Development of an Upper Extremity Fracture Model

Kimberly Maciolek (Leader) [1], Hope Marshall (Communicator) [1],
Kevin Beene (BWIG) [1], Gabe Bautista (BSAC) [1]

University of Wisconsin-Madison - Biomedical Engineering Department
Client: Dr. Matthew Halanski[2]
Advisor: Professor Thomas Yen[1]

[1]Department of Biomedical Engineering, University of Wisconsin-Madison
[2]Department of Orthopedics and Rehabilitation, University of Wisconsin
School of Medicine & Public Health

December 12, 2012

Table of Contents

1. Abstract

Bone fractures to the distal radius and ulna are one of the most common pediatric ailments. Casting is often preferred as the common treatment method for pediatric forearm fractures and is assumed to be safe. However, residents are expected to learn through trial and error or through textbooks. This can result in complications arising from cast application or removal. Therefore, a pediatric fracture simulator is needed to teach proper techniques of cast application, reduction, immobilization, and cast removal. The fracture simulator must provide immediate feedback to the user and monitor fracture reduction, force applied during three-point molding, and temperature of skin surface.

Components for the force, temperature and alignment sensors were evaluated using a design matrix. The final design will utilize latex surgical tubing to maintain the angulated position of the forearm fracture and will also include a pressure mapping system, thermistors, and finally flex sensors to indicate proper alignment. Additional design components include an aluminum screen double-layer to protect the sensors, inductive sensors to measure cast saw location relative to skin, and tissue-mimicking materials for skin, soft tissue and bone. After successfully verifying the compatibility of all of the material and electronic components, the complete final prototype must now be fabricated, including integrating the sensors with a simple user interface. Once the client and other medical professionals have tested the final device and all appropriate adjustments have been made, the design will be validated through use by casting experts at the Pediatric Orthopedic Society of North America (POSNA) national meeting in spring.

2. Introduction

2.1. Occurrence of Pediatric Fractures

Bone fractures are a common pediatric injury due to the porous developing skeleton of young children. Nearly one-third of all children suffer at least one fracture before the age of 17 [1]. Additionally, forty percent of all pediatric fractures involve the forearm [2]. These fractures represent 9% of all injuries reported to health professionals [1].

Children’s bones are often more flexible than adult bones and when impacted, a child’s bones tend to bend and only partially break. This is known as a greenstick fracture, as seen in Figure 1A, and is the most common type of distal radial and ulnar fracture in pediatric patients. Such fractures are often treated via casting to immobilize the fracture, as seen in Figure 1B.

Figure 1: A. X-ray of green stick fracture. B. X-ray of properly reduced green stick fracture [3].

2.2. Casting Process

Before the physician begins the casting process, it is important that an x-ray image is taken of the fracture so as to confirm whether or not casting is the best treatment option or if surgery should be considered. In pediatrics, the most common treatment for greenstick fractures is known as reduction to reverse deformity. First, the healthcare provider must correct or restore length, rotation, and angulation of the ulna and radius. This process often requires exaggeration or reversal of mechanism of injury. In pediatric fractures with an intact periosteum, the soft tissues interfere with the convex side of the fracture [4].

Next, the forearm is covered with a stockinette and padding and the bones are aligned using three-point molding (Figure 2). A plaster or fiberglass layer is then applied to maintain the pressure. While plaster is most commonly utilized among pediatric orthopedists due to its ease of molding, fiberglass is much stronger, lighter and more durable.

It is important for the physician applying the cast to mimic the limb they are immobilizing as closely as possible to preserve proper reduction of the fracture [2]. According to Chess et al., casts that successfully maintained reductions had sagittal to coronal ratio (or "cast index") near 0.7 [5]. Therefore, the cast cross section should be an oval rather than a circle [2].

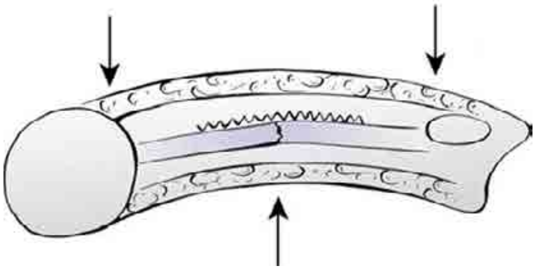

Figure 2: Sketch of three-point molding [3]

Proper reduction of the fracture is confirmed with additional x-rays. Acceptable angulation is less than 15 degrees but more angulation is allowed in younger children and preferred over a surgical option. Rotational alignment is verified by observing minimal distance in bone widths at fracture edges [4].

After four to six weeks, that cast is removed. A cast saw is used to cut two straight lines down either side of the cast by moving the saw in and out with brisk movements. The cast saw contains an oscillating blade that moves about a quarter of an inch and is designed to cut only hard objects. Therefore, it only will cut the cast padding or skin if excess pressure is applied. Once cut, cast spreaders are used to increase the size of the opening created by cast saw. Finally, a blunt tipped scissors is used to cut through the padding and stockinette. Upon complete removal of the cast, the skin should be inspected for any injury [6].

2.3. Dangers of Casting

While casting is often assumed to be a relatively low-risk treatment option for forearm fractures, it has also become a lost art among physicians and medical residents, putting patients at an increased risk of injury. Cast immobilization is frequently overlooked as more and more residents are being trained on newer forms of treatment and surgical techniques. However, despite this shift in training, the conservative treatment of casting still remains to be one of the key forms of treatment for many orthopedic conditions. As most young physicians have come to learn casting techniques – both application and removal – through trial and error, casting has become an area of greater concern for orthopedists across the nation [2]. In fact, in a recent study of about 400 physicians from a large multispecialty, multi-location pediatric group of approximately 400 physicians, casting was the number one cause of litigation, and about 35% of the claims paid an average of $120,000 each [7].

When applying the cast to the immobilized limb, the physician should be concerned about potentially burning the patient's skin from the hot casting materials as well as failing to properly immobilize the limb. If the cast is applied too light, it becomes a rigid tourniquet that restricts proper blood flow to the limb. On the other hand, if the cast is applied too loosely, it will fail to hold the proper reduction and the fracture will heal improperly [2].

Complications with the cast removal process largely revolve around the cast saw and the potential burns and cuts that can result from incorrect cast saw usage. While there are several factors that can contribute to a cast saw burn, the most easily preventable factor is the technique used by the physician when cutting through the cast, further emphasizing the importance of proper training on methods of cast removal. Shuler and Grisafi explain that the cast saw blade should be removed from the cast material after each cut to allow for cooling of the blade. When the blade is not allowed to cool, the average temperature of the blade at any given time during the cast removal is about 41°F higher, putting the patient at an increased risk of obtaining a burn or abrasion [8].

Figure 3: Example of burns and cuts from cast saw [3].

2.4. Client's Current Model

The client currently utilizes a forearm model designed and constructed by a former design team (Figure 4). The model is made primarily of PVC pipes that are connected to a thick wood board measuring about 0.5 m x 0.5 m. The PVC pipes fit together in an L-shape, the tail end of which serves as the forearm and is wrapped in a thin layer of copper foil. This copper layer simulates the skin surface of the forearm. The physicians and training residents apply the casting materials to copper-coated PVC pipe and practice removing the cast with a cast saw, trying their best not to damage the copper in the process. The cast saw (not see in Figure 4) has temperature loggers on its blade that track the temperature of the blade throughout the duration of its use. While this model has proven to be sufficient at recording the cast saw blade temperature and displaying to the user whether they have cut through the copper layer, it fails to teach the user proper fracture alignment techniques, lacks the ability to monitor the amount of force the user applies to the forearm during immobilization, and does not track the temperature of the skin surface (in addition to the temperature of the saw blade).

Figure 4: Photo of client's current forearm model [9].

3. Problem Statement

The focus of this project is to construct a distal radial and ulnar fracture simulator to teach proper techniques of cast application, reduction, immobilization, and cast removal. The fracture simulator must provide immediate feedback to the user and monitors fracture reduction, force applied during three-point molding, and temperature of skin surface.

4. Design Specifications

Since the fracture simulator device will mimic a child's forearm, it should be 18cm in length and 5cm in width. The device must be reusable, easy and safe to transport, as it will be used as a training tool in various hospitals throughout the nation. The device must measure and display saw blade temperature and artificial skin temperature in real time. In addition, the device should measure the forces applied to the arm in real time and display them on a nearby monitor or laptop. The device must clearly indicate a successful fracture reduction, which is characterized by an angulation of less than 15 degrees and displacement of less than 2mm. The device must last at least 10 years when stored at room temperature and effective for training residents in the three point molding technique for fracture reduction as well as cast removal.

