Tempo and Beat Tracking for Interactive Real-Time Performance
By Joseph Newmann

Abstract:
Humans accustomed to standard Western music have become quite good at detecting all rhythmic aspects of music. They are so good at this, in fact, that they will oftentimes associate sounds that are heard outside of a musical context (the rumbling of a train, or footsteps on a sidewalk) as being rhythmic in some way. When it comes to interactive performance, the ability for computer systems to understand rhythm at any level, let alone the level of humans, is incredibly valuable. Similarly to how two musicians may interact based on the rhythms that the other plays, this could create a much more intimate feeling in how musicians interact with computer systems. This ability will enhance interactive real-time performance by adding another layer of interactivity. Past research has been done into different methods of rhythmic analysis based on incoming audio/MIDI signals, with varying results. Often, these methods seek to obtain the rhythm of a player while preemptively providing information that aides in the detection, such as tempo (Ellis, 2008). There are also many instances where temporal and/or rhythmic analysis is used in interactive performance for score following (Artz & Widmer, 2010). In order to increase the tempo options for interactive real-time performance, this paper will outline an attempt to, classify tempo and rhythm based on a monophonic audio signal in real time, with no relevant timing information provided beforehand.

Perception of tempo:
In order to build such a system, it is useful to first understand how humans may perceive tempo. Various research has been done on the topic, revealing a couple of possible means by which tempo is perceived. In 1978, a study on this topic was conducted by H.H. Schulze, a psychological researcher, in order to test three different models on how humans perceive tempo. As summarized by later research, one model suggested that listeners will continually compute and compare the inter-click-interval (ICI), the time between two successive clicks (such as clicks of a metronome), in order determine whether a sequence of clicks is synchronous. Another model, the internal beat model, suggests that listeners will create an internal sequence of beats, with further irregularities in perceived tempo marked by a change in this sequence. The third model, the running average model, states that the listener will, over some number of clicks, calculate an average ICI, which is then compared against any future ICI to determine synchronicity. (Vos, Assen, Franek 1996).

Future studies were done to test these possibilities, with one model – the running average model – coming to the forefront (Keele, Nicoletti, Ivry, & Pokorny, 2004). The running average model is also of low complexity to implement, and it can also be parameterized quite easily, so that different implementations can be tried to test performance. First, the equation for the average ICI can be extracted, as shown below:

$$ICI_{av} = \frac{1}{N}\sum_{i=1}^{N-1} t_i - t_{i-1}$$

The main parameter that can be changed in this case is *N*, the number of successive clicks over which to measure the average ICI. For each value *i*, we accumulate the difference of the time of the current click and the previous one, and then divide by *N* to compute the average. It might be assumed that this ICI can then be directly correlated with tempo, although this is not always the case. Before delving into this

problem, the next question that must be answered is how to extract clicks – or any timing information at all – from an audio signal. A helpful tool to do this is transient analysis.

Transient analysis:

A transient is a "short-lived aspect of a signal, such as the attack and decay of musical tones" (Everest & Pohlmann, 2009). The plucking of a guitar string, for example, could be considered as the attack transient of a guitar note. Although the string may be left to resonate, the actual instant of the pluck is quite short, and signals the moment in time at which the note began. Transients also hold a significant amount of spectral information that is useful for the classification of sounds. Some recent research found that removal of attack transients from a sound significantly impairs the ability for humans to recognize the source of that sound (Siedenburg, 2019). By analysis of the transients present in an audio signal, we can begin to extract some relevant temporal information, through looking at the times at which those transients occur.

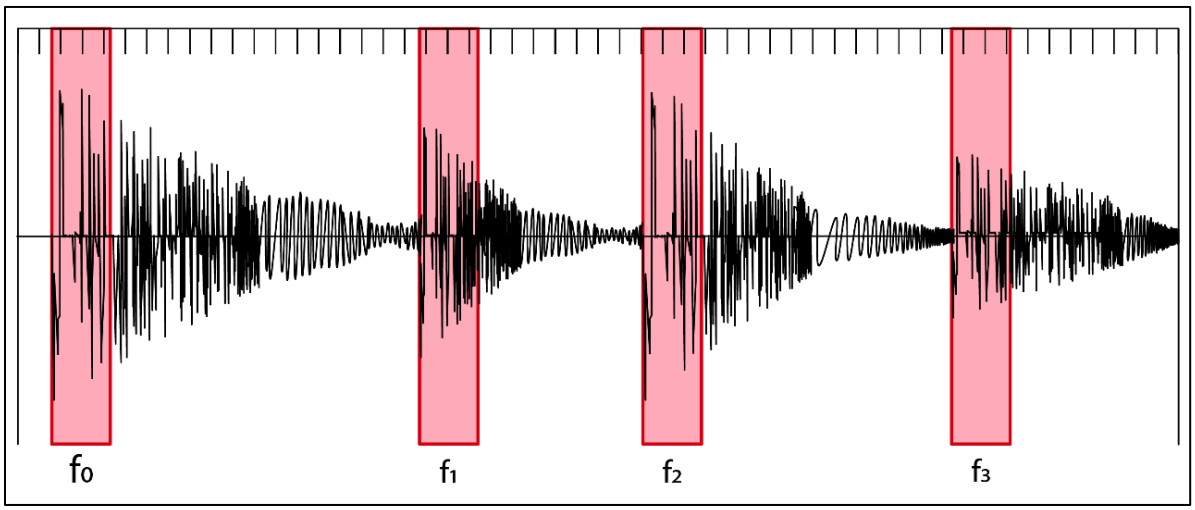

Figure A.

