TIME-VARYING AZIMUTH DISCRIMINATION AND RESYNTHESIS: A NEW METHOD FOR MUSIC REPURPOSING

Salil Apte

Department of Computer Science
Brown University
salil@cs.brown.edu
September 2005

1 ABSTRACT

By extracting one or more instruments from one piece of music and mixing them with instruments from another, a professional DJ is able to create a new, "remixed" music track. As a result, there has been a growing need for instrument-only tracks fueled by amateur DJs wanting to mix their own music. In this paper, we discuss an algorithm for improving upon a previous method of separating instrument tracks from a piece of music. We also discuss how effective the previous separation is compared to this improved method by introducing a Euclidean-based wavelet distance metric. Lastly, we illustrate how sound separation can be used to remix arbitrary tracks of music without having the individual instruments *a priori*. All the example instrument separations and remixes in this paper can be found on the project website at:

http://www.rasik.com/salil/html/index.php?name=Newscatid=2

2 INTRODUCTION

The research in this paper helps to ameliorate the current state-of-the-art sound separation techniques by using Fourier analysis and wavelet decomposition. The general class of problems in sound separation involves trying to separate independent sources from mixed observed data. In other words, the goal is to recover the individual signals from a mixed signal. The algorithm being presented in this paper is a special case of the sound separation problem, specific to musical recordings.

The need to separate sounds from a mixed signal is an old one and has many implications in areas as far reaching as signal processing, medicine, etc. [12]. Although the signal sources differ in each one of these cases, it is common to want to unmix a signal and analyze the constituent sources individually. In the case of musical recordings, instrument separation has many uses [10]. For instance, by separating the trumpet from a recorded symphony, it might be easier a computer algorithm to transcribe the notation. Indeed, it has been noted that even humans have difficulty in transcribing an instrument in the presence of other instruments. Thus, by separating an instrument, we can make the transcription task easier. A separated instrument can also be used for cataloging. By separating an instrument from a piece of music, we can catalog our output for later retrieval. Searches can be done on the melody of a single instrument, for instance. Without instrument separation, this would not possible.

An important third use, and the one we focus on in this paper, is the case of DJ remixes. In today's music, the role of the DJ has changed dramatically. While a DJ from decades ago merely played back exisiting recordings sequentially, today's DJ can mix two or more recordings at the same time thereby creating a completely new musical performance [4]. Although a DJ can always mix two existing recordings, he/she will usually want to select individual instruments from different recordings and mix them together. This unfortunately requires access to the original instrument-only tracks before they were mixed together into a final song. To complicate the matter, getting access to these instrument-only musical tracks from the record companies is difficult due to copyright issues. Due to these obstacles, DJs only have access to the instrument-only recordings that the music industry publishes. At present, the recordings that are released in this instrument-only form are restricted to certain genres or types of music. The Azimuth Discrimination and Resynthesis (ADRes) algorithm solves this problem by allowing an amateur DJ to separate instruments from arbitrary pieces of music. In its current state, however, ADRes leaves certain extraneous audio artifacts after it performs the instrument separation.

In this paper, we will discuss a way to remedy some of the audio artifacts produced by the Azimuth Descriminiation and Resynthesis approach through the usage of wavelets. First, by detecting when an instrument comes in and goes out, we can selectively resynthesize the instruments in a time-varying fashion in order to yield a higher quality separation. Secondly, we use the wavelet coefficients to define a metric by which we can compute the quality of the separation mathematically. Finally, we discuss how a DJ would be able to use instrument separation to remix music without having access to the record company owned instrument-only tracks.

3 RELATED WORK

There has been extensive research in the area of sound separation. The main areas of research can be grouped under two headings: ICA and CASA.

By assuming a linear mixture model for the sources being separated, Independent Component Analysis (ICA) techniques attempt to demix a signal by guessing the mixing parameters that are used to arrive at the final output. ICA assumes the sources to be non-Gaussian and statistically independent for the process to work. Finally, traditional ICA requires the number of observations to be equal to the number of sources, but recent work has eliminated this requirement [14].

The three main algorithms used in ICA are the Infomax algorithm, FastICA, and JADE. All three approaches follow a similar princi-

ple – they attempt to reproduce the output recorded by an array of sensors by trying to recover the mixing coefficients as well as the sources being mixed by iteratively tweaking parameters. The Infomax algorithm [1] assumes the input sources to be statistical probability density functions and tries to guess the pdfs as well as the mixing parameters using a maximum likelihood esimation (MLE) technique. FastICA [11] makes the algorithm quicker by first prewhitening the data meaning it tries to decorrelate the data using PCA (Principal Component Analysis). Then, FastICA makes the observation that since our sources must be non-Gaussian, they must have a small kurtosis. The algorithms opts to use a measure that closely approximates the kurtosis called negentropy. The algorithm then tries to find a set of basis functions which might transform the pre-whitened data to a set of axes which are as independent as possible. The independence property is measured using the contrast function. This set of independent basis function then corresponds to our non-Gaussian sources of minimal kurtosis. Lastly, the JADE algorithm [8] provides another possible contrast function. Instead of explicitly pre-whitening the output data, it works from the statistics of the data using the second-order cumulants. It then tries to make the data as independent as possible using an approach called Joint Diagonalization. In all cases, the advantage of ICA is that it requires no parameter tweaking on the part of the user. Unfortunately, because of the high statistical dependence of musical instruments and ICA's requirement of non-Gaussian, independent sources [8], we are not able to use these techniques for instrument separation.

