# CHAPTER 1

# INTRODUCTION

The human body contains many parts and out of these, hand is one of the vital parts. Hand parts are very gentle and intricate structure. This gives muscles and joints in the hand part a wide variety of actions and accuracy. Having the ability to perform important essential actions symbolizes multiple degrees of freedom and is capable of performing integrated functions. Bones are highly accountable for rigidity within a segment of a hand, joints provide the convenient freedom of movement and muscles serve to transfer rigid segments on each other. A lot of robotic hands have been designed and developed by many organizations all over the world in modern years. The ubiquitous developments in science and engineering technologies have led to the development of prosthetic hand based on muscle operated sensors. Prosthetics are non-natural devices designed to replace a missing body part, for example a leg or a hand, which may be missing in a trauma, infectious disease, accident or due to birth defect. The working principle of prosthetics is based on EMG signals. EMG is abbreviated as Electro Myo-Graph. EMG signal is nothing but study of electrical signals in muscles. The nervous system in our human body controls the contraction and relaxation activities of the muscles. Hence this signal is a complex signal and is dependent on the anatomical and physiological properties of the muscle movements. The EMG signal gets corrupted with noise as it encounters different tissues in the human body. Muscle tissue conducts minute electrical voltages analogous to the nerve cells.

Approximately 1 in every 2000 people in the India has undergone an amputation. Annually, in India, approximately 1560 people go through an amputation operation, and, there are currently over 1.7 million people in India with an amputated limb. Amputations (because of trauma) have been declining over recent years, as have amputations from cancer. However, amputations due to vascular diseases, such as diabetes, are increasing.

Artificial hand designs and existing prototypes aim to basically satisfy amputee needs. Costs range from 20000 to 200000 Rupees depending on functionality and different features. Regardless, prosthetic hands may be divided in two groups considering price, size, feedback sensing, and strength of force for griping, weight, and movement capabilities. The first group includes the most expensive hands that are similar to a natural human hand size and replicate natural hand movements. Such designs look like a natural hand, they have good sensing and feedback mechanisms and they are strong enough for daily life activities. A representative design of this group is the steeper’s BeBionic hand , which costs 3500000 Rs; it has the ability to grip following 14 different patterns, finger sensing to help the user sense, a rear magnet to improve hand strength and speed of gripping. Its weight is 390 grams . The second group of hands varies in size reaching the size close to the normal human hand; however, designs in this group lave limited strength .

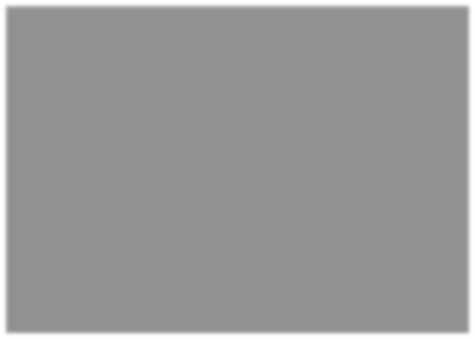


Figure 1.1 BeBionic hand of a female human [1].

* 1. **Motivation**

At the start of this project, complete upper arm prosthetics as a whole were strongly considered. There is essentially no product on the market today which resembles a complete functional prosthetic arm. There are several companies that produced complete lower arms including retired elbow, but the shoulder has long been overlooked. The marketplace essentially has no room for a complete upper arm prosthetic device because the cost would be so high the third be no available customers. Since 2007, the United States military began expressing interest in revolutionizing prosthetic devices to give wounded soldiers replacement limbs. Many soldiers were injured on the battlefield from improvised explosive devices and landmines. Commonly those soldiers would lose limbs directly or require amputation from shrapnel damage. The military had the budget to help wounded soldiers who had given their limbs fighting for their country. It was clear that lower extremity prosthetics were functional and readily available, but upper extremity prosthetics that essentially not advanced since the Civil War. Most people simply made do with what they had when it came to having an arm amputation. Several soldiers however even loss both of their arms and tragedies and it was clear that something needed to be done for them. There are over an estimated 100,000 upper extremity amputees

currently living in the United States alone. Many of those people could benefit from the psychological gains and physical usefulness of a simple powered prosthesis. It is a sad fact that people who are viewed as “different” in our society stand out, but those people simply want to blend in and be treated normally, and be able to lead normal high functioning lives. Amputees are strong and capable people, who make do with what they have, and are able to overcome adversity. There is room for improvement in all aspects of current prosthetic technology relating to mechanical design, electrical signal processing, and overall system performance. There are not a large number of major companies developing competing products because the market is still quite small and limited from a business perspective. Shown in Figure 1 is an early prosthetic hook and socket created during the Civil War. Modern prosthetic hooks remain very similar aesthetically and it is time to move into the 21st century. The main goal for this project was to produce a complete mechanical design of a standalone prosthetic hand. The hand would be considered a basis for a future product, but the design would also serve as a mechanical investigation into discovering what should be possible with a different design approach. Ideally, a functional prototype could be produced using the design developed in this project. That prototype could serve as a platform for future MQPs, and academic research both at WPI and other institutions. Therefore, the design had to be thorough enough to enable the final production of a functioning prototype after complete of the project.

* 1. **Problem Statement**

The human body is amazing. Each part of the body has a specific function, yet when combined with all the other parts working together, the body is a magnificent machine. Just as with other machines, when parts of the machine/body break down, some human parts can be transferred from one person to another; some human parts can be replaced with man-made parts; and when some human parts no longer work, and the body cannot be ‘repaired’, it will stop working altogether. One of the most important parts of the human body is the hand. The hand movements help the body perform such actions as pushing, pulling and/or grasping objects; feeding one’s self, writing, etc.

The hand is structured to handle objects of different sizes, shapes, and weight. It is covered by skin with nerve endings to sense the external world through touch. Use of the arms and hands assists in balancing the person’s body in space, either while being still, as on a tightrope, or while moving, as in running.

Living without a hand or hands becomes very challenging for the amputee. Many attempts have been made towards resolving problems for amputees to help them live a more normal life. History reports that an early surgery, done between 950 – 710 B.C., discovered on a mummified Egyptian corpse, attempted to create a solution for the loss of a big toe by using a piece of wood to replace a big toe [15]. The first

documentation of the use of an artificial hand occurred between 218-201 B.C., for Marcus Sergius [16], a Roman general who lost his right hand during a battle. Sergius used an iron hand to hold his shield and went back to the battle to fight again [17].

Since then, the field of creating / developing artificial hands has continued to grow. The first study of EMG was done by Francesco Redi in 1666 [18], followed by the first record of EMG signals by Jules Marey in 1890 [19]. Marey was also the first person to use the term “electromyography”. Great progress in the field of artificial hands occurred when the EMG signal was discovered to be able to control the prosthetic hand. Researchers continued to try to design a prosthetic hand which could satisfy the needs of the amputees. The best designs found to be useful to amputees are those that can mimic the actual movements of the human hand, have good sensory feedback, and create enough force to handle objects with smooth movements and have very good EMG signal detection and processing. A prosthetic hand that includes all these requirements in one prosthetic device would make it cost prohibitive for most amputees. This research has focused on the creation of an affordable prosthetic hand that functions ‘well enough’ to make it available and useful for amputees.

* 1. **OBJECTIVES**

This thesis is motivated by the challenge to develop a prosthetic hand that is not heavy, it is very cost-effective (to be affordable by almost every amputee), with effective and efficient feedback sensing mechanisms and capable of reproducing most of a human hand’s movement and gripping patterns. The prosthetic hand should be strong enough to hold different object shapes and weight, and its size should be comparable to a human hand.

As such, the thesis focuses on designing a cost-effective five-finger prosthetic hand. Aluminum is chosen as the building material. Each finger is actuated separately by controlling the angle of a corresponding servo motor. The five fingers combined, account for 14 DOF. Each finger except the thumb, has three links, one base and three joints. The thumb has two links, one base and two joints. The hand design is shown in Figure 1.2. Each finger uses a force sensor to sense objects (in touch with the finger) and returns a feedback signal. The prosthetic hand is controlled by surface EMG signals obtained from surface electrodes that are connected to the arm muscle(s). Obtained signals from the electrodes go through a controller to generate specific finger/hand motions, and also provide the necessary feedback. For the proposed design, five muscles are used to reproduce hand motions.