5. Design Components

5.1. Force Sensors

5.1.1. Design Options

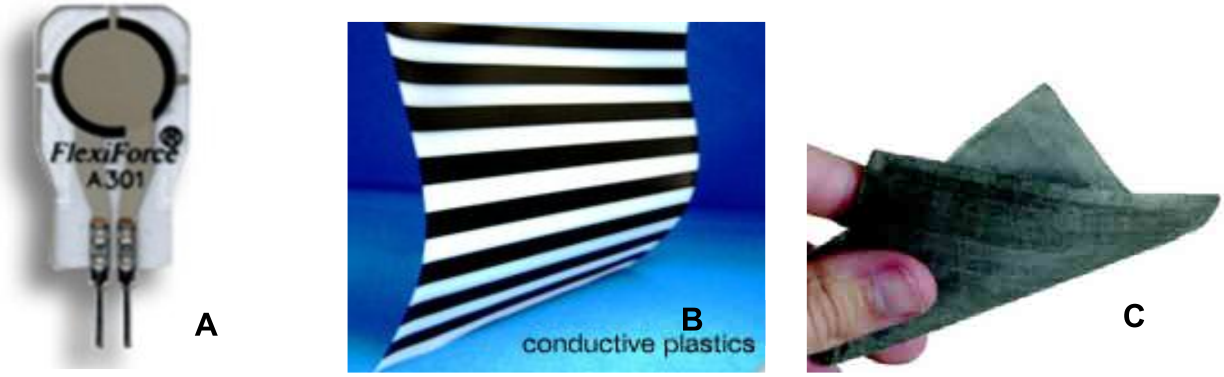

Figure 5: Possible pressure sensors. A) FlexiForce® Sensor [10]. B) Zebra™ Pressure Mapping Sensor [11]. C) TactArray Sensor [12].

The first alternative the team looked at for the pressure mapping required for this project was the use of individual force sensors (Figure 5A). These sensors are printed circuits and are therefore readily available at a low cost (~$15 a piece [10]). As force is applied to the sensor its conductance increases in a linear fashion and using the circuit shown below the force can be quantified as a voltage (Figure 6).

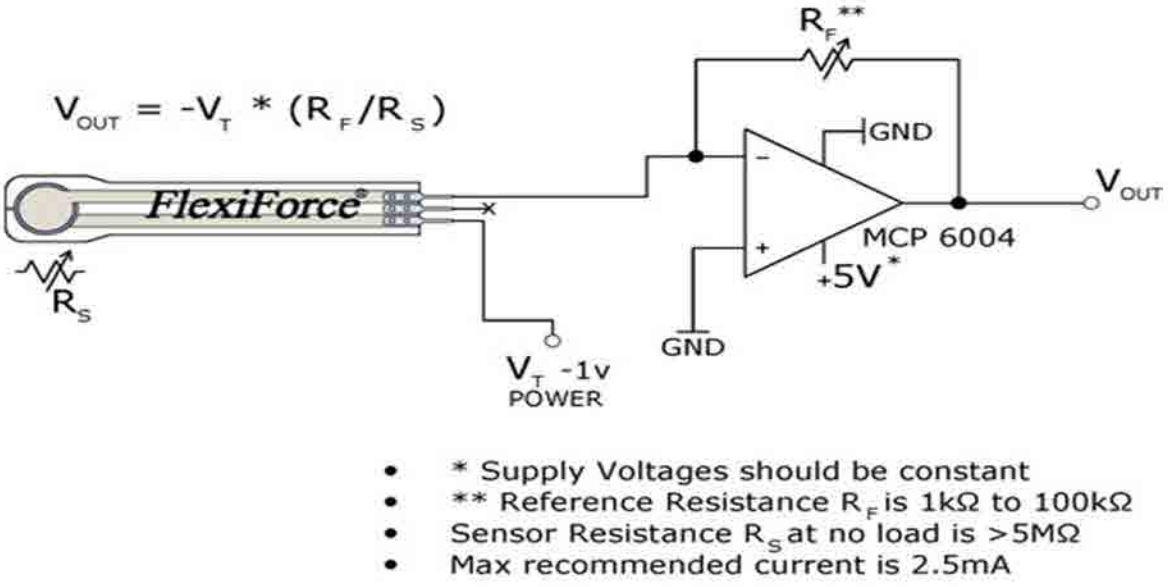

Figure 6: Recommended circuit for FlexiForce® Sensors [10]

By setting up several of these sensors in a matrix, it is possible to determine the pressure being applied to certain parts of the arm. This solution, however, would be limited in resolution by the number of sensors used and would require vast amounts of programing and circuit design in order to function properly.

The Zebra™ Pressure Mapping Sensor from SensorTech™ was the second option looked at for this aspect of the project (Figure 5B). This type of sensor is created by overlapping Zebra™ sheet sensors (sheets with parallel strips of piezo resistive material) at right angles (Figure 7). Each point where the strips overlap forms a sensor location capable of independently measuring pressure. This leads to sensors with a maximum spatial resolution of ~6 sensing points/in.^{2} with up to 1024 points. This type of sensor is stretchable and can be thermoformed or machined depending on its use and thickness [11]. SensorTech™ also provides its own DAS (Data Acquisition System) and software for pressure mapping.

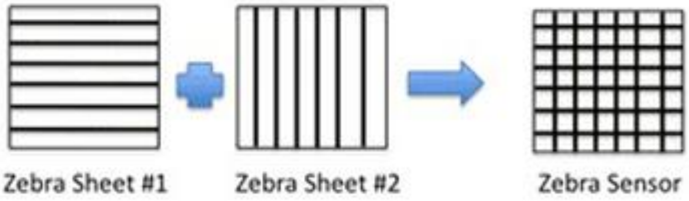

Figure 7: Zebra™ Sensor construction [11]

The last sensor considered for this part of the project was the TactArray line of pressure mapping sensors from Pressure Profile Systems (PPS) (Figure 5C). This type of sensor is built out of a flexible, conductive cloth that can be wrapped or molded in various shapes. They have a maximum resolution of 2mm with up to 10,240 elements. Just like SensorTech™, PPS provides a USB drive and real time visualization & acquisition software. However, the PPS software allows custom 3D scanning in order to show pressures on the actual shape to which the sensors are bonded [12].

5.1.2. Design Matrix

Table 1: Design Matrix for Force Sensors.

	Weight	TechScan (Individual Sensors)	Zebra Sensor (SensorTech)	Tactile Sensor (Pressure Profile)
Cost	20%	5	3	1
Compatibility	15%	1	4	4
Precision	30%	3	3	5
Resolution	25%	1	2	5
Ease of Use	10%	1	4	4
TOTAL:	100%	2.4	3	3.95

The three proposed sensors were assessed on five categories (cost, compatibility, precision, resolution, and ease of use) to determine the best option for this project (Table 1). Each of the sensors was assigned a score of one to five, with five being the highest, for each category. Since each category has a unique weight based on their relative importance, the scores from each category were multiplied by their respective factors. The final scores for each design were reached by adding the scores of each individual category.

Precision and resolution were considered the most important categories in the design matrix due to the client's requirement of being able to create and reproduce normalized data for resident training from Pediatrics experts. In these categories the TactArray sensor scored the highest possible score due to its very high resolution of 2mm and high precision. In the compatibility and ease of use categories the Zebra™ sensor and TactArray sensor scored very high due to the hardware and software that comes with them as well as for the fact that they can be wrapped or molded into various shapes. Finally, in the cost category, the single Flexiforce® sensors scored the highest due to their relatively cheap value.

After summing the scores for each of the designs, TactArray Sensor had the highest score, followed by the Zebra™ Sensor with the second highest score, and then the Flexiforce® sensor with the lowest score of the three alternatives. The TactArray Sensor was chosen as our final design.

5.2. Temperature Sensors

5.2.1. Design Options

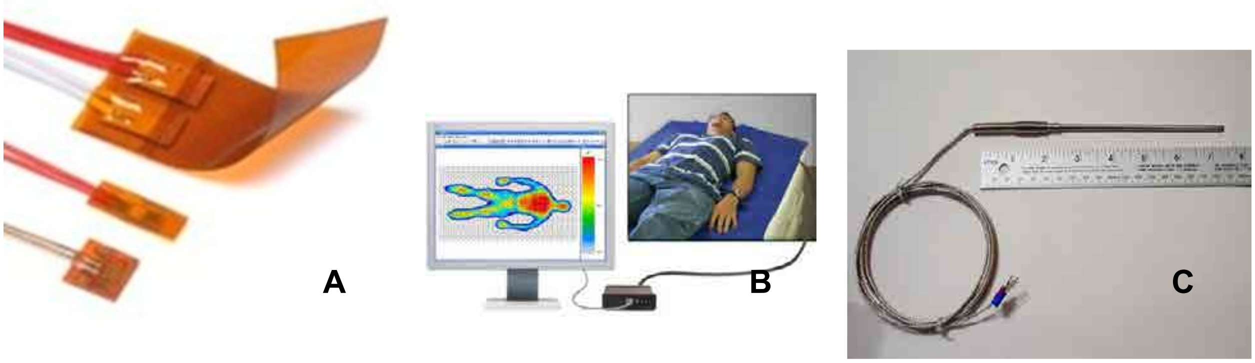

Figure 8: Possible temperature sensors. A) Thermal Ribbon Thermistors [13]. B) Tactilus Temperature mapping [14]. C) Thermocouple [15].

The team analyzed three different sensors in order to measure and display artificial skin temperature during cast application and removal, shown in Figure 8. The first design alternative is the thermal ribbon thermistors, which would be placed at various locations underneath the skin and protective sleeve of the fracture model simulator. These thermistors use semi-conductor materials to measure temperature by using electrical impulses and internal electrodes to sense surrounding heat. A decrease in thermistor resistance indicates an increase in temperature of the surroundings.

