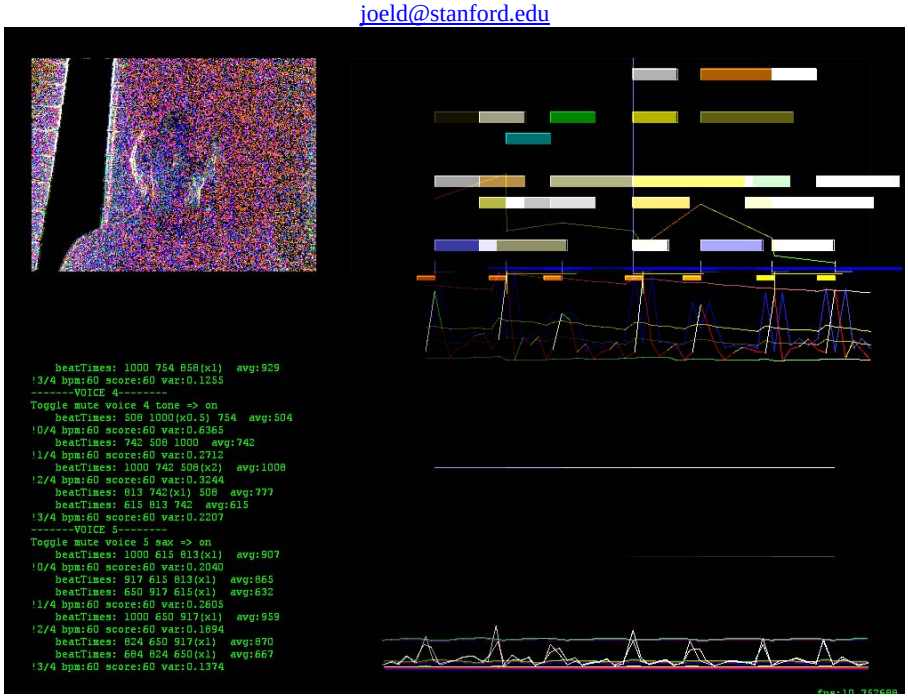
Moflo 09.06.09 – Using Motion to Influence Musical Time

Joel Darnauer
Stanford University
MUS 220C
joeld@stanford.edu



Dear Reader – I'm not yet happy enough with the results of my experiments to submit a technical paper. As a result the tone of this paper is more discussion oriented and I cover a lot of territory.

Please take a moment after reading it to email your immediate impressions to me at the address above. Or if you are reading a paper copy please make notes and I will pick it up from you. ***I'm especially interested in hearing from you about what you thought was interesting or has potential for continued work.***

Also, if you have time I'd like to get your personal feedback over lunch either at Stanford or at Google. Please let me know what time works for you.

Thanks,

joel

650 714 7688

ABSTRACT

We present a method for using feedback from a video camera to optimize computer generated music in real time. The goal of the system is to survey the space of musical possibilities using the visible and unconscious evidence of peoples movements to define the limits of goodness. Our system is built on top of Chuck and Processing and uses the simple frame difference as the metric of choice. Despite technical limitations in frame rate we are able to extract some motion like signals, and to explore the tradeoff between instrument and measuring device.

MOFLO OPTIMIZES MUSIC BY MEASURING MOTION

Advances in signal processing and video make several things possible were not possible before.

Specifically computer vision provides us with an opportunity to extract rhythmic motions from scenes of

musicians performing or dancers dancing. The information extracted from these scenes could be used to objectively measure audience participation, and then to modify or augment a musical performance or experience.

INTRODUCTION

I'm interested in how and why music gets people's attention. Performing musicians and DJs have an intuitive sense of whether they are connecting with an audience, perhaps based on crowd motion or where people are looking, or their facial expressions. They often adjust aspects of their performance to maximize participation. This forms a feedback control system.

The music generation function in this system can now be easily replaced by a computer. If we also had an automated way to measure the audience's enjoyment of the music, we can imagine using it to systematically study what features in music are interesting to a population, or even one day to develop an automated system for maximizing crowd participation. (Figure 1).

