FACULTY OF ENGINEERING

UNIVERSITY OF LEEDS

ELEC3810 Final Report

Using Spectral Mapping to Exploit and Enhance an Object's Latent Harmonicity

Matteo Fabbri
201071767
el16mf@leeds.ac.uk

Reuben Frankel
201042778
el16rf@leeds.ac.uk

Rhun Gwilym
201024698
el16rpg@leeds.ac.uk

Zai'En Pan
201072683
el16zep@leeds.ac.uk

Abstract

Spectral mapping is a technique well-documented in the field of audio signal processing, in which inharmonic sounds can be made harmonic and vice-versa. The remapping of sonic information has been used as a compositional tool, where existing musical instruments have been morphed into unusual timbres without affecting their sonic identity. Since the reverse is also possible, this project builds upon the concept of spectral mapping to transform ordinary objects into musical instruments. Existing systems such as Mogees [1] have achieved the conversion of objects into instruments, where sounds are produced as the result of the user executing physical gestures on a surface. It is important to note that interaction with these objects simply triggers user-defined physical modelling synthesiser patches, of which the output sound bears no acoustic resemblance to the input. Spectral mapping provides the possibility of creating an instrument from any object, with a heavy emphasis on the reflection of its original tonal character at the output. Its sonic properties will be analysed, remapped, and reconstructed into a sound that is more harmonic (see section in background research), whilst still retaining qualities of its original form. The result will be a unique instrument that is made consonant through the object's own properties.

Acknowledgements

We would like to thank our project supervisor Mr. David Moore for his unrelenting support on this project. Additional thanks to Dr. William A. Sethares and Dr. Andrew J. Milne for providing many helpful spectral mapping resources, as well as Dr. Gabriele Bunkheila for assisting with MATLAB Audio Toolbox.

Table of Contents

1. Introduction

Overview

We are surrounded by objects in our daily lives that seem relatively unmusical when compared to traditional musical instruments that are rich in tonal qualities. However, we believe that with modern digital signal processing (DSP), an object's hidden harmonic qualities can be extracted and enhanced to sound much more like their musical counterparts. Their potential in compositions have been explored by existing studies like William Sethares' *TransFormSynth* and Mogees Ltd's *Mogees,* where it has been proven that such objects can be transformed into musical instruments in their own right.

Aims and Objectives

The aim of this project is to explore how sounds can be made tonal and musical from inharmonic sources. This notion was exciting to us as the experimentation could generate unexpected sounds from everyday objects around us. While previous studies have focussed heavily on achieving a high level of harmonicity from the object, we are hoping to improve the balance of the original timbre while still enhancing its hidden harmonicity. It is not an easy task to perform without introducing major changes to its spectral character, but the usage of software processing methods such as spectral mapping can make this possible. In this paper we will document a software system developed in MATLAB that for any given input signal, performs frequency analysis and determine partials to be shifted with the goal of maximising the consonance produced. We will also present the methods used in analysis and the manipulation stage to retain the underlying characteristic timbre of the input signal, ensuring that it is not lost through aggressive processing.

2. Background Research

Existing Developments

Mogees [1], developed by Mogees Ltd. is a way of transforming everyday objects of any material such as metal chairs and wooden tables into a musical instrument. It operates via a dedicated app on the iOS or Windows platform to detect specific pre-made gestures e.g. tapping, scratching in real-time using custom vibration sensors (high-quality contact microphones) which are attached to the desired instrument's surface. When detected, these gestures will play the desired sounds which can trigger a MIDI note, samples or in-house libraries, virtually replacing the need for a MIDI controller. Although relatively easy to set up with the use of machine learning, this system will not work on every object as it must possess a material that will carry the vibration energy throughout unlike a pillow for example which will dampen it.

Looper [2] is a project developed by Lukas Siegele that includes two products, an input and an output object which are used to create musical rhythms. The input object is similar to a drum pad where the user taps a specific beat (with their hand) which is then sent as digital data to the output object via Wi-Fi. This output object, after receiving the required information, will then repeat this rhythm in the form of a mechanical-knocking-arm which uses kinetic energy to create a sound from hitting a dedicated sound body object. An example of this can be any object that is nearby such as a household object which allows for one to generate new, never heard before sounds and sophisticated patterns.

Of Nature and Things is a interactive sound installation by Dutch sound artist Fredde ten Berge. It involves utilising the vibrational qualities of materials. Several sensors, electret, and piezo microphones are used in conjunction with a Bela embedded system to generate sound in Pure Data (Pd) or in C-code. In one of the objects, *The Trunk,* specially designed ceramic materials rest upon a tree trunk where piezo microphones amplify its woody resonance, making it a percussive instrument. Its sonic properties are processed via different filter banks prior to sonication [3].

Key Literature

Spectral Mapping

The originator of this technique, William Sethares, implements analysis/resynthesis methods by extracting partials of an input sound and then remapping these frequencies to a user-specified or predefined harmonic template. Spectral peaks, identified as points of maxima, are separated from inharmonic floor noise (having very low amplitude) in a signal. The original character of the sound is maintained by the unaltered noise components, while only the extracted peaks are processed. In his program *TransFormSynth*, which later became *Transformer,* several mapping methods exist for the extracted tonal features, such as a fixed spectral mapping, continuous morphing, and "Dynamic Tonality". The fixed

configuration, which is also the most straightforward, maps each partials to a fixed destination such as a harmonic series based on a fundamental frequency. Our usage of spectral mapping techniques will be documented and compared in the **Design and Development** section below.

Harmonicity

As our system places a heavy emphasis on harmonicity, in this section we will detail the thought process in our approach to morphing our object from its original state to a transformed instrument. Woodhouse [4] states that not every excitable object will sound as pleasant as an musical instrument. This is because the frequency response of such objects may often include resonant tones that are not harmonically related to each other. A typical instrument on the other hand usually has a clear, defined pitch (a high-level definition clarified by Planck [3:117-119]), these natural frequencies need to be related by a numerical pattern comprised of ratios - formally known as a harmonic series. This series describes the sound having a fundamental frequency where each increasing overtone is a multiple of the fundamental. Most objects typically produce complex tones featuring multiple frequency components, whereas most Western musical instruments produce harmonically complex tones following the harmonic series based on the fundamental frequency [5]. Therefore, in order to make our target object more harmonically pleasing to the ears, these overtones should be in whole-number multiples (f, 2f, 3f, nf) or close to these values.

This has been investigated further and proved by analysing the consonance and dissonance of chords and intervals on a piano discussed in McDermott's work [6:236-240]. He describes these same ratios when comparing each individual harmonic intervals played simultaneously on the piano, showing results of most consonance and dissonance with a perfect fifth (ratio of 3:2) and a tritone (ratio of 45:32) respectively. This can be backed up with a phenomenon known as 'beating', when two frequencies shift in and out of phase, resulting in a cyclic constructive and destructive interference. This is what creates the apparent 'roughness' in harmonicity, a property that determines an interval's unpleasantness where a combined wave will not repeat until a certain amount of periods has passed as explained by Sethares [7:81-82]. Many pairs of these beating frequencies can be found in dissonant chords but not as many in consonant ones. As a result, a rougher waveform is generated when compared against each other.

Using a struck cymbal as an example, it produces a complex sound that is considered inharmonic, as its wide range of frequencies are not consonantly related to each other. Kuratani et al. [8] states that a cymbal's nonlinear resonance is affected by its size, shape, and hammering process when the cymbal is fashioned. With spectral mapping, it is possible to make an inharmonic object such as the cymbal sound more harmonically desirable according to the user's specifications, so that it can be used in a myriad of situations. By just relying on its sonic information, its partials can be altered to improve its consonance without changing the physical structure of the object.

Harmonic Entropy

Harmonic entropy is used as a measure of dissonance in psychoacoustics. Sethares mentions that harmonic entropy ‘provides a physical correlate of tonalness, one aspect of the psychoacoustic concept of dissonance’ [7:371]. The scientific method to measure such tonalness or harmonic entropy of a given sound is to calculate how close its sinusoidal partials are to the harmonic template of the fundamental frequency, composed by all the frequencies equal to integer multiples of the fundamental frequency.

Peak Detection

Determining Fundamental Frequency

Plack states that the production of a pitch is predominantly dependent on waveform repetition in a signal [9, 118]. Inharmonic sources tend to be non-periodic and according to Gerhard [10], various methods exist for different input sources, thus the context is important. Furthermore, sources with plenty of roughness or noise can make the detection of lowest frequency component difficult. Sethares’ approach allows for a movable fundamental frequency in the output that can be changed on the fly. This should be avoided in our project, as it moves away from the goal of retaining an object’s original properties. It would be ideal for our harmonic enhancement to revolve around the fundamental oscillation of the object so that the sound generated is unique to the object.