The second temperature sensor alternative is the Tactilus® temperature sensor system by Sensortech, which includes software that displays temperature mapping on a Windows computer. The mapping system is composed of sensing cells which cover the surface of the skin-like material which allow for temperature analysis at any point over the region of contact. These data are then captured from the sensor points and assimilated into a colorized temperature map display. Thermistor technology is used in the sensing cells and the mapping system is accurate up to +/-10% and has a customizable resolution of 9mm.

The final design alternative that the team considered is the thermocouple, which measures the difference in temperature between two points using two distinct materials. The difference in temperature from end to end creates a voltage between the two ends. A greater difference in temperature yields a higher current, and the temperature differential is calculated based on that.

5.2.2. Design Matrix

Table 2: Design Matrix for Temperature Sensors

	Weight	Thermal Ribbon Thermistors	Temperature Mapping	Thermocouple
Cost	35%	3	1	5
Compatibility	10%	2	4	2
Precision	10%	4	4	3
Resolution	25%	3	5	3
Ease of Use	20%	2	4	2
TOTAL:	100%	2.8	3.2	3.4

The three proposed sensors were assessed on five categories (cost, compatibility, precision, resolution, and ease of use) to determine the best option for this project (Table 2). Each of the sensors was assigned a score of one to five, with five being the highest, for each category. Since each category has a unique weight based on their relative importance, the scores from each category were multiplied by their respective factors. The final scores for each design were reached by adding the scores of each individual category.

Cost and resolution were considered the most important categories in the design matrix due to the wide range of differences in cost and the client's requirement of being able to measure and display temperature at any point on the artificial skin surface. Ease of use is also important, as the sensor data must be easy to interpret and display quickly. In these categories the Thermocouple sensor scored the highest possible score due to its high resolution and low cost. In the compatibility and ease of use categories the Tactilus® temperature mapping system and Thermocouple sensor had high scores. This was due to the software that comes with the sensor and the ease of reading and displaying temperature differential based on information provided by the thermocouple.

After summing the scores for each of the designs, the Thermocouple sensor had the highest score, followed by the Temperature mapping system with the second highest score, and then the Thermal Ribbon Thermistors with the lowest score of the three alternatives. The Thermocouple was chosen as the temperature sensor for the final design.

5.3. Alignment Sensors

5.3.1. Design Options

The client desires to have a real time indicator to determine correct reduction (i.e. proper positioning) of the bone. The initial though was a mechanical option were the bone snapped into place once enough force was applied. The client expressed that this would less ideal as he wants the person applying the cast to maintain proper pressure throughout the casting process. In addition, it was determined that it would be advantageous to run wires down the long axis of the bone model. Therefore, three other sensors were identified: an optical sensor, a complete circuit/button pressed, and a capacitive sensor.

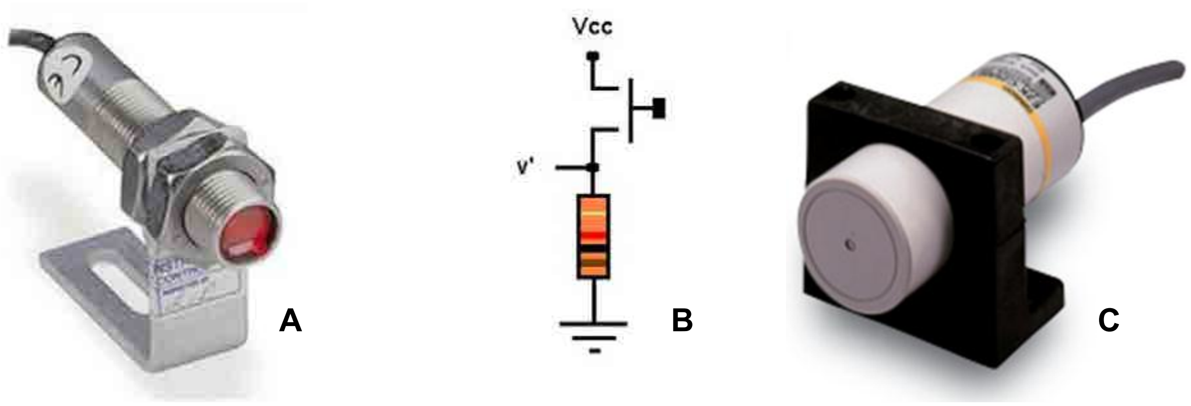

Figure 9: A. Optical sensor [16]. B. Complete circuit [17]. C. Capacitive sensor [18].

An example of an optical sensor is shown in Figure 9 A. This type of sensor functions by measuring an intensity change in one or more light beams and converting it to an electrical signal. Benefits of optical sensors include greater sensitivity, wide dynamic range, electrical passiveness, and no electromagnetic interference [19]. In the forearm model, one side of the “bone” would contain a light source at the plane of the fracture. The opposing face of the fracture would contain the optical sensor. Alignment of the bone pieces would be determined based upon the amount of light that reaches the sensor. Perfect alignment would produce the largest signal.

Another design is the capacitive sensor, as seen in Figure 9 C. This type of sensor functions based upon the electrical property of capacitance. When supplied with alternating current, the opposing sides of the sensor act as two flat conductive surfaces with an electric field between them. The amount of capacitance is directly proportional to the surface area of the sensor and the dielectric constant of the material between them and inversely proportional to the distance between the plates. The area of the sensors and the material between them are assumed to be constant. Therefore, the changes in capacitance result from a change in distance of the sensors. This can be measured as changes in voltage. Advantages of capacitive sensors include high resolutions, little sensitivity to material changes, inexpensive and small. Capacitive sensors do not function well in dirty or wet environments or when there is a large gap between the sensor and target [20]. For the forearm model, the capacitive sensors would be placed on the mating surfaces at the plane of the fracture. When perfectly aligned, the capacitive sensors would touch and there would be no voltage change (i.e. a complete circuit is created).

The final and most simple design option is the complete circuit as seen in Figure 9 B. This functions like the capacitive sensors but does not give information about the positioning of the bone pieces when not in contact. The circuit would also contain a light or signal on the user interface to alert the user when the bones are aligned.

5.3.2. Design Matrix

Table 3: Design Matrix for Alignment Sensors

	Weight	Optical Sensor	Complete Circuit	Capacitive Sensor
Cost	35%	1	5	2
Compatibility	20%	3	5	3
Precision	30%	2	2	4
Ease of Use	15%	3	5	3
TOTAL:	100%	2.00	4.10	2.95

A design matrix was created to compare the alignment sensors as seen in Table 3. The sensors were evaluated based upon cost, compatibility, precision and ease of use.

The optical sensor ranked the lowest. This option had the high cost (relative to other options) and a low precision due to possible inaccuracy resulting from light scattering inside the forearm model.

The capacitive sensor also ranked relatively low. Despite the high level of precision, this option had a high cost. It remains unclear how the gel tissue-mimicking components would affect the dielectric constant.

The complete circuit ranked the highest due to its simplicity. By nature, the circuit is composed of inexpensive parts resulting in a low total cost. The complete circuit option is also easy to assemble and highly compatible with materials and easy to incorporate into a user interface. This option was also supported by the client.

5.4. Protective Sleeve

In order to protect the many expensive sensors that our design will include, a metal mesh protective sleeve will be added directly below the skin-mimicking layer. The sleeve will be a stainless steel material that will prevent the cast saw blade from cutting through and damaging the force and temperature sensors. Since the material will be very thin and also made of metal, it should not drastically affect the output of the pressure and temperature sensors. However further testing of the compatibility of our materials in conjunction with one another will be performed in the future to determine any necessary adjustments.

5.5. Inductive Sensors

In order to detect the saw as it comes close to the skin during the cast removal simulation, inductive sensors will be used. Inductive sensors detect whether or not a metal object is near the sensor without touching the object. The inductive sensor as seen in Figure 10 consists of an induction loop, where a magnetic field is generated by electric current. As the metal object nears, power is consumed by the currents because of the induced resistance in the loop, which causes a change in energy. The inductance loop is paired with sensing circuitry, which communicates the change in energy as it is detected to another device [21].

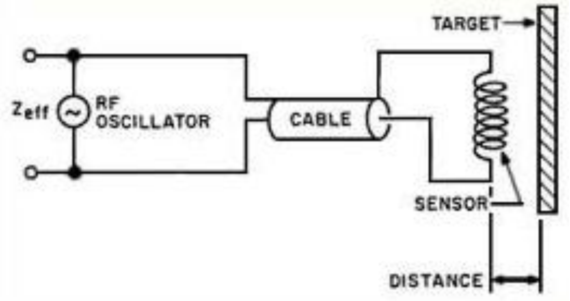

Figure 10: Circuit diagram of an inductive sensor [22].

5.6. Tissue-Mimicking Materials

In order to create a realistic fracture simulation device, the materials used for the skin, soft tissue and bone of the forearm must have realistic physical and mechanical properties and be compatible with the sensors used in the device.

5.6.1. Skin Material

Thin layers of low density Polydimethylsiloxane (PDMS) gel will be used to mimic the skin of the forearm. This material was recommended and has been shown to effectively mimic skin.