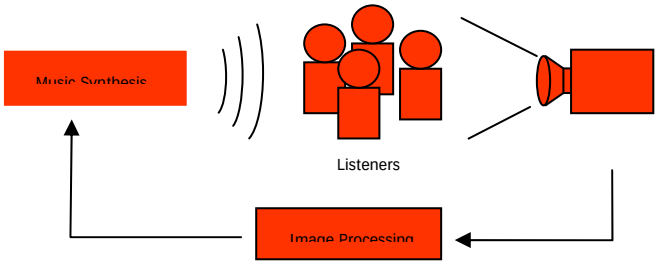


Figure 1: Concept for an automated audience measurement in a feedback control system.

The purpose of this project was to build the computer-vision part of this system with two types of applications in mind:

(1) "Activity Measurement" - tempo-based motion analysis with the goal of detecting how much motion in a scene is related to or caused by a musical beat or event. This could be used for grading samples of music in a dance hall where the music is relatively fixed.

(2) "Motion-based Instrument" – an event-based motion analysis that controls actual sounds.

Notice right away that there is a tension between these two objectives as the "instrument" functions want fast reaction times but the tempo-monitoring functions want stability over some window. We hold out the hope that there is some interesting place between these two modes of operation where the boundary between performer and spectator becomes blurred. We will return to that discussion later.

RELATED WORK

There is ample prior work on motion capture of human motions in character animation, and some of this even employs knowledge about the repetitive nature of dance-like motions[7]. Gabayan developed a camera to trigger musical sounds based on motion in a camera interface [1], but did not explicitly test for the rhythmic nature of the input video. Several papers are present on trying to use computer vision to automatically rate the amount of emotion a dancer is expressing [2,3], but these systems appear to primarily focus on things like the dancers speed, posture, or change in posture.

Several systems exist for tracking the repetitive motions of a conductor as part of a performance either for locking the playback of a musical score [4] or for controlling interactive displays [8]. A number of papers have also used computer vision to create virtual drum sets [10] or to augment existing instruments [11]. Simple games exist that use input video to trigger dance like games [13]. There is even interesting work in correlating audio and video streams [11] and in finding camera-independent periodic motions in video streams [12].

Finally, there are many good algorithms for extracting beat from audio [14][15][16]. As we will discuss later, these problems are related, but different in important ways.

THIS IS THE THIRD MOFLO PROTOTYPE

Before continuing let me mention three two early protoypes of the system and lessons learned from them in previous quarters.

The very first prototype was build from OpenCV in Chris Chafe's class. (Figure 4). It had very limited sound capabilities with PD (only theremin type sounds) while OpenCV is fast it does not have a very large or rich collections of 3D graphics routines like PD.

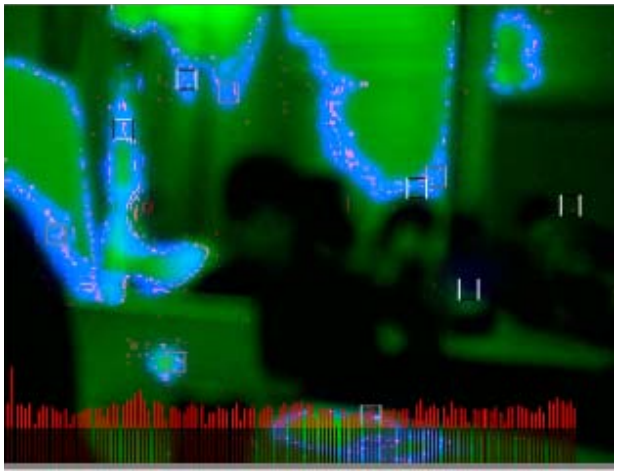


Figure 2: First instrument prototype using OpenCV and PD. Higher frame rates (20fps) were possible with OpenCV, but Processing has a richer set of features.