Peak Enhancement

Hearing the Changes

Performing spectral mapping in an already harmonically rich source, such as an instrument, makes it easily discernible. However, if it were to be performed on an inharmonic object, the changes might not be translated. To circumvent this, a careful use of amplification and enhancement can be executed after the spectral mapping process. While extreme values can detach it from its original timbre, it should remain a user-controllable modifier so as to effectively hear the difference. There are several ways that could be used to control this enhancement, such as shaping the frequency bins around the shifted peaks, or even applying specially designed amplification curves. Using the former would be more aligned with the goal of keeping the output sounding more natural.

Summary of Background Research

Past and existing research regarding **Spectral Mapping**, **Harmonicity**, **Peak Detection**, and **Peak Enhancement** are key areas that will help us implement our system.

The notion of utilising everyday objects with products such as *Mogees* and *Looper* in terms of accessibility for musical and other purposes is very appealing. It opens up possibilities for experimentation and creativity that otherwise would never be witnessed. Not only this but also how they can be used within a live-performance environment which could create many unique and interesting experiences, experiences that are rare to see within the music industry.

TransFormSynth and *Transformer* are systems that are strongly regarded due to their spectral mapping execution. Emails exchanged between Andrew Milne and Bill Sethares, who developed these projects, helped us to confirm if we were on the right track with our goal of spectral mapping. Sethares noted the possibility of using spectral mapping in our project, adding that it could be challenging to remap objects containing complex spectra such as cymbals.

We have explained how harmonicity is defined, as it is an important principle that our spectral mapping system is based on. It gave us a clear direction on how to approach the system, where we will use the harmonic template based on the object's detected fundamental frequency to determine the remapping of spectral peaks. Since low harmonicity correlates to high entropy and vice-versa, our system converts these high entropy sounds into lower entropy sounds by amplifying the shifted partials, making the output more tonally coherent. The process will make the fundamental frequency more perceivable and psychoacoustically relevant, in turn fulfilling a portion of our objectives.

3. Design and Development

Overview

The concept of our system should be able to perform spectral mapping on an input signal to an output signal. Fundamentally, this would be achieved by means of operating on amplitude and phase spectra values frequency domain.

The system would require:

- Input and output processing
    - Framing
    - Windowing and de-windowing
    - FFT and IFFT
- Peak detection
    - Smoothing
    - Maxima and minima identification
    - Fundamental frequency extraction
- Spectral mapping
- Enhancement

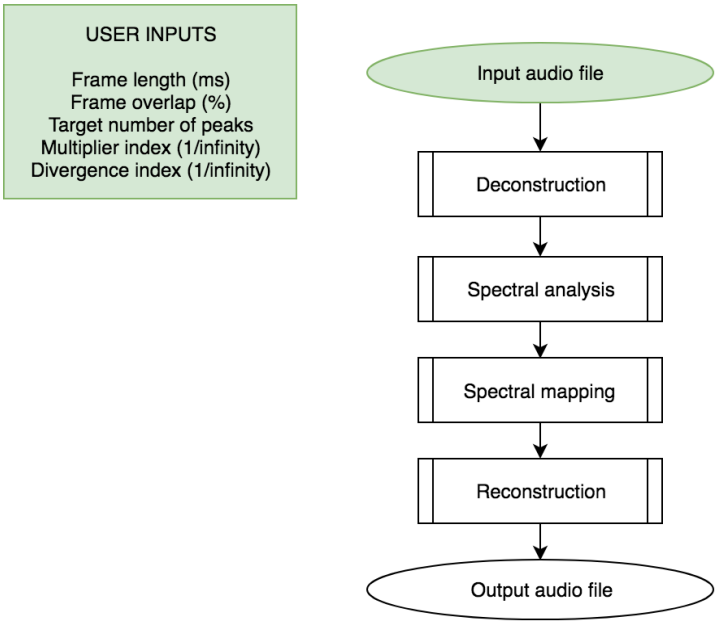

Figure 1: High-level system flow

System Requirements

Input and Output

Correct deconstruction and reconstruction of the input audio file would be a base requirement for further system design and testing. This would consist of both time-to-frequency and frequency-to-time domain conversions, to directly manipulate frequency content of a signal and play it back again. Such processes, which are common in any signal processing algorithm (especially in software implementations), require framing and windowing in order to operate properly (see *Deconstruction* and *Reconstruction* in *System Architecture*). Ideally, the input and output processes should be able to prepare an audio signal for spectral mapping and recover it with as little degradation or distortion as possible (more on that *Input and Output* in *Testing and Validation*).

Peak Detection

The detection of spectral peaks would be critical in forming the subject matter of spectral mapping. Due to the noisy nature of inharmonic signals and the large number of data points, a signal would need to be smoothed in order to filter out high-frequency fluctuations. Without smoothing, the number of peaks obtained would be too high for meaningful spectral mapping. This is due to the intrinsic nature of the program attempting to define a peak, which can change drastically depending on the type of sound source. As such, it is important to operate with the highest defined level of local maxima, and instead employ smoothing to refine peaks in the context of the entire spectrum.

Furthermore, eliminating noise via smoothing would make possible trends and patterns in the signal much more evident. The smoothed frequency spectra simply act as references to the peak locations and amplitude values in the original spectra, allowing the extraction of a fundamental frequency (lowest identified peak) and subsequent peaks for spectral mapping. Therefore, a smoothing algorithm will be an essential requirement for this stage of processing.

Spectral Mapping

Spectral mapping would function as the core process of the program, responsible for any kind of harmonic transformation. Andrew Milne sent us working Max patches and files associated with *Transformer,* and was able to confirm that each individual detected peak was only mapped to a single target partial. In contrast to this, our system design would be capable of mapping entire peaks along with its surrounding region, formed of multiple sinusoidal components. A self-contained spectral warping algorithm would be used to frequency-quantise peak regions to a nearest fundamental frequency multiple [11:75-76], but only if such a value exists within their own bounds. This is because a spectral entity with fixed endpoints can only be morphed so far, before it begins to exhibit invalid theoretical spectral behaviour by bending back on itself. In practice, this would result in unipolar component redistribution outside the scope of the original peak region, causing spectral discontinuities and corruptions.

Enhancement

Since inharmonic sounds naturally tend to contain high percentages of floor noise, a method of amplitude enhancement would need to be implemented to pronounce spectral mappings - particularly those at higher frequencies. The system should reflect the amplification in the right amount so as to retain a sense of its original timbre. Enhancement values will not be fixed, as it should be determined by the user so as to effectively hear the difference. This trade-off could result in an unnatural sounding result, but it would be much more preferred to be able to hear more than just a slight difference. Steps will have to taken to further refine our enhancement methods.

System Architecture

Deconstruction

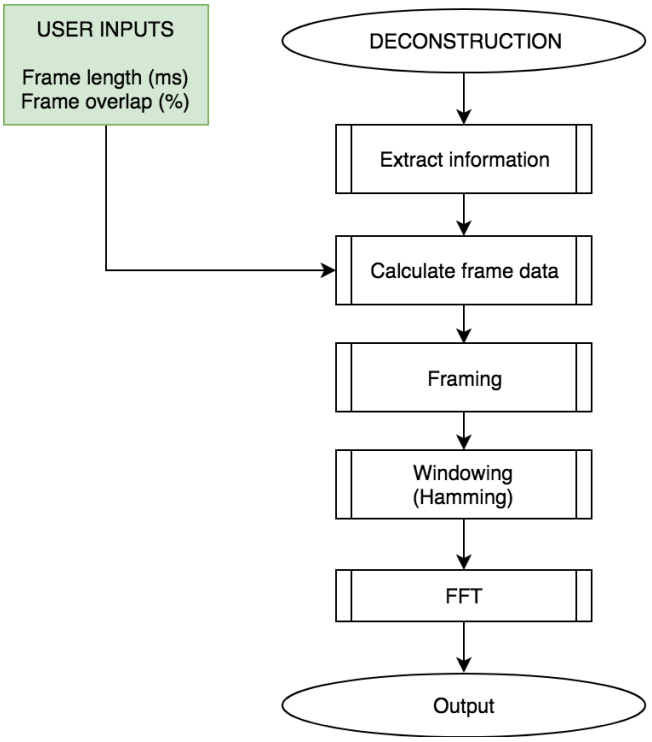

Figure 2: Deconstruction process flow

In addition to the number of audio samples, the sample rate of an input audio file was needed to properly calculate data for precise frame-by-frame deconstruction. This data includes:

- Frame length - the number of samples in each frame
- Frame overlap - the number of samples each frame was overlapped by the next
- Frame step - the number of samples between the starting position of each frame

Furthermore, framing is also an inevitable process in converting from time to frequency domain representations, since a single stationary FFT representation of an entire audio file does not factor in temporal or spectral changes. While a frequency domain representation of a single frame is stationary for any given input signal, the subsequent time domain representation of the overall output will be non-stationary, displaying the spectral development over time as a result of the overlapped concatenation of each consecutive frame.