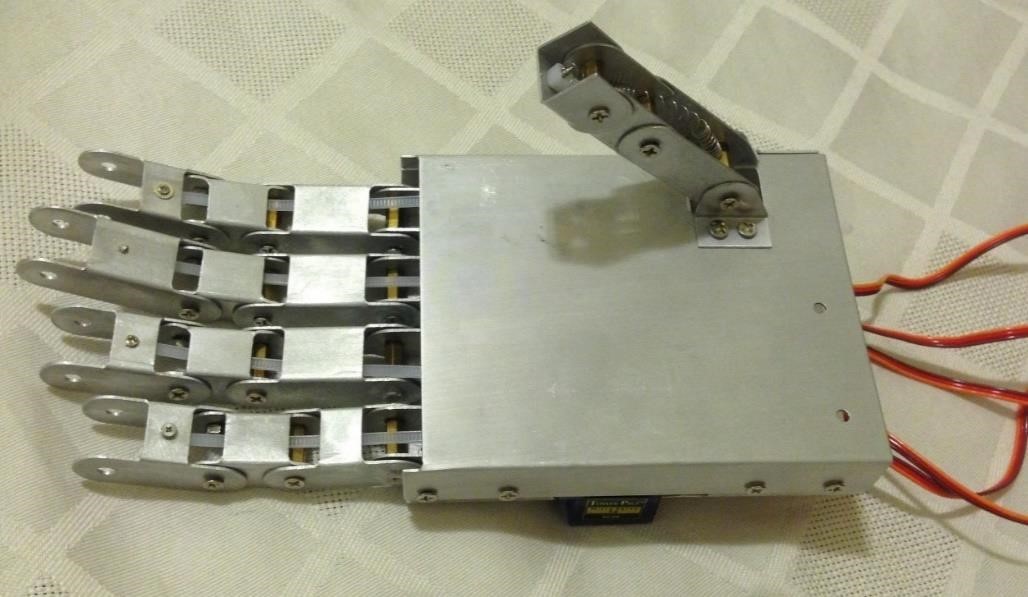
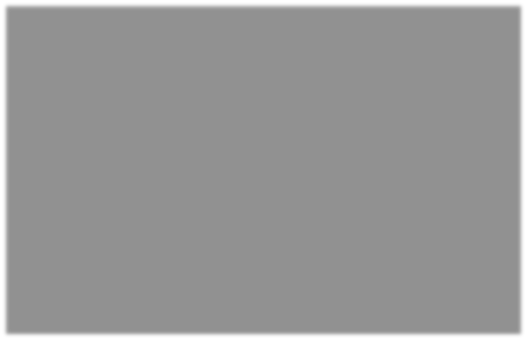


Figure 1.2 Five-finger prosthetic hand structure.

The most challenging task of designing the prosthetic hand is EMG signal detection and processing. It starts with connecting (three) multi-useable electrodes, positive, negative and ground, to the skin/surface covering the muscle, in order to detect muscle movement. Five muscles are used to detect the EMG (one per finger) using a total of 11 electrodes, one serving as the common ground. The ground electrode is connected close to any bone; the positive/negative electrodes of each muscle are connected to the top and bottom end or to the middle of the muscle.

The EMG signal passing through the electrode wires may be affected by electromagnetic fields close to the wires [3], and this may generate undesirable noise in the EMG signal. Twisting the electrode wires (see Figure 3.1b) prevents interference and reduces noise as electromagnetic fields in opposite direction cancel each other. After EMG signal detection, EMG signal processing follows, which includes differential, preamplifier, filtering and amplifier stages. The five fingers are controlled by an Arduino UNO depending on the received EMG signal(s). A force sensor is attached to each fingertip to sense any object in touch with a finger. These force sensors turn on and off vibration motors connected to the amputee’s arm, informing the amputee if there is anything in touch with the fingertip.

**2. LITERATURE SURVEY**

In A Studied Paper the authors present development of a low cost three finger robotic hand with six degrees of freedom. The hand has three actuators and it can grasp different shapes such as oval, cuboid, circular and cylindrical as shown in Figure 2.1. The grasping command depends on the EMG signals coming from three surface electrodes. By using a fuzzy classifier, the grasp planner can recognize a shoulder’s abduction or adduction movement based on the root mean square value of the EMG signal. A proportional controller associated with position touch sensors has been used to provide feedback signals. The raw EMG signals from electrodes go to the EMG preprocessing unit. This unit consists of a differential amplifier, which subtracts the two signals that come from the muscle; the first signal comes from the electrode that is connected to the middle of the muscle and the second signal comes from the electrode connected to the end of muscle. This step is used to ‘stabilize’ the signal.

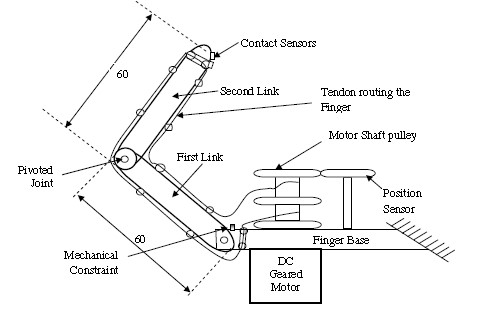
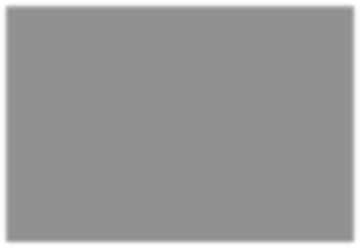


Figure 2.1 Developed three fingered robotic hand

Then, the signal passes into a fuzzy classifier to recognize the abduction or adduction movement of the shoulder to open or close the hand. The skeleton of this hand is made from nylon and each finger has two equal-length links. Each finger is actuated by a DC motor for flexing and extension movements. There is a pulley with tendons wrapped around it. In order to transmit the force from the DC motor to the pulley, a nylon thread is used to extend and flex the links of each finger. There are limitations to this design. The first limitation is that there are just two links in each finger so the fingers cannot reach very many points in the working area. Also, the hand cannot catch other shapes like a triangle. In addition, during the filtering stage the designer used a notch filter to remove noise, which removes frequencies that are between 50-60 HZ. This may make the EMG signal lose much of the signal powered data.

In another Paper research has been done to evaluate recognition of various patterns, and to study real-time implantation. A surface EMG signal is used to allow for the prosthetic hand the ability to evaluate six different hand motions. There are three electrodes connected to each muscle, two electrodes which measure the deferential voltage in two different points in the muscle and the third electrode that serves as ground. Signals from the electrodes pass through a protection circuit (for the subject’s safety) followed by the differentiation stage, which makes the signal more stable by using a differential amplifier (in this case, an instrumentational amplifier). Any unwanted signal that has high and low frequencies is removed using a RC bandpass filter. The signal, then, amplifies to be readable by the controller. Much focus has been given to getting data from training the six hand motions to imitate such motions. Stored data are processed in two to three stages depending on the computational complexity of the data, feature reduction, and classification.

Fingers have been designed using CAD software. Components were built by using rapid prototyping techniques with polypropylene. Each finger has two revolute hinge joints and three degrees of freedom. The reason behind having this simple structure is that abduction/adduction movements of the metacarpophalangeal joint have been ignored. There are holes made on phalanges to route the drive string through all the phalanges. Using the DC motor, nut, and bolt mechanisms, the string can be pulled/ pushed so the finger will be closed or opened, depending on the motor’s direction of rotation, which is decided by the order coming from the EMG’s signals.

Research in another paper focused on development of high performance, low degrees of freedom EMG prosthetic hand, which uses two motors to realize various motions. One of these motors is connected to the thumb to generate rotary motion from 0 to 90 degrees. The second motor is connected to the other four fingers. To increase contact area and static friction, soft material has been used to cover the fingers. Also, nails added to the end of each finger assist in helping fingers catch small objects. This design has a protection device to optimize its mechanical structure, keeping the design safe from the impact caused by a sudden external shock or an overload current. In addition, the design has a spring parallel to the plane of the palm. During a sudden shock, the springs bend laterally alongside it. The limitations of this hand are that there is no feedback, and all four fingers move together; it is also not strong enough. The hand configuration is shown in Figure 2.2.

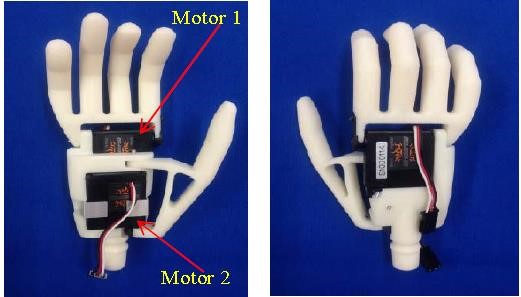


Figure 2.2 EMG prosthetic hand with two motors

Research in another paper focuses on a biologically inspired parallel actuation system that attempts to control the actuators of the prosthetic hand using the Flexor Digitorum Profundus (FDP) and Flexor Digitorum Superficialis (FDS) muscles. By looking at Figures 2.3 and 2.4, one may observe that Region 1 represents the more frequently dexterous tasks, while Region 2 represents the less frequent movements determining movements that need more force.

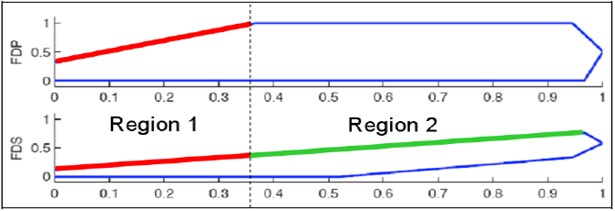


Figure 2.3 Divided strength space of FDS and FDP muscles; x-axis is normalized hand position, and y- axis is normalized force

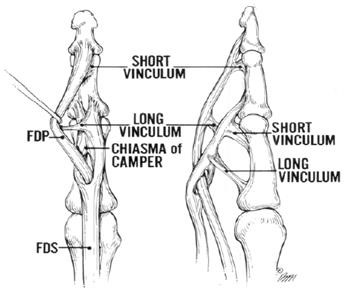
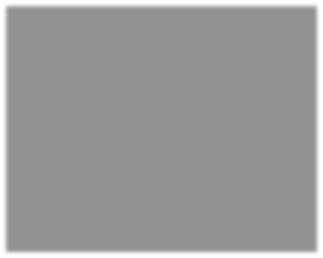


Figure 2.4 Graph of FDS and FDP muscles [7].