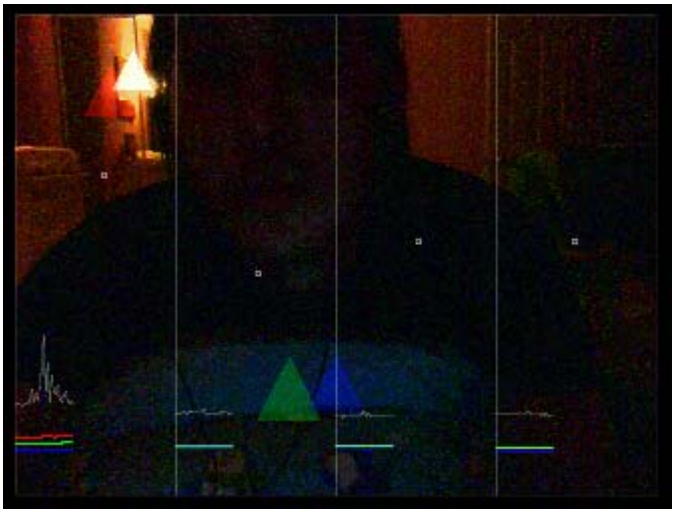


Figure 3: Second prototype video instrument. The screen is split into four vertical rectangles, each of whch sends motion and rgb messages over OSC with each frame (line graph in lower left). In addition brigness and movement moments in the rectangles (grey dots) can manipulate tone. Colored triangles use a mean-shift algorithm to track color blobs. The instrument has a plethora of degrees of control, but is difficult to play music with for this reason.

I build a second prototype in CS 220B which focused on expressive control of a bank of up to eight instruments, each with knobs for pitch, volume, and tone. The number of modes of this and the difficulty of control quickly revealed a problem with using video as an instrument. The potential with video is that it is natural and requires almost no learning. If we break that by making players do complicated things then we are misusing the gift of video.

MOFLO USES PROCESSING, OSC AND CHUCK

The current system design uses Processing to handle video and graphics and Chuck to handle sound. OSC messages are sent back and forth for communication. (Figure 4). This system achieves only 10fps in my current hardware configuration, mostly because of interactions between processing and the webcam driver. There is some CPU usage limitation however.

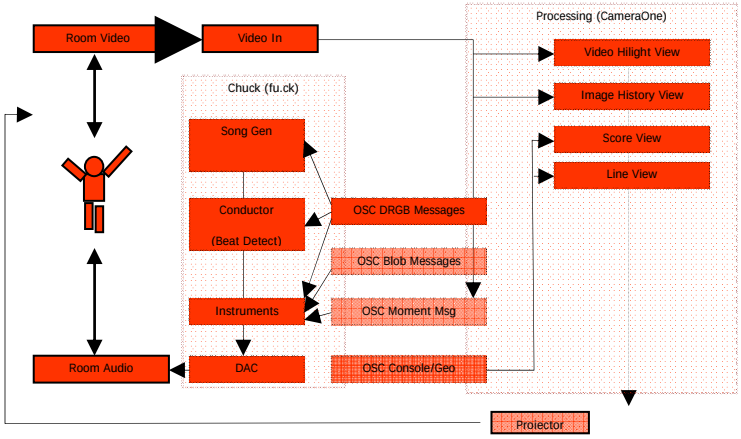


Figure 4: Moflo system architecture. Video input is handled by the CameraOne Processing sketch, which passses OSC messages to Chuck for sound synthesis.

To be effective the camera needs to have about 30fps with a small latency, so improvements are needed in this area.

MOFLO EXPLORES MUSICAL SPACE

A key component of the system needs to be the music that is generated. It needs to be musically engaging, but also experimental and new. Figure 5 shows the relationship between the explored musical space and potential new territory.

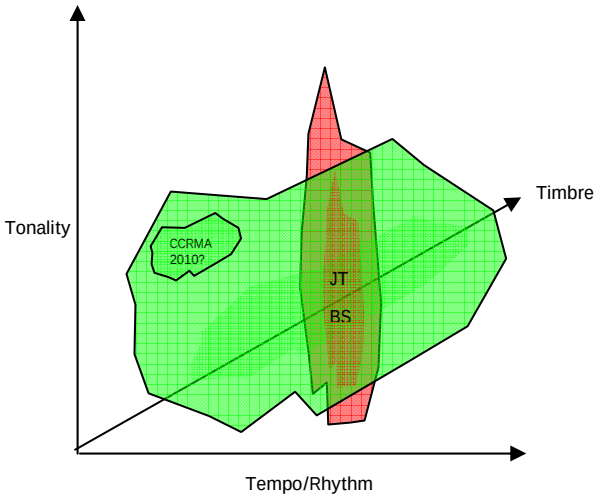


Figure 5. A simpifiied view of musical space represented by three dimensions: rhythm, tonality/pitch and timbre. The red subspace represents areas of musical space which have been explored while the green represents areas that sound "good". A central goal of computer music should be to efficiently explore more of the green area.