Windowing is needed when overlapping frames, in order to avoid a “tremolo” effect in the output audio file. For the overlapping regions to have the same weight as the non-overlapping parts, a hamming window (similar to a bell-shape) is applied. More importantly, it attenuates both ends of each frame, making the frame’s samples more periodic. This additionally improves the performance of the FFT, which is supposed to be operating on a periodic signal [7:335]. For the FFT calculation process, it is worth mentioning that, due to the discrete nature of it, the frequency bin resolution (how many Hz are all the frequency bins spaced apart, which affects the spectral resolution of the spectral mapping process) depends both on the sampling frequency of the input audio file and on the sample length in samples (entered by the user). For the same sample frequency, a longer frame results in a better FFT frequency resolution:

$$frequency\ bin\ resolution = \frac{sample\ rate}{FFT\ size}$$

Spectral Analysis

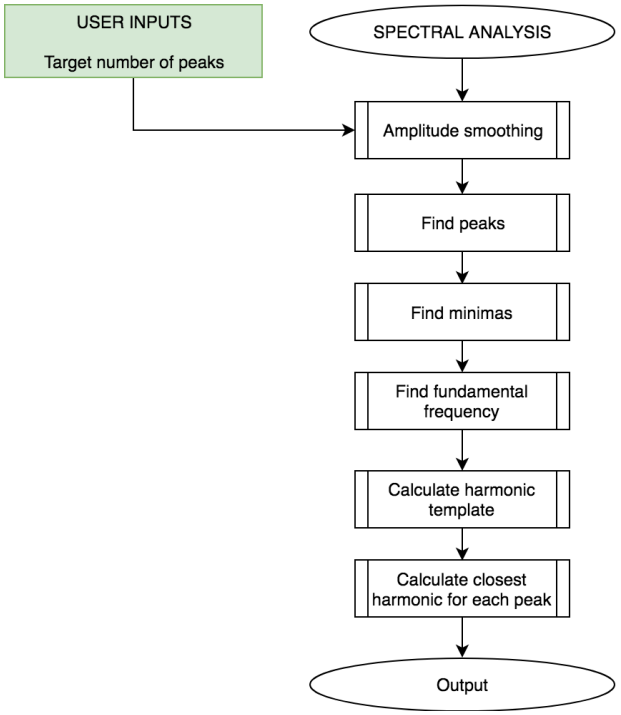

Figure 3: Spectral analysis process flow

Amplitude Smoothing

Spectrum smoothing is an iterative process which is carried out until a user input target number of peaks is met. The system utilizes a Savitzky–Golay filter [12] for each smoothing operation and always evaluates one pass ahead before reapplication. In this way, the program will decide to continue the smoothing process or not, by observing if the next smoothing pass will result in a peak count less than the target. The frame length argument for the filter is calculated as 1/64th of the elements in the spectrum, which means that consistent smoothing will be applied at each pass, for different files with the same combination of inputs.

Peak and Fundamental Frequency Detection

Points of maxima and minima are located in the final smoothed spectrum, through respective non-inverted and inverted analysis (maxima in inverted spectra become minima in non-inverted spectra). Subsequently, the system classifies a peak region as the data between any two adjacent minima indices. Neighboring peak regions share a minima point, such that they successively span the spectrum from the first to last identified minima indices. The first detected peak classifies the fundamental frequency for the current frame. Importantly, this value is a bin index, and subsequent processes operate without converting to frequency equivalents unnecessarily.

Spectral Mapping

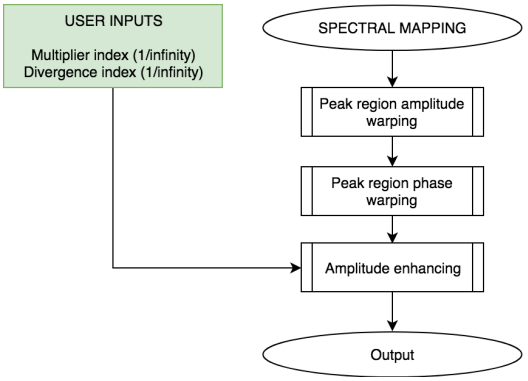

Figure 4: Spectral mapping process flow

The spectral mapping algorithm morphs each detected and valid peak and its surrounding region, such that an existing maxima index is quantised to its corresponding harmonic of the detected fundamental. A peak region is considered valid for spectral mapping if the original and destination peak both fall within of the region bounds by at least three indices. These constraints ensure that the minimum requirements for the interpolation processing are met, and the bordering indices are left unprocessed for seamless substitution of the peak region back into the spectrum. In short, the processing prevents abrupt forms of shifting from happening.

Initially, the unprocessed peak region is split into left and right parts at the maxima location. Interpolation is then performed to either side consecutively, in order to transform each data

set to new horizontal dimensions. The degree of interpolation for the left side is determined by the factor difference between the original and destination peak. The degree of interpolation for the right side is automatically calculated with respect to the left, so that appending both data sets results in an output the same size as the input. Upon insertion of the processed peak region back into its original position, this last step is crucial in preventing unwanted data discontinuities and constraint breaches in the context of the entire spectrum.

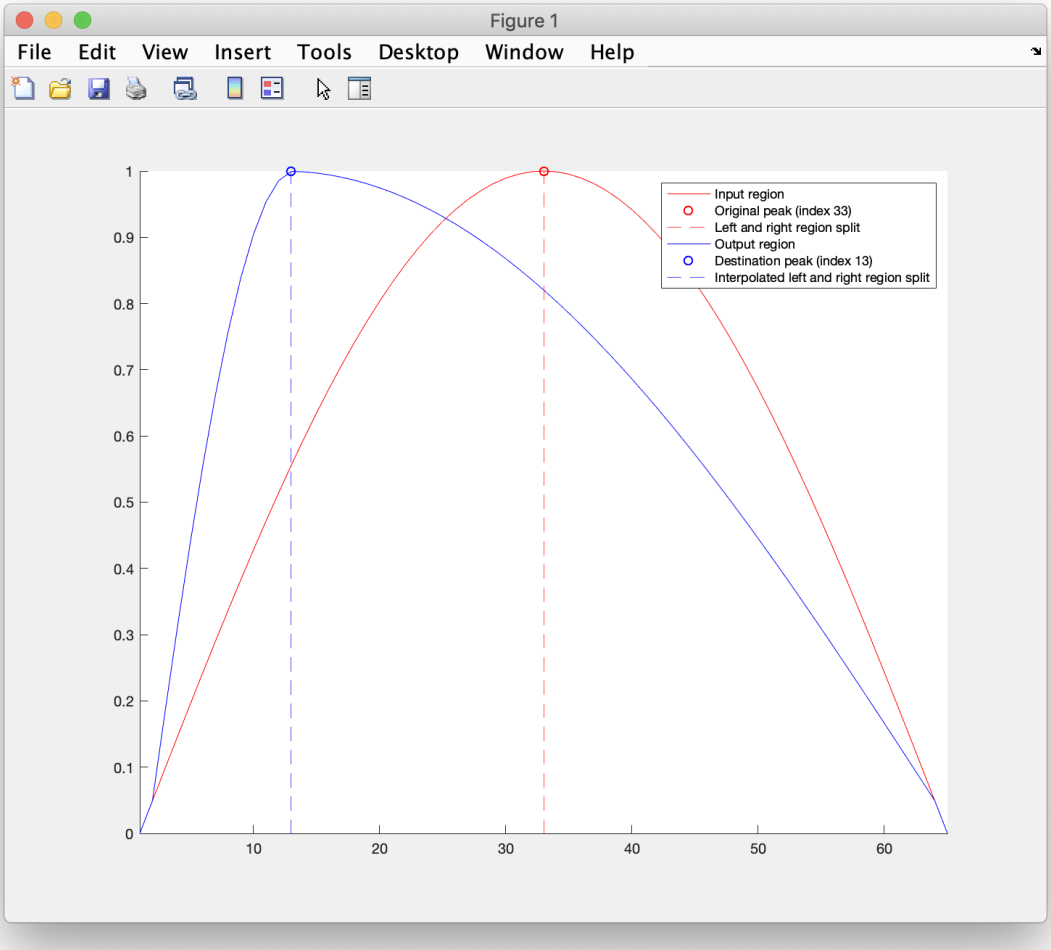

Figure 5: Transformation of an input signal peak to a specified destination through interpolation

Enhancement

The enhancement of all the shifted peaks is achieved through multiplication scaling. A user input defines the multiplication factor, which is automatically truncated such that peak region amplitude values do not exceed the largest existing partial in that frame. It functions as a limiter to prevent these shifted peaks from excessive amplification that could mask the quality of the original timbre.

