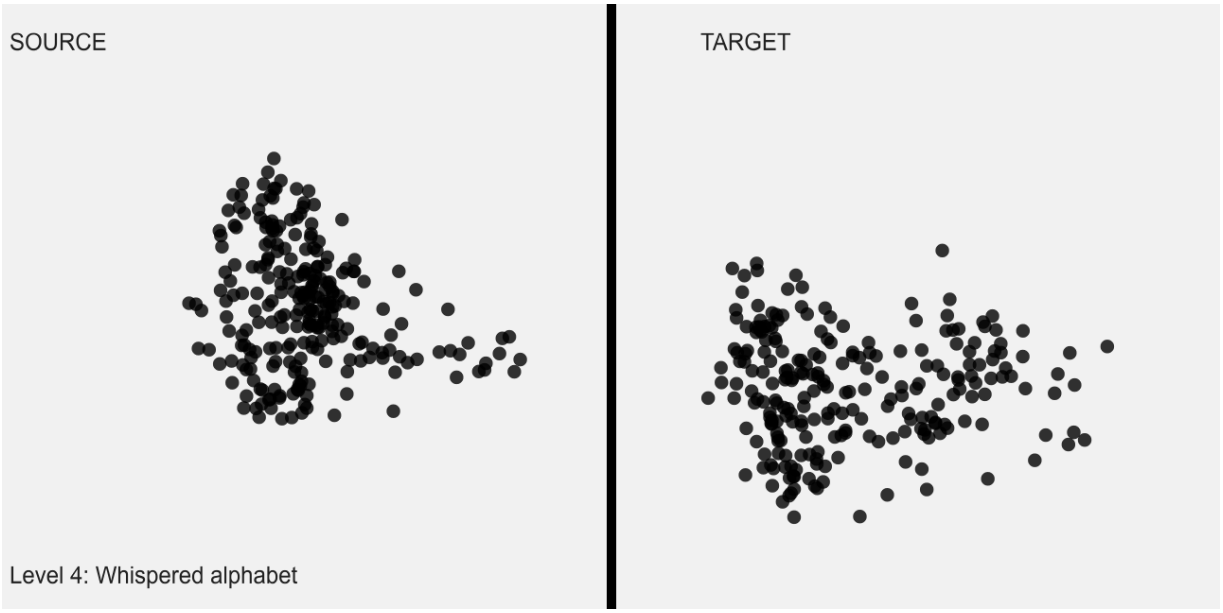


Figure 8: PCA Plot Level 4

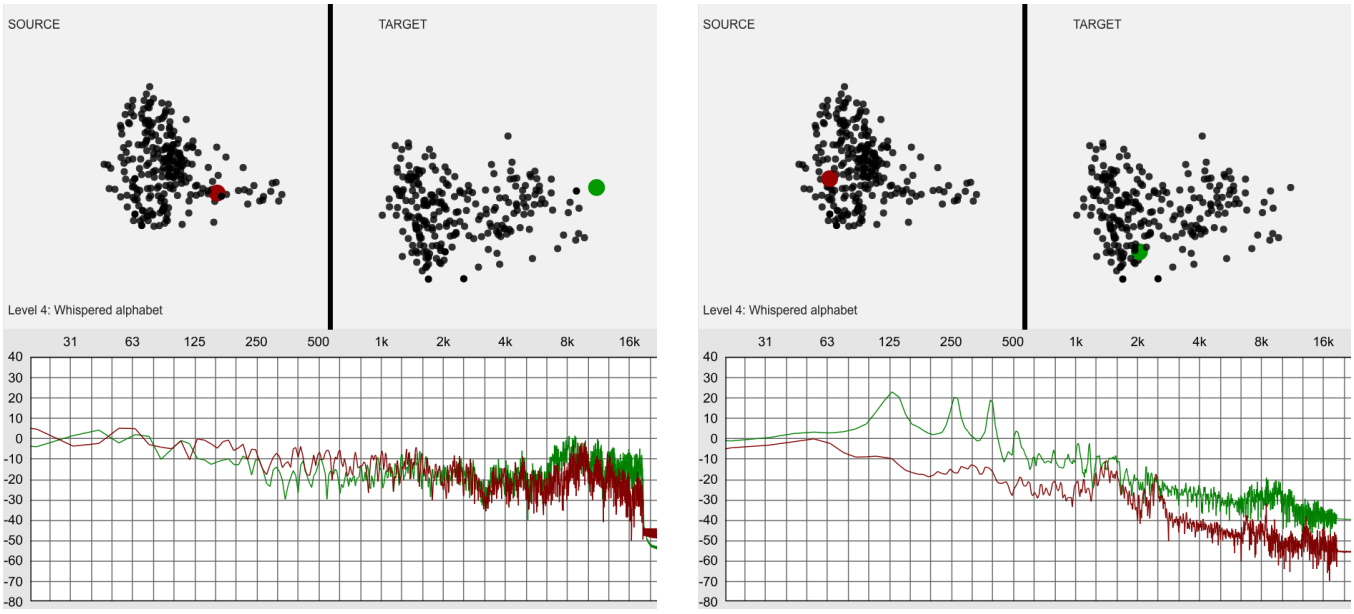
trends upwards a bit faster compared to the source spectrum. Again, this difference is negligible, especially if we listen to what the slices represent. The target slice has a distinct /f/ sound (as in "**f**loat") that the source is capable of accurately reproducing.