Besides the ICA approach, Computational Auditory Scene Analysis (CASA) takes a more psycho-acoustical and perceptual approach by trying to classify the independent sources by features such as fundamental frequency and common onset. The set of features used to classify the independent sources parallels the human's approach to separate sound [17]. Because of the rudimentary nature of these features, the system works best for simple sound sources like speech and tends to fall apart for sources which overlap in the frequency domain, such as musical instruments. As a precursor to CASA, Bregman [3] developed the idea of Auditory Scene Analysis by modeling the human's perception of sound as a method for sound separation. Computation Auditory Scene Analysis then describes how computers can be used to compute ASA. Early work in [9] and [5] try to separate sounds by using a flow-based CASA approach. The idea is to model the human ear as stages and to get the final output only at the end. The flow starts with modeling the cochlea of the ear by using filter-banks to separate the input sound into different response ranges. It then attempts to structure the sound into atomic components such as those with the same fundamental frequency or a similar frequency onset/offset. After this stage, the algorithm attempts to group these components together based on special rules. Lastly, it resynthesizes these groups to get the individual sounds. The work in [10] improves on these early techniques by introducing a predicitive-model that separates sounds at each instance and keeps a running likeliness of each sound separation. Much like the human perceptual system, the paper keeps an accurate account of the sound separation at each instant.

There exist other algorithms that do not quite fit into these headings but are alternative attempts at the instrument separation problem. One large area of research looks at breaking up the frequency-time plot of a recording into subspaces of typical instruments. For instance, certain instruments tend to stay within frequency pockets and by filtering these pockets in a smart way, it is possible in some cases to attain the instrument separation desired. These techniques tend to use some learning method such as Non-negative Matrix Factorization [22] as a means to understanding the relationships of instruments and their corresponding subspaces in the frequency-time domain. Another learning-based approach [21] borrows the mixture estimate ideas from ICA. They pose the instrument separation problem as a multiple-cause mixture problem. By assuming the instruments (or causes) are all mixed in a linear fashion, the algorithm attempts to estimate the volumes of the instruments. It requires a database of instruments that might be used in the recorded output and then it guesses the mixing parameters using a neural network whose objective is to get as close as possible to the recorded output in a least-squares sense. Although the algorithm works well for instrument separation, it is slow and more importantly, it requires a catalog of instruments which it can then mix to produce an output at each iteration of the algorithm. This database requirement is impractical in the general case of the instrument separation problem where the characterstics of a instrument vary from recording to recording. Finally, we are seeing some early work in using unsupervised machine-learning techniques like LLE to understand the underlying structure of sound and music. Using LLE, the authors in [] use LLE to project a high-dimensional monophonic instrument track into a lower dimension to see if they can visualize any underlying structure in the music. While the lower dimensional representation only seems to describe some rudimentary characteristics about the pitch and harmonic structure of the music, it is possible that future techniques might use LLE to understand the structure of the mixed instruments and then use this to demix the signal.

In addition to these main techniques for sound separation, another more special case technique, and the one this paper will be focusing on, exists for the separation of instruments in a piece of music called Azimuth Descriminiation and Resynthesis (ADRes) [2]. This technique exploits the interaural intensity difference in the left and right channels of an audio recording for each instrument. By using phase cancellation in the frequency domain as well as magnitude estimation, the algorithm is able to recover each instrument (or group of instruments) in a piece of music. The algorithm works well in practice though it tends to leave artifacts in recovered instruments.

A closely related related approach to ADRes is the DUET (Degenerate Un-mixing and Estimation Technique) algorithm [14] – it is an ICA approach which assumes the sensors recording the signals are at different positions in space. This means the sensors will record the sources with different time-delays due to the distance from the sensor and the source. It also means the recordings will differ in amplitude. If the sources being separated are W-disjoint orthogonal, they can be separated by analysing the ratio of the amplitudes of bins in the frequency domain. The W-disjoint requirement means the sources need to be disjoint in the frequency domain or that two sources cannot overlap in any one frequency bin. Unfortunately, because of the harmonic overlap of instruments, this approach doesn't work well for commercial recordings.

