Gunther Huebler
Industrial Mathematics
Project Report
12-20-19

Abstract

In this paper, a method for detecting the time at which a ping pong ball noise exists within a given audio file is developed. The method is built around the audio feature detection capabilities of wavelets, and has been implemented with software within Matlab. Alternative methods are also discussed briefly, but we're not implemented or tested in any way. The results show that a ping pong ball noise can easily and accurately be detected when general background noise is present, but has trouble differentiating similarly sounding impulse noises. The method used to do this isn't capable of real time performance, but tweaks to make this possible are recommended. The potential use of these results are of this are also discussed, and involve a number of industrial applications

Introduction

The goal of this project was to develop software that is capable of reading an audio file, and determining at what times in that audio file a sound of a ping pong ball can be heard. This is essentially and audio feature detection problem, and will require some type of signal analysis. The most common implementations of feature detection utilize Fourier transform to determine the frequency components within an audio signal. However, this is not a sufficient method of the detection of a ping pong ball noise.

Fourier transform are excellent at resolving present frequencies in a signal, however, those results have no time component associated with them. This makes it impossible to determine the time at which an audio feature is present in a signal. One workaround to this is a Short-Time Fourier transform (STFT), which analyzes parts of a signal over windows of given frequency and time ranges. The main issue with this is that lower frequency components require more samples to resolve accurately in the frequency domain, while higher frequency components need less samples and can be resolved more accurately in time. This issue is solved by using wavelets, rather than classical Fourier analysis techniques.

Wavelet analysis is similar to STFT analysis in the sense that it analyzes signals over a range of time and frequency, but wavelet analysis resolves different frequency components at different time resolutions. This allows wavelets to be far more capable of detecting non-periodic transients within a signal, which happens to be exactly what a ping pong ball noise is.

The noise a ping pong ball makes lasts only about 3 milliseconds, and has no simple frequency components associated with it. It covers a range of frequencies from about 500Hz to 2500Hz, and on a large time scale appear to be an impulse noise. This is why wavelets are necessary, we need a method capable of detecting short noises and localizing them well within time.

