Heavy Feet

an idiosyncratic wireless electronic foot controller

Carey Dodge
Sonic Arts Research Centre
Queen's University Belfast
Belfast, UK
doodgeroo@yahoo.ca

ABSTRACT
Heavy Feet is a wireless electronic device that is worn on the feet. The device measures the movement and impact of the feet and sends the data to a computer where it can be mapped according to the user's wishes. It was first used to create a comically exaggerated and dynamically changeable stomping sound for a solo dance performance.

Keywords
feet, accelerometer, force sensitive resistor, wireless, bluetooth Arduino, wearable electronic device, Max/MSP, Ableton Live

1.INTRODUCTION
Heavy Feet came from a desire to create a wireless wearable electronic device that a performer(s) could use to control audio. The criteria for the creation of the device was that it be robust, invisible (i.e. able to be woven or hidden in a costume) and expressive.

My philosophy in creating *Heavy Feet* was congruous to Joseph A. Paradiso in his work on *FootNotes*,

> Our philosophy has been to give the dancer access to a sufficiently large selection of simple sound-action mappings to enable them to assemble and deliver an engaging performance, rather than to abstract too deeply into a more sophisticated level of gestural determination[1].

I wanted to have "simple sound-action mappings" in order to "deliver an engaging performance". This was the main philosophy driving all my choices in the design of *Heavy Feet*.

2.HARDWARE DESIGN
2.1Motion Sensing
Initially, there was to be solely accelerometers on each foot measuring the X, Y and Z axes and no other sensors. I was planning on using these sensors for the measurement of movement and impacts. After some experimentation, I switched to only X and Y accelerometers for the measurement of movement and a Force Sensitive Resistor (FSR) for the measurement of impacts. This was due to the fact that a sharp flick of the foot had the same gesture in the sensor data as an impact. This proved to be too unreliable for the measurement of impacts. I could make a calibration that would be correct 80-90% of the time but I needed it to be correct 100% of the time.

There were two reasons for cutting away the Z axis. One was that I only had six analogue inputs on my micro-controller (bluetooth Arduino). The other was that I found there was little interesting data coming from the Z axis or at least the data did not add anything meaningful that was not picked up by the X and Y axes. This, of course, might have changed depending on the performance. The prototype performance was based on stomping motions which generally remained on a 2-dimensional plane. The interesting data from the accelerometers came from the deviation from the 2-dimensional plane. I could imagine an equally interesting interaction with the Z-plane if the performer spent more time in a horizontal or inverted position or if more precise tilt, pitch and yaw measurements were necessary.

2.2Impact Sensing
As mentioned above, I switched from an accelerometer to a Force Sensitive Resistor early on in the development stage as experimentation proved that the accelerometer was not 100% reliable for the measurement of impacts.

The FSR was placed on the heel of the performer. The biggest issue in this project was the sweatiness of the performer's feet. As the performer's feet began to sweat and as they stomped and moved their feet about, the FSR gradually began to stick to the foot and this created a pressure reading from the sensor. Originally, the FSR was in a shoe but this approach was discarded as the wearing of the shoe created pressure between the foot and the shoe sometimes into the high range of the sensor. Fixing the sensor to the bare foot of the performer proved to be the most successful. Wearing a sock did not alter the sensor data and this allowed the sensor to be invisible.

Possible solutions that were not explored are larger shoes (e.g. flip flops or clown shoes), a more advanced shoe design that mechanically creates space between the foot and the FSR and another solution (that is not really a solution but another approach) is to put the sensor on the ball of the foot. This alters the concept of the device somewhat but it might prove more reliable than having the FSR on the heel as there is generally less pressure placed on the ball of the foot and it is easier to control pressure placed on the ball than on the heel.

2.3Arduino Hardware
For details on the Arduino please see the website (http://www.arduino.cc/). There is detailed documentation available there. I used the Arduino BT-V06.

3.SOFTWARE DESIGN
3.1Arduino Software
I adapted a preexisting Arduino patch for use with *Heavy Feet.* The patch I used simply reads the six analogue inputs of the Arduino and outputs the data at a baud rate of 115200[1].

[1] An important technical note about the bluetooth Arduino: it only works at a baud rate of 115200. This can be a serious technical limitation especially when receiving and sending data on multiple bluetooth Arduinos with a single computer.

Again, please see the Arduino website for documentation and example code (http://www.arduino.cc/).

