Semester Project: Shape Sensing, Context Aware Garment
ECE 5984 Wearable & Ubiquitous Computing

Joshua N. Edmison
Sookyoung Kim
Kevin Liu
Abstract

1. Introduction

2. Previous Work
   2.1. E-textiles
   2.2. Shape sensing
   2.3. Context Awareness

3. Design Preliminaries
   3.1. Ptolemy
   3.2. Human Motion Data Collection
   3.3. Sensor Selection

4. Simulation Structure
   4.1. Ptolemy phase structure

5. Development of Sensor Models
   5.1. Piezoelectric Film Model
   5.2. Accelerometer Model
   5.3. Verification
   5.4. Noise Addition and Error

6. Shape-sensing Application Design
   6.1. Sensor Placement
   6.2. Dead-Reckoning
   6.3. Neural Networks

7. Future Work

8. If We Had To Do It All Over Again

9. Conclusions

10. Acknowledgements

Works Cited

Appendix A
   Simulation Users Guide
Abstract
This paper presents the development of the beginnings of a simulation framework for wearable
electronic textiles and the application of this tool to explore the design space of a shape-sensing
context-aware garment, specifically, to meet the requirements of the ECE 5984: Wearable &
Ubiquitous Computing semester project.

1. Introduction
As a result of being closely coupled to the human body, wearable electronic textiles
provide a very unique set of constraints and criteria. Current design methodologies for wearable
electronic textiles have relied on an iterative design process typically consisting of multiple
prototypes. This method, while proven throughout time, is costly, time consuming, and is
capable of exploring a very limited subset of the design space. The evaluation and testing of
these prototypes usually encompasses a small number of users; insufficient for representing the
physical characteristics of the population, effecting generalization of the device. Sensors for
these prototypes are typically placed in seemingly obvious locations for lack of a more refined
set of placement rules. A simulation framework for these systems provides a tool capable of
alleviating, if not eliminating, many of these problems. By allowing the virtual placement of
sensors, swapping of sensor varieties or different sensors all together, it is possible to quickly
develop and evaluate various architectures and applications, that when finalized, map directly to
the actual hardware for prototyping. This paper presents the development of such a simulation
framework and its application to the preliminary development of a shape sensing, context-aware
electronic textile (e-textile) application as defined in the project description.

2. Previous Work
The three areas of research most closely related to the work described in this paper are e-
textiles, shape sensing, and context awareness.

2.1 E-textiles
E-textiles represent the merging of everyday garments, fabrics, etc. with various
computational elements. Despite being a relatively new field, e-textiles have appeared in a
variety of introductory applications such as antennas built into soldiers’ uniforms [1], mp3
players embedded into clothing [2], and roll up LCD displays [3]. Current research in e-textiles
is focused on novel materials such as conductive threads, simulation development, and
fundamental architectural issues such as communication and networking topologies. An
example of electronic textiles includes the beam forming fabric [Zahi’s Mobysis Paper, citation
not available, paper still in production], capable of detecting large moving vehicles.

2.2 Shape Sensing
For the purposes of this project, shape sensing was defined as follows: the ability to
detect the relative or absolute positioning in 3-space or the appropriate context of the human
body and its extremities. This has been an often-attacked problem in the areas of user input,
wearable computing, medicine, and sports training. The development of gloves to sense the
position of the fingers has been a popular application. An early example of this is the Nintendo
PowerGlove™. More modern methods include the LightGlove that incorporates lasers [4], the
air pressure glove [5], and the piezoelectric glove keyboard [6]. Shape sensing for the entire body has been usually limited to medical studies and performance enhancement sports training [7]. This is largely due to the cost and implementation types of systems that have this capability, usually consisting of camera arrays, tethers, or electrode arrays. An attempt at creating a shirt for shape sensing using variable resistive strips can be found in [8].

2.3 Context Awareness

Context awareness, while being a rather broad and encompassing term, will, for the purposes of this project, be limited to higher level abstractions of body movement such as walking, sitting, moving arm, etc. Achieving this type context information from wearable systems typically involves the fusion of many different sensors. Due to the complexity and high level of dimensionality associated with these systems this problem is often left to various types of neural networks, learning algorithms, state machines, and Markov chains [9].

3. Design Preliminaries

3.1 Ptolemy

The simulation framework was developed in C++, Ptolemy II, and MATLAB. Ptolemy II is a set of Java packages supporting heterogeneous, concurrent modeling and design [10]. This platform was chosen as required by the project description and because it provided both development freedom and a graphical user interface (GUI). This environment provides not only a common simulation base but also a higher level of abstraction for those wishing to build models of wearable systems without concern for the underlying details, achieved through the block-based GUI.