The design mimics a male’s human hand size with the same degrees of freedom. It couples both the DIP with PIP joints of the fingers, and the IP with MCP joints of the thumb. This is done by using a single actuator for both the PIP and DIP joints. There are two DC motors, the first one connected to the metacarpal phalange of the finger to actuate the horizontal degree of freedom of the MCP joint, while the second one is connected to the base of the thumb to actuate the CMC joint to obtain an approximation of the abduction/adduction motion. The limitation of this design is that there are no feedback sensors.

The goal of another paper is to develop a cost-effective robotic hand controlled by EMG signals collected from surface skin electrodes. There are two ways to obtain signals from muscles. The first way is invasive (surgery or implantation required); the second way is non-invasive (external sensing). Each method has advantages and disadvantages. The invasive way allows for very readable signals but, sometimes, there may be some side effects, like infection or body rejection to electrodes. The hand’s design consists of motorized digits, which are either extended or contracted depending on the EMG signal that comes from the muscle. Each digit is driven by a DC motor and gearbox. There is sensing feedback to determine the amount of torque that should be applied to the object by the hand to avoid hand fatigue.

EMG signals from the electrodes pass through three processing steps. In the first step, an instrumentational amplifier is used to amplify the differential between signals coming from the two electrodes. In the second step, the signal from the first step passes through a 25 - 400 HZ band pass filter to remove unwanted low and high frequencies. Finally, to make the signal slower and suitable for a microcontroller to read, the raw EMG signals are converted to the mean average voltage signal MAV. Depending on the EMG signals’ values, the microcontroller receives varying voltages from the MAV to control the motorized hand.

This hand has motorized digits, which can be spread out, and can contract depending on the EMG signals. There is a DC motor in each digit to actuate it. Also, there is a current sensor to measure the maximum current to prevent the motor from having any damage. Limitations of this design are: there are no feedback sensors; the used filter does not account for the signals with 50 HZ (which is the electricity waves).

Research in another paper focuses on designing a cable-driven anthropomorphic robotic hand that has 20 degrees of freedom. The mean body of the hand is made using a 3D printer, where the hand has five fingers. Each finger has three joints, namely, the metacarpophalangeal (MCP), proximal interphalangeal (PIP), and distal interphalangeal (DIP) joints. There is one degree of freedom for each DIP and PIP joints. Two degrees of freedom result from the MCP joint. The first degree of freedom is to achieve the flexion extension motion; the second degree is used to achieve the abduction-adduction finger motion. The three joints in the thumb are: Carpometacarpal (CMC), Metacarpophalangeal

(MCP), and Interphalangeal (IP) joints. The thumb’s IP and MCP joints are used to process one degree of freedom in the flexion-extension direction, respectively, while the CMC joint has two degrees of freedom, as shown in Figure 2.5.

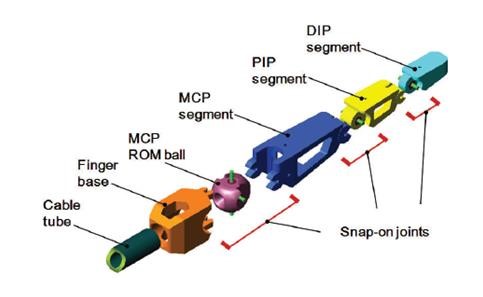
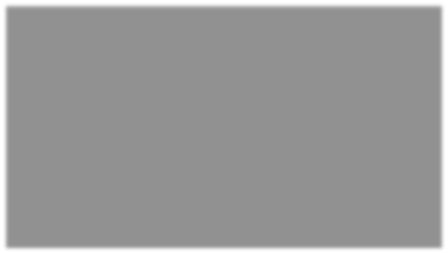


Figure 2.5 Components of each finger unit [9].

An objective of this design is on reducing cost, by depending on 3D printing to design the hand’s parts. The LEGO-Style snap - on joint was used in each joint connection between two finger segments. The mechanism of snap - on joint on one side consists of 3D printed parts with a C- shape clip. There is a steel

shaft in the center of the other side of the joint. The snap - on joint is formed when the steel shaft is closed by friction engagements, and by snapping on of the clip. For each finger there are four pairs of antagonistic tendons to control the finger’s DOF. Also, in the sensing field, there are three layers in each 16 independent skin pads that exist in the hand. These three layers include: Velcro that is embedded in artificial skin, a tactile sensing element, and a 3D printed frame. The artificial skin is made from a silicon rubber which is cleanable, and is resistant to water and oil, and also an anti-smudge to any adhesive. The second layer has the ability to identify the magnitude and the amount of pressure points on the hand because it has sent an array from a five fingers grip system, to determine the position of the pressure. There is a flexible paper-thin, and this paper will bind with the Velcro’s surface and the sensor layer will carefully have wrapped onto the 3D printed frame.

Research in [10] discusses the design of a low-cost prosthetic hand controlled by EMG. This hand has been designed with three fingers and one degree of freedom to grasp objects. These three fingers are controlled by one DC motor. The design is shown in Figure 2.6. The microcontroller depends on incoming data from EMG to decide about the PWM required for the DC motor to complete required tasks. The microcontroller monitors the amount of current required by the DC motor to prevent it from any damage. To get the EMG signals, electrodes made from silver/silver chloride were used. The EMG signal that comes from the electrodes is between 20 microvolts to 2 millivolts, and it passes through three stages to be readable by the microcontroller.

The first stage in EMG processing is the pre-amplifier stage that uses the INA129 instrumentational amplifier. In this stage, the difference between the two signals coming from the electrodes (connected to the middle and end of the muscle) is found. The amplifier gain in this step is about 10 - it is not that high because the signal has noise that should not be amplified. In the second stage, the signal passes through the RC high pass filter followed by a TL082 low pass filter operational amplifier. The gain for this step is 200, and frequencies that pass are between 20 HZ and 400HZ. Finally, the signal converts to a digital signal by using an A/D converter, so the controller can easily deal with it. The limitation with this hand is that there is no feedback, and the design has low degrees of freedom.

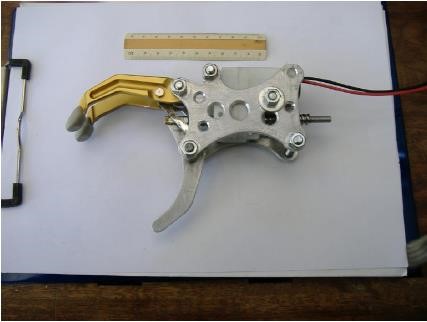


Figure 2.6 Mechanical structure of the hand

In another paper a simplified 3D printed EMG prosthetic hand was developed. The design is actuated by two motors and controlled by an EMG signal. This prosthetic hand has been covered by a highly stretchable cosmetic glove that has the human color, and its fingers look like the human fingers. After detecting the signal from the electrodes, the signal is processed by using a 32-bit microprocessor for pattern recognition and controlling of the motors. The design has two motors. The first motor is connected to the thumb to create a rotary motion ranging from 0 to 90 degrees. The second motor is connected to the other four fingers to create the opening and closing motions. These two motors are embedded inside the palm. Also, the wearable electrodes have been designed to detect the EMG signal. Limitations with this design are that there is no feedback signal that helps the amputee to sense his/her environment, and the design has low degrees of freedom.

In another paper a low degree of freedom prosthetic hand is designed for child amputees. Two small servo motors are used to keep the design small and not heavy. A wire-driven mechanism is used to actuate the fingers. During the age of 0 to 8 years, children learn most of their hand’s functions and movements. Therefore, the amputee children learn how to use their muscles (at least) for several movements. The limitations in this design are the low degrees of freedom, and lack of feedback signal in the design.

In another a prosthetic hand was developed having 10 joints including 4 active joints. This hand was built according to the intuition of the phantom-hand motion. There are four motors that are embedded into the palm and wrist to actuate the fingers. This hand has three fingers with 10 degrees of freedom as shown in Figure 2.7. The thumb has two active joints.

This design has low degrees of freedom and there is no feedback signal.



Figure 2.7 Three fingers prosthetic hand [2].

In another paper a low-cost electrically powered prosthetic hand was developed. The supplement of the amputee-prosthetic EMG control of this hand is the novel haptic user interface (HUI). The skeleton of this hand is made by using 3D printed material to be light, as shown in Figure 2.8.



Figure 2.8 Touch hand [13].

The designer used HUI to create a communication between the prosthetic hand and the amputee. The HUI uses a haptic medium (vibrotactile motors) to display the information to the amputee, instead of

using the vision display. This design uses flex sensors as position sensors in each finger to give feedback to the microcontroller to control the motors. The microcontroller checks the flex sensor for the finger’s position. When the sensor value shows that the finger is fully opened or fully closed, the microcontroller turns off the motor. The limitation of this design is that there is no feedback signal to inform the amputee if the finger touches an object or not, and whether the touch was soft or hard.

In another paper a three fingers robotic hand with four degrees of freedom is designed. The hand has palm, thumb, index finger, and middle finger. These fingers are tendon-driven. Each finger (except the thumb) consists of three parts; distal phalange (DP), a proximal phalange (PP), the middle phalange (MP), and metacarpal bone (MB). The thumb does not have a middle phalange MP. The process of actuating the fingers begins from the proximal phalange. The proximal phalange will be actuated by pulling the wire using a DC motor. Then, the distal-middle phalange will be actuated synchronously by a link mechanism. The design steps start with designing a single phalange called a distal-middle phalange (DMP). Here the DMP phalange connects to the DP and the MP. The second step will be done by using a link mechanism between the MCP joint (which is the joint that connects the MP and the PP) and the PIP joint (which is the joint that connects the PP with the MP). In the final step a spring will be used as an extension muscle. The design is shown in Figure 2.9. The limitations in this design are; there is no feedback signal, and there are few fingers so this hand cannot imitate many of the human hand movements.