5.6.2. Soft Tissue Material

Either high density PDMS or Platsil Gel-10 will be used to mimic the soft tissue surrounding the bone. PDMS gel was recommended as an effective soft tissue material. Platsil Gel-10 is a translucent Platinum Silicone skin material that is easy to mold and shape, which will be necessary when molding for the final prototype [23].

5.6.3. Bone Material

Either acrylic or PVC will be used to mimic the bone for the fracture model simulator. Since the bone is the most internal element of the forearm model, it doesn't have to correspond to the exact shape of the radius and ulna bones as long as the correct amount of resistance can be applied in order to practice the three point molding technique for reducing the fracture.

6. Final Design

Based upon the design matrices, the final design is a combination of pressure mapping system, thermistors and complete circuit for alignment. The design also will include a metal mesh protective sleeve and tissue mimicking materials as seen in Figures 11 and 12.

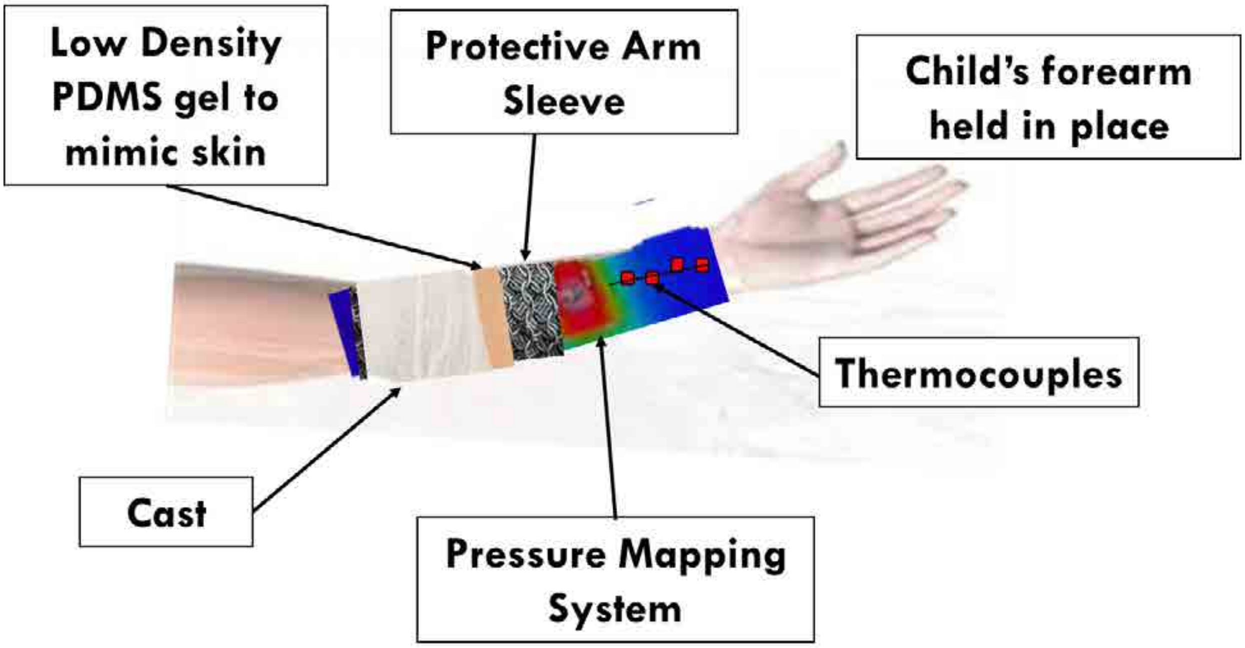

Figure 11: Diagram of forearm model with all design components.

The forearm model will be mounted on a vertical PVC post on a piece of plywood for stability as seen in Figure 12. The spring serves as resistance to create more realistic conditions for the fracture reduction. All electronic design components will be connected to a laptop based user interface to display alignment, pressure and temperature in real time.

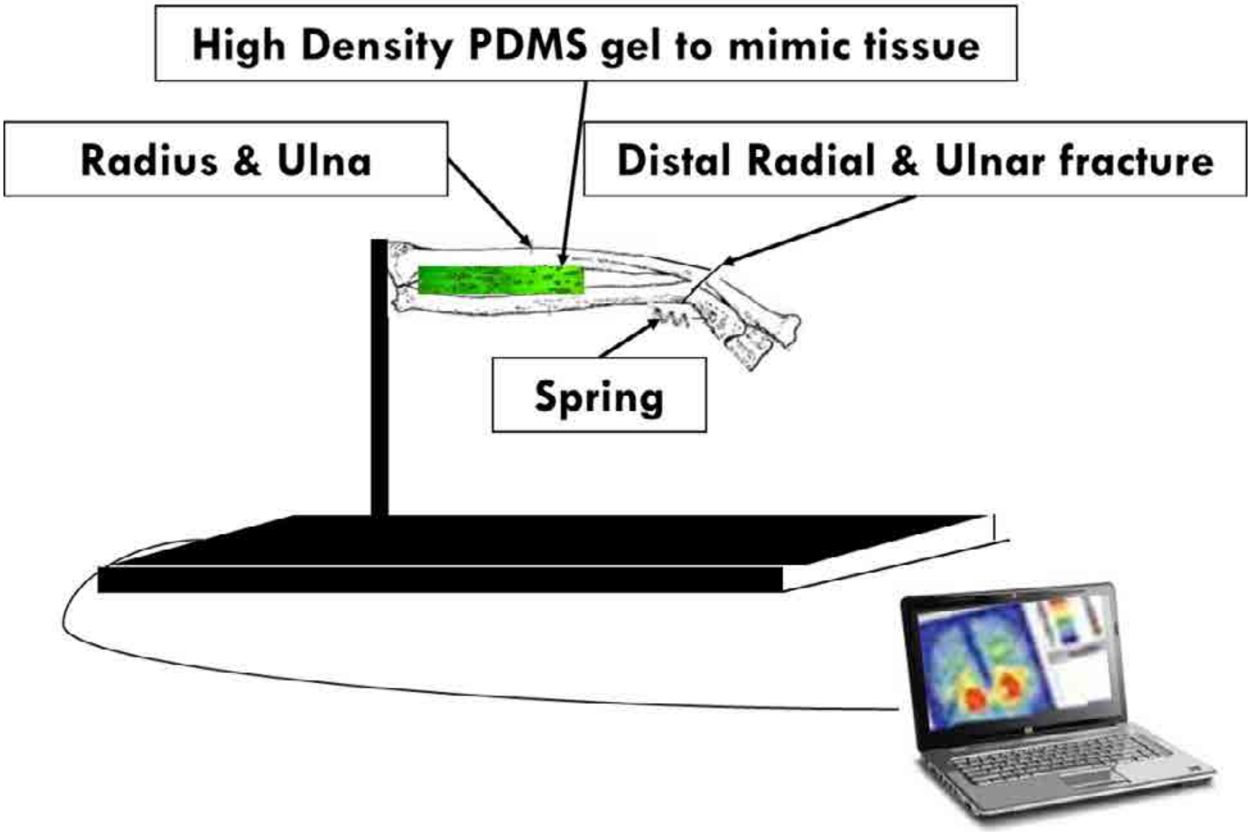

Figure 12: Diagram of entire casting model.

7. Design Change

Two mechanical designs were created to approximate the physiological conditions of a fracture. A prototype of a mechanical spring design was created. This design used both tension through a cable and threaded rod as well as a compressive spring down the long axis of PVC pipe “bones” to create a fracture. A second prototype was also developed in hopes of providing a point of comparison for future testing protocols. This second design consists of the same basic setup with PVC pipes but latex surgical tubing is used instead of the springs to hold the simulated forearm in the fracture position. The tubing essentially acts as large elastic bands that provide tension and resistance to the fracture. The client later tested these two different designs in order to assess their ability in accurately simulating a distal radial and ulnar fracture. The following section on design construction highlights the materials and methods used to fabricate this second design as well as the original spring prototype.

As well as introducing changes into the mechanics of the design, modifications were also made with the materials used in both designs. While the protective layer was initially going to be made from a stainless steel mesh sleeve, a much less expensive aluminum screen mesh will instead be used in the design. Also, flex sensors will be incorporated into the final prototype to test for proper reduction and alignment of the bones when using the three-point mold. These flex sensors change their resistance when they are bent and this resistance change will be monitored through a simple light indicator, which will alert the user about whether or not the fracture is properly aligned.

Lastly, force-sensing resistors will be used in the initial testing prototype to monitor the force that is being applied to the forearm model. The TactArray tactile sensor will still be used in the final design but due to the high cost of the mapping system, the client avidly supported the decision to delay the ordering of the TactArray until the final prototype is constructed next semester.

8. Design Construction

8.1. Internal Components

8.1.1. Spring Fracture Design

Figure 13: Photo of spring design.

The design used 1 inch (2.54 cm) outer diameter (OD) schedule 80 PVC water pipe to create the bone structure. The sections of pipe are 6 inches long. The frame is composed of a custom copper coupling. The coupling is secured to a part from an old bike seat. By simple adjustment with a 6 mm Allen wrench, the copper coupling and PCV pipes can be rotated a full 180° around the y-axis. A similar adjustment can also be made to pivot the arm 360° with respect to the fiberglass counter top base.

