Checklist of Deliverables for the Final Project

Jonathan Burnham, Christopher Hoffman, and Kevin Miu

The basic features are marked with just a number.

More advanced features are marked with an additional *. A functioning project could result even without these goals or scaled-down versions of these goals.

The riskiest and less-likely goals are marked with a **. These goals would improve the game, but not in an absolutely necessary manner.

1 Video

1.1 Video Input

1. Modification ZBT video storage to accept 18-bit color. (Modification to Javy's Module by Kevin)

2. Modification ZBT and VRAM addressing to improve image resolution. (Modification to Javy's Module by Kevin)

3. Shift and horizontally reverse video image and clean up spurious pixels. (Modification to Javy's Module by Kevin)

1.2 Video Processing

1. Calibration and identification of 4 different colors. The colors are calibrated by setting relative thresholds. (Module written by Kevin)

2. Calculation of the center of mass for each of the 4 colors. (Module written by Kevin)

3.* Velocity determination of each of the center of masses. (Module written by Kevin)

4.* Creating a relation between movement and beat for each of the center of masses and then detecting beats. (Module to be written by Kevin)

1.3 Output Module

1. Display of the individual playing the game and the location of the four center of masses. (Module written by Kevin)

2.* Display of the reference quadrants. (Module to be written by Kevin)

3.** Display of swatch positions relative to quadrants. (Module to be written by Kevin)

4.** Display of a virtual "Stickman" emulating the dance of the user. (Module written by Jon)

5.** Make various bode parts of "Stickman" scalable. (Module written by Jon)

6. Creation of reference command images. (Kevin)

7. Script to process images and store them on the ROM. (Kevin)

8.* Display of 144 different reference dance commands. (Module written by Kevin)

9.* Scrolling of reference dance commands timed with beat. (Module to written by Kevin)

2 Audio

1. Script to convert WAV files into a two's complement data. (Kevin)

2. Beat verification in MATLAB. (Kevin)

3. Demonstrate data storing on Flash Rom and read back with logic analyzer. Data load demo will be either a partial demo with a BRAM or a full load using USB. (Module written by Chris)

4. Demonstrated music playback from Flash memory. (Module written by Chris)

5. Demonstrate that data stored on Flash ROM is persistent by power cycling labkit after loading data and then playing music. (Chris)

6. Demonstrate beat detection at real speed, by blinking a LED to the beat. (Module to be written by Chris)

7. Demonstrate storing detected beat in a BRAM. (Module to be written by Chris)

8.* Perform beat detection at a clock speed higher than the audio's 48Khz, preferably at 27Mhz. (Module to be written by Chris)

9. Demonstrate normal game operation, were music is played back and beat data is read out of BRAM on request from the controller. (Module to be written by Chris)

10.** Audio filtering to improve the sound quality. (Module to be written by Chris)

11.** Beat detection with band pass filtering to improve beat detection accuracy. (Module to be written by Chris)

12.** Demonstrate tone detection by displaying it on screed or using leds. (Module to be written by Chris)

13.** Demonstrate storing performing tone detection preprocessing and storing it in a BRAM. (Module to be written by Chris)

3 FSM

3.1 Stickman Calculations

1.** Averages shoulder y positions to determine the top of the user's torso. (Module to be written by Jon)

2.** Connects hand and shoulder center of mass positions while drawing a neck, head, torso, and legs in real time in a way that resembles a person. (Module to be written by Jon)

3.2 Beat Comparison

1. Is able to access the beat memory and compare the real-time user beat with the previously determined audio beat. (Module to be written by Jon)

2. Computes a reasonable score based on the difference in arrival of the beats. (Module to be written by Jon)

3.3 User Inputs

1. Is able to process buttons and switches set by the user to control the systems operation, including: scrolling dance pictures, sending calibration data to the video component, doing beat detection, selecting the proper song, stopping/playing back the song, comparing the quadrant position with the instructed position, and calculating a total score. (Module to be written by Jon)

4 USB Interface

1.** Using a laptop and a DLPDesign USB adapter to send data to labkit. (Module to be written by Kevin)

2.** Storing the USB data on the Flash ROM. (Module to be written by Kevin)

3.** Creating a means of addressing such that 60K (data limit of USB adapter) pieces of data can be incrementally stored on the ROM. (Module to be written by Kevin)

5 Other

1. Putting together a shirt with 4 reference colors. (Kevin)

2.** Improvement of the HEX display. (Kevin)