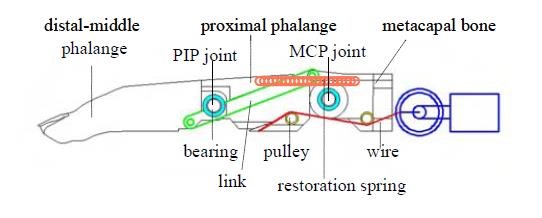


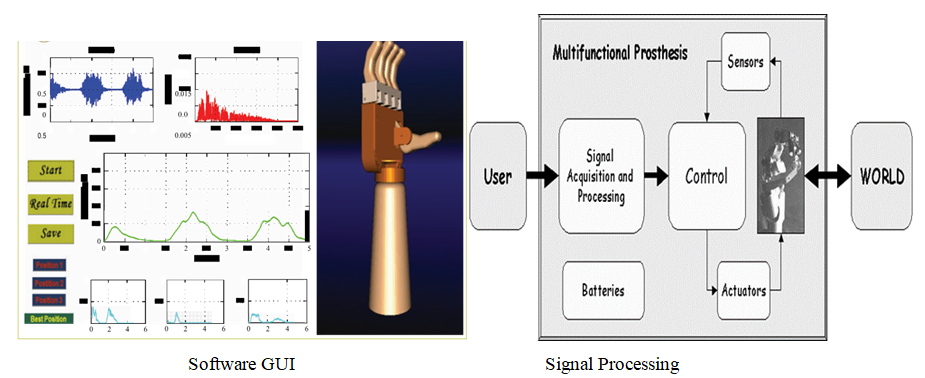
Figure 2.9 The structure of the finger

**CHAPTER 3 METHODOLOGY**

**3.1 INTRODUCTION**

The proposed system is based on the use of EMG signals to support the mechanism of the EMG powered prosthetics. For the aim of the project, the proposed system was established through two stages as follows.

A software that is responsible for extracting the EMG signal from the patient and analyze it to figure out the most suitable position on the stump of amputee to collect the signals, and a view of the planned GUI interface is illustrated in the below Figure. Also, this software can help to train the patient how to control the three-dimensional hand in the program created using CAD. Each patient has a different signal as variable resistor to change the gain suitable for every patient individual. The electrodes were placed on the flexor carpi radialis, the signals from the sensor after muscle contraction were sent to the controller. To prevent unexpected movements into the body of control code threshold was added. After obtaining of the desired value, the control system sends the signals to the motors which then actuate the closing and opening of the prosthetic hand. An EMG signal is an Electric Potential generated by Muscle Contraction. It may be measured on the skin surface or by embedding sensors into deeper layers of the muscle. Medical reference suggests that different compartments of the forearm muscle related to hand and finger movement and EMG signal can still be measured from the forearm muscle even after the hand is amputated. Therefore, theoretically, it is possible to use the EMG signal to control hand and ﬁnger movements. Then the EMG signals need to be measured through the electrodes placed systematically. Then the EMG signal needs to be filtered and pre-processing needs to be done Then the processed signal needs to be input to a microcontroller. Then the signals are used to control micro switches and the Servo Motors in the Artificial Arm Then the assembly is placed in a Container and placed on the arms of the User Then Successive Iterations of Testing Is done and the Accuracy of the Grips Is Fine Tuned.



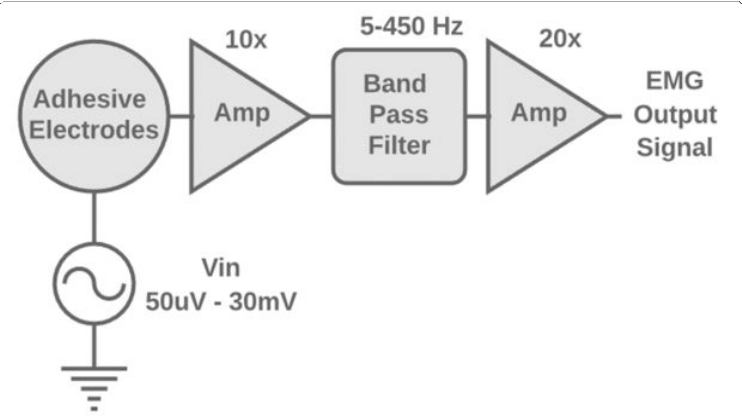


Fig 3.1 Basic over view of EMG sensor circuit blocks

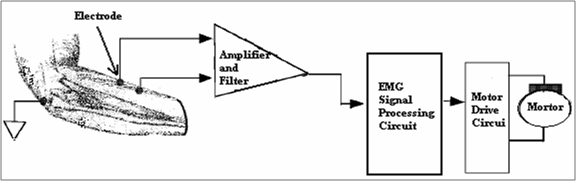


Fig 3.2 Basic over view of Methodology

* 1. **Block Diagram Representation of proposed method**

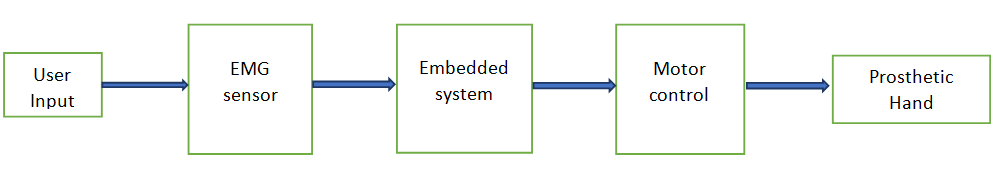


Fig 3.3 A Block Diagram of The Project

USER INPUT:

* Obtained from the Contraction and Relaxation of the Flexi Carpi Radialis Muscle.

EMG SENSOR:

* It is the Sensor used to collect raw data from the Muscles through Electrodes placed on the arm.

EMBEDDED SYSTEM:

* It is the Microcontroller used to manipulate the input from EMG and the output to the Artificial Arm.

MOTOR CONTROL:

* It is controlled with the help of Micro Switches and through the output from the Micro Controller.

PROSTHETIC ARM:

* It is the Artificial Arm that is 3-D printed and it Mimics the actions of the real hand using the output from the Micro Controllers with the help of Servo Motors.

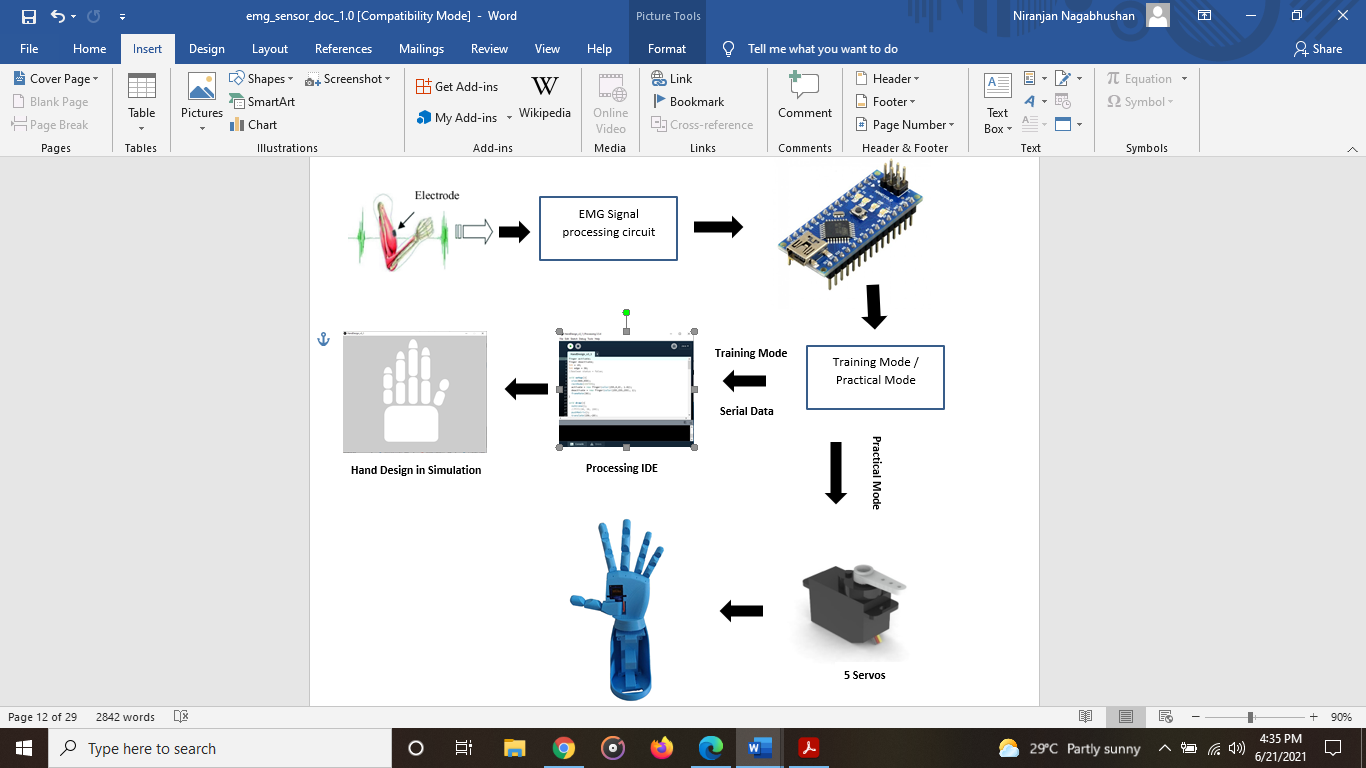


Fig 3.4 Connection Diagram:

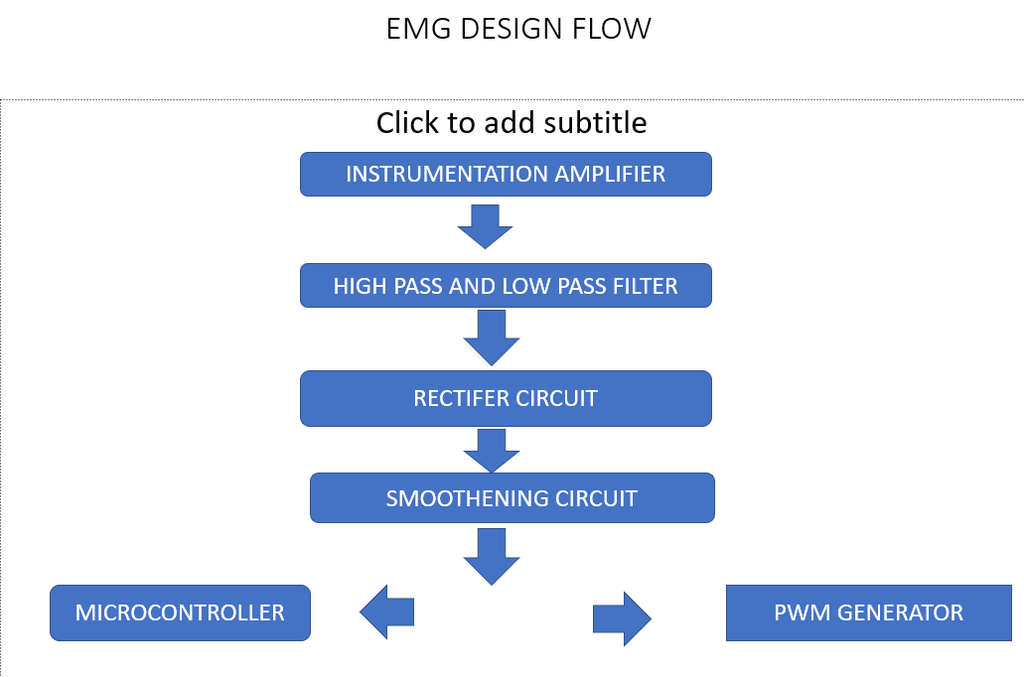
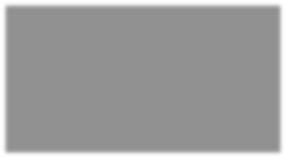
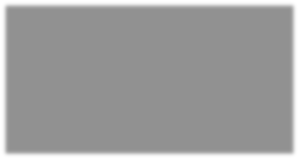


Fig 3.5 Proposed Flow Chart of The Hardware Circuit

**3.2 Hardware and software requirements**



(a)(b)

Figure 6.3 (a) Is invasive electrode and (b) is non-invasive electrode

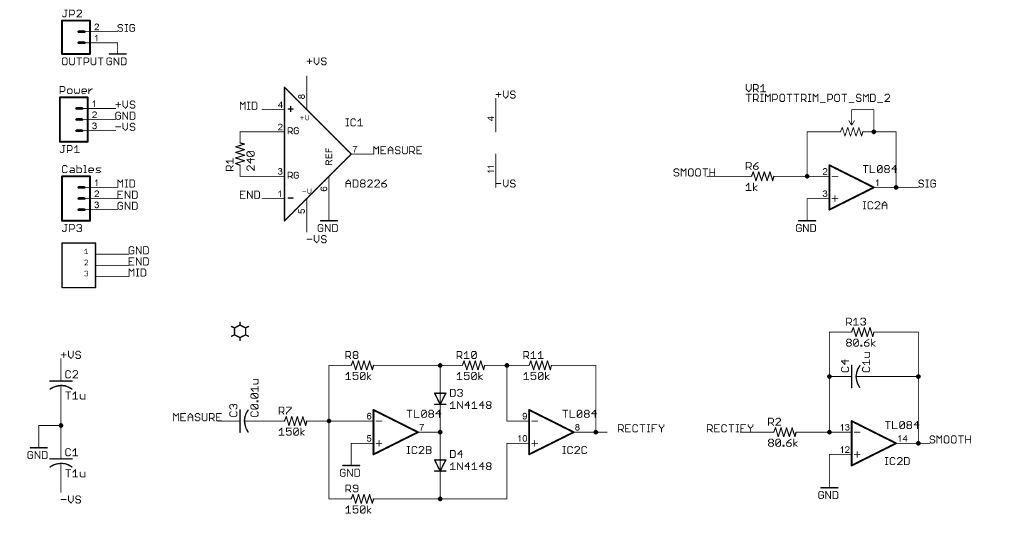


Fig3.6 Circuit Diagram of All the Hardware

An EMG, or electromyogram, is a measurement of the electricity produced by the movement in muscle tissue. Three electrodes will be used as sensors to provide input voltage to the circuit, and the output voltage will provide a reading of muscle activity. The stages of the circuit are as follows: an instrumentation amplifier, a band pass filter, and a non-inverting amplifier. The instrumentation amplifier provides high input impedance to match the high output impedance of skin. The band pass filter removes frequency content out of the bandwidth of the EMG. Finally, the non-inverting amplifier provides enough gain to make the small EMG signal large enough to be usable.

Here we are building an EMG amplifier that allows us to translate the tiny electrical signals produced by motor-neurons into mechanical action using servo to control the prosthetic arm.

EMG signals are measured differentially, meaning the signal we amplify and analyze is actually the difference in electrical potential between two points on the muscle.

Our main goal with this EMG amplifier is to amplify the signal we're interested in (muscle activity) without increasing the noise (often from external electrical interference). A differential amplifier operates on the assumption that any noise signal interfering with our EMG recording

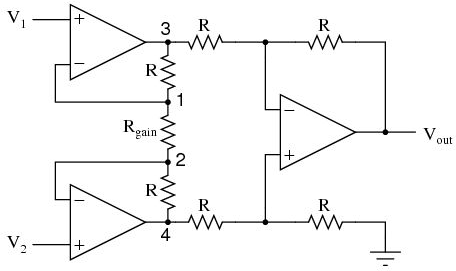
will uniformly affect the local region of muscle we're recording from. In other words, a large source of interference across the room will introduce the same amount of electrical noise in our recordings whether we record from a muscle in the middle of your forearm or a few centimeters away from that same position.

Making use of this assumption, we're able to achieve our goal of maximizing the [signal to noise ratio](https://en.wikipedia.org/wiki/Signal-to-noise_ratio) by amplifying the difference between two recording sites close together on a muscle. The components of the signal that are common between the recording sites (the electrical noise) will be removed when we take the difference (i.e. subtraction) between the two recordings. The ability of an amplifier to reject the common signal is referred to as the [common-mode rejection ratio](https://en.wikipedia.org/wiki/Common-mode_rejection_ratio) (CMRR). An ideal differential amplifier would have an infinite CMRR, rejecting all the noise that is the same at either input, but when it comes to practical the CMRR is defined so it removes some part of noise.

# Step 1: Instrumentation Amplifier

The instrumentation amplifier is a circuit with high input impedance which amplifies the difference between two input signals. Because the skin has high output impedance, its voltage signal can only be measured by a circuit that has high input impedance. This phenomenon can be easily understood by thinking of the skin impedance and circuit impedance as resistors in a voltage divider. If both resistors are of equal value, only half of the input voltage will be measured across the circuit impedance. As the circuit impedance is increased above the skin impedance, more voltage will be applied across the circuit. We want to maximize the voltage going into the circuit.

Furthermore, the EMG is the difference of the voltage signals at the ends of a muscle, so a differential amplifier is required for the first stage.



Differential amplifier

# Specifications of a good instrumentation amplifier

**Accurate and stable gain:**

As the device amplifies signals of the very low level, thus its basic need is its gain must be finite and accurate. Usually, gain lies in the range of 1 to 100.

**Easy gain adjustment:**

When we are talking about the gain. Then it is important that it varies properly inside the specified limit. Usually, the gain is adjusted using a potentiometer or by making use of switches like JFET and MOSFET.

**High CMMR:**

An infinite CMMR is the most preferred range in case of the instrumentation amplifier. As the output of transducer has large common mode noise signals during its long-distance transmission.

An instrumentation amplifier must completely eliminate the common mode noise components in order to amplify the difference of input only.

**High input impedance:**

It is preferred to have an almost infinite value of input impedance in order to avoid the loading effect at the input.

**Low output impedance:**

The low value of impedance at the output must be exhibited by the instrumentation amplifier. In ideal cases, it is assumed to be approximately 0 to avoid loading.

**Low time and temperature drift:**

To have the desired output, it is always recommended that various characteristics and elements of the device must not change with variation in time or temperature.

**Low power consumption:**

For any device, it is always recommended that it must be power efficient. So, an instrumentation amplifier must also consume less power.

**Differential input:**

To have the desired amplification, the device must amplify only the difference of the input signal.