3.2 Human Motion Data

One of the more difficult aspects of simulating wearable e-textiles is the acquisition and incorporation of human motion data. Early attempts at finding publicly available datasets yielded only two-dimensional data sets collected by the BIOMECH [11] project. As one would expect, three-dimensional data is ideal. Fortunately, 3D motion datasets are available online [13], and the Locomotion Research Laboratory of Virginia Tech [12], has the capability to perform three-dimensional motion capture. They were generous enough to allow us to use their existing data sets and provided the software necessary to view the data in three-dimensions [14]. The three-dimensional data collection setup consisted of 6 cameras, a track, and a subject wearing reflective markers on the points of interest. The subject is then recorded doing various activities along the track and the output video is processed using proprietary software to obtain the final XYZ coordinates of the various markers on the subject. An example of this data can be seen in [Figure 1]. The accuracy of these reflective capturing motion systems is stated as 2mm, using 6 cameras in a 4.5m x 4.5m space. This value can vary widely depending on the calibration of the system and care in placing markers. The data files obtained provided a wide range of motion from walking and jumping to pantomiming and multi-person interactions. The data files obtained were encoded in a popular three-dimensional data file format with the extension .c3d. A freeware API, C3Dserver from Motion Lab Systems Inc. [15] was acquired to provide quick access and avoid the overhead of writing our own parser. Analysis of the obtained XYZ data, particularly the calculation of the pure acceleration, displayed a large amount of high frequency “noise”. [Figure 1] This noise is believed to be the result of calibration and capturing
inaccuracies. This resulted in the input data sets requiring smoothing prior to their input into the Ptolemy simulation. The smoothing formed the datasets to an accuracy of 2mm and produced much more appropriate acceleration curves. [Figure 1]

![Figure 1](image)

3.3 Sensor Selection

While the simulation is independent of the type of sensors used, it is still important to select the sensors to be used early in the design processes so that the application phase can be designed in parallel with the other steps of the simulation flow. As specified in the project description, we were required to use piezoelectric materials as sensors in our garments. Using the results from [6] and prior experience with piezoelectric sensors we decided to use piezoelectric film. Other form factors included coaxial cable and spray paint, however the interface method to the coaxial piezoelectric was quite difficult and the spray paint is not commercially available, thus the choice to use the film form factor was quite easy. In addition to the piezoelectric films, we explored a variety of other sensor options including gyroscopes, compasses, ultrasonic, infrared, and accelerometers. Of the considered sensors, only accelerometers were selected. Our decision was based on the idea that enough accelerometers placed strategically on the body coupled with the piezoelectric film sensors should be sufficient information to infer body position. While other types of sensors could be incorporated and may be required for less specific applications, we felt that other types of sensors would not sufficiently contribute to the richness of the sensor data for this application. The two types of sensors selected were piezoelectric films and accelerometers. In order to add realism and make feasible the construction of a prototype using the results of the simulation, the decision was made to model commercially available parts. The specific parts chosen were the Measurement Specialties Inc. DT04-52K piezoelectric film and the Analog Devices ADXL 150/250 analog accelerometer.

4. Simulation Structure

Having constrained the input type, it is necessary to develop a general simulation flow from which the various subcomponents can be constructed. After several iterations of manual
data manipulation, it became clear that the simulation would need to be capable of batch processing to achieve thorough exploration of the design space. The finalized simulation flow consisted of five phases: data retrieval, pre-processing, system simulation, and application. [Figure 2] The data retrieval phase consists of fetching the data from the specified c3d files and outputting it to a format suitable for input to MATLAB. The data is then smoothed in the pre-processing phase and stored in files where it waits to be input into the Ptolemy simulation. The system simulation phase contains the models for the piezoelectric films and accelerometers that are “virtually” attached to the various body points represented by the input data. The data produced by the simulation is then read into MATLAB where it is processed by the application phase. The application phase for our particular simulation consists of a dead-reckoning component and a neural network component but could contain any multitude of applications appropriate to the data. Finally, returning to the data retrieval phase and allowing the variation of the input data or various simulation parameters enables batch processing.

4.1 Ptolemy phase structure

The next major design decision involves the layout of the Ptolemy simulation. The primary goal behind this step of the design process was to make it as easy as possible to place “virtual” sensors on the body. A statement such as “place an accelerometer on the knee” or “place a piezo strip behind the knee joint” can usually sum up the act of placing sensors on the body; with this in mind, the body was logically dissected into commonly used abstractions and in such a manner that was appropriate for the provided data format.