Instead of amplifying each peak region by a constant value, a window function is applied to each spectral region, in order to prevent instantaneous transitions caused as a result of the

former method. Multiplying each region by a modified Bartlett window gives no multiplication at either edge (1x), and transitions into a user-defined amount at the region maxima point.

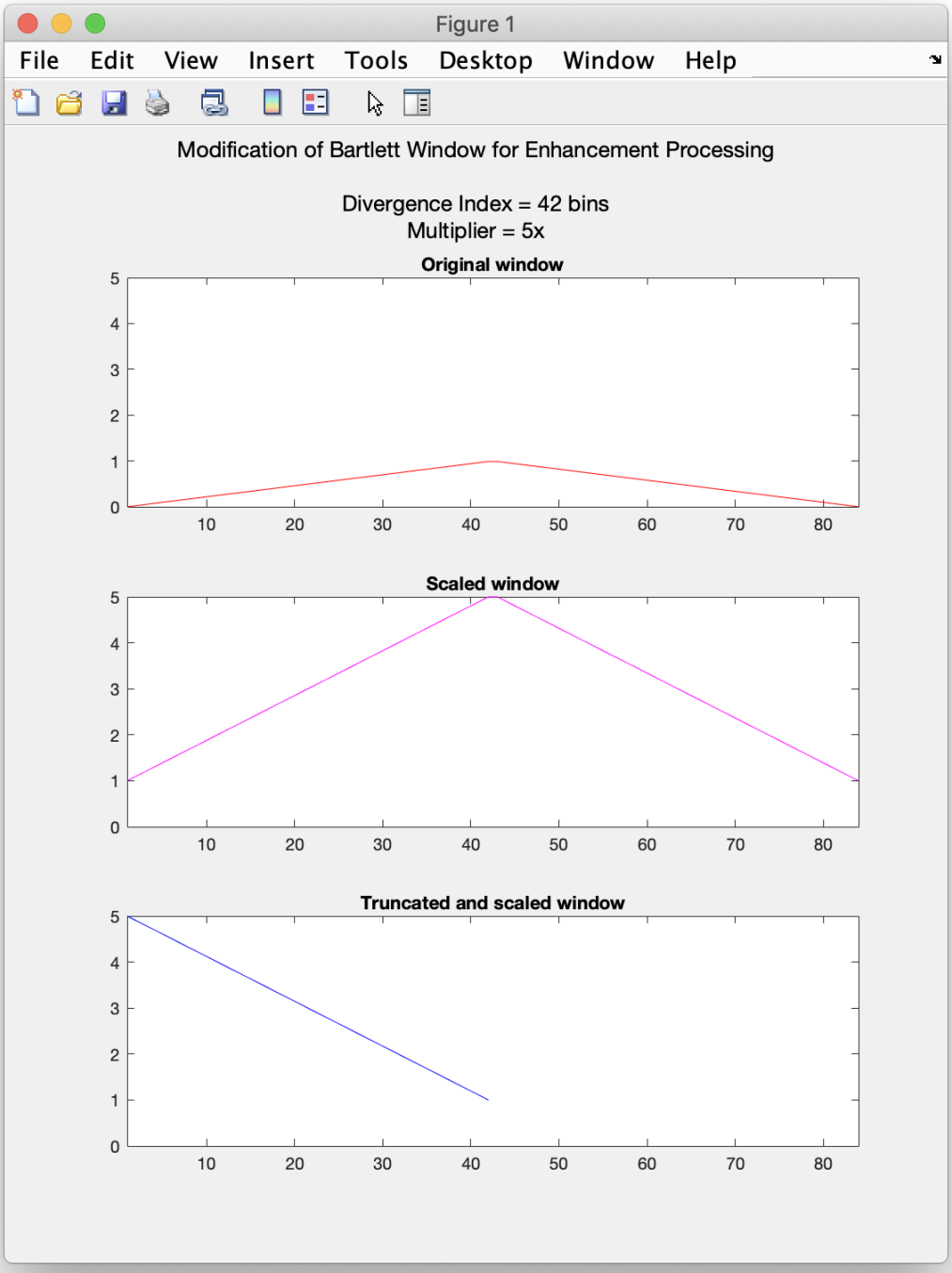

Figure 6: modification of a Bartlett window for enhancement processing

A divergence index value controls the overarching reach of the enhancement process over each peak region, through varying the length of the applied window function. For a given peak region, its local maximum value index is identified and the window function applied to

either side. The divergence is constrained to the bound closest to the maxima point, in order to prevent enhancement outside of the peak region.

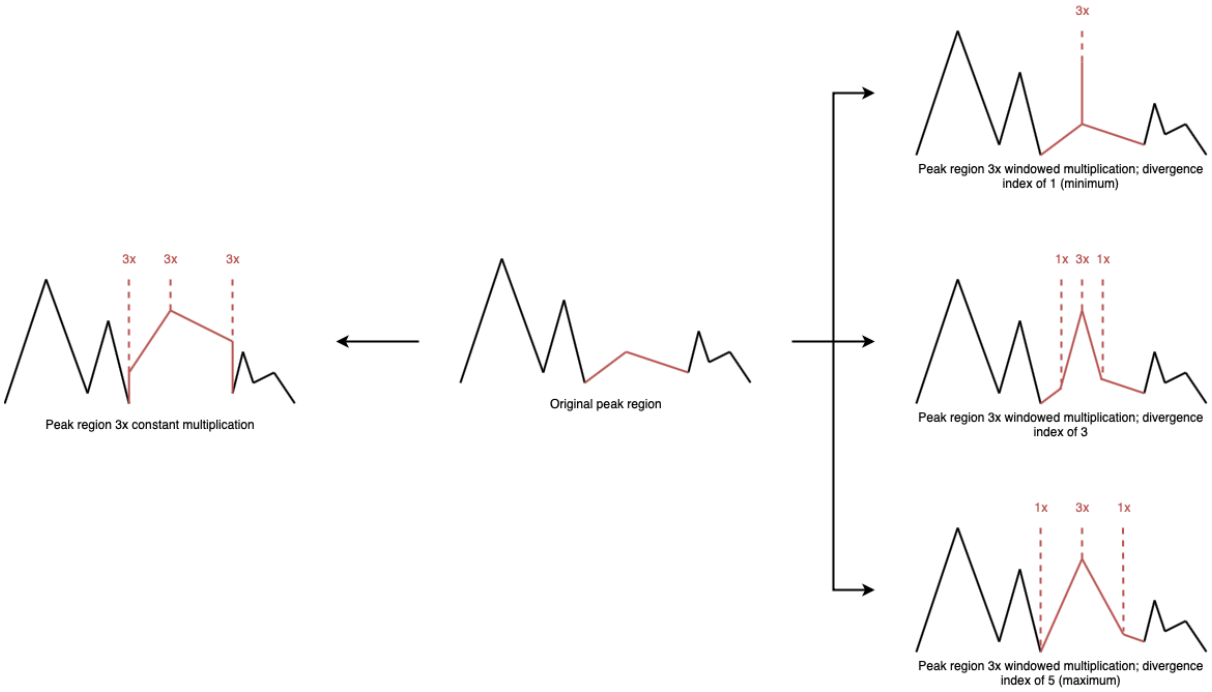

Figure 7: Demonstration of constant multiplication against the divergent windowed multiplication

Reconstruction

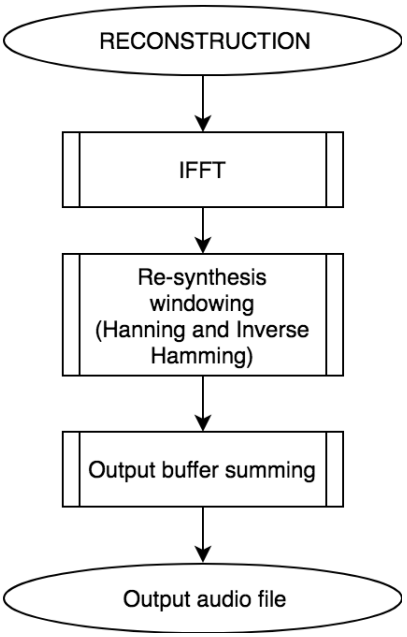

Figure 8: Reconstruction process flow

Once the spectral mapping process has been performed in the frequency domain, the current frame is then converted back to the time domain. In order to do this, the IFFT

computation operates on both the modified amplitude and phase spectra to produce the output.

