2014-04-01

Biomechanical Applications and Modeling of Quantum Nano-Composite Strain Gauges

Taylor David Remington
Brigham Young University - Provo

Biomechanical Applictions and Modeling of Quantum Nano-Composite Strain Gauges

Taylor D. Remington

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

David T. Fullwood, Chair
Anton E. Bowden
Brian D. Jensen

Department of Mechanical Engineering

Brigham Young University

March 2014

ABSTRACT

Biomechanical Applictions and Modeling of Quantum
Nano-Composite Strain Gauges

Taylor D. Remington
Department of Mechanical Engineering, BYU
Master of Science

Biological tissues routinely experience large strains and undergo large deformations during normal physiologic activity. Biological tissue deformation is well beyond the range of standard strain gauges, and hence must often be captured using expensive and non-portable options such as optical marker tracking methods that may rely upon significant post-processing. This study develops portable gauges that operate in real time and are compatible with the large strains seen by biological materials.

The new gauges are based on a relatively new technique for quantifying large strain in real-time (up to 40 %) by use of a piezoresistive nano-composite strain gauge. The nano-composite strain gauges (NCSGs) are manufactured by suspending nickel nanostrands within a biocompatible silicone matrix. The conductive nickel filaments come into progressively stronger electrical contact with each other as the NCSG is strained, thus reducing the electrical resistance that is then measured using a four-probe method.

This thesis summarizes progress in the understanding, design and application of NCSGs for biomechanical applications. The advanced understanding arises from a nano-junction-level finite element analysis of gap evolution that models how the geometry varies with strain in the critical regions between nickel particles. Future work will incorporate this new analysis into global models of the overall piezoresistive phenomenon.

The improvements in design focused on the manufacturing route to obtain a reliable thin and flexible gauge, along with a modified connection and data extraction system to reduce drift issues that were present in all previous tests. Furthermore, a pottable data logging system was developed for mobile applications. Finally, a method of analyzing the resultant data was formulated, based upon cross-correlation techniques, in order to distinguish between characteristic wave-forms for distinct physical activities. All of these improvements were successfully demonstrated via a gait-tracking system applied to the insole of standard running shoes.

Keywords: nanocomposite, high deflection, strain gauge, microcontroller

ACKNOWLEDGMENTS

Throughout my research my dear and loving wife has been a great help and blessing. Dr. Fullwood and Dr. Bowden have both been indispensable as mentors and guides to me while working on and discovering the gait sensing insole application. I would also be remiss if I didn't acknowledge all the wonderful work of my peers; Jake Merrell, Daniel McArthur, Adam Biladeau, and Dean Stolworthy. They have been helpful, kind, and have given me great advice. This material is based in part upon work supported by the National Science Foundation under Grant Award Number CMMI-1235365. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of the National Science Foundation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. SENSOR DEVELOPMENT

1.1 Motivation

In our world today, technology has become a way of life. All around are various electronic gadgets designed to simplify every task in the hope of making life easier and more enjoyable. While these advancements seem to permeate almost every aspect of life, there are a few areas that have been seemingly left behind. Interestingly enough, one such area is exercising. It is intriguing that this area has been so neglected while in the US alone over 50 million hours are spent exercising every day [1, 2].

Delving into this technological void, the following research develops nano-composite strain gauges (NCSGs) that can be taken out of the laboratory and into the real world. These sensors will be able to give valuable insight into the development of better training and exercising equipment, and making exercise safer, more effective and fun. The strategy in this project was to develop improved and portable gauges for specific application that would serve as a test-bed for all other developments. The chosen focus relates to gait analysis in people via in-sole embedded gauges.

The developed NCSGs have great potential as effective high deflection strain sensors. Reliable portable high deflection strain sensors do not currently exist. Previous renditions of the strain gauges exhibited numerous problems that have been solved with time and effort as part of the project reported in this manuscript. For instance, past manufacturing processes produced gauges that were thick, stiff, and gave inconsistent readings. Also, the gauges had only been developed using a non-portable method that included an Instron pull-testing machine and a desktop computer. Past gauges also exhibited an undesirable drifting effect in the recorded resistance.

Achievement of the project's purpose to create a reliable, portable high deflection strain sensor required improvements in several aspects of the strain gauge. First, an improved manufacturing process was necessary to decrease the stiffness and size of the gauges. Preferably, the gauge material would be spray-able which would produce a sensor with the minimum possible stiffness

and volume. Design improvements to the gauge and accompanying data extraction system were applied to reduce the observed drift. The portability of the gauges needed to be improved such that the gauges and their recording system were attachable to a human subject and he/she would be minimally affected by the system. Lastly, a proof of concept of the portability needed to be conducted. This proof was to be accomplished by developing a gait-sensing insole system.

1.2 Background on Current Wide-Range Strain Gauges

For decades, strain gauges have been used extensively to help understand the properties and performance of materials. Strain is a measure of the amount of deformation of an object. Perhaps the most important and widely used gauges exploit changes in electrical-resistance during deformation. Electrical-resistance gauges typically supply an electrical current through a metal sensor and measure the change in voltage as the gauge is strained. This type of strain gauge is used for at least 80% of industrial experimental stress analyses [4]. The metal-foil strain gauge is a common electrical-resistance strain gauge. It is attached to two points on a surface, and as the surface is strained, wires in the strain gauge increase in length while decreasing in cross-sectional area. The changing dimensions of the gauge change the resistance in proportion to the wire's resistivity, which can be calibrated to a certain strain [5]. Metal-foil strain gauges are very useful and accurate in measuring strain within certain limits. The main limitation of these gauges is that they only measure small strains (0-5%).

1.2.1 Biomechanical Strain Sensors

Though widely used, metal-foil strain gauges are generally not suitable for biomechanical strain applications because elastic biological materials strain 10 - 40 %. This amount of strain is well beyond the 2-5 % strain that metal-foil gauges are able to measure. Deeper understanding of many biological systems is most likely to be achieved through in-situ measurements, yet viable portable, large strain measurement systems do no currently exist. Non-portable, lab-based large strain sensors, however, are available.

Liquid Mercury Strain Gauge

One electrical-resistance strain sensor is the liquid mercury strain gauge. This gauge was introduced as a device to measure strain on soft tissues and is widely used due to its advantage of being small and implantable. The liquid mercury gauge has a column of liquid mercury that acts as one resistor of a Wheatstone bridge circuit. As the column is stretched and thinned, the resistance changes. The change in resistance is calibrated per gauge to a definable strain. While effective, this technique has a number of limitations including: determining zero tissue strain, the fact that the gauge needs to be sown into the tissue being inspected, and the unknown movement of the tissue relative to the device through the sutures. Tests must also be performed in a laboratory and cannot be monitored in the field [6].

Extensometer

Extensometers are another strain sensing mechanism suited for biomechanical systems. A light, flexible ring attaches to a support, designed to ensure that the weight of the extensometer does not hinder measurement, and holds the cantilever arms. The cantilever arms rigidly attach to the specimen and deflect different amounts at the two attachment points as the tissue is strained. The difference in the displacement between the two arms is the strain between the two points [7].One major draw-back of this method is that tissues to be measured must be extracted from the organism. The extensometer also fails to be useful outside of the laboratory, due to its need for external support.

Digital Image Correlation

Optical systems can be designed to track and measure strain on the human body and on extracted soft tissues. The most widely used optical method employs digital image correlation, or video dimensional analysis. This method is based on the technique of analyzing video pictures for dimension and uses tracking and image registration techniques to measure changes in images [7,8]. In order to get three-dimensional motion of objects, two or more cameras must be used. While useful, this technology has several drawbacks: the camera equipment can be expensive, software to correlate the images and determine strain is often expensive or difficult to set up, and cameras

must be placed correctly so no marker drop-out occurs. Using cameras also limits the ability of researchers to leave the lab [9].

1.3 Nano-Composite Strain Gauge (NCSG) Composition and Manufacturing Process

1.3.1 Previous work on NCSGs

BYU researchers have been developing NCSGs over the past 5-6 years. Initial steps produced a nano-composite polymer material that displayed characteristic properties of a potential high deflection strain gauge. Primitive NCSGs were born from inserting Nickel nano-strands (NiNs) into a silicone base and observing a negative piezoresivity when strained [10]. Optimization of rudimentary NCSGs was performed by Oliver Johnson, including adding chopped Nickel coated Carbon fiber (NCCF) to the gauges [11]. After the first optimization of the strain gauges, further understanding was developed through a model of their piezoresivity, based on percolation modeling and the quantum tunneling effect [12]. This research focused on finding a Quantum Barrier Height of the silicone base material as a necessary parameter in the quantum model. The gauges' development up to this point was verified using bovine soft tissue under prescribed loading conditions while verifying with optical methods [13].

Thomas Calkins was the first researcher to put an emphasis on developing an application to use the strain gauges. He further optimized some aspects of the strain gauge, such as what weight percents of NiNs and NCCF gave the most consistent results. He then developed a human/machine interface that consisted of a glove that could distinguish between the first five letters of the American Sign Language alphabet [14]. Michael Koecher then retested the Quantum Barrier Height for several different types of base materials, including silicone, and attempted to find an average junction gap for the samples at 0% strain [15]. This previous research has begun to open the door for almost limitless applications of the NCSGs.

1.3.2 Composition of Nano-Composite Strain Gauges

The nano-composite strain gauges (NCSGs) were comprised of three primary ingredients: silicone, Nickel nano-strands (NiNs), and Nickel coated Carbon fiber (NCCF). Silicone was used

as the base material because it had the desirable properties of a flexible elastomer, and the silicone could withstand strains up to 60% before plastic deformation or failure occured. Silicones used were commonly two part polymers that cure at room temperature, which allowed a simplified manufacturing process. Also, the silicone material had a high electrical resistance, meaning that current couldn't readily pass through. This ensured that without embedded conductive fillers, the silicone base material had practically an infinite resistance, as measured by a voltmeter.

Silicone

The original silicone material used was Sylgard 184, produced by Dow Chemical headquartered in Midland, Michigan. Sylgard was chosen because it is common and well-known. Using this material allowed the research to progress more quickly since quite a lot of data concerning the material could be looked up instead of having to be tested. After pre-straining methods (explained below) were applied to the Sylgard material, it could strain up to 60% without major plastic deformation, allowing gauges from this material to measure the natural strains found in most biomechanical situations.

Fig. 1.1 depicts the stress vs. strain curve of the Sylgard material strained from 0 to 40%. The red line depicts the tensile stress-strain curve. As evidenced by the straightness of the line, the material is linearly elastic, a property which increased the simplicity of calibration and reduced the difficulty of modeling. Also, there is little to no hysteresis during the loading and unloading conditions.

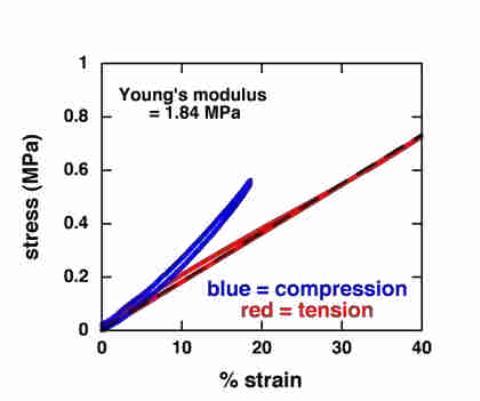

Figure 1.1: Stress/strain graph of Sylgard in compression (blue) and tension (red).

A second base material, QM 235, manufactured by Quantum Silicones LLC., located in Chesterfield,VA was also used. QM has elastic properties that allowed it to strain beyond the limits of Sylgard silicone before any fillers are added (˜80%). The QM material is also more flexible than the Sylgard material, requiring less force to strain. A dilutant, specifically manufactured for QM by Quantum Silicones LLC., further decreases the stiffness of the QM material. Less information was known about the QM material, but due to the possible benefits, further testing included the QM material to determine if increases in the desirable traits of the NCSGs were possible.

Nickel Nano-Strands (NiNs)

Another material in the NCSGs is Nickel nano-strands (NiNs). NiNs were added to the silicone base to promote conductivity in the NCSG. NiNs are formed through a proprietary chemical vapor deposition that produced a cake-like mixture of Nickel material (Fig. 1.2). This cake-like mixture was blended to separate strands of Nickel material. The deposition process produces NiNs with a unique structure; because they have a nanometer scale diameter, a high aspect ratio, and are highly branched [16]. Nickel was chosen as the metal to produce the nano-strands because it is conductive, magnetic, and corrosion resistant [16].

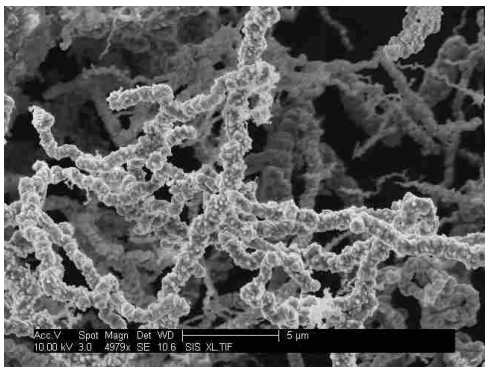

Figure 1.2: NiNs cake formed by proprietary chemical vapor deposition.

The Nickel nano-strands can be placed in paints, plastics, resins, adhesives, prepegs, silicones, etc. to induce electrical conductivity in these substances. In the current application, the NiNs' nanometer scale diameter facilitated conductivity in the silicone material without significantly changing the properties of the material, ensuring that the finished NCSG was still degrada-

tion resistant and flexible. Produced NiNs had high aspect ratios, ranging from 50:1 to 500:1 [17], which allowed a substantially low volume percent (about 7%) to make the strain gauge conductive.

Nickel Coated Carbon Fiber (NCCF)

The third ingredient added to the NCSG to increase stability and consistent resistance readings was Nickel coated Carbon Fiber (NCCF), as seen by magnification through a scanning electron microscope (SEM) in Fig. 1.3. The NCCF is also formed by Chemical Vapor Deposition. The Chemical Vapor Deposition provided unique advantages to the NCCF: Nickel coatings provide excellent coating thickness control, ductility, and uniformity [18]. The NCCF was added to support the current flow within the gauge and stabilize the resistance curve. NCCF used were long (0.5mm or 1mm) and straight, connecting long distances between NiNs. It was believed that the NCCF caused a monotonic response in the resistance readings of the NCSG, ensuring that the NCSGs read distinct values for different strains. Also, the NCCF is substantially less expensive to produce than the NiNs and therefore used to help reduce the overall cost of the gauge. To ensure that the NCCF had similar electrical properties to the NiNs, the NCCF used was 35% Ni and 65% carbon, by weight.

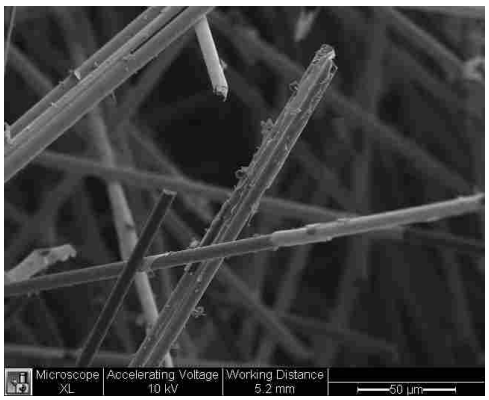

Figure 1.3: NCCF embedded in Sylgard 184. Nickel (white) covers the Carbon fibers (black strands).

1.3.3 Manufacturing Process

The current work builds upon manufacturing route developments implemented by previous BYU students, but with alterations made to overcome weaknesses in the previous methods. First, the thickness and stiffness of the gauges were too high to be integrated into many biomechanical systems and the variability from one NCSG to the next was unacceptable. Also, the previous process produced gauges that did not always cure properly. Lastly, the use of chemical thinner occasionally produced gauges that were incurable, wasting valuable time and resources. To fix these problems two key changes, as well as several smaller improvements, were applied to the process. The first key change was using a different silicone material, QM, chosen because of its ability to withstand higher strains than Sylgard. The second was forming the NCSGs in molds instead of pouring it onto a flat plate and letting it cure. Both of these changes, and other improvements, are explained in detail as they occur in the manufacturing process.

Through the trial and error of past students working at Brigham Young University, ranges of materials that produced working strain gauges were found. The amount of NiNs added was generally within 7% and 15% and NCCF generally was between 0 and 5%. The length of the NCCF strands was either 0.5 mm or 1 mm. The filter mesh for ensuring a specific distribution of NiNs strand sizes had options of 40, 60, and 80 gauge.

In the current work, the NCSGs were produced with 9% NiNs and 3% NCCF, determined by Calkins to give the best and most consistent readings [14]. The amount of material produced in cubic centimeters (CCs), was chosen on a per trial basis according to the amount and size of gauges needed. The type of silicone was chosen between Sylgard and QM. Matching gauges were made with both silicone bases and the results between the two were compared. Table 1.1 contains all the ingredients and mixing ratios to make NCSG.

Past students used Methyl-Ethyl-Ketone (MEK) as a thinning agent, which was undesirable since it needs to be used in conjunction with a fume hood. Working closely with an undergraduate assistant, Jake Merrell, changes were made in the type of thinner used.

The new thinner used was Octamethyltrisiloxane (OS-20), produced by The Dow Chemical Company, based out of Midlan, MI. OS-20 was chosen because it was produced specifically to be a thinner for Sylgard material and because of having a low evaporation temperature. The thinner reduces the viscosity of the composite material so that the material could be stirred together and

Table 1.1: Input percentages and weights of a NCSG with 9% NiNs, 3% NCCF volume percent and QM base.

Trial 1	Date	1/23/2013
Sample Volume	8	cc
Nanostrands	9% **6.42**	by Volume grams
Carbon Fiber	3% **0.63**	by Volume grams
Length of strands	1	mm
Filters	40	Gauge dry
Mixing time	30	sec
(Max with NiNs 1 min)		
Thinner (mark with x)	OS-20 x	
Thinner Density	0.82 20% **1.5**	 by Volume grams
Dilutant (only QM) (5% by weight max)	5% **0.45**	by Weight grams
Silicone Type (mark with x) Silicone Density	Sylgard 1.29	QM x g/cc
Total Silicone	9.08	grams
Base	**8.26**	grams
Catalyst	**0.83**	grams
Total Sample	16.58	grams
Notes:		

create a pourable and/or spreadable substance. Without the thinner the uncured composite material would end up too dry to maneuver. The maximum volume percent of OS-20 that could be added was 20% due to matrix deconstruction when using higher amounts. When using the QM material, 5% dilutant was usually added as this was the maximum amount before bonds in the material would begin to break. The maximum dilutant was normally chosen because it resulted in higher flexibility of the gauge.

Before mixing the constituent materials, the mesh was placed over a plastic cup (Fig. 1.4), so that the NiNs would fall into the cup when screened. The NiNs were screened through the chosen mesh by placing the NiNs on the screen and using a glass rod to spread and rub the NiNs. Screening the NiNs reduced the maximum size of the NiNs to less than the size of the chosen mesh, which ensured that large clumps of NiNs were broken up so that they distributed more evenly throughout the cured NCSG.

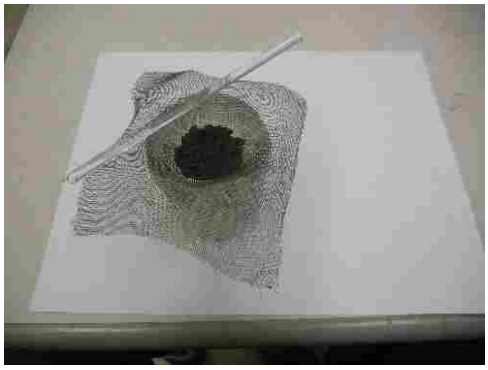

Figure 1.4: The screen mesh with a glass rod and cup to hold the screened NiNs

A scale measuring to the hundredth decimal place (in grams) was used to measure the constituent materials. The NiNs were poured into the cup until the desired weight was reached (Fig. 1.5). The NCCF was precut to the desired 0.5mm or 1mm length, such that it was mix-ready as it came out of the bag. The NCCF was scooped out of the bag and measured into a separate cup on the scale until it too reached the desired weight.

Figure 1.5: 500g scale with an Aluminum cup and screened NiNs.

Next, the silicone base was measured into a plastic cup using a dropper that was labeled either "Sylgard" or "QM," depending on the base chosen for the trial. The correct catalyst for the chosen base was then measured into the cup along with the silicone base. If the QM brand silicone was being used, the dilutant was measured into the cup. If the Sylgard was used, this step would be skipped. The thinner was then measured into the cup. The previously weighed NCCF was also added to the plastic cup at this time. This compilation of ingredients created a composite material with an uneven distribution of solid and liquid particles.

An even distribution of NCCF within the silicone before the NiNs were added was extremely important in making a viable gauge. To ensure that the uncured composite material was well mixed, the contents of the cup were mixed both by hand as well as by machine. The uncured composition was first mixed by hand with a glass rod and then mixed again using a THINKY ARM 310, THINKY Corporation, Tokyo. The THINKY would rotate at 2000 rpm, mixing the uncured composite material, while at the same time putting shear forces on the contents of the cup. To mix the materials sufficiently, hand stirring and mixing in the THINKY were alternated until the composite material had an even distribution and a silvery sheen.

The NiNs were then added to the cup and the glass rod was used to preliminarily mix the uncured composite (Fig. 1.6). The shear forces induced from the THINKY could damage the NiNs if placed in the THINKY for too long (upwards of 2 minutes). This was why all the other constituents, including the NCCF, were first mixed in the cup. To ensure that the NiNs were both mixed properly and protected from damage, the cup was placed in the THINKY and mixed for only 30 seconds. The resulting uncured composite was visually inspected, and if any clumps remained or if there were any solid particles that were not fully wetted, the glass rod was used to break up the clumps and/or wet the particles. The cup was then returned to the THINKY and mixed for another 30 seconds. The cycling between hand-mixing and machine-mixing continued until the entirety of the particles were wetted and the uncured composite was uniformly fluid and adequately viscous to be uniformly spread.

The baseline production method used at the start of the project poured the composite material onto plastic sheets. The plastic sheets were then placed in a vacuum chamber where a 10 cubic feet per minute (CFM) vacuum pump, as shown in Fig. 1.7, created a vacuum (500 mmHG) in the chamber. The vacuum was switched to "on" for 2 minutes to draw up air cavities in the silicone.

Figure 1.6: A plastic cup holding Silicone, NiNs, and NCCF that are to be placed in the Thinky.

The vacuum was then switched to "off" for two minutes and the uncured composite returned to atmospheric pressure. This allowed the uncured composite to fill any cavities formed by air bubbles leaving the composite when the vacuum was running. This process was repeated twice before the plastic sheets were removed from the chamber. The plastic sheets were then set out until the composite material was completely cured.

Figure 1.7: The vacuum and vacuum chamber configured to vacuum extra air out of the composite material

Pouring the gauges produced NCSGs with two major problems. First, the gauges were too thick and stiff to be used in all the desired applications. Second, the gauges took from 2-5 days to cure and sometimes did not ever completely cure. Therefore, the pouring method was improved to produce thinner, more consistent gauges that cured more quickly and fully.

1.3.4 Mold vs. Scraping Method

In an effort to arrive at a thin gauge of consistent thickness, the manufacturing process was modified from pouring the composite material to spraying the material onto the plastic sheets. Spraying the composite material was chosen because the spraying method allowed for much thinner gauges, reducing their stiffness and cure-time. Although it seemed to be an ideal method, spraying the NCSGs did not work due to the large amounts of solid material mixed in with the base material. Although unsuccessful, the entire process used can be found in Appendix A, due to its possible resurgence in future work.

After it was determined that spraying the material was not a viable option to produce gauges, two new methods were tested. These methods were: using a mold to control the size of the gauges, and scraping the gauges to a desired thickness after pouring the composite material.

After the gauges were poured out onto a plastic sheet a hard plastic scraper with a straight edge was used to spread out the composite material around the sheet, decreasing the overall thickness of the gauge. After reaching a minimum possible thickness the composite material was placed in the vacuum chamber and vacuumed using the same method as with the poured gauges.

The method that used the molds was concurrently developed with, and compared to, the scraping method. The mold was designed to keep the gauges at a constant thickness. The first mold created was made of Aluminum with a male and female portion that, when placed together, formed a void of 3.175mm (1/8") (Fig. 1.8). The mold was sprayed with LPS Teflon Mold Release spray to help the cured NCSGs release from the mold after the cure time. The uncured composite material was then poured or scooped into the desired female portion of the mold and spread around to approximately the correct thickness of the finished gauge.

The mold was then placed in the vacuum chamber and vacuumed several times to release air bubbles as done in all other methods (Fig. 1.7). The male portion of the mold was then placed on the female portion of the mold. After 2-5 days the male portion of the mold was removed and the gauges were extracted from the female portion of the mold.

The two methods were compared together visually at first (Fig. 1.9). As expected, the gauges produced by the method using the mold looked much more appealing than those that were scraped. The strain gauge manufactured by scraping the composite material around the plastic sheet was also not evenly thick, it had many holes from trying to get the gauge sufficiently thin,

Figure 1.8: The Aluminum mold with the female portion of the mold on the left and the male portion on the right.

and was not reliably conductive. The strain gauge manufactured using the mold was by no means perfect, but overall it was better than the gauge made by the scraping method. Therefore, it was decided that the process using the molds provided the best method to produce thin gauges, and the process was studied and improved.

Figure 1.9: The gauge formed by the mold process (left) and the scraped method (right).

The first change made to the molding method was producing two new sizes of molds. These molds formed 0.8mm (1/32”) and 0.4mm (1/64”) thick gauges. Using these two new molds provided an avenue to decrease the stiffness of the gauges by decreasing the thickness. To further improve the gauges the thinner was studied since it was essential in the process but was not well understood.

Table 1.2: Heat transfer coefficient values used in determining length of time that gauge takes to reach 100 °C.

Silicone density (ρ)	18480 $\frac{kg}{m^2}$
Volume (V)	$3.693x10^{-7}m^3$
Degrees Celsius (C)	1100 $\frac{J}{kgK}$
Convective heat transfer coefficient (h)	5 $\frac{W}{mK}$
Area of width and length of silicone (As)	$9.82x10^{-4}m^2$
Silicone conduction (K)	7.84 $\frac{W}{m^2K}$
Area of depth and width of silicone (A)	$9.3704x10^{-4}m^2$
Final temperature difference (θ)	- 55.56 K
Initial temperature difference (θ_1)	-72.2 K

Although OS-20 was essential in creating a composite material fluidity that was spreadable into the mold shape, it was undesirable in the finished NCSG since it hindered the curing of the Silicone material. Older production methods did not remove the thinner through evaporation, but it was determined that evaporation was important for consistent, useful gauges. The thinner evaporates at 34.4 $^{\circ}C$. To ensure that the entirety of the OS-20 was evaporated, the uncured composite was heated to 37.7 $^{\circ}C$ and then set out to cool. 37.7 $^{\circ}C$ was chosen because it was above the evaporation temperature and heated up quickly enough that the NCSGs did not cure in the oven. A basic heat transfer equation, Eq. 1.1, gave the amount of time that the mold needed to be in an oven at 93.3 $^{\circ}C$ before the uncured composite reached 37.7 $^{\circ}C$ [19].

$$T = \frac{\rho * V * C * log\frac{\theta_i}{\theta}}{h * A_s * l + kA} \tag{1.1}$$

The inputs into the equation are shown in Table 1.2.

Using the approximated values and solving Eq. 1.1 for temperature (T), the time the composite took in the oven to heat up to 37.7 $^{\circ}C$ was 13 minutes and 42 seconds. Therefore, the female portion of the mold and the uncured composite were placed in an oven at 93.3 $^{\circ}C$ for 14 minutes, allowing the composite material to heat up just above 37.7 $^{\circ}C$. The heat transfer equation was verified by measuring the uncured composite with a thermocouple as the mold was removed from the oven.

The mold was then placed in the vacuum chamber and the vacuum was switched “on” and “off” as described for the pouring method. Once the vacuum was finished, the female portion of

the mold was removed from the vacuum chamber and the male portion of the mold was placed on top of the female portion. Another improvement was made by using C-clamps to clamp the mold until the gauges had the correct thickness. The filled mold would sit for at least two days while the composite material cured. Heating the material to 37.7 $^{\circ}C$ also helped in the curing process, producing gauges that were always completely cured after the two day waiting period. After the material was cured, the male portion of the mold was removed and the NCSG was extracted from the female portion. The cured nano-composite material yielded a specific resistance when a voltage was applied to it (10-1,000,000 Ω depending on the mix ratio and base material).

A new mold was then milled to decrease the NCSGs thickness even further (Fig. 1.10). The mold consisted of two flat plates with one plate having feet that rose 0.34 mm above the flat surface. Although most desired gauge shapes could be formed from the mold shapes above, it is possible that other gauge shapes could be needed, particularly gauges that were long. Because the new mold only had two flat plates without edge barriers, new NCSG shapes could be manufactured.

Figure 1.10: The flat plate mold has legs approximately 0.34 mm tall.

The uncured material was spread around one of the flat plates to a thickness that was slightly thicker than the desired thickness. The mold was then placed in the oven and the OS-20 was forced to evaporate before the mold was placed in the vacuum as before. Finally, the second flat plate was placed on the first flat plate, clamped, and allowed to cure for two days. The flat-plate mold has produced gauges that are very thin, ranging from 0.34 mm to 0.37 mm (Fig. 1.11). These gauges are currently the thinnest gauges ever produced by a Brigham Young University researcher.

Figure 1.11: The gauge created by the flat mold. It has an average thickness of 0.35 mm.

Foam Sensing System

Platinum Silicone foam was also studied as a possible replacement base material, taking the place of the flexible elastomer. A conductive foam gauge would potentially open many new applications, especially in the realm of compression. It was hypothesized that a foam embedded with NiNs and NCCF would be able to measure compressive strains and/or forces, while the elastomer was ideally suited to measuring tensile strains. Knowing the force on a certain component could be usefully applied to shoes, protective pads worn during sporting events, reactive seats, motorcycle helmets, etc. The process of developing the foam sensors from inception to a proof-of-concept is included in the Appendix B because it doesn't deal specifically with high deflection NCSGs, the topic of this Thesis.

1.4 Possible Factors Causing Undesirable Observed Drift

After reducing the gauge stiffness via the thin mold-produced method, the next step was to determine factors that contributed to an observed drift phenomenon by isolating potential drift-causing effects and eliminating them. A drift effect seen in the resistance of the gauges during cyclic strain cycles made correlating the change in resistance with strain difficult. The drift phenomenon has been observed commonly in resistance gauges but needed to be understood and minimized if creation of useful strain gauges was possible. There were several factors that potentially lead to excessive drift that could be optimized to minimize it. The factors tested were capacitance, temperature, and the contact points between the NCSG and data collector.

The first contributor to drift arose from the attachment method of the wires. The Instron previously attached to the gauge through wires and alligator clips. The alligator clips were possibly slipping and/or pulling and continually changing the contact resistance throughout the test. To adjust this possible drift causing effect, wires were embedded into the strain gauge, maximizing the chance of a constant electrical connection between wires carrying the signal to and from the gauge.

The next factor tested was the capacitance that builds up in the gauge during testing. The resistance measurements in the Instron were being measured by sending a direct current (DC) across the gauge. The DC constantly increased the capacitance of the gauge. To eliminate the resistance drift due to the increasing capacitance, an alternating current (AC) was used. An alternating current does just what the name implies, it switches the direction of the flow of electrons at the desired frequency. Switching the electron flow (current) discharges any capacitance that has built up in the strain gauge by constantly introducing current in the opposite direction as the capacitance charges. Therefore, by using an AC instead of a DC the capacitance effect was virtually eliminated.

The AC system was set up with a signal generator that created an AC voltage across the strain gauge. The signal was set at 1000 Hz with a 5 V peak-to-peak signal. The signal was sent to one end of the NCSG. The other end of the NCSG was attached to the DAQ system that Labview was using when running its programs. The DAQ system was connected to ground through a voltage divider. Since the resistance in the resistor and the input signal were known, the changing resistance in the strain gauge could be determined.

Lastly, the temperature drift of the gauges and the resulting change in resistance were recorded. While working with the gauges it was determined that some of the gauges were getting very hot while other gauges were not experiencing the same effects. From this observation it became clear that variable temperatures were possibly playing large roles in the gauges' drift. The gauges were tested at different temperatures to determine how much, if any, the change in temperature of the gauge changes the resistance in the gauge.

