User Documentation:
Using the Nintendo Wii to Assess Motility

Rebecca Groveman

Before you begin, you should download a Java Bluetooth implementation – I used Avetana (`http://www.avetana-gmbh.de/avetana-gmbh/produkte/jsr82.eng.xml`), but another option is Bluecove (`http://code.google.com/p/bluecove/`). You should then install Wiigee (`http://wiigee.sourceforge.net/`) and my software. You will also need access to MATLAB, and to install the Camera Calibration Toolbox (`http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`), for the stereo triangulation of the IR points. Instructions for the various next steps are below.

1 Constructing a Calibration Square

The calibration square is a square pattern consisting of four infrared LEDs and a power source. MATLAB does its calculations based on assuming that each side of the square is exactly length one, so it's important that the LEDs be the same distance from each other. A screenshot of the calibration square, from the "Hacking Wiis for 3D tracking" YouTube video (`http://www.cl.cam.ac.uk/~sjeh3/wii/`), can be seen in Figure 1.

I used a nine-volt battery as a power source, but others have used rechargeable batteries. Note that as the battery runs low, the LEDs may blink on and off, making data gathered useless.

2 Connecting the Camera Wiimotes

To connect to the camera Wiimotes, I recommend a script like the following:

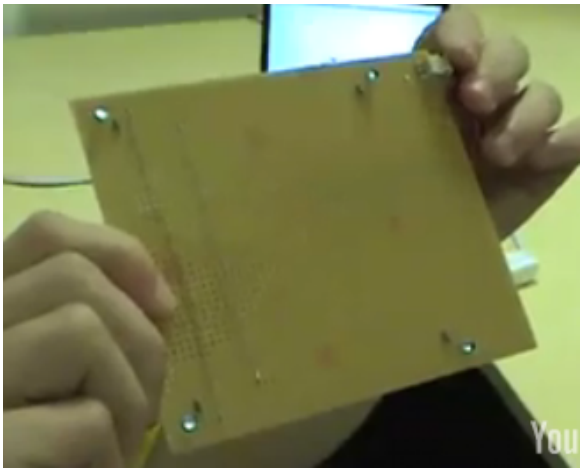

Figure 1: Screenshot of Calibration Square

```
#! /bin/bash

cd /path/to/project/bin

java -ea -cp .:/path/to/avetanaBluetooth.jar
-Djava.library.path=/path/to/libavetanaBT.jnilib View/WiimotesConnectorGUI 2
```

The script creates the WiimotesConnectorGUI with two Wiimotes – it can be initialized with 1 or 3 Wiimotes by changing the parameter. From there, you can save the IR output to file using the `Record Output` button. For each camera Wiimote connected, you need to specify a different file. You can then press the `Find Wiimotes` button to start Wiimote discovery. If Wiimote discovery is unsuccessful for two minutes, discovery halts.

A screenshot of the WiimotesConnectorGUI can be seen in Figure 2.

You can also change the LEDs on the face of the Wiimote using the `Set LEDs` checkboxes. (This is a bit buggy. Think of it as an adventure!)

Note that at any time you can change the output files by again pressing the `Record Output` button. When you have finished gathering data, you can disconnect the Wiimotes and quit using the `Exit Program` button.