4 Background

4.1 ADRes Algorithm

While the complete details of the Azimuth Descrimination and Resynthesis process is beyond the scope of this paper, we will describe the most importants parts. In this way, we may build off the results of the Azimuth Descrimination and Resynthesis algorithm and extend it to produce better results.

We begin by describing a typical recording of a piece of music. In today's mixing studios, it is common to combine instruments into the left and right channels to create a stereo recoding [16]. The contents of the left and right audio channels differ slightly in that an instrument mixed into one channel of audio will have a slightly different volume as compared to the other audio channel. The human brain interprets this interaural intensity difference (IID) [19] as giving a distinct location in 3d space for the instrument being mixed. Thus, the mixing engineer will give an instruments different volume settings, or panning value [16], for each audio channel to separate the instruments into sound pockets. Furthermore, multiple instruments will be mixed into each channel of audio. The combination of these two facts gives us a way of describing the mixing model for a channel of audio. The model can be defined as:

$$L(t) = \sum_{j=0}^{J} I_j(t) Pl_j \tag{1a}$$

$$R(t) = \sum_{j=0}^{J} I_j(t) Pr_j \tag{1b}$$

where $L(t)$ and $R(t)$ correspond to the left and right audio channels in time respectively, $I_j(t)$ describes one of the J independent instruments being mixed, and Pl_j and Pr_j define the amount the mixing engineer wanted to scale the volume (or pan) the instrument in the left and right channels. We can also express the ratio of the left panning value of the j^{th} instrument to the right panning value of the j^{th} instrument as:

$$s(j) = \frac{Pl_j}{Pr_j} \tag{2}$$

From these equations, we can see that the left and right audio channels only differ in the amount of each instrument they receive in the final mix. Furthermore, we can use $s(j)$ to express the ratio of the left audio channel to the right audio channel. In fact, if we scale the right audio channel by $s(j)$, we would expect to be scaling the right channel to the same volume as the left channel for a given instrument j. In fact, if we were to subtract the two audio channels after performing the scaling, we would also expect the instrument j to cancel out. The aforementioned operations can be performed in the amplitude-time domain or in the frequency-time domain. We can look at these signals in the frequency domain by performing a short-time Fourier transform on one frame of the time signal using the following equations:

$$Lf(k) = \sum_{t=0}^{T-1} L(t) e^{-j\frac{2\pi tk}{T}} \tag{3a}$$

$$Rf(k) = \sum_{t=0}^{T-1} R(t) e^{-j\frac{2\pi tk}{T}} \tag{3b}$$

We can illustrate all this visually by supposing we had the two audio channels in Figure 1. Notice how the right audio channel contains the same signal as the left audio channel except it has been scaled by 70%. We can also look at the same two signals in the frequency domain (Figure 2) and find we are still able to see the right audio channel being the same but scaled version of the left audio channel. If we scale the right audio channel by 70% and subtract from the left audio channel, we get the bottom plot in Figure 2 which shows the two frequencies cancelling each other out.

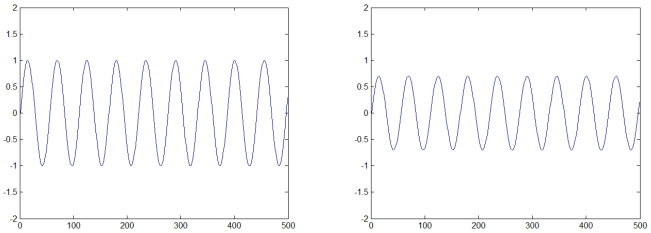

Figure 1: *The image on the left shows the time-domain plot of the left audio channel containing a pure tone at 100% volume. The image on the right shows the time-domain plot of thr right audio channel containing the same pure tone as the image on the left image except the volume has been scaled to 70% of its original amplitude.*

This example is slightly contrived since in a real musical piece, we are not dealing with single pure tones but multiple, independent instruments. Furthermore, we do not know what panning values were used in the original studio mix. Luckily, we do know that if we have the correct panning value, we can perform the scale and substract operation and an instrument in the piece of music will cancel out of the audio channel. By knowing where an instrument cancels out, we can guess the amplitude before it was cancelled out, zero out all other frequencies and then resynthesize to get out instrument.

Thus, the Azimuth Discrimination and Resynthesis algorithm begins by tranforming the input audio channels into the frequency domains. It then proceeds by scaling one of the audio channels by one of a range of values and then it performs the subtraction with the other channel. This can be expressed as:

$$AzL(i,k) = |Rf(k) - s(i,k)Lf(k)| \tag{4a}$$

$$AzR(i,k) = |Lf(k) - s(i,k)Rf(k)| \tag{4b}$$

where,

$$s(i,k) = \frac{i}{\beta} \tag{5}$$

for integer values of i such that $0 \leq i \leq \beta$. Depending on the choice of β, the algorithm can create more azimuth planes though in practice, 10 azimuth planes suffices.