Changing the temperature of the gauges was done using a heat lamp. The heat lamp was placed on a stool next to the Instron pull-testing machine. A thermocouple was attached to the back of the strain gauge to determine the temperature of the strain gauge as the intensity of the heat lamp increased. The DAQ data logging system was used to capture the temperature of the

gauge from the thermocouple and the root mean square (RMS) voltage across the gauge while the temperature was in fluctuation. The first resistances were taken at room temperature, the heat lamp was changed in intensity from zero to three, held until the heat of the gauge started to level off and then increased to six. Once the gauge started to level off again the heat lamp was returned to intensity zero and the gauge was allowed to cool. The process was repeated a second time to check consistency of results.

Temperature effects were important to understand since fluctuating temperatures could occur in many potential applications, including the gait-sensing insole described in this work. The gait-sensing insole fits into the shoe under the insole. This meant that in exercise situations, as well as on very warm days with or without copious amounts of exercise, the gauges potentially experienced rather large fluctuations in temperature. If the effects of temperature on the gauge were significant and understood, the measurements from the gauge could be adjusted to compensate for the temperature of the gauge to produce more accurate strain measurements.

1.5 Portability

1.5.1 Benefits of Portability

As mentioned earlier in this paper, one of the major drawbacks of current biological strain gauge systems is their lack of portability. The number of applications of any strain gauge could be multiplied if the strain gauge could be used outside of the laboratory setting. Although valuable data has been collected in the laboratory setting, such as research at Brigham Young University comparing the movement of llama spines to the human spine, portability of the gauge would make the difference between a gauge with several applications and one with almost limitless applications.

Many industries have used biomechanical strain sensing techniques and would greatly benefit from a portable strain sensing system. The medical industry could use strain gauges embedded into insoles to help determine the steadiness of someone's gait, athletes could be better protected by knowing how their body moves, the video gaming industry could create more life-like characters through strain tracking, the movie industry could create human movement in animated films and CGI dimensions of movies incorporated with real-life actors, etc. Though many industries currently benefit from understanding human motion, bodily strains during these motions is relatively

unknown. Increased understanding of these strains could be potentially powerful in improving the afore-mentioned fields.

Making a portable system could have sweeping benefits. For instance, when testing in a laboratory setting, care and precaution should be taken to ensure that the subject is not harmed [20]. While it is very important that human research subjects have that right, it has stifled the ability of researchers to ethically test the maximum physiological strains that the body undergoes during many activities without portable technologies [21–23]. A portable strain gauge, however, allows data to be gathered when people really push their bodies to the limit without crossing, or being tempted to cross, ethical lines. These gauges could be attached to an athlete, actor, etc. and strain could be tracked as they perform in their natural environment. For instance, instead of measuring a basketball player's gait as they are wearing bulky equipment or attached to a computer, the portable strain sensing system could measure the natural gait on the basketball player as they run, cut, and shoot. This opportunity would allow researchers to better understand gait in real situations and realistic strains that the body undergoes during naturally performed motions.

The portable strain sensing system could also be set up to increase the safety of performing athletes. The strain sensing system could be equipped with indicators so that an athlete knows when physiological limits are being pushed. This would allow an athlete to know if any actions they perform are putting their bodies at risk. They could then adjust their movements in a manner that is safer for their bodies and/or new rules can be put into place to make such movements illegal. This could possibly reduce the risk of injury to the athletes and their competitors.

Increasing the NCSGs' portability was beneficial and important because it allowed users of the strain sensing system to interact in normal domains. Interactions occurring outside of the laboratory setting would be more natural and, if tracked correctly, provide more realistic data.

1.5.2 Arduino Uno Microcontroller

A major focus of this research was developing the portability of the strain sensing system. Previous research exclusively tested the strain gauges using Labview to decipher the change in resistance. This limited the strain gauge to the laboratory setting. To test whether the strain gauge was measurable by a portable system, a microcontroller was inserted into the system. The microcontroller chosen was the Arduino Uno because it was built in an easy to use coding environment

and designed to be an easy interface with many different types of hardware [24]. To develop a viable portable system it was vital that the microcontroller chosen was simple to program and use, powerful enough for the application, and small enough to be easily worn on the body.

1.5.3 Coding the Microcontroller

In order to code the microcontroller, a USB cord attached the microcontroller to the computer. The controller was programmed with an open source software downloaded from arduino.cc, called "Processing" and programmed in Arduino code (Appendix C). The code was written to the 32 KB of SRAM available on the microcontroller. The controller had 2 KB of EEPROM (flash memory) built-in. Due to this being insufficient, a microSD shield was attached to the top of the microcontroller printed circuit board (PCB). The controller was set up to capture data at 100Hz. Without the SD card the flash memory held less than 0.1 hours of data. The SD shield was capable of holding a 4gb microSD card that recorded the data output, with the capacity to hold approximately 1500 hours of data. The microcontroller sampled at 100Hz to ensure that all motions were captured, where the rate of capture should be at least 10 times the frequency of any movements and the human body moves at maximum speeds of 5Hz [25].

1.5.4 Microcontroller Setup

The maximum length and width of the PCB were 68.5 mm (2.7 inches) and 53.4 mm (2.1 inches) respectively, with the USB connector and power jack extending beyond the former dimension [24]. The height of the PCB along with the SD shield attached was 23.9mm. The microcontroller could operate at a minimum of 6v but required at least 7v to ensure a constant output of 5v from the output pin. The controller was powered by a 9v battery, exceeding the 7v threshold needed by the microcontroller to function, attached to the power jack on the PCB (Fig. 1.12. An armband configured to hold electronic devices while exercising was used to secure the microcontroller to the test subject.

The strain sensing system was set up as a voltage divider circuit to determine the change in resistance in the strain gauge. The voltage divider used was a simple circuit consisting of two resistors (Fig. 1.13) that had the useful property of changing a higher voltage (Vin) into a lower

Figure 1.12: The Arduino Uno microcontroller with a 9 V battery.

one (Vout) in proportion to the difference in resistance of the two gauges. It did this by dividing the input voltage by a ratio determined by the values of the two resistors (R1 and R2) [26]. The equation used to get the voltage out for the voltage divider circuit was [27]:

$$VOUT = \frac{R2}{R2+R1}VIN \tag{1.2}$$

The classic voltage divider equation, Eq. 1.2, was rearranged to find the resistance in the NCSG (R1) and solved for R1:

$$R1 = \frac{VIN}{VOUT}R2 - R2 \tag{1.3}$$

All the variables on the right side of Eq. 1.3 were either known or found during testing, which allowed for computation of R_1.

The microcontroller sent a 5v signal (Vin) through an attached wire to one end of the strain gauge (R1). Another wire from the analog pin (Vout) was attached to the opposite end of the NCSG (R1). In series electrical connection with this wire was a 500Ω resistor (R2) placed on the mini breadboard attached to the SD shield. This meant that the output voltage (Vout) would read between 0 and 5V with the analog pin reading 0V when the resistance of the gauge was much higher than the resistor (R2) and 5V when the resistance in the resistor (R2) was much lower than

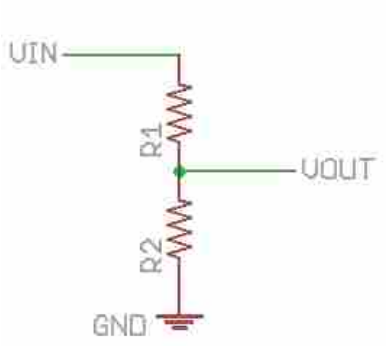

Figure 1.13: Drawing of a simple voltage divider [26].

the NCSG (R1). Ground (GND) was in series electrical connection with the resistor, creating a voltage divider with the analog input pin as the voltage reader.

In one iteration of the design a 10 turn $100k\Omega$ potentiometer replaced the resistor (R2). The 10 turn potentiometer used was a resistance-varying device that changed resistance from 0Ω to $100k\Omega$ over the course of 10 turns of the adjustment knob. To ensure the maximum sensitivity in the gauge, adjustment of the potentiometer for different strain gauges was required. The Processing program was adjusted to run a live feed to the computer screen, outputting the resulting voltage from the voltage divider when needed. The potentiometer was adjusted until the output voltage was near 2.5V, where the gauge was the most sensitive to changes in resistance. However, when NCSGs with similar resistances were tested it was easier to use a resistor close to the resistance of the gauges rather than a potentiometer.

1.6 Institutional Review Board (IRB) Acceptance

Human research testing is extremely important to improving our quality of life. Human research testing is any research or clinical investigation that involves human subjects. Testing must take place to ensure the viability and safety of any product before it is produced and marketed to the general public [20, 28]. An institutional review board is a committee that is formally designated to approve, monitor, and review biomedical research involving humans [28]. Before testing could begin on human subjects an Institutional Review Board (IRB) approval was applied for from Brigham Young University. The IRB passed the insole sensing system in Approval X130135, which was valid from April 2, 2013 through April 1, 2014. Testing began after April 24, 2013 and

ended on June 28, 2013, within the approval range of the IRB. Gait measurements were taken by attaching the strain gauge to the bottom of the insole of a shoe while the subjects walked, jogged, ran, did jumping jacks, carried heavy equipment, stretched, etc.

1.6.1 Gait Testing

Gait testing was chosen as a suitable test bed for the portable sensor system because it is an extremely powerful tool to measure a person's functionality. Of all the clinical measures that are available to predict mortality and a person's likelihood of being institutionalized, gait disorders are one of the best [29]. As gait is such a powerful diagnostic tool, it is extremely important to characterize it correctly so that patients could get the help that they need as quickly as possible before injury, deconditioning, or mobility impairment occurs. Gait and balance disorders in the elderly are difficult to recognize and diagnose in the family practice setting, as doctors are not trained or equipped to measure the gait of their patients. This lack of available systems opened the doorway to development of gait recognizing devices [30].

The strain sensing system using the NCSGs helped recognize gait. Several iterations of the strain sensing system were developed. The first setup used is seen in Fig. 1.14. In this setup the microcontroller and 9V battery were placed in an mp3 player case, designed to be worn while exercising. Wires exited the case and attached to each end of the NCSG. The NCSG was taped onto the insole under where the ball of the foot naturally rests. During testing the microcontroller was strapped around the lower calf of the person wearing the insole Fig. 1.15.

Figure 1.14: The insole with a NCSG duct taped to the bottom to gather data.

Figure 1.15: The microcontroller is attached to the ankle of the subject with wires traveling under the insole.

The first round of insole testing used the single strain gauge attached to the bottom of the insole. A single walking test followed the following format. First, the subject put the instrumented shoe on his foot and the matching shoe without the sensor on the left foot. Next, the battery was plugged into the microcontroller to turn it on, which was evidenced by the illumination of a red light on the microSD memory shield (Fig. 1.15). When the subject was ready to start the test the reset button on the microSD card was engaged. Successful resetting of the microcontroller resulted in a green LED light blinking rapidly for four seconds as the microcontroller searched for the microSD card. If the microctoller found the microSD, the LED turned to "on" and the test started. The subject then walked normally for one minute. An aide with a stopwatch told the subject when one minute had expired. The subject then checked to see if the LED on the microcontroller was blinking slowly, indicating that the test was complete.

The subject had an aide that timed the test to ensure that the subject walked for the full minute before stopping to rest, checked to see if the recording light had gone off when the test was over, and overall ensured that the entire test was performed according to instructions. After the test was completed the microcontroller and the shoes were removed from the subject's ankles and feet. The microSD card was taken from the microSD shield and inserted into a computer where the data was read into Matlab so it could be analyzed and visualized.

While the one gauge insole was useful in several applications the strain sensing system would be much better at determining gait if there were more gauges attached to the bottom of the insole. The Arduino Uno microcontroller had 6 analog pins printed on the PCB. This meant that theoretically one microcontroller could measure up to 6 gauges' resistance at the same time. Some other microcontrollers only have four analog pins, and to make the system more feasible across platforms, the Arduino was setup to measure four gauges at the same time (Fig. 1.16).

Figure 1.16: The insole system with four gauges embedded into the silicone base. The microcontroller is attached to the embedded gauges.

The four gauges were placed on the bottom of the insole in areas that normally contact ground when walking. One gauge was placed under the ball of the foot along the sagittal axis to pick up the strain as the subject moves. The other sensors were placed under the pad of the foot along the coronal axis, under the arch of the foot along the sagittal axis, and beneath the heel along the sagittal axis.

The gauges were cast into a silicone insole (Fig. 1.16) to ensure that the gauges would not be felt by the test subjects. An Aluminum mold was produced, where pouring the Sylgard into the mold, curing the Sylgard, and removing the Sylgard from the mold resulted in the shape shown in Fig. 1.16. A second insole of the same thickness but without the gauges was produced so that the subject would be more comfortable walking on insoles that match. These insoles were placed in the shoe after the original insole of the shoe was removed. The original insole was then replaced on top of the Silicone insoles to increase comfort to subjects.

The four gauge strain sensing system measured several factors that contributed to gait. The system determined how much the ball of the foot strained as the subject walked, how much a subject was using different portions of the foot, and the frequency of placing the wired foot

on the ground. Knowing these factors could help identify subjects with multiple sclerosis and Parkinson's disease, as well as other problems [31, 32]. Further programming of the sensor could yield a computer model that checks the gait data and determines whether the person has a normal gait or if the subject has gait problems, leading to more accurate diagnoses.

1.7 Results and Discussion

1.7.1 Testing the NCSGs Developed Through New Manufacturing Techniques

The new manufacturing process placed the gauges in molds where they were constrained on all sides, not allowing air or gas to escape during the curing process. To ensure that the formation of air bubbles and the lack of air during curing would not render the gauges non-functional, the new molded gauges were tested on the Instron.

Wires that attached to the ends of the NCSG (Fig. 1.17) were connected into a NI-9219 console. There were four wires attached to the NCSGs because the console had an internal four wire Wheatstone bridge [33]. The console output a known current (400mA) and determined the incoming voltage to determine the resistance in the measured item.

Figure 1.17: The NCSG is placed in the Instron in plastic non-conductive grippers.

The NCSG was mechanically clamped by clear plastic clamps configured to inhibit current traveling through the Instron. Embedded wires protruded from each end of the NCSG and were attached to the four lead wires from the console. The Instron performed the pull testing experiment by pulling the bottom clamp the preset distance and holding the top clamp constant. The gauge

was cycled 20 times at 20% strain to test how the strain gauge mechanically reacted to the strain and how the resistance changed as the gauge was pulled. The gauge had a Sylgard base, nine percent NiNs, and 3% NCCF by volume. The NCSG was approximately 0.6mm thick, 12.7mm wide and 25mm long. The distance between the gauges was 19.1mm long thus the gauge was strained 19.1*.2 or 3.82mm.

The manufactured gauges output a resistance when attached to the four wire method in the Instron. The reaction of the resistance is shown in Fig. 1.18 (Left graph). The resistance of the NCSG before testing started was approximately 700Ω, well within the range of normal NCSG resistance (between 10-10,000,000Ω) [14]. The newly manufactured gauges tested exhibited the piezoresistive effect seen in earlier trials of the thicker gauges. During the first several cycles the NCSG drastically rose in resistance as the gauge relaxed between tension portions of the cycle. This anomaly occurred the first two or three cycles before entering steady state fluctuations. After the few initial cycles the gauge steadily increased in resistance as the gauge was cyclically strained. Both the minimum and the maximum resistances increased during straining, with the maximum resistance increasing faster than the minimum resistance. The linear drift in the resistance was accounted for and readings were fit fairly well to the displacement records of the Instron.

Fig. 1.18 (Right) shows the displacement from the Instron measurement (blue line) and the scaled resistance calibrated to give the displacement (green line). The first 3 cycles of the test were discarded because they deviated from the rest of the data since the strain gauge was relaxing during those cycles. The fit started when the upper resistance peak was the lowest and began to steadily grow from one cycle to the next. The resistance was fit to the displacement in Matlab by finding the values of a, b, and c that best correlate resistance and displacement by minimizing the function: $0 = displacement – a * (resistance)^b – c^2$. Before cycling began, the NCSG was displaced to 10% strain, or 1.9mm. The initial strain was added to ensure that the NCSG was always in tension during the test. The average error calculated over the 20 cycle test was 8.55%, within the 10% mark measured by Tommy Hyatt using older production methods [13]. Since the gauges measured strain as well as, if not better than, the gauges produced by the older production methods the molding method was continued.

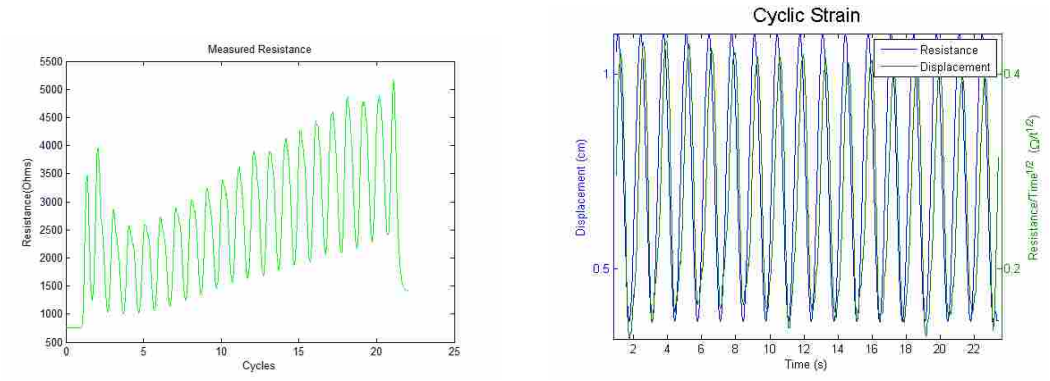

Figure 1.18: Resistance measured by the Instron and resistance fitted to the displacement.

1.7.2 Stiffness Tests

Once it was determined that the formed gauges acted like the previously non-formed gauges, further testing intended to determine whether the mold-formed gauges were less stiff than non-formed gauges. High stiffness was an undesirable characteristic of a strain gauge since a stiff gauge would interfere with the natural response of the object being measured.

The stiffness of the gauges was tested by placing them into the Instron and measuring the force needed to pull the gauges from 0 to 50% strain. The formed and non-formed gauges were cut to the relatively same width to ensure that the forces from the different tests were comparable. The formed gauges had an averaged thickness of $7.79x10^{-1} \pm 2.74x10^{-2} mm$ with a width of $5.96 \pm 3.54x10^{-1} mm$ resulting in a cross-sectional area of $4.65 \pm 3.00x10^{-1} mm^2$. The non-formed gauges had a thickness of $2.24 \pm 3.07x10^{-1} mm$, a width of $5.17 \pm 5.28x10^{-1} mm$, and a cross-sectional area $1.18x10^{1} \pm 6.45 mm^2$.

Figure 1.19 shows the disparity between the stiffness of the formed and non-formed gauges by showing the different forces required to pull the two gauges. The maximum pull forces for the four formed gauges (pulled from 0 to 50 % strain) were 0.681, 0.588, 0.701, 0.811N while the maximum pull forces for the non-formed gauges were 3.4319 and 5.0543 N. The forces per width for the thin gauges were on average 2.96N/mm while the forces per width for the thick gauges were 15.6 and 26.7N/mm. When compared to the average force needed to pull the formed NCSGs to 50% strain, the first non-formed gauge tested needed 5.30 and the second gauge needed 9.06 times as much force.

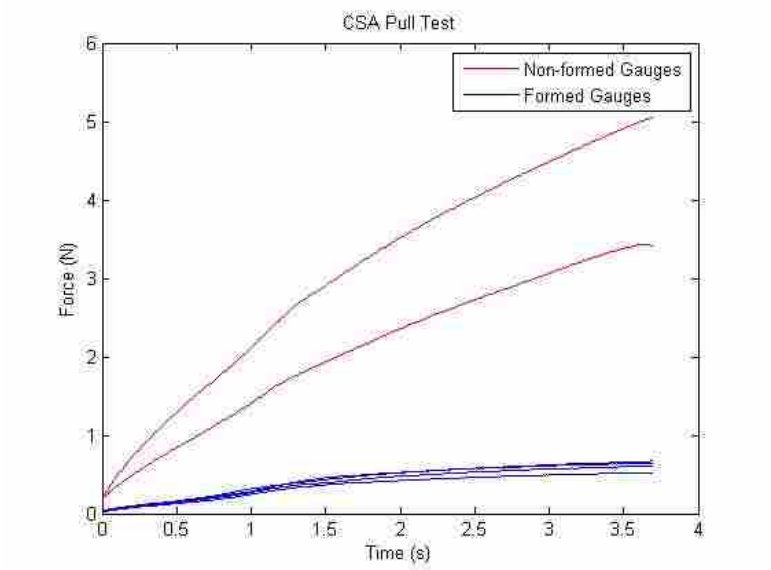

Figure 1.19: The red lines represent the force required to pull the NCSGs produced through old manufacturing techniques and the blue lines represent pulling force to failure of the NCSGs produced through the molding method.

The formed gauges required a much lower force to strain, meaning that the thinner gauges would have much lower impact on the mechanical system being measured. Also, the mold-formed gauges had much closer groupings in the amount of force needed to break the gauges. The disparity in the results for the older gauges demonstrated previous variability in manufacturing processes that has been greatly improved in the new gauges.

1.7.3 Drift Factors

Embedded Electrical Contacts

With the results from the manufacturing process showing that the formed gauges greatly decrease stiffness the next stage of testing began. The embedded electrical contacts were tested by setting up the NCSGs like as described in section 1.7.1, where the NI-9219 supplied a DC voltage to the gauge and measured the voltage drop over the gauge to determine the resistance. The gauge was cycled as the resistance was constantly being read. The results in Fig. 1.20 depict the resistance reading when the NCSG was cycled using embedded electrical contacts to connect

the strain gauge to the NI data acquisition system (DAQ). The gauge was cycled in the Instron from 0 to 20% strain for 500 cycles.

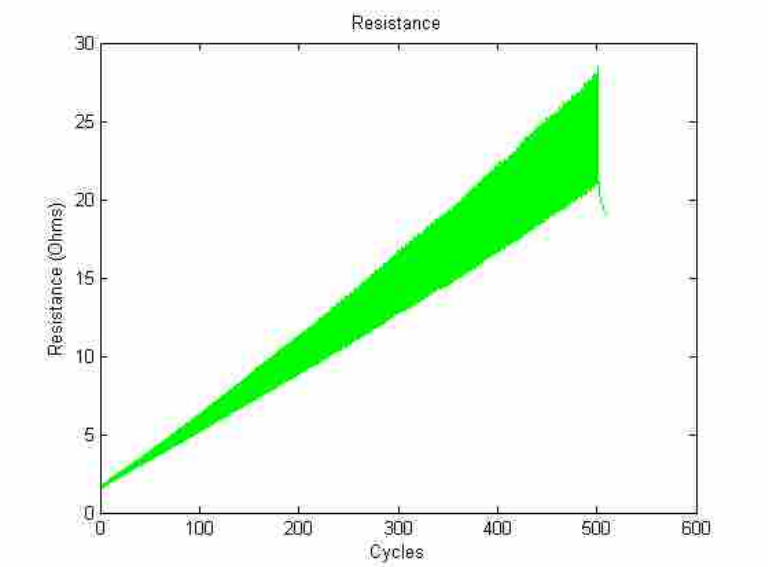

Figure 1.20: NCSG resistance response to 500 cycles of 20% strain.

The resistance in the gauge began at approximately 3Ω and changed approximately $1/2\Omega$ per cycle. The resistance continually increased in sensitivity with the average peak to peak value moving from $1/2\Omega$ to 8Ω per cycle. At the end of the 500 cycles the average resistance was 24Ω and each cycle was 8Ω per cycle. Though both the average and the amplitude increased, they did so fairly linearly. This result was not ideal, but correctable.

Embedding wires into the strain gauges linearized the drift, meaning that the drift substantially constantly and linearly increased. This was true for both the increasing sensitivity and the increasing mean resistance. The linearly increasing drift was much easier to program for than non-linearly increasing drift, simplifying post-processing strain determination from the resistance output.

AC Voltage Supply

Given the nature of the drift seen in the previous section, it was hypothesized that it was at least partially caused by charge build-up in the material due to the DC voltage being applied.

Hence an AC signal was applied, resulting in a significant decrease in drift, as record in Fig. 1.21 (in terms of the RMS voltage reading across the gauge).

Reading the change in resistance in the AC power supply method was not as easy as the DC power supply method. The AC power supply method required that the root mean square (RMS) voltage was determined to get a signal that was readable to a person viewing the data. Changing the raw data into the RMS was programmed into Labview and chosen as a post-processing option. When the resistance in the NCSGs decreased the RMS voltage signal increased and vice versa.

The RMS voltage signal drifted downward over the course of the 500 cycles (Fig. 1.21). This correlated to an increasing resistance in the gauge. The signal drift was much faster at the beginning than at the end of the signal. The amplitude of the signal also changed throughout the test but much less so than when using the DC power supply. The signal began cycling at about 0.6 Ω per cycle and by the end of the test the gauge cycled at 0.4Ω.

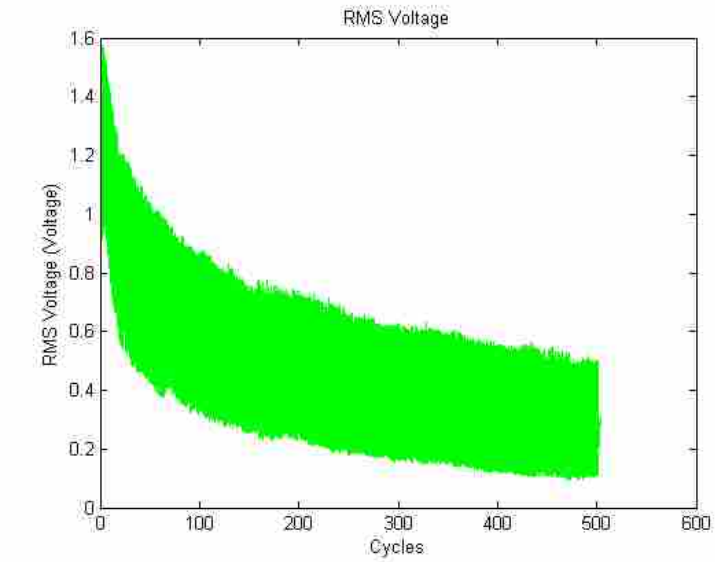

Figure 1.21: The AC response of the NCSG over 500 cycles straining at 20%.

The AC voltage decreased the amount of drift that the strain gauges exhibited. By the end of the test the drift seemed to be leveling out and possibly with more cycles the drift would have been eliminated for the test. Also, the first initial drop over the first 100s was much steeper than the rest of the drift. This same general pattern played out at tests conducted at 10, 20 and 40% strain.

Since using the AC wave eradicates the capacitance building up in the strain gauge over the course of the test the drift seen in this graph must correlate to other factors besides capacitance, such as: piezoelectric effects, the contacts sliding during testing, temperature, and stress relaxation of the Sylgard material.

Temperature

With the electrical contacts embedded and the alternating current supplied to the NCSG the reaction of the NCSG to temperature was tested. The gauge's temperature was fluctuated from 25.5 $°C$, to a maximum temperature of 44 $°C$ through the use of a heat lamp. At first the lamp was set to medium to heat the gauge, after allowing it to cool, the lamp was switched to high to heat up the gauge faster. After turning off the heat lamp again it took 3.77 minutes to cool down to 27 $°C$.

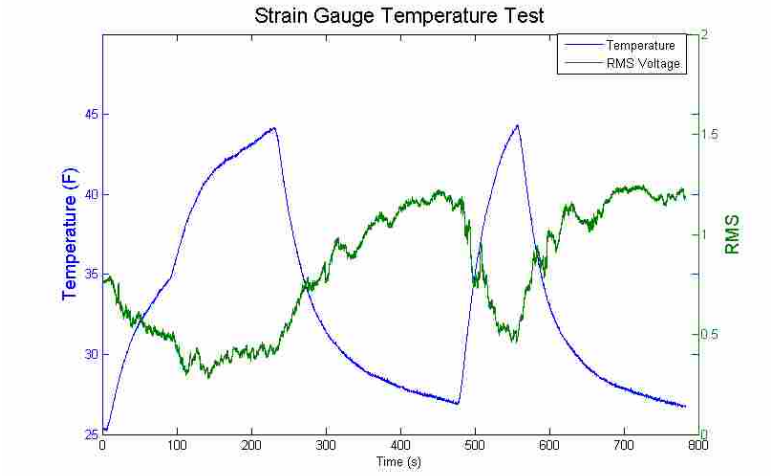

Figure 1.22: The blue line represents temperature and the green line voltage. The voltage responds to the changes in temperature.

The RMS voltage decreased as the temperature in the gauge increased. The decreasing RMS signal was a result of the strain gauges' increasing resistance. This meant that the resistance increased as the temperature of the gauges increased. The strain gauge started out at 0.77 RMS voltage and ranged from 0.28V to 1.2V. The large change in the resistance indicated that the resistance of the NCSG was susceptible to large temperature fluctuations.

1.8 Microcontroller

Following the move to an AC signal across the gauge, the microcontroller measured the voltage drop across the NCSG instead of the resistance measured by the Instron in earlier tests. To determine the strain in the system the voltage was calibrated to the strain to the level of the gauge. In the first tests, the battery powered microcontroller was attached to the NCSG with alligator clips (Fig. 1.23). It was important to know if the microcontroller could reliably measure the strain via the change in voltage in the strain sensing system. The Instron was equipped with an encoder accurate to within $\pm$ 0.5% strain which was sufficiently accurate to calibrate the strain in the NCSG [34]. When the test was complete the displacement results from the Instron and the voltage readings from the microcontroller were loaded into Matlab. Matlab was used to correlate the voltage readings and displacement results.

Figure 1.23: The microcontroller is attached to the Instron to measure NCSG response to displacements.

The NCSG was pre-strained 10% before the cycling began, or pulled to 10% longer than NCSG's length before testing began. The displacement from the Instron consistently oscillated between a maximum distance of 5.59mm (30% strain) from the unstrained position and a minimum distance of 1.91mm (10% strain) from the unstrained position as seen in Fig. 1.24 (Right). The gauge was cycled at 1Hz so the 20 cycle test ended after 20s. The data recorded from the microcontroller was fit to the displacement using the equation: $displacement = a * (resistance)^b + c^2$.

Matlab found values for a, b, and c that minimize the difference between the left and right sides of the equation.

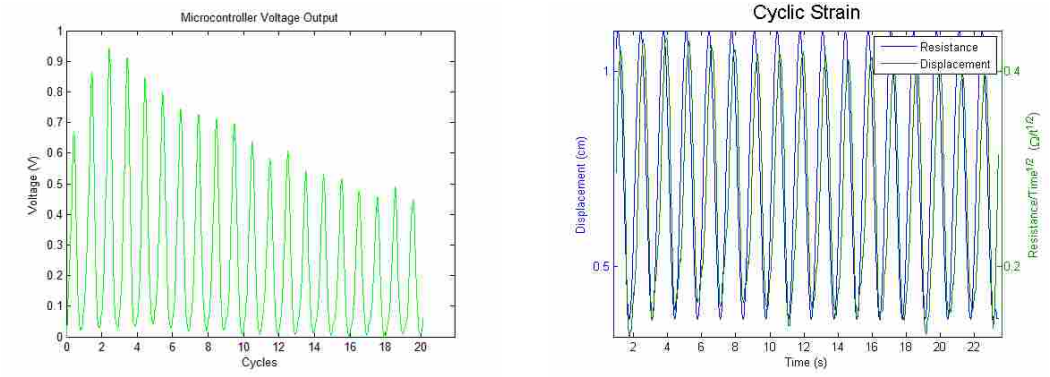

Figure 1.24: The graphs show the microcontroller's response to a cyclic test. The blue line represents the displacement from the Instron and the green line represents the voltage.

Using the voltage divider circuit described earlier the voltage response (displayed in Fig. 1.24, Left) was inversely proportional to the resistance. This meant that the response from the microcontroller, where the cycles got larger at first and then smaller, was, in fact, the same pattern as the resistance response measured by the NI-9219. Though the voltage increased the first few cycles and decreased the rest of the test, the resistance response looks similar to earlier tests. The fit achieved an average error of 13.13% over the 20 cycles, just above the 10% error that Tommy Hyatt found when testing NCSGs made through older production methods, as well as the 8.55% error measured and mentioned earlier within this paper [35].