When exploring the centre, some discrepancies come to light. Take a look at subfigure 9b. The target has a /ʊ/ sound (as in "g**oo**d") while the source sounds more like an /h/ (as in "**h**ello"). They sound completely different. Not only that, but the spectral qualities are also very different. The source has a large peak from 125 Hz to about 500 Hz, creating that /ʊ/ sound. The source does not follow this trend at all and even goes downwards a little bit around those frequency ranges. Even though the cepstral coefficients are seemingly pretty close to each other as they matched during the k-d tree process, the sound is still drastically different. This is an issue throughout the whole resynthesis process of this level, as these odd sounds repeatedly play out that do not resemble the target nor any human speech qualities. It shows that speech is extremely complicated; merely having the letters of the alphabet present within the source is clearly not enough to cover all of the nuances that come with using these different letters in all the different contexts within human speech.

5.5 Corpus Level 5 - Whispered Target

5.5.1 Subjective Listening Test

The three configurations are rated as follows:
As Table 5 shows, a significant jump in resynthesis quality has been achieved. Configurations 1 and 2 score a rating of 4, while configuration 3 scores a 3. Compared to the previous level, where every configuration scored a 2, this improvement is very notable. Both configurations 1 and 2 offer great results, while configuration 3 lags slightly behind.



(a) (b)

Figure 9: Spectral analyses corpus level 4

minslice	Rating (1-5)
1	4
2	4
3	3

Table 5: Listening test synthesis quality rating for corpus level 4

Configuration 3 has a larger minslice value, causing each slice match to have a more substantial impact on the final result. If a wrong slice is picked, the decrease in quality is more significant compared to configurations that use smaller slices. However, if the right slice is picked, the benefit is greater. This makes it a bit more unreliable, and in this instance, the result was unfortunately poorer than the other configurations. Since configurations 1 and 2 offer similar results, we will continue with configuration 2 in the next sections.

The first three words, "ik zoek een," sound phenomenal within the resynthesis. The intelligibility, accuracy, and consistency of these three words are significant. After this, the result decreases slightly in quality. It is not able to produce the r sound in "groot," reducing the intelligibility of this word considerably. However, its consistency and accuracy remain high. This trend continues in the later resynthesized words; one or two mismatches in slices cause the words to lose intelligibility a bit too much for it to be considered a "perfect" resynthesis.

5.5.2 PCA Analysis

The PCA plot of the source at this level, as shown in Figure 10, do not appear to cover the target particularly better than in the previous level. There are fewer points, and the area in the (top) right seems underrepresented. Nevertheless, the source exhibits a strong sonic presence and diversity as the resynthesis is much better compared to the previous level.

Given that the source is identical to the target, except for being whispered, one might assume that all necessary components are present for a perfect resynthesis of the target. However, in practice, this ideal scenario is not always the outcome, as we will discover in the spectral analysis. Nonetheless, this level appears to outshine all other previous levels.

5.5.3 Spectral Analysis

The results of the spectral analysis can be found within Table 11.