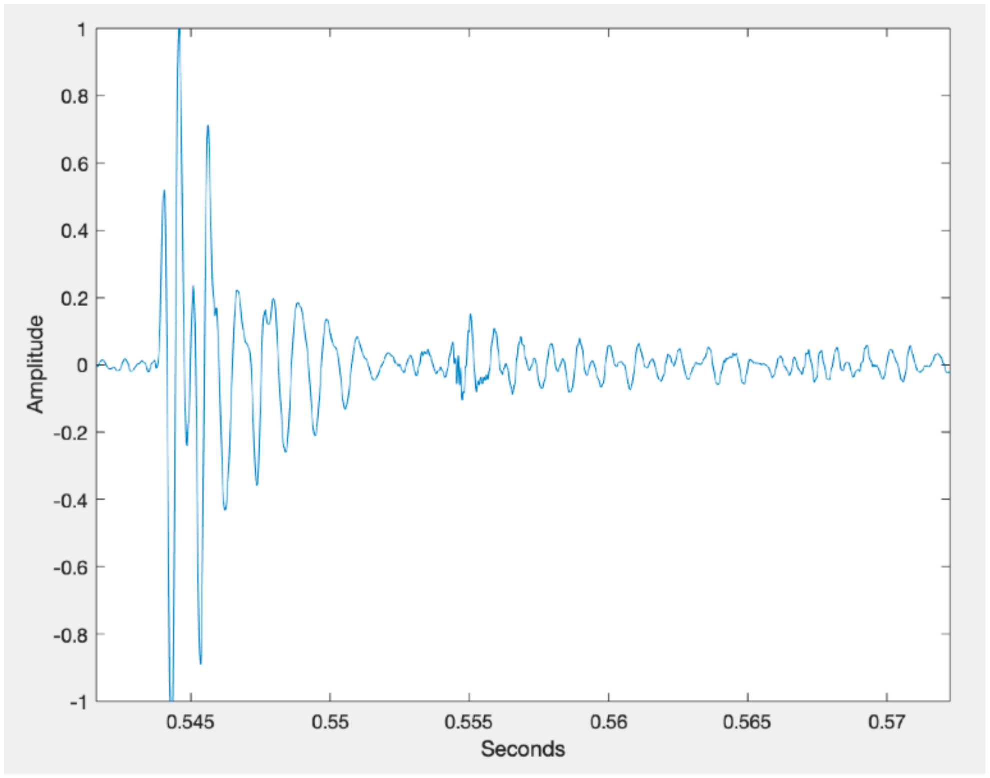

Figure 1. Time space visualization of a ping pong ball noise.

From the time space visualization of the noise, we can see that there are many frequency components present, and over such a short period of time.

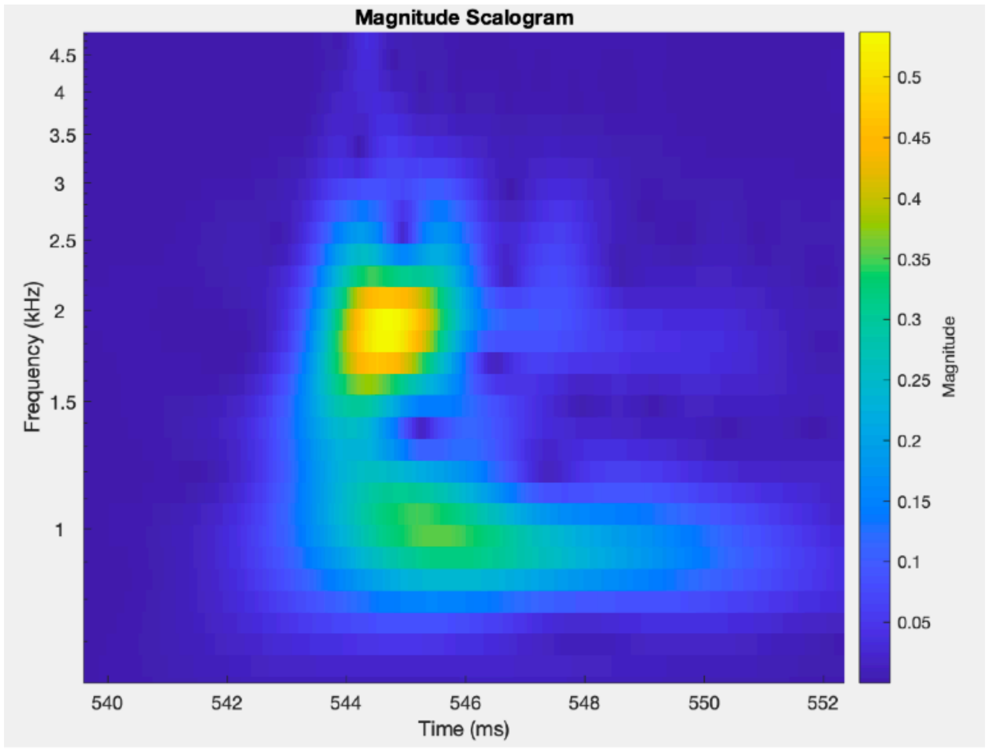

Figure 2. Time and frequency visualization of a ping pong ball noise, using Matlab's cwt function.

Using a time-frequency analysis we're able to discern a feature of the ping pong ball noise that is localized in both time and frequency to some extent. The method of detecting this noice can now be developed since we have a well defined feature that we're looking for.

Method

The method of detecting this noise is based on the time-frequency feature seen in figure 2. The visual shown is colored based on the magnitude of the wavelet coefficients present within the signal for a given analysis method. The feature that we need to detect is the localization of frequency components in the range of 500-2500Hz within a 3ms time window. This can be done by measuring the average magnitude of the wavelet coefficients within two windows.

If we take the wavelet coefficients present in a window of 500-2500Hz that is 3ms long, we are able to narrow in on where we expect the sound to be in time and frequency. Then comparing these coefficients to a larger window surrounding this one, we can determine if the energy of the signal is localized in the smaller window.