1.9 Insole Results

1.9.1 Proof of Concept

The graphs shown in Fig. 1.25, labeled QM and Sylgard, were the voltage results from the microcontroller using the insole with a single gauge. The graph labeled QM had a QM base and the graph labeled Sylgard had a Sylgard base. To ensure that the tests between the two different types of NCSGs were the same, the same person performed both tests, the NCSGs were placed in the same position on the insole, and the same walking course was used.

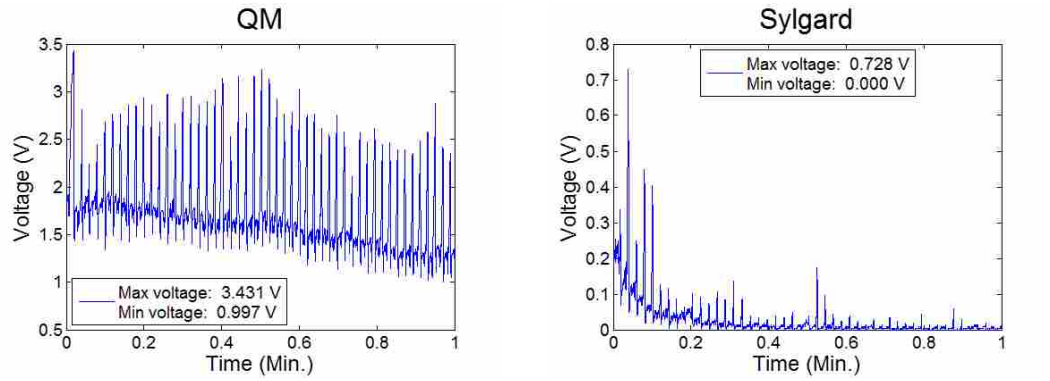

Figure 1.25: Strain gauge results when attached to the bottom of the insole while the subject walks at a normal, steady pace.

The upward spikes in the readings were the result of the subject flexing the pad of the foot during the normal walking motion. Between the spikes all signals produced by the test subjects have unique characteristic signals that repeat over each stride; each person's unique stride reside there. In the QM gauge test the signal sloped downward at 0.5V/min for the QM gauge while the sensitivity of the gauge stayed remarkably constant throughout the test.

The measured voltage using the Sylgard gauge degraded very fast and quickly lost sensitivity during the test. As this was only a one minute test, the Sylgard material was not suitable for strain measurement using the insole method without changes to the system. The decrease in voltage, as seen in these results, was a result of an increase in resistance in the gauge; the Sylgard gauge was increasing in resistance from the beginning to the end of the test. It was hypothesized that the Sylgard degraded so quickly due mainly to temperature effects in the Sylgard gauge. Both the QM and the Sylgard gauge tested were comprised of 9% NiNs and 3% NCCF. The QM gauge, however, had a resistance in the range of 200-900Ω while the Sylgard gauge had a resistance in the range of 10-60Ω.

1.10 Conclusions

The NCSGs' manufacturing process was improved and streamlined in the present work. Although the preferred spray-able method did not work the mold-formed gauges gave the same benefits that the spray method, but to a slightly lesser degree. Compared to the original gauges formed with out molds, the mold-formed gauges substantially decreased the thickness of the gauges as

well as making the properties (conductivity, stiffness, etc.) more consistent from one gauge to the next.

The drift mitigation factors applied to the NCSGs helped reduce some of the aspects of the drift. Embedding electrical contacts and using AC instead of DC decreased the observed drift. The temperature test determined that temperature was an important factor to consider while straining the gauges. If the temperature was held constant for the length of tests then temperature effects would be mitigated.

The portability of the gauges was tremendously improved from previous iterations. All past iterations included measuring the NCSGs with a computer. Using a microcontroller to measure data vastly improved the system's portability. The microcontroller attached to subjects directly with a commonly worn smartphone athletic holder. Also, an application of the NCSG in a portable system gave consistent results. A new matrix material, QM silicone, of lower modulus was also tested, with immediate advantages The insole testing that compared the QM and Sylgard silicone supported the decision to use QM silicone since it gave superior results to the Sylgard. Also, the results proved that the microcontroller gave useful signals in a working environement and that the NCSGs reacted appropriately when put into an insole system.

Although the NCSG development has come forward rapidly in the mentioned work. The temperature effect could be mitigated by measuring the temperature of the gauge while it performs tests and using that data to find more exact strains. The portability of the NCSGs have been increased vastly yet smaller microcontrollers could possibly attach to the shoe of the subject, removing even the small annoyance of strapping smartphone holders to a subject's leg.

CHAPTER 2. SENSOR MODELING

2.1 Motivation for Modeling

Manufacturing NCSGs is a time intensive process, even when changing only a few variables. For instance, when QM and Sylgard bases were being compared on how they reacted as they were strained, two separate batches of NCSGs' had to be made by hand taking approximately four hours of manpower and 2-3 days. In order to find a NCSG with the optimal reaction to strain, many different gauges had to be produced by hand varying the weight percent of NiNS and NCCF slightly in each. A well thought-out model that includes realistic assumptions and correct boundary conditions could predict that information much faster and with less human error and manual labor. The desired weight percents could be inputted into the computer with the simulations running automatically, not wasting anybody's time manufacturing gauges or introducing human error into each production run. The end goal of developing a predictive computer model is to create a model where the desired weight percents of NiNS and NCCF and the properties of a base material could be inputted, and the model would accurately predict the resistance each gauge exhibits under strain.

This final destination is far beyond previously developed models' reach and, while not expected to be achieved in the current work, is the final end-goal of the modeling work. The sub-goal of this work was to accurately predict the local reaction of nano-junctions between conductive particles to globally applied strain, using Finite Element Analysis (FEA) methods.These junctions, and their evolution, are thought to control the fundamental electrical properties of the NCSGs. Also, an analytical mathematical model was developed to corroborate results from the FEA, ensuring that the FEA would be an accurate reflection of the physical reaction of the material. The analytical model was developed by Adam Bilodeau, an undergraduate research assistant who was working on the research team. Finally, the current work was designed to answer whether the relatively simple previous models were sufficiently advanced to correctly predict the evolution

of junction gap distance with strain and, if not, provide a more substantial model that correlated better to observed phenomena.

2.2 Background on Previous Modeling Work

Johnson et al. published a work, accomplished at BYU, that modeled changing junction gap distances [10]. The work required quite a few assumptions that were not supported by any tests or analysis. First, he assumed that a Weibul distribution was a good representation of the average junction sizes in an unstrained NCSG. He further assumed that the orientations of the NiNs weren't relevant, but that the gaps between strands were randomly oriented across all possible orientations. A simple Hooke's law approach was used to determine the change in gap size with respect to the macroscopically applied strain. Any or all of these assumptions could make the previous model irrelevant or less accurate.

Using those assumptions, he created a Monte Carlo Simulation in Matlab that tested the junction gap orientation relative to the applied strain and the size of the gap. The Monte Carlo Simulation took 10,000,000 possible junction gap orientations and sizes and strained them in steps to model development of the gap size distribution. This way he was able to observe the evolution of gaps with strain, and apply those predictions to a quantum tunneling/percolative resistance model (not discussed in this paper).

2.3 Analytical Mathematical Model

The analytical mathematical model, produced by Adam Bilodeau, was built simultaneously with the Finite Element Analysis (FEA) model. The two were developed together to compare gap evolution predictions. The analytical model was more sophisticated than the previous model because it only accounted for two different particular geometries. The analytical model could not realistically be expanded to complex geometries; the FEA was corroborated using the model's results with simple geometries but also had the ability to be expanded to a representation of all possible geometries.

While it was not expected that the models would give the exact same results if the two were not similar they both would be reviewed to find errors in the programming, logic, and/or math.

Following this validation method resulted in higher confidence in the FEA model; knowing that under simple conditions there was some evidence that the FEA model was working as expected.

2.3.1 First Mathematical Model

The mathematical model and the FEA needed a common reference coordinate system. The easiest coordinate system to adhere to was the spherical coordinate system, as shown in Fig. 2.1 [36]. Each NiN's center was placed at the origin of the axis with one endpoint of the assumed straight strand lying at point P in the figure. The length of the strand is $2 * \rho$. Since ρ was consistent from test to test (500nm) the only inputs needed to orient the NiNs were θ and ϕ. Once oriented the origin of the strand coordinate system needed to be placed in the Silicone to produce the desired gap distance and orientation relative to a single nearby strand.

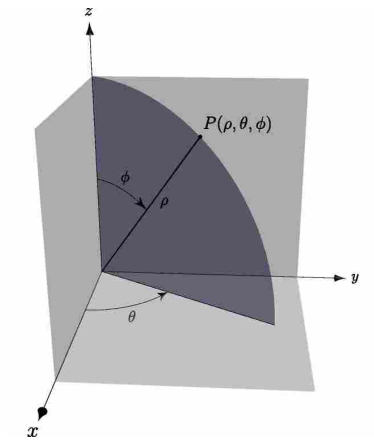

Figure 2.1: Mathematical spherical coordinate system [36].

The first validation oriented the NiNs at $\phi(1) = \phi(2)$ at 0° and $\theta(1) = \theta(2)$ at 90° with the NiNs standing parallel along the a-direction (Fig. 2.2). The strain occurred in the x-direction which meant the NiNs were aligned perpendicular to the strain. The two NiNs were separated by a small gap in the x-direction; thus the 'gap direction' was parallel to the x-axis. Orienting the NiNs in this manner meant that the NiNs would not experience any unbalanced forces that might cause a rotational displacement of the NiNs. This allowed for a corollary assumption that the closest gap between the two NiNs did not change location with strain. Initial gap sizes of 1, 3, 5, 7, 9, 10, 11, 13, 15, 17, 19, and 20nm were compared at 10% strain.

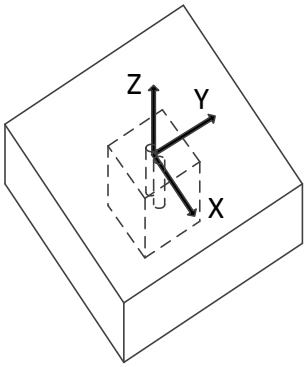

Figure 2.2: The NiNS were vertically aligned with the strain occurring in the horizontal plane. Surrounding the NiNs in the central (dashed) box is Sylgard silicone material. Outside of the dashed bos is assumed to be an homogeneous composite material of silicone and Nickel.

The Sylgard material was assumed to be linearly elastic in the range of 0 - 40% tensile strain [37, 38]. Therefore, the mathematical model employed classical linear stress and strain equations, beginning with the linear definition of mechanical stress:

$$\sigma = \frac{F}{A} \tag{2.1}$$

The area (A) represented cross-sectional areas of the faces on the end of the specimen, i.e. faces parallel to the y-z plane (as defined by Fig. 2.3). As the hypothetical specimen was strained, the cross-sectional areas remained on this plane. Another assumption made was the force (F) was the same on all x-z cross-sectional areas of the model since the faces were completely attached to each other and all reactionary forces were equal and opposite. Therefore, since the areas on each face were the same and the forces were equivalent (with the approximation that the cross sectional slice have equivalent areas throughout the test):

$$\sigma_1 = \sigma_2 = \sigma_T \tag{2.2}$$

Where σ_1 was the stress in the homogeneous composite material (Sylgard plus Nickel) near the two faces of the dashed box that are perpendicular to the x-axis, σ_2 was the stress in the Sylgard material, and σ_T was the total stress in the model. Using Hooke's Law (Eq. 2.3) we substituted the

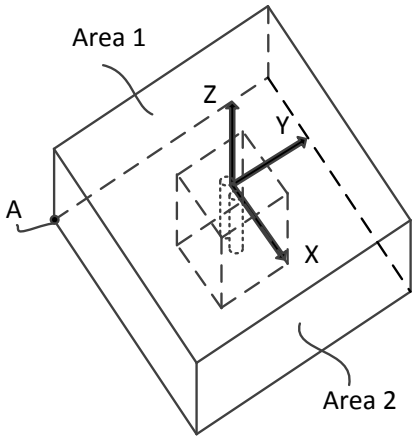

Figure 2.3: FEA Model with the NiNs embedded into the Sylgard embedded into the composite material.

Young's Modulus and strain of the material into Eq. 2.2 then simplified and solved for ε_2, where ε_2 was located in the material in the junction gap.

$$\sigma = E * \varepsilon \tag{2.3}$$

$$E_1\varepsilon_1 = E_2\varepsilon_2 = E_T\varepsilon_T \tag{2.4}$$

$$\varepsilon_2 = \frac{E_T\varepsilon_T}{E_2} \tag{2.5}$$

The composite material was modeled as springs to solve for the specimen macro-modulus (E_T). The following equation relates spring stiffness (K) to a material's modulus of elasticity when the material was stretched axially:

$$K = \frac{EA}{L} \tag{2.6}$$

In this model, "A" was the same area used in Eq. 2.1 and "L" was the same length used to calculate strain along the tensile axis. Since the different material segments were connected end-to-end in

series, finding the total stiffness of the materials in series was found through the spring-in-series stiffness equation:

$$\frac{1}{K_T} = \frac{1}{K_1} + \frac{1}{K_2} + ... + \frac{1}{K_n} \tag{2.7}$$

When the model was strained in the x-direction there was a plane of symmetry parallel with the x-z plane along the center of the model. Because the forces on either side of this plane were equal and opposite, there were no unbalanced forces to move material across the center line of the model. With no mass transfer through the center line (the x-axis in Fig. 2.2), the mathematical model only needed to determine the strain along the center line to understand how the junction gap changes. However, there were three different materials in this plane that needed to be accounted for. Each of these materials were modeled as springs with different spring constants for the model, as depicted in Fig. 2.4.

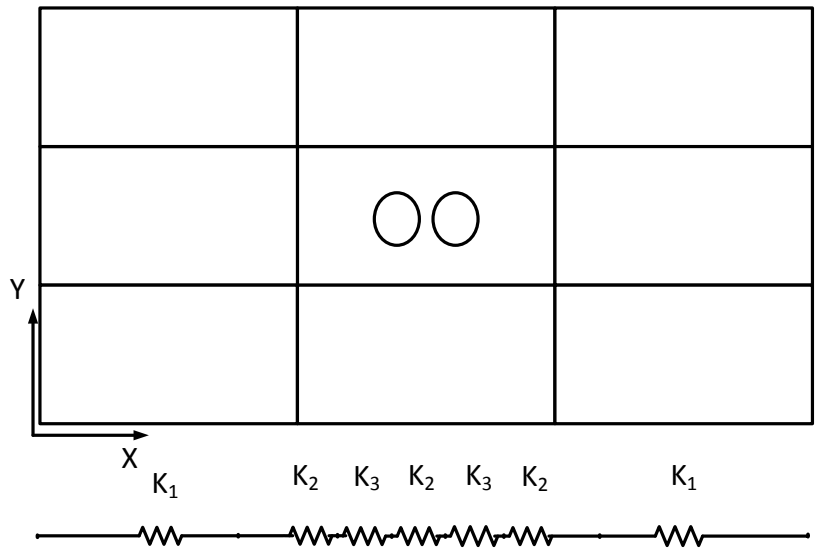

Figure 2.4: Composite material modeled as springs in series with their ends connected.

Combining the information in Fig. 2.4 and Eq. 2.7 results in the equation:

$$\frac{1}{K_T} = \frac{1}{K_1} + \frac{1}{K_2} + \frac{1}{K_3} + \frac{1}{K_2} + \frac{1}{K_3} + \frac{1}{K_2} + \frac{1}{K_1} \tag{2.8}$$

The material's modulus was directly related to the spring stiffness, the variable in the denominators of Eq. 2.8. Therefore, the larger the modulus of elasticity, the less impact the material had on the total stiffness. The Nickel strands have a modulus that was approximately 10,000 times greater than the modulus of the surrounding silicone material. Because the difference in magnitude was so large, the forces required to strain the silicone a significant distance caused negligible deformation in the NiNs; insignificant enough that the stiffness in the NiNs was ignored. Furthermore, to simplify the model, it was assumed that the Sylgard material surrounding the NiNs underwent the same strain as the rest of the homogeneous composite material. This assumption was inaccurate due to strain-concentrations, but a strain concentration factor was added later. Therefore, the springs-in-series connection was simplified into a three-spring model (with the spring in the center $\frac{5}{6}$ the length of one of the elements to account for the shorter length of the Sylard material along the x-axis of the central box; the Nickel material occupying the final $\frac{1}{6}$) as seen in Fig. 2.5 and Eq. 2.9.

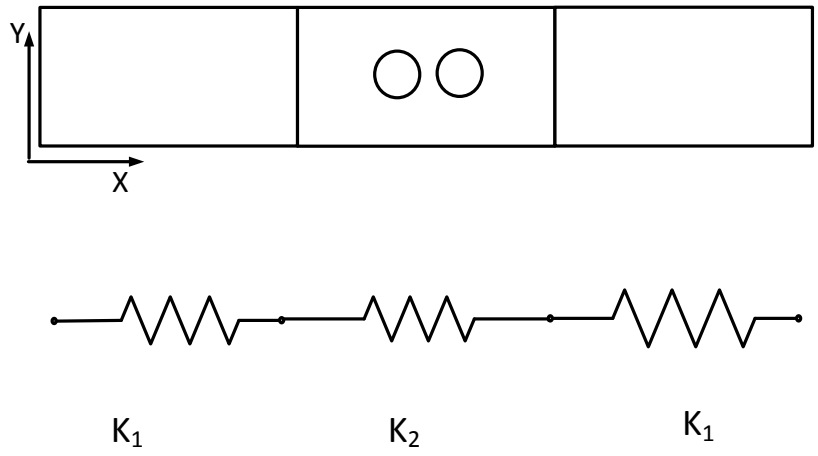

Figure 2.5: Springs connected together in a series connection.

$$K_T = (\frac{1}{K_1} + \frac{1}{K_2} + \frac{1}{K_1})^{(-1)} \tag{2.9}$$

Eq. 2.6 was inserted into 2.9 and solved for E_T:

$$\frac{1}{\frac{E_T A}{L}} = \frac{1}{\frac{E_1 A}{\frac{L}{3}}} + \frac{1}{\frac{E_2 A}{\frac{L}{3}\frac{5}{6}}} + \frac{1}{\frac{E_1 A}{\frac{L}{3}}} \tag{2.10}$$

$$E_T = \frac{18E_1E_2}{12E_2 + 5E_1} \tag{2.11}$$

Next,ε_2 was solved for by inserting Eq. 2.11 into Eq. 2.5:

$$\varepsilon_2 = \frac{18E_1\varepsilon_T}{12E_2 + 5E_1} \tag{2.12}$$

$$\varepsilon_2 = \frac{\Delta l}{l} \tag{2.13}$$

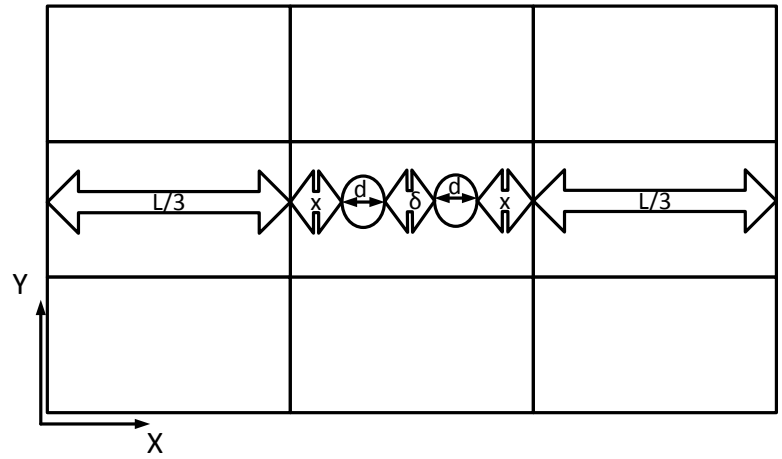

Figure 2.6: Definition of the materials that undergo strain in the mathematical model.

Combining Eqns. 2.12 and 2.13 gave:

$$\frac{\Delta l}{l} = \frac{18E_1\varepsilon_T}{12E_2 + 5E_1} \tag{2.14}$$

The length "l" corresponded to the total length of each segment in the center section along the central axis. From Fig. 2.6 "l" and "Δl" were defined as:

$$l = 2d + 2x + \delta \tag{2.15}$$

$$\Delta l = 2\Delta d + 2\Delta x + \Delta\delta \tag{2.16}$$

But since the NiNs had a negligible change of their length ($\Delta d = 0$), Eq. 2.16 reduced to:

$$\Delta l = 2\Delta x + \Delta\delta \tag{2.17}$$

While the central axis of symmetry implied that no material transfers across the central axis, it also indicated that net forces along that axis were parallel to the axis. Therefore, by Newton's third law, all forces along this axis must have been the same magnitude. Since we were also dealing with areas nearly infinitesimal in size, the equivalent forces translate into equivalent stresses (much like in Eq. 2.1). Next, Hooke's Law was useful:

$$\varepsilon = \frac{\sigma}{E} \tag{2.18}$$

From Hooke's Law, any two material elements with the same modulus undergoing the same stress strain the same amount [39]. Therefore, it was possible to solve for Δx:

$$\varepsilon_{Sylgard} = \frac{\Delta x}{x} = \frac{\Delta\delta}{\delta} \tag{2.19}$$

$$\Delta x = \frac{\Delta\delta x}{\delta} \tag{2.20}$$

Combining Eqs. 2.17 and 2.20 and simplifying gave:

$$\Delta l = \Delta\delta(2\frac{x}{\delta} + 1) \tag{2.21}$$

Inserting Eqs. 2.16 and 2.21 into Eq. 2.14 and simplifying finally resulted in an equation with only the variable that we were solving for, "$\Delta\delta$":

$$\Delta\delta = \frac{18E_1(\delta + 2x + 2d)}{(12E_2 + 5E_1)(1 + 2\frac{x}{\delta})}\varepsilon_T \tag{2.22}$$

This equation was used to verify the ANSYS results using the gap sizes and strain declared earlier.

2.3.2 Second Mathematical Model

The second mathematical model was setup similar to the first model but with the gap situated perpendicular to the direction of strain instead of parallel. The strain occurred in the x-direction as before. Also, any variables that are the same between the two models, such as E_1 and E_2 represent the same properties as in the first model.

The validation test cases varied the junction sizes from 1 to 20nm and were strained 10% in the same way as before.

We label two strains useful to this study: ε_2, which was the strain in the Sylgard material along the x-direction, and ε_3 which was the strain in the x-direction in the homogeneous composite material sections that were adjacent to the Sylgard (Fig. 2.7). The orientation in Fig. 2.7 is a top view of the model with accentuated features that represent what happens to this geometry when it undergoes uniform strain in the x-direction. The surrounding composite material had a modulus of elasticity (E_1) that was about 50 times greater than the modulus of the pure Sylgard (E_2) in the center section of the model. Ignoring the relatively small Nickel strands resulted in the center material stretching over a much greater distance than the composite material.

The difference between E_1 and E_2 caused the greater strain already explored mathematically in the first half of the model (explained by comparing the tensile axis of material to a spring). The only difference between the first model and this one was that now the NiNs are no longer the center axis. This resulted in a simpler three-tension spring model with no difference in initial length between the two outer sections and the center section. With the simplified model, Eq. 2.10 changed to the simpler form:

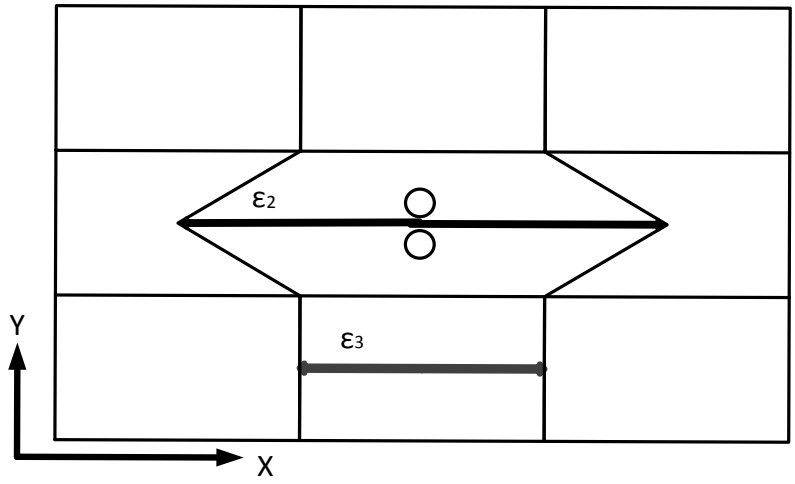

Figure 2.7: The displaced junction with accentuated strained features.

$$\frac{1}{\frac{E_T A}{L}} = \frac{1}{\frac{E_1 A}{\frac{L}{3}}} + \frac{1}{\frac{E_2 A}{\frac{L}{3}}} + \frac{1}{\frac{E_1 A}{\frac{L}{3}}} \tag{2.23}$$

Which then resulted in Eq. 2.11 changing slightly to become:

$$E_T = \frac{3E_1E_2}{2E_2 + E_1} \tag{2.24}$$

ε_2 (measured along this central axis) was the largest strain represented in the model (as seen in Fig. 2.7). ε_3 was being measured across a section of composite material. Since the elements of composite were assumed to have exhibited uniform isotropic material properties, the strain across the central section of composite material was the same as the strain in the two adjacent sections, so $\varepsilon_3 = \varepsilon_T$. This caused a difference in strains between ε_2 and ε_3, even though they both began with the same initial length. Since E_1 was 50 times greater than E_2, the softer pure-Sylgard section had a minimal impact on how ε_3 changed. For simplicity's sake, it was assumed that the Sylgard had no impact on the changing length of the composite material and that there was a linear change in strain where Sylgard and the composite material sections met. These assumptions were evident in the accentuated features of Fig. 2.7.

Some new variables needed to be introduced now to develop the model. The new variables were b, q, and Q, as seen in Fig. 2.8. This figure is a representation of only the central portion of Sylgard with the two NiNs inside. The variable b was the perpendicular distance from the edge of the Sylgard material to the point on the nearest nanostrand that was part of the junction gap being measured. q was any perpendicular distance from the edge of the Sylgard, and Q was the perpendicular distance from the edge to the middle of the junction (one half the length of the material section).

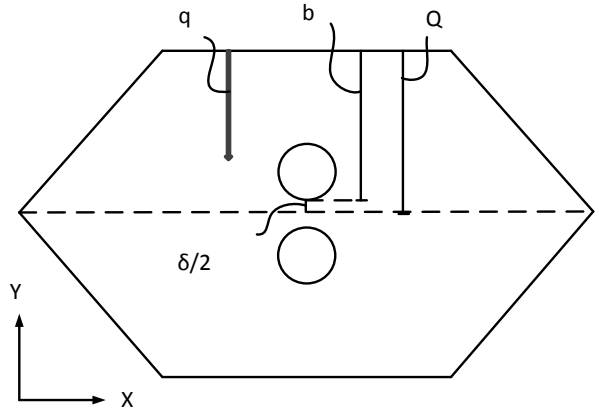

Figure 2.8: Sylgard material situated in the X-Y plane from the center of the gauge (see Fig.2.7) with newly defined variables: b, q, and Q.

Next, an equation was developed to find the strain a distance q away from the composite material (which was straining by the amount ε_3). ε_{axial} indicates the strain along any straight line parallel to the applied strain, whereas $\varepsilon_{transverse}$ indicates strain perpendicular to ε_{axial}. Since the changing strain between ε_3 and ε_2 was assumed to be linear, the instantaneous axial strain at any location q (with $q < Q$) away from the composite material was defined as:

$$\varepsilon_{axial} = \varepsilon_3 + (\varepsilon_2 - \varepsilon_3)(\frac{q}{Q}) \tag{2.25}$$

Poisson's ratio in the Sylgard material was then applied (using a rearranged version of the definition) to convert between ε_{axial} and $\varepsilon_{transverse}$:

$$\varepsilon_{transverse} = -\nu * \varepsilon_{axial} \tag{2.26}$$

Eq. 2.25 was then inserted into Eq. 2.26 to get the instantaneous transverse strain—the strain that caused a change in the gap between the two NiNs.

$$\varepsilon_{transverse} = -\nu[\varepsilon_3 + (\varepsilon_2 - \varepsilon_3)(\frac{q}{Q})] \tag{2.27}$$

The before-mentioned assumption, $\varepsilon_3 = \varepsilon_T$, was used in conjunction with the work derived in the first model. More specifically, the work leading up to and including Eq. 2.12 (with the notable exceptions explained above with Eq. 2.24). The assumption and work were then used to expand Eq. 2.12 to include material properties. The assumptions and equations were input into Eq. 2.27 and simplified, giving:

$$\varepsilon_{transverse} = -\nu\varepsilon_T[1 + (\frac{3E_1}{2E_2 + E_1} - 1)(\frac{q}{Q})] \tag{2.28}$$

Because $\varepsilon_{transverse}$ was an instantaneous strain along a 1-dimensional horizontal axis, it became necessary to find the average strain over the junction distance. Using the assumption of linearity again, average transverse strain (ε_{Tavg}) between the center Q and b was computed:

$$\varepsilon_{Tavg} = \frac{\varepsilon_Q + \varepsilon_b}{2} \tag{2.29}$$

"Q" and "b" were substituted into "q" in Eq. 2.28 and simplified:

$$\varepsilon_{Tavg} = \frac{-\nu\varepsilon_T[(1 + (\frac{3E_1}{2E_2+E_1} - 1))\frac{Q}{Q} + (1 + (\frac{3E_1}{2E_2+E_1} - 1))(\frac{b}{Q})]}{2} \tag{2.30}$$

$$\varepsilon_{Tavg} = \frac{-\nu\varepsilon_T}{2}(2 + (\frac{3E_1}{2E_2 + E_1} - 1)(1 + \frac{b}{Q})) \tag{2.31}$$

In order to remove the extraneous variable "b", it was noted in Fig. 2.7 that "Q" minus one half of the gap size ($\frac{\delta}{2}$) was equal to exactly "b", with "δ" representing the gap size. Also, due to symmetry along the central axis, the total gap change was found by multiplying the change in half of the gap by 2.

Since the strain correlated to the change in length of the gap (see Eq. 2.13) the change in the junction gap was computed through:

$$\Delta\delta = \delta\varepsilon_{Tavg} \tag{2.32}$$

Inserting Eq. 2.32 into Eq. 2.31 resulted in:

$$\Delta\delta = \frac{-\nu\delta\varepsilon_T}{2}[2+(\frac{3E_1}{2E_2+E_1}-1)(1+\frac{Q-\frac{\delta}{2}}{Q})] \tag{2.33}$$

This was the final equation used to compute "$\Delta\delta$". From this equation all variables were known except for "$\Delta\delta$", the change in gap size. This equation was then solved for a range of gap sizes and compared to the ANSYS results.

2.4 Finite Element Analysis Model

The FEA model used to analyze evolution of junction gaps with material strain was built using the computer modeling program, ANSYS.

2.4.1 Purpose of the Model

The model determined how adjacent NiNs reacted when an outlying material was strained. The hypothesis of the piezoresistive effect in the NCSGs at the time of model creation was that the effect happened due to a decrease in gap size, minimum distance between two adjacent NiNs, for a majority of junctions in the material. The gap size decreased when the NCSGs were strained because of Poisson's thinning effect occurs. The model tested whether the NiNs' gap size increased or decreased and how much the gap size changed when the NiNs were in different orientations, experienced different strains, and had a range of gap sizes (the computer code is given in Appendix A).

2.4.2 Model Setup

The model measured the junction distance as surrounding material was stretched. In the model, only one junction was modeled with the resulting strain coming from adjacent silicone-

like material. The junction distance was defined as the minimum distance between two adjacent NiNs since this distance defined the gap that needed to be crossed by electrical current. The Ansys program comprised of two NiNs with the silicone base material enveloping the NiNs. The material properties of the constituents of the strain gauge are listed in Table 2.1.

Table 2.1: Material property of constituents in the FEA model [37, 38, 41, 42].