Even though the quality of the resynthesis has drastically improved, and the PCA analysis looked very good too, there are still discrepancies to be found that cause for poor and jarring results. As seen within subfigure 11a, there is another mismatch on the low-end. This is so crucial because certain important vowel-inducing frequencies are present here. We can see that the peak from about 100 Hz to 300 Hz is not covered at all by the matched source. In fact, during this peak the source even trends downwards a little bit. This is immediately evident within the sound of slices. The target is a very clear /z/ sound (as in "**z**ip"), while the source is a noisy percussive sound, almost sounding like a /t/ yet it repeats much more and completely fails to reproduce any sensible sound used within human speech. While the high end frequencies around 2k Hz to 16k Hz are covered very accurately by the source, it proves to be insufficient for resynthesis. Looking further through the source corpus, however, no sound that resembles the /z/ can be found. When the word "zoek" gets whispered, the /z/ sounds more like an /s/ We can conclude from this that whispering the target sentence will not be sounding exactly like the regularly spoken target.

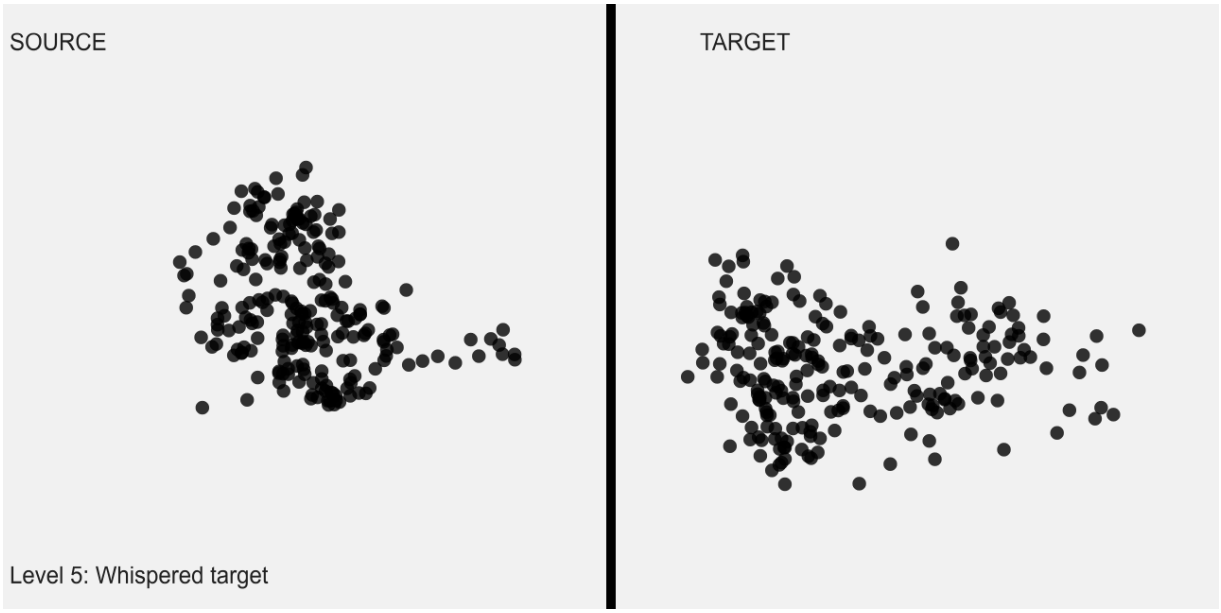


Figure 10: PCA Plot Level 5

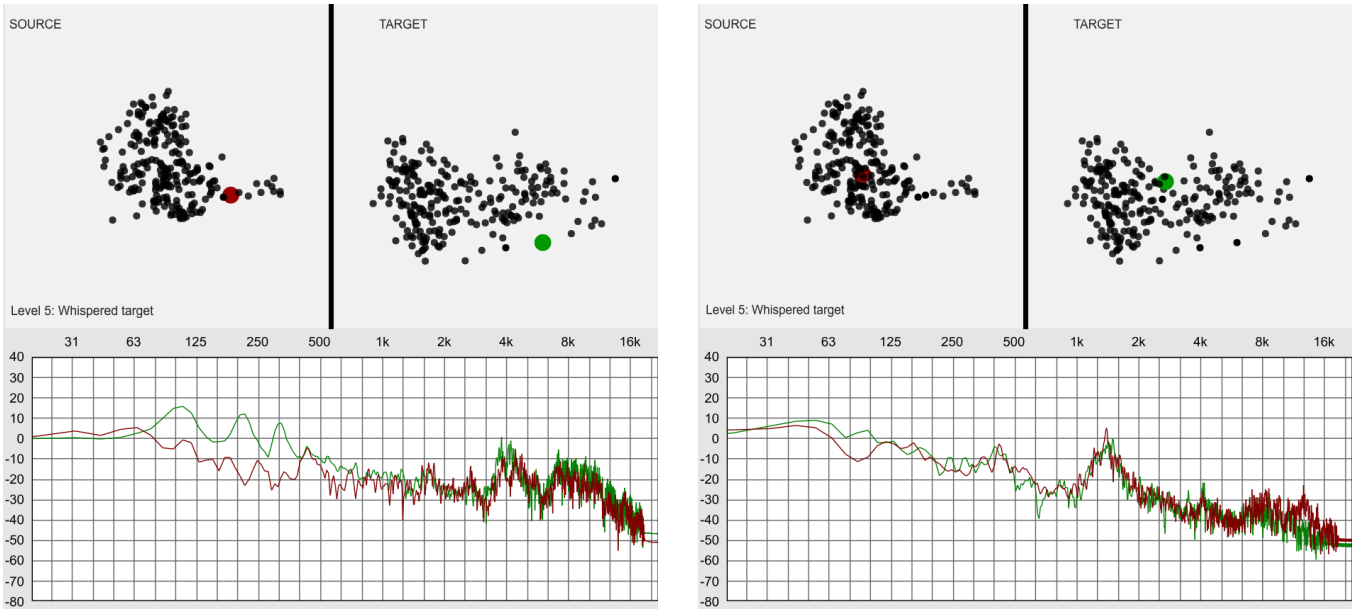
This slice, seen within subfigure 11b, shows where this source corpus shines. Not only does the source spectrum almost fully cover the target spectrum, the slices sound practically the exact same. They are both a plosive /k/ sound. The only difference is that the source slice has a more noticeable treble compared to the target, which can bee seen within the source spectrum as it peaks around 12k Hz. Within the realm of intelligibility and resynthesis quality this is a negligible difference.

6 Conclusions

In this study, we investigated the effectiveness of concatenative speech synthesis using various noisy corpus diversities and configurations. Our goal was to determine the minimal diversity of a noisy corpus required to successfully resynthesize speech. Through systematic experiments, where we iteratively increased corpus diversity, we approached good resynthesis of the target sentence. Notably, this was achieved using a source corpus where the target sentence was whispered; the other levels did not have enough diversity for a high quality resynthesis.