As part of the output audio waveform reassembly process, a resynthesis window is applied before the insertion of a frame. This resynthesis window is calculated by multiplying a Hanning window with the inverse of the original Hamming window [13:26-28]. The shape of the Hanning window is determined by the frame overlap percentage (see Figures in Appendix A1-4). On application of the resynthesis, the overlapping portions of each frame are weighted less than the non-overlapping portions, such that linear equal amplitude is achieved in the final audio output waveform reconstruction.

The final output audio waveform is then successfully reconstructed simply by placing each frame at the point which it was originally extracted. Due to the sample weighting applied by the resynthesis window, overlapping samples are successively summed together as each frame is inserted. Since the number of total audio samples of the output audio file waveform should be identical with that of the input audio file (the sampling frequency and time durations are also identical), this process is carried out sample-by-sample.

4. Testing and Validation

Quantitative testing of the system was primarily done with MATLAB plots. These provided important visual feedback during debugging at each stage, allowing issues to be fixed quickly. Spectrum plots of each processed frame verify that spectral mapping and enhancement are working as intended.

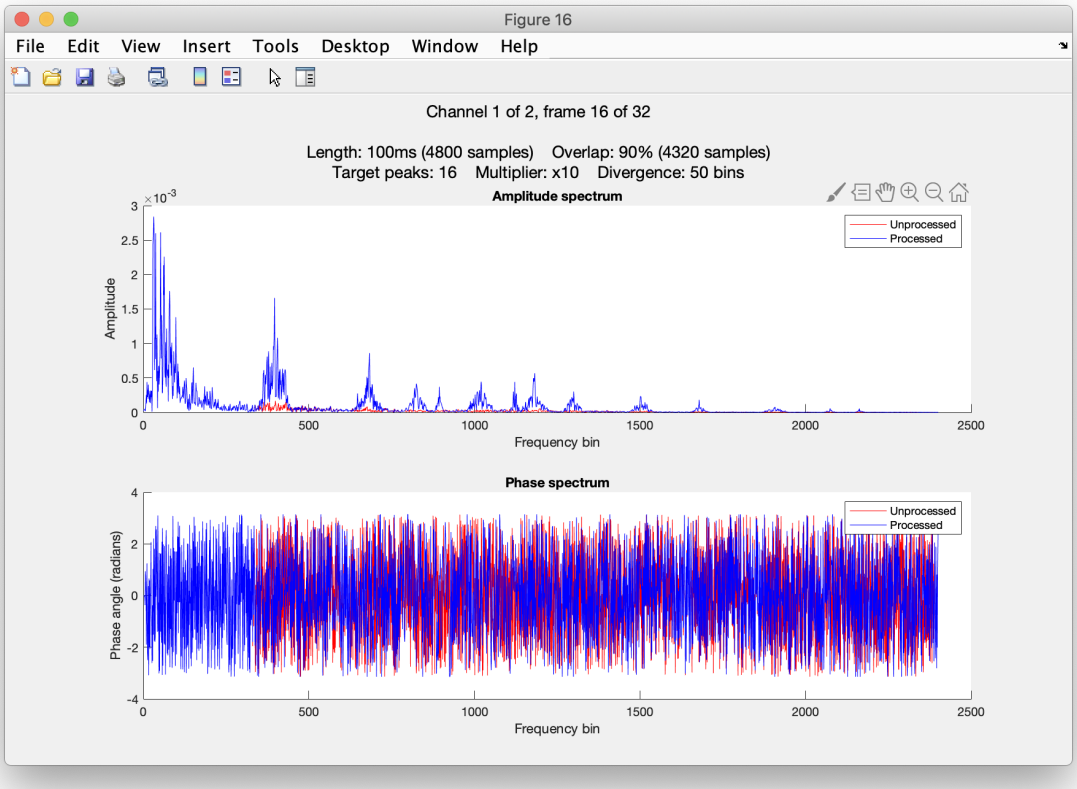

Figure 9: A spectrum plot showing the differences between the unprocessed and processed signal

Input and Output

We ran the first tests on the Deconstruction and Reconstruction parts, prior to designing the Spectral Mapping algorithm; in addition to listening tests (the output sound file had to sound exactly as the input sound file), we checked the numerical values of the input and audio waveforms samples to be identical, and finally we ran visual tests on the plots of both waveforms. Only when all these tests were successful we started to design the Spectral Mapping algorithm.

Testing Results

The controllable parameters each had a sweet spot depending on the source material. Generally, a shorter **Frame Length** decreased the high frequency spectral resolution of the input and resultant output file, while decreasing the **Frame Overlap** percentage resulted in more discontinuities and a choppier sound. The **Target Number of Peaks**, **Peak Amplification Multiplier**, and **Divergence Index** should be used in combination to find the ideal sound. Too high a target number of peaks along with a higher amplification amount will result in an artificial sounding output.

After testing the system with various inharmonic samples, it seems that short bursts of high frequency based samples such as hi-hats and white noise contain tonal properties that are more difficult to hear accurately in the output sound. This is perhaps due to the fundamental frequency being too high, mentioned briefly by Plack [9:125] where he describes from research that ‘a fundamental frequency less than 5 KHz is necessary for a clear pitch’. An explanation of this could be formed in how it is very hard for the human ear to distinguish separate partials at high frequency ranges, as supported by the Mel Scale [14]. An example of this would be where the input sound has a fundamental frequency of 5KHz, in which case the second and third harmonics are of 10 KHz and 20 KHz respectively and thus the pitch would be hardly identifiable. A crash cymbal however, produced better results perhaps due to the way it resonates. We were successful at mapping its many partials into providing consonance. Processing a harmonic sample such as a musical instrument resulted in a higher harmonic content similar to using an harmonic exciter plugin.

MATLAB Audio Test Bench (VST Development)

The Audio Test Bench, which is part of MATLAB’s Audio Toolbox, was used as a form of audio plugin validation as well as an emulation. It offers a quick solution to prototyping and testing of our spectral mapping program. Prior to the emulation, the software is ran through Audio Test Bench’s stringent validation test. Upon passing this test the plugin emulation can be executed within MATLAB, with the option of exporting into a Virtual Studio Technology (VST) plugin to be hosted in a DAW environment.