	Young's Modulus (GPa)	Poisson's Ratio (m/m)	Density (kg/m3)
NiNs	207	0.31	8890
Base	0.015	0.45	1.10
Composite	0.574	0.449	1.14

Two straight cylinders represented two NiNs. As mentioned previously, the NiNs used were bifurcated structures with high aspect ratios. While the straight cylinders created had relatively high aspect ratios they were not bifurcated. The cylinders had a diameter of 100nm and a length of 1000nm, resulting in an aspect ratio of 10:1 in the model, which was a typical aspect ratio of a portion of a NiNs that was relatively straight and non-bifurcating. The cylinders were fully encapsulated by the base material with the junction approximately at the origin of the model. The cylinders were attached to the base material by 'gluing' intersecting areas.

The cylinders representing the NiNs were enveloped by a cubic block representing silicone surrounding the NiNs (Fig. 2.9). Each side of the cube was 1200nm. The material are given in Table 2.1 [37, 38]. No matter the positioning of the NiNs they were assumed to be completely wetted by the surrounding silicone material.

The silicone (represented by the square in the center of the block) was surrounded by homogeneous composite material (representing a mixture of silicone and NiNs) on four planar sides, as shown in Fig. 2.10. The thickness of the surrounding composite material was 1200nm, equal to the thickness of the silicone cube. The resulting rectangle had sides of 3600nm, meaning the silicone cube was 1/9 of the total volume (Fig. 2.10). The composite material acted as a constraint on the silicone material, allowing it to move approximately how it would move if it was surrounded by the composite mixture of silicone and NiNs. The NiNs took up approximately

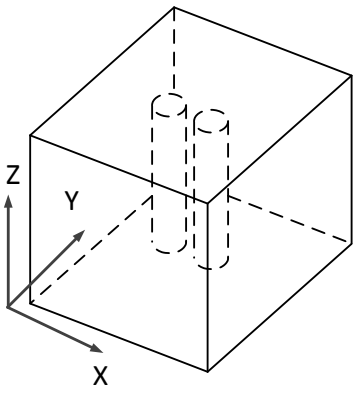

Figure 2.9: The silicone cube was surrounded on four planar sides by the composite material.

0.45% of the volume covered by the base material in the ANSYS model; therefore, the surrounding composite material had the properties of the silicone mixed with 0.45% by volume of NiNs.

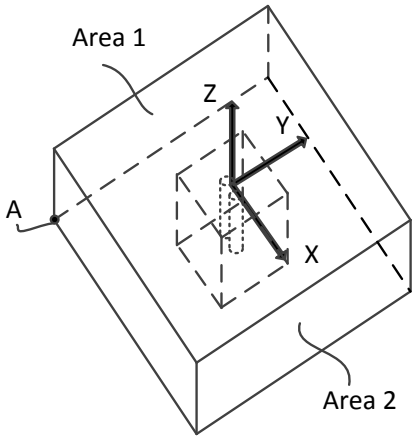

Figure 2.10: FEA Model with the NiNs oriented perpendicular to the strain direction.

The composite material had a point, point "A" in Fig. 2.10, that was constrained in all directions. "Area 1" in the y-z plane that enveloped point "A" was constrained in the x-direction. This ensured that the model could not get pulled in the x-direction but the constrained area could

expand or constrict in y- or z-direction, representing natural reactions. The model was strained by selecting all the nodes in "Area 2" (y-z plane on the area opposite the constrained area) and moving them a distance correlating to the desired strain.

User Code Inputs

The user first chose the angles at which the NiNS would be oriented. Next, the user input the initial gap distance and the overall percent strain of the system. This meant that the strain desired must be multiplied by the length of the model (3600nm) to get the necessary displacement. ANSYS meshed the geometries and solved for the displacement at each node in the mesh. After solving for the node displacements ANSYS wrote a file with the displacement and original location for each node located in the two NiNs. Matlab then read the file and added the original locations to the displacements to get the final locations. Each node location in the one cylinder was checked against each node in the adjacent cylinder to find the new minimum distance between the cylinders. The change in gap distance was found by subtracting the original junction distance from the final junction distance.

2.4.3 Analysis

After the hand calculations and ANSYS FEA model were completed the model was evaluated to develop a library of solutions. The entire library is a database with thousands of combinations of orientations, gap sizes, and strains. A user could potentially input all of the previously mentioned data into the library and determine the final gap size without running a new FEA model. This system would be much faster and easier computationally than running new computations through Matlab and ANSYS.

The orientation directions for the NiNs were inputted using θ and ϕ for each NiN, as shown in Fig. 2.1 and described in section 2.3.1: θ_1 and θ_2 ranged from 30° to 180° and ϕ_1 and phi_2 ranged from 30° to 90° in steps of 30°. The gap size was set to 4.5 nm, a possible average gap size for a NCSG. All possible combinations were tested, resulting in results for 324 different permutations. The gap changes were saved into the library such that any orientation could be

quickly solved by interpolating the previously solved data. The final gap sizes were compared to the older, simpler model put forth by Oliver Johnson et al. [10].

2.5 Results and Discussion

2.5.1 Validation of Finite Element Analysis with Analytical Mathematical Model

First Model Validation

The FEA and mathematical model were compared using the first validation geometry (see section 2.4.2), and 10% strain, with the results displayed in Fig. 2.11. In the first orientation the difference in the results for the two models was much greater for small junction sizes than larger junctions. At the junction distance of 1 nm, there was a difference of 50.7%. While the percentage was high the difference between the two models was only 0.517 nm. As the initial gap size increased the overall difference decreased. It reached a minimum of 0.892% at a gap size of 11 nm. At a gap size of 7nm or larger the error between the two models was below 7%.

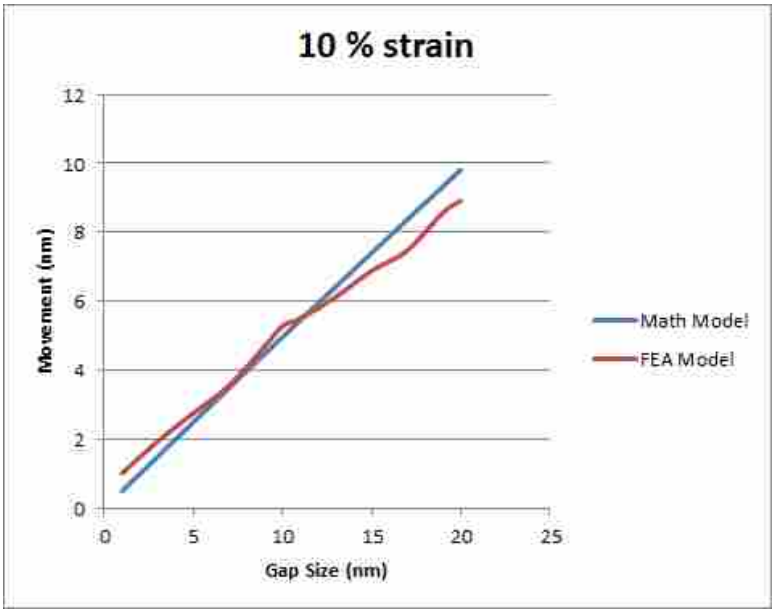

Figure 2.11: The validation results comparing the FEA (red) and mathematical model (blue).

An Excel curve-fit was applied to the FEA results to see how closely the slopes matched. The FEA line had a R^2 value of 0.9945, a close fit. The line had a slope of 0.409 and a y-intercept of 0.792. The analytical model had a slope of 0.4898 and the y-intercept was 0.0512.

The FEA was being affected by many different factors that were not accounted for in the mathematical model. The factors that interplay in the analysis caused the non-constant slope in Fig. 2.11 to occur. The bumps around 10nm and 17nm consistently occur in the FEA, though further work needs to study what was happening to better understand why.

The two models had closer predictions of the gap change at larger gap sizes, in terms of percent difference. However, even though the percent difference was less at higher gap sizes the two seemed to be diverging as the gap size increased. Further work using larger gap sizes could answer whether that trends continues or not. The two models gave values that were similar enough to build some confidence that the FEA was being built without major errors for the first orientation.

Second Validation

The results for the second validation, derived from Eq. 2.37, are displayed in Fig. 2.12. The two models underwent 10% strain for the gap sizes ranging between 1 and 20nm. The largest difference was once again at 1nm at 64.4%. The offset caused the large errors. The best correlated junction distance was 17nm, which had a difference of 10.97%.

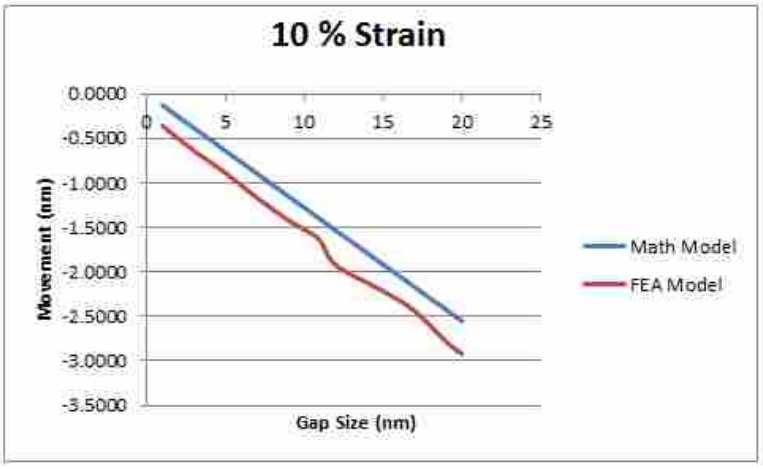

Figure 2.12: The validation results comparing the FEA (red) and mathematical model (blue).

The major difference between the two models was the contrasting y-intercepts. The mathematical calculation predicted that the final gap size would be linear from the point (0,0) in Fig. 2.12 throughout, hence the y-intercept of 0. The FEA, however, had a much larger y-intercept, 0.2254. This meant that either the FEA doesn't work well for very small initial gap sizes (perhaps due to meshing constraints) or the fairly straight line displayed in Fig. 2.12 curves toward the (0,0) point on the graph.

Although the offset between the two models was not ideal it was encouraging that they had very similar slopes. Also, with both validations in the same range of answers it was assumed that the FEA was using more advanced techniques than the hand calculation method and the error came from the additional complex factors and not mistakes in the setup or modeling of the FEA.

2.5.2 Validation of Finite Element Analysis with Previous Modeling Results

Since the FEA gave values close to the mathematical model, the next step in the FEA validation began. The graph below depicts the final gap distance for each orientation at the four different strain values with interpolation applied between the known points (Fig. 2.13).

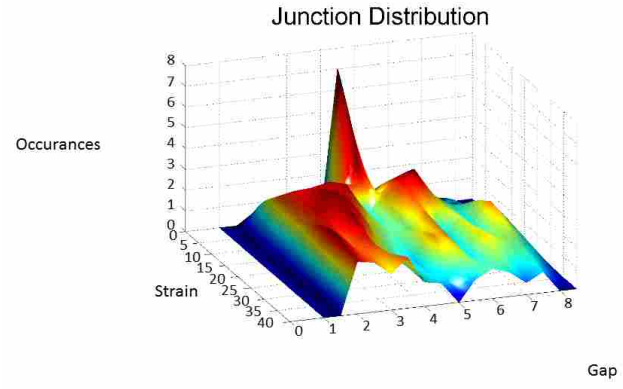

Figure 2.13: The surface represents the final gap sizes after being strained by the amount shown on the strain axis, for an initial gap of 4.5nm, and a full range of possible orientations of the two NiNs.

The most common junction gap distance consistently decreased as the strain was increased. At zero strain all orientations of NiNs had the original 4.5nm gap. When the model was increased to 10% strain the maximum number of final gap distances ended up at approximately 3nm. At 20%

the peak was at 2.7nm. At 30 and 40% the peak was closer to 2nm. This trend fits in well with what is observed when a gauge made of this material is strained; the gauges continue to decrease in resistance as gauges are increasingly strained, possibly indicating a closing of the average junction gap between conductive NiNs.

The graph also had an interesting double hump. The larger of the two humps was below 4.5nm and drifted toward 2nm as the strain increased. However, not every orientation had a junction distance that decreased, as predicted by the analytical model in subsection 2.5.1. The peak value of gaps that increased in size was above 4.5nm and below 7nm.

The results from the FEA were compared to work done by a past BYU student, Oliver Johnson [10]. Oliver took a much simpler model to determine the final junction sizes. His results are shown below (Fig. 2.14):

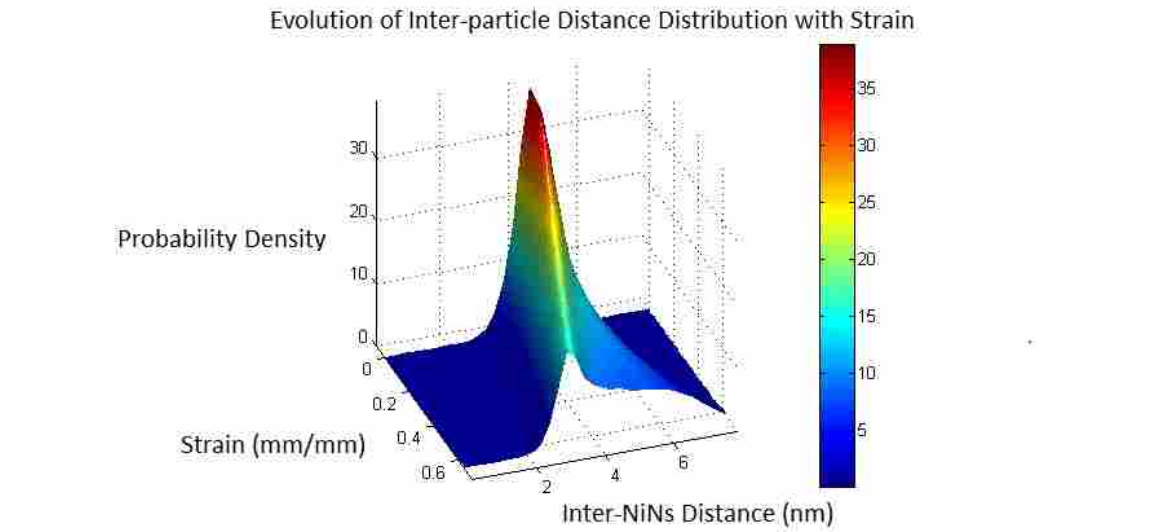

Figure 2.14: The change in junction gap sizes for different strains ranging from 0 to 60 % according to the model of Johnsen et al. [10].

The two models had the same general trend: as the strain was increased the overall junction gap distance decreased. Also, each had a smaller grouping of final gap sizes larger than the original gap size. Although similar, there were differences between the results from the two models; one was that Oliver's model had a smaller spread in junction distances' results.

2.6 Conclusions

The FEA approach was successful in developing a more realistic (and presumably a more accurate) model of gap evolution with strain. The previous (baseline) model assumed that the NiNs' relative orientations had no effect in the evolution of the gap size, that the Weibul distribution was appropriate, and that simple Poisson's contraction was the only applicable phenomenon happening as the NCSGs were strained. The FEA model took into account the correct relative orientation of the NiNs, and other geometrical effects. It also analyzed a single initial gap, and can be applied to any assumed distribution of such gaps. The mathematical model was developed and tested against the FEA with supportive results. Finally, the FEA was compared to Oliver's previous model and similarities and differences were noted.

Future work is needed to incorporate the predicted gap evolution into a full model of piezoresistance for the material, and compare this with the resistance of NCSGs at different strain levels using different weight percents of NiNs. Also, it would be useful to correlate the results of the FEA with current work performed by Adam Bilodeau who is measuring the changing average junction gap distance as the NCSGs are strained.

CHAPTER 3. GAIT DETERMINATION USING INSOLE-EMBEDDED NANO-SENSORS

3.1 Abstract

Every person has unique characteristics in the manner in which they walk, collectively termed gait. A person's gait is like a person's fingerprint; each is unique but similarities run throughout. Understanding and measuring gait can correctly determine how effective different exercises are, count how many calories a person burns while exercising, help assess a person's mental stability, and help one understand how to improve their lifestyle.

This paper describes a new measurement tool constructed to measure gait using a sensing insole. The sensing insole is a lightweight, comfortable, and compact gait sensing system comprised of a novel nano-composite strain gauge and Arduino Uno microcontroller. The nano-composite strain gauge is composed of a silicone base with Nickel nano-strands (NiNS) and Nickel coated Carbon fiber (NCCF) embedded to conduct electricity. The resultant piezoresistive response to strain is used to characterize gait.

Feature analysis of the gait data revealed distinct, recognizable characteristics for different people. Furthermore, a range of different exercises could be identified, including walking, jogging, biking, and climbing stairs. By combining this information with readily available exercise data, an estimation of the number of calories burned by the subject while exercising was available.

3.2 Introduction

Characterizing gait is important to understanding how a person moves as they walk, exercise, play, etc. Human gait can be broken down into cycles; a gait cycle begins when one foot contacts the ground and ends when that same foot contacts the ground again [43]. The gait cycle can be broken up into two different phases. First, there is the stance phase, where the foot is in contact with the ground. Second, there is a swing phase where the foot is swinging forward in the

air. Depending on whether the subject is walking or running there may be a time when there is a double stance portion, when both feet are in contact with the ground, or an air-born portion, where both feet are off the ground.

Human motion is further characterized by measuring different aspects of the human body during movement. For example, one area of research is devoted to understanding how muscles create motion of body components (kinesiology) [44]. Other studies examine the movement of the parts of the body that contribute to human gait (kinetics) [45]. Further studies look into the energy balance in the human body as the body locomotes [46]. These and other studies are slowly uncovering all aspects of how humans move.

Gait has been measured and evaluated since the time of Aristotle [47]. A more recent champion of modern gait techniques is David H. Sutherland. In the 1970s, he synchronized electromyography (EMG) readings to a movie camera, comparing the signals to the filmed motion of the subjects walking [44]. This system allowed researchers to understandwhich limbs during gait, and also which muscles are used to stabilize the human body during locomotion. The methodology was further developed into a standard gait analysis setup: a multi-camera motion capture system with a force platform having the capability to measure ground-reaction forces [48]. This framework ensures that various aspects of gait are recorded, as well as measuring interactions of the body with the ground.

This system is often paired with other sensors, such as electromyography (EMG) sensors, to determine muscle reactions during movement as well as the kinematics of the body and the forces acting on the contact points between the body and the ground. Joint angles (such as the ankle or knee) can be determined using the cameras and synchronized with the information from force plates embedded into the floor. This results in merged data from the kinematics of the human body and the ground reaction forces that subjects produce while performing those kinematics. These operations are in controlled environments where researchers can determine and analyze abnormalities relating to both limb positioning and measured forces.

Although extremely useful, the traditional lab setup to measure gait does have setbacks. One such setback in using this system is that only one stride can typically be recorded on the force plate. The only data from the force plate occurs when the subject actually steps onto the force plate and ends when the subject's foot leaves the force plate. Also, the controlled atmosphere leads to a

limited understanding of human gait and locomotion. Laboratory testing misses an element of the real-world wherein movement is not occurring in the ideal frame; obstacles arise that need to be overcome, uneven terrain breaks up the subjects' gait, etc.

Beyond the aforementioned problems the in-lab gait measurement techniques often rely heavily on the ground reaction force (GRF) plates. The GRF plates measure the ground reaction force of the overall foot. Whenever tests are performed where the subjects wear shoes, the insole and sole of the shoe often redistribute the center of pressure and the ground reaction forces. Therefore, the resultant data is a combination of the actual desired reaction from the subject's foot and the undesired reaction and redistribution of pressures and forces from the shoe. The redistribution greatly varies in different types and brands of shoes, a mainly undesirable consequence.

Due to the setbacks of in-lab measurement techniques, it became apparent that while lab setups were producing good information, portable measurement systems could gather different information and utilize different testing strategies. Therefore, researchers began searching for ways to extricate gait measurement from the laboratory. Over the past 20 years, various researchers have been working on the difficult task of developing and improving self-contained units, meant to measure gait. Different units approach gait measurement using different methods such as accelerometers, gyroscopes, magnetoresistive sensors, flexible goniometers, electromyography sensors (EMGs), electromagnetic tracking systems (ETS), conductive fabrics, and 3-axis force sensors [48]. Combinations of these sensors are beginning to give a much more comprehensive picture of the kinematics and kinetics of a person in motion in their natural environment.

Some promising methods that attempt to capture the natural human gait include embedding sensors into the shoe [49–57]. Shoe embedded sensors include capture of three dimensional ground reaction forces occurring in the shoe. Some such sensors are quite bulky and/or awkward and affect how the subjects move. However, various of these systems are now able to monitor the sensors in real-time and send feedback to either the subjects or researchers [53, 55, 57].

Live feedback opens up new avenues of patient monitoring to help doctors and patients diagnose and correct disabilities. In fact, measuring a person's gait and understanding their gait disorders are some of the best clinical measures available today [29]. Therefore, a comfortable gait measurement system capable of correctly monitoring patients' gait can be a great predictor and inhibitor of future illnesses. Some illnesses, such as Alzheimer's disease, Parkinson's disease,

and Multiple Sclerosis greatly affect a person's gait [31, 32]. There is some work suggesting that by looking at ones' gait, Alzheimer's and Parkinson's diseases can be diagnosed [58]. Since all three diseases affect gait it is hoped that accurate gait measurement devices that are simple and easy enough to be used in the home can help doctors to diagnose these problems earlier than most currently methods [59].

Beyond diagnosing diseases, gait measurement systems can also assist healthy people. Correcting a person's gait via gait retraining and can be useful in reducing stress concentrations on a person's body from walking and/or running incorrectly. Shull et al., developed a system to test whether subjects could decrease the adduction in their knee while walking using haptic feedback devices. Using real-time haptic feedback the subjects were able to decrease the amount of adduction by 29-48 % [60]. Other research uses visual cues along with tactile feedback that described a decrease in unwanted motions during gait [61]. These feedback devices do not typically use portable gait sensing technology. However, incorporating the haptic and visual feedback with portable gait sensing may allow an even wider range of use of gait retraining to reduce lower extremity injuries and fatigues.

Outdoor running is an extremely popular sport; A study by the "Outdoor Foundation" found that approximately 4.8 million Americans participated in trail running in 2009 [62]. Running outdoors results in a different workout from running on a treadmill. For instance, the hamstrings work much harder and have a much more intense workout when running outdoors than on a treadmill [63]. It is recommended that runners who will run outside at events like marathons, half-marathons, 10ks, 5ks, the Ironman, etc. should train outdoors [64]. These runners are training outdoors without the use of efficient measurement or calorie counting devices.

With 570 marathons in the US in the year 2011 alone and 551,811 finishers, there are hundreds of thousands of Americans that are running without the aid of advanced technology. Approximately 0.5% of Americans have run a full marathon, meaning 2,000,000 people in the US alone have run a marathon [65]. With specific training and instructions for marathon runners and the massive amount of exercising that they do, understanding the power they are expending and how many calories they are burning could be very important. Furthermore, a sensing insole could tell if a runner's gait is disjointed or incorrect as well as when an athlete's gait has changed due to fatigue in a race as well as training. A sensing insole, with the ability to sense power and changes

in gait could potentially revolutionize training regimes for the hundreds of thousands of marathon runners in the US.

Many systems have been developed to measure people's gaits. These systems have enlightened our understanding in the realm of human locomotion. However, there is still room for improvement and much more information to gather to better understand how people move, what can be done to help people move safer and more efficiently, and diagnose medical conditions from human movement.

3.3 Background

The sensing insole uses a novel high deflection nano-composite strain gauge (NCSG). The nano-composite strain gauge has three main ingredients: a silicone base produced by Quantum Silicones, QM 145, Nickel nano-strands (NiNS), and Nickel-coated carbon fiber (NCCF). The gauge can repeatedly reach strains up to 60% [14]. Embedded within the base are Nickel nano-strands (NiNS) and Nickel coated Carbon fiber (NCCF). The combination of the additives results in an overall conductive material. Depending on the volume fraction of additives, the resulting resistance of the gauge varies between 0.1 Ω and 10 MΩ. An optimal amount chosen for the gauges through various tests is 9% NiNS and 3% NCCF, by volume [14]

The QM silicone base material was chosen because it strains over 200% repeatedly (in the absence of any filler material) with high recoverability. Silicone is a dielectric material, which means it has a high electrical resistance, ensuring that any current preferentially travels through the embedded conductive fillers.

The second ingredient in the NCSG is NiNS. The NiNS are formed through a proprietary chemical vapor deposition process that produces a cake-like mixture of Nickel material (Fig. 3.1). This cake-like mixture is chopped into small clusters and strands of Nickel material. The deposition process forms NiNS that are unique due to their nanometer scale diameter, high aspect ratio, and their being highly branched [16]. In the current application, the Nickel nano-strands have little effect on the desirable qualities of silicone. The high aspect ratio, ranging from 50:1 to 500:1 [17], allows a substantially low volume percent (about 7%) to make the entire strain gauge conductive.

The third ingredient added to the NCSG to increase stability and consistent resistance readings was NCCF, as seen in Fig. 3.2. The NCCF is also formed by a Chemical Vapor Deposition of

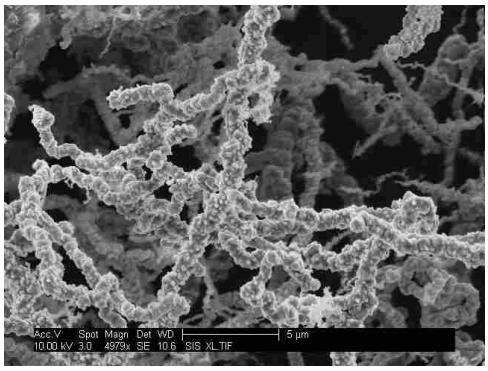

Figure 3.1: Image of NiNS when in a cake-like mixture.

nickel onto standard carbon fibers. To ensure that the NCCF has similar electrical properties to the NiNs the NCCF is coated in Nickel, specifically 35 % Ni and 65%. carbon, by weight. The NCCF is added to support the current flow within the gauge and stabilize the resistance curve. NCCF are long (0.5mm or 1mm) and straight, connecting long distances between NiNS clusters. The NCCF produce a monotonic resistance with strain, ensuring that the NCSG reads distinct values for different strains. Also, the NCCF is substantially less expensive to produce than the NiNS and is therefore used to help reduce the cost of the overall gauge.

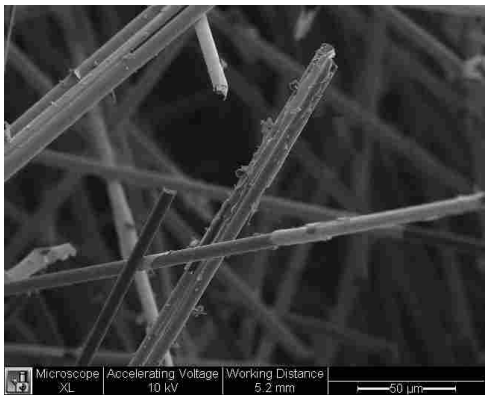

Figure 3.2: NCCF formed by proprietary chemical vapor deposition.

When strained or compressed axially the gauges exhibit a unique negative piezoresistive effect. Whereas the piezoresistive response of traditional strain gauges occurs by Poisson thinning/thickening of the conductive material, nano-composites gauges utilize both Poisson thinning/thickening and the phenomenon of quantum tunneling [13,66]. When there is a small enough

gap between two conductors, an electrical charge may "jump" across a dielectric material from one conductive element to another, essentially tunneling through the highly resistive dielectric matrix. The probability of tunneling increases as the distance between the conductive elements decreases. As the nano-composite is strained in tension the cross-sectional area decreases due to Poisson-thinning, further decreasing gap junction distances, which in turn decreases the resistance of the NCSG. As more tunneling sites become active, more routes are available to percolate the electrical signal [67–69].

Most strain gauges need a power source and data collection/analysis device. The sensing insole used an Arduino Uno microcontroller to supply voltage to the strain gauge and read the appropriate values from the gauge. The Arduino Uno microcontroller was used because it's built in an easy to use development environment and designed to interface with many different types of hardware [24]. It was expedient that the microcontroller chosen was simple to program and use, powerful enough for the application, and small enough to be easily worn on the body.

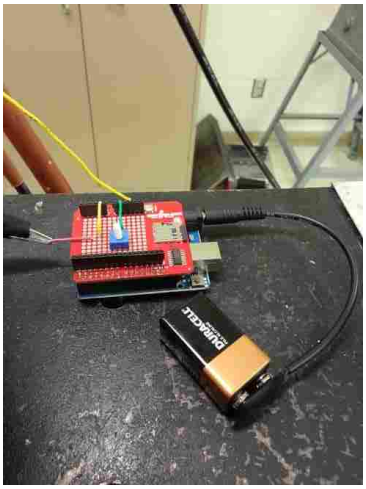

Figure 3.3: The microcontroller board (blue) attaches to the microSD shield (red) and is powered by a 9V battery.

A microSD shield was attached to the microcontroller. The microSD shield wrote output signals to a microSD card, which greatly increased the amount of storage available. The microSD shield and the microcontroller have a maximum length and width of 68.5mm and 53.35mm respectively, with the USB connector and power jack extending beyond the former dimension [24]. The height of the controller and SD shield was 23.9mm. The total weight of the microcontroller system

was approximately 105g. The size of the microcontroller allowed the subject to carry the controller without severely interfering or limiting the subject's movement. The microcontroller operated at a minimum of 6V but required at least 7V to ensure a constant output of 5V from the 5V output pin.

Future iterations of the sensing insole could take advantage of smaller microcontrollers that wirelessly stream the data to a computer for real-time analysis. Current wireless microcontrollers such as the Synapse RF 2266 with a rechargeable battery would reduce the size by approximately 93%. The overall size would be reduced to approximately 8mm tall, 25mm long, and 35mm wide. The proposed controller would result in a weight reduction of 105g to a mere 14g, which is an 86% reduction in weight. This smaller and lighter design would reduce the system's effect on the subject and allow for more accurate readings.

3.4 Method

3.4.1 Hardware

The NCSGs were cast into shoe insole molds to create sensing insoles (Fig. 3.4). Sensing insoles were developed to measure a person's gait outside of the lab environment. The sole was made from Sylgard 184, a silicone material that repeatedly stretched up to 60% elastically without breaking. Embedded within the silicone insole were four NCSG. These gauges were used because they stretch up to 60% strain with high recoverability as well as react to and withstand the pressure forces in the insole.

Figure 3.4: The sensing insole is depicted with wiring and the Arduino Uno microcontroller.

The gauges had metal clips attached to each end that were connected to the microcontroller through wires attached to the metal clips. The microcontroller sent a 5V signal through the attached wires to one end of each of the strain gauges. The wire attached to the opposite end of the strain gauge received a signal that depreciated in relation to the resistance in the gauge. This wire was in connection with an analog pin in the microcontroller. In series electrical connection with this wire was a 500Ω resistor placed on the mini breadboard attached to the SD shield, forming a voltage divider. The voltage divider read an output voltage between 0 and 5V, which was used to determine the change in resistance in the strain gauge. Eq. 3.1 is the classic voltage divider equation:

$$V_{out} = \frac{R_2}{R_2 + R_1} V_{in} \tag{3.1}$$

Eq. 3.1 was rearranged to find the resistance in the strain gauge (R_1):

$$R_1 = \frac{V_{in}}{V_{out}} R_2 - R_2 \tag{3.2}$$

Through the microcontroller setup and testing every part of Eqn. 3.2 was known except for R_1, the resistance in the gauge. Vin and R_2 were constants in the equation, relating to how the gauge was setup. Vout changed as the gauge was strained and was the data recorded by the microcontroller. R_1 changed in relation to Vout and corresponding constants in Eqn. 3.2.

The microcontroller was placed in a Belkin case, meant to carry mp3 devices while exercising. The 9V battery and the microcontroller both fit into the Belkin case. The case was strapped around the subject's ankle when a test was performed, as seen in Fig. 3.5. The case firmly held the microcontroller in place, protecting it from sweat and heat from the subject, and helped all the wires stay connected to the microcontroller. The wires that connected to the gauges traveled from the top of the microcontroller down into the shoe, under the insole, and attached to the four gauges embedded in the insole.