Our findings showcase the complexity of speech as a phenomenon; while whispered and spoken versions of the same content may have identical meanings and maintain intelligibility, they can exhibit distinct acoustic characteristics. This difference posed challenges within the resynthesis of the target. The biggest cause of low quality resynthesis were poor mismatches in slice selection which notably affected the intelligibility of the resynthesis.

Despite these challenges, our study highlights considerable potential for improvement in noisy speech resynthesis. It shows that a corpus where the target is whispered has clear potential to



(a) (b)

Figure 11: Spectral analyses corpus level 5

encompass all of the necessary spectral characteristics for high quality resynthesis. Ample room for improvement remains, and some suggestions for future research topics within the realm of noisy speech resynthesis can be found in the following section.

7 Future Research

In the future, experiments can be made with implementing the other program configurations outlined within this paper. It can yield different results for different applications. Among this, the following suggestions can build upon the research within this paper:

7.1 Exploring Larger Slice Lengths

Experimenting with larger slice lengths could reduce computational overhead and provide new insights into the balance between granularity and synthesis quality. It is worth investigating if larger slices maintain sufficient detail for accurate speech re-synthesis, especially in more diverse noise corpora.

7.2 Amplitude Matching and Crossfading

One aspect that can improve the intelligibility and naturalness of the re-synthesised audio is matching the source amplitude with the target amplitude. Further, implementing crossfading between slices can reduce artifacts like clicks or pops, enhancing the smoothness of transitions in the synthesized audio.

7.3 Dynamic Parameter Adjustment

Developing methods to dynamically adjust slicing and analysis parameters based on the characteristics of the input audio could optimize the re-synthesis process. Machine learning techniques could be employed to predict optimal parameters for different types of audio, potentially improving the adaptability and accuracy of the system.