The image above displays an audio signal, with the different attack transient regions highlighted. While there is more signal information (in this case, decay) beyond the transient, for the extraction of tempo, it can be ignored. For the extraction of rhythm, the decay (and sustain) portions of the sound will need to be considered.

In order to extract tempo from an audio signals, there are several different algorithms that can be used. For some purposes, it is necessary to perform spectral analysis on a signal, in order to isolate specific instruments and separate their transients from others. As the goal of this system is to work on a monophonic signal, this is not necessarily required. One method to extract a transient is as follows. We continually sample different "frames" of the signal (short time snippets) and calculate the signal power for each frame. In order to calculate the power of a signal, the root-mean-square (RMS) function can be used (displayed below).

$$x_{rms} = \sqrt{\frac{1}{n}({x_0}^2 + {x_1}^2 + \cdots + {x_n}^2)}$$

In the equation above, *n* denotes the size of our frame, in samples. For each sample in the frame, we calculate the amplitude squared, accumulate their values, and then divide by *n*. The reason for taking the amplitude squared values, and not just the amplitude, is because amplitude values can take the range of -1 to 1. As the direction of amplitude (in the positive or negative direction) is not relevant to the power, we square this value to get rid of the sign. After taking the average of these values, the square root is taken in order to normalize the values.

Once the power for a frame is calculated, it can then be compared to the power of the previous frame. If the difference between these two powers is significant enough, passing above a certain threshold, then a transient has been registered. See below for a visual representation of this process.

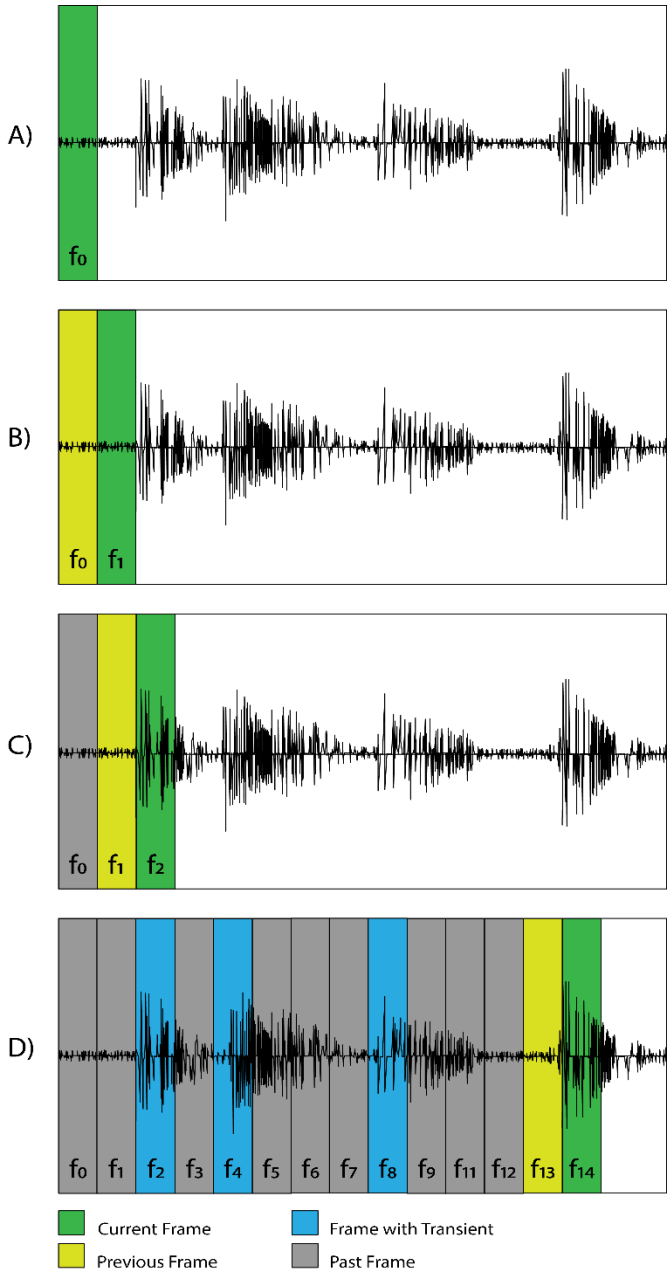

Figure B.

A) The first frame (f_0) has been selected, and the power of this frame is calculated and stored.

B) Frame f_1 has been selected, and the power is calculated. Then, the difference in power of the two frames is calculated. Since this does not exceed the threshold, there is no registering of a transient.

C) Frame f_2 has been selected, and the power is calculated. At this point, the power of the f_0, which was calculated in step A, is discarded and replaced by the power in the f_1. The difference between f_2 and the f_1 is taken, and as it exceeds the specified threshold, a transient is registered.

D) This process continues through the rest of the signal.