Inside the PVC pipe, a balance of tension and compression elements creates misalignment, resembling a fracture. A ¼-20 threaded rod is position down the long axis of the pipe section closest to the copper coupling. One end of the rod protrudes from the pipe (e.g. negative X-axis). A spring with washers on each end was fitted over the rod. The tension is adjusted by moving the set of locking nuts (wing nut and regular nut). The other end of the rod was hammered flat and a hole was drilled in the end. This hole is connected to a cable. The cable is then secured to the second PVC pipe section using a screw parallel to the X-Z plane.

A concentric pipe section is located at the mating ends of the fracture. This was glued into the interior surface of the 1 inch OD (2.54 cm OD) PVC pipe using epoxy. The interior pipe closest to the copper coupling also has a short section of 3/8 inch (0.9525 cm) threaded rod protruding from it. This serves as support for the compression spring that provides resistance to the fracture alignment. The spring is supported in the other PVC pipe by epoxy over the concentric pipe. A ball and socket joint was fabricated from welding rod and was attached using epoxy to the exterior of the PVC pipe underside (lowest part of the pipe in the Y-direction, parallel to the X-Z plane) to create a pivot point for the fracture.

8.1.2. Latex Tubing Fracture Design

This design also used two 6 inch (15.24 cm) sections of 1 inch (2.54 cm) outer diameter schedule 80 PVC water pipe to create the bone structure with the same adjustable frame and countertop base. In the current simple prototype, two ¼ inch (0.635 cm) inner diameter sections of latex surgical tubing, 10 inch (25.40 cm) in length run through the set of PVC pipes. Sets of holes were drilled 2 inches (5.08 cm) away from where the PVC overlaps with the frame and 1 inch (2.54 cm) from the joint of each PVC tube. Four slots 1/8 inch (0.3175 cm) in thickness are were sawed out so that the latex tubing is pinched in the slots and secured using a washer. The tubing is secured by washers at the set of holes nearest to the frame and runs externally parallel to the proximal base section (A). Near the joint, the tubing runs through the holes, across the joint, and back out through the set of holes nearest to the joint in the distal pipe.

In order to prevent the two pipes from overcorrecting during the simulated reduction or slipping, an internal lip was fabricated using a narrow hinge. Currently, the tension is adjusted by stretching the latex tubing, pinching it into the slots on the distal pipe and securing it with a washer. For this current prototype, there is no permanent or modular way to adjust the resistance of the fracture.

8.2. Tissue-mimicking materials

The PDMS skin layer was created by using a Sylgard 184 silicone elastomer kit, which contains a silicone elastomer base and curing agent. The base and curing agent were mixed in 3:1 and 5:1 ratios by weight in order to produce two different skin layers of varying toughness for testing purposes. The two mixtures were placed in individual 10 cm plates, heated at 80°C for 5 hours and then cooled.

Platsil Gel-10 was used for the muscle tissue layer, which will serve as the foundation on which the sensors will be embedded. The Platsil Gel-10 kit comes with two components (A and B), which are mixed in a 1:1 ratio, mixed thoroughly, and poured into the desired mold. For testing purposes, the mixture was poured into a plastic container and allowed to sit for approximately 1 hour until completely gelled.

8.3. Sensors

8.3.1. Pressure Sensors

The pressure sensor chosen for the final design was the TactArray pressure mapping sensor from Pressure Profile Systems. However, due to the high cost and long shipping time of this sensor the team decided to first test the design using a less expensive Force Sensing Resistor that has similar material properties as those of the TactArray sensor. The conductance of this sensor increases linearly when force is applied to it. By placing a resistor in series with the sensor and measuring the voltage across the resistor as shown below in Figure 14 it is possible to calculate the general range of the force being applied to the sensor.

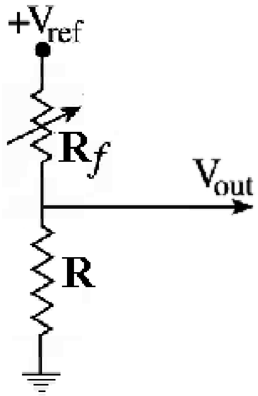

Figure 14: Circuit for the Force Sensing Resistor

8.3.2. Temperature Sensors

Although thermocouples were originally chosen as the best option for measuring temperature in this device, they are more effective at measuring very high temperatures which this device will not be submitted to. Therefore, the team chose to use thermistors for the final design, due to their accuracy in the 0 to 120 °C range. Thermistors will be placed in various locations near the surface of the device in order to measure temperature changes on the "skin" due to the application and removal of the cast. The circuit used to measure the change in the resistance of the thermistors is exactly like the one used for the Force Sensing Resistors, shown in Figure 14 above.

8.3.3. Alignment Sensors

In order to check alignment of the fracture the team will be using a pair Flex Sensors, shown below in Figure 14. The resistance of these sensors increases as they are bent which allows detection of any deflection in the device. Two sensors, one placed in the Sagittal plane the other in the Coronal plane of the device, provide the measurements necessary to assess if the fracture is set correctly, since rotation is not being taken into account. The circuit required for this device is the same as that required for the other sensors shown above in Figure 15. Any reduction in voltage signifies that the sensor is being bent and that the device is not properly aligned.

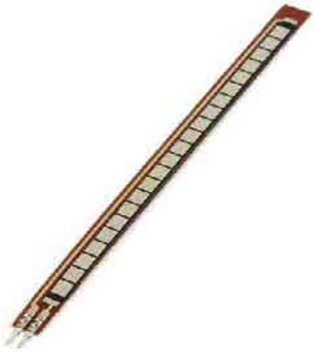

Figure 15: Flex Sensor

8.3.4. Interface

Data from these sensors will be collected with the use an Arduino microcontroller. The data will be processed on the Arduino itself and then output through Serial communication. A Graphical User Interface written in Java, shown below, allows users to connect with the Arduino and display the data. This Java program will be able to record all the data from the sensors in CSV format for later analysis of each trial with the device. As of right now, the GUI displays data from one of each of the different sensor types.

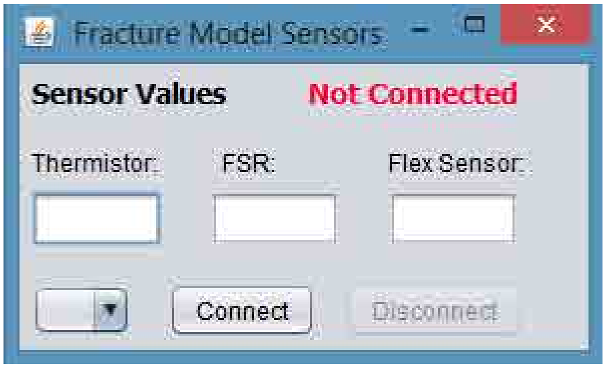

Figure 16: Graphical User Interface for the device

9. Testing and Results

9.1. Internal Resistance Component

9.1.1. Spring Constant-Spring Design

In order to evaluate the resistance of the mechanical fracture model component, the spring constant was calculated based on Hooke's Law:

$$F = -Kx$$

Here, F represents the force exerted, k is the spring constant and x is the displacement of the spring. Using a set of weights ranging from 200-1000 GM, the values for F and x were plotted and the line of best fit was determined, indicating the spring constant, k for the spring used. The calculated value for the spring constant is $k=-4.077\text{x}10^{-4}$N/m. Values for F and x are represented in the Table 4 below:

Table 4. Mass, force and displacement in spring constant calculation.

Mass (kg)	F (N)	x (m)
0	0	0
0.5	4.905N	0.001
1.0	9.81N	0.003
1.5	14.72N	0.005
1.7	16.67N	0.006

9.1.2. Spring Constant-Band Design

The same method was used in order to approximate the spring constant for the ¼” inner diameter elastic latex material below. After finding the line of best fit, the value of k was calculated as 8.0x10^-3 N/m, which is significantly higher than the spring constant for the spring design. The latex material, which stretches up to 700%, has much more stretch than he spring used in the alternative design.

Table 5. Mass, force and displacement in spring constant calculation.