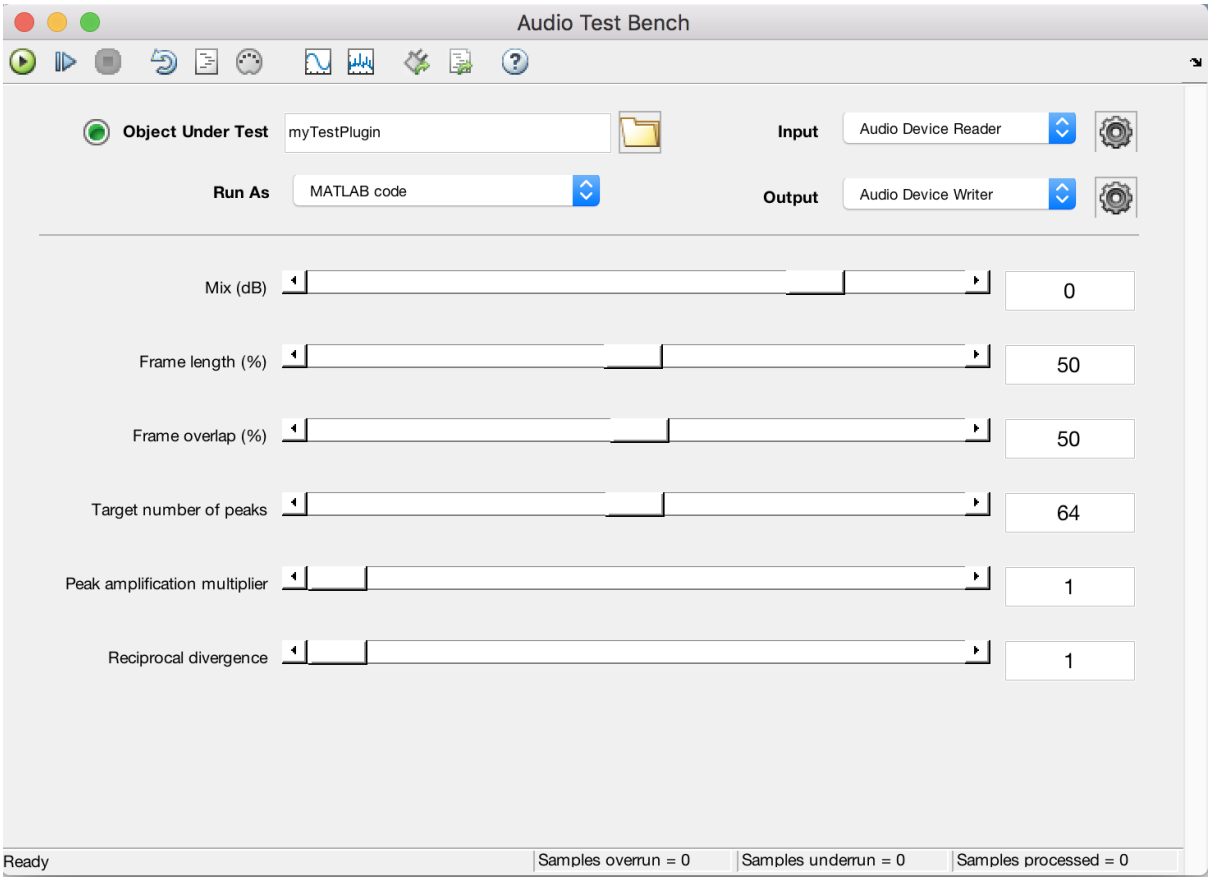

Figure 10: Prototyping our program in MATLAB Audio Test Bench

The Audio Test Bench enables us to present the program in a way that the end user will be able to understand better as compared to manually changing lines of code and re-executing the program. The selectable options are available to be tinkered with in the form of sliders, which changes the sound in real-time in this plugin emulation. For this we have unanimously chosen the ability to change the **frame length**, **frame overlap percentage**, **target number of peaks**, **peak amplification multiplier**, and **reciprocal divergence**. While the emulation is running in real-time, it is still possible to modify code in the program. These options allow us or the user to perform quick prototyping limited only by the computational power of the end user. It includes other developmental tools such as a time scope, a spectrum analyser, as well as other user-customisable visualisation.

5. Conclusion

We have developed a system that transforms samples of real-world inharmonic objects into harmonic ones as reflected in our testing. The system has also fulfilled the goal of ensuring the resultant sound stays true to its original sample's characteristics.

At this stage its full computational power can be harnessed in MATLAB, but when exported as a VST, its lightweight form will allow for an easier experimentation experience. Despite not strictly being a real-time system, the transformed output signal can be used with a sampler in a live-performance environment. In our interim report one of the goals was to prototype this system in software, and this has been realised in the form of a VST plugin and MATLAB program. This system could prove highly useful to experimental sound artists who are looking to expand their sound library.

In this project we began to hit several halts in progression due to a lack of specialised knowledge in the signal processing field, there are a couple of ways in which we can improve upon which will be discussed thoroughly below.

Technical Discussion and Challenges

We made our important first dig into the universe of digital signal processing algorithm design, which was particularly difficult because it was a programming-based and research-oriented project where we needed to recalibrate the same algorithm many times. Since this was a group project, we learned how to write good code in addition to commenting and formatting which had to be readable and reusable by other programmers/colleagues.

One path that we took was the decision of building upon Sethares' *TransFormSynth*. There are now key differences between both software processes. One is that *TransFormSynth* map partials to single sine components where ours deals with multiple components. Moreover, the main difference over Sethares' is how we have incorporated warping. Other changes include simpler user controls in its VST form and the fact that we use the calculated fundamental frequency for each frame as opposed to changing the fundamental according to the user.

Although similar in usage, both systems use different platforms. An aspect of MATLAB is that you have relatively easy low-level control meaning that we could more specifically control what was happening within the different processes. Max however tends to utilise objects with supported by external Javascript code to execute complex processing alongside Max.

General Processing

A big challenge was finding a balance between keeping the original sample's characteristics and processing speed as one must be compromised for the other. To determine this, a trial and error process had to be conducted which ultimately meant that we had to create a non-real time system as only prioritising speed would have resulted in an unwanted, changed sound.

Early versions of the project performed each processing stage on every frame simultaneously. Matrices were employed to handle data in this format, as they could represent frame spectra over time. With a higher number of frames and greater frequency resolution, these matrices became increasingly demanding on program memory.
To combat this, sequential frame processing was implemented so that only one frame was ever passed through processed at one time. For each frame, variables were reused at each stage of analysis, spectral mapping and reconstruction, which mitigated memory issues substantially. The output sample vector was also sequentially constructed, where it was written to after each frame pass.

Spectral Mapping

Early implementations of our spectral mapping method included the summation of band-rejected and band-passed spectra, as well as linearly frequency shifting chunks of spectral content. Upon further testing, these concepts were flawed in that processing a singular peak was destructive to other spectral content. In sounds even slightly more complex than a few sinusoidal components, these approaches will not be practical they begin to add spectral holes and overwrite existing frequency content; worsening as more spectral mappings were attempted. As such, it became apparent that to achieve both characteristic retainment of the input signal through non-destructive processing, a peak would need to be mapped via inter-region warping over traditional shifting.

Evaluation and Critical Analysis

Output

Although perhaps subjectively not the most pleasant sounding, the output sound is a result of staying true to the original sample's characteristics, and therefore, it was allowed to dictate most of the decisions made when building the system. Because of this, it ultimately did not matter what the output sound turned out to be as that was what we set out to do. The system transforms the sample according to our presented definition of harmonicity.

With that being said, the system works most effectively when the input file is inharmonic, similar to the sounds included in testing such as cymbals, snare drum etc. which is what the system was intended and designed for. It is entirely possible there are unforeseen cases where an inharmonic sample is not mapped and transformed correctly.

The aforementioned problems could be ironed out but as previously mentioned, the system was only intended to transform real-world inharmonic objects and not with unrealistic cases in mind. If the system were to be used in a live-performance environment, this would not pose much of a problem.

Efficiency and Accuracy

The system is incredibly fast but perhaps not fast enough. However, It would have been extremely difficult to create a system where it produces the output file in real-time as a large

number of processes are performed in the algorithm. This could possibly just be a case of slightly inefficient code where given more time, could potentially be improved. An example of this is the smoothing process, which requires multiple passes to reduce a signal to an arbitrary number of peaks. This stage can be sped up considerably by smoothing a higher proportion of the signal at once, at the cost of major resolution changes at each pass (i.e. how many peaks will be identified).

Additionally, the locations of peaks in each smoothed spectrum do not necessarily conform to those in the original equivalent. Rather, they point to areas of collective maxima over a single strict outstanding component. For sounds rich in harmonic content, this is not necessarily an issue. In testing of a sinusoidal signal with a few distinct components, however, the identified peaks were not mapped correctly to their closest harmonic template index. In hindsight, it would have possibly been better if we had instead decided to continue with developing our own peak identification function. It perhaps would have recognised each peak more efficiently.

One disadvantage of its current implementation is that there could be a case where a very big file is imported and thus as a result, it would take much longer than anticipated to process the number of frames that it includes.

Fundamental Frequency Detection

Our system relies on the lowest detected peak at each frame to dictate the fundamental frequency and following harmonic template. Inharmonic sounds are known as such because they are non-periodic, so identifying a fundamental frequency in this way is somewhat redundant because the value is sometimes inconstant, Consequently, template spectral mappings conform to are inconsistent, often resulting in the lack of a perceivable pitch.

This is mentioned and interpreted in the **Testing** section where the fundamental frequency (in every frame) is limited to 5 KHz for the sample to turn harmonic. Despite being the case, it is unlikely that a sample like this would be imported into the system as most real-world inharmonic objects do not have a (though not constant) very high fundamental frequency to begin with.

To counteract this, a better approach may have been to pre-process the input file to find a mean or modal lowest peak value, to then apply a fixed harmonic template for spectral mapping over every frame.

Spectral Mapping

Due to the nature of nearest neighbor mapping and the frequency response of inharmonic sound, many lower elements of the harmonic template are often not mapped to. The perceivable pitch of a sound is generally established by only a few of its initial harmonics, whereas the timbre is a product of the majority [9:126-8]. This is why the system sometimes seems to only colour the input signal; especially at a lower number of target peaks (more smoothing; identified peaks are distributed further apart). In this case, sequential alignment

mapping [11:75-76] is not a solution as data corruption via repeated warping over existing spectral mappings would inevitably occur.