This method is of relatively low complexity, but unfortunately, this can lead to some issues in accuracy in some cases. While the first transient in the signal, detected in f_2 can be accurately associated with the start time of that frame, it is not always the case that the frames will line up exactly with the transients. This can be seen with the transient that occurs at frame f_3 and f_4. Thus, there will be an inherent amount of error in the computed time of the transient as compared to the actual time of the transient. In fact, many tempo and/or beat tracking algorithms do have an inherent amount of error, but it is often quite minimal (McKinney, Moelants,

Davies, & Klapuri, 2007). Decreasing the size of the frame would help to some extent, as it would provide a greater time resolution as to when a transient occurs, thus removing error in accuracy. For example, if the size that was used above was halved, the error that occurred between f_8 and f_9 would no longer be present. However, there is a lower bound to the frame size that can be used. If the frame size is small enough, each frame may capture individual peaks and troughs in an audio signal. Another method with which the resolution issue can be fixed would be to overlap each frame by some duration. By doing so, each frame would have more accurate time resolution. Again, care must be taken to determine the amount of overlap between each frame, so that the power measurements are not "oversampled". This could lead to an issue where the difference in power between subsequent frames is not significant enough to pass the threshold, failing to detect a transient when one exists. Once the times of transients in a signal has been detected, we can begin to parse these transient events for an understanding of tempo.

Beat Extraction

After detection of transients, the next step involved is being able to pull some meaning of tempo from them. As the transient detection stage outputs a sequence of times, one possible step would be to calculate the ICI, as described above, and then correlate the resulting value to tempo. For example, if the ICI was calculated to be 450 ms, then we could divide 60 by that value to obtain a tempo of 133.33 bpm. Doing so makes the assumption that each transient corresponds with whatever beat value the tempo relates to. For example, if the musicians' intention is to be playing at a tempo where there are 60 quarter notes per minute, and the ICI is 1 second, then the inferred tempo will be correct. If instead, they are playing eighth notes, but the system assumes that the value of each eighth note is a quarter, then the inferred tempo will be completely inaccurate (double the speed).

For the sake of explanation, let us reconsider the pulses mentioned above to be strikes of a kick drum. Figure C represents the example above, where there is one pulse (strike of a kick drum) every second. Figure D represents a slightly different example, however. In Figure D, the written tempo of the excerpt is 120bpm, twice as fast as that in Figure C. Instead of a kick being played on every beat of each measure, a kick is played on the first and third beat, in this case over two bars. If the same tempo extraction method was applied to both of these cases, the extracted tempo would be 60 bpm.

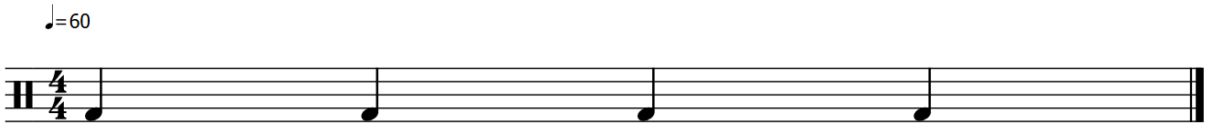

Figure C.

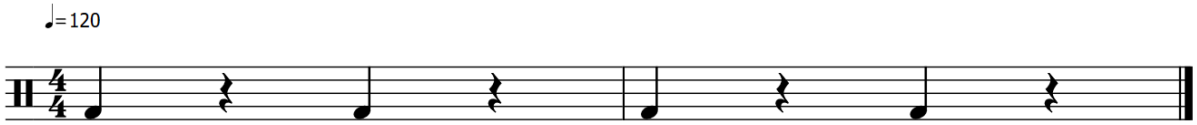

Figure D.

To detect tempo more accurately, it is useful to reconsider how humans might go about detecting tempo from an audio source. There is a concept known as *tactus*, which describes both the intended (controlled by the musician) and absorbed (controlled by the listener) "main beat" of whatever music is being performed (Martens, 2011). Sometimes referred to more causally as "feel", it is what influences the manner in which listeners will nod their head or tap their hands in accordance with what they are hearing. In some cases, it is in sync with the tactus that the musician is performing with, but it is not always the case. Thus, it is a relatively subjective concept, also heavily influenced by the background of the performer and the listener.

A heavy influence of tactus, although not the only one, is the accents that are given to certain beats over others. For example, if the second and fourth beats of Figure C were replaced with hits of the snare drum, this would inform a different tactus than if a snare hit was placed only on the third beat. Over time, there are certain tacti that are quite easily recognizable, such as the ones mentioned above, by strong and weak beats. Since we have the power of each transient that we detected, we can use this as another element in our tempo detection and make more accurate predictions.

To use tactus as an aide to tempo extraction, we must also make an assumption about the tempo that the tactus suggests. For example, if the tactus is such that every other beat is accented, often the tempo is double the ICI of those accented beats. As we are measuring transients based upon their power, we can use not only the time, but the power of the transients, in relation to others, to extract a kind of tactus. To do so, we extract a certain likelihood of one tactus over another, based on the relevant power values. Then, we can apply a weight to different tacti (such as every other beat, every beat, every two beats, etc.), that can be used to help determine the most likely feel. We then can use this in conjunction with the ICI in order to make a more informed estimation of tempo.