This is done by calculating the ratio of average square magnitude of the coefficients in the small window compared to the average square magnitude of the coefficients in the large window. The larger the ratio, the more localized the energy of the signal is within the range of where the ping pong ball spikes.

Determining this energy ratio, centered at a specific time, will tell us how localized the energy is within the ping pong ball window at that specific time. By stepping through the entire audio signal, we can discretely slide the window across all samples of the signal, generating this energy ratio at a subset of the times. Once these energy ratios are determined, their magnitudes can be plotted an analyzed.

Calculating the energy ratio in this way provides decent results initially, but by filtering these ratios further we can improve the results. As the widow slides across samples, first the ping pong ball noise appears in the larger window, then the smaller window, then the outer window again before being passed completely. This results in an unusually small ratio, followed by a large ratio, then a small ratio again, all within succession. To detect this feature, a standard deviation of the energy ratios is calculated, better determining those low-high-low energy ratio areas of the processed signal.

Results

The plots shown will have the energy ratios on top, followed by the standard deviation of the surround 5 sample points of those energy ratios plotted on the bottom.

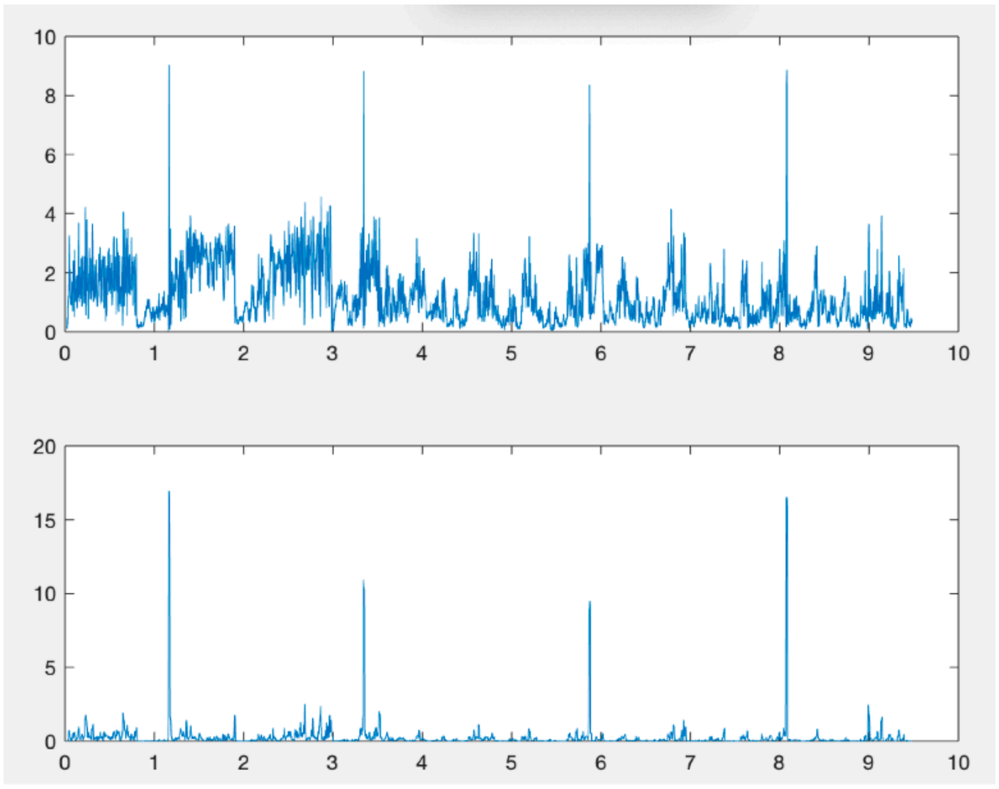

Figure 3. Audio signal containing 4 ping pong ball noises. Accurately and clearly detected.

