Automatic Remixes and Music Videos Using MIDI Files

Lucas Karahadian

December 15, 2015

1 Introduction to YTPMVs

In 2007, YouTube user Vorhias uploaded YTPMV: You Are An Egghead, the earliest example of what is now known as a YTPMV. YTPMVs are music videos that take short memetic video clips and remix the audio and video following a preselected backing track. Here is a more modern example of a YTPMV. Typically these videos are very carefully handmade and take several painstaking hours of timing and tuning to get them sounding good. In this paper we attempt to automate a very simplified version of this process. The videos we create consist of side-by-side clips without any special effects autotuned to match the notes in a MIDI file.

2 MIDI Files

We will be using MIDI files to guide the creation of our videos. MIDI files are musical files that hold musical instructions as opposed to raw waveforms. They consist of a collection of musical notes containing (importantly for this project) fields for track number, MIDI number, start time, and instrument. MIDI files are used in this project to determining timing and pitch for the clips.

2.1 MIDI Numbers

MIDI numbers are an encoding of audio frequency used to record exact notes in a MIDI file. The MIDI numbering system is defined such that a concert A (440 Hz) has a MIDI number of 69 and an additive increase of 12 MIDI numbers corresponds to a doubling of frequency.

3 Clip Selection

The process described here works best assuming a certain format for the supplied video clips. These clips should ideally be short and have a single sound/volume peak. Clips like these can be treated as a musical note with few problems with overlapping sounds. It is conceivable that this process could be done automatically (given a single video file, the computer can chop it up into good clips), but for the purposes of this paper, we will assume that this process has been completed manually.

4 Audio Manipulation

Tuning our video clips can bee seen as a problem of pitch-shifting a clip until it's dominant FFT bin is at a desired frequency. This can be easily accomplished by using a phase vocoder to time-scale the audio signal and resample it back to its original length. However, in doing this naively we run into an interesting problem. When complex sounds are pitch-shifted in this manner by a large amount, the resulting sound can sometimes be not recognizable as the original. This isn't a problem in traditional YTPMVs because they do the pitch shifting by hand and can avoid sections that sound bad. Because we want the computer to do this automatically and we don't have a way for the computer to detect bad sounding shifts, we need a safeguard to prevent large pitch-shifts. The safeguard I decided on was to shift the numbers in a single MIDI track such that the mean is between 57 and 69, and shift the dominant frequency in the clip so that it is in the same range. From these two guidepoints we can calculate the desired pitch-shift

for the original clip that minimizes the total pitch-shift in a whole track while maintaining the original note-to-note relationships.

5 Video Localization

Since we aren't concerning ourselves with any fancy visual effects, the most we need to do for the spatial placement of the clips is to divide the final frame size into as many sections as the number of MIDI tracks we are reading.

5.1 Temporal Localization

The final step is timing the clips so that they line up with the MIDI file. Luckily for us, we have access to the note starting times in the MIDI file. These times are stored in seconds so we have to multiply them by the desired framerate of the video to find which frame the note should start at. Also in order to place the clips correctly, we need to decide what point in the clip corresponds to the actual attack of the note. I defined the attack of the clip to be the time of the maximum value of the audio signal. This definition holds for sort, single-peaked sounds as described in the Clip Selection section of this paper. By lining up the defined attack-frame with the MIDI's note start value, we guarantee that the note will sound exactly when it needs to.

However, this does not account for what happens when there is overlap between clips. For example, when two notes are in such rapid succession that the whole clip does not have time to play. For this, I decided to crop the videos temporally so that if there is overlap, both clips are cropped at the halfway point between the notes. This results in an envelope around each note containing as much of the clip as it can.