Alternatively, some kind of non-destructive method of frequency shifting spectral regions through intelligent component summing would fix this issue, as constraints for mapping within a fixed area would not need to exist. It could be argued, however, that the manipulation of non-mapped data in this way is not congruent with the retainment of underlying characteristics of a sound - a key aim of this project.

Possible Future Directions

For future work, it would be appealing if the system could be turned into a physical product with sensors attached to capture audio input. This would be a portable product which could be used anywhere to spontaneously morph objects into instruments providing creative outcomes. A real-time implementation can be realised with faster processing speed such as an FPGA platform.

User-selectable windowing types could also be an additional option as it would allow the user to produce different fft output spectra shape. Adding certain lower harmonic components artificially could also be introduced in order to define the pitch of the sound to a greater degree, although this would go against the principles of staying true to the object. The concept of Xentonality can be considered where for a given input spectrum, its most consonant musical scale can be expressed. It is usually performed by calculating the dissonance curve [7:97-104], where a different output audio file is produced for each pitch present in the scale, with fundamental frequency equal to the frequency of each different tone.

There are several ways to expand on the amplification section. We could develop the system further in such a way to allow the user to decide the amplification shaping, modelling after wave characteristics such as square, sawtooth, and triangle. A specific configuration of odd and even harmonics can be adjusted to provide richness in the harmonicity. If performed with good judgement these improvements could add a different harmonic flair.

Division of Workload

Team Member	Sections

Matteo	• Background Research
	  ○ Key Literature
	  ■ Spectral Mapping
	  ■ Harmonicity
	  • Harmonic Entropy
	• Design and Development
	  ○ System Requirements
	  ■ Input and Output
	  ■ Peak Detection
	  ■ Spectral Mapping
	  ○ System Architecture
	  ■ Deconstruction
	  ■ Reconstruction
	• Testing and Validation
	  ○ Input and Output
	• Conclusion
	  ○ Challenges
	  ○ Potential Future Directions
	• Other
	  ○ UML flowcharts

Reuben	• Introduction
	  ○ Aims and Objectives
	• Design and Development
	  ○ Overview
	  ○ System Requirements
	  ■ Input and Output
	  ■ Peak Detection
	  ■ Spectral Mapping
	  ■ Enhancement
	  ○ System Architecture
	  ■ Spectral Analysis
	  • Amplitude Smoothing
	  • Peak and Fundamental Frequency Detection
	  ■ Spectral Mapping
	  ■ Enhancement
	  ■ Reconstruction
	• Testing and Validation
	• Conclusion

	○ Technical Discussion and Challenges ■ Spectral Mapping ■ General Process ○ Evaluation and Critical Analysis ■ Efficiency and Accuracy ■ Fundamental Frequency Detection ■ Spectral Mapping ● Other ○ MATLAB plots ○ Diagrams
Zai	● Abstract ● Introduction ○ Overview ○ Aims and Objectives ● Background Research ○ Key Literature ■ Spectral Mapping ■ Harmonicity ■ Peak Detection ■ Peak Enhancement ○ Summary of Background Research ● Design and Development ○ System Requirements ■ Spectral Mapping ■ Enhancement ○ System Architecture ■ Spectral Mapping ■ Enhancement ● Testing and Validation ○ Testing Process ○ MATLAB Audio Test Bench (VST Development) ● Conclusion ○ Technical Discussion and Challenges ○ Evaluation and Critical Analysis ○ Potential Future Directions
Rhun	● Introduction ○ Overview ○ Aims and Objectives ● Background Research ○ Existing Developments ■ Mogees ■ Looper ○ Key Literature ■ Harmonicity

	■ Spectral Mapping ● Design and Development ○ System Requirements ■ Overview ○ MATLAB Audio Toolbox (VST Development) ● Testing and Validation ○ Testing results ● Conclusion ○ Overall Summary ○ Challenges ○ Evaluation and Critical Analysis ○ Potential Future Directions