Currently Moflo uses a generative grammar approach to synthesize musical details. Patterns are chosen randomly to generate verses from songs, progressions from verses, measures from progressions, and rhymic details of a "feel" from a measure. The system is tilted toward randomness at the moment and the basic data structures lack compositional tools needed to tie together neighboring sections of a song in a smooth way. Moflo can produce some interesting sequencer-like drum samples and bass riffs within a stationary set of pitches.

MOFLO HAS A GRANULAR SYNTHESIS PLUGIN

I experimented briefly with using existing tracks of songs with strong dance beats as source material and using the motion estimation to control the advancement of time in the song. For example, a song might be allowed to play 200msec long clips whenever a certain MIDI channel triggers and then pause. Coupled with an ADSR envelope and variations of pitch stretching and rewind, the source sound can be reinterpreted lightly or very heavily.

This produced some nice results. The especially interesting part is when the controlling tempo is different from the tempo in the song – which leads to an attention-getting but sometimes dissonant confusion about where the time is coming from.

More work is certainly needed here to create appealing sound design and to map the sounds to the controls. Let's shift our attention to the control side for the moment and look at different metrics that can be used to gauge motion.

MOTION SIGNALS ARE GREAT IF WINDOWED

I have a short video of a glockenspiel being played at a steady tempo that can be used for data collection (Figure 6). If you know where to look in the frame, for example at one extreme of the swing, you get very crisp images that should be great for tempo and trigger events. (Figure 7).

Figure 6: Sucessive frames from the glock96 data set.

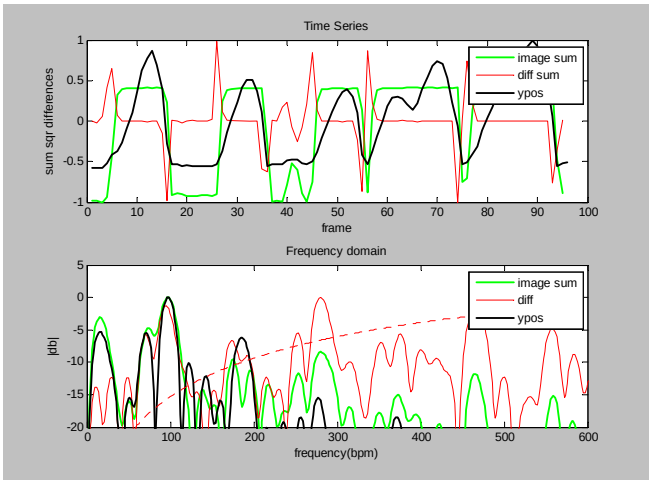


Figure 7: Time series and frequency analysis from a small 10x10 window showing the ground truth motion (black), the image sum in the window(green) and the first-difference of the image sum(red). The ground truth and the window sum show good alignment in the time and frequency domain.

DIFFERENT WINDOWS WON'T AGREE ON PHASE

The only problem with this is that the performer or camera can move so we would constantly need to revise the location of the windows. (Figure 8).

Figure 8: An object travelling in a periodic path through multiple windows will report a different phase in each window.

In addition, the choice and size of window locations is fixed an arbitrary and not likely to be adapted. For Moflo 09 we chose to simply take the pixel by pixel differences and sum them.

In addition to latency there are a number of other confounds extracting phase from video.

THERE IS A TRADEOFF BETWEEN ABS() AND SQR()

There is more than one way to do this, however. One simple way would be to look at the brightness of each pixel and use that as the signal of interest. The main problem with this is the constant-background problem. If we look at the pixel by pixel sum of a white object moving on a black background, the overall brighness of the image doenst change.

Instead we could sum the squared differences of each pixel, or the absolute difference. The squared difference has the advantage of giving a lower weight to small changes (noise). However, it will tend to underweight the motion contribution of blurry objects smeared over many pixels. (Figure 9).

Both of these techniques rectify the signal, which produces a frequency doubling effect. This is not so desireable.

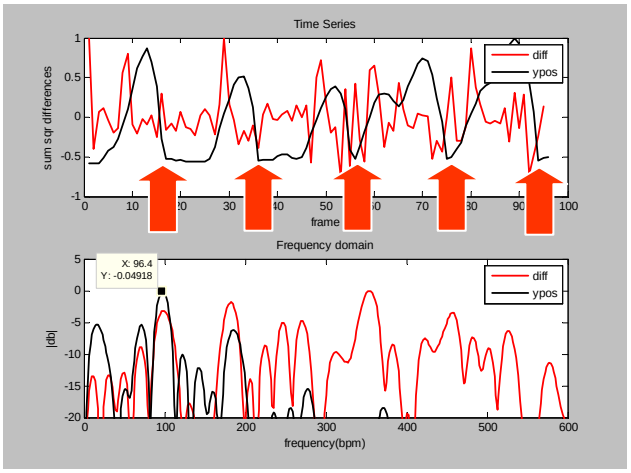


Figure 9: Ground truth of mallet position (black) along with difference signal (red) and beats (orange arrows). Notice that the beat occurs on the smaller motion event. This is probably because the fast mallet motion is blurred out.