For each frame of the short-time Fourier transform, we can perform this scaling and subtraction operation and then graph our result in a set of planes (called azimuth planes) one for each $s(k)$ value – each plane will show the STFT magnitude plot for the corresponding azimuth value. We then look for places where the graph crosses zero since these are places where we can postulate an instrument has been cancelled out. At each of these zero-crossings, we can estimate the original magnitude of the cancelled out instrument and zero out the same frequency for all other azimuth planes since only one plane for a given frequency will have a zero crossing. This operation constitutes the "Azimuth Discrimination" step in "Azimuth Discrimination and Resynthesis" and the algorithm for it looks like:

Zero-Crossing Detection and Magnitude Estimation

Perform Azimuth Scaling/Subtraction for an audio channel, $Az(i, k)$

for each frequency bin $k_{current}$ **do**
  Find the minimum frequency magnitude $Az(k_{current})_{min}$
  Find the maximum frequency magnitude $Az(k_{current})_{max}$
  Compute the estimated magnitude,
    $Magnitude$ = $Az(k_{current})_{max}$ - $Az(k_{current})_{min}$
  for each azimuth plane $i_{current}$ **do**
    if $Az(i_{current}, k_{current}$ == $Az(k_{current})_{min}$ **then**
      $Az(i_{current}, k_{current})$ = $Magnitude$
    else
      $Az(i_{current}, k_{current})$ = 0
    end if
  end for
end for

After performing this normalization operation, ideally, each instrument should be separated on an azimuth plane. By taking one azimuth plane and running an inverted short-time Fourier transform using either overlap-add or overlap-save allows, we can resynthesize the original instrument. This constitutes the "Resynthesis" step in "Azimuth Discrimination and Resynthesis."

Figures 4-7 illustrate a more complete example, showing the Azimuth Discrimination and Resynthesis process from start to finish.

4.2 Problems with Azimuth Descrimination and Resynthesis

If each instrument was independent in the frequency domain and had a distinct panning value, the Azimuth Discrimination and Resynthesis process would work perfectly. Unfortunately, in the case of instruments in a piece of music, there is a large degree of harmonic overlap which leads to "azimuth smearing." [2] In effect, this means that two instruments contribute to one frequency bin of the STFT and this results in instruments not grouping perfectly on the azimuth planes.

The net effect of this smearing is the resynthesis of not just one instrument but stray frequencies from other instruments as well.

The inclusion of these stray frequencies leads to a poor separation for those instruments which come in and out of the piece of mu-

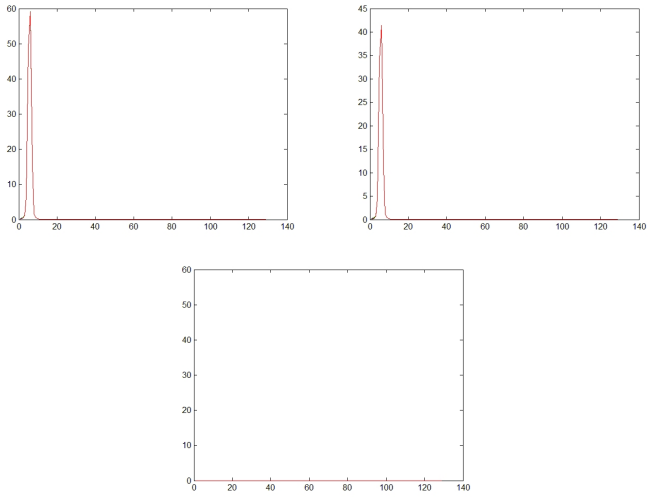

Figure 2: *The top-left image shows the left audio channel from Figure 1 in the frequency domain. The top-right image shows the right audio channel from Figure 1 in the frequency domain. The bottom image shows the left audio channel scaled by 70% and subtracted from the audio channel, resulting in frequency cancellation.*

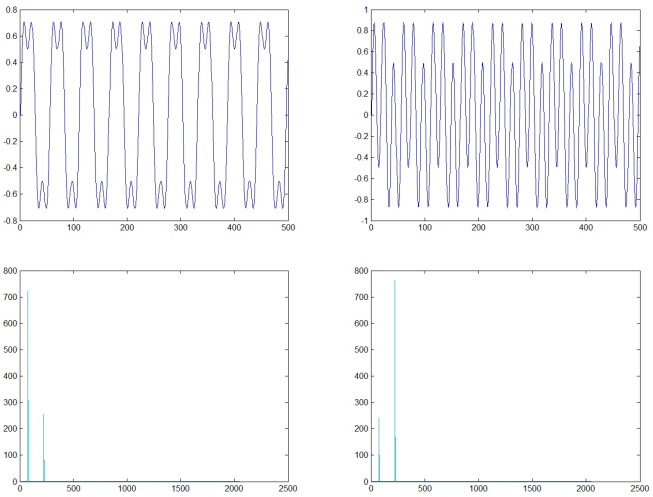

Figure 3: *The top-left and top-right images show the left and right audio channels in a test sound clip. Two pure tones have been mixed into the left and right audio channels. One tone is panned 40% to the left and 90% to the right. The second tone is panned just the opposite. The lower-left and lower-right plots show the magnitude frequency plots for one frame of the STFT of the left the right audio channels, respectively.*