We selected seven major subsections: feet, ankles, knees, hips, shoulders, elbows, and wrists. For each of these subsections a “source” block was developed which allows the user to access the motion data for various parts of that subsection. For example, the hip source block provides access to the XYZ coordinates for the left hip and the XYZ coordinates for the right hip. If the user wishes to place a sensor on the hip, say an accelerometer, or use it as contact point for another sensor such as a piezoelectric film, the user simply inserts a hip source block along with the desired sensor block and creates the appropriate connections. The interior of the source blocks simply consisted of a file I/O block called a Double Reader and the necessary output ports. In addition to source blocks, block models are needed for the selected piezoelectric and accelerometer sensors and file I/O blocks were used to store the results of the simulation.

5. Development of Sensor Models

These first iteration models attempt to capture the fundamental operating characteristics of the piezoelectric films and accelerometers. As with any simulation model, there is always the possibility for more advanced models, however one must weigh the complexity of these models with the amount of improvement they offer.

5.1 Piezoelectric Films

According to the data sheet from Measurement Specialties Inc.,[13] the equivalent analog circuit for a piezoelectric film is a strain controlled voltage source in series with a capacitance [Figure 4]. Calculating Vout as a function of Vin yields the result in [Figure 3]. This result simply states that the Vout $\alpha$ rate of change Vin. Since Vin $\alpha$ applied strain, the output of the piezoelectric film is simply the rate of change of the physical stimulus. Note that this results relies upon the assumption that $\omega << 1/RC$, which for the case of human motion is valid considering that even fast motions like running have a frequency of about 4Hz. If the
piezoelectric is be used to measure high frequency signals, it would be necessary to model the piezoelectric in the frequency domain.

The verification of the piezoelectric film model incorporated a simple pendulum as seen in [Figure 4]. The piezoelectric film, seen in red, was placed at the pendulum joint and its output attached to a data acquisition board (DAQ board). The analytical model for the pendulum yields \( \theta(t) = M * \cos(\omega_n * t) \) where \( \omega_n = \sqrt{g/l} \), \( M = \theta_{init} \). Differentiating \( \theta(t) \), yields \( \frac{d\theta}{dt} = -M * \omega_n \sin(\omega_n * t) \). Plugging in our physical model values yields \( M = 0.38 \) rad and \( \omega_n = 0.2258 \). The plot of the collected data and the results of the analytical model can be seen in [Figure 5]

\[
\begin{align*}
R & \ll \frac{1}{\omega C}, \quad \text{so} \quad V_{in} \approx \frac{1}{\omega C} \\
\text{For frequencies} \quad \omega \ll \frac{1}{RC}, \quad V_{in} & \approx V_C \\
V_{out} & = V_R = iR = R \frac{d\theta}{dt} = R \frac{d}{dt} (\cos(\omega_n * t)) \\
V_{out} & = RC \frac{d}{dt} V_{in}
\end{align*}
\]

The phase shifting seen in the later time slices of the plot are a result of dampening of the actual pendulum motion caused by friction. This was not represented in the analytical model, as we simply wanted to verify that our understanding of the piezoelectric output was correct. The plot clearly shows that the simulation model is a representation of ideal piezoelectric film output. The Ptolemy block model for the piezoelectric film includes nine input ports of the three XYZ triples making the angle to be observed and a single output port. (ex: hip, knee, heel)
5.2 Accelerometer

The datasheet for the analog devices ADXL150/250 states that the output of the device
V_{out}=V_{dd}/2 – (Sensitivity*V_{dd}/5*a) where V_{dd} is the supply voltage, Sensitivity is between 38-
400mv/g, and a is the input acceleration in the direction of the corresponding axis. This equation
is the basis of the accelerometer simulation model. Verification of this model was achieved
using the same pendulum setup that was used for the piezoelectric film test [Figure 6].
Application of accelerometers to regular human motion is problematic due to the fact most
acceleration of the body is very low g when compared to the dynamic ranges of available
accelerometers. The dynamic range of the ADXL150/250 is 50g’s whereas the acceleration we
need to detect are in the mg range. This can be achieved by increasing the sensitivity rate of the
accelerometer using a 2-pole Bessel filter provided in the data sheets. Due to the available
surface mount board, we were unable to construct the filter circuitry required to achieve the
higher sensitivity values. The maximum attainable on-chip sensitivity is 76mv/g. When
characterized using the pendulum apparatus, the output signal is somewhat noisy as a result,
however it is possible to see a general acceleration curve within the noise. To simulate the low
pass circuitry we were unable to construct, we used MATLAB to apply a low pass filter having a
full power frequency of 0.5Hz and a 3dB frequency of 1Hz to acquired accelerometer signal.
Using the analytical equations derived earlier for \( \theta(t) \), the equation for the x position coordinate
x(t)=M*\sin(\theta) can be obtained. The normalized plot of x(t) and the filtered experimental signal
can be seen in Figure[8]. The accelerometer block used in Ptolemy was constructed using three
input ports and three output ports to accommodate an XYZ triple.