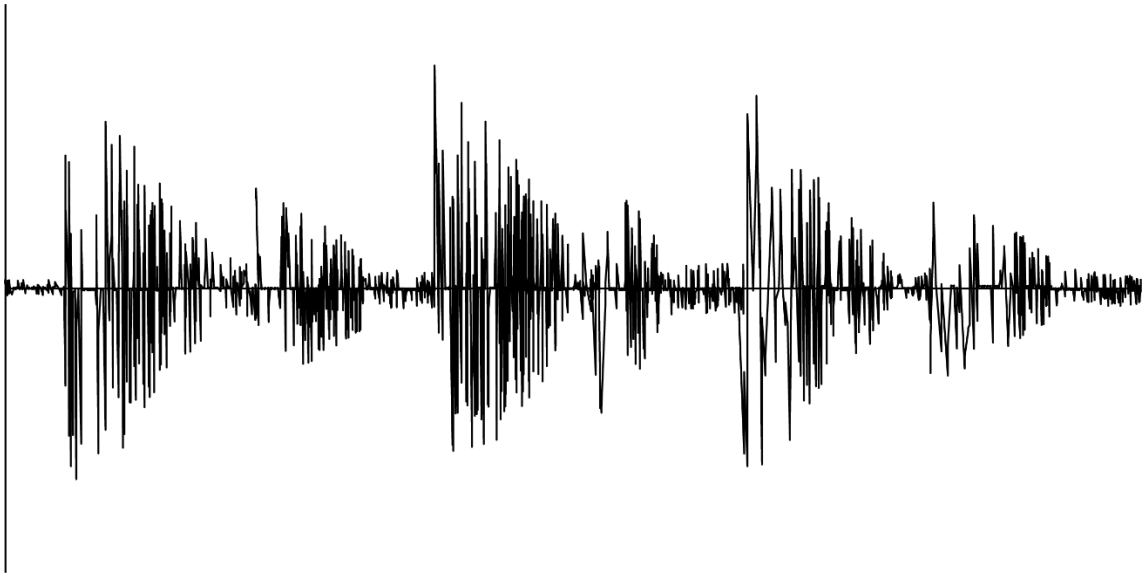

Figure E.

From the audio signal pictured in Figure E, for example, there would be a greater weight associated with a tactus that has a strong beat every other beat than one with a strong beat every beat or every two beats.

For less complex input rhythms, the methods for tempo extraction above will work quite well. However, for inputs that have non-even subdivisions (such as triplets, quintuplets, etc.), the task is more challenging. For example, consider the rhythm in Figure F.

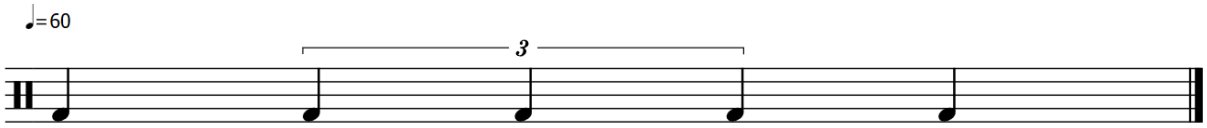

Figure F.

If we begin to keep track of every input signal for tempo detection as we have been doing, we will have the following list of time differences (in seconds):

{1, .67, .67, .67, 1}

In this case, we are not able to apply the same method of strong and weak pulse detection to infer the correct tempo. Since we would have one strong beat at the beginning of the phrase and the next strong beat is at the beginning of the next phrase (a measure later); i.e. there is no strong third pulse present, and therefore, we would most likely anticipate half of the actual tempo. So how can tempo be extracted from this type of phrase?

There are two ways that we can alter the system to account for these types of phrases. The first is to also include weak beats into our calculation in some way. Even though the difference between each strong beat will be four seconds, the difference between each week and strong beat will be one second. Then, during tempo calculation, we interpolate between the two differences, which are a) time between each strong beat and b) time from weak to strong beats. We can then assume that the time between each weak and strong beat will be half of the time between each strong beat. If these two differences are not in sync, then we have not correctly calculated the tempo. We then use the value with more time resolution (in this case, the weak to strong beat), and give it a greater weight in terms of how it affects the tempo calculation. This new method of using weak beats in conjunction with strong beats can be applied to the examples above to provide a greater accuracy when detecting tempo.

The other method involves a rethinking of how this system is going to be used. In our case, the system is going to be used in a real-time performance, instead of building a system that is robust enough to try to do this measure-by-measure tempo detection, we can have the musician do something to inform the tempo, before they begin to delve into more complicated structure. As this system will respond more specifically to different kinds of rhythms, and use those rhythms to trigger some kind of events within the piece, it is required that we have some kind of consistent backdrop against which those rhythms can be understood.

Thus, we can have the performer play some initial phrases that provide a very clear sense of tempo, more similar to Figures C and D. Once a tempo has been established, we can then move forward to rhythm detection. Although this will force the piece to have sections where tempo is established, this can easily

be done in one or two measures. Then, if the player/piece requires a change in tempo, some sort of triggering system can be implemented to begin listening for a new tempo.