PHASE IS AMBIGUOUS IN MUSICAL TIME

Furthermore, one must ask the question, especially for trigger based algorithm, we have to ask how we migh disambiguate such a signal to extract the trigger. Consider the motion of a bouncing person depicted in figure 10. Beats for played instruments like drums might correspond in time to the peak of the force in the image, which is the time when acceleration is highest, not when velocity is highest. For uniform motion, these two quantities will actually be 90-degrees out of phase.

Even without this effect, cultural matters confound where the beat lies. Consider the downbeat styles of music (rock) and upbeat styles (reggae) for examples.

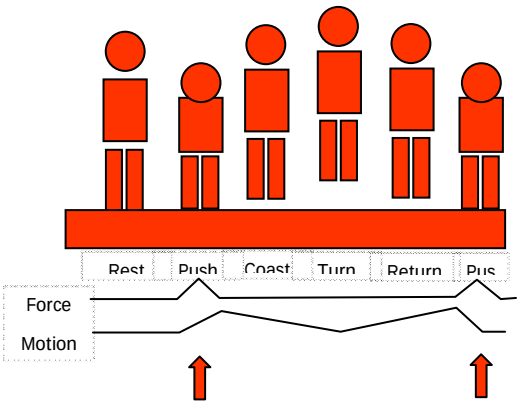


Figure 10: Sketch of a person bouncing and the force and motion that they experience.

MOTION TRACES EXHIBIT LARGE SCALE STRUCTURE, FREQUENCY DOUBLING AND LOTS OF NOISE

To investigate this relationship a little more I recorded several samples of motion from the introduction of "Us and Them" – a 1970s song by Pink Floyd. I used head bob, clapping and dancing and just recorded the sum of squares difference signal as recorded from the camera. (Figure 11abc).

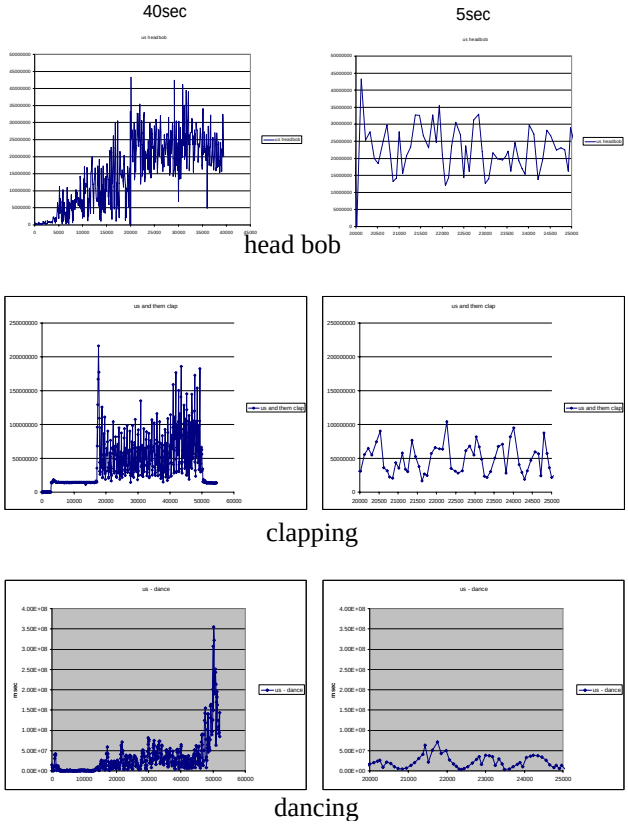


Figure 11: Sum of squares motion traces for three motions during a song intro: head bob, clapping and dancing. The left column shows large scale (40s) structure, while the sample-by-sample structure is shown at right. Most plots show a significant amount of noise.