**High slew rate:**

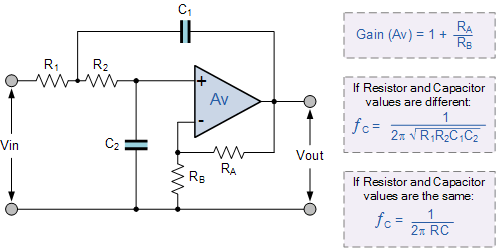
Slew rate provides us with the idea about the change in output voltage with any change in the applied input. So, for an instrumentation amplifier, slew rate must be high.

# Reason to Instrumentation Amplifier

* The gain of a three op-amp instrumentation [amplifier circuit](https://www.elprocus.com/stereo-amplifier-circuit-using-tda2822/) can be easily varied by adjusting the value of only one resistor Rgain.
* The gain of the amplifier depends only on the external resistors used.
* The input impedance is very high due to the emitter follower configurations of amplifiers 1 and 2
* The output impedance of the instrumentation amplifier is very low due to the difference amplifier3.
* The CMRR of the op-amp 3 is very high and almost all of the common mode signal will be rejected.

# Step 2: Band Pass Filter (Low pass and then high pass)

The majority of the EMG signal is between 5-450Hz, so we chose our cutoff frequencies for this range. Our band pass filter was a Salley-Key low pass filter followed by a Sallen-Key high pass filter.



# Step 3: Non-inverting Amplifier

The final stage is meant to increase the output signal to be read. Depending on how we intend to use our EMG signal, what muscle groups we intend to measure from, and the quality of our electrodes, we will require a different output voltage, and therefore, a different gain. Using this amplifier we will be increasing the amplitude of the EMG signal

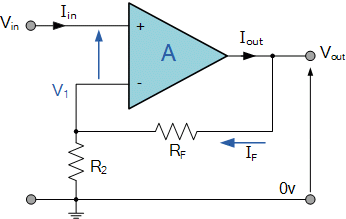


Fig 3.7 Non Inverting Amplifier

**Arduino Nano:**

The **Arduino Nano** is a small, complete, and breadboard-friendly board based on the ATmega328P released in 2008. It offers the same connectivity and specs of the Arduino Uno board in a smaller form factor.

The Arduino Nano is equipped with 30 male I/O headers, in a dip-30 like configuration, which can be programmed using the Arduino Software integrated development environment (IDE), which is common to all Arduino boards and running both online and offline. The board can be powered through a type-b micro-USB cable, or through a 9V battery.

In 2019, Arduino released the **Arduino Nano Every**, a pin-equivalent evolution of the Nano. It features a more powerful ATmega4809 processor, and twice the RAM.

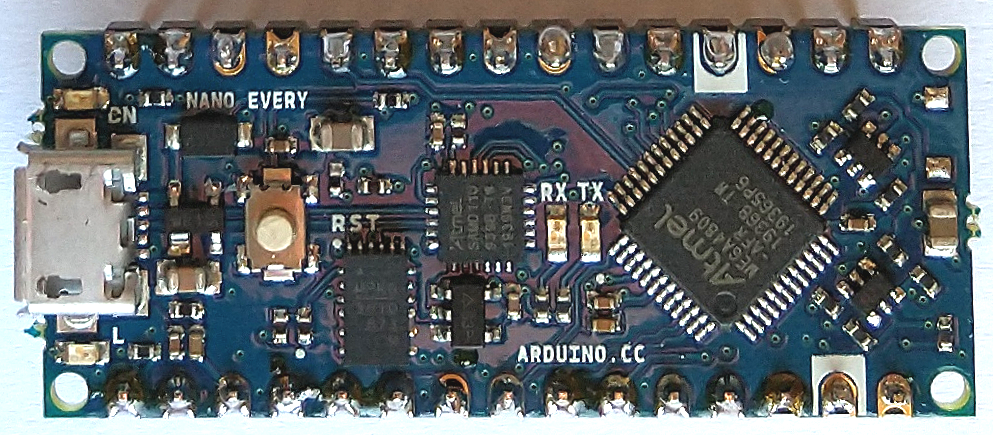


Fig 3.8 An Arduino Nano

* Microcontroller: Microchip ATmega328P
* Operating Voltage: 5 Volts
* Input Voltage: 6 to 20 Volts
* Digital I/O Pins: 14 (plus 6 can PWM output pins)
* Analog Input Pins: 8
* DC Current per I/O Pin: 40 mA
* DC Current for 3.3V Pin: 50 mA
* Flash Memory: 32 KB of which 0.5 KB used by bootloader
* SRAM: 2 KB
* EEPROM: 1 KB
* Clock Speed: 16 MHz
* Length: 45 mm
* Width: 18 mm
* Weight: 7 g

## **Communication**

The Arduino Nano has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provide UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An FTDI FT232RL on the board channels this serial communication over USB and the FTDI drivers (included with the Arduino software) provide a virtual com port to software on the computer. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the FTDI chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A Software Serial library allows for serial communication on any of the Nano's digital pins.

The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus.

### Automatic (software) reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Nano is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the FT232RL is connected to the reset line of the ATmega328 via a 100 nano farad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.

This setup has other implications. When the Uno is connected to a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.

**Motor (Servo Motor) SG 90**

### **Wire Configuration**

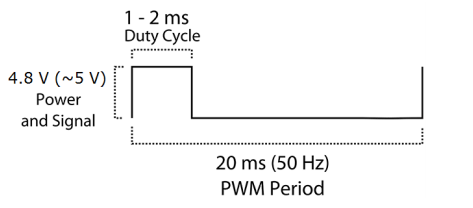
|  |  |  |
| --- | --- | --- |
| **Wire Number** | **Wire Colour** | **Description** |
| 1 | Brown | Ground wire connected to the ground of system |
| 2 | Red | Powers the motor typically +5V is used |
| 3 | Orange | PWM signal is given in through this wire to drive the motor |

### **TowerPro SG-90 Features**

* Operating Voltage is +5V typically
* Torque: 2.5kg/cm
* Operating speed is 0.1s/60°
* Gear Type: Plastic
* Rotation : 0°-180°
* Weight of motor : 9gm
* Package includes gear horns and screws

### **How to use a Servo Motor**

After selecting the right Servo motor for the project, comes the question how to use it. As we know there are three wires coming out of this motor. The description of the same is given on top of this page. To make this motor rotate, we have to power the motor with +5V using the Red and Brown wire and send PWM signals to the Orange colour wire. Hence we need something that could generate PWM signals to make this motor work, this something could be anything like a 555 Timer or other Microcontroller platforms like Arduino, PIC, ARM or even a microprocessor like Raspberry Pie. Now, how to control the direction of the motor? To understand that let us a look at the picture given in the datasheet.



From the picture we can understand that the PWM signal produced should have a frequency of 50Hz that is the PWM period should be 20ms. Out of which the On-Time can vary from 1ms to 2ms. So when the on-time is 1ms the motor will be in 0° and when 1.5ms the motor will be 90°, similarly when it is 2ms it will be 180°. So, by varying the on-time from 1ms to 2ms the motor can be controlled from 0° to 180°

### **Applications**

* Used as actuators in many robots like Biped Robot, Hexapod, robotic arm etc..
* Commonly used for steering system in RC toys
* Robots where position control is required without feedback
* Less weight hence used in multi DOF robots like humanoid robots

### **SG90 Servo Motor Dimensions**

**Softwares:**

**Arduino IDE**

The **Arduino Integrated Development Environment (IDE)** is a cross-platform application (for Windows, macOS, Linux) that is written in functions from [C](https://en.wikipedia.org/wiki/C_(programming_language)) and C++.It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards.

Arduino IDE is a derivative of the Processing IDE, however as of version 2.0, the Processing IDE will be replaced with the Visual Studio Code-based Eclipse Theia IDE framework.

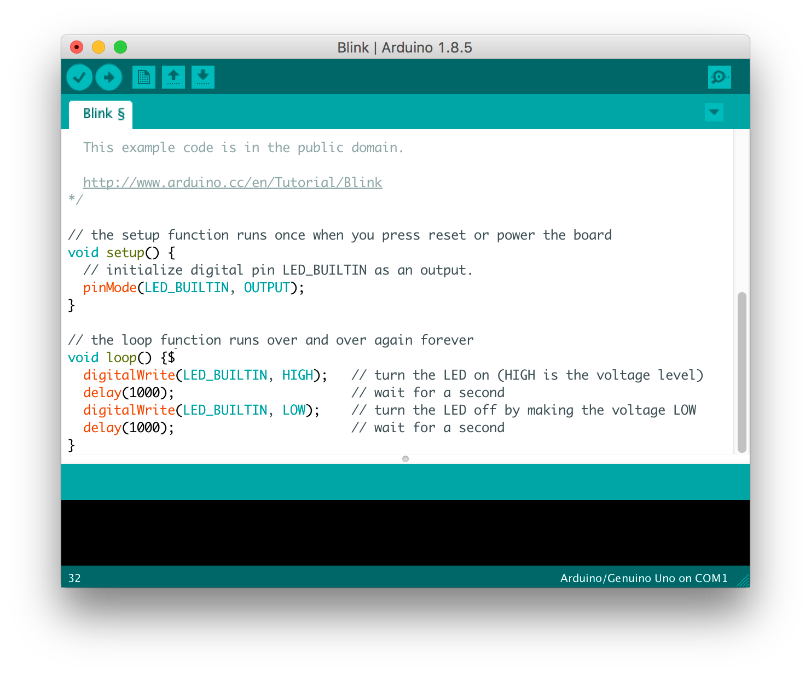


Fig 3.9 IDE Snapshot

**Processing IDE:**

**Processing** is a free graphical library and integrated development environment (IDE) built for the electronic arts, new media art, and visual design communities with the purpose of teaching non-programmers the fundamentals of computer programming in a visual context.