3.4.2 Testing

Before testing could begin on human subjects an Institutional Review Board (IRB) was applied for from Brigham Young University. The IRB passed the insole sensing system to be used on consenting, healthy adults. The only remuneration for each subject was a gift card. A further

Figure 3.5: The sensing insole system attached to the leg of a subject.

benefit was to both subjects and society in general was possible deeper understanding of healthy biomechanics which could lead to future development of assistive or rehabilitative technologies. The IRB was valid from April 2, 2013 through April 1, 2014. Testing began on April 23, 2013 and ended on June 28, 2013, within the approval range of the IRB.

The first tests performed with the sensing insole used a system with only one strain gauge. The strain gauge was placed under the pad of the foot in the sagittal plane (Fig. 3.6). In order to test if the gauge placed in that position and orientation would record distinctive, unique signals, three human adults (two with normal gaits and one with an abnormal gait) tested the device.

All of the subjects performed the same test so the information for the tests would correlate. In the test each subject wore a shoe with the sensing insole embedded and the Belkin armband around their ankle. The subject was instructed to walk straight for 30 seconds while a volunteer timed and monitored the subject. The simple test was completed once for each subject and the data was recorded and analyzed.

After the one gauge system seemed to give useful feedback, expansion on the design was desired. The first expansion was to create the sensing insole using four gauges to get more data

Figure 3.6: The sensing insole system with one strain gauge.

from the subjects. After designing and manufacturing the sensing insole, the next battery of tests were undertaken.

To test the new insole, a single subject was used in all the tests. While in the controlled gym environment the subject performed four different types of exercises: walking, jogging, biking, and climbing stairs. The subject walked and jogged on a treadmill, biked on a stationary bicycle, and climbed stairs on a stair climbing machine. Once the subject finished the tests, the signals from the four gauges were weighted by the distances of the signals from the center of the insole. This data was used to determine the "center of pressure" of the subject's foot.

Next, the data was analyzed to see whether it was possible to recognize what type of exercise a person was doing. In order to learn the distinct characteristics of the different exercises, the subject wore the sole and calibrated the sole to four activities: walking, jogging, biking, and climbing stairs. Calibrating the insole required the subject to perform each test individually. The subject went to the gym and performed 30 second tests walking at 3 mph, jogging at 6 mph, biking at approximately 72 rotations per minute (rpm), and climbing stairs at 75 steps per minute.

The sensing insole system was then tested initially to determine whether it could tell the difference between walking and jogging. In this test, the subject wore the sensing insole and alternated walking and jogging for 2 minutes. First he walked for 30 seconds, and then jogged for 30 seconds, walked for 30 more seconds, and lastly jogged for 30 seconds. This test was repeated a second time to ensure that good data was gathered and to compare between two separate tests.

Then the sensing insole setup was tested against all four exercises chosen. The subject went to the gym and performed each test for 45 seconds before moving on to the next. The subject first walked, then jogged, biked, and lastly climbed stairs.

Finally, four different subjects were individually outfitted with the sensing insole and taken to the gym to perform all four exercises. The subjects performed each exercise for 3 minutes, starting a new test each time they started a different exercise. The subjects first walked, then jogged, biked, and climbed stairs. The four individual tests were then compiled together to create a single dataset.

3.4.3 Software

For the sensing insole to recognize different exercises it needed to have a sample of each exercise. After the data was collected it was scaled so that the different exercises could be tested together. Each exercise was scaled from 0 to 10,000 every five seconds. For each exercise a representative cycle was determined from the calibration test data (the walking signal is pictured in Fig. 3.7(a)). Calibration was found to improve by considering a full representative cycle plus an additional 50 ms from the signal, with the exception of the jogging signal, where 100 ms were added.

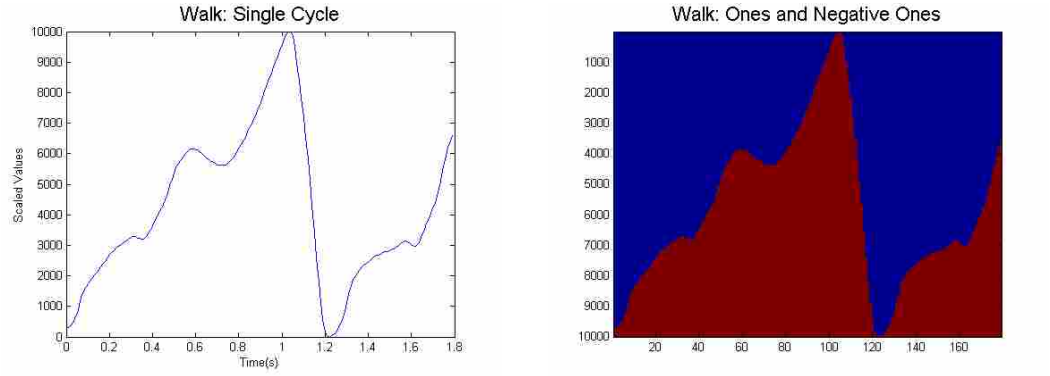

Figure 3.7: Normalized representative cycle from strain gauge results when attached to the bottom of the insole while the subject walks at a steady pace.

The cycle was converted to a matrix of ones (under the curve) and negative ones (Fig. 3.7(b)). Since the testing frequency was 100 Hz there was a data point every 0.01 s. Since the data was scaled between 0 and 10,000, the size of the matrix was 10,000 data points by the length.

This process was repeated with the full length test signals (Appendix E). The local minima of the test signals were found and used as starting points to compare each cycle with the representative cycle.

The matrix from a given test cycle was multiplied, point-by-point, with the representative cycle matrix. The result was then summed to find a percent match, or the percent of points from the representative cycle that match the corresponding points from the full test.

This process was repeated with each representative cycle at each local minimum and using the representative cycles from each of the different exercises. To determine which exercise the subject was most likely undertaking, the percent matches for each exercise type were compared.

A threshold of an 84.5% match was determined to give the best results in the walk-jog-walk-jog tests. This meant that if the representative cycle matches any test above the 84.5% threshold, the exercise with the highest percent match was taken. For the four exercises in the gym however, an 85% cutoff frequency gave much better results. Once the program determined that a certain exercise was being performed, it assumed that the subject continued doing that exercise until a new exercise was matched above the threshold, and higher than any other exercise.

3.4.4 Calories Burned

After matching the subject's exercises the program determined the approximate amount of calories burned during all the exercises. Each exercise was assigned a number of calories according to the subject's weight: Walking was set at 300 calories/hour, jogging was 600 calories/hour, bicycling was 780 calories/hour, with climbing the stairs at 465 calories/hour [70, 71]. As all subjects were within 30 pounds, the same caloric output per hour was used for all subjects. The program took the amount of time that the subject did each exercise to determine approximately how many calories that person has burned. For instance, if the subject walked for 20 minutes the amount of calories burned would be determined as follows:

$$20min. * \frac{1hour}{60min.} * \frac{300cal.}{1hour} = 100calories \tag{3.3}$$

The sensing insole was initially tested using only walk and jog signals. After the walk-jog-walk-jog test, described above, was performed the data from the test was converted to ones and negative ones by the match program. The match program then identified when the person was walking and jogging and the match program results and the actual results were compared. Next, the sensing insole was tested using all four original exercises (also described above), and lastly four other subjects performed the same four original exercises.

3.5 Results and Discussion

The results of characterization of walking motion from the two subjects with normal gait are shown below in Fig. 3.8. It was clear that the two signals were similar. The difference in the voltage readings stems from the fact that the two subjects performed their tests on different occasions and in between the tests the resistor in the voltage divider was replaced to increase the voltage response. The data from the subjects' tests were left in the raw data form that the microcontroller received so it could be compared. The shapes of the first two subjects' walking gait pattern started out low, rose to a higher level and then dipped down before spiking up. This process was then repeated during the next cycle. Normal Gait 1 had a three cycle span of 5.83 seconds and Normal Gait 2 had a span of 5.32 seconds. These two different subjects, when told to walk at a normal pace walked within 0.2 seconds of each other per cycle. The tests had similarities that ran throughout while each was also unique.

A third subject was tested to determine how similar the signal would be between two people with normal gait and one person with abnormal gait (Fig. 3.9). The abnormal gait subject used two canes to facilitate walking. The subject's three cycle time was 9.63 seconds. Beyond the increased time, the overall shape of the signal for the subject was quite different. The biggest difference was that the spike lasts much longer than in the subjects with normal gait, both in time duration and in the percentage of the overall cycle. Also, the signal only had two distinct levels, at 1.1V and 1.25V while the normal gait subjects had three levels.

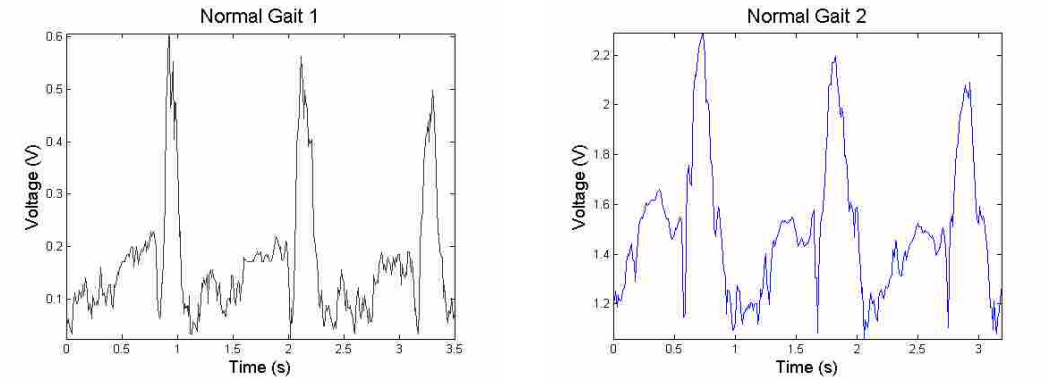

Figure 3.8: Gait signals from two subjects walking at a normal pace.

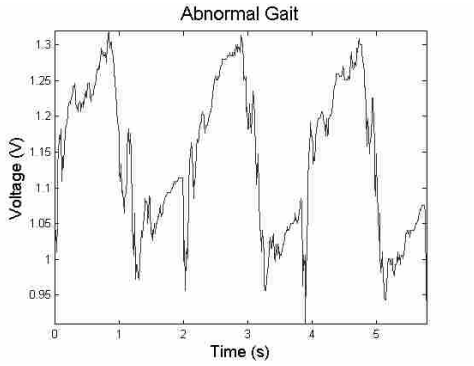

Figure 3.9: The abnormal gait pattern measured by the sensing insole.

The distinct difference between the signals in the two subjects with normal gait and the subject with abnormal gait demonstrated the sensing insole's ability to distinguish different types of gait.

To facilitate more in-depth testing, the one sensor insole was expanded into the four sensor insole, as mentioned in Subsection 3.4.1. The sensing insole, in combination with post-processing, measured the center of pressure (Fig. 3.10). Post-processes estimated the center of pressure by taking the distance from the approximate center of the foot to each gauge multiplied by the signal coming from the gauge. This worked fairly well whenever the subject's foot was in contact with the ground. In exercises where the subject's foot was in the air for extended periods of time points would collect in the center of the insole. Each graph shows the center of pressure of a subject performing different types of activities. The dark blue dots represent areas where the center of

pressure resided for only a moment with a sliding scale to the red dots which represent places where the center of pressure resided at for the longer periods of time.

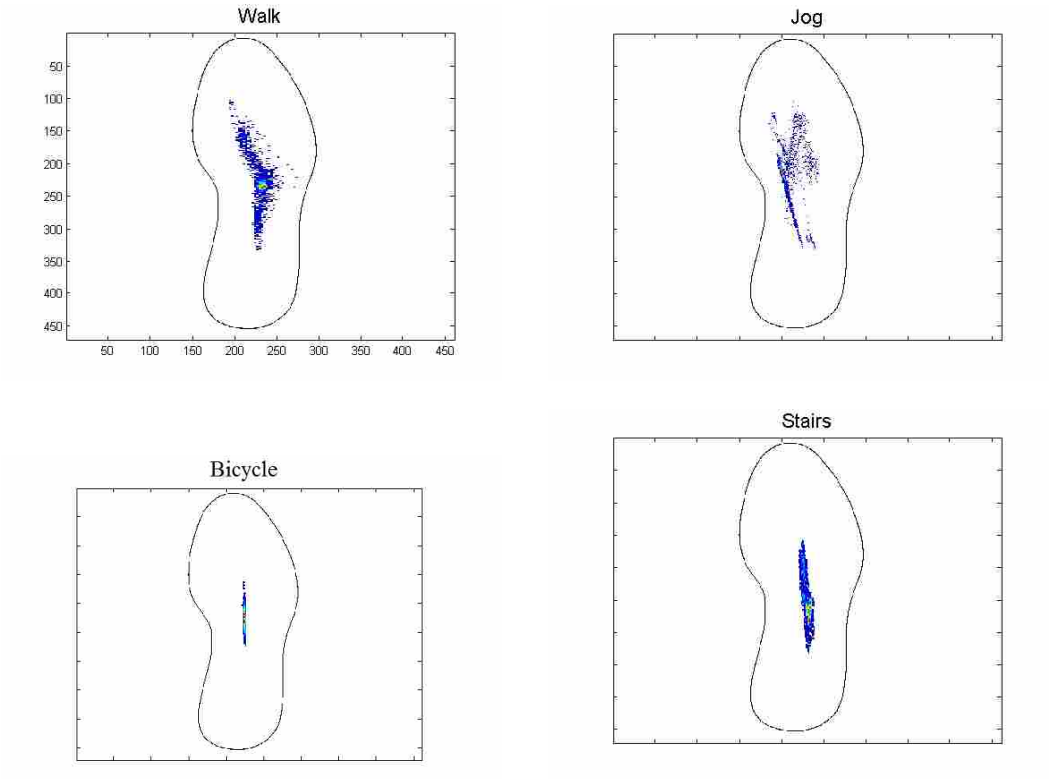

Figure 3.10: The centers of pressure for the four different exercises. The red areas represent higher density areas.

The walking signal traveled from the back of the foot to the front of the foot with a relatively tight grouping. This was what was expected since walking was done in a controlled environment where it was assumed the subject was walking steadily.

The center of pressure for the jogging signal was more scattered than any other signal. This seemed to indicate that while jogging the subject's balance required more sophisticated control via the motion of the foot.

The bicycle signal was by far the most compact signal. This was unsurprising since the subject rode a stationary bicycle, meaning that the center of the foot was pressed against the pedal continuously so that most of the pressure stayed near the center of the foot. The center of pressure

was also shifted toward the front of the foot, and as the subject pedaled with the balls of his feet more pressure was being applied to the front, corroborating the data.

The stairs signal was very compact, similar to the bike signal. It might seem that the center of pressures should generally be more toward the front of the foot since only the top half of the foot was contacting the stairs while climbing, but the data indicates significant flexing of the rear of the foot to maintain a fairly central overall "pressure".

The match program analyzed the data by matching the representative cycles with the gathered information. The results from the walk-jog-walk-jog test data run through the match program is shown below in Fig. 3.11. The graph on the left represents the results of the real test where blue was when the subject was walking and green was when the subject was jogging. The graph on the right displays the results from the match program identifying which exercise the subject was doing and when the subject changed exercises.

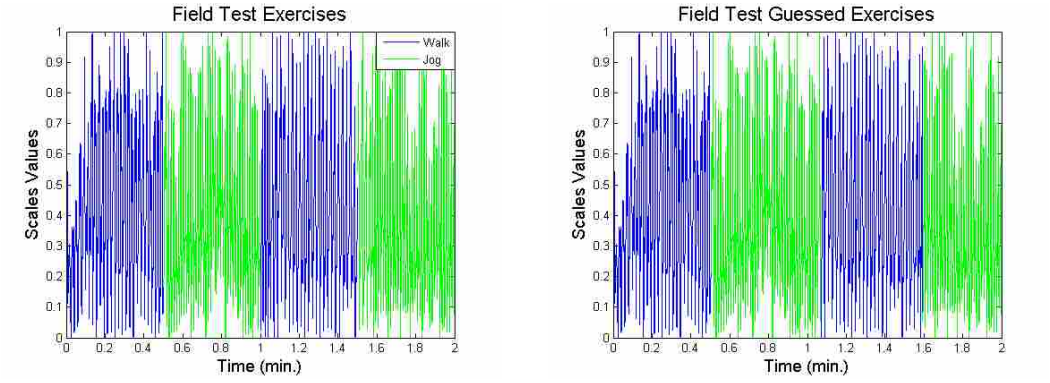

Figure 3.11: Field test exercises that the subject performed. The graph on the left represents the actual change in exercises while the graph on the right represents the sensing insole analysis.

The subject walked from 0 to 0.5 minutes, started jogging at 30 seconds or 0.5 minutes, began to walk at 60 seconds or 1 minute, and jogged at 90 seconds or 1.5 minutes. The match program recognized walking at 3.58 seconds, jogging at 30.21 seconds, walking at 64.18 seconds and jogging at 95.64 seconds. It took on average 3.40 seconds for the match program to change exercises after the subject changed the exercise. Using the time that the program took to correctly make the change from one exercise to the next, the program had a time error of 8.36%.

After analyzing what exercise the subject was doing, the match program calculated how many calories the subject burned, according to the method described in Section 3.4.4. The number of calories burned during the actual test, according to the method, was 15. The match program calculated that the subject burned 14.865 calories. Therefore, the error between the amount of calories burned and the match program value was 0.9%.

The amount of error in the determining how many calories the subject burns was much closer than the time error in the identified exercise. For instance, the program takes a few seconds to recognize the change from one exercise to the next so error that was induced when the program was late transferring from walking to jogging, was negated by the late transfer back from jogging to walking. Also, another reason that the caloric error was so low was because both walking and jogging burn calories and even through jogging burns calories faster, when the system incorrectly picks the exercise for a short amount of time only a small amount of error was introduced to the overall caloric total.

One other important aspect to note was that once the match program sensed a change from walking to jogging or vice versa it didn't change back until the subject changed exercises. There are no false positives in the program when determining walk and jog signals. This meant that the longer a test subject did each exercise, or the less the subject switches exercises, the less error there was be in the results.

The match program was able to match the walking and jogging signals very consistently. This was seen repeatedly with this test as well as others. This was a very good start for the system as this would be sufficient for many athletes who train outside of the gym. Deciphering between walking and jogging would be enough for the athletes that solely train for marathons and for new exercisers who only perform those two exercises. Also, this could be useful for the person who works out in the gym and wants to continue to count calories from one exercise to the next.

It was hypothesized that the match program worked so well with walking and jogging because there was an obvious difference in the two signals as the jogging signal was faster than the walking signal (Fig. 3.11). The match program didn't explicitly use the frequency of the signal to determine the difference between walking and jogging, but it did not normalize each cycle in the time dimension, hence the difference in frequency resulted in a significant difference in the representative cycle.

The subject next performed the four gym exercises; walking, jogging, climbing stairs, and biking, to test whether the match program could tell the difference in all the exercises. This was more difficult than tests that include only walking and jogging since there were four signals, some of which are similar, to match against the tested signal. The results are depicted in Fig. 3.12.

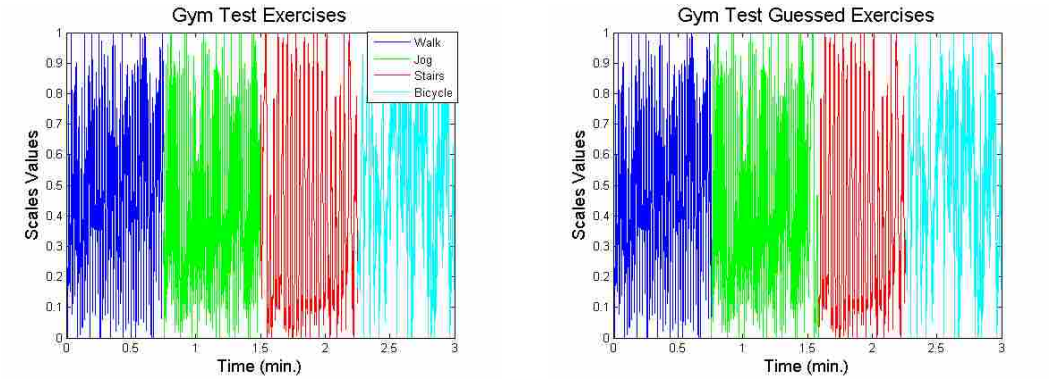

Figure 3.12: Gym test exercises that the subject performed. The graph on the left represents the actual change in exercises while the graph on the right represents the sensing insole's identified exercise.

The subject starting out walking and it only took the match program 0.44 seconds before identifying that the subject was walking. This first identification lag didn't induce error into the system, however, because the match program assumes that the first match, walking, was the exercise performed from the beginning. It then took 0.54 seconds to recognize that the subject started jogging. After the subject started climbing the stairs the program took 5.33 seconds to recognize the change in signal. Lastly, the match program found that the subject started biking 0.45 seconds after the subject actually started. All together, the match program assumed the wrong exercise a total of 6.32s, giving an error of 3.51%. The subject burned 29.5 calories, according to the method in Section 3.4.4, from doing all these exercises for this amount of time and the match program calculated that the subject burned 29.33 calories; an error of 0.577%.

The match program was very accurate when comparing the subject's original exercise cycles to different tests performed by the subject. This immediately meant that a person desiring to use the sensing insole could calibrate the sole to the exercises they want and then use the sole to determine exercises and count calories. It would be much more useful, however, if the match program could match exercises from many different people.

The sensing insole was tested with the four secondary subjects to determine if the patterns established previously would hold when the measurement system and match program were applied to different people. The sensing insole successfully matched walking, jogging, and climbing stairs using the first subject's signal and applying it to three of the four secondary subjects. Fig. 3.13 depicts a sample graph from one of the three subjects, Fig. 3.13(a), and then what the match program identified the subject was doing, 3.13(b). Unfortunately, the gauges' resistances increased over time due to stress-relaxation in the strain gauges and by the time that the fourth subject performed the tests the gauges had unreadably high resistances so the necessary data was not recorded.

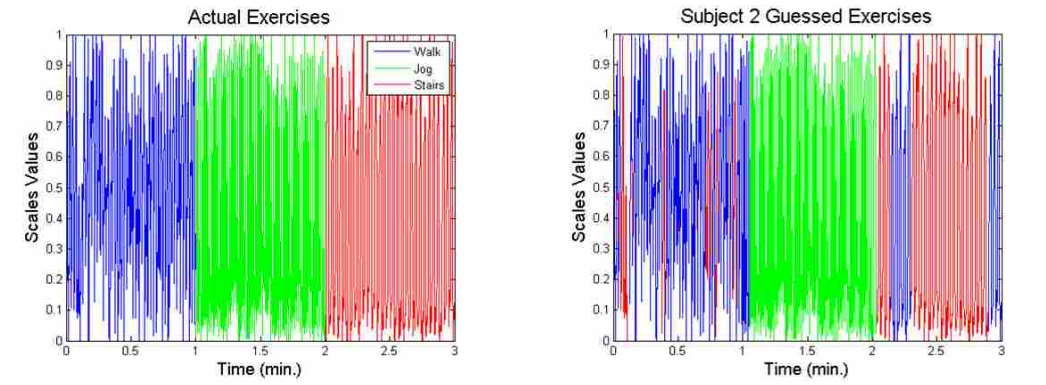

Figure 3.13: One of the gym test results that the three subjects performed. The graph on the left represents the actual change in exercises while the graph on the right represents the match program's identified exercises.

The following results were from all three subjects that performed the tests with the working sensing insole measuring their exercises. Using the method in Subsecton 3.4.4, the match program identified the different exercises with 4.85% error, 3.59% error, and 1.03% error in calories and 12.16% error, 15.05% error, and 1.03% error in the time, respectively, for subjects 1, 2, and 3. The caloric average error for the tests was 2.08% and the average time error was 9.41%. For the first time the match program incorrectly matched exercises in the middle of other exercises. While the subject was walking the match program identified some cycles were climbing stairs and vice versa. Even with the incorrect matching in the program the caloric error for the three tests are all below 5%.

Testing the bicycling measurements proved the most challenging. The first subject's biking signal matched infrequently with the subsequent subjects' signals and also had incorrect matches. The unfiltered biking signals for the four signals are shown below in Fig. 3.14. The biking signals were consistent in their inconsistency and lack of pattern. Also, the centers of pressure were consistent within the biking realm. While the other exercises have a repetitive pattern this was nonexistent in the biking signal and so could be used to determine when the subject was biking if only these four exercises were undertaken. While the current match program didn't match biking signals a more robust matching program that considers the tight grouping of the centers of pressure for biking or machine learning techniques should be able to match the signals.

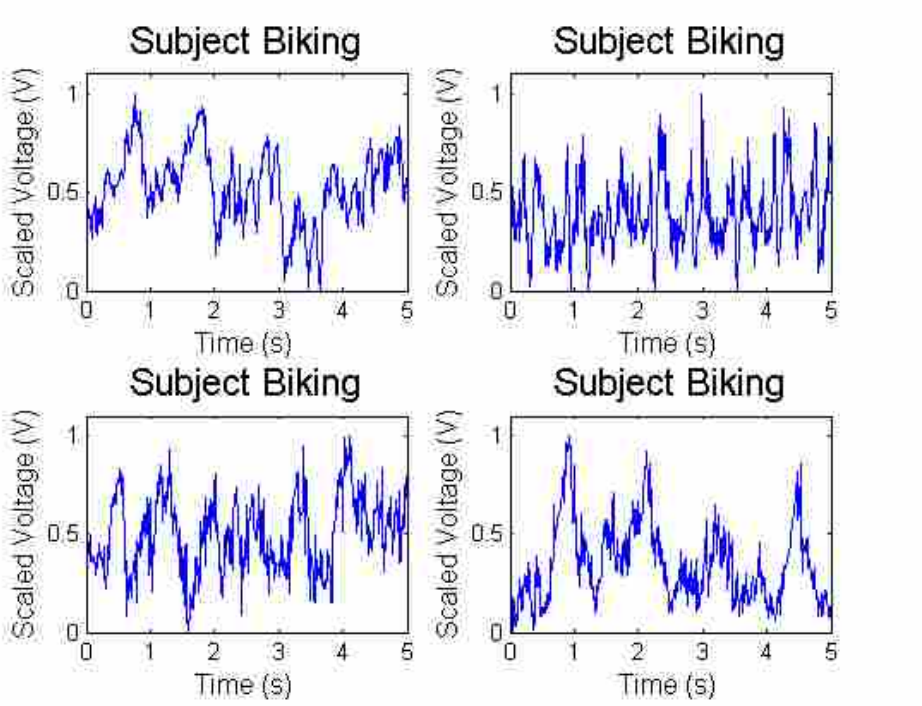

Figure 3.14: Excerpts from the biking tests for the four different subjects.

The current match program assumed that the person was continually exercising. For instance, if a person was walking and stopped the match program would use the previous matched signal, walking, and assume that the person was doing that exercise. This method was used because the signals match only once every several cycles, therefore the program needs to make assumptions about the person's exercise in between the cycles that match. This worked well when the subject

was using the sensing insole while exercising but would be less effective if the subject continually took breaks during testing.

3.6 Conclusions and Future Work

The present work successfully developed a suitable sensor and insole, which is portable and compact that successfully measured different signals when subjects performed different exercises. Center of pressure patterns were clearly different for each exercise. The match program used to determine the different exercises was fairly primitive but reliably picked the difference between walking, jogging, and climbing stairs. It was supposed that a professional in gait research could reliably use these sensors to further understand gait when subjects act in more natural environments.

The sensing insole was very good at matching the four different exercises when the person that was being matched had been calibrated for the exercises. The match program also determined the difference between walking and jogging, whether the person using the sole had been calibrated to the insole or not.

To increase the accuracy of the sensing insole the match program could look at the different exercises on a per sensor basis as opposed to the composite signal that the program used. Using the per sensor basis would require more computing power as more data would be scaled and matched; therefore, it would desirable to streamline the match program to increase the speed of matching.

Increasing the accuracy of the match program may be done in other ways as well. The chosen matching method was tested and used because it worked to pair the different exercises. However, there were countless other ways to match two signals that have not been explored due to time. One such method was looking at the Fourier Transforms for the different signals. Matching the Fourier Transforms may more correctly match the different exercises together. The single cycles may also be cross correlated to the test signal to determine the likelihood that the two are same exercises.

The sensing insole needed to be as compact as possible. This could be further accomplished by replacing the microcontroller with preprogrammed microcontrollers available on the market that were much smaller. The smaller microcontroller would be able to communicate to electronic devices wirelessly, evaluate in real-time, and send updates to the person about how long they have been doing different exercises and how many calories they've burned.

CHAPTER 4. OVERALL CONCLUSIONS AND FUTURE WORK

The present work focused on developing improved NCSGs, furthering NCSG manufacturing techniques and understanding, and creating the required tools and hardware for real-life applications. Previous manufacturing processes were studied and enhanced. FEA modeling combined with analytical math models were used to gather information about how NiNS surrounded by Sylgard reacts to strain, at the core of the NCSG performance. An application, specifically the gait-sensing insole, was developed to show the versatility and usefulness of the gauges.

The first portion of the research was dedicated to improving the properties and readings from the NCSG. The improved manufacturing process created gauges that were thinner, more flexible, and more consistent. The drift effects seen in previous NCSGs was mitigated. The NCSGs were incorporated to be setup into a portable system for mobile applications of biomechanical sensing. Finally, a proof-of-concept proved the NCSG work in real-life applications.

The manufacturing process was changed in several steps, most significantly from a pouring method to a mold method, and greatly increased all the desirable attributes of the gauges. The drift effects were singled out and studied and changes were made to greatly reduce the drift in the system, although some still remains. The system was made extremely portable with the use of microcontrollers and an insole system was setup to portably measure gait on one subject.

The task was devoted to advancing previously produced models that calculated junction distances between NiNs when strained. The goal of developing the model began with developing a FEA model that could use fewer unproven assumptions than previous methods. A mathematical model was produced to corroborate results with the FEA. The NiN placement was then expanded to test many different orientations.

The FEA model eliminated many unproven assumptions from previous models. An analytical model based upon two simple test cases were used to build confidence in the FEA. The FEA was then expanded to a representation of all possible junction and NiN orientations. The FEA was

then compared to the previous model, showing similar trends, but in a much more powerful model. Application of the model to predict piezoresistive response of the NCSG was not part of the scope of this work.

Finally, the insole gait-sensing system was to be tested in much more robust conditions. The gait-sensing system was tested using different subjects with normal and abnormal gaits to determine if the system gave distinct results for two very different types of gait. Next, different exercises were characterized and matched using a newly developed match program, starting out with just walking and jogging but then expanding to further applications in a normal exercise environment. Finally, the insole system was to be tested with several different people to determine if the insole was applicable over a large range.

The insole gait-sensing system was developed using four gauges with two under the pad of the foot, one under the arch, and the last one under the heel. The insole system was tested using subjects with normal and abnormal gait and large differences were seen between the two. The matching program was then developed and successfully matched four exercises: walking, jogging, bicycling, and climbing stairs for one subject. Lastly, the match program labeled walking, jogging and climbing stairs correctly for different subjects, though it had problems recognizing biking.

The present research quickly advanced high deflection strain gauges beyond the realm of hope and thought and into possibility. Even though great strides have taken place in this research there remains a higher potential that can be reached using NCSGs as strain sensors. The beginning of this paper talked about the lack of viable, portable high deflection strain gauges. This work proved the use of portable strain gauges measuring large deformations but further work into the high deflection realm is feasible and could greatly expand the use of high deflection strain gauges beyond what is known today. The possible applications for these promising NCSGs are limited more by the imagination of the inventor than the abilities of the materials.

REFERENCES

[1] http://news.psu.edu, 2012. Americans fall short of federal exercise recommendations, May. 1

[2] Martinez, J., 2012. Americans exercise about 2 hours a week, fall short of the recommended 4, May. 1

[3] wiseGEEK, 2013. What is a strain gauge?