Notice that the different types of motion have dramatically different fine scale structure. The headbob is noisy with just a hint of periodic structure. The clapping is fairly regular with frequency doubling, and the dance is a smooth almost sinusoidal motion.

TRIGGERING MOTION FROM VIDEO IS HARD

The combination of camera latency, motion blur, frequency doubling, and interpretation of beat make this a much harder problem than it might seem at first. That wont prevent us from pressing ahead and trying, but it may condition our expectations somewhere. There is hope that more sophisticated machine-learning approaches could solve the problem of what features are worth looking at. For now we will have to make do with the simple framework we have now.

In particular I want to underscore that this problem is somehow intrinsically harder than the task of detecting beat in music, principally because the motion in video is a noisy and delayed copy of an interpretation of the beat present in the music.

MOFLO USES THRESHOLD DETECTORS TO GENERATE TRIGGERS

Once the motion estimate has been made, Processing sends and OSC message to chuck with the frame difference. Chuck keeps estimates of the min and max signal and average signal using envelope detectors and a moving average filter with a time constant of about 5 seconds. In order to generate a trigger the motion signal must be a local maximum and also exceed the average motion value by a certain amount. This will tend to preclude certain large dynamical shifts, but will work for a lot of music. A diagnostic is show in figure 12.

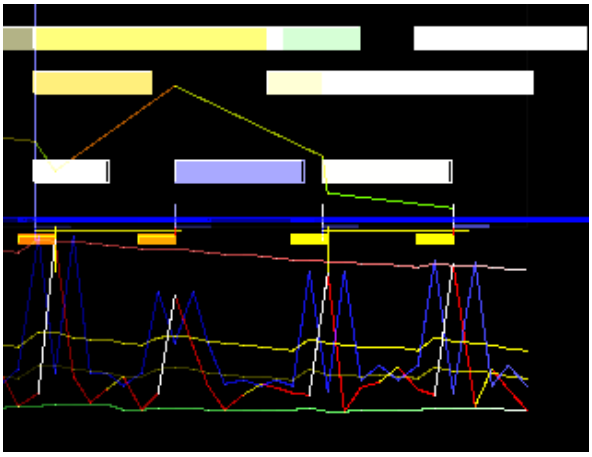


Figure 12: A screen shot of the trigger detection. Large white bars in the top of the page are beats in a sequencer while the motion signal (red and white line) is below.

THE SPACE BETWEEN MUSICIAN AND AUDIENCE

After the triggers are generated there are many ways that they can be used. Moflo has in general three modes. In the first mode, each trigger advances an internal sequence one beat, but the sequencer never advances on its own. Motions then produce little squirts of music. This is perhaps the most engaging mode that Moflo has.

The second mode, Moflo runs as a free-running music player. The motions are measured but never used.

The third mode is in between. Moflo will play it's programmed sequence at its notion of the curent tempo, but if it sees motion triggers it will adjust its sense of phase slightly. How is this done?

Other techniques [15] that do beat detection in audio typically split the audio signal into subbands, followed envelope detector/rectifier/differentiator to find attacks fed into a bank of comb filters to find the most likely tempo. An important observation in [15] is that the breaking into subbands is a crucial step and we cannot combine the results of this too early. If frequency subbands are like spatial windows in video processings, there is perhaps some reason to reconsider our approach.

The advantage of the comb filter approach is that it is relatively statistically stable – the comb filter looks at a processed version of every sample, while Moflo only looks at triggers on some samples. A disadvantage is that it integrates the results of several seconds of audio and then makes a good prediction about overall tempo and phase. It doesn't generate trigger type events on its own. In addition the comb filter approach can be computationally expensive.

TRIGGERS MODIFY THE PHASE AND OR TEMPO DEPENDING ON WHEN THEY ARE RECEIVED

In contrast Moflo is based on the algorithm that an actual human performer might use to synchronize foottaps to a beat. Moflo has a simple two-variable state: one for tempo and another for phase. When a trigger arrives, the Moflo updates these variables.

Figure 13 shows a state space framework that captures the essence of algorithms that update their state with each trigger. Basically any algorithm can be conceived of a function u:

[1] $u: [t,p] \rightarrow [t,p]$

Tempo might be defined as the number of milliseconds per beat (t) and phase in terms of the number of milliseconds until the next beat (p). We can see that

[2] $t < p$

is required. When no trigger is present, p gradually decreases with time until it wraps after reaching zero. When a trigger arrives, the algorithm applies the update function u.