Detection of Rhythm

As mentioned above, rhythmic detection must also analyze the decay portions of a signal in order to build an accurate representation of duration. Similarly to how attack transients are detected, the decay transient can be detected by comparing the power difference in frames in reverse order, where the power of the next frame is subtracted from the power of the previous frame. The exact duration of the note can be obtained by subtracting the time of the detected decay transient from the attack transient. In cases where there is failure to detect the decay transient (such as notes that are slurred together), we can use the next attack transient to mark the "decay" of the previous note. An example of the duration detection can be seen in Figure G below.

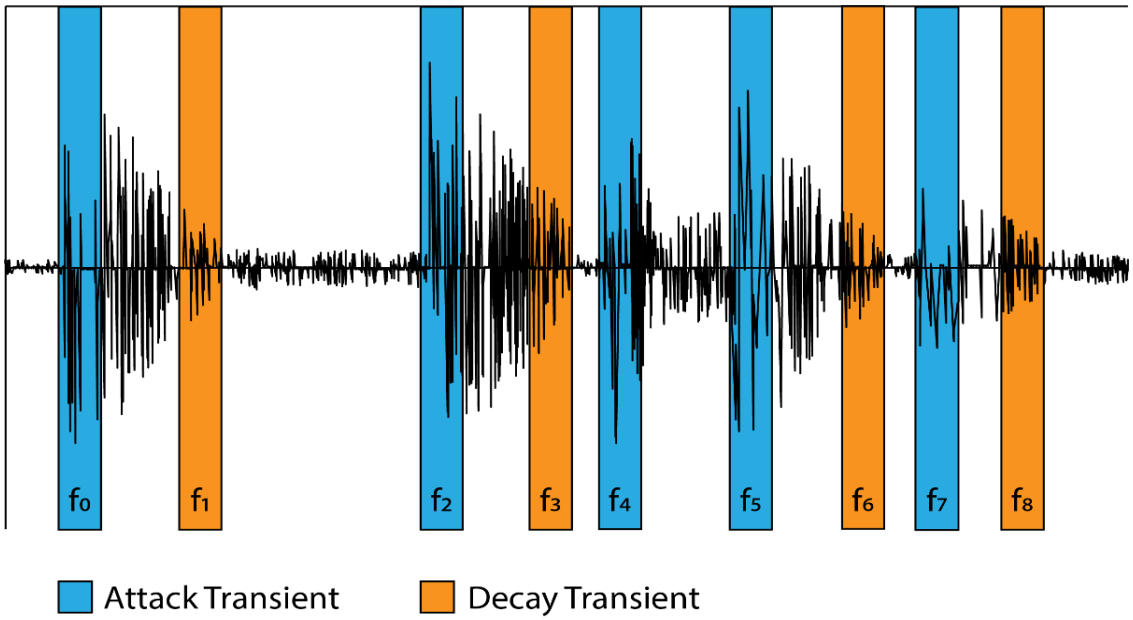

Figure G.

In the example, since there is no decay transient that directly follows the attack at f_4, the attack transient detected at f_5 would be used. As rests have currently been unaccounted for, we can now calculate them as occurring in the space between subsequent decay and attack transients (for example, between f_1 and f_2, and f_3 and f_4). Since we can calculate the amount of time that each beat consumes (from the tempo detection stage) we can make a mapping of beat-time to rhythmic value, as seen below:

Percentage of beat	Rhythmic value
100%	Quarter note
50%	Eight note
200%	Half note
400%	Whole note
67%	Quarter note triplet
25%	Sixteenth note
33%	Eighth note triplet
20%	Quintuplet note

While these are not all of the possibly rhythmic values that could be encountered, they are some of the more standard ones. In order to provide the rhythm, we can pass each value in our set above to this map, which will return a new set. In the case of what we have above, if our tempo is 60 bpm, the following set would result:

{quarter, eighth, eighth, eighth, eighth, quarter}

If the tempo was 120 bpm, we would have the following set:

{eighth, sixteenth, sixteenth, sixteenth, sixteenth, eighth}

We can then pass these sets to our score-triggering system, which will have a mapping of rhythmic phrases to certain actions. An example is outlined below:

Rhythmic content	Action
{quarter, quarter, quarter, quarter}	Reset tempo
{eight, eighth, quarter, eighth, eighth, quarter}	Trig ger scene 1
{quarter, quarter, quarter note triplet, quarter note triplet, quarter note triplet }	Trigger scene 2

In practice, the durations that are detected may not precisely align, so our rhythmic parser will have to somehow “smooth” the timings of the beats that it is provided. In many Digital Audio Workstations on the market today there is a concept of “quantization”, which will “snap” an input to a predefined grid, adjusting for any input error. The image below depicts this quantization.