[4] Perry, C., 1984. "The resistance strain gage revisited." *Experimental Mechanics,* **24**, pp. 286–299. 2

[5] http://www.omega.com, 2013. A historical perspective. 2

[6] Shelton, J. R. O. J. C., 1997. *Optical Measurement Methods In Biomechanics*. No. 168. Chapman & Hall. 3

[7] Thornton, S. B. W. H. B. M. N. N. S. G., 2007. *Biomechanics of the Musculo-Skeletal System*. Wiley. 3

[8] Contributers, 2012. Digital image correlation. 3

[9] Mundermann, L., Corazza, S., and Andriacchi, T., 2006. "The evolution of methods for the capture of human movement leading to markerless motion capture for biomechanical applications." *Journal of NeuroEngineering and Rehabilitation,* **3**(1), p. 6. 4

[10] Hansen, O. J. C. G. D. S. N. M. A. D. P. R. D. F. G., 2011. "Multiscale model for extreme piezoresistivity in silicone/nickel nanostrand nanocomposites." *Metallurgical and Materials Transactions A: Physical Metallurgy and Materials Science,* **42**, pp. 3898–3906. 4, 40, 56, 59

[11] Hansen, O. K. J. G. C. K. T. A. M. D. T. F. G., 2011. "Optimization of nickel nanocomposite for large strain sensing applications." *Sensors and Actuators A: Physical,* **166**, pp. 40–47. 4

[12] Johnson, O. Fullwood, D., 2010. "A percolation/quantum tunneling model for the unique behavior of multifunctional silicone/nickel nanostrand nanocomposites." *TIB*. 4

[13] Hyatt, T., 2010. "Nano-composite sensors for wide range measurement of ligament strain." In *Proceedings of the SEM Annual Conf & Exposition on Experimenal and Applied Mechanics*. 4, 28, 66

[14] Calkins, T., 2011. "Nano-composite high displacement strain gauges for use in human-machine interfaces: Applications in hand pose determination." Master's thesis, Brigham Young University. 4, 8, 28, 65

[15] Koecher, M., 2012. "Evaluation of advanced conductive nickel materials for strain sensing in carbon fiber reinforced polymers." Master's thesis, Brigham Young University. 4

[16] http://conductivecomposites.com, 2011. Nanostrands. 6, 65

[17] Ghose, S., Watson, K., Working, D., Connell, J., Jr., J. S., Lin, Y., and Sun, Y., 2007. "Thermal conductivity of copoly(ethylene vinyl acetate)/nano-filler blends." In *SAMPE 2007 Symposium and Exhibition - Baltimore, MD*. 7, 65

[18] http://conductivecomposites.com, 2011. Nickel chemical vapor deposition (cvd) coated fibers. 7

[19] Lavine, F. I. D. D. T. B. A., 2007. *Fundamentals of Heat and Mass Transfer*. John Wiley & Sons Inc. 15

[20] http://www.accessdata.fda.gov, 2012. Cfr - code of federal regulations title 21, April. 20, 23

[21] Lurie, P., Wolfe, S., et al., 1997. "Unethical trials of interventions to reduce perinatal transmission of the human immunodeficiency virus in developing countries." *New England Journal of Medicine-Unbound Volume,* **337**, pp. 853–856. 20

[22] Marsden, S., and Melander, M., 2001. Historical cases of unethical research, January. 20

[23] Annas, G., and Grodin, M., 1995. *The Nazi doctors and the Nuremberg Code: human rights in human experimentation.*. Oxford University Press, USA. 20

[24] Meike, R., 2012. Arduino uno vs beaglebone vs raspberry pi, October. 21, 67

[25] http://www.astro med.com Frequently asked questions. 21

[26] Grusin, M., 2010. Voltage divider, November. 22, 23

[27] Aiken, R., 1999. The voltage divider rule. 22

[28] http://www.research.uci.edu, 2011. Activities that require irb review, October. 23

[29] Iqbal, Y., 2013. How to detect and treat gait disorders. 24, 63

[30] Lam, R., 2011. Office management of gait disorders in the elderly, July. 24

[31] http://www.nationalmssociety.org Walking (gait), balance, & coordination problems. 27, 64

[32] http://www.ncbi.nlm.nih.gov, 2011. Parkinson's disease, September. 27, 64

[33] http://sine.ni.com, 2012. Ni 9219. 27

[34] Instron, 2011. Instron: The difference is measurable. 825 University Ave, Norwood, MA 02062-2643, USA. 34

[35] Hyatt, T., 2010. "Piezoresistive nano-composites, characterization and applications." Master's thesis, Brigham Young University. 35

[36] Arnold, D., 2012. Spherical coordinates in matlab, September. 41

[37] http://www.sandia.gov, 2010. Sylgard 184: Young's modulus at 23c. 42, 53

[38] Fellner, F. S. T., Wilde, J., and Wallrabe, U., 2008. "Mechanical properties of silicones for mems." *Journal of Micromechanics and Microengineering,* **18**, p. 1. 42, 53

[39] http://www.ecourses.ou.edu, 2013. Mechanics - theory. 47

[40] http://www.ansys.com, 2013. Ansys.

[41] Kopeliovich, D., 2012. Commercially pure nickel 200, June. 53

[42] http://www.nickel alloys.net Nickel - properties, fabrication and applications of commercially pure nickel. 53

[43] Thompson, D., 2002. Conventions for naming parts of the gait cycle, March. 61

[44] Sutherland, D. H., 2001. "The evolution of clnical gait analysis part 1: kinesiological emg." *Gait & Posture,* **14**, pp. 61–70. 62

[45] Sutherland, D. H., 2001. "The evolution of clinical gait analysis part ii: Kinematics." *Gait & Posture,* **16**, pp. 149–179. 62

[46] Sutherland, D. H., 2005. "The evolution of clinical gait analysis part iii - kinematics and energy assessment." *Gait & Posture,* **21**, pp. 447–461. 62

[47] Baker, R., 2007. "The history of gait analysis before the advent of modern computers." *Gait & Posture,* **26**, pp. 331–342. 62

[48] Feng, W. T. T. L. R. Z. H., 2012. "Gait analysis using wearable sensors." *Sensors,* **12**, pp. 2255–2283. 62, 63

[49] Shibata, T. L. Y. I. K., 2009. "A wearable force plate system to successively measure multi-axial ground reaction force for gait analysis." In *Proceedings of the 2009 international conference on Robotics and biomimetics*. 63

[50] Howell, A. M., 2012. "Insole-based gait analysis." Master's thesis, University of Utah. 63

[51] Schmidt, M. G., 2013. "Real-time feedback methods for gait rehabilitation through a mobile platform." Master's thesis, University of Utah. 63

[52] Paradiso, S. J. M. B. A. Y. B. D. M. S. D. E. K. J. A., 2008. "Gait analysis using a shoe-integrated wireless sensor system." *IEEE Transactions on Information Technology in Biomedicine,* **12**, pp. 413–423. 63

[53] Wheeler, L. L. S. B. E. P. C. S. J., 2012. "An elastomeric insole for 3-axis ground reaction force measurement." In *The Fourth IEE RAS/EMBS Inernational Conference on Biomedical Robotics and Biomechatronics*. 63

[54] Lincoln, L. M. Q. B. R. C. S. J. W., 2012. "An optical 3d force sensor for biomedical devices." In *The Fourth IEEE RAS/EMBS Internation Conference on Biomedical Robotics and Biomechatronics*. 63

[55] Sabut, S. K. Kumar, R. M. M., 2010. "Design of an insole embedded foot pressure sensor controlled fes system for foot drop in stroke patients." In *Systems in Medicine and Biology (ICSMB), 2010 International Conference on*. 63

[56] Paulick, Peyton Djalilian, H. B. M., 2011. *StabilitySole: Embedded Sensor Insole for Balance and Gait Monitoring*. Springer Berlin Heidelberg. 63

[57] http://www.getsmartmove.com, 2013. Accuracy you can trust. 63

[58] Belluck, P., 2012. Footprints to cognitive decline and alzheimer's are seen in gait. 64

[59] http://techtransfer.universityofcalifornia.edu, 2008. Measuring human gait and movement for diagnosis of neurological disease. 64

[60] Pete B. Shull Kristen, L. L. M. C. T. B., 2011. "Training multi-parameter gaits to reduce the knee adduction moment with data-driven models and haptic feedback." *Journal of Biomechanics,* **44**, pp. 1605–1609. 64

[61] Wheeler, Jason Shull, P. B. B. T., 2011. "Real-time knee adduction moment feedback for gait retraining through visual and tactile displays." *Journal of Biomechanical Engineering,* **133**, pp. 1–5. 64

[62] http://www.outdoorfoundation.org, 2013. A special report on trail running. 64

[63] Chock, C., 2013. Indoor vs. outdoor running: 3 things to know about treadmill training. 64

[64] Quinn, E., 2011. Treadmill vs. outside running: Which is best? 64

[65] http://www.statisticbrain.com, 2012. Marathon running statistics. 64

[66] Brown, Thomas D. Sigal, L. N. G. O. N. N. M. S. R. J. B. R. A., 1986. "Dynamic performance characteristics of the liquid metal strain gage." *Journal of Biomechanics,* **19**, pp. 165–173. 66

[67] Chalker, J. T., 1988. "Percolation, quantum tunnelling and the integer hall effect." *Journal of Physics C-Solid State Physics,* **21**, pp. 2665–2679. 67

[68] Baxter, B. S. F. E. P. G. S. C., 2012. "Three-dimensional evolution of mechanical percolation in nanocomposites with random microstructures." *Probabilistic Engineering Mechanics,* **30**, pp. 1–8. 67

[69] LT, K. K. F. H. D., 2010. "A route for polymer nanocomposites with engineered electrical conductivity and percolation threshold." *Materials,* **3**, pp. 1089–1103. 67

[70] Samuels, D., 2013. Differences between bikes vs.treadmills vs. elliptical machines. 73

[71] http://www.nutristrategy.com, 2013. Calories burned chart. 73

[72] Collins, S., 2013. Super bowl rating dip slightly from last year, February. 93

[73] Dailey, K., 2012. Nfl safety: Is american football too violent?, September. 93

[74] Breslow, J. M., 2013. Nfl concussions: The 2012-13 season in review, February. 93

[75] Lopresti, M., 2013. Concussions in football: The president is watching, January. 93

[76] Dorsey, P., 2009. Helmet tech aimed at concussions, September. 93

[77] Sauser, B., 2008. Preventing concussions, February. 94

[78] http://jmmedical.com, 2013. Nitinol - shape memory. 94

[79] http://jmmedical.com, 2013. Transformation temperature hysteresis in nitinol alloys. 94

[80] KH, C. N. L. H. J. S. L. S. P., 2008. "Electroactive polymer actuator with high response speed through anisotropic surface roughening by plasma etching." *PubMED,* **8**, pp. 5385–8. 94

[81] http://www.factor2.com, 2008. A-3240 platinum silicone foam. 94, 96

[82] Higgins, M., 2009. Football physics: The anatomy of a hit, December. 94

[83] COLE, M. D. A., 2013. Are nfl football hits getting harder and more dangerous?, February. 94

[84] http://www.spraygunworld.com. The Challenge - Hvlp vs Conventional Spray Guns / Hvlp vs Air Assisted Airless / Turbine, LVLP, RP Etc. 91

[85] http://www.parasolinc.com, 2011. Guns: HVLP AS1009 (Paint Gun 9). 91

APPENDIX A. SPRAYING THE COMPOSITE MATERIAL

The first step taken to produce sprayed composite NCSGs was to buy a high volume low pressure (HVLP) spray gun. HVLP spray guns have low overspray, high finish quality, and allow for great control [84]. The HVLP gun purchased is from parasol Inc. and comes with a choice of a 15, 17, or 19 mm tip size [85]. The 19 mm tip size was chosen because it was the largest, which allows the largest particles to exit.

Testing of the HVLP and spraying process began by obtaining polyurethane as a base. The reason that this was the first tested base was because Conductive Composites has had success spraying NiNs using a polyurethane as a base. Also, since the developers of this spray system were slightly novice, proving that the NiNs could spray with a proven base was a good first step. The mixing process of the NiNs and the polyurethane was rather rudimentary. Both were mixed together using 10 % (by weight) NiNs. The polyurethane is an opaque lacquer but with the NiNs mixed into the polyurethane the resultant material was black with some white showing around the edges. As the polyurethane is near inviscid the NiNs had the tendency to sink to the bottom of the cup several minutes after the two were mixed. To ensure a good distribution of NiNs the solution was vigorously mixed by hand directly before filling the hopper of the HVLP gun.

The HVLP gun was hooked into the compressed air available in the laboratory. After loading the hopper and ensuring that the HVLP gun was level the polyurethane-NiN solution was successfully sprayed onto a plastic sheet.

Next, the NCCF was tested with the polyurethane to test whether the much larger NCCF would clog the spray gun. The NCCF were mixed with the polyurethane in the same amount as the NiNs. Once again, the solution was vigorously stirred directly before spraying the NiNs onto a separate plastic sheet. The NCCF was also successfully sprayed onto the plastic sheet.

Since both the NCCF and NiNs were sprayed separately it was assumed that they would be able to be sprayed together. Therefore, the next step was to make a batch of composite mix using

9 % NiNs and 1 % NCCF. The silicone is much more viscous than the polyurethane so using this method the gauges were impossible to spray. To decrease the viscosity of the gauges OS-20 or OS-30 was added in continually increasing amounts until the solution appeared inviscid enough to spray. This solution was then poured into the hopper of the HVLP gun and sprayed onto a plastic sheet. The first trial sprayed several globules of composite material onto the plastic sheet but then refused to spray. After inspecting the spray gun it was determined that a combination of NCCF and NiNs were clumped up and clogging the sprayer nozzle. This process was repeated several times but the materials continued to clog the spray gun.

Once it was determined that the current method would not work, several different aspects of the system were changed to increase the amount of composite material that the gun sprayed. Less NiNs and NCCF were tried in the gun and then when that didn't work NiNs alone were tried in the gun to see if they would spray. After these failed more material thinning solution was added to the solution to lower the viscosity even more than before.

When all these measures failed the HVLP gun was inspected more thoroughly. The gun has a pin such that when the trigger is released the pin fills the nozzle and doesn't allow spraying materials to exit the gun. When the trigger is pulled the pin is pulled back and forms a passageway for the material to exit the gun. After exiting the gun tip air from surrounding nozzles atomizes the material and sprays it outward. Since the NiNs and NCCF clogs were forming inside the gun, as opposed to at the nozzle, it was determined that the passageway formed by the retreating pin may be the limiting factor. To increase the passageway washers were inserted into the gun on the back end of the pin. The washers held the pin back so that it could not close all the way, although it did cut off the composite material from exiting when the trigger was fully released. When the trigger was pulled the pin retreated much farther into the gun, allowing a much larger passageway for the composite material to exit. This method was no more successful than the previous methods and at this time the spray method was disbanded and new methods were tested.

APPENDIX B. FOAM SENSING SYSTEM

B.1 Foam Sensors Motivation

The goals with the foam sensors was to create a foam sensor that was conductive replacing silicone polymer with a silicone foam and compressively strain the foam to prove that the foam reacts to strain.

The foam sensing system was developed due to the potentially incredibly powerful application. One example of a use of a conductive foam gauge is sensing force impacts on a helmet. The force impact on a foam material could potentially be related to the force applied to the head of a helmet wearer. Without the use of sensors in helmets, it has previously been difficult to pinpoint exactly how much protection is actually being provided to the wearer. This has stymied attempts to improve the design and thus the protective power of helmets.

One arena where people wear helmets for protection without knowing how much protection is really added is in the National Football League (NFL). In the NFL, players get paid millions of dollars a year to put their bodies on the line and entertain hundreds of millions of fans. The largest NFL sporting event, the Super Bowl, has drawn more than 100 million viewers each of the past three years [72]. Concussions have taken an increasingly large share of discussion and debate due to high concussion rates in the NFL and the possible life-long detrimental effects of concussions [73–75]. Concussions occur when a force to the head occurs in symphony with symptoms such as headaches, blurred vision, amnesia, etc. [76].

Concussion data could be gathered without risking athlete's health by inserting the conductive foam into a football helmet and instrumented to send a wireless signal to trainers and doctors on the sideline to monitor the force of impacts occurring to a player's head. This would help the trainers determine whether the player needs to exit the game to be tested for concussion symptoms as well as gather data on what types of hits and the forces of hits that result in concussions. Beyond helmets, the foam could also be placed in soles that attach to the bottom of an athlete's

cleat to measure their gait as they walk. A disrupted gait could lead to a more accurate concussion diagnosis method.

A third, and possibly most promising tool, is producing a helmet with the conductive foam embedded into a helmet with materials that have a variable stiffness; such as shape memory alloys and electroactive polymers. The conductive foam could be used to sense a force hitting the helmet and start a process where the foam in the helmet changes properties to better protect against the force of the blow, similar to recent work done at MIT [77]. There are several different types of materials where a microcontroller could provide this change in the properties of the materials. Shape memory alloys are intriguing because they return to a preset configuration when heated to a certain temperature [78]. One such alloy, Nitinol, changes properties in a range from 25 to 70 $^{\circ}C$, depending on composition [79]. Electroactive polymers are materials that change shape when a current is passed through them [80]. Electroactive polymers could be placed in the foam and change the properties of the foam as the helmet experiences different impact forces.

Because of the potentially powerful effect that a foam sensor could have in many different areas research began on foam sensors with embedded NiNs and NCCF. These gauges are similar to the NCSG, but with one major difference the base material is a foam instead of a flexible elastomer. The foam, manufactured by Factor II, Inc.,is named A-3240 Platinum Silicone Foam. It is a two component foam that cross-links at room temperature by an addition cure reaction. The foam is mixed at a 1:1 ratio and stirred vigorously at which time it expands to 6 to 7 times the original volume [81].

The conductive foam was tested using two different methods. The first method applied a compressive displacement in the Instron. The compressive displacement moved 0.08" traveling at approximately 0.6 in/sec. Though the change in displacement is only 0.08" the foam resulted in a change of approximately 1.36 x 10^5 grams (300 lbs.). This is on the lower end of the possible forces that occur on a helmet during physical impacts. In the NFL, helmet impacts are suspected to reach forces up to 8.16 x 10^5 (1,800 lbs.). [82, 83].

The Instron was instrumented to record the displacement in the system; an attached load cell measured the force needed to displace the foam, and an attached DAQ measured the change in resistance in the conductive foam due to the displacement. The change in resistance was found by attaching leads perpendicular to the compressive force, exciting 0.4 mA across the foam, and

measuring the voltage drop across the leads. The excitation and measurement were performed by the NI-9219 console.

These three applications of the use of a conductive foam are just the beginning. The conductive foam, like the NCSG, has a wide range of possibility to improve the lives of many people. More work is needed to refine the foam measurement system, but preliminary testing has yielded optimistic results that, if developed, the measurement system will become a powerful measuring device.

B.2 Foam Manufacturing Process

Since the foam measurement system is brand new the first step was to manufacture conductive and reliable gauges. These gauges needed to be spongy after production so that they can be compressed. The gauges also needed to be conductive to measure force in a similar manner as the NCSG. Lastly, testing needed to determine if the gauges would react to compressive forces. To develop and tests these aspects of the foam a manufacturing process was developed following the general steps followed to produce the NCSG.

The foam manufacturing process is quite a bit simpler than the elastic silicone manufacturing process. Part of this is due to the fact that the foam used, Platinum Silicone Foam, is a newly developed product that doesn't have the published density of the different parts. This means that only weight percentages are used to measure the constituent components together. A few different manufacturing processes have been developed and, as any one may turn out the best, all will be included in this paper as possible avenues of development.

Before manufacturing began, an Excel worksheet (Table B.1) was programmed to give weights of the constituent materials after inputting the desired weight percentages of the NCCF and NiNs and the weight of the foam for the trial. For instance, if the manufacturer wants 20% NiNs and 5% NCCF with 6 grams of foam he would measure out 3 grams of each type of foam 1.6 grams of NiNs and 0.4 grams of NCCF.

When a batch of foam product was produced, the first step was to open the worksheet (Table B.1) and input the desired weight of the two foam components together. Next, the desired weight percent of NiNs and NCCF was chosen. The table then generated the necessary weight measurements needed to create the foam. A clean plastic cup, suitable to be put into the Thinky

Table B.1: Foam weights (grams) of foam constituent materials.

Total Foam	6	grams
	Input (Percent)	Input (Grams)
NiNs	20	1.6
NCCF	5	0.40
Foam (Part 1)	38	3
Foam (Part 2)	38	3

centrifugal mixer, was placed on the Ohaus force sensor and the “Tare” button was pushed. One of the foam bases was measured into the cup and the cup was removed from the sensor. Next, an Aluminum cup was placed on the force sensor and the “Tare” button was pressed. The screened NiNs were then weighed into the cup and removed from the sensor. A second Aluminum cup was then placed on the sensor and the ”Tare” button was pushed. The recorded weight of NCCF was measured into the cup. The contents of the two Aluminum cups were then poured into the plastic Thinky cup.

The first tests were stirred with a metal rod until the NiNs and NCCF were wetted. Next the second foam base material was added to the conductive foam mix and the entire solution was stirred for 45 seconds, as directed in an explanatory video on the foam manufacturer’s website [81]. The foam was then placed on a flat plastic plate and allowed to set and foam up for 10 minutes. As this point the foam could be easily peeled off the plastic plate. This production method resulted in flat pieces of foam material that did not look or behave very much like a foam product.

Following this procedure, and varying the amount of NiNs and NCCF that were used, an ideal conductivity was found. The tests were conducted using weight percents instead of the before-mentioned volume percents for the NCSG. This is due to the volume of the foam being variable as it expands. The first tests tried were 20, 30, and 40 weight% NiNs without any NCCF. The 20% sample was not conductive but the 30% sample was conductive. A 25% sample was produced and was slightly spongy and slightly conductive. Although conductive the sample was not spongy enough. Due to the product not being too hard and having a low volume a new production method was tested.

It was proposed that the female molds previously used to create NCSG’s should be used to form the foam material. For this change, the complete stirred conductive foam mixture was

scooped into one female mold crevice. Two identical female molds were placed together with crevices lining up to double the size of the cavity that the foam was to fill. The two molds together create a foam approximately 1/8" thick, as pictured in Fig. 9 (right). The first trials that used this method looked much more regular than the foam poured onto the flat plate and were much easier to compress, resulting in a more "foamy" substance.

The weight percentages of the NiNs and the NCCF were increased while holding the density approximately constant to try to create a soft feeling foam that was also conductive. Batches with NiNs ranging from 20% to 40% were tried but none were conductive. It is not clear why the foam gauges that were poured onto the flat plate were conductive with 30% NiNs while the molded foam gauges were not conductive even with 40% NiNs. The next step was to increase the density of the foam by adding more foam base while holding the volume constant as well as adding NCCF to increase conductivity. Increasing the foam base from 6 grams to 8 grams gave us good readings using the 25% NiNs and 1% NCCF, giving an output of approximately 20-100 Ω using the Ohmmeter. This foam is conductive but not optimized for "foaminess." For the current research a conductive foam that measures compressive changes is the goal, therefore the conductive foam that is not optimally "foam" is still sufficient.

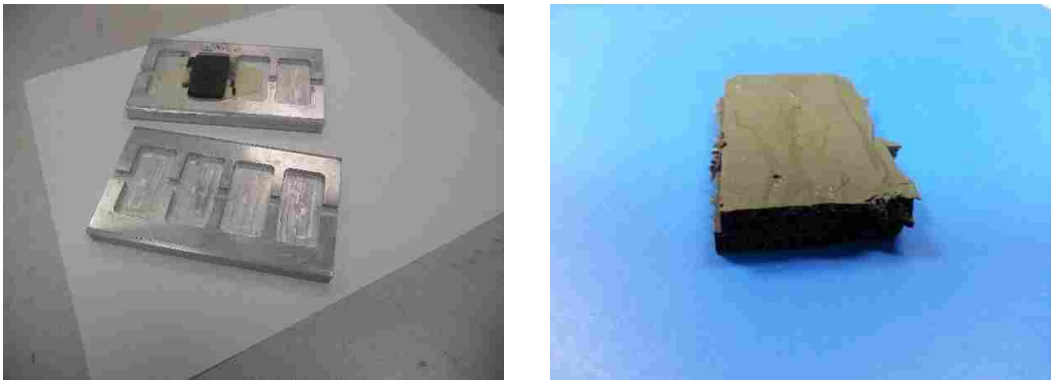

Figure B.1: The two female molds after the foam is expanded and the molds are separated (left). The conductive foam made from using two female molds pressed together (right).

B.3 Results and Discussion

The results from the impact test are seen in Fig. B.2. The resistance response from the first change in the resistance response to the peak was 0.15 seconds. The magnitude of the resistance response from the first change in displacement to the maximum resistance was 144.5 Ω and occurs in 0.17s. The displacement measured by the Instron occurs over 0.11s. The delay between the movement of the Instron and the first change in the resistance measurement was 0.04s. The delay from the beginning resistance of 211.5 Ω to the measurable change of 285.5 Ω was 0.04s (chosen because the resistance leveled off here for approximately 0.1s) and the delay from the beginning of the movement of the Instron was 0.06s. This extremely fast response may be used instead of the slower peak to peak time mentioned earlier to increase the response time and effectiveness of a reactionary pad.

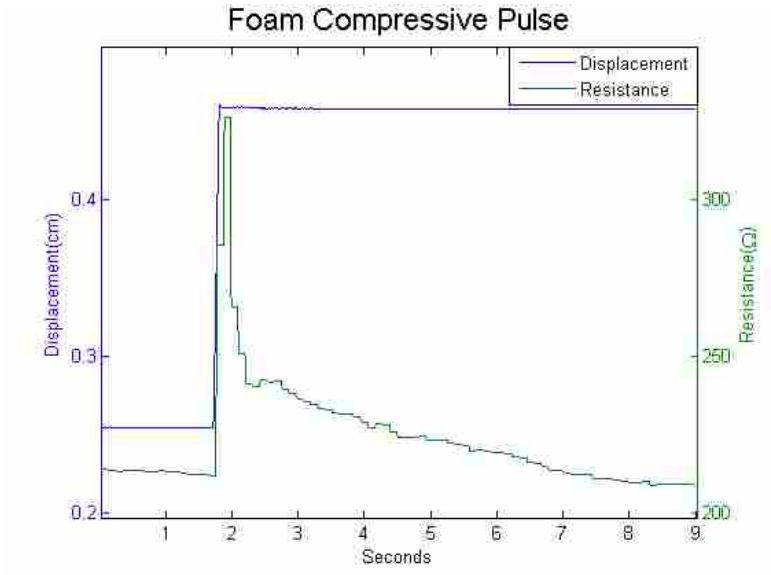

Figure B.2: The voltage response (green) to a rapid displacement from the Instron.

The second test performed was compressively cycling the foam gauge and measuring the change in resistance. This test checked to see how the foam reacts as it was repeatedly compressed, since most force sensors need to be able to withstand multiple forces. The foam reacted to the displacement with near zero lag. It is also important to note that in this test the resistance was

quickly fitted and very little post-processing was used to get the change in resistance to follow the displacement.

The results from the foam look similar to the results from the NCSGs when exciting using the direct current technique, even though this test was in compression while previous tests were in tension. The first several cycles continually decreased in maximum peak resistance before increasing for the rest of the constant cycling period. The resistance range was between 16 and 450 Ω.

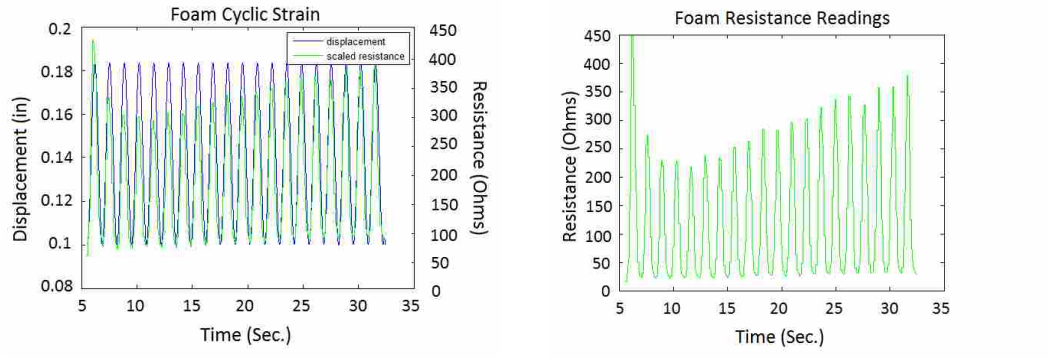

Figure B.3: The graphs show the foam response to a cyclic test. The blue line represents the displacement from the Instron and the green line represents the voltage.

From these tests it appears that the foam acted much like the strain gauges, even though the foam was under compression and the strain gauges were tested in tension. Therefore, there was some evidence that the foam will work as a compressive version of the NCSG. The foam may be better suited than the NCSGs for the data-sole as the insole of a shoe experiences more compressive forces than tensile forces.

B.4 Conclusion

The foam manufacturing process succeeded in determining a ratio of NiNs, NCCF, and weight of silicone foam that was conductive and reacted to strain. The foam reacts very quickly to changes in strain, supporting the use of the foam in both concussion detection and reactive foam padding. The foam needed further development in optimization of the filler amounts, portability, and application.