7.4 Improved Evaluation Metrics

Developing more sophisticated evaluation metrics beyond subjective listening tests, PCA- and spectral analysis can provide deeper insights into the quality of re-synthesis. Objective measures of intelligibility and naturalness, possibly incorporating perceptual audio quality metrics, can offer a more rigorous assessment of synthesis success. An example of such an objective metric is Mel Cepstral Distortion[Kub93]. It allows for a detailed comparison of the spectral characteristics of synthesized speech to a reference, and incorporating mel-warped critical band filters can enhance the correlation with subjective quality assessments by better modeling perceived audio quality.

References

[BZT09] Alan W. Black, Heiga Zen, and Keiichi Tokuda. Statistical parametric speech synthesis. *Speech Communication*, 51(11):1039–1064, November 2009.

[flu] FluCoMa. https://www.flucoma.org. Accessed on: July 17, 2024.

[HIS] HISSTools. https://research.hud.ac.uk/institutes-centres/cerenem/projects/thehisstools/. Accessed on: July 17, 2024.

[Ima05] Shoichi Imai. Cepstral analysis synthesis on the mel frequency scale. *International Conference On Acoustics, Speech, And Signal Processing*, 2005.

[Kla80] Dennis H. Klatt. Software for a cascade/parallel formant synthesizer. *Journal Of The Acoustical Society Of America*, 67(3):971–995, 1980.

[Kub93] R. Kubichek. Mel-cepstral distance measure for objective speech quality assessment. In *Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing*, volume 1, pages 125–128 vol.1, 1993.

[LGHR17] M. S. Likitha, Sri Raksha R. Gupta, K. Hasitha, and A. Upendra Raju. Speech based human emotion recognition using mfcc. *IEEE WiSPNET 2017 Conference*, 2017.

[Mak75] J. Makhoul. Linear prediction: A tutorial review. *Proceedings Of The IEEE*, 63(4):561–580, January 1975.

[max] Max. https://cycling74.com/products/max. Accessed on: July 17, 2024.

[MZX+15] Ian McLoughlin, Haomin Zhang, Zhipeng Xie, Shuicheng Yan, and Xiao Wei. Robust sound event classification using deep neural networks. *IEEE/ACM Transactions On Audio, Speech, And Language Processing*, 23(3):540–552, 2015.

[PAG95] Roy D. Patterson, Mike H. Allerhand, and Christian Giguère. Time-domain modeling of peripheral auditory processing: A modular architecture and a software platform. *The Journal Of The Acoustical Society Of America/The Journal Of The Acoustical Society Of America*, 98(4):1890–1894, 1995.

[Ran17] R.B. Randall. A history of cepstrum analysis and its application to mechanical problems. *Mechanical Systems and Signal Processing*, 97:3–19, December 2017.

[SBVB06] Diemo Schwarz, Grégory Beller, Bruno Verbrugghe, and Sam Britton. Real-time corpus-based concatenative synthesis with catart. In *9th International Conference on Digital Audio Effects (DAFx)*, pages 279–282, Montreal, Canada, Sep 2006. https://hal.archives-ouvertes.fr/hal-01161358.

[Sch00] Diemo Schwarz. A system for data-driven concatenative sound synthesis. In *Digital Audio Effects (DAFx)*, pages 97–102, Verona, Italy, Dec 2000. https://hal.archives-ouvertes.fr/hal-01161115.

[Sch05] Diemo Schwarz. Current research in concatenative sound synthesis. *International Computer Music Conference (ICMC)*, September 2005.

[STW00] David S. Stoffer, David E. Tyler, and David Wendt. The spectral envelope and its applications. *Statistical Science*, 15(3), August 2000.

[SY12] Siwat Suksri and Thaweesak Yingthawornsuk. Speech recognition using mfcc. *International Conference On Computer Graphics, Simulation And Modeling (ICGSM'2012) July 28-29, 2012 Pattaya (Thailand)*, January 2012.

[TK02] Minoru Tsuzaki and Hisashi Kawai. Feature extraction for unit selection in concatenative speech synthesis: comparison between aim, lpc, and mfcc. *ICSLP2002*, September 2002.

[WJG05] Jason D. Warren, A. R. Jennings, and Timothy D. Griffiths. Analysis of the spectral envelope of sounds by the human brain. *NeuroImage*, 24(4):1052–1057, February 2005.