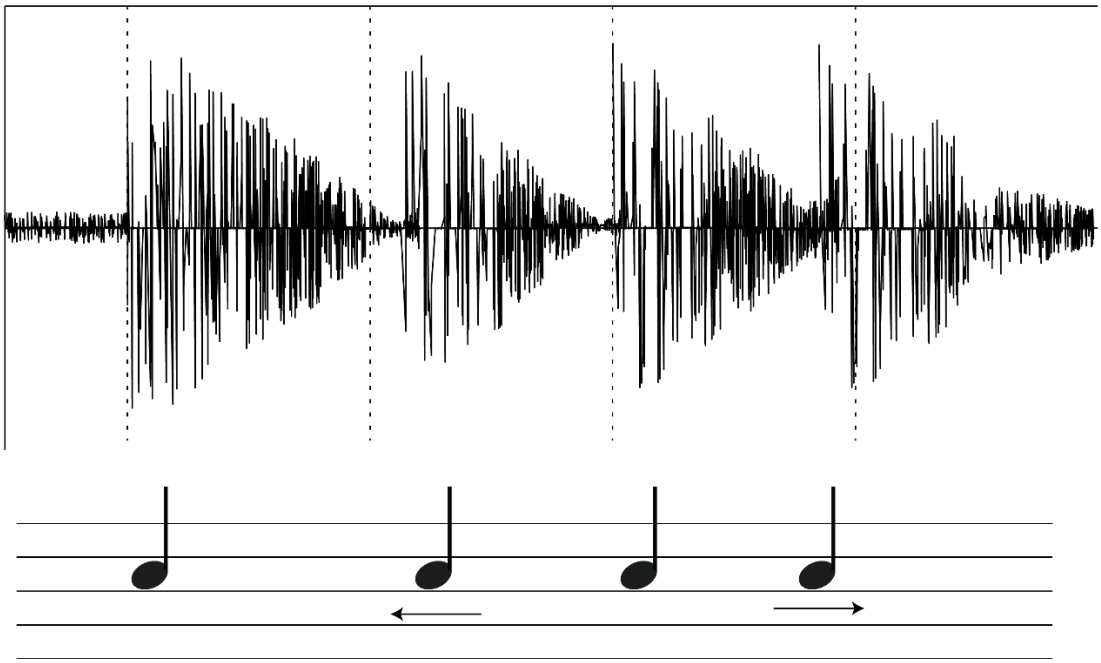

Figure H.

With the dashed lines representing the tempo that has been extracted (from a previous input), in this section, there are impulses that are not exactly on the beat. Thus, the actual timing of the beats will have to be shifted to be lined up correctly, and the score below depicts one possibility for alignment. This system will use a similar strategy, although the snapping will occur once the percentages have been calculated.

Some interesting situations will arise, where there may be ambiguity as to which value we should snap to. For example, if we read a time that was 59%, would we snap it to an eight note or a quarter note triplet? In the end, this resolution will have to be an adjustable parameter in the system, so that we can fine tune it based on experimentation.

Implementation of the system:

An initial implementation of the system will be done in Max/MSP, which is a graphical programming language which can be used to build "patches" (see below) to do audio processing of any kind. At its current state, the Max programming environment has become quite portable, so that many people will be able to use the system without having to worry about having the right platform (Mac or Windows, for example). Ideally, this system would not require Max, but for a prototype Max will be quite useful, as it has many pre-built functions for doing the in depth signal processing that this system will require. Before diving into the implementation, however, it is useful to think about the separate stages involved in rhythmic detection so that the patch can be built in a clean and maintainable way. Specifically, we should think about the inputs and outputs at each step.

1) Beat/transient detection
    a) Input: Mono audio signal
    b) Output: Continuous sequence of times with a power value associated to each time
2) Tempo extraction
    a) Input: Time & power sequence of length *n* from beat/transient detection
    b) Output: BPM based on provided time & power sequence
3) Rhythmic extraction
    a) Input: Time sequence, BPM, and a variable length of transients to use
    b) Output: Sequence of rhythms of a variable length
4) Triggering system
    a) Input: Rhythmic sequence
    b) Output: Value to trigger

So far, the implementation of the system up to tempo extraction has been completed. Due to time constraints, a full implementation of rhythmic extraction and triggering was not completed this semester. Once those two systems are complete, it will be able to be integrated with other Max patches for use in interactive-real-time performance.

A Implementation of beat/transient detection:

This section will take in an audio signal, an adjustable time window and a decibel threshold level. If the incoming signal passes the threshold level within the time window, it will have detected a transient. It will construct a message that is a pair of (current time, signal power), and send that message to an outlet for other parts of the system to use. During testing, the time window and threshold should be adjusted to account for a) the attack speed of the instrument and b) the level of the signal that the musician generates.

Before creating a real-time version of the patch, a non-real-time version was created for ease of debugging and understanding any of the more complex pieces of the system. The resulting patch is pictured below, in Figure I. The patch iterates over each sample in a buffer, continually accumulating the power of each sample as described in the equations above. Once the frame size has been reached, the RMS will be calculated and compared to the value of the previous frame. Due to how events are triggered in Max, there were some issues that arose due to values not being set correctly after the calculation of each frame.

Figure J depicts the real-time transient detection patch that was built, and it was able to reuse some parts of the non-real-time version quite easily. Interestingly, the real-time transient detector patch was more straightforward to implement. This is partly due to the fact that the major components were completed in the non-real-time patch, but also because the real-time patch does not require the iteration over a buffer object, which is how recorded sounds are stored in Max. The real-time feedback made the debugging of issues much easier.