The choice of u will influence whether updates are quick or slow and how much tempo offset induces a frequency change. There are many other choices including the size of the dead band (where triggers don't affect the phase or tempo).

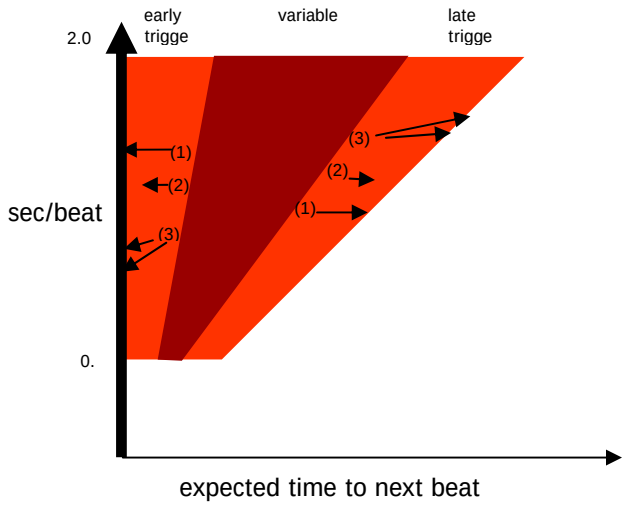


Figure 14: tempo/phase space. Different beat tracking algorithms (1), (2), (3) might apply different update rules. Algorithm (1) snaps the phase to the phase implied by the most recent beat while (2) averages the expected phase with the trigger input. Algorithm (3) also updates the bpm in addition the phase updates. The dark area is a dead band where triggers are ignored.

For now Moflo implements an algorithm like (3) from figure 14 – updates from triggers produce gradual changes in phase and bpm. This produces reasonably good results if the input video has a strong periodic component, but more tuning is needed. Even better we should extend our formal framework to derive expressions for the lock time and stability of various update functions and then select one that is somehow optimal.

TEMPORAL RUNAWAY - AUDIENCE FEEDBACK ADDS INSTABILITY

In addition to the controlled input response function, we have to consider the possibility that the audience will adapt their behavior to the input. My initial experience was that changes in the tempo of sound produced strong changes in the user behavior, which would feed back and result in a kind of "temporal runaway" as shown in figure 15.

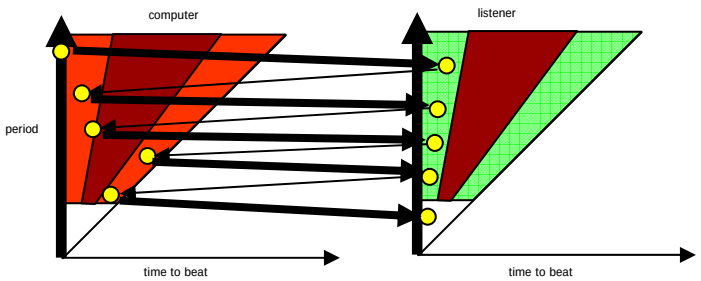


Figure 15: "Temporal Runaway" from two coupled beat tracking systems. The computer is on the left and the user

is on the right. Both adapt to a change in the other by increasing tempo until something breaks.

It is true that temporal runaway generates a lot of activity on the part of the user, though it doesn't last very long. Something about the optimization objective of the system has to include a notion of joint stability. More advanced systems could even play with the level of stability as a parameter for a more engaging experience. More clarity is needed here.

SUMMARY LESSONS LEARNED

Creating this round of prototypes was a satisfying and educational experience. We learned that:

- beat detection in video is a hard problem with counfounds of latency, motion blur, noise, and relation of beat to motion.
- motion traces from the camera show a promising level of structure but things right now are very dirty.
- granular and sampled sounds can be quite engaging even when created on the fly with very little planning. The time constant of engagement is on the order of seconds.
- users are most engaged (so far) by the direct mapping of triggers to musical time. Locking to the beat produces confusion about who is in control, often resulting in temporal runaway.

FUTURE WORK

There are many possible directions to go from here and it certainly seems fruitful to continue. Top priorities might be:

- improving signal quality and fps by upgrading hardware and drivers
- solving the beat tracking stability and lock time math problems and re-testing the beat locking with a controller which we expect to be jointly stable with what humans do
- as a part of the previous step measuring the step response of motion to tempo to determine how humans do beat tracking
- improving the overall musicality of the piece, especially by being able to call on a diverse library of sound samples which can be interesting over longer periods of time

I'm very interested in any help of advice you can provide. Please email me your thoughts and suggestions as soon as possible.