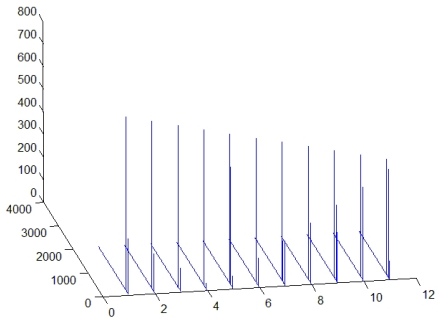

Figure 4: *Azimuth planes, which are the magnitude frequency plots rotated into the YZ plane, for the left channel, before the normalization step. Notice the magnitude frequency in the front start out high at plane 3, go lower in plane 4, and then rise again in plane 5. At plane 4, we have phase cancellation and therefore, an instrument.*

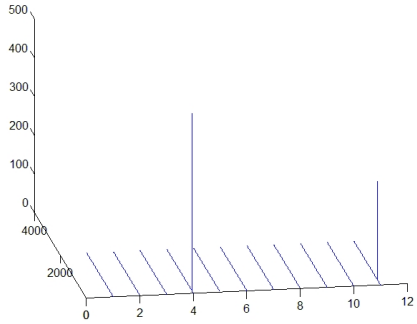

Figure 5: *Azimuth planes after the normalization step. Here, we see the estimated magnitude for one of the tones at plane 4.*

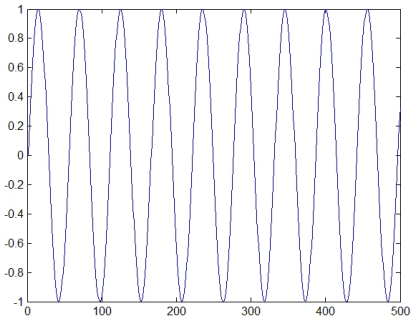

Figure 6: *After using overlap-save, we see one of the pure tones recovered by resynthesizing azimuth plane 4.*

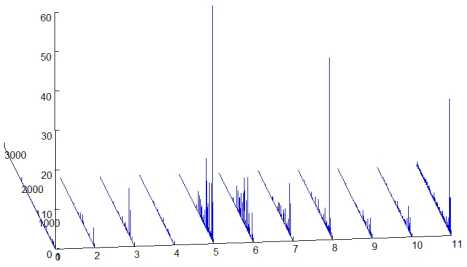

Figure 7: *After performing Azimuth Discrimination, we see an instrument smeared across the azimuth planes instead of being neatly grouped into the planes.*

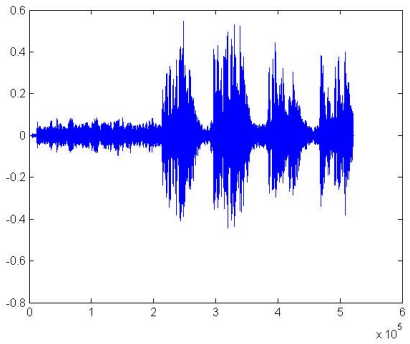

Figure 8: *Separated vocals from a Doors tracks. The vocals begin 15 seconds into the clip, evidenced by the large peaks in the plot. However, because of the "azimuth smearing", we see stray noise in the clip, especially at the beginning.*

sic (see Figure 8). This stray noise is particularly evident when the instrument itself isn't audible or playing. As a result, we now describe a way to selectively resynthesize an instrument in a time-varying fashion so that when an instrument isn't playing, we are able to insert silence and avoid the stray noise. In the end, this will give a higher quality instrument separation for special cases where the separated instrument isn't always playing.

5 METHOD

5.1 Wavelet Decomposition for Instrument Detection

In order to resynthesize a separated instrument in a time-varying fashion, we need to be able to detect the presence or absence of an instrument in a piece of music. Thus, we would like a representation of the instrument that allows us to make this distinction. It is possible to use the time-domain signal itself but we may detect frequencies we are not interested in for the purpose of instrument detection. An example of unwanted frequencies would be transient noise introduced by the resynthesis step of the Azimuth Descrim-

ination and Resynthesis algotihm. Since the time-domain representation makes no distinction between instruments and noise in different frequencies, this turns out to be a suboptimal approach.

A better approach to this problem uses wavlet decomposition which transforms our data into the scale domain (scale being inversely proportional to frequency). Wavelets do not provide as accurate a view of the frequency information of our signal compared to the FFT or STFT but it provides excellent time-resolution. In other words, the wavelet representation of our signal allows us to capture the frequency and more importantly, when the frequencies occur, accurately.