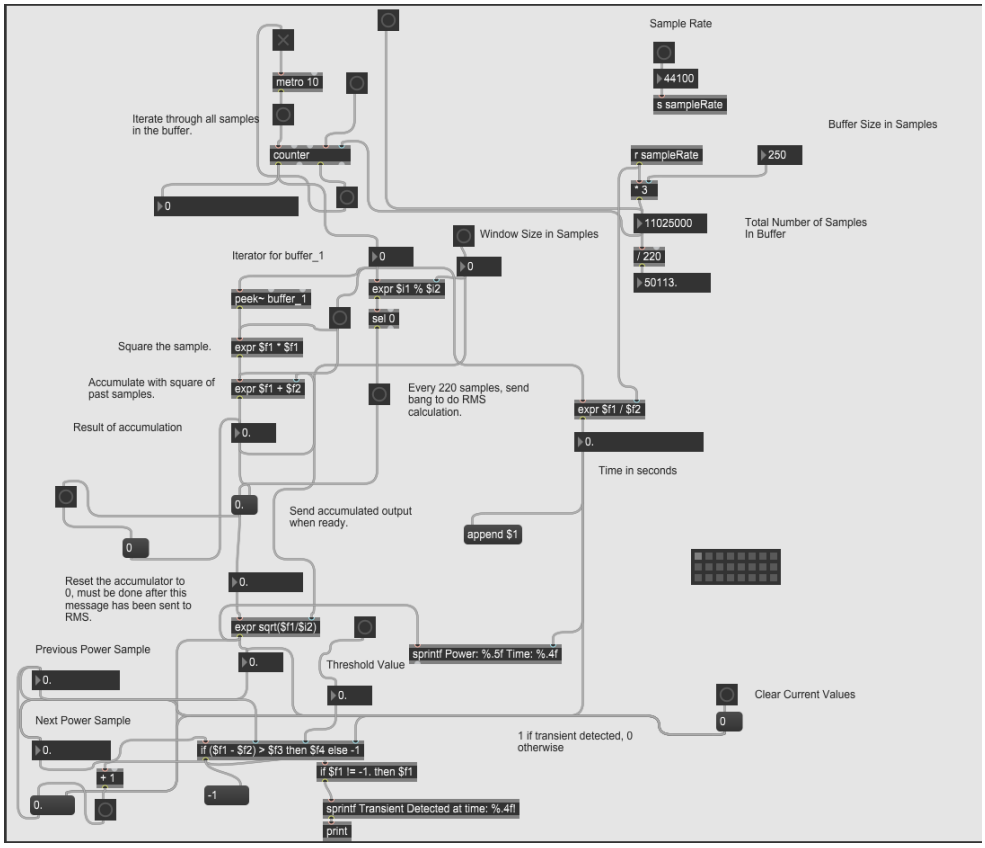

Figure I

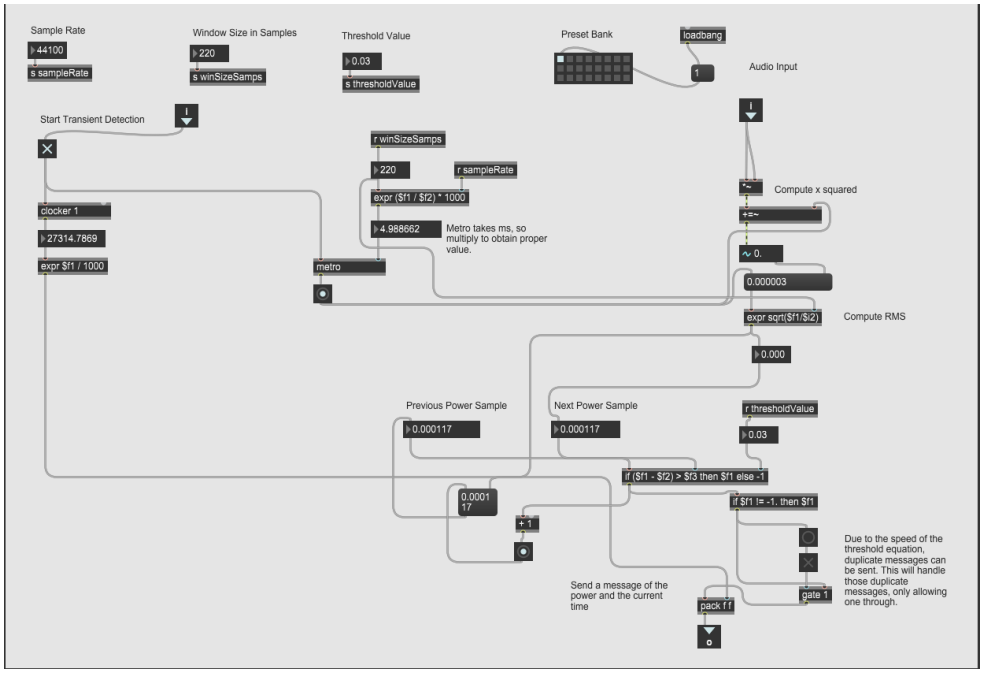

Figure J

B Implementation of tempo extraction:

This section will take in time/power events from the transient detection patch, and add those values to a list. Once the list has grown to a certain size (which will be an adjustable parameter, determining how many beats should be considered when calculating tempo), it will perform tempo calculation. To do this tempo calculation, it will compute the ICI, and also give weights to different tactus, as mentioned above. Interpolating between these two values, it will output an estimated tempo in BPM. This will be based on how constant the time/power events are, as well as the difference between the power (dynamic level) for each event. Figure K depicts the current state of the tempo detection patch.