APPENDIX C. ARDUINO CODE

C.1 Find Gap Change for All Orientations

```
#include <SD.h>
#include <SPI.h>
//Set up variables to write to the SD card
File myFile;
char filename[30]; //Create an array that contains the name of our file.
int chipSelect = 8;
boolean nameFound = false; // To determine the name of the test data file
//Set up variables for the data collection
const int gaugePin = 1; // Select the input pin for the strain gauge
int led = 13; // Select the pin for the LED
int sensVolt = 0; // Variable to store the value coming from the sensor
int Hz = 100; // Set the sampling rate (samples / sec)
int baudRate = 9600;
boolean collect = false; // A trigger to start/stop collecting data
float delayTime = (1.0 / Hz) * 1000; //This is the delay in milliseconds
int sampleTime = 10; // The number of seconds to sample
boolean done = false;
char * filePrefix = “TEST”;
void setup() {
Serial.begin(baudRate);
// On the Ethernet Shield, CS is pin 4. It’s set as an output by default.
// Note that even if it’s not used as the CS pin, the hardware SS pin
// (10 on most Arduino boards, 53 on the Mega) must be left as an output
```

```
// or the SD library functions will not work.
pinMode(chipSelect,OUTPUT);
pinMode(10, OUTPUT);
pinMode(led, OUTPUT);
}// END SETUP
void loop()
{
if(!done) {
int total = sampleTime * Hz; //Determine the number of samples to record
//Flash light for 4 seconds before collecting data
for(int i=0;i<20;i++) {
digitalWrite(led, HIGH);
delay(100);
digitalWrite(led, LOW);
delay(100);
}
//Set up the SD card communication
Serial.print("Initializing SD card...");
if (!SD.begin(chipSelect)) {
Serial.println("initialization failed!");
return;
}
Serial.println("initialization done.");
// Determine the filename (sequential, starting with 1)
int file = 1;
while(!nameFound) {
sprintf(filename, "%s_\%i.dat",filePrefix,file);
if(SD.exists(filename)) {
file++;
}
```

```
else {
Serial.println("FILENAME: ");
Serial.println(filename);
nameFound = true;
}
}
myFile = SD.open(filename, FILE_WRITE);
if(!myFile) {// the file didn't open, print an error:
Serial.print("error opening ");
Serial.println(filename);
done = true;
}
//Read the voltage on the gauge for 'sampleTime' seconds
for(int i=0;i<total;i++) {
sensVolt = analogRead(gaugePin); // Read the voltage on the gauge
long mv = ((sensVolt * 500000/1023L)) / 100; //scale the voltage between 0 and 5000 millivolts
String output = "";
output += mv/1000; // Print the integer value of the voltage
output += '.';
int fraction = (mv \% 1000); // calculate the fraction
if(fraction == 0)
output += "00"; // Add two zeros for total of 3
else if(fraction <10) // if fraction <10 the 0 is ignored giving a wrong value, so add the zeros
output += "00"; // Add two zeros
else if(fraction <100)
output += "0";
output += fraction; //print the fraction
myFile.println(output);
```

```
delay(delayTime); // Set the sample rate
}
done = true; //Signal to stop data collection
myFile.close();
SPI.end(); //Free up the LED pin for signaling
}
else {
//Flash the LED slowly to show that the board is idle (not collecting data)
digitalWrite(led,HIGH);
delay(1200);
digitalWrite(led,LOW);
delay(1200);
}
}
```

APPENDIX D. MATLAB CODE RELATING TO THE FEA

D.1 Find Gap Change for All Orientations

```
%ANSYS Junction Model
%To do list:
%Hill Average for surround material
clc
clear all
format compact
n = 1; %number of tests to be performed
% Orientation of the NiNS
percentnins = 9;
percentnccf = 3;
strain = [10 20 30 40];
% strain = 30;
m=1;
%Make all possible orientations
for i = 30:30:180
for j = 30:30:90
for k = 30:30:180
for l = 30:30:90
allpos(m,:) = [i j k l];
m = m+1;
end
end
end
```

```
end
% allpos = [30,30,30,60];
%Convert strain and percentages
strain = strain.*(3600)/100;
percentnins = percentnins/100;
percentnccf = percentnccf/100;
%base material options
sylgard = 3;
basematerial = sylgard;
%Initial gap distance
s = 4.5;
%radius of NiNS
r = 50;
%Distance from bottom of the gauge to the middle of the bottom strand
vert = 600;
% set size to hurry process up
% dgap = zeros(324,1);
% orientations = zeros(324,5);
for chet =1:1
for i=1:20
fid = fopen('FEAstraingage.txt','wt'); %wipes out existing FEAstraingage.txt file
%Clear old Ansys program
fprintf(fid, 'FINISH\n');
fprintf(fid, '/clear, start\n');
fprintf(fid, '/PREP7\n');
fprintf(fid, '/TITLE, Strain Gage\n');
fprintf(fid, '\n');
%geometry
%-2*l = length of half the part
l = -600;
```

```
fprintf(fid, '!geometry\n');
fprintf(fid, 'wpoffs,-600,0,-600\n');
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-2*l,-2*l,-2*l); %xcorner,ycorner,w,h,depth
fprintf(fid, '\n');
% Determine position of center of second cylinder
% Direction vectors of the cylinders
V1 = [sind(allpos(i,2))*sind(allpos(i,1)) cosd(allpos(i,2)) sind(allpos(i,2))*cosd(allpos(i,1))]';
%Direction of the bottom cylinder
V2 = [sind(allpos(i,4))*sind(allpos(i,3)) cosd(allpos(i,4)) sind(allpos(i,4))*cosd(allpos(i,3))]';
%Direction of the top cylinder
if(allpos(i,1) == allpos(i,3) && allpos(i,2) == allpos(i,4))
if(allpos(i,1) == 0 && allpos(i,2) == 90)
d1 = [1 0 0];
elseif(allpos(i,1) == allpos(i,3) == allpos(i,2) == allpos(i,4) == 0)
d1 = [0 1 0];
else
d1(1) = 1;
d1(2) = -V1(1)/V1(2);
d1(3) = 0;
d1 = d1./norm(d1);
end
P2 = d1'*(2*r+s);
pointa = d1'*r+[0 vert 0]';
pointb = d1'*(r+s) + [0 vert 0]';
else
%Find direction that the second cylinder lies in
if(allpos(i,3) == 90 && allpos(i,2) == 0)
directionvec = [0 0 -1]';
else
directionvec = cross(V1,V2);
```

```
directionvec = directionvec/norm(directionvec);
end
P1 = [0,vert,0]';
P2 = directionvec*(s+100);
pointa = directionvec*50+P1;
pointb = directionvec*(50+s)+P1;
end
% Set up the positioning
% Rotate the global frame and create the first cylinder
fprintf(fid, 'wplane,3,0,%g,0,1,%g,0,1,%g,0 \n',vert,vert,vert+1);
fprintf(fid, 'wprota,,,%g\n',allpos(i,1));
fprintf(fid, 'wprota,,%g\n',allpos(i,2)-90);
fprintf(fid, 'cylind,50,,-500,500 \n');
fprintf(fid, '\n');
% Rotate the frame for the second cylinder and create the cylinder
fprintf(fid, 'wplane,3,0,%g,0,1,%g,0,1,%g,0 \n',vert,vert,vert+1);
fprintf(fid, 'wpoffs,%g,%g,%g\n',P2(1),P2(2),P2(3));
fprintf(fid, 'wprota,,,%g\n',allpos(i,3));
fprintf(fid, 'wprota,,%g\n',allpos(i,4)-90);
fprintf(fid, 'cylind,50,,-500,500\n');
fprintf(fid, '\n');
% Subtract the area of the epoxy from where the cylinders are
fprintf(fid, 'vsbv,1,all,,delete,keep\n');
fprintf(fid, 'vglue,all\n');
fprintf(fid, '\n');
% Create surrounding material
fprintf(fid, 'wplane,1,,,,1,,,1,1\n');
fprintf(fid, 'wpoffs,%g,%g,%g\n',3*l,-1,l-1);
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-6*l,-2*l+2,-2*l+2);
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-6*l,-2*l+2,2*l+1);
```

```
fprintf(fid, 'wplane,1,,,,1,,,1,1\n');
fprintf(fid, 'wpoffs,%g,%g,%g\n',3*l,-1,-l+1);
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-6*l,-2*l+2,-2*l-1);
fprintf(fid, 'vsbv,1,4,,delete,keep\n');
fprintf(fid, 'vsel,all\n');
fprintf(fid, 'vadd,5,6,7\n');
fprintf(fid, '\n');
%volumes in Ansys
%2-bottom cylinder
%3-top cylinder
%4-Sylgard
%5-Surrounding material
% attribute material properties
nimod = 207; %modulus nickel:
nidens = 8.912e-21; %density ni:
nipoisson = 0.31; %poisson's ratio:
nccfmod = 230; %modulus nickel:
nccfdens = 2.605e-21; %density ni:
nccfpoisson = 0.25; %poisson's ratio:
if(basematerial == 4)
basemod = 1.5e-2; %modulus nickel:
basedens = 1.1e-21; %density ni:
basepoisson = 0.45; %poisson 's ratio:
end
if(basematerial == 3)
basemod = 15e-3; %modulus nickel:
basedens = 1.29e-21; %density ni:
basepoisson = 0.45; %poisson 's ratio:
%material properties of the surrounding materials
densitysurround = (nidens*.0054 + basedens*(1-.0054));
```

```
modulussurround1 = nimod*.0054 + basemod*(1-.0054);
modulussurround2 = 1/(.0054/nimod+(1-0.0054)/basemod);
modulussurround = (modulussurround1+modulussurround2)/2;
poissonsurround = nipoisson*.0054+basepoisson*(1-.0054);
%material properties of cube and nins
fprintf(fid, '!attribute material properties\n');
fprintf(fid, 'MP,EX,1,%g\n',nimod); %Modulus of pure Ni
fprintf(fid, 'MP,PRXY,1,%g\n',nipoisson); %Poisson's ratio of pure Ni
fprintf(fid, 'MP,DENS,1,%g\n',nidens); %Density of pure Ni
fprintf(fid, 'MP,EX,2,%g\n',basemod); %Modulus of Sylgard
fprintf(fid, 'MP,PRXY,2,%g\n',basepoisson); %poisson's ratio of material
fprintf(fid, 'MP,DENS,2,%g\n',basedens); %Density of Sylgard or QM
fprintf(fid, 'MP,EX,3,%g\n',modulussurround); %Modulus of Sylgard
fprintf(fid, 'MP,PRXY,3,%g\n',poissonsurround); %poisson's ratio of material
fprintf(fid, 'MP,DENS,3,%g\n',densitysurround); %Density of Sylgard or QM
fprintf(fid, 'vsel,s,,,2,3,,1\n');
fprintf(fid, 'vatt,1\n');
fprintf(fid, 'vsel,s,,,4\n');
fprintf(fid, 'vatt,2\n');
fprintf(fid, 'vsel,s,,,1\n');
fprintf(fid, 'vatt,3\n');
%element type
fprintf(fid, 'allsel\n');
fprintf(fid, 'et,1,solid285\n');
fprintf(fid, '\n');
%&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&& %Extended Mesh
fprintf(fid, '!mesh\n');
fprintf(fid, 'vsel,s,,,2,3,1,1\n');
fprintf(fid, 'smrtsize, 7\n');
fprintf(fid, 'vmesh,2,3,1\n');
```

```
fprintf(fid, '\n');
%Refine the nodes at the junction
fprintf(fid, 'vsel,s,,,2,,,1\n');
fprintf(fid, 'a = node(%g,%g,%g)\n',pointa(1),pointa(2),pointa(3));
fprintf(fid, 'nrefine,a,,,5,,off\n');
fprintf(fid, 'a = node(%g,%g,%g)\n',pointa(1),pointa(2),pointa(3));
fprintf(fid, 'nrefine,a,,,4,,off\n');
fprintf(fid, 'a = node(%g,%g,%g)\n',pointa(1),pointa(2),pointa(3));
%Select and fix the second node
fprintf(fid, 'vsel,s,,,3,,,1\n');
fprintf(fid, 'b = node(%g,%g,%g)\n',pointb(1),pointb(2),pointb(3));
fprintf(fid, 'nrefine,b,,,5,,off\n');
fprintf(fid, 'b = node(%g,%g,%g)\n',pointb(1),pointb(2),pointb(3));
fprintf(fid, 'nrefine,b,,,4,,off\n');
fprintf(fid, 'b = node(%g,%g,%g)\n',pointb(1),pointb(2),pointb(3));
fprintf(fid, '\n');
fprintf(fid, 'vsel,s,,,1\n');
fprintf(fid, 'smrtsize,7\n');
fprintf(fid, 'vmesh,1\n');
fprintf(fid, '\n');
fprintf(fid, 'vsel,s,,,4,,,1\n');
fprintf(fid, 'smrtsize,7\n');
fprintf(fid, 'vmesh,4\n');
fprintf(fid, '\n');
%apply boundary conditions and loads
fprintf(fid, '!apply boundary conditions and loads\n');
fprintf(fid, '/solu\n');
fprintf(fid, 'local,11,0,-1800,-1,-1800\n');
fprintf(fid, 'nsel,s,loc,x,0\n');
fprintf(fid, 'nsel,r,loc,y,0\n');
```

```
fprintf(fid, 'nsel,r,loc,z,0\n');
fprintf(fid, 'd,all,all,0\n');
fprintf(fid, '\n');
fprintf(fid, 'allsel\n');
fprintf(fid, 'asel,s,area,,21\n');
fprintf(fid, 'nsla,r,1\n');
fprintf(fid, 'd,all,uz,0\n');
fprintf(fid, '\n');
%Pull one side of the gauge
fprintf(fid, '!Strain Gage\n');
fprintf(fid, 'allsel\n');
fprintf(fid, 'asel,r,area,,28\n');
fprintf(fid, 'nsla,r,1\n');
fprintf(fid, 'd,all,uz,%g\n',strain(chet));
fprintf(fid, '\n');
fprintf(fid, 'local,12,0,0,0,0,0\n');
fprintf(fid, '\n');
%solve
fprintf(fid, '!Solve\n');
fprintf(fid, 'allsel\n');
fprintf(fid, 'solve\n');
fprintf(fid, '\n');
%% Pick the nodes relating to the two cylinders
% Pick the nodes relating to the first volume and find place and displacement
fprintf(fid, 'vsel,s,,,2\n');
fprintf(fid, 'nslv,s,1\n');
fprintf(fid, '*get,numn,NODE,,COUNT ! get number of nodes \n');
fprintf(fid, '*get,nmin,NODE,0,NUM,MIN ! get min node number \n');
fprintf(fid, '! Open file before you enter the loop \n');
fprintf(fid, '*cfopen,volume11,txt,\n');
```

```
        fprintf(fid, '*do,j,1,numn \n');
        fprintf(fid, '*get,pickednodes1,node,nmin,loc,x\n');
        fprintf(fid, '*get,pickednodes2,node,nmin,loc,y\n');
        fprintf(fid, '*get,pickednodes3,node,nmin,loc,z\n');
        fprintf(fid, '*get,pickednodes4,node,nmin,u,x\n');
        fprintf(fid, '*get,pickednodes5,node,nmin,u,y\n');
        fprintf(fid, '*get,pickednodes6,node,nmin,u,z\n');
        fprintf(fid, '*vwrite,pickednodes1,pickednodes2,pickednodes3,
pickednodes4,pickednodes5,pickednodes6\n');
        fprintf(fid, '(F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2)\n');
        fprintf(fid, '! get next higher node number in selection \n');
        fprintf(fid, '*get,nmin,node,nmin,nxth \n');
        fprintf(fid, '*enddo \n');
        % Select the nodes relating to the other cylinder
        fprintf(fid, 'vsel,s,,,3\n');
        fprintf(fid, 'nslv,s,1\n');
        fprintf(fid, '*get,numn,NODE,,COUNT ! get number of nodes \n');
        fprintf(fid, '*get,nmin,NODE,0,NUM,MIN ! get min node number \n');
        fprintf(fid, '! Open file before you enter the loop \n');
        fprintf(fid, '*cfopen,volume21,txt,\n');
        fprintf(fid, '*do,n,1,numn \n');
        fprintf(fid, '*get,pickednodesv1,node,nmin,loc,x\n');
        fprintf(fid, '*get,pickednodesv2,node,nmin,loc,y\n');
        fprintf(fid, '*get,pickednodesv3,node,nmin,loc,z\n');
        fprintf(fid, '*get,pickednodesv4,node,nmin,u,x\n');
        fprintf(fid, '*get,pickednodesv5,node,nmin,u,y\n');
        fprintf(fid, '*get,pickednodesv6,node,nmin,u,z\n');
        fprintf(fid, '*vwrite,pickednodesv1,pickednodesv2,pickednodesv3\n');
        fprintf(fid, '(F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2)\n');
        fprintf(fid, '! get next higher node number in selection \n');
```

```
        fprintf(fid, '*get,nmin,node,nmin,nxth \n');
        fprintf(fid, '*enddo \n');
        fprintf(fid, '*cfclose \n');
        fprintf(fid, '\n');
        %% Matlab call Ansys
        % How to run ANSYS automatically
        fid = fopen('runme.bat', 'w+'); % Opens the file with read/write permission
        fprintf(fid,'%s\n','@echo off');
        fprintf(fid,'%s\n','set ANSYS140PRODUCT=aata');
        fprintf(fid,'%s\n','set ANSYSLOCK=off');
        fprintf(fid,'%s\n','set ANSCONSEC=YES');
        fprintf(fid,'%s\n','set ANSWAIT=1');
        fprintf(fid,'%s%s%s%s%s\n','"C:
Program Files
ANSYS Inc
v140
ansys
bin
winx64
ANSYS140" -b -I "',...
        cd, '\FEAstraingage.txt" -O "',cd,'\nickelMovement.txt"');
        fclose(fid);
        !runme
        %% Read data from writ file
        [gapdist A2 B2] = newnodedist(n);
        dgap(i) = gapdist-s;
        newgap(i) = gapdist;
        if(chet == 1)
        orientations10(i,:) = [allpos(i,1),allpos(i,2),allpos(i,3),allpos(i,4), newgap(i)];
        end
```

```
if(chet == 2)
orientations20(i,:) = [allpos(i,1),allpos(i,2),allpos(i,3),allpos(i,4), newgap(i)];
end
if(chet == 3)
orientations30(i,:) = [allpos(i,1),allpos(i,2),allpos(i,3),allpos(i,4), newgap(i)];
end
if(chet == 4)
orientations40(i,:) = [allpos(i,1),allpos(i,2),allpos(i,3),allpos(i,4), newgap(i)];
end
end
end
```

D.2 Find the New Gap Size

```
function [gapdist A2 B2] = newnodedist(n)
% ================================================================
%
% clear all
% n = 1; % Be sure to erase this before running FEAstraingauge.txt
% ================================================================
%
% Iterate the distance between the two nodes
for run = 1:n
nerror = 1;
% Read data from written file
if(run == 1)
[Ax1,Ay1,Az1,dAx1,dAy1,dAz1] = textread('volume11.txt','%f %f %f %f %f %f');
[Bx1,By1,Bz1,dBx1,dBy1,dBz1] = textread('volume21.txt','%f %f %f %f %f %f');
end
if(run == 2)
[Ax1,Ay1,Az1,dAx1,dAy1,dAz1] = textread('volume12.txt','%f %f %f %f %f %f');
```

```
[Bx1,By1,Bz1,dBx1,dBy1,dBz1] = textread('volume22.txt','%f %f %f %f %f %f');
end
if(run == 3)
[Ax1,Ay1,Az1,dAx1,dAy1,dAz1] = textread('volume13.txt','%f %f %f %f %f %f');
[Bx1,By1,Bz1,dBx1,dBy1,dBz1] = textread('volume23.txt','%f %f %f %f %f %f');
end
if(run == 4)
[Ax1,Ay1,Az1,dAx1,dAy1,dAz1] = textread('volume14.txt','%f %f %f %f %f %f');
[Bx1,By1,Bz1,dBx1,dBy1,dBz1] = textread('volume24.txt','%f %f %f %f %f %f');
end
% Make the nodes at the new points
% Find the length of the first file
Axlength = length(Ax1);
Ax2 = zeros(Axlength,1);
Ay2 = zeros(Axlength,1);
Az2 = zeros(Axlength,1);
% Create for loop to make each new point
for i = 1:Axlength
Ax2(i) = Ax1(i) + dAx1(i);
Ay2(i) = Ay1(i) + dAy1(i);
Az2(i) = Az1(i) + dAz1(i);
end
%Find the length of the second file
Bxlength = length(Bx1);
Bx2 = zeros(Bxlength,1);
By2 = zeros(Bxlength,1);
Bz2 = zeros(Bxlength,1);
for i = 1:Bxlength
Bx2(i) = Bx1(i) + dBx1(i);
By2(i) = By1(i) + dBy1(i);
```

```
Bz2(i) = Bz1(i) + dBz1(i);
end
% Find the minimum distance
% [gapdist A2 B2]=nearneighbour(Ax1,Ay1,Az1,Bx1,By1,Bz1);
[gapdist(run,:) A2 B2]=nearneighbour(Ax2,Ay2,Az2,Bx2,By2,Bz2);
clear Ax1 Ay1 Az1 dAx1 dAy1 dAz1 Bx1 By1 Bz1 dBx1 dBy1 dBz1 Ax2 Ay2 Az2 Bx2 By2 Bz2
end
```

D.3 Solve Validation Orientations

```
%ANSYS Junction Model
%To do list:
%Hill Average for surround material
clc
clear all
format compact
n = 4; %number of tests to be performed
%Input values of theta1,phi1,theta2,phi2
theta1 = [30 90 90 90]; %90 0 0 0 90]; %0 0]; %theta is the horizontal plane angle (0-180) in degrees
phi1 = [90 90 90 90]; %90 0 90 0 90]; %-45 90]; %phi is for the vertical plane (0-90) in degrees
theta2 = [40 90 90 90]; %90 90 90 0 90]; %90 90];
phi2 = [90 90 90 90]; %90 90 90 0 90]; %90 45];
% Orientation of the NiNS
percentnins = 9;
percentnccf = 3;
strain = [10 20 30 40];
% strain = 1;
%Convert strain and percentages
```

```
strain = strain*(3600)/100;
percentnins = percentnins/100;
percentnccf = percentnccf/100;
%base material options
sylgard = 4;
QM = 3;
basematerial = sylgard;
%Initial gap distance
% s = [1,3,5,7,9,10,11,13,15,17,19,20,40];
% s = [13,15,17,19,20,40];
s = 40;
chet = 1;
%radius of NiNS
r = 50;
%Distance from bottom of the gauge to the middle of the bottom strand
vert = 600;
% j = 1; % Comment out this and comment in the next line to run full test
for j = 1:chet
for i=1:n
fid = fopen('FEAstraingage.txt','wt'); %wipes out existing FEAstraingage.txt file
%Clear old Ansys program
fprintf(fid, 'FINISH\n');
fprintf(fid, '/clear, start\n');
fprintf(fid, '/PREP7\n');
fprintf(fid, '/TITLE, Strain Gage\n');
fprintf(fid, '\n');
%geometry
%-2*l = length of half the part
l = -600;
fprintf(fid, '!geometry\n');
```

```
fprintf(fid, 'wpoffs,-600,0,-600\n');
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-2*l,-2*l,-2*l); %xcorner,ycorner,w,h,depth
fprintf(fid, '\n');
% Determine position of center of second cylinder
%Direction vectors of the cylinders
V1 = [sind(phi1(i))*sind(theta1( i)) cosd(phi1(i)) sind(phi1(i))*cosd(theta1(i))]'; %Direction of the bottom cylinder
V2 = [sind(phi2(i))*sind(theta2(i)) cosd(phi2(i)) sind(phi2(i))*cosd(theta2(i))]'; %Direction of the top cylinder
if(theta1(i) == theta2(i) && phi1(i) == phi2(i))
if(theta1(i) == 0 && phi1(i) == 90)
d1 = [1 0 0];
elseif(theta1(i) == theta2(i) == phi1(i) == phi2(i) == 0)
d1 = [0 1 0];
else d1(1) = 1;
d1(2) = -V1(1)/V1(2);
d1(3) = 0;
d1 = d1./norm(d1);
end
P2 = d1'*(2*r+s(j));
pointa = d1'*r+[0 vert 0]';
pointb = d1'*(r+s(j)) + [0 vert 0]';
%&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&%
%Set point and b for verification 1
% P2 = [0 0 -100-s(j)]';
% pointa = [0 600 -50]';
% pointb = [0 600 -50-s(j)]';
%&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&% else
%Find direction that the second cylinder lies in
if(theta2(i) == 90 && phi1(i) == 0)
```

```
directionvec = [0 0 -1]';
else
directionvec = cross(V1,V2);
directionvec = directionvec/norm(directionvec);
end
P1 = [0,vert,0]';
P2 = directionvec*(s(j)+100);
pointa = directionvec*50+P1;
pointb = directionvec*(50+s(j))+P1;
end
% Set up the positioning
% Rotate the global frame and create the first cylinder
fprintf(fid, 'wplane,3,0,%g,0,1,%g,0,1,%g,0 \n',vert,vert,vert+1);
fprintf(fid, 'wprota,,,%g\n',theta1(i)); fprintf(fid, 'wprota,,%g\n',phi1(i)-90);
fprintf(fid, 'cylind,50,,-500,500 \n');
fprintf(fid, '\n');
% Rotate the frame for the second cylinder and create the cylinder
fprintf(fid, 'wplane,3,0,%g,0,1,%g,0,1,%g,0 \n',vert,vert,vert+1);
fprintf(fid, 'wpoffs,%g,%g,%g\n',P2(1),P2(2),P2(3));
fprintf(fid, 'wprota,,,%g\n',theta2(i));
fprintf(fid, 'wprota,,%g\n',phi2(i)-90);
fprintf(fid, 'cylind,50,,-500,500\n');
fprintf(fid, '\n');
% Subtract the area of the epoxy from where the cylinders are
fprintf(fid, 'vsbv,1,all,,delete,keep\n');
fprintf(fid, 'vglue,all\n');
fprintf(fid, '\n');
% Create surrounding material
fprintf(fid, 'wplane,1,,,,1,,,1,1\n');
fprintf(fid, 'wpoffs,%g,%g,%g\n',3*l,-1,l-1);
```

```
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-6*l,-2*l+2,-2*l+2);
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-6*l,-2*l+2,2*l+1);
fprintf(fid, 'wplane,1,,,,1,,,1,1\n');
fprintf(fid, 'wpoffs,%g,%g,%g\n',3*l,-1,-l+1);
fprintf(fid, 'blc4,0,0,%g,%g,%g\n',-6*l,-2*l+2,-2*l-1);
fprintf(fid, 'vsbv,1,4,,delete,keep\n');
fprintf(fid, 'vsel,all\n');
fprintf(fid, 'vadd,5,6,7\n');
fprintf(fid, '\n');
%volumes in Ansys
%2-bottom cylinder
%3-top cylinder
%4-Sylgard
%5-Surrounding material
% attribute material properties
nimod = 207; %modulus nickel:
nidens = 8.912e-21; %density ni:
nipoisson = 0.31; %poisson's ratio:
nccfmod = 230; %modulus nickel:
nccfdens = 2.605e-21; %density ni:
nccfpoisson = 0.25; %poisson's ratio
if(basematerial == 4)
basemod = 1.5e-2; %modulus nickel:
basedens = 1.1e-21; %density ni:
basepoisson = 0.45; %poisson 's ratio:
end
if(basematerial == 3)
basemod = 15e-3; %modulus nickel:
basedens = 1.29e-21; %density ni:
basepoisson = 0.45; %poisson 's ratio:
```

```
end
%material properties of the surrounding materials
densitysurround = (nidens*.0054 + basedens*(1-.0054));
modulussurround1 = nimod*.0054 + basemod*(1-.0054);
modulussurround2 = 1/(.0054/nimod+(1-0.0054)/basemod);
modulussurround = (modulussurround1+modulussurround2)/2;
poissonsurround = nipoisson*.0054+basepoisson*(1-.0054);
%material properties of cube and nins
fprintf(fid, '!attribute material properties\n');
fprintf(fid, 'MP,EX,1,%g\n',nimod); %Modulus of pure Ni
fprintf(fid, 'MP,PRXY,1,%g\n',nipoisson); %Poisson's ratio of pure Ni
fprintf(fid, 'MP,DENS,1,%g\n',nidens); %Density of pure Ni
fprintf(fid, 'MP,EX,2,%g\n',basemod); %Modulus of Sylgard
fprintf(fid, 'MP,PRXY,2,%g\n',basepoisson); %poisson's ratio of material
fprintf(fid, 'MP,DENS,2,%g\n',basedens); %Density of Sylgard or QM
fprintf(fid, 'MP,EX,3,%g\n',modulussurround); %Modulus of Sylgard
fprintf(fid, 'MP,PRXY,3,%g\n',poissonsurround); %poisson's ratio of material
fprintf(fid, 'MP,DENS,3,%g\n',densitysurround); %Density of Sylgard or QM
fprintf(fid, 'vsel,s,,,2,3,,1\n');
fprintf(fid, 'vatt,1\n');
fprintf(fid, 'vsel,s,,,4\n');
fprintf(fid, 'vatt,2\n');
fprintf(fid, 'vsel,s,,,1\n');
fprintf(fid, 'vatt,3\n');
%element type
fprintf(fid, 'allsel\n');
fprintf(fid, 'et,1,solid285\n');
fprintf(fid, '\n');
%&&&&&&&&&&&&&&&&&&&&&&&&&&& fprintf(fid, '!mesh\n');
fprintf(fid, 'vsel,s,,,2,3,1,1\n');
```

```
fprintf(fid, 'smrtsize, 7\n');
fprintf(fid, 'vmesh,2,3,1\n');
fprintf(fid, '\n');
%Refine the nodes at the junction
fprintf(fid, 'vsel,s,,,2,,,1\n');
fprintf(fid, 'a = node(%g,%g,%g)\n',pointa(1),pointa(2),pointa(3));
fprintf(fid, 'nrefine,a,,,5,,off\n');
fprintf(fid, 'a = node(%g,%g,%g)\n',pointa(1),pointa(2),pointa(3));
fprintf(fid, 'nrefine,a,,,4,,off\n');
fprintf(fid, 'a = node(%g,%g,%g)\n',pointa(1),pointa(2),pointa(3));
%Select and fix the second node
fprintf(fid, 'vsel,s,,,3,,,1\n');
fprintf(fid, 'b = node(%g,%g,%g)\n',pointb(1),pointb(2),pointb(3));
fprintf(fid, 'nrefine,b,,,5,,off\n');
fprintf(fid, 'b = node(%g,%g,%g)\n',pointb(1),pointb(2),pointb(3));
fprintf(fid, 'nrefine,b,,,4,,off\n');
fprintf(fid, 'b = node(%g,%g,%g)\n',pointb(1),pointb(2),pointb(3));
fprintf(fid, '\n');
fprintf(fid, 'vsel,s,,,1\n');
fprintf(fid, 'smrtsize,7\n');
fprintf(fid, 'vmesh,1\n');
fprintf(fid, '\n');
fprintf(fid, 'vsel,s,,,4,,,1\n');
fprintf(fid, 'smrtsize,7\n');
fprintf(fid, 'vmesh,4\n');
fprintf(fid, '\n');
%apply boundary conditions and loads
fprintf(fid, '!apply boundary conditions and loads\n');
fprintf(fid, '/solu\n');
fprintf(fid, 'local,11,0,-1800,-1,-1800\n');
```

```
fprintf(fid, 'nsel,s,loc,x,0\n');
fprintf(fid, 'nsel,r,loc,y,0\n');
fprintf(fid, 'nsel,r,loc,z,0\n');
fprintf(fid, 'd,all,all,0\n');
fprintf(fid, '\n');
fprintf(fid, 'allsel\n');
fprintf(fid, 'asel,s,area,,21\n');
fprintf(fid, 'nsla,r,1\n');
fprintf(fid, 'd,all,uz,0\n');
fprintf(fid, '\n');
%Pull one side of the gauge
fprintf(fid, '!Strain Gage\n');
fprintf(fid, 'allsel\n');
fprintf(fid, 'asel,r,area,,28\n');
fprintf(fid, 'nsla,r,1\n');
fprintf(fid, 'd,all,uz,%g\n',strain(i));
fprintf(fid, '\n');
fprintf(fid, 'local,12,0,0,0,0,0\n');
fprintf(fid, '\n');
%solve fprintf(fid, '!Solve\n');
fprintf(fid, 'allsel\n');
fprintf(fid, 'solve\n');
fprintf(fid, '\n');
%% Pick the nodes relating to the two cylinders
% Pick the nodes relating to the first volume and find place and displacement
fprintf(fid, 'vsel,s,,,2\n');
fprintf(fid, 'nslv,s,1\n');
fprintf(fid, '*get,numn,NODE,,COUNT ! get number of nodes \n');
fprintf(fid, '*get,nmin,NODE,0,NUM,MIN ! get min node number \n');
fprintf(fid, '! Open file before you enter the loop \n');
```

```
        if(i == 1)
        fprintf(fid, '*cfopen,volume11,txt,\n');
        end
        if(i == 2)
        fprintf(fid, '*cfopen,volume12,txt,\n');
        end
        if(i == 3)
        fprintf(fid, '*cfopen,volume13,txt,\n');
        end
        if(i == 4)
        fprintf(fid, '*cfopen,volume14,txt,\n');
        end
        fprintf(fid, '*do,j,1,numn \n');
        fprintf(fid, '*get,pickednodes1,node,nmin,loc,x\n');
        fprintf(fid, '*get,pickednodes2,node,nmin,loc,y\n');
        fprintf(fid, '*get,pickednodes3,node,nmin,loc,z\n');
        fprintf(fid, '*get,pickednodes4,node,nmin,u,x\n');
        fprintf(fid, '*get,pickednodes5,node,nmin,u,y\n');
        fprintf(fid, '*get,pickednodes6,node,nmin,u,z\n');
        fprintf(fid, '*vwrite,pickednodes1,pickednodes2,pickednodes3,
pickednodes4,pickednodes5,pickednodes6\n');
        fprintf(fid, '(F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2)\n');
        fprintf(fid, '! get next higher node number in selection \n');
        fprintf(fid, '*get,nmin,node,nmin,nxth \n');
        fprintf(fid, '*enddo \n');
        % Select the nodes relating to the other cylinder
        fprintf(fid, 'vsel,s,,,3\n');
        fprintf(fid, 'nslv,s,1\n');
        fprintf(fid, '*get,numn,NODE,,COUNT ! get number of nodes \n');
        fprintf(fid, '*get,nmin,NODE,0,NUM,MIN ! get min node number \n');
```

```
fprintf(fid, '! Open file before you enter the loop \n');
if(i == 1)
fprintf(fid, '*cfopen,volume21,txt,\n');
end
if(i == 2)
fprintf(fid, '*cfopen,volume22,txt,\n');
end
if(i == 3) fprintf(fid, '*cfopen,volume23,txt,\n');
end
if(i == 4)
fprintf(fid, '*cfopen,volume24,txt,\n');
end
fprintf(fid, '*do,n,1,numn \n');
fprintf(fid, '*get,pickednodesv1,node,nmin,loc,x\n');
fprintf(fid, '*get,pickednodesv2,node,nmin,loc,y\n');
fprintf(fid, '*get,pickednodesv3,node,nmin,loc,z\n');
fprintf(fid, '*get,pickednodesv4,node,nmin,u,x\n');
fprintf(fid, '*get,pickednodesv5,node,nmin,u,y\n');
fprintf(fid, '*get,pickednodesv6,node,nmin,u,z\n');
fprintf(fid, '(F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2,2X,F9.2)\n');
fprintf(fid, '! get next higher node number in selection \n');
fprintf(fid, '*get,nmin,node,nmin,nxth \n');
fprintf(fid, '*enddo \n');
fprintf(fid, '*cfclose \n');
fprintf(fid, '\n');
%% Matlab call Ansys
% How to run ANSYS automatically
fid = fopen('runme.bat', 'w+'); % Opens the file with read/write permission
fprintf(fid,'%s\n','@echo off');
fprintf(fid,'%s\n','set ANSYS140PRODUCT=aata');
```

```
        fprintf(fid,'%s\n','set ANSYSLOCK=off');
        fprintf(fid,'%s\n','set ANSCONSEC=YES');
        fprintf(fid,'%s\n','set ANSWAIT=1');
        fprintf(fid,'%s%s%s%s%s\n','"C:
Program Files
ANSYS Inc
v140
ansys
bin
winx64
ANSYS140" -b -I "',...
        cd, '\FEAstraingage.txt" -O "',cd,'\nickelMovement.txt"');
        fclose(fid);
        !runme
        end
        %% Read data from writ file
        [gapdist A2 B2] = newnodedist(n);
        dgap(:,j) = gapdist'-s(j)*[1,1,1,1];
        gap(:,j) = gapdist';
        end
```