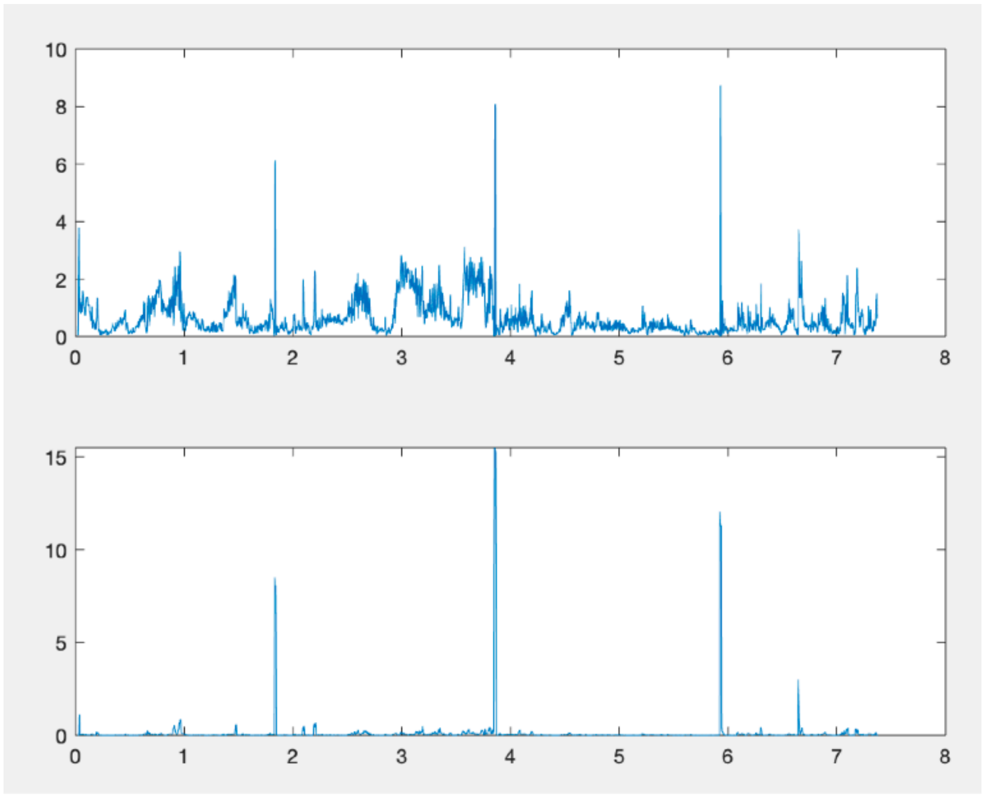

Figure 4. Audio signal containing 3 ping pong ball noises. Accurately and clearly detected.