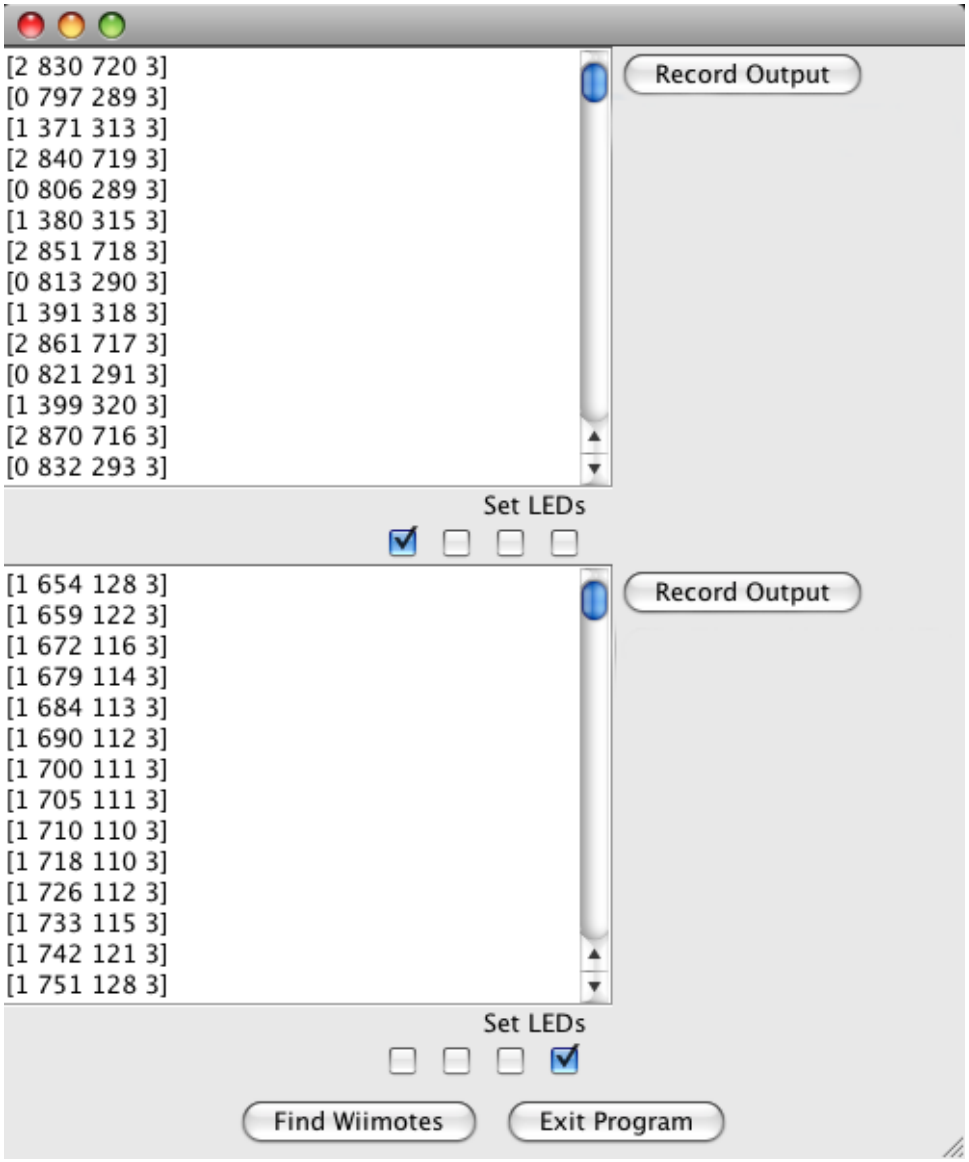

Figure 2: Screenshot of Wiimote Interface

3 Using the Calibration Square

The calibration square is used to calibrate the camera Wiimotes, as described in the documentation for the Camera Calibration Toolbox (`http://www.vision.caltech.`

edu/bouguetj/calib_doc/index.html). To calibrate the cameras, hold the square where it can be seen by both camera Wiimotes, and move it around. You should move the square pattern in all three dimensions, so that each camera sees many different configurations of the square. A good example of this calibration can be seen in the "Hacking Wiis for 3-D Tracking" YouTube video (http://www.cl.cam.ac.uk/~sjeh3/wii/).

As long as you keep the camera Wiimotes in the same locations, you only need to collect this calibration data once.

4 Gathering Additional IR Data

After you've collected calibration data, you're ready to collect regular IR data. There are two options here: you can just gather the IR data, or you can gather it while also gathering gesture data using a third Wiimote.

My system is set up to always expect and read in four IR points at a given time, so if you want to use fewer IR sources than that, it would need a little tweaking, though I expect that would be fairly straightforward.