Appendix

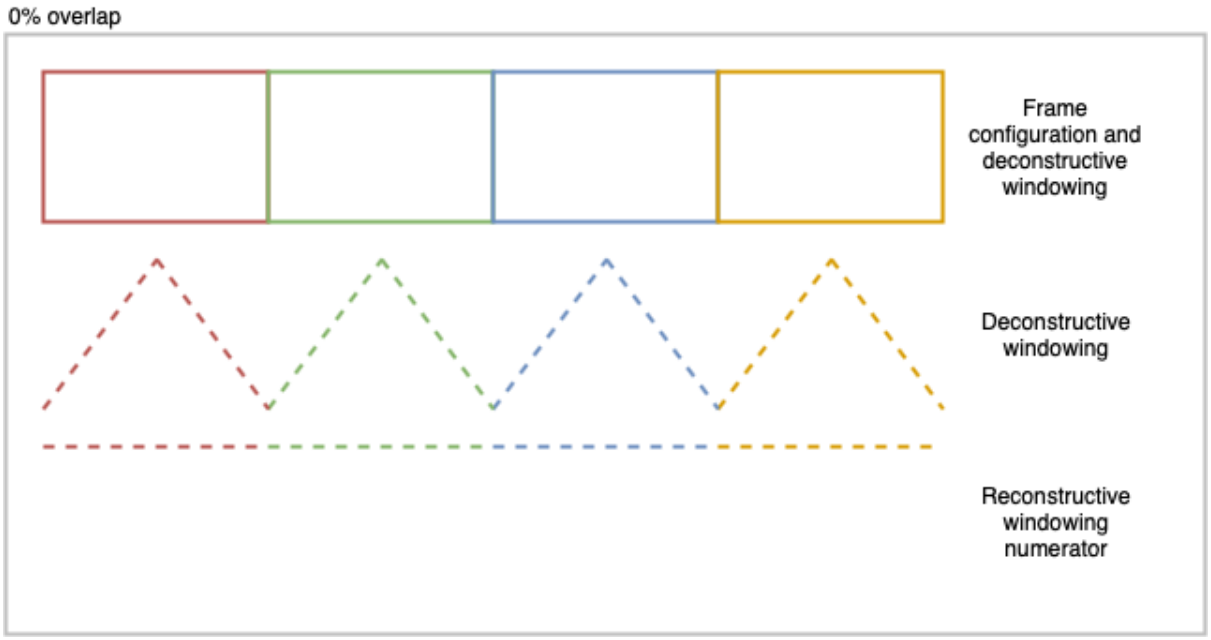

Figure A1: Demonstration of windowing at a frame overlap of 0%

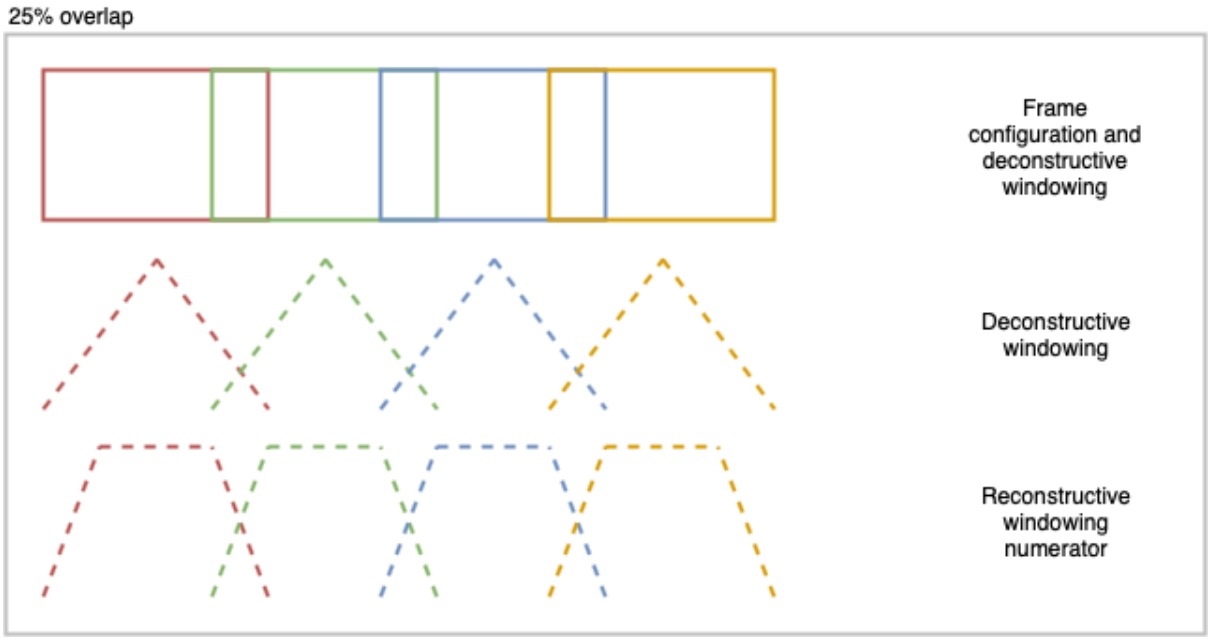

Figure A2: Demonstration of windowing at a frame overlap of 25%

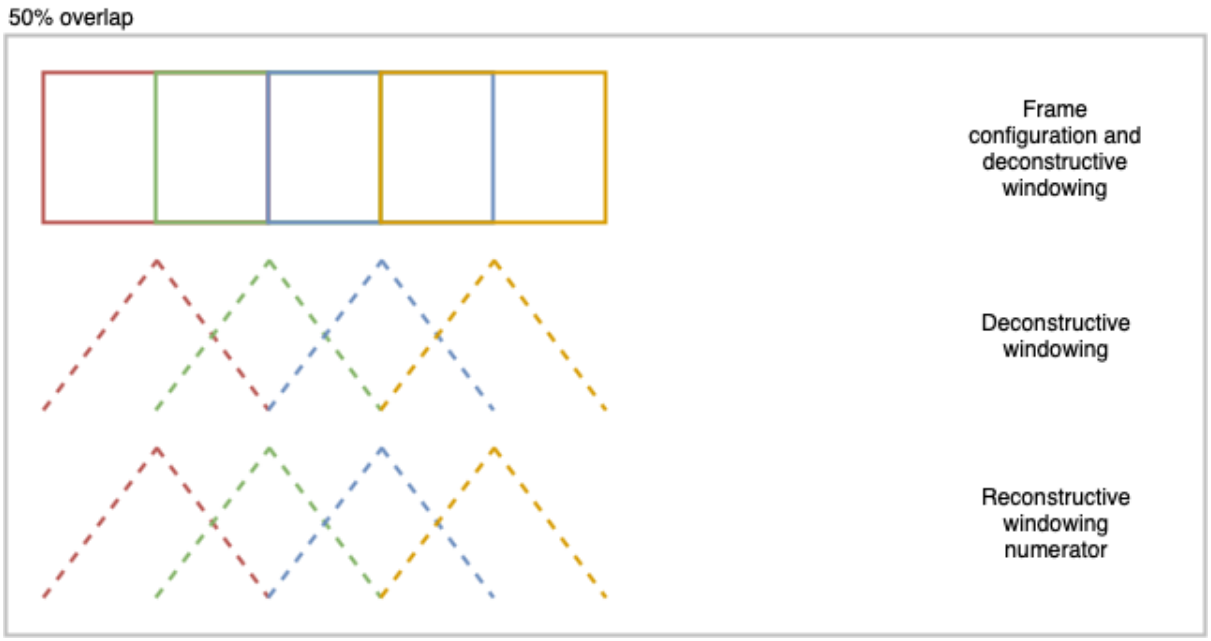

Figure A3: Demonstration of windowing at a frame overlap of 50%

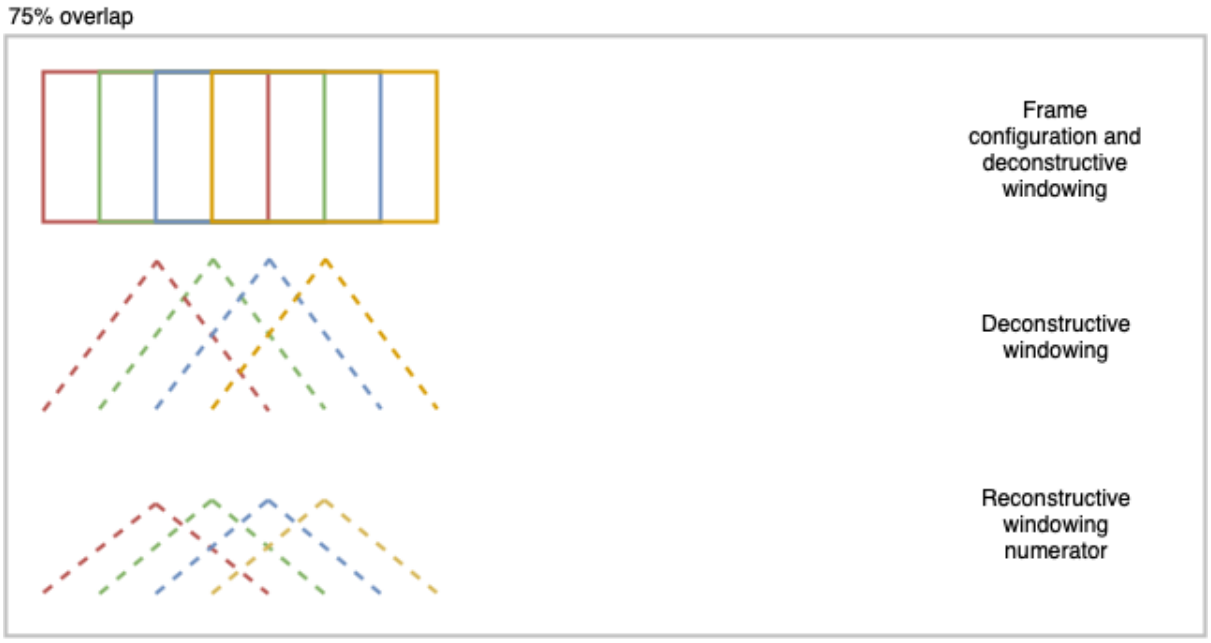

Figure A4: Demonstration of windowing at a frame overlap of 75%

References

[1] B. Zamborlin, B. Caramiaux, and C. C. Emanuele, “Method to Recognise a Gesture and Corresponding Device,” 2017.

[2] L. Siegele, “Looper - Transform any object into a sound body,” *lukassiegele.com*, [Online]. Available: http://lukassiegele.com/Looper.html. [Accessed Apr. 13, 2018].

[3] Bela, “An Ensemble of Instruments by Fedde ten Berge: of Nature and Things” *blog.bela.io*, [Online]. Available: https://blog.bela.io/2017/12/12/fedde-ten-berge-instruments-bela/. [Accessed May. 04, 2019]

[4] J. Woodhouse, “What makes an object into a musical instrument?” *Plus*, 03 Feb., 2011.

[5] A. J. Milne, R. Laney, and D. B. Sharp, “Testing a Spectral Model of Tonal Affinity with Microtonal Melodies and Inharmonic Spectra,” Music. Sci., vol. 20, no. 4, pp. 465–494, 2016.

[6] J. H. McDermott, “Auditory Preferences and Aesthetics: Music, Voices, and Everyday Sounds” in *Neuroscience of Preference and Choice*, R. J. Dolan and T. Sharot, Eds. New York: Academic Press, 2011, pp-227-256.

[7] W. A. Sethares, “*Tuning, Timbre, Spectrum, Scale”,* London: Springer London Ltd, 2005.

[8] F. Kuratani, T. Yoshida, T. Koide, T. Mizuta, and K. Osamura, “Understanding the effect of hammering process on the vibration characteristics of cymbals,” J. Phys. Conf. Ser. 744, 2016.

[9] C. J. Plack, *The Sense of Hearing*, 2nd ed. London & New York: Psychology Press, 2014.

[10] D. Gerhard, “Pitch extraction and fundamental frequency: History and current techniques,” in *University of Regina*, 2003, pp. 0–22.

[11] W. A. Sethares, A. J. Milne, S. Tiedje, “Spectral Tools for Dynamic Tonality and Audio Morphing”, Computer Music Journal, vol. 33 - n. 2, pp. 71-84, Sum 2009.

[12] MATLAB, “Savitzky-Golay filtering.” [Online]. Available: https://uk.mathworks.com/help/signal/ref/sgolayfilt.html. [Accessed: 08-May-2019].

[13] U. Zolzer, “DAFX: Digital Audio Effects”, Second Edition, Hamburg: John Wiley & Sons, 2011.

[14] S. S. Stevens, J. Volkmann, “A Scale for the Measurement of the Psychology Magnitude Pitch” *The Journal of the Acoustical Society of America*, vol. 3, issue 3, 1937.