GPU-Accelerated Beat Detection for *Dancing Monkeys*

In music-based rhythm games such as *Dance Dance Revolution (DDR)*, a player must accurately perform actions based on visual patterns that match the beat of the background song. These visual patterns are often created by manually by the game developers. This manual process can be eliminated through the usage of software that use beat detection algorithms to aid in the automated generation of such visual patterns. The open source program *Dancing Monkeys*, written by Karl O'Keefe of Imperial College London, employs beat detection to generate DDR-style stepfiles for arbitrary songs[1]. In this project, I propose the modification of the beat detection algorithm in *Dancing Monkeys* to take advantage of the parallel processing capabilities of the GPU.

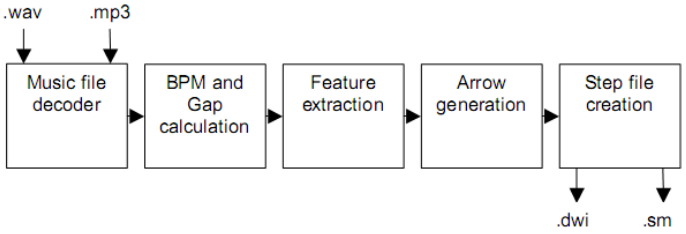

Figure 1. *Dancing Monkeys* system architecture

The details of O'Keefe's *Dancing Monkeys* program are outlined in his project report[2]. For the beat detection portion of his project, O'Keefe implements a brute-force algorithm similar to the one described by Will Archer Arentz's paper, "Beat Extraction from Digital Music"[3]. First, a waveform is generated consisting of the highest peaks from the song's original waveform. This waveform is then smoothed via a low pass filter for easier analysis. The BPM (beats per minute) of the waveform is then determined through brute force comparison tests across the range of 90-200 BPM and checked for best fit. These comparison and fit test are then repeated with narrower ranges. The closest fit is determined to be the BPM if it falls within the acceptable error margin.

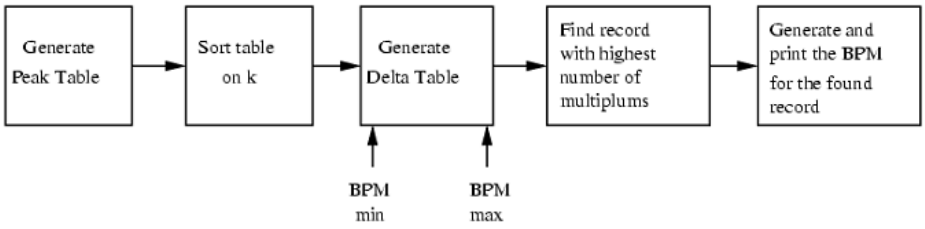

Figure 2. Arentz's BPM determination algorithm; *Dancing Monkey* uses something similar

The beat detection part of *Dancing Monkeys* is written in MATLAB, compiled to run on the CPU, and linear in execution. Due to the brute-force nature of the algorithm and its multiple passes, the process can be very lengthy, often taking twice the duration of the song itself or more (depending on your CPU). Because the comparison and fit tests across the BPM range being checked are independent events, however, this process can be parallelized. In addition, the test process itself is pure arithmetic calculations, which GPU units are well suited for. MATLAB also supports some CUDA kernel integration[4] and previous works have shown over 10x speedups in GPU-accelerated music beat analysis[5]. By modifying the beat detection algorithm to run tests in parallel and on the GPU, it is possible to significantly reduce the time needed to accurate detect the BPM of the loaded song files. This would allow for significantly faster pattern generation times (compounded if the user wants to batch analyze an entire song library).

[1] http://www.monket.net/dancing-monkeys-v2/Main_Page
[2] http://www.monket.net/files/dancingmonkeys/DancingMonkeys.pdf
[3] http://www.idi.ntnu.no/~willa/papers/bpm.pdf
[4] http://www.mathworks.com/discovery/matlab-gpu.html
[5] http://blog.accelereyes.com/blog/2011/08/11/music-beat-analysis-with-jacket/