You have two options for the IR sources. One is to use IR LEDs as with the calibration square. The only problem with this choice is that each LED needs a power source, which is, to use a technical term, fiddly, to connect to the body.

The other alternative is to use reflective tape. The problem with this choice is that you need a fairly large and strong infrared source for the tape to reflect. During my project I was unable to get the reflective tape to work, exactly because I didn't have an IR source strong enough.

What this means is that you're pretty much in uncharted territory here...but have fun! If you find a good solution, I'd love to hear about it (you can find me at grove20r@mtholyoke.edu).

5 Connecting the Playing Wiimote

The playing Wiimote is connected to the computer using Wiigee's demo GUI. I use a script like the following to connect:

```
#! /bin/bash

cd /path/to/Wiigee/bin
```

```
java -ea -cp .:/path/to/wiigee.jar:/path/to/avetanaBluetooth.jar
-Djava.library.path=/path/to/libavetanaBT.jnilib wiigeemain
```

The script starts Wiimote device discovery automatically, so once you've run the script you should press buttons 1 and 2 on the Wiimote so the computer can find it. Wiigee automatically saves accelerometer and gesture output to file at its project root.

Once you've connected the playing Wiimote, you're ready to do gesture recognition!

6 Writing MATLAB Scripts

There are two main types of MATLAB scripts generated by my project: calibration scripts and triangulation scripts. The calibration scripts are necessary to calibrate the camera Wiimotes individually. The steps are as follows: use my program to generate calibration scripts from the desired raw IR data, open MATLAB, run the calibration scripts for the left and right cameras, run the stereo calibration to generate a `Calib_Results_stereo.mat` file, and then you're ready to run the triangulation scripts.

To generate scripts, run my `IRControlPanel` program. From there, you can select the left and right files, select either to create a calibration script or a triangulation script, and generate the scripts. If you're calibrating the system, once you have your calibration scripts, open MATLAB.

At the MATLAB command line, CD to the directory your script is in. Run your script by typing its name without its extension (for example, `calibL0`, rather than `calibL0.m`). Once you have run the script, it should generate two files: `Calib_Results.m` and `Calib_Results.mat`. Rename these according to left and right (`Calib_Results_left.m`, for example). Once you have both the left and right results files, you're ready to do stereo calibration.

The easiest way to do stereo calibration is to type `stereo_gui` at the MATLAB command line. This command will bring up a GUI that will allow you to load the left and right files, do the stereo calibration, and see a visualization of the stereo calibration.

After stereo calibration is complete, you should have a `Calib_Results_stereo.mat` file. Now you can calculate the 3D location of any data you've gathered while the camera Wiimotes are in the same place they were during calibration.

Generating triangulation scripts is exactly like generating calibration scripts: run my `IRControlPanel` program as before, but this time, select the `Regular Data` radio

button. After you run it, it will generate a triangulation script from the left and right files that can be run through MATLAB (by typing the name of the script without the extension, as before). Note that in order for the script to be successful, it must be in the same directory as the `Calib_Results_stereo.mat` file. Running this triangulation script will load the left and right matrix files, do the stereo triangulation calculation, and will graph both the left and right data in 3 dimensions.

7 Using the Gesture Visualization Program

To use the gesture visualization program, run my `GestureGrapher` program. From there it should be fairly straightforward: you can select a new file using the `Choose New File` button. When a new file is selected, the program breaks the accelerometer data into gestures, which you can select between using the dropdown menu. The accelerometer data from that gesture can then be seen in the text box. You can then graph the data, using the `Graph Data` button, or clear the graph, or quit the program.

A screenshot of the gesture visualization program can be seen in Figure 3.

8 Separating the IR Data into Gestures

To separate the IR data into gestures, use my `IRGesturePanel` program. It takes in a raw gesture file from Wiigee and raw left and right IR files from my GUI. From there, it walks the three files and generates left and right matrix files for each gesture seen, and associated triangulation scripts that load the left and right files, perform the stereo triangulation, graph the data, and save the three-dimensional output. Just as before, the triangulation scripts must be in the same directory as the `Calib_Results_stereo.mat` file.