5.3 Error/Noise Addition

In order to increase the realism of the Ptolemy sensor models, a random Gaussian noise
block was added to the design. This block includes parameters for adjusting the mean and
standard deviation of the noise characteristic. This noise is simply added to the output signal.
This addition of noise places an upper bound on the resolution of the system. While random
Gaussian noise is easy to produce, further sensor experiments may reveal individualistic noise
characteristics.

6. Shape-sensing Application Design

6.1 Sensor Placement

The original placement of the sensors in the simulation “virtually” positioned the sensors
directly on the points defined by the input data. However, a Ptolemy block was created that
accepts two XYZ triples defining the point where the sensor originates and the point that the user
wants to move the sensor towards. For example, if an accelerometer was placed on the knee but
the user wanted to move it 6 inches below the knee, the inputs to the block would be the knee
XYZ triple and the corresponding ankle XYZ triple. The amount of displacement is controlled
by a configurable parameter value. The effects of this translation along the knee-ankle limb can
be seen in [Figure 7]. As would be expected as the displacement value approaches the knee to
ankle length of 439mm, the curve of the sensor value approaches that of the ankle. The
displacement of interest, however, is 180mm (light blue) which is nearly the midpoint between
the knee and ankle. Note that this curve exhibits characteristics of both the knee curve and the
ankle curves. While placing sensors at the joints seems like the obvious method, placing them
in-between joints may allow you to reduce the number of sensors without losing critical joint
motion information.
6.2 Dead-Reckoning

Dead reckoning is simply using sensor output data to directly calculate some correlated variable. For example, the accelerometer output can be integrated once to obtain velocity and integrated again to obtain position. We experienced difficulty implementing this technique due to the error propagation. The resulting curves displayed the correct shape, however many of the curves followed a linear increasing line resulting in inaccurate position values. This is a continuing problem.

6.3 Neural Networks

Neural networks are systems composed of neurons, nodes which apply a weight, a bias, and a transfer function. When several of these nodes are combined with a learning algorithm they are capable of modeling complex functions or performing pattern recognition. For our purposes, they are be used to distinguish between different types of motion such running, walking, and jumping. We chose to use a feed-forward back propagation network and a self-organizing map. For the first attempt, we used twelve inputs corresponding to eight piezoelectric films in the left and right knee, ankle, elbow, armpit joints and two accelerometers on the left and right hip focusing on the X and Z accelerations. Prior to training the networks it was necessary to pre-process the data. Since we wanted to recognize patterns over time, we decided to use a sliding window mechanism and compute the average of the absolute value for each window. The various parameters for the back propagation such as number of hidden layers, number of hidden nodes, epoch length, training goal etc, were difficult to choose and mostly left to the MATLAB defaults. Two major parameters that were necessary for us to decide upon were the number of hidden nodes and the size of the sliding window. The documentation concerning neural networks in MATLAB led us to choose ten as the number of hidden nodes. Since an obvious window size wasn’t apparent, we chose to reiterate the training and simulation process over several window sizes and compare the results [Figure 9]. The neural network was able to distinguish the motion as shown in [Figure 8]. After several simulation iterations it was clear that 80 samples seemed to be the optimal window size. Using this configuration the network was able to obtain and maximum accuracy of 54%, as compared to 33% for random guessing.
7. Future Work

While this simulation framework provides a solid foundation from which to continue exploration of wearable electronic textiles there are still many features left to add. First, as with any simulation, the sensor models could be improved or tweaked to be more realistic. For a wearable system one must also consider the effects of fabric, clothing etc. on the outputs and accuracy of the sensors. To achieve this a detailed model of fabric-body interaction is necessary. Given a lack of codification and experience in the field of neural networks, and the many neural network parameters chosen without thorough reasoning, there is much improvement to be made in the neural network application. Once a solid neural network foundation is established it would be possible to possibly search the design space for optimal points over several variables. Finally, if the system is to be implemented in reality, the simulation must also include models that represent the computational framework, encompassing and processing the communication of the various sensors and systems.