Our instrument signal can be taken into scale domain using the continuous wavelet transform (CWT) [23] with the following equation:

$$\gamma(s,\tau) = \int f(t)\psi^*_{s,\tau}(t)\,dt \tag{6}$$

where our instrument $f(t)$ is transformed into $\gamma(s,\tau)$ which represents our signal at different scales, s , and time-dilations, τ of the mother wavelet $\psi^*_{s,\tau}(t)$. The mother wavelet itself can be any of a number of basis functions that can be scaled and dilated with the following equation:

$$\gamma(s,\tau) = \frac{1}{\sqrt{(2)}}\psi(\frac{t-\tau}{s}) \tag{7}$$

While the theory of wavelets is beyond the scope of this paper, it is clear that the CWT produces a large amount of information because of the mother wavelet being continuously scaled and dilated in time. In practice, looking at discrete scalings and dilations of the mother wavelet produces enough information for our discrete signal. Thus, our discrete signal can be transformed into the scale domain using the discrete wavelet transform (DWT) [23] wavelet decomposition:

$$\gamma(s,\tau) = \sum f(t)\psi^*_{s,\tau}(t)\,dt \tag{8}$$

5.2 Implementation

After transforming our signal into the scale domain, we would like to be able to detect large increases in the amount of energy in the signal as well as large decreases. The first corresponds to an instrument becoming audible while the latter occurs when the instrument becomes inaudible.

In our experiments, a wavelet scale level of 4 tends to be sufficient to filter out unwanted frequencies and detect the instrument's comings and goings. Also, the Daubechies 10 (db_{20}) mother wavelet works well to capture signals with large changes in energy as is the case with our instrument. Figure 9 shows the wavelet decomposition for the voice track shown in Figure 8.

By looking at the bottom layer of the wavelet decomposition in Figure 8, we can see large increases and large decreases of the energy in the signal evidenced by the changes in color. These changes in energy correspond to the instrument (or vocals in this case) becoming audible and inaudible.

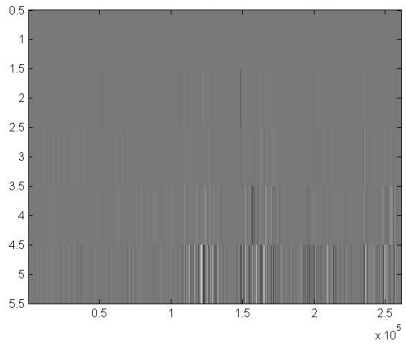

Figure 9: *Wavelet decomposition for the voice track in Figure 8.*

To be able to detect these changes, we implement a straightforward running total algorithm which first breaks the lower scale data into blocks of 1024 samples. We then compute the amount of energy in this block by squaring the samples and summing (squaring the samples allows us to estimate the magnitude of the energy). In addition to the individual blocks of energy, we also keep a history of the energy for the last second. If our sampling rate is 44.1k, we keep 43 blocks in our history assuming our blocks are 1024 samples long. It should also be noted that the choice of 1024 sample blocks was made under the assumption that even if we miss the detection of an instrument's coming or going by 1024 samples, it shouldn't make a big difference because at a sampling rate of 44.1k, 1024 samples represents roughly $1/44^{th}$ of a second. We now step through each block of the wavelet decomposition and compare the current block to the previous 1 second history blocks and see if there has been a large increase or decrease in energy. If so, we mark the area as the instrument coming in or going out respectively. The algorithm is as follows:

Instrument Detection

Use wavelet decomposition on instrument and extract an appropriate scale, $W(t)$
Compute energy on first block of $W(t)$, $E_{current}$
Create history buffer of 43 elements, $E_{history}$ filled with $E_{current}$

for each block in $W(t)$ **do**
  Compute energy on current block of $W(t)$, $E_{current}$
  Compute average energy for history, $E_{avg} = \frac{1}{43}\sum E_{history}$

  if $E_{current} \geq \alpha E_{avg}$ **then**
    Instrument is now audible
  else
    if $E_{current} \leq \beta E_{avg}$ **then**
      Instrument is no longer audible
    end if
  end if

  Add $E_{current}$ to $E_{history}$ by replacing oldest value
end for

Compute Energy of a Block

energy = $\sum_t^{t+1024} W(t)^2$

The α and β variables can be used to tweak how much of an increase or decrease in energy constitutes an instrument coming in and out. After performing the instrument detection phase of the algorithm, we can zero out those samples in the original signal which do not represent the separated instrument. The resulting waveform with stray noise removed can be seen in Figure 10.