Mass (kg)	F (N)	x (m)
0	0	0
0.5	4.905N	0.012
1	9.81N	0.033
1.5	14.72N	0.072
1.7	16.67N	0.088

9.2. Electronic Sensors

9.2.1. Temperature Sensor Calibration

The Thermistors for this device were calibrated using the Steinhart-Hart equation, where T is in Kelvin and R is in Ohms. Resistance values of the thermistors are calculated using the voltage divider equation. Three measurements at different temperatures were made in order to solve for the Steinhart-Hart coefficients. These coefficients are then used to calculate the temperature in the Arduino. Data collected, as well as the calculated coefficients, are shown below.

$$\frac{1}{T} = A + B\ln(R) + C(\ln(R))^3$$

Steinhart-Hart Equation

$$V_{\text{out}} = \frac{R_2}{R_1 + R_2} \cdot V_{\text{in}}$$

Voltage Divider Equation

Table 6. Temperature Sensor Calibration Data

Temperature (°F)	Temperature (K)	Voltage (V)	Resistance (Ω)
93.8	307.5	2.91	7200.0
77.0	298.15	2.50	1000.0
106.0	314.3	3.20	5625.0

Table 7. Steinhart-Hart Coefficients

A: 0.0029319431	B: -0.00118805	C: 0.000006401

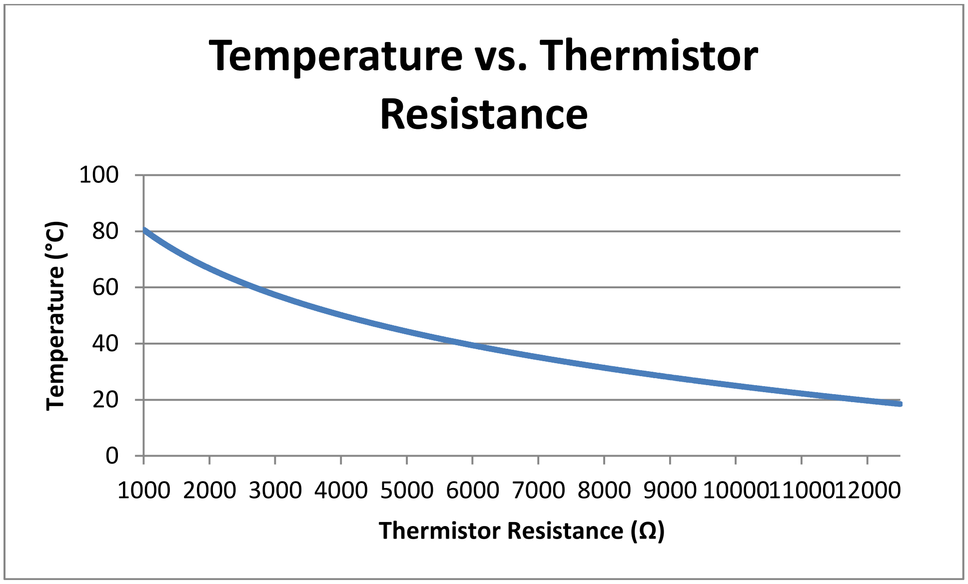

Figure 17. Steinhart-Hart Graph for device Thermistors

9.3. Tissue-Mimicking Materials Testing

9.3.1. Qualitative Analysis

In order to test the effectiveness of the skin and tissue layers, the client used the cast saw to attempt to cut the through the skin and protective layers. The client found that though both layers appropriately modeled the skin of a forearm, it was relatively easy for the cast saw to cut through the 5:1 PDMS skin layer. However, the client was unable to cut or damage the 3:1 PDMS skin layer with the cast saw. While the sensors were not damaged in either case, the client believed that the 3:1 PDMS skin layer should be used in the final prototype since it more effectively protected both the aluminum screen and various sensors. The client was also very satisfied with the PlatSil Gel-10 tissue layer and, having used this material in the past, was confident that it would more than accurately simulate real muscle tissue.

9.4. Protective Material Testing

9.4.1. Qualitative Comparison: Nylon vs. Aluminum

Both Nylon and Aluminum screen material were purchased in small 1 foot square (0.1 square meter) sections in order to test the effectiveness of each material as a protective layer over the sensors and tissue-mimicking material. When the Nylon was doubled and wrapped around a PlatSil Gel-10 arm prototype, the saw used for cast removal cut through the Nylon screen material. When the aluminum screen material was doubled and wrapped around the same arm prototype, the aluminum screen material protected the arm from harm. Aligning the two layers of aluminum screen directly on of one another was compared to an offset between the two layers (see Figure 18). The red and black coloring is used for visual clarity but both represent the material for the aluminum or nylon screen. The offset layering allowed for greater protection from the saw, especially for the aluminum screen material. The aluminum material in the offset arrangement was therefore chosen as the protective sleeve material for the test design strip.

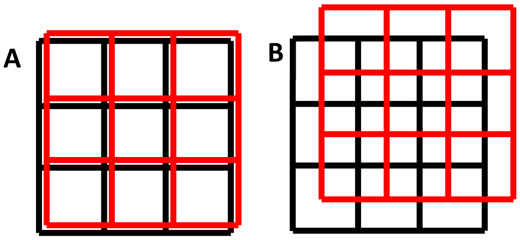

Figure 18: Screen Alignment. A. Overlapping layers. B. Offset layers.

9.4.2. Qualitative Integrated Analysis

After developing the various sensor circuits and interface, a qualitative integrated test was performed in order to evaluate the effectiveness of the protective aluminum screen material with the sensors present on the PlatSil Gel-10 prototype. In this test, the offset position of the aluminum screen protective material effectively protected the sensors from damage. This material with be used in the final design next semester.

10. Future Work

Since all of the sensors and materials have been successfully tested for compatibility and the latex surgical tubing design has been opted for over the original spring design, the next stages of the design process include fabrication of the final prototype, further testing, and continue adjustments and modifications. However, the two ratios of PDMS gel used to mimic skin must be tested for temperature conduction to ensure that the sensors will be able to detect the temperature changes two layers above the sensors.

The fracture forearm model will be composed of two small-diameter PVC pipes (one representing the radius, the other the ulna) and latex surgical tubing to hold the PVC pipes in an angulated fracture position. A pediatric forearm mold using the Platsil Gel-10 will be crafted after the client provides the desired dimensions of the model. As well, the TactArray tactile sensor system will need to be purchased. Further communication with the company will be necessary so that the pressure profile is manufactured specifically for this model. While significantly more expensive than other force sensors, the TactArray will provide top-of-the-line mapping of the forces applied to the forearm and be able to display the information in a user-friendly manner. A crank system to modulate the resistance of the surgical tubing must be designed and tested as well to complete the mechanical component of the design.

Once the final device has been fabricated, the client will conduct further testing to assess how consistent the model is with real forearm fractures as well as how informative the sensors and displays are to the user. At this point, the appropriate adjustments will be made to the design before obtaining normative data from other pediatric orthopedic surgeons at the Pediatric Orthopedic Society of North America (POSNA) national meeting in May 2013.

11. Budget

The client allotted no more than $20,000 for the completion of this project. Current preliminary cost estimates are seen in Tables 8 and 9 below. The team is still inquiring about cost for the pressure mapping device from Pressure Profile Systems.

Table 8: Cost Analysis

Material	**Quantity**	**Cost**
Plywood Base [24]	1	$5
PVC Pipes [24]	1	$0
Thermistor [35]	3	$3.24
Force Sensing Resistor [36]	1	$20
Arduino Mega Microcontroller[27]	1	$47.99
Arduino Starter kit [32]	1	$22.50
Protective Sleeve material (33)	2 square ft.	$0.85
PDMS [31]	500 grams	$60
PlatSil Gel-10 [23]	2 lbs	$47
USB A-B Cable [34]	1	$4.00
1/4” ID Latex Surgical Tubing	5 ft	$13.50
Prewrap material	1 roll	$5
Flex Sensor [37]	2	$24.90
Miscelaneous Mechanical Components	-	$0
	TOTAL:	$248.98

Table 9: Future Cost Analysis

Additional Materials	**Quantity**	**Cost Estimate**
PVC Pipes [24]	1	$20
Thermistors[35]	12	$13.00
Protective Sleeve material [33]	10 square ft.	$5.00
PlatSil Gel-10 [23]	10 lbs	$250.00
1/8” ID Latex Surgical Tubing	10 ft	$27.00
Flex Sensors [37]	6	$75.00
Pressure Mapping system[]	1	$4500
Miscelaneous Mechanical and electrical Components	-	$30
	TOTAL:	$4920.00

*Note: The total cost of the fracture simulator based on future cost analysis is: $5168.98, which is within the budgetary limits.

12. Timeline

Table 10: Projected Schedule

	September				October				November					December				January				February				March					April				May	
Task	7	14	21	28	5	12	19	26	2	9	16	23	30	7	14	21	28	4	11	18	25	1	8	15	22	1	8	15	22	29	5	12	19	26	3	10
Project R&D																																				
Background Research	🟦 X	🟦 X	🟦 X																																	
Design Brainstorm		🟦	🟦 X	🟦 X	🟦 X	🟦																														
Final Design Selection					🟦	🟦 X															🟦	🟦	🟦													
Manufacturing					🟦	🟦	🟦	🟦 X	🟦 X	🟦 X	🟦 X	🟦 X	🟦 X								🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦				
Testing												🟦 X	🟦 X	🟦 X	🟦 X						🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦				
Deliverables																																				
Progress Reports	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X	🟥 X						🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
PDS	🟥 X	🟥 X	🟥																		🟥	🟥	🟥													
Fall Mid-semester Presentation					🟦 X	🟦 X																														
Fall Mid-semester Paper					🟦 X	🟦 X	🟦 X																													
Fall Final Poster												🟦	🟦 X	X	X																					
Fall Final Paper												🟦	🟦 X	🟦 X	X																					
Spring Mid-Semester Presentation																									🟦	🟦										
Spring Mid-Semester Paper																									🟦	🟦	🟦									
Spring Final Poster																																🟦	🟦			
Spring Final Paper																																🟦	🟦	🟦		
Meetings																																				
Team	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X	🟩 X						🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Advisor	🟩	🟩 X	🟩 X	🟩	🟩 X	🟩 X	🟩	🟩	🟩 X	🟩 X	🟩 X	🟩	🟩 X	🟩 X	🟩						🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Client	🟩 X		🟩		🟩	X	🟩		🟩		🟩		🟩 X	X	🟩 X						🟩		🟩		🟩		🟩		🟩		🟩		🟩		🟩	
Website																																				
Updates	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X	🟪 X						🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪

The schedule for the academic year is seen in Table 10. The goal of the fall semester was to integrate all of the sensors and construct the forearm fracture. The spring semester will involve testing and design modification as needed.