Processing uses the Java language, with additional simplifications such as additional classes and aliased mathematical functions and operations. It also provides a graphical user interface for simplifying the compilation and execution stage.

The Processing language and IDE have been the precursor to other projects including Arduino, Wiring and p5.js.

Processing includes a *sketchbook*, a minimal alternative to an integrated development environment (IDE) for organizing projects.

Every Processing sketch is actually a subclass of the PApplet Java class (formerly a subclass of Java's built-in Applet) which implements most of the Processing language's features.

When programming in Processing, all additional classes defined will be treated as inner classes when the code is translated into pure Java before compiling. This means that the use of static variables and methods in classes is prohibited unless Processing is explicitly told to code in pure Java mode.

Processing also allows for users to create their own classes within the PApplet sketch. This allows for complex data types that can include any number of arguments and avoids the limitations of solely using standard data types such as: int (integer), char (character), float (real number), and color (RGB, RGBA, hex).

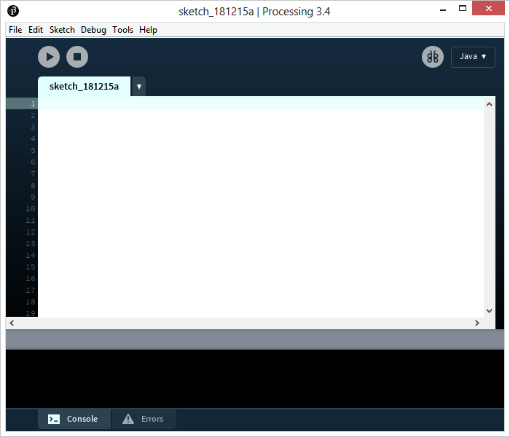
****

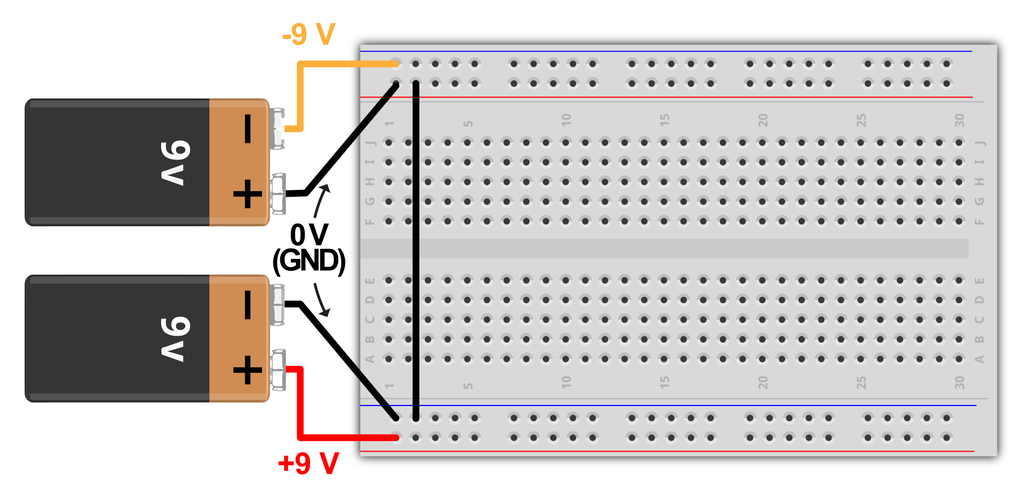
Fig 3.10 Processing IDE Snapshot

**CHAPTER 4 SYSTEM IMPLEMENATION**

# POWER SUPPLY

The electrical activity generated by muscles can either be positive or negative with respect to zero volts. The amplifier must therefore have both a negative and positive supply in order to accommodate and amplify these bipolar muscle signals. Without a negative supply, the amplifier would be unable to produce any outputs less than zero volts.

Two 9 volt batteries are both a convenient and safe way to power our EMG amplifier. Because we'll be attaching ourselves to the circuit, we don't want our circuit connected in any way to AC power from the wall. The diagram below illustrates how to connect the batteries in a ±9 volt arrangement.



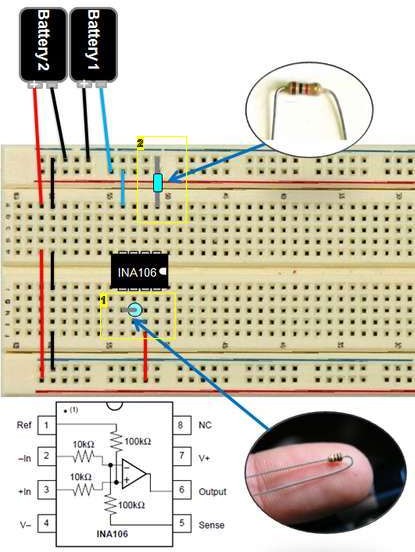
# SIGNAL ACQUISITION

Next, we will work on the signal acquisition phase of your EMG circuit which we will use to measure our body’s nervous system’s electrical impulses used to activate muscle fibers.

EMG signals are measured differentially, meaning the signal we amplify and analyze is actually the difference in

electrical potential between two points on the muscle.

A differential amplifier operates on the assumption that any noise signal interfering with our EMG recording will uniformly affect the local region of muscle we're recording from. In other words, a large source of interference across the room will introduce the same amount of electrical noise in our recordings whether we record from a muscle in the middle of your forearm or a few centimeters away from that same position.



# 3. SIGNAL CONDITIONING – Amplification

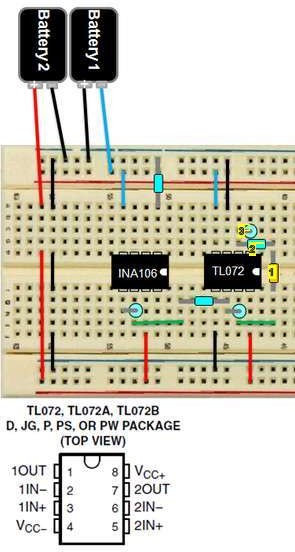
In this phase, we’re going to take those very small signals measured in the SIGNAL ACQUISITION phase and amplify them.

First we will be inverting amplifier with a gain of -15. An inverting amplifier does exactly what it sounds like. It amplifies your signal but also inverts it.

We can calculate the gain by G=-R2/R1 or in this case G=-150 kOhm / 10 kOhm. (See image 1)

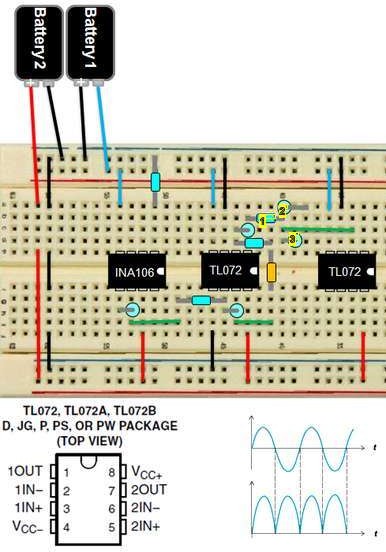
Next, we are going to add a capacitor to AC couple the signal. AC coupling is useful in removing DC error offset in a signal.

Next we are going to add another inverting amplifier with a gain



# 4. SIGNAL CONDITIONING - Rectification

In this phase, we will be rectifying the signal using an active [full-wave rectifier](http://en.wikipedia.org/wiki/Rectifier#Full-wave_rectification). Our rectifier will take the negative portion of our signal and turn it positive so the entire signal falls within the positive voltage region. We will use this coupled with a low pass filter to turn our AC signal in to a DC voltage; readying the signal to be passed to a microcontroller.

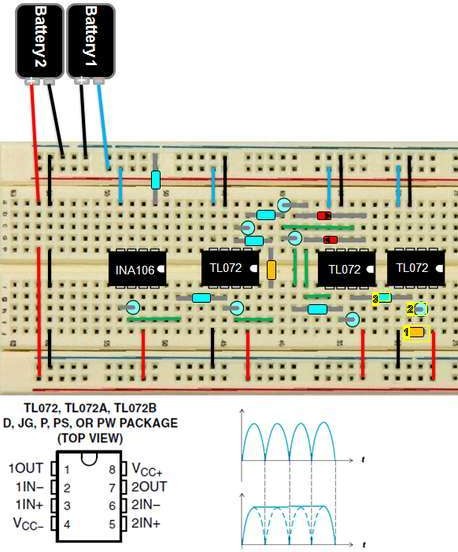


# SIGNAL CONDITIONING - Smoothing + Amplification

In this phase of circuit assembly, we will be using an active low-pass filter to filter out the humps of our signal to produce a smooth signal for our microcontroller.