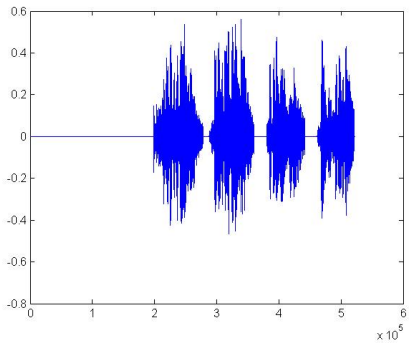

Figure 10: *The altered voice track from Figure 8, after detecting the instrument and removing the stray frequencies.*

6 Results

6.1 Instrument Separation Quality Metric

In order to objectively measure the quality of the instrument separation, we need to define an appropriate metric and apply it to our results. To this end, we use the wavelet once again. By using the wavelet decomposition of our final signal, we can compare our results to the original Azimuth Discrimination and Resynthesis paper. We choose to use wavelet decomposition for a metric because it allows us to look at different scales of our signals (again, in practice, a scale level of 4 tends to work well) thereby comparing our instrument separation to the original instrument track in different frequency (or scale) bands. We have also found that using the Haar basis functions is sufficient for this calculation though other functions like the higher tap Daubechies functions would probably work just as well.

After performing the wavelet decomposition, we treat the output coefficients as a vector in high dimensional Euclidean space. We can then perform a standard L2 norm distance calculation on the vectors to see how close they are and in effect, how similar they are in the scale and time domains. In other words, if we are comparing instrument separations A and B to the original instrument, C, and A is closer to C than B, then we know A provides a better instrument separation. The algorithm for this metric looks like:

Quality of Separation Calculation

Perform wavelet decomposition for original instrument track, $\gamma(s,\tau)_{original}$

Perform wavelet decomposition for the separated instrument track, $\gamma(s,\tau)_{separated}$

Normalize both vectors, $\gamma(s,\tau)_{original}$ and $\gamma(s,\tau)_{separated}$

$$\text{distance} = \sqrt{\sum_{s,\tau}(\gamma(s,\tau)_{original} - \gamma(s,\tau)_{separated})^2}$$

6.2 Instrument Separation Results

We performed three instrument separations for which we not only had the mixed track but also the individual instruments before the mix. In this way, we could compare the quality of our separation to the ideal separation we would have liked to achieve. We also computed the distance between the original instrument with random noise added and the original instrument so that we could get a feel for how close our separations come to a noisy version of the ideal separation. The tracks we used for the separation were:

Track 1	”Yeah” by Usher, Performed by Amit Apte
Track 2	”Soniya (Girl)” by Amit Apte
Track 3	”No Name” by Amit Apte

The results of the separation are listed in the table below:

	Original w/ Noise[3]	ADRes[1,3]	T-V ADRes[2,3]
Track 1	0.2059	0.7624	0.7241
Track 2	0.2132	0.3485	0.3167
Track 3	0.2028	0.3598	0.3574

[1] ADRes = Azimuth Discrimination and Resynthesis
[2] T-V ADRes = Time-Varying Azimuth Discrimination and Resynthesis
[3] All distances are normalized

In all cases, we see that our Time-Varying Azimuth Discrimination and Resynthesis approach produces better results from a purely mathematical standpoint. In one case, the results are better by a factor of the distance between the original instrument track and the original instrument track with added noise track. This leads us to the believe that indeed, we have successfully removed unwanted noise and stray frequencies from our instrument separation. The improvement in separation can also be expressed perceptually by listening to the example recordings and noting the absence of noise artifacts when the instrument in the music sample isn't playing.

Finally, while we cannot use this metric to compare our instrument separation of the example tracks from [2] (as we do not have access to the original instrument-only tracks), the results have been included anyway for completeness.

6.3 DJ Remix Example

In addition to the sample instrument separation, we also demonstrate the possibility of taking two or more pieces of music, separating out instruments and creating a new remix song. By using the Time-Varying Azimuth Discrimination and Resynthesis approach, we present two examples to demonstrate the repurposing of existing music – an amateur DJ might be follow this exact process when remixing a song.

For the first example, we take the track ”Broken Dreams” by Green Day and remove the vocals (removing instruments can easily be accomplished by the Azimuth Discrimination and Resynthesis algorithm by resynthesizing azimuth planes before the zero crossing/magnitude estimation phase). We then separate the vocals from

"Wonderwall" by Oasis and overlay them on top of the instrumental Green Day track. Because both songs have the same chord progression and are in the same key, the remix works perfectly.

The second example illustrates a more complicated usage of the instrument separation. In this remix, we take the vocals from the "Bonus Track" from Dido's "Life for Rent" album and use the backing drum and bass from Zero 7's "Speed Dial No. 2". Lastly, we add a higher range string orchestration from Morcheeba's "Fear and Love". Because all these tracks were played in major keys, with a bit of key transposition, we were able to mix the separated instrument into one cohesive remix track.