13. Acknowledgements

A special thanks to our client, Dr. Matthew Halanski, for proposing this project and our advisor, Dr. Thomas Yen, for his continued insight and support throughout this project.

14. Bibliography

[1] Erik M Hedström, Olle Svensson, Ulrica Bergström, and Piotr Michno. "Epidemiology of fractures in children and adolescents: Increased incidence over the past decade: a population-based study from northern Sweden." *Acta Orthopaedica* 2010; 81 (1): 148–153.

[2] Halanski M, Noonan KJ. Cast and splint immobilization: complications. *J Am Acad Orthop Surg* 2008 January;16(1):30-40.

[3] Photos. Dr. Matthew Halanski.

[4] Duke Orthopaedics. "Green Stick Frx of Forearm." *Wheeless' Textbook of Orthopaedics.* Online: http://www.wheelessonline.com/ortho/green_stick_frx_of_forearm

[5] Chess DG, Hyndman JC, Leahey JL, Brown DC, Sinclair AM. "Short arm plaster cast for distal pediatric forearm fractures." *J Pediatr Orthop* 1994 March;14(2):211-3.

[6] Queen's School of Medicine. "Cast Applicatoin." Online: http://meds.queensu.ca/courses/assets/modules/cast-application/step_3_cut_through_padding.html

[7] Killian, J., White, S., & Lenning, L. (1999). Cast-saw burns: comparison of technique versus material versus saws. Journal of Pediatric Orthopedics, 19(5), 683-687.

[8] Shuler, F., & Grisafi, F. (2008). Cast-saw burns: evaluation of skin, cast, and blade temperatures generated during cast removal. The Journal of Bone and Joint Surgery, 90, 2626-2630. doi: 10.2106/JBJS.H.00119

[9] Photo. Kevin Beene.

[10] TekScan (Individual Sensors). www.tekscan.com.

[11] Zebra Sensor (SensorTech). www.sensortechcrop.com.

[12] Tactile Sensor (Pressure Profile). www.pressureprofile.com.

[13] Thermal Ribbon Thermistors. www.directindustry.com.

[14] Temperature Mapping. www.sensorprod.com.

[15] Thermocouple. www.auberins.com.

[16] Optical Sensor. Photo. http://dc190.4shared.com/doc/Fo4NuAQE/preview.html.

[17] Complete Circuit. Photo. www.acroname.com.

[18] Capacitive Sensor. Photo. www.ia.omron.com.

[19] "Optical Sensor Technologies." http://www.wtec.org/loyola/opto/c6_s3.htm.

[20] "Capacitive Displacement Sensors - An Overview." Lion Precision. http://www.lionprecision.com/capacitive-sensors/index.html

[21] “Inductive Proximity Sensors.” Rockwell Automation. http://ab.rockwellautomation.com/Sensors-Switches/Inductive-Proximity-Sensors.

[22] Inductive Sensors. www.archives.sensorsmag.com

[23] John Kao. Email correspondence. Inquiry about tissue-mimicking materials.

[24] Menards. www.menards.com.

[25] “SOMSO Radius and Ulna.” Global Technologies. http://www.gtsimulators.com/product-p/qs19-3.htm?gclid=CPjKl9fHl7MCFRBgMgodTmYA_w.

[26] “Thermocouple Type-K Glass Braid Insulated.” Ada Fruit Industries. http://adafruit.com/products/270

[27] “Thermocouple Amplifier MAX31855 breakout board (MAX6675 upgrade) - v1.0” Ada Fruit Industries. http://www.adafruit.com/products/269

[28] “Cut Resistant Sleeves.” Bunzl Processor Division. https://www.bunzlpd.com/product.php?productid=193633&cat=991415&page=1

[29]” Sammons Preston 834003 3" Plaster Cast Material.” Medical Solutions. http://www.4mdmedical.com/3-plaster-cast-material.html?gclid=CKfDn5jKl7MCFStgMgodgFsAfQ

[30] “Poly(dimethylsiloxane) viscosity 0.65 cSt (25 °C)” Sigma-Aldrich Co LLC. http://www.sigmaaldrich.com/catalog/product/aldrich/469300?lang=en&region=US

[31] “Poly(dimethylsiloxane) viscosity 1.0 cSt (25 °C).” Sigma-Aldrich Co LLC. http://www.sigmaaldrich.com/catalog/product/aldrich/469319?lang=en&region=US#

[32] Arduino Starter Kit (x1, $22.50) http://www.amazon.com/Arduino-Sidekick-Basic-Kit/dp/B007B14HM8/ref=pd_sim_pc_7#productPromotions

[33] Protective Screen Material-Aluminum Screening

http://www.qualitywindowscreen.com/store/aluminum-screen-wire-72x100-full-roll-p-190.html?gclid=CJOb7NfNkbQCFY9DMgodaHEA-A&zenid=3d6c2d86c2c8ad367ef22a8af47ef169

[34] USB A-B Cable http://www.adafruit.com/products/62

[35] Digikey Thermistor http://www.digikey.com/product-detail/en/USP12397/615-1093-ND/2715994

[36] Force Sensing Resistor

http://www.adafruit.com/products/1075

[37] Flex Sensor http://www.adafruit.com/products/182

15. Appendix

15.1. Product Design Specifications

Development of an Upper Extremity Fracture Model (fracture_model)

Product Design Specifications

Kim Maciolek (Team Leader), Hope Marshall (Communicator),
Kevin Beene (BWIG), Gabe Bautista (BSAC)
October 23, 2012

Function:
Currently residents learn casting techniques by trial and error on pediatric patients. Our client, Dr. Matthew Halanski, requires a forearm fracture model that teaches safe cast application and removal techniques. The design should incorporate sensors for temperature, pressure and contact (i.e. laceration) along the simulated limb to assess safe fracture reduction. The data obtained will be used to develop self-teaching modules in which persons using the model will be able to compare their casting techniques with those of the experts.

Client requirements:
Our client desires a device that:

- Is portable and reusable
- Clearly indicates successful fracture reduction
- Measures temperature and force applied to forearm
- Detects cast saw contact with skin surface

Design requirements:
1. Physical and Operational Characteristics

a. *Performance requirements*: The device should be reusable, easily sanitized, and teach safe placement and removal of a cast for a distal radius and ulna fracture. The device should be easily transportable for training purposes in various hospital environments. The device should indicate when fracture is properly reduced.

b. *Safety*: The device cannot harm training residents or doctors using the device as a teaching tool. Device should be stored safely including protection from all circuits or electronic components.

c. *Accuracy and Reliability*: The device should mimic common fracture conditions including the force necessary to immobilize the fracture. The device should be accurate enough to allow the same person to complete the casting procedure and achieve the same results.

d. *Life in Service*: Design must be reusable and last at least 10 years in a hospital setting.

e. *Shelf Life*: Device should be stored at room temperature and atmospheric pressure. Device may require battery changes.

f. *Operating Environment*: The device should be used at room temperature, atmospheric pressure, low humidity, in a dry environment with minimal dirt or dust. The device should be handled by healthcare professionals and students. The simulated limb should withstand forces necessary for standard casting procedures.

g. *Ergonomics*: The device must be easily used by one or multiple medical professionals at a time. A single user should be able to view pressure readings and thermal readings while using the device without the help of another user. The user should be notified when the simulated limb fracture has been correctly immobilized or when the cast saw is in close proximity to the skin without the help of another user. The device should be easily maneuverable by one person including transportation to and from various hospital environments.

h. *Size*: The model, in its entirety, should be no larger than 1 cubic meter.
i. *Weight*: The maximum weight of the device should be no more than 5 kg.
j. *Materials*: The device should be composed of materials that have similar properties to those of a human forearm. Materials should be durable to allow for repeated use.
k. *Aesthetics*, *Appearance, and Finish*: Model should be aesthetically pleasing to users and easy to identify as a forearm cast placement and removal training tool.

2. Production Characteristics

a. *Quantity*: One functional device is needed.
b*. Target Product Cost*: Total production cost should be less than $20,000.00.

3. Miscellaneous

a. *Standards and Specifications*: IRB approval required prior to testing.
b. *Customer*: Functionality and multi-faceted uses as an all-encompassing training model are the main priorities for the client. Client is especially concerned with measuring blade temperature, preventing accidental cutting of wires, and visually displaying if fracture is reduced correctly.
c. *Patient-related concerns*: Device should be easy to sterilize as it will be used in a hospital setting.
d. *Competition*: Currently, devices on the market only include only a few aspects of the desired device: simulated limb, pressure monitoring, thermal monitoring, user-friendly monitor and use as a teaching tool for application and removal of a cast. This device will be unique in its ability to combine all aspects.