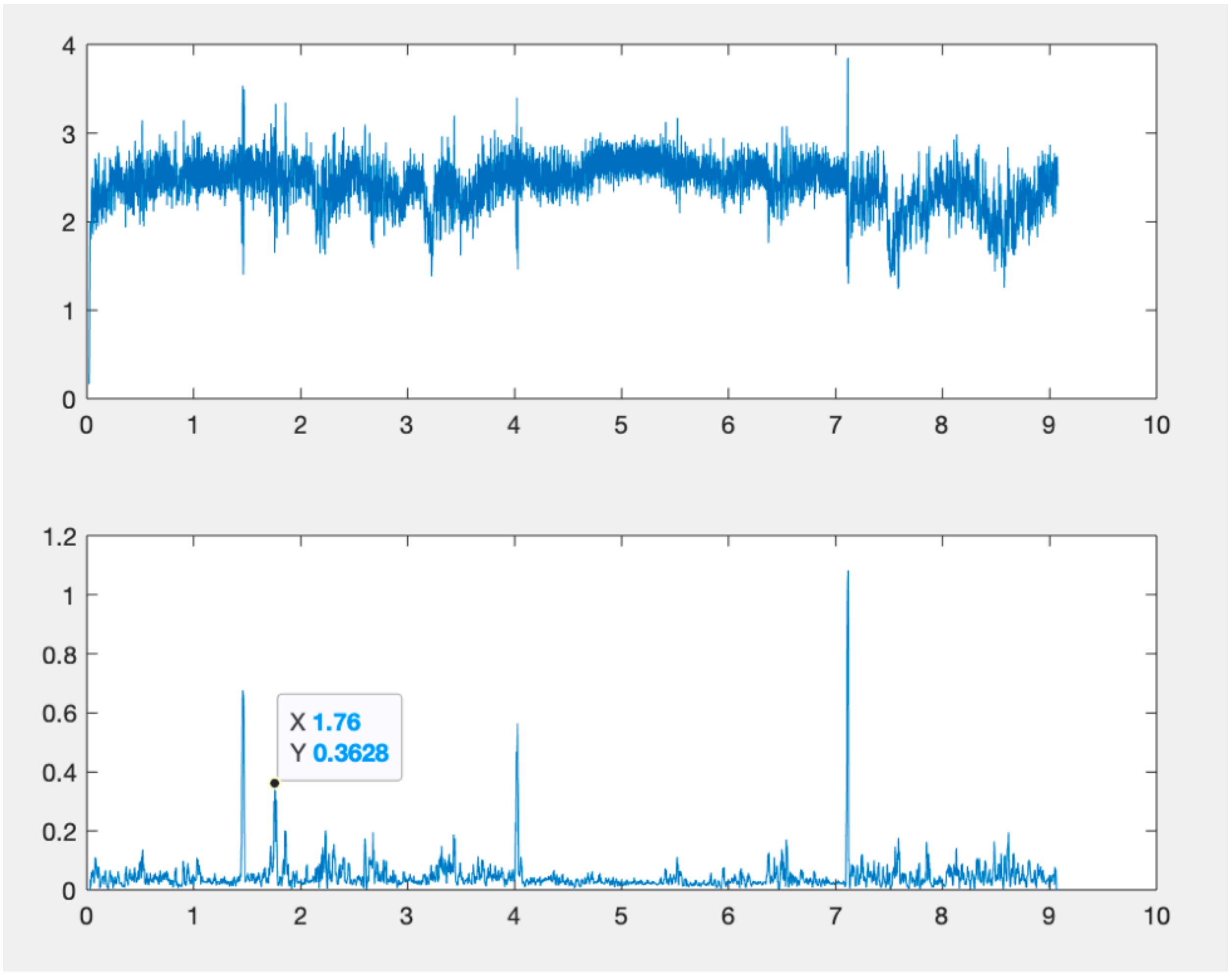

Figure 5. Audio signal containing 3 ping pong ball noises. Accurately and clearly detected, only when filtered.

All the figures shown contain a number of different types of background noise, yet the ping pong noise is easily picked out at the correct times. Figure 5 shows a particularly difficult test of the system, the background noise contained 3 persistent frequencies, all of which lie in the smaller window. The difficulty of this test is observable in the top portion of the figure, where we can see that the average energy ratio is generally high, because those 3 frequency components are within that small window at all times throughout the signal. However, the previously discusses spikes and dips can still be made out slightly, but once the standard deviation filter is applied, the spikes appear only where the ping pong ball noises are present.

There is a significant spike in figure 5 below. It's labeled for visibility. At first this was suspected to be a false positive, but after closer analysis of the audio file with a high quality output device, it was determined that the spike was actually caused by the ping pong ball making a small noise as it was caught after the bounce. So rather than a false positive, it is probably the best example of the robustness of the method.

These results so far demonstrate that the system excels at determine the time at which a ping pong ball noise is made, relative to significant, but dissimilar background noise. The last test determines how well the system can differentiate similar impulse noises from the ping pong ball noise.