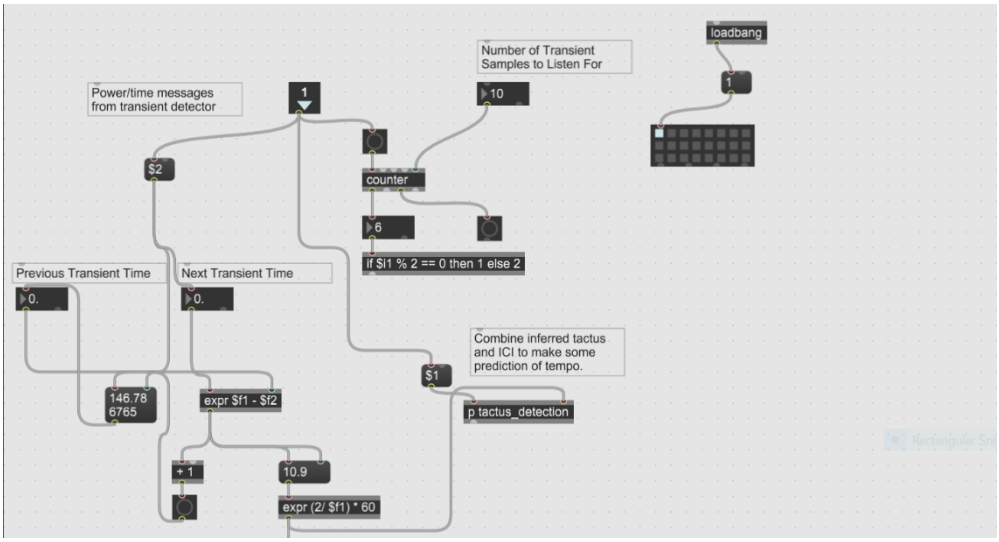

Figure K

C Implementation of rhythmic extraction:

This section will take in the tempo extracted by section B, and the time/power sequence from section A (although it will not take into account the power of each item in the sequence). Although the tempo may/may not change every *n* beats (as determined by the adjustable parameter in the tempo detection section), this section will have to constantly be listening. This section will also have an adjustable parameter of how many notes it should look for before sending it off to the triggering section. For each item in the time/power sequence, it will calculate the ratio of that time to the beats-per-second value as determined from the bpm. This ratio will then be sent into a map of percentage to rhythmic value, obtaining a rhythmic value and adding it to a new list. Once that list has obtained the required number of elements, it will be sent off.

D Implementation of rhythmic triggering:

This section will take in a list of rhythmic values, calculated by section C, and will send each list into a map of value-list to trigger events. If there are no matches, there should be no events that are triggered. If a match does occur, the trigger signal will be sent off into the external patch to trigger the corresponding scene/action.

Bibliography:

Ellis, Daniel P. W. “Beat Tracking by Dynamic Programming.” *Journal of New Music Research* 36, no. 1 (March 1, 2007): 51–60. https://doi.org/10.1080/09298210701653344.

Quinn, Sandra, and Roger Watt. “The Perception of Tempo in Music.” *Perception* 35, no. 2 (February 2006): 267–80. https://doi.org/10.1068/p5353.

Vos, Piet G., Marcel van Assen, and Marek Franěk. “Perceived Tempo Change Is Dependent on Base Tempo and Direction of Change: Evidence for a Generalized Version of Schulze’s (1978) Internal Beat Model.” *Psychological Research* 59, no. 4 (February 1997): 240–47. https://doi.org/10.1007/BF00439301.

Everest, F. A., & Pohlmann, K. C. (2009). *Master handbook of acoustics* (5th ed). McGraw-Hill.

Allen, Paul & Dannenberg, Roger. (1998). Tracking Musical Beats in Real Time.

Arzt, A., & Widmer, G. (2010). *Simple Tempo Models for Real-Time Music Tracking*. http://www.cp.jku.at/research/papers/Arzt_Widmer_SMC_2010.pdf

Glover, J., Lazzarini, V., & Timoney, J. (2011). Real-time detection of musical onsets with linear prediction and sinusoidal modeling. *EURASIP Journal on Advances in Signal Processing*, *2011*(1), 68. https://doi.org/10.1186/1687-6180-2011-68

Keele, S., Nicoletti, R., Ivry, R., & Pokorny, R. (2004). Mechanisms of perceptual timing: Beat-based or interval-based judgements?. Psychological Research, 50(4), 251-256.

Siedenburg, K. (2019). Specifying the perceptual relevance of onset transients for musical instrument identification. *The Journal of the Acoustical Society of America*, *145*(2), 1078–1087. https://doi.org/10.1121/1.5091778

McKinney, M. F., Moelants, D., Davies, M. E. P., & Klapuri, A. (2007). Evaluation of audio beat tracking and music tempo extraction algorithms. *Journal of New Music Research*, *36*(1), 1–16. https://doi.org/10.1080/09298210701653252

Martens, P. A. (2011). The ambiguous tactus: Tempo, subdivision benefit, and three listener strategies. *Music Perception: An Interdisciplinary Journal*, *28*(5), 433–448. JSTOR. https://doi.org/10.1525/mp.2011.28.5.433