15.2. Arduino Code

```
#include <math.h>

char tV[5];
char rV[5];
char fV[5];

void setup(){
  Serial.begin(9600);
  pinMode(22, OUTPUT);
  pinMode(23, OUTPUT);
}

void loop(){

  float thermistorVoltage = analogRead(A0) * (5.0 / 1023.0);
  float fsrVoltage = analogRead(A1) * (5.0 / 1023.0);
  float flexVoltage = analogRead(A2) * (5.0 / 1023.0);

  float temp0 = vToC(thermistorVoltage);

  String t = "T";
  String r = "R";
  String f = "F";

  dtostrf(temp0,1,1,tV);
  dtostrf(fsrVoltage,1,2,rV);
  dtostrf(flexVoltage,1,2,fV);

  t+=tV;
  r+=rV;
  f+=fV;

  if(flexVoltage < 2.6){
    digitalWrite(23,LOW);
    digitalWrite(22,HIGH);
  }
  else{
    digitalWrite(23,HIGH);
    digitalWrite(22,LOW);
  }

  Serial.println(t);
  Serial.println(r);
  Serial.println(f);

  delay(500);
}

//converts thermistor voltage to temperature in Celsius
float vToC(float voltage){

  float temp;
  float resistance;
```

```
  //Steinhart-Hart Coefficients
  float a = 0.0029319431;
  float b = -0.00118805;
  float c = 0.000006401;

  //Thermistor Resistance
  resistance = ((5.0*10000.0)/voltage) - 10000.0;

  //Temperature calculation using Steinhart-Hart equation
  temp = log(resistance);
  temp = a+b*temp+c*(temp*temp*temp);
  temp = 1/temp;
  temp = temp - 273.15; //Convert from Kelving to Celcius

  return temp;
}
```

15.3. Java-Serial Reader Code

SerialRead.java

```
import gnu.io.*;

import java.io.IOException;
import java.io.InputStream;
import java.util.Enumeration;
import java.util.HashMap;
import java.util.TooManyListenersException;

/**
 * Class that connects to serial port to read incoming data, also instances GUI
 * to display said data
 *
 * @author Gabriel Bautista
 */
public class SerialRead implements SerialPortEventListener
{
    //Passed from main GUI
    GUI window = null;

    //Contains the ports that will be found
    private Enumeration<?> ports = null;

    //Map the port names to CommPortIdentifiers
    private HashMap<String, CommPortIdentifier> portMap =
            new HashMap<String, CommPortIdentifier>();

    //Contains the opened port
    private CommPortIdentifier selectedPortIdentifier = null;
    private SerialPort serialPort = null;

    //Input stream for receiving data
    private InputStream input = null;

    //The timeout value for connecting with the port
    final static int TIMEOUT = 2000;

    //Boolean flag set if connected
    private boolean bConnected = false;

    //String that will hold data from input stream
    private String display = "";

    public SerialRead(GUI window)
    {
        this.window = window;
        window.btnDisconnect.setEnabled(false);
        window.btnConnect.setEnabled(true);
        window.comPorts.setEnabled(true);
    }

    //Searches for all the serial ports
    public void searchForPorts()
    {
        ports = CommPortIdentifier.getPortIdentifiers();
```

```
        while (ports.hasMoreElements())
        {
            CommPortIdentifier curPort =
(CommPortIdentifier)ports.nextElement();

            //Get only serial ports
            if (curPort.getPortType() == CommPortIdentifier.PORT_SERIAL)
            {
                window.comPorts.addItem(curPort.getName());
                portMap.put(curPort.getName(), curPort);
            }
        }
    }

    //Connects to the selected port
    public void connect()
    {
        String selectedPort = (String)window.comPorts.getSelectedItem();
        selectedPortIdentifier = (CommPortIdentifier)portMap.get(selectedPort);

        CommPort commPort = null;

        try
        {

            commPort = selectedPortIdentifier.open("TigerControlPanel",
TIMEOUT);

            serialPort = (SerialPort)commPort;

            window.btnDisconnect.setEnabled(true);
            window.btnConnect.setEnabled(false);
            window.comPorts.setEnabled(false);
            window.jLabel5.setForeground(new java.awt.Color(0, 0, 255));
            window.jLabel5.setText("Connected");
            setConnected(true);
        }
        catch (PortInUseException e)
        {
        }
        catch (Exception e)
        {
        }
    }

    //Opens the input stream
    public boolean initIOStream()
    {
        boolean successful = false;

        try {
            //
            input = serialPort.getInputStream();
```

```
            successful = true;
            return successful;
        }
        catch (IOException e) {
            return successful;
        }
    }

    //Initializes the event listener for the serial port
    public void initListener()
    {
        try
        {
            serialPort.addEventListener(this);
            serialPort.notifyOnDataAvailable(true);
        }
        catch (TooManyListenersException e)
        {
        }
    }

    public void disconnect()
    {
        //close the serial port
        try
        {
            serialPort.removeEventListener();
            serialPort.close();
            input.close();
            window.btnDisconnect.setEnabled(false);
            window.btnConnect.setEnabled(true);
            window.comPorts.setEnabled(true);
            window.jLabel5.setForeground(new java.awt.Color(255, 0, 0));
            window.jLabel5.setText("Not Connected");
            setConnected(false);
        }
        catch (Exception e)
        {
        }
    }

    final public boolean getConnected()
    {
        return bConnected;
    }

    public void setConnected(boolean bConnected)
    {
        this.bConnected = bConnected;
    }

    //When an event is received this writes the byte to the display string and
updates the
```

```
    //text boxes if necessary
    public void serialEvent(SerialPortEvent evt) {
        if (evt.getEventType() == SerialPortEvent.DATA_AVAILABLE)
        {
            String value;
            try
            {
                byte oneChar = (byte) input.read();

                value = new String(new byte[] {oneChar});
                display += value;

                if(display.charAt(display.length()-1) == '\n'){
                    System.out.print(display);


                    switch(display.charAt(0)){
                    case 'T':
                        window.thermistorText.setText(display.substring(1));
                        break;
                    case 'R':
                        window.fsrText.setText(display.substring(1));
                        break;
                    case 'F':
                        window.flexText.setText(display.substring(1));
                        break;
                    default:break;
                    }
                    display = "";
                }
            }
            catch (Exception e)
            {
            }
        }
    }
}
```

15.1. Java-GUI Code

GUI.java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 * GUI for displaying data
 *
 * @author Gabriel Bautista
 */
public class GUI extends javax.swing.JFrame {

    private static final long serialVersionUID = 1L;

    SerialRead serialReader = null;

    /**
     * Creates new form GUI
     */
    public GUI() {
        initComponents();
        createObjects();
        serialReader.searchForPorts();
    }

    private void createObjects()
    {
        serialReader = new SerialRead(this);
     }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */

    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        comPorts = new javax.swing.JComboBox<String>();
        btnConnect = new javax.swing.JButton();
        btnDisconnect = new javax.swing.JButton();
        thermistorText = new javax.swing.JTextField();
        fsrText = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        flexText = new javax.swing.JTextField();
        jLabel5 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Fracture Model Sensors");
```

```
        jLabel1.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText("Sensor Values");

        btnConnect.setText("Connect");
        btnConnect.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnConnectActionPerformed(evt);
            }
        });

        btnDisconnect.setText("Disconnect");
        btnDisconnect.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnDisconnectActionPerformed(evt);
            }
        });

        thermistorText.setEditable(false);
        fsrText.setEditable(false);
        flexText.setEditable(false);

        jLabel2.setText("Thermistor:");
        jLabel3.setText("Flex Sensor:");
        jLabel4.setText("FSR:");

        jLabel5.setFont(new java.awt.Font("Tahoma", 1, 14));
        jLabel5.setForeground(new java.awt.Color(255, 0, 51));
        jLabel5.setText("Not Connected");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(thermistorText,
javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel2))
                        .addGap(24, 24, 24)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(6, 6, 6)
                                .addComponent(jLabel4))
                            .addComponent(fsrText,
```

```
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(24, 24, 24)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel3)
                            .addComponent(flexText,
javax.swing.GroupLayout.PREFERRED_SIZE, 65,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addGap(41, 41, 41)
                        .addComponent(jLabel5))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(comPorts)
                        .addGap(18, 18, 18)
                        .addComponent(btnConnect)
                        .addGap(14, 14, 14)
                        .addComponent(btnDisconnect)))
                .addContainerGap(32, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jLabel1)
                    .addComponent(jLabel5))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(flexText,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(thermistorText,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
                        .addComponent(fsrText,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(comPorts,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnConnect)
                    .addComponent(btnDisconnect))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    private void btnDisconnectActionPerformed(java.awt.event.ActionEvent evt) {
        serialReader.disconnect();
    }

    private void btnConnectActionPerformed(java.awt.event.ActionEvent evt) {
        serialReader.connect();
        if (serialReader.getConnected() == true)
        {
            if (serialReader.initIOStream() == true)
            {
                serialReader.initListener();
            }
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting
code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the
default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
```

```
java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(GUI.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new GUI().setVisible(true);
            }
        });
    }
    // Variables declaration - do not modify
    public javax.swing.JButton btnConnect;
    public javax.swing.JButton btnDisconnect;
    public javax.swing.JComboBox<String> comPorts;
    public javax.swing.JTextField flexText;
    public javax.swing.JTextField fsrText;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    public javax.swing.JLabel jLabel5;
    javax.swing.JTextField thermistorText;
    // End of variables declaration
}
```