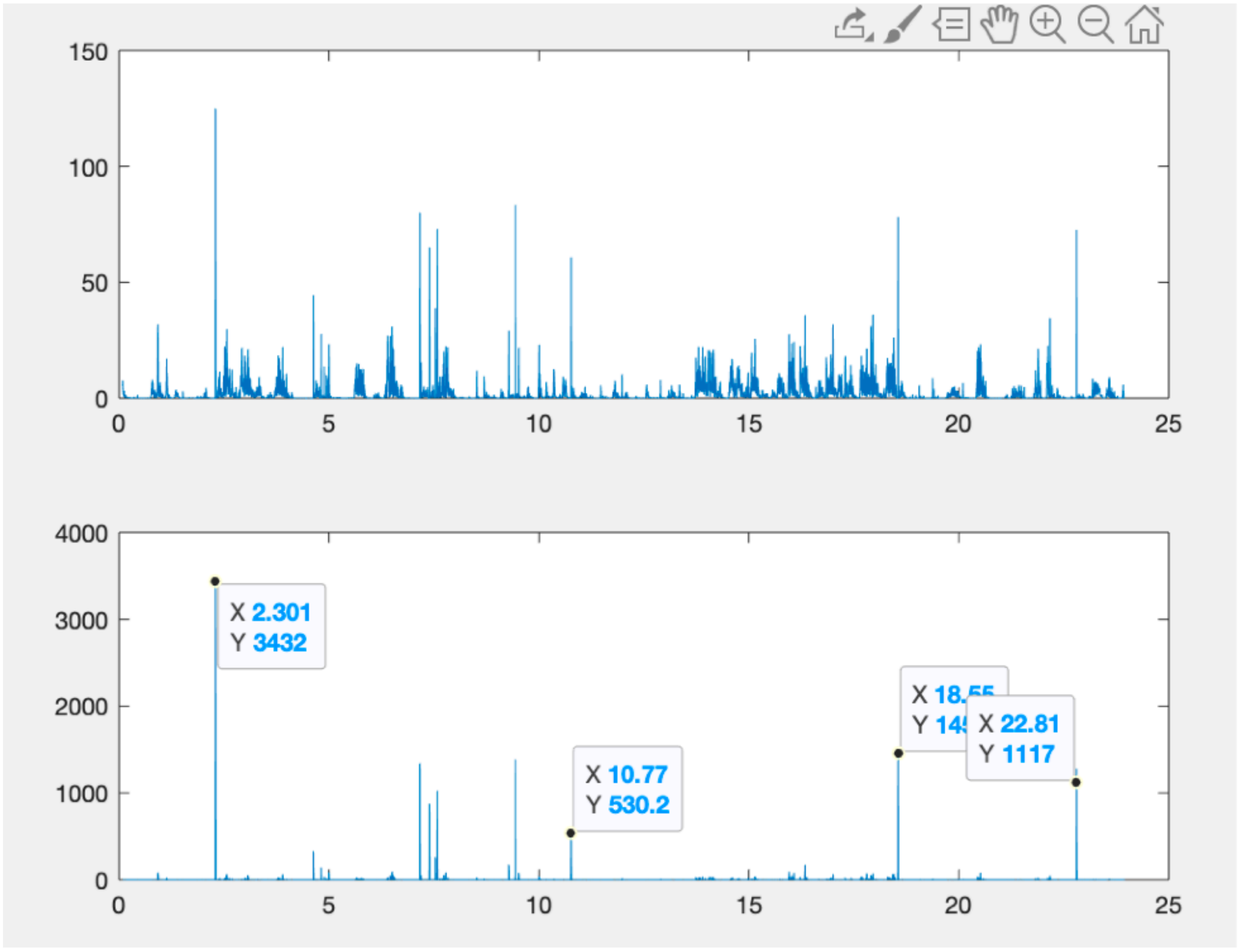

Figure 6. System run on audio file containing similar impulse noises. Ping pong ball noises are labelled.

From this we can see that the method has a number of false positives when similar impulse like noises are present. The false positive are due to a specific pen that was dropped a number of times. Besides this pen, the system was able to ignore a number of taps on glass, dropped metal, and various clicks.

It was found that the audio feature of the pen was similar to that of the ping pong ball which was trying to be detected.