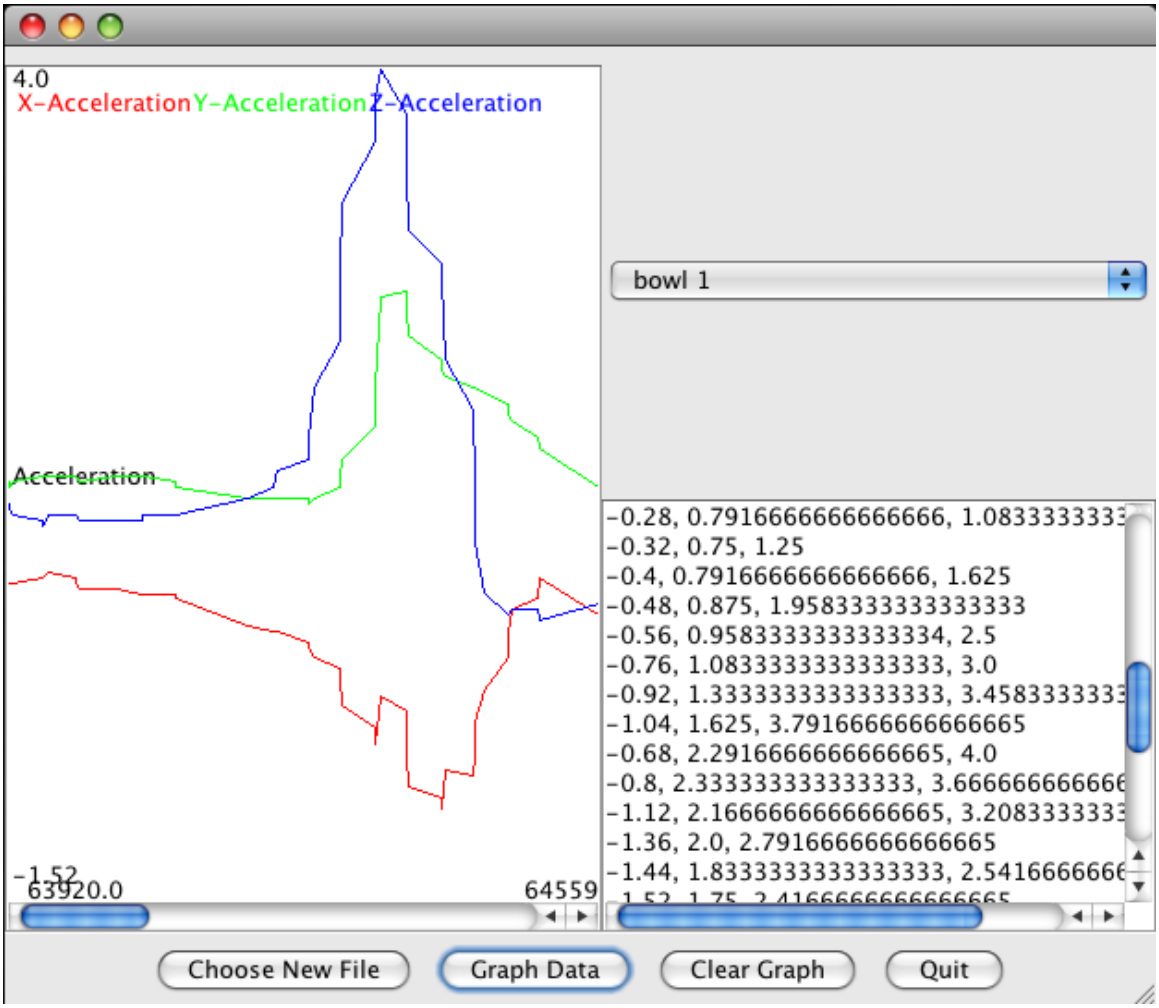

Figure 3: Gesture visualization program