7 Conclusion

In this paper, we have shown an improved method for instrument separation. By taking the output from the original ADRes algorithm, we can use wavelet decomposition to detect the presence or absence of an instrument in time. In this way, we can resynthesize our instrument in a time-varying fashion so that the more noticible audio artifacts left behind by ADRes are less obvious. Additionally, we defined an objective metric which can be used to measure the quality of our separation compared to an ideal separation. Lastly, we have illustrated a possible application of instrument separation, namely a way to repurpose existing music so as to create new remix tracks.

8 Acknowledgements

Chad Jenkins helped develop this research topic and acted as my advisor. Anna Lysyanskaya served as a reader for this paper. Ketaki Gokhale helped to edit this paper during the early drafts. Finally, Amit Apte provided the music used to test the Time-Varying AD/R approach.

9 References

[1] A. J. Bell and T. J. Sejnowski. "An information-maximization approach to blind separation and blind deconvolution", *Neural Computation*, Vol. 7, pp. 1129-115, 1995.

[2] Barry, D., Lawlor, B., and Coyle, E. "Sound Source Separation: Azimuth Discrimination and Resynthesis", *In Proc. Conference on Digital Audio Effects*, Naples, Italy, 2004.

[3] Bregman, A. S., "Auditory Scene Analysis", *MIT Press*, 1990.

[4] Brewster, B. and Broughton, F. "Last Night a DJ Saved My Life: The History of the Disc Jockey", New York, NY, Grove Press, 2000.

[5] Brown, G. J. Computational auditory scene analysis: A representational approach, *Ph.D. thesis* CS-92-22, CS dept., Univ. of Sheffield, 1992.

[6] Brown, G. J. and Wang, D. "Separation of Speech by Computational Auditory Scene Analysis", *Speech Enhancement*, New York, NY, 2005, pp. 371-402.

[7] Cardoso, J. F. "Infomax and maximum likelihood for blind source separation", *Signal Processing Letters, IEEE*, Vol. 4, Issue 4, pp. 112-114, 1997.

[8] Cardoso, J. F., and Souloumiac, A. "Blind Beamforming for Non-Gaussian Signals", *IEEE Proceedings* Vol. 140, No. 6, December 1993.

[9] Cooke, M. P. "Modeling auditory processing and organisation," *Ph.D. thesis*, CS dept., Univ. of Sheffield, 1991.

[10] Ellis, D., "Prediction-driven computational auditory scene analysis", Ph.D. dissertation, MIT, 1996.

[11] Hyvarinen, A. and Oja, E. "A Fast Fixed-Point Algorithm for Independent Component Analysis", *Neural Computation* Vol. 9, pp. 1483-1492, 1997.

[12] Hyvarinen, A., Karhunen, J., Oja, E. "Independent Component Analysis", Wiley, New York, 2001.

[13] Jain, V. and Saul, L. K. "Exploratory Analysis and Visualization of Speech and Music by Locally Linear Embedding", IEEE International Conference on Acoustics, Speech, and Signal Processing, 2004.

[14] Jourjine, A., Rickard, S., and Yilmaz, O. "Blind separation of disjoint orthogonal signals: Demixing N sources from 2 mixtures", in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 5, Istanbul, Turkey, pp. 2985-2988, June 5-9, 2000,.

[15] Klingseisen, J. and Plumbley, M. D. "Towards Musical Instrument Separation Using Multiple-Cause Neural Networks", In *Proceedings ICA 2000*, 2000.

[16] McIan, P. and Wichman, L. "The Musician's Guide to Home Recording: How to Make Great Recordings From Cassette Portastudios to Digital Multitrackers" Simon and Schuster, New York, pp. 236-242, 1988.

[17] Parra, L. C. "An Introduction to Independent Component Analysis and Blind Source Separation", Princeton, NJ, 1999.

[18] Plumbley, M. D., Abdallah, S. A., Bello, J. P., Davies, M. E., Monti, G., and Sandler, M. B., "Automatic Music Transcription and Audio Source Separation", *Cybernetics and Systems: An International Journal*, Vol. 33, pp. 603-627, 2002.

[19] Rayleigh, L., "On Our Perception of Sound Direction", *Phil. Mag.*, Vol. 13, pp. 214-232, 1907.

[20] Rickard, S., Balan, R., Rosca, J. "Real-Time Frequency Based Blind Source Separation", In *Proceedings of International Conference on Independent Component Analysis and Signal Separation (ICA2001)*, San Diego, CA, pp. 651-656, December 2001.

[21] Saund, E. "A multiple cause mixture model for unsupervised learning", *Neural Computation*, 7:51-71, 1995.

[22] Smaragdis, P. and Brown, J. C. "Non-Negative Matrix Factorization for polyphonic music transcription", In *Proceedings 2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Platz, New York, USA, 2003.

[23] Valens, C. "A Really Friendly Guide to Wavelets", *http://perso.wanadoo.fr/polyvalens/clemens/wavelets/wavelets.html*.

[24] Viste, H. and Evangelista, G. "On the use of spatial cues to improve binaural source separation", In *Proceedings of DAFX*, 2003.