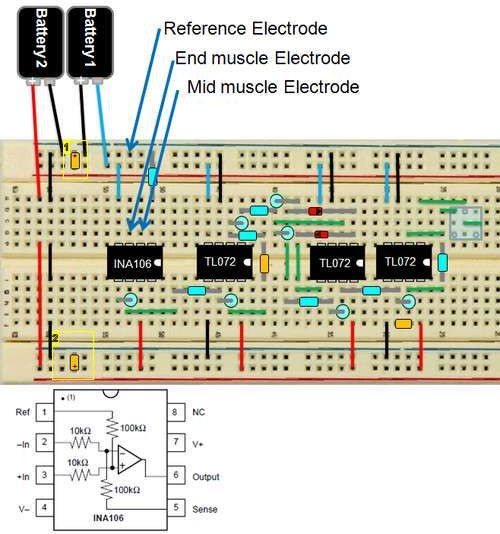
However, since this is an active filter, there is a side effect of inverting the signal. We will need to invert the signal one more time (and have the ability to amplify it more if desired) using another inverting amplifier circuit with a trimmer configured as a variable resistor.

By using a screw driver and turning the trimmer, you will be able to adjust the gain of your signal to account for different signal strengths from different muscle groups.



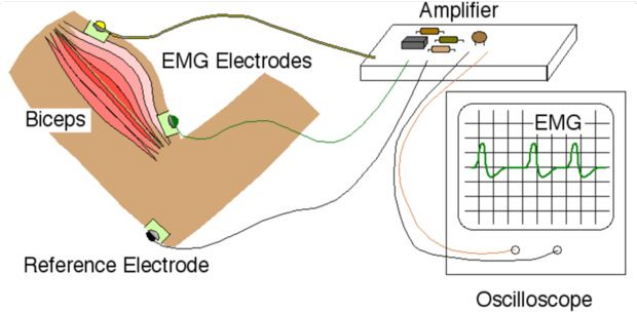
# Connecting Electrode Cables

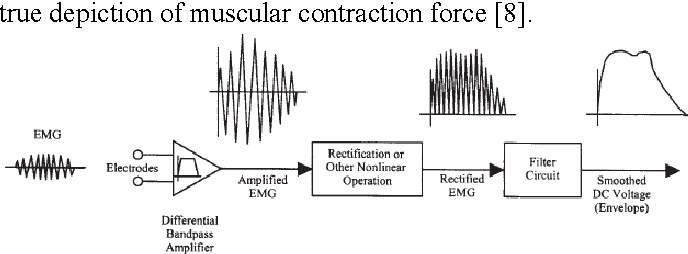
Connect the reference electrode to the GND rail of the circuit. Connect the mid muscle electrode to chip A's pin 2 connect the end electrode to chip A's pin 3



# Connecting the Electrodes

The EMG circuit requires three electrodes: positive input, negative input, and ground. The placement of the electrodes will vary based on the muscle that you intend to measure. For the bicep, the elbow is a suitable placement for ground. The positive and negative electrodes should be placed on the upper arm as shown in the figure. In my experience, the signal was stronger when electrodes were placed closer to the center of the body (medially) when the palm is facing upward.





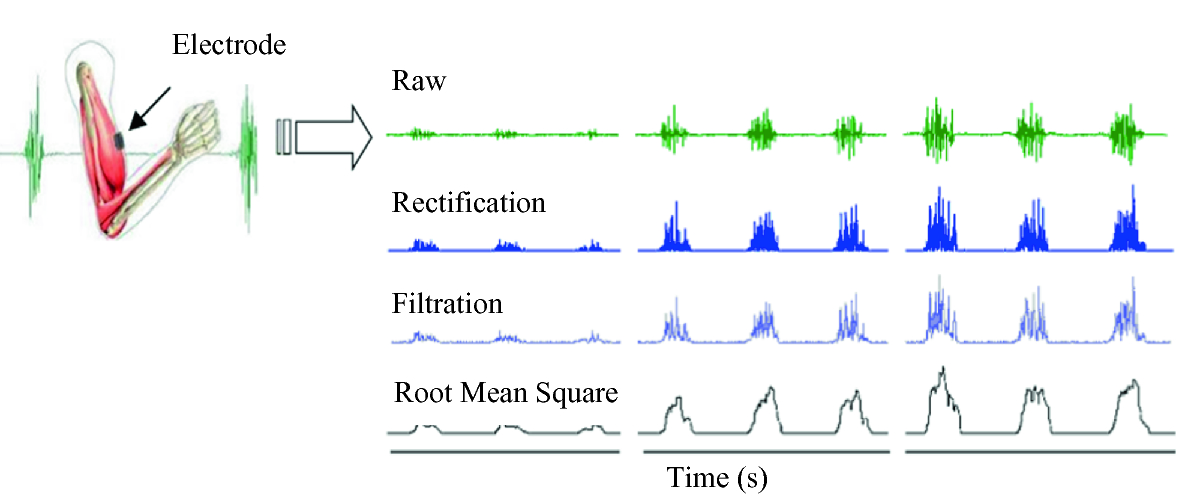


Fig 4.5 Signals at Different Stages

**Processing Codes Used in IDE’s:**

int lastsensorState = HIGH; // the previous reading from the input pin

unsigned long lastDebounceTime = 0; // the last time the output pin was toggled

unsigned long debounceDelay = 30; // the debounce time; increase if the output flickers

bool sensorstate = HIGH; //saving state of the switch

byte tapCounter; //for saving no. of times the switch is pressed

int timediff; //for saving the time in between each press and release of the switch

bool flag1, flag2; //just two variables

int analogpin = 0;

int val = 0;

int ledPin = 13;

//calibration

int temp=0;

int sens\_max = 0;

int sens\_min = 1023;

int threshold = 0;

bool state = HIGH;

//fist release should be last state

int prev\_state = 0;

int hand\_state = 0;

long double presstime, releasetime; //for saving millis at press and millis at release

void setup() {

Serial.begin(115200); //for serial monitor

pinMode(ledPin, OUTPUT);

digitalWrite(ledPin, HIGH);

hand\_calibration();

digitalWrite(ledPin, LOW);

threshold = (sens\_max - sens\_min) \* 0.25;

// digitalWrite(red, LOW);

// delay(1000);

// digitalWrite(red, HIGH);

// delay(1000);

// digitalWrite(green, LOW);

// delay(1000);

// digitalWrite(green, HIGH);

// delay(1000);

// digitalWrite(blue, LOW);

// delay(1000);

// digitalWrite(blue, HIGH);

}

void loop() {

val = analogRead(analogpin); //muscle sensor connected to pin A0 being stated as val

//Serial.println(val);

if( val < threshold) //if you flex and the sensor value is greater than 550 then close servos--adjust this value to your muscle sensor value

{

digitalWrite(ledPin, LOW);

delay(20);

//Serial.println("LOW");

state = HIGH;

}

else if(val > threshold)

{

digitalWrite(ledPin, HIGH);

delay(20);

//Serial.println("HIGH");

state = LOW;

}

int reading = state;

if (reading != lastsensorState) {

// reset the debouncing timer

lastDebounceTime = millis();

}

if ((millis() - lastDebounceTime) > debounceDelay) {

// whatever the reading is at, it's been there for longer than the debounce

// delay, so take it as the actual current state:

// if the button state has changed:

if (reading != sensorstate) {

sensorstate = reading;

}

}

//Serial.println(sensorstate);

//when switch is pressed

if (sensorstate == 0 && flag2 == 0)

{

presstime = millis(); //time from millis fn will save to presstime variable

flag1 = 0;

flag2 = 1;

tapCounter++; //tap counter will increase by 1

//delay(10); //for avoiding debouncing of the switch

}

//when sw is released

if (sensorstate == 1 && flag1 == 0)

{

releasetime = millis(); //time from millis fn will be saved to releasetime var

flag1 = 1;

flag2 = 0;

timediff = releasetime - presstime; //here we find the time gap between press and release and stored to timediff var

//Serial.println(timediff);

//delay(10);

}

if ((millis() - presstime) > 400 && sensorstate == 1) //wait for some time and if sw is in release position

{

if (tapCounter == 1) //if tap counter is 1

{

if (timediff >= 500) //if time diff is larger than 400 then its a hold

{

//Serial.println("Fist release");

hold(); //fn to call when the button is hold

}

else //if timediff is less than 400 then its a single tap

{

//Serial.println("fist full close");

singleTap(); //fn to call when the button is single taped

}

}

else if (tapCounter == 2 ) //if tapcounter is 2

{

if (timediff >= 500) // if timediff is greater than 400 then its single tap and hold

{

//Serial.println("Fist half close");

tapAndHold(); //fn to call when the button is single tap and hold

}

else // if timediff is less than 400 then its just double tap

{

//Serial.println("point");

doubleTap(); //fn to call when doubletap

}

}

else if (tapCounter == 3) //if tapcounter is 3 //then its triple tap

{

//Serial.println("pinch");

tripleTap(); //fn to call when triple tap

}

else if (tapCounter == 4) //if tapcounter is 4 then its 4 tap

{

//Serial.println("fingers close and open");

fourTap();//fn to call when four tap

}

tapCounter = 0;

}

lastsensorState = reading;

//Serial.print("status = ");

//Serial.println(state);

//delay(100);

Serial.print(hand\_state);

Serial.print("/");

// Serial.print(prev\_state);

// Serial.print("/");

Serial.println(val);

}

void nolight()

{

//digitalWrite(red, HIGH);

//digitalWrite(green, HIGH);

//digitalWrite(blue, HIGH);

//Serial.println("No light");

}

void singleTap()

{

nolight();

if(prev\_state == 0)

{

hand\_state = 1;

prev\_state = hand\_state;

}

else

{

hand\_state = prev\_state;

}

return hand\_state;

return prev\_state;

}

void doubleTap()

{

nolight();

if(