3.2Max/MSP

Max/MSP was used for scaling, calibration and playing of sound material.

3.2.1Scaling and Calibration

For the accelerometers, the patch first scaled the data to get the maximum range of the sensor input. The data was then 'smoothed' to reduce jitter. The 'line' object was used to have the data *walk* from integer to integer as opposed to *jump* from integer to integer. Another part of the patch dynamically found the mean of the data. Finally, the data went through a final scaling from 0 to 127 as the data was used for midi controlled sound effects. For example, as the accelerometers moved above and below the mean, there was a corresponding increase and decrease in sound effects.

For the FSR's, the patch did some minimal smoothing and limiting. It was necessary to gate data below a certain point as even with the hardware and physical solutions described above there was still a small amount of pressure data that seeped in depending on the sweatiness of the feet and/or if the sensor slipped from its original position. The pressure data was then used to trigger sounds, control amplitude and alter low frequency notch filter.

Depending on the rapidity of the impacts there would be further effects added. (e.g. if the performer stomped with the right foot five times in one second a more extreme effect was triggered than if they were stomping one time per second.)

3.2.2Sounds

Max/MSP was used to randomly play a selection of prerecorded exaggerated impact sounds. A gating system was necessary to prevent the triggering of a sound player that was already playing. This allowed the performer to stomp as quickly as they desired.

3.3Ableton Live

Ableton Live was used for the digital signal processing of the sounds. There was a potpourri of effects used, mostly along the granular synthesis and fft vein.

This aspect of *Heavy Feet*, along with the sounds, is the part that is most open to the choice of the performance. The sounds and effects used could be anything. That being said, the more work that goes into a meaningful artistic relationship between the performer's movements and the sounds that are created, the stronger the performance will be.

4.CONCLUSIONS

Although I was satisfied with the first performance using *Heavy Feet* there are some improvements that are definitely necessary for more reliability and flexibility.

First of all, the sensors were taped onto the performer's foot. This is effective but not reliable. Building the sensors into an insole (like Paradiso's Dance Shoes[1]) would be a more robust solution.

Secondly, it would be better to have more sensor inputs. The Arduino is limited to six analogue inputs. It would have been beneficial to have X, Y and Z accelerometers and a combination of FSR's and piezos (and possibly other types of sensors). The primary reason for additional sensors would not be for more sensor data to map to musical parameters (although that would definitely be a useful possibility) but to reduce ambiguities in the overall sensor data ([1] p.11).

Finally, I would like to add a concluding thought: As open source, (relatively) easy to use micro-controller, sensor and software platforms become more readily available an interesting shift in the intersection of technology and art is possible. No longer do artists need to hire technical specialists (studio technicians of the large electroacoustic studios in the 1950's come to mind) but they can develop a 'one-off' piece of idiosyncratic technology that does not need to be adaptable to a broad range of uses. In contrast, commercial technology has to be adaptable and useful in a broad range of circumstances to be economically viable. An artist can create a specific technology for each project as opposed to trying to adapt their art to available technologies. This puts the artistic purpose before the technological capabilities. Although the dialogue between what is technically possible and what is artistically desirable is constructive and valuable, a more engaging performance is created when an artistic goal is more important than showcasing a technical marvel.

> *Everything should be made as simple as possible, but not one bit simpler.*
>
> -- Albert Einstein

5.ACKNOWLEDGMENTS

Many thanks to Ben Knapp, Paul Stapleton, Michael Gurevich, Nicholas Ward and my fellow MA's at SARC for their assistance and fearless council.

6.REFERENCES

[1] Paradiso, J., “FootNotes: Personal Reflections on the Development of Instrumented Dance Shoes and their Musical Applications,” in Quinz, E., ed., Digital Performance, *Anomalie, digital_arts* Vol. 2, Anomos, Paris, 2002, pp. 34-49.

[2] Paradiso, J., “Dual-Use Technologies for Electronic Music Controllers: A Personal Perspective” from Conference on New Interfaces for Musical Expression, Montreal, Canada, 2003.

[3] Bevilacqua, F., Müller R., Schnell N., “MnM: a Max/MSP mapping toolbox” from Conference on New Interfaces for Musical Expression, Vancouver, BC, Canada, 2005.

[4] Various authors, “Fuzzy Logic” Wikipedia, http://en.wikipedia.org/wiki/Fuzzy_logic