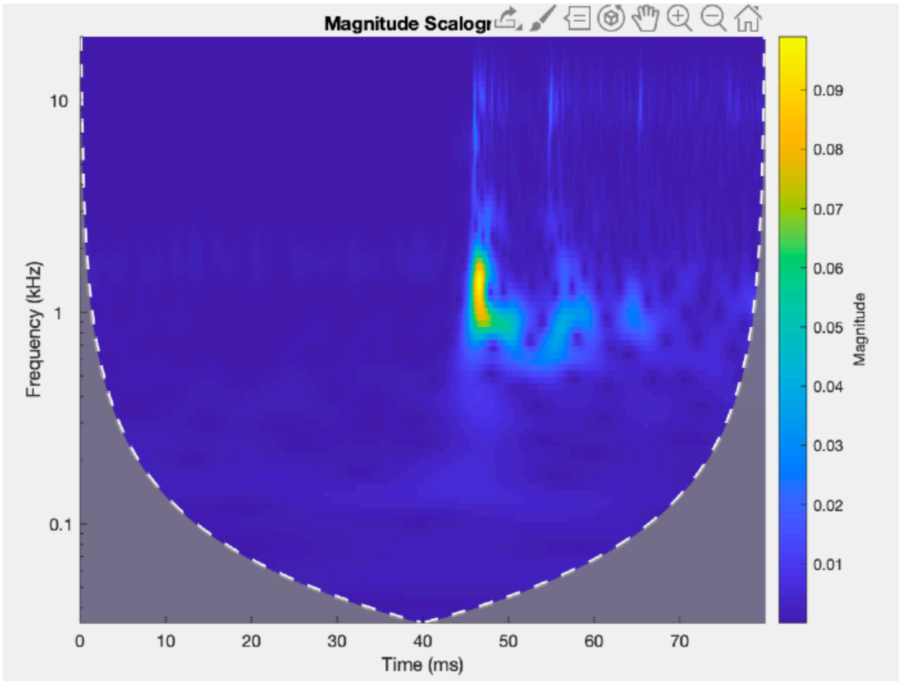

Figure 7. Ping pong ball noise feature.

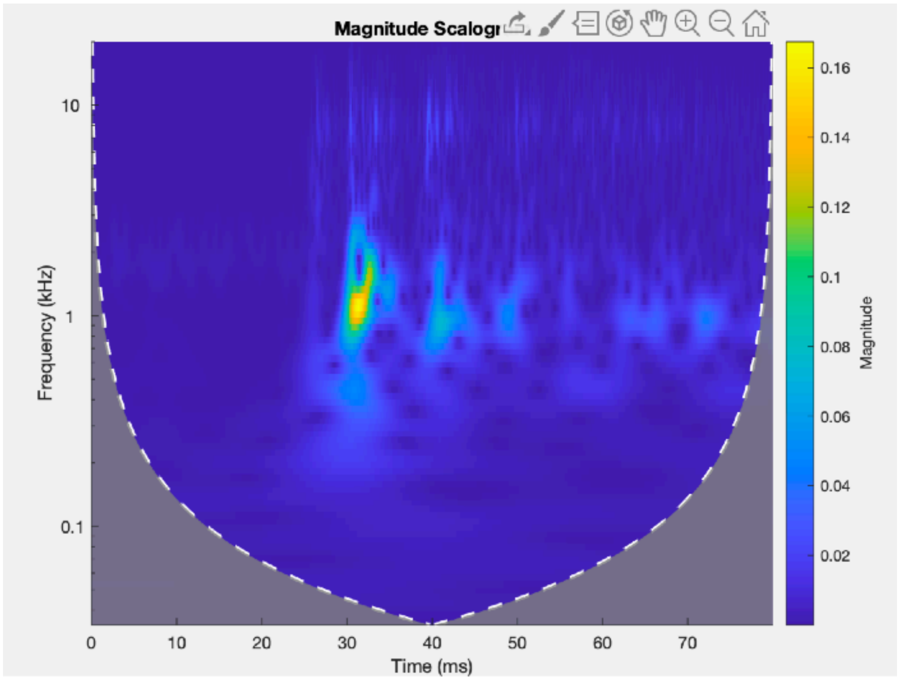

Figure 8. Pen noise producing similar feature.