Figure 16: Intial customer beta tests looked promising…

Figure 17: Until we realized the customer wanted a tactile system. ;)

REFERENCES

[1] Gabayan, K. "A Vision-Based Whole Body Interface," Demonstration at the IEEE International Workshop for Multimedia Signal Processing 2005 (MMSP'05), November 2005, Shanghai, China.

[2] Camurri, A., Krumhansl, C., Mazzarino, B., and Volpe, G. "An Exploratory Study of Anticipating Human Movement in Dance". 2nd *International Symposium on Measurement, Analysis and Modeling of Human Functions.* June 14-16, 2004, Genova, Italy.

[3] McAleer, P., Mazzarino, B., Volpe, G., Camurri, A. Paterson, H., Smith, K., and Pollick, F. "Perceiving Animacy and Arousal in Transformed Displays of Human Interaction". 2nd *International Symposium on Measurement, Analysis and Modeling of Human Functions.* June 14-16, 2004, Genova, Italy.

[4] Kensen, K. and Andersen, T. "Beat Estimation on the Beat". 2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. October 19-22, 2003. New Faltz, NY.

[5] Lee, E., Grull, I., Kiel, H., and Borchers, J. "conga: a Framework for Adaptive Conducting Gesture Analysis". *NIME* 2006, June 4-8, Paris, France.

[6] Murphy, D. Andersen, T. and Jensen, K. "Conducting Audio Files Via Computer Vision". Gesture-based Communication in Human-computer Interaction. Springer Berlin. 2004.

[7] Kim, TH., Park, S., and Shin S.Y., "Rhythmic Motion Synthesis Based on Motion-Beat Analysis". ACM Transactions on Graphics, 22:3. July 2003. pp393-401.

[9] Maki-Patola, T., Hamalainen, P., and Kanerva, A. "The Augmented Djembe Drum – Sculpting Rhythms". *Proceedings of the 2006 International Conference fon New Interfaces for Musical Expression (NIME).* Paris, France. pp 364-369.

[10] Lane, J., and Lalioti, V. "Interacting with Reflections in Virtual Environments". APHRIGRAPH, 2001, Capetown, South Africa.

[11] Kidron, E., Schechner, Y and Elad, M. "Pixels that Sound". Proc. IEEE Computer Vision and Pattern Recognition. 2005.

[12] Seitz, S. and Dyer, C. "View-Invariant Analysis of Cyclic Motion". *International Journal of Computer Vision*, 25, 1-23, 1997.

[13] EyeToy Groove™. Sony Corporation, 2004.

[14] Mary Mikhail; Giovanni Palumbo; Jinane Mohammad; Mohamed El-Helaly; Aishy Amer. "An Online System for Synchronized Processing of Video and Audio Signals", *Canadian Conference on Electrical and Computer Engineering, 2006.* May 2006 Page(s):2065 – 2068.

[15] Scheirer, E. 1998. "Tempo and Beat Analysis of Acoustic Musical Signals." Journal of the Acoustical Society of America 103(1):588–601.

[16] Weinberg, Gil and Driscoll, Scott. "Toward Robotic Musicianship". Computer Music Journal 30:4 pp. 28-45, Winter 2006.

[17] Puckette, M., T. Apel, and D. D. Zicarelli. 1998. "Real-Time Audio Analysis Tools for Pd and MSP." Proceedings of the 1998 International Computer Music Conference. San Francisco, California: International Computer Music Association, pp. 109–112.

[18] Desain, P., and H. J. Honing. 2002. "Rhythmic Stability as Explanation of Category Size." Paper presented at the 2002 International Conference on Music Perception and Cognition, University of New South Wales, Sydney, July 17–21.

[18] Wang, G. and Cook, P. 2004. ChucK: a programming language for on-the-fly, real-time audio synthesis and multimedia. In *Proceedings of the 12th Annual ACM international Conference on Multimedia* (New York, NY, USA, October 10 - 16, 2004). MULTIMEDIA '04. ACM, New York, NY, 812-815.

[19] Reas, C., Fry, B., and Maeda, J. 2007 *Processing: a Programming Handbook for Visual Designers and Artists*. The MIT Press. Processing citation