8. If We Had To Do It All Over Again

If we had the option of doing it all over again we would have been more thorough in the exploration of the various types of available motion data. Had we acquired the various data sets earlier in the semester the design would have had likely been much more polished. We would have also found/purchased/borrowed the hard to find de-facto standard textbooks for neural network design. Finally, we would produce more advanced models of the sensors that included second order effects and more accurate noise representations.
9. Conclusions

The ability to use a simulation framework to explore various design configurations and applications is a very powerful design tool that not only improves design quality but also saves time. The current simulation system, while only 54% accurate, obtained these results without many design iterations or a thorough consideration of the multitude of neural network design options. However, it is clear that using piezoelectric materials in combination with other sensors to detect the shape and context of the body is an attainable goal. Given the multitude, uniqueness of the design constraints and criteria of wearable systems, it is clear that to fully explore the design space or achieve a successful design simulation will be a necessary tool.

Works Cited
[10] PtolemyII. http://ptolemy.eecs.berkeley.edu/ptolemyII/
[14] Qualysis QTRACs Software. www.qualysis.se

Appendix A

Simulation Users Guide

As mention in the above report the simulation consists of C++ code, Ptolemy models, and MATLAB scripts. The C++ module named c3d_qd.exe accepts a configuration file called fileconfig.txt and requires that the C3D server from MLS systems be installed on a window machine. The c3d files used for processing must be in the working directory of the program. The format for the configuration file is as follows (note line numbers are in red and are not part of the file):

```
1 FullSimTF.xml
2 /home/jeedmison/ptII/ptII2.0.1/project/shpgarnv/data
3 /home/jeedmison/ptII/ptII2.0.1/project/shpgarnv/simres
4 12_01.c3d 09_01.c3d 16_46.c3d 06_01.c3d 07_10.c3d
08_03.c3d 16_55.c3d 02_03.c3d 35_20.c3d 13_41.c3d 13_19.c3d
13_13.c3d 05_01.c3d 35_24.c3d 16_07.c3d
5 feet.m 0 LTOE XYZ RTOE XYZ LHEE XYZ RHEE XYZ ZERO XYZ
6 ankles.m 0 LANK XYZ ZERO XYZ RANK XYZ ZERO XYZ
```
Line 1 is the name of the Ptolemy simulation to be executed.

Line 2 is the path for the resulting data files.

Line 3 is the path for the simulation results.

Line 4 is a list of file for which to retrieve data.

Lines 5-11 are the name of the files to be generated from the indicated data files and the name of

The body part categories to retrieve including specification of coordinates (ie LANK is

The left ankle and XYZ specifies the user wants the X, Y, and Z coordinates. These four

Letter body part designations can be found in the markers file.

This file format is to be extended soon to include the type(s) of neural networks to be simulated,

methods for designating if a data set is for training or validation, the range of window sizes to be used for

pre-processing the data prior to the neural networks, various neural network parameters such as epochs

and training goal, the classes for the various data sets (to be used in training and results analysis), options

for including dead reckoning calculations, options to produce various types of statistics about the data sets

for use in system design preliminaries, the ability to reference data sets by group or subject, parameters

for Ptolemy simulations, and parameters for auto-generation of Ptolemy code.

After the data sets the MATLAB workspace must be initialized using parameters from the above

configuration file. This is done by running the MATLAB script config.m. The next step is to smooth the

data by using splines. This can be achieved by running ptolsplines.m. After all the data is splined it is

ready for input into the Ptolemy simulation. Statistic may also be generated by executing destat.m. The

simulation is started by running the MATLAB script ptosim.m. Once the Ptolemy simulation has

completed dead reckoning and neural network training/validation are completed by running the

MATLAB scripts dead.m and nb.m. If the users wishes to combine execution of these MATLAB

scripts they are called successively in the MATLAB script Go.m. Due to the data retrieval phase being

located in Windows and the rest of the simulation in Linux the process may not be fully batched at this

time. This problem is currently being resolved. The results of the simulation will be plots of the neural

network output and a plot showing accuracy as a function of window size. More advanced forms of

output such as surface plots and other methods of visualizing highly dimensional data are being explored.

Currently the user must manually change the a variable called offset to specify which of the input files they want to be training and simulation. An offset value of three would indicate that the first twelve files are to be used for training and the last three for validation, and offset of two would specify thirteen training sets and two validation sets. The user must also manually change nb.m if they wish to specify the variation on window size, and must make sure that the vector class has the same number of elements as the number of input files. The parameters for the Ptolemy models must currently be modified using the Ptolemy GUI. Any modifications to lower level parts will propagate to the rest of those gates through the simulation environment.