Conclusions and Discussion

From the results of the tests, it appears the method is very effective for determining the time at which ping pong ball noises exist, with or without dissimilar background noise present, but it is incapable of significantly differentiating the ping pong ball noise to some similar impulse like noises.

One way of improving on this may be to determine and detect additional audio features of the ping pong ball. Listening to the audio files and using the cwt viewer comparing the pen vs the ping pong ball, there seems to be some high frequency components present in the ball but not the pen. These features may be enough to differentiate the two, but the high frequency components are far less well defined than the initial feature we set out to detect, so it is unclear to what extent this additional feature detection could add.

One hope for this project was that this could be converted into a real time system, able to detect a ping pong ball noise as soon as it happen. However, in its current form, the Matlab code analyzes roughly 1 second of an audio file in 4 seconds, far from real time. Additionally, the code is being run in parallel on 4 CPU cores, each analyzing a different window in time, whereas a real time system could only analyze one window at a time. There does seem to be real time potential to this however. Converting the code to a faster language, using a better compiler, and performing general optimizations would no doubt go a long way. It may be the case that this method requires specialized hardware, as does so many other digital signal processing techniques. So currently it is unclear whether this has the potential to be a real time system.

The types of applications this could have vary, but in general all determine some aspect of how the ping pong ball noise is generated. If a more in depth analysis of the noise is done, it may be possible to detect defects in the balls without the need to tediously examine them. It could also be possible to determine features of the surface that the ball bounces off of. Another may even be the determination of the speed at which the ping pong ball is moving once it is hit.

One general concern about this method is that the energy ratio measurement is somewhat arbitrary. There does not seem to be a clear value at which the spike is definitely a ping pong ball (or pen), but rather it seems to depend on the type of background noise present. This could likely be solved by comparing the ratio to the average ratio of the surrounding points, but this was not explored experimentally at all.

These results serves a proof of concept for the effectiveness of this type of feature detection using wavelets, and illustrates a number of specific aspects of the method that could be improved upon.

Matlab Code

```
[sig,Fs] = audioread( 'PingAndNoises.wav' );

dur_short = 0.003;      % In seconds
dur_long = 0.08;
f_short = [ 500 2500 ]; % In Hertz
f_long = [ 0 5000 ];
step = 36;              % In samples

% ---------- Begin Code ---------- %

fb_short = cwtfilterbank( 'SignalLength', dur_short*Fs, 'SamplingFrequency', Fs, ...
                'FrequencyLimits', f_short, 'VoicesPerOctave', 10, ...
                'Wavelet', 'morse');

fb_long = cwtfilterbank( 'SignalLength', dur_long*Fs, 'SamplingFrequency', Fs, ...
                'FrequencyLimits', f_long, 'VoicesPerOctave', 10, ...
                'Wavelet', 'morse');

energy_ratio = zeros( ceil( ( length(sig) - 2 * dur_long * Fs ) / step ), 1 );
t = zeros( length(energy_ratio), 1 );

parfor i = 1 : length(t)
   s_time = dur_long*Fs+(i-1)*step;

   t_start = s_time - round(dur_short*Fs/2);
   t_end = s_time + round(dur_short*Fs/2)-1;
   e_short = mean( abs( cwt( sig(t_start:t_end), 'FilterBank', fb_short).^2 ), [1 2]);


   t_start = s_time - round(dur_long*Fs/2);
   t_end = s_time + round(dur_long*Fs/2)-1;
   e_long = mean( abs( cwt( sig(t_start:t_end), 'FilterBank', fb_long).^2 ), [1 2]);

   energy_ratio(i) = e_short / e_long ;

   t(i) = s_time / Fs;
end

energy_ratio_new = zeros( length(energy_ratio), 1 );
sample_range = 5;
d = floor( sample_range/2 );

for i = sample_range : length( energy_ratio_new )-sample_range
   energy_ratio_new(i) = std( energy_ratio(i-d:i+d) )^2;
end

figure
plot( t, energy_ratio_new )
```