APPENDIX E. GAIT RELATED CODE

E.1 Match the Signals Program

```
% Taylor Remington
% Match walking and jogging signals to composite walking then jogging
% signals
clc
clear all
close all
%% Gather the super signals and the single cycles
% Call function to get full data
% Load the values
% Adam _ too long on the jogging
% Adam _ too quick on the walking
% Tester 1
% Brad_1_4 _ good
% Tester 2
% Adam_1 _ sensors 2_4
% Adam_2 _ same
% Adam_3 _ can use sensors 1_4 if using last 60 seconds
% Adam_4 _ sensors 2 _ 4
% Tester 3
% Steve_1_3 _ good
% Steve_4 _ no sensor_2
% Jeff_4 _ Nothing is usable
% Load Brad's data
```

% tester $\bar{1}$;

% test_1 $\bar{\text{B}}$RAD_1;

% test_2 = BRAD_2;

% test_3 = BRAD_3;

% test_4 = BRAD_4;

% % Load Adam's data

% tester = 2;

% test_1 = ADAM_1;

% test_2 = ADAM_2;

% test_3 = ADAM_3;

% test_4 = ADAM_4;

% % Load Steve's data

% tester = 3;

% test_1 = STEVE_1;

% test_2 = STEVE_2;

% test_3 = STEVE_3;

% test_4 = STEVE_4;

% Load Jeff's Data

% tester = 3;

% test_1 = JEFF_1;

% test_2 = JEFF_2;

% test_3 = JEFF_3;

% test_4 = JEFF_4;

% Subject 1 = Steve, 4.85 % error in calories

% 12.16 % error in time

% Subject 2 = Brad, 3.59 % error in calories

% 15.05 % error in time

% Subject 3 = Adam, cutoff frequency = 83.9 %

% 1.03 % error in calories, 1.19 % error in time

% Read in the gym tests

```
% m=1;
% [super_signal] = take_signals(m,test_1,test_2,test_3,test_4,tester);
% Fast Fourier Transform
% fast = fft(super_signal(18001:24000));
% figure
% f = linspace(0,100,length(fast));
% plot(f,abs(fast)/length(fast));
% ylabel('Magnitude')
% xlabel('Hz')
% title('Fourier Transform')
% Frequencies according to highest FT peak
% Brad Walk frequency = 0.7335
% Brad Jog frequency = 1.317
% Brad Bike total = 0.1667
% Brad Stairs total = 0.4834
% Call function to get the data
tester = 3;
m=1;
[GW_comp,GJ_comp,GBF_comp,GST_comp] = read_sensor_values(m,tester);
% Get the first trial from running on the lawn
m=500; mm = 500;
[comp_zeroed,composite_lawn,comp_zeroed_2] = read_lawn_signal(m,mm);
super_signal = comp_zeroed_2;
% super_signal = comp_zeroed;
% Load in the long test data
% [super_signal] = looong_gym_test(variable);
% Zero walk signal
m = 500; % Seconds(*100) to zero out data (5 secs)
[GW_comp] = zero_signals(GW_comp,m);
% Zero jog signal every 3 seconds
```

```
m = 300;
[GJ_comp] = zero_signals(GJ_comp,m);
GJ_comp = GJ_comp(300:3000);
GJ_comp = padarray(GJ_comp,300,0,'pre');
% Zero biking signal
m = 500;
[GBF_comp] = zero_signals(GBF_comp,m);
% Zero stairs signal
m = 500;
[GST_comp] = zero_signals(GST_comp,m);
% super signal use to test original signals
% super_signal = [GW_comp;GJ_comp;GBF_comp;GEF_comp;GJ_comp;GE_comp];
% super_signal = [GBF_comp];
%Call functions to get numbered cycle
% Corrected for different exercisers
% Walk cycle function
[matrix_cycle_walk] = walk_cycle(GW_comp);
% Jog cycle function
[matrix_cycle_jog] = jog_cycle(GJ_comp);
% Bike cycle function
[matrix_cycle_bike] = bike_cycle(GBF_comp);
% Elliptical cycle function
[matrix_cycle_stairs] = stairs_cycle(GST_comp);
% Frequencies of single cycle tests
walk_cycle_freq = 0.8003;
jog_cycle_freq = 1.233;
bike_cycle_freq = 0.4001; % This fourier transform is not clean, as expected
stairs_cycle_freq = 0.6002; % This is faster than I would have thought
%% Next stage of code
% Turn super_signal to ones and zeros
```

```
[super_signal_matrix] = ones_and_zeros(super_signal);
% Find derivatives
% matrix_cycle = matrix_cycle_walk;
[super_derivs] = derivatives(super_signal);
% Match the walking or jogging signal with the super signal
% Find the matches for walking
[match_walk] = find_match(super_signal_matrix,super_derivs,matrix_cycle_walk);
% Find the matches for jogging
[match_jog] = find_match(super_signal_matrix,super_derivs,matrix_cycle_jog);
% Find the matches for biking
[match_bike] = find_match(super_signal_matrix,super_derivs,matrix_cycle_bike);
% Find the matches for the stairs
[match_stairs] = find_match(super_signal_matrix,super_derivs,matrix_cycle_stairs);
%% Find the best matches and when the matches change exercises
% Make matrix telling where the exercise are
% Walking = 1, Jogging = 2, Biking = 3
cutoff = 0.839; % choose the cutoff percentage to match the exercise
[match_decide] = decide_exercise(match_walk,match_jog,match_bike,match_stairs,
cutoff,super_derivs);
% Find where the exercise changes
[change_exercise] = changing_exercises(match_decide);
%% Graph matched signal
% Graph one cycle to the super_signal starting at point k
% k = 0.1083*60*100;
% k = uint16(k);
% matrix_cycle = matrix_cycle_walk;
% graph_matched_signal(k,matrix_cycle,super_signal_matrix);
%% Plot the final result
% Plot the different exercises
[segmented_signal] = plot_exercises(change_exercise,super_signal);
```

%% Find out the calories burnt in real_life and according to this exercise

% Display the amount of calories per excercise per hour According to

% http://healthyliving.azcentral.com/differences_between_bikes_vs_treadmills_ vs_stairs_machines_2790.html Walking = 300; % calories per hour

Jogging = 600; % calories per hour

Stairs = 680; % calories per hour

Biking = 780; % calories per hour

% Display the calories burnt per exercise per hour

display_calories_2(Walking,Jogging,Stairs,Biking);

% Find walking factor

[walk_factor] = signal_to_cal(Walking);

% Find jogging factor

[jog_factor] = signal_to_cal(Jogging);

% Find biking factor

[bike_factor] = signal_to_cal(Biking);

% Find Stairs factor

[stairs_factor] = signal_to_cal(Stairs);

% Find how many calories were burnt and compare to actual

% Change this if the super signal is changed

[total_cal_from_signal] = percent_err_all(segmented_signal,change_exercise,walk_factor,jog_factor,bike_factor,stairs_factor);

E.1.1 Related Functions

Read Sensor Values

function [GW_comp,GJ_comp,GBF_comp,GST_comp] = read_sensor_values(m,tester)

load GYM_JOG.dat

load GYM_WALK.dat

load GYM_STAIRS.dat

load G_STAIRS_BIKE.dat

```
G_STAIRS_BIKE = G_STAIRS_BIKE(6000*4:9000*4);
% length(G_STAIRS_BIKE)
for i=1:4:length(GYM_WALK)
sensor1GW(m) = GYM_WALK(i);
sensor2GW(m) = GYM_WALK(i+1);
sensor3GW(m) = GYM_WALK(i+2);
sensor4GW(m) = GYM_WALK(i+3);
sensor1GJ(m) = GYM_JOG(i);
sensor2GJ(m) = GYM_JOG(i+1);
sensor3GJ(m) = GYM_JOG(i+2);
sensor4GJ(m) = GYM_JOG(i+3);
sensor1GBF(m) = G_STAIRS_BIKE(i);
sensor2GBF(m) = G_STAIRS_BIKE(i+1);
sensor3GBF(m) = G_STAIRS_BIKE(i+2);
sensor4GBF(m) = G_STAIRS_BIKE(i+3);
sensor1GST(m) = GYM_STAIRS(i);
sensor2GST(m) = GYM_STAIRS(i+1);
sensor3GST(m) = GYM_STAIRS(i+2);
sensor4GST(m) = GYM_STAIRS(i+3);
m = m+1;
end
span = 15;
if tester = 1
GW_comp = sensor2GW + sensor3GW + sensor4GW;
GW_comp = smooth(GW_comp,span);
GJ_comp = sensor2GJ + sensor3GJ + sensor4GJ;
GJ_comp = smooth(GJ_comp,span);
GBF_comp = sensor1GBF + sensor2GBF + sensor3GBF + sensor4GBF;
GBF_comp = smooth(GBF_comp,span);
GST_comp = sensor2GST + sensor3GST + sensor4GST;
```

```
GST_comp = smooth(GST_comp,span);
end
if tester = 2
GW_comp = sensor2GW + sensor3GW + sensor4GW;
GW_comp = smooth(GW_comp,span);
GJ_comp = sensor2GJ + sensor3GJ + sensor4GJ;
GJ_comp = smooth(GJ_comp,span);
GBF_comp = sensor2GBF + sensor3GBF + sensor4GBF;
GBF_comp = smooth(GBF_comp,span);
GST_comp = sensor2GST + sensor3GST + sensor4GST;
GST_comp = smooth(GST_comp,span);
end
if tester = 3
GW_comp = sensor1GW + sensor2GW + sensor3GW + sensor4GW;
GW_comp = smooth(GW_comp,span);
GJ_comp = sensor1GJ + sensor2GJ + sensor3GJ + sensor4GJ;
GJ_comp = smooth(GJ_comp,span);
GBF_comp = sensor1GBF + sensor2GBF + sensor3GBF + sensor4GBF;
GBF_comp = smooth(GBF_comp,span);
GST_comp = sensor1GST + sensor3GST + sensor4GST;
GST_comp = smooth(GST_comp,span);
end
end
```

Read Signal from Lawn Test

```
function [comp_zeroed,composite_lawn,comp_zeroed_2] = read_lawn_signal(m,mm)
load WEB__1.dat % Running on the field (walk _ jog _ walk _ jog) each for 30 s
load WEB__2.dat % Running on the field (walk _ jog _ walk _ jog) each for 30 s again
mn = 1;
for i=1:4:length(WEB__1)
```

```
sensor1(mn) = WEB__1(i);
sensor2(mn) = WEB__1(i+1);
sensor3(mn) = WEB__1(i+2);
sensor4(mn) = WEB__1(i+3);
sensor12(mn) = WEB__2(i);
sensor22(mn) = WEB__2(i+1);
sensor32(mn) = WEB__2(i+2);
sensor42(mn) = WEB__2(i+3);
mn = mn+1;
end
% Compile signal, smooth data
span = 15;
comp_zeroed = sensor1 + sensor2 + sensor3 + sensor4;
comp_zeroed = smooth(comp_zeroed,span);
% make the super signal from a 0 to 1
n = 1;
p = m;
for i = 1:length(comp_zeroed)/p
comp_zeroed(n:m) = comp_zeroed(n:m) _ min(comp_zeroed(n:m));
comp_zeroed(n:m) = comp_zeroed(n:m)./max(comp_zeroed(n:m));
n = n + p;
m = m + p;
end
% Smooth and scale the data from the second test
comp_zeroed_2 = sensor12 + sensor22 + sensor32 + sensor42;
comp_zeroed_2 = smooth(comp_zeroed_2,span);
% make the super signal from a 0 to 1
n = 1;
p = mm;
for i = 1:length(comp_zeroed_2)/p
```

```
comp_zeroed_2(n:mm) = comp_zeroed_2(n:mm) _ min(comp_zeroed_2(n:mm));
comp_zeroed_2(n:mm) = comp_zeroed_2(n:mm)./max(comp_zeroed_2(n:mm));
n = n + p;
mm = mm + p;
end
% Zero data then add
sensor1 = sensor1 _ min(sensor1);
sensor1 = sensor1./max(sensor1);
sensor2 = sensor2 _ min(sensor2);
sensor2 = sensor2./max(sensor2);
sensor3 = sensor3 _ min(sensor3);
sensor3 = sensor3./max(sensor3);
sensor4 = sensor4 _ min(sensor4);
sensor4 = sensor4./max(sensor4);
composite_lawn = sensor1+sensor2+sensor3+sensor4;
composite_lawn = composite_lawn _ min(composite_lawn);
composite_lawn = composite_lawn./max(composite_lawn);
end
```

Zero signals

```
function [comp] = zero_signals(comp,m)
n = 1;
p = m;
for i = 1:length(comp)/p
comp(n:m) = comp(n:m) _ min(comp(n:m));
comp(n:m) = comp(n:m)./(max(comp(n:m)));
n = n + p;
m = m + p;
end
end
```

Read Walk Signal

```
function [matrix_cycle_walk] = walk_cycle(GW_comp)
%% Zoom into one cycle and create matrix (WALKING)
%Zoom into one cycle
xmin_walk = 309;%15.61*100;
xmax_walk = 438+50;%16.9*100+50;
xmax_walk = uint16(xmax_walk);
GW_cycle1 = GW_comp(xmin_walk:xmax_walk) _ min(GW_comp(xmin_walk:xmax_walk));
GW_cycle1 = GW_cycle1./max(GW_cycle1);
n = 1;
% % Plot a nice plot to put into paper of one cycle
% t = 0:0.01:length(GW_cycle)/100_0.01;
% figure
% plot(t,GW_cycle)
% title('Walk: Single Cycle', 'FontSize', 16);
% ylabel('Scaled Values')
% xlabel('Time(s)')
% Change values for different subjects due to time
% change for adam's test
for i = 1:4:179
GW_cycle(n) = GW_cycle1(i);
GW_cycle(n+1) = GW_cycle1(i+1);
GW_cycle(n+2) = GW_cycle1(i+2);
% GJ_cycle(n+3) = GJ_cycle1(i+3);
n = n + 3;
end
GW_cycle = GW_cycle*10000;
%create matrix
%First create counter (x_values)
counter_walk = zeros(1,length(GW_cycle));
```

```
for i = 1:1:length(GW_cycle)
counter_walk(i) = i;
end
%Create for loop to get ones and negative ones so we can check positive and
%negative match
matrix_cycle_walk = zeros(10000:length(counter_walk));
for i = 1:1:length(counter_walk)
for j = 1:1:10000
if j <GW_cycle(i)
matrix_cycle_walk(10001_j,i) = 1;
else
matrix_cycle_walk(10001_j,i) = _1;
end
end
end
%Check to see if the for loop worked correctly then comment out
% figure
% image(matrix_cycle_walk.* 255);
% title('Walk: Ones and Negative Ones', 'FontSize', 16);
end
```

Read Jog Signal

```
function [matrix_cycle_jog] = jog_cycle(GJ_comp)
%% Create Jogging Matrix
%Zoom into one cycle
xmin_jog = 7.59*100;
xmax_jog = 8.4*100+100;
xmax_jog = uint16(xmax_jog);
GJ_cycle1 = GJ_comp(xmin_jog:xmax_jog) _ min(GJ_comp(xmin_jog:xmax_jog));
GJ_cycle1 = GJ_cycle1./max(GJ_cycle1);
```

```
n = 1;
% Correct for brad
% if(max(GJ_comp = 3.24) && min(GJ_comp) = 0)
% for i = 1:6:180
% GJ_cycle(n) = GJ_cycle1(i);
% GJ_cycle(n+1) = GJ_cycle1(i+1);
% GJ_cycle(n+2) = GJ_cycle1(i+2);
% GJ_cycle(n+3) = GJ_cycle1(i+3);
% GJ_cycle(n+4) = GJ_cycle1(i+4);
% n = n + 5;
% end
% end
% Correct for Adam
% for i = 1:4:180
% GJ_cycle(n) = GJ_cycle1(i);
% GJ_cycle(n+1) = GJ_cycle1(i+1);
% GJ_cycle(n+2) = GJ_cycle1(i+2);
% % GJ_cycle(n+3) = GJ_cycle1(i+3);
% n = n + 3;
% end
GJ_cycle = GJ_cycle1*10000;
% Check signal
% t = 0:0.01:length(GJ_cycle)/100_0.01;
% figure
% plot(t,GJ_cycle)
% title('Jog: Single Cycle', 'FontSize', 16);
% ylabel('Scaled Values')
% xlabel('Time(s)')
%create matrix
%First create counter (x_values)
```

```
counter_jog = zeros(1,length(GJ_cycle));
for i = 1:1:length(GJ_cycle)
counter_jog(i) = i;
end
%Multiply GW_cycle by 10,000 to get the vaules into integers
% GJ_comp = GJ_comp./(max(GJ_comp));
%Create for loop to get ones and negative ones so we can check positive and
%negative match
matrix_cycle_jog = zeros(10000:length(counter_jog));
for i = 1:length(counter_jog)
for j = 1:10000
if j <GJ_cycle(i)
matrix_cycle_jog(10001_j,i) = 1;
else
matrix_cycle_jog(10001_j,i) = _1;
end
end
end
%Graphically check data
% figure
% image(matrix_cycle_jog.* 255);
% title('Jog: Ones and Negative Ones', 'FontSize', 16);
```

Read Bike Signal

```
function [matrix_cycle_bike] = bike_cycle(GBF_comp)
%% Zoom into one cycle and create matrix (Biking)
%Zoom into one cycle
xmin_bike_fast = 10.02*100;
xmax_bike_fast = 12.77*100;
xmax_bike = uint16(xmax_bike_fast);
```

```
GBF_cycle = GBF_comp(xmin_bike_fast:xmax_bike)
_ min(GBF_comp(xmin_bike_fast:xmax_bike));
GBF_cycle = GBF_cycle./max(GBF_cycle);
GBF_cycle = GBF_cycle*10000;
% t = 0:0.01:length(GBF_cycle)/100_0.01;
% figure
% plot(t,GBF_cycle)
% title('Bicycle: Single Cycle', 'FontSize', 16);
% ylabel('Scaled Values')
% xlabel('Time(s)')
%create matrix
%First create counter (x_values)
counter_bike = zeros(1,xmax_bike_xmin_bike_fast);
for i = 1:1:xmax_bike_xmin_bike_fast
counter_bike(i) = i;
end
%Create for loop to get ones and negative ones so we can check positive and
%negative match
matrix_cycle_bike = zeros(10000:length(counter_bike));
for i = 1:1:length(counter_bike)
for j = 1:1:10000
if j <GBF_cycle(i)
matrix_cycle_bike(10001_j,i) = 1;
else
matrix_cycle_bike(10001_j,i) = _1;
end
end
end
%Check to see if the for loop worked correctly then comment out
% figure
```

```
% image(matrix_cycle_bike.* 255);
% title('Bicycle: Ones and Negative Ones', 'FontSize', 16);
end
```

Read Stairs Signal

```
function [matrix_cycle_stairs] = stairs_cycle(GS_comp)
% Zoom into one cycle and create matrix (Biking)
% Zoom into elliptical cycle
xmin_stairs_fast = 12.08*100;
xmax_stairs_fast = 13.35*100 + 50;
xmax_stairs = uint16(xmax_stairs_fast);
GS_cycle = GS_comp(xmin_stairs_fast:xmax_stairs)
_ min(GS_comp(xmin_stairs_fast:xmax_stairs));
GS_cycle = GS_cycle./max(GS_cycle);
GS_cycle = GS_cycle*10000;
% t = 0:0.01:length(GS_cycle)/100_0.01;
% figure
% plot(t,GS_cycle,'b')
% title('Stairs: Single Cycle', 'FontSize', 16);
% ylabel('Scaled Values')
% xlabel('Time(s)')
%create matrix
%First create counter (x_values)
counter_stairs = zeros(1,xmax_stairs_xmin_stairs_fast);
for i = 1:1:xmax_stairs_xmin_stairs_fast
counter_stairs(i) = i;
end
%Create for loop to get ones and negative ones so we can check positive and]
%negative match
matrix_cycle_stairs = zeros(10000:length(counter_stairs));
```

```
for i = 1:1:length(counter_stairs)
for j = 1:1:10000
if j <GS_cycle(i)
matrix_cycle_stairs(10001_j,i) = 1;
else
matrix_cycle_stairs(10001_j,i) = _1;
end
end end
%Check to see if the for loop worked correctly then comment out
% figure
% image(matrix_cycle_stairs.* 255);
% title('Stairs: Ones and Negative Ones', 'FontSize', 16);
end
```

Turn Signal into Ones and Negative Ones

```
function [matrix] = ones_and_zeros(comp_signal)
%Make the long signal ones and negative ones
matrix = zeros(10000,length(comp_signal));
% For loop to create matrix of ones and zeros by pixel
for i = 1:1:length(comp_signal)
for j = 1:1:10000
if j/10000 <comp_signal(i)
matrix(10001_j,i) = 1;
else
matrix(10001_j,i) = _1;
end
end
end
% % Plot to check to see if the matrix waas created correctly
% figure
```

```
% image(matrix.* 255);
end
```

Take Derivatives

```
function [deriv_zeros] = derivatives(comp_signal)
%for loop to find where derivative <0
deriv_signal = diff(comp_signal);
% Create a for loop and find where the derivative is 0
m = 1;
for i = 1:length(comp_signal)_2
if deriv_signal(i) <0 && deriv_signal(i+1) >0
deriv_zeros(m) = i;
m = m+1;
end
end
deriv_zeros;
end
```

Find match

```
function [match] = find_match(super_signal_matrix,super_derivs,matrix_cycle)
% Find the length of the cycle that we'll be testing
cycle_length = size(matrix_cycle);
% Make super signal the right way
% super_signal_matrix = super_signal_matrix';
% Find amount of tests to try
for i = 1: length(super_derivs)
if super_derivs(i) <(length(super_signal_matrix(1,:))_cycle_length(2))
shortened_derivs(i) = super_derivs(i);
end
```

```
end
match = zeros(length(shortened_derivs),1);
%Multiply the matrices starting at the derivatives
for i = 1:length(shortened_derivs)
for j = 1:cycle_length(2)
for k = 1:10000
number = matrix_cycle(k,j)*super_signal_matrix(k,(shortened_derivs(i)+j_1));
if number = 1
match(i) = match(i) + 1;
end
end
end match(i) = match(i)/(10000*cycle_length(2));
end
end
```

Decide Which Exercise is Best

```
function [match_decide] = decide_exercise(match_walk,match_jog,
match_bike,match_ellip,cutoff,super_derivs)
q = 1;
%If two exercises both have a match take the higher exercise
match_decide = zeros(1,3);
for i = 1:length(match_bike)
if(match_walk(i) >cutoff —— match_jog(i) >cutoff —— match_bike(i) >cutoff —— match_ellip(i)
>cutoff)
if(match_walk(i) >match_jog(i) && match_walk(i) >match_bike(i) && match_walk(i) >match_ellip(i))
match_decide(q,1) = 1;
match_decide(q,2) = super_derivs(i);
match_decide(q,3) = match_walk(i);
q = q + 1;
elseif(match_jog(i) >match_walk(i) && match_jog(i) >match_bike(i) && match_jog(i) >match_ellip(i))
```

```
match_decide(q,1) = 2;
match_decide(q,2) = super_derivs(i);
match_decide(q,3) = match_jog(i);
q = q + 1;
elseif(match_bike(i) >match_walk(i) && match_bike(i) >match_jog(i) && match_bike(i)
>match_ellip(i))
match_decide(q,1) = 3;
match_decide(q,2) = super_derivs(i);
match_decide(q,3) = match_bike(i);
q = q + 1;
elseif(match_ellip(i) >match_bike(i) && match_ellip(i) >match_walk(i) && match_ellip(i)
>match_bike(i))
match_decide(q,1) = 4;
match_decide(q,2) = super_derivs(i);
match_decide(q,3) = match_ellip(i);
q = q + 1;
end
end
end
end
```

Find When Exercises Change

```
function [change_exercise] = changing_exercises(match_decide)
% Set the first exercise
change_exercise(1,:) = match_decide(1,:);
this_size = size(match_decide);
r = 2;
% Find where the exercise changes from one to the next
for i = 2:this_size(1)
if(match_decide(i,1)  = match_decide(i_1,1))
```

```
change_exercise(r,1) = match_decide(i,1);
change_exercise(r,2) = match_decide(i,2);
change_exercise(r,3) = match_decide(i,3)*10000;
r = r + 1;
end
end
end
```

Plot Exercises

```
function [partial_signal] = plot_exercises(change_exercise,super_signal)
% Create the time vector
t = 0.01:length(super_signal);
t = t./60/100;
partial_signal = cell(1,length(change_exercise(:,1)));
figure
% Plot the first bit of exercise as the first signal
i = 1;
if(length(change_exercise(:,1)) >1)
if(change_exercise(i,1) = 1)
plot(t(1:change_exercise(i+1,2)),...
super_signal(1:change_exercise(i+1,2)),'bl')
partial_signali = super_signal(1:change_exercise(i+1,2));
hold on
elseif(change_exercise(i,1) = 2)
plot(t(1:change_exercise(i+1,2)),...
super_signal(1:change_exercise(i+1,2)),'g')
partial_signali = super_signal(1:change_exercise(i+1,2));
hold on
% elseif(change_exercise(i,1) = 3)
% plot(t(1:change_exercise(i+1,2)),...
```

```
% super_signal(1:change_exercise(i+1,2)),'c')
% partial_signali = super_signal(1:change_exercise(i+1,2));
% hold on elseif(change_exercise(i,1) = 4)
plot(t(1:change_exercise(i+1,2)),... super_signal(1:change_exercise(i+1,2)),'r')
partial_signali = super_signal(1:change_exercise(i+1,2));
hold on
end
end
if(length(change_exercise(:,1)) >2)
for i = 2:length(change_exercise)_1
if(change_exercise(i,1) = 1)
plot(t(change_exercise(i,2):change_exercise(i+1,2)),...
super_signal(change_exercise(i,2):change_exercise(i+1,2)),'bl')
partial_signali = super_signal(change_exercise(i,2):change_exercise(i+1,2));
hold on elseif(change_exercise(i,1) = 2)
plot(t(change_exercise(i,2):change_exercise(i+1,2)),...
super_signal(change_exercise(i,2):change_exercise(i+1,2)),'g')
partial_signali = super_signal(change_exercise(i,2):change_exercise(i+1,2));
hold on % elseif(change_exercise(i,1) = 3)
% plot(t(change_exercise(i,2):change_exercise(i+1,2)),...
% super_signal(change_exercise(i,2):change_exercise(i+1,2)),'c')
% partial_signali = super_signal(change_exercise(i,2):change_exercise(i+1,2));
% hold on
elseif(change_exercise(i,1) = 4)
plot(t(change_exercise(i,2):change_exercise(i+1,2)),...
super_signal(change_exercise(i,2):change_exercise(i+1,2)),'r')
partial_signali = super_signal(change_exercise(i,2):change_exercise(i+1,2));
hold on
end
end
```

```
end
% Move i to the next (and last) point
if(length(change_exercise(:,1)) >1)
i = i + 1;
if(change_exercise(end,1) = 1)
plot(t(change_exercise(end,2):length(t)),super_signal(change_exercise(end,2):end),'bl')
partial_signali = super_signal(change_exercise(end,2):end);
hold off
elseif(change_exercise(end,1) = 2)
plot(t(change_exercise(end,2):length(t)),super_signal(change_exercise(end,2):end),'g')
partial_signali = super_signal(change_exercise(end,2):end);
hold off
elseif(change_exercise(end,1) = 4)
plot(t(change_exercise(end,2):length(t)),super_signal(change_exercise(end,2):end),'r')
partial_signali = super_signal(change_exercise(end,2):end);
hold off end
end
% Plot the signal if its only one exercise
if(length(change_exercise(:,1)) = 1)
if(change_exercise(1,1) = 1)
plot(t,super_signal,'bl')
partial_signal1 = super_signal;
elseif(change_exercise(1,1) = 2)
plot(t,super_signal,'g')
partial_signal1 = super_signal;
elseif(change_exercise(1,1) = 4)
plot(t,super_signal,'r')
partial_signal1 = super_signal;
end
end
```

title('Subject 3 Guessed Exercises','Fontsize',16)

xlabel('Time (min.)','Fontsize',14)

ylabel('Scales Values','Fontsize',14)

% legend('Walk','Jog')

% % Plot the signal with correct changes (only for match_cycle_to_comp.m)

figure

plot(t(1:3000),super_signal(1:3000),'bl',t(3001:6000),
super_signal(3001:6000),'g',... t(6001:9000),super_signal(6001:9000),'bl',
t(9001:12000),super_signal(9001:12000),'g')

title('Field Test Exercises','Fontsize',16)

xlabel('Time (min.)','Fontsize',14)

ylabel('Scales Values','Fontsize',14)

legend('Walk','Jog')

end

Convert Signal to Calories Burnt

function [factor] = signal_to_cal(actual_calories)

% Find multiplier

factor = actual_calories/(360000);

end

E.1.2 Find Total Error

function [total_cal_from_signal] =
percent_err_all(segmented_signal,change_exercise,walk_factor, jog_factor,bike_factor,stairs_factor)

% Find out how many calories were burnt

% Integrate signal and divide by factor by pieces

total_cal_from_signal = 0;

for i = 1:length(segmented_signal)

if(change_exercise(i,1) = 1)

```
total_cal_from_signal = length(segmented_signali)*walk_factor + total_cal_from_signal;
end
if(change_exercise(i,1) = 2)
total_cal_from_signal = length(segmented_signali)*jog_factor + total_cal_from_signal;
end
if(change_exercise(i,1) = 3)
total_cal_from_signal = length(segmented_signali)*bike_factor + total_cal_from_signal;
end
if(change_exercise(i,1) = 4)
total_cal_from_signal = length(segmented_signali)*stairs_factor + total_cal_from_signal;
end
end
% Add actual amount of calories (will only work for this setup)
total_cal_real = 6000*walk_factor+6000*jog_factor+6000*stairs_factor;
% Find percent error between the two tests
Percent_error = (abs(total_cal_from_signal _ total_cal_real)/total_cal_real)*100;
%Display the calories guessed from signal
format compact;
format short;
disp(' ')
name = 'Total calories derived from the signal';
X = [name, ' = ' num2str(total_cal_from_signal)];
disp(X)
name = 'Actual total Calories';
X = [name, ' = ' num2str(total_cal_real)];
disp(X)
disp(' ')
name = 'Percent Error (Calories)';
X = [name, ' = ' num2str(Percent_error)];
disp(X)
```

```
% Check the calories burnt from signal by exercise
% combined_signal = [segmented_signal1;segmented_signal2;segmented_signal3;segmented_signal4
;segmented_signal5;segmented_signal6;segmented_signal7;segmented_signal8;segmented_signal9];
% combined_signal = padarray(combined_signal,15000_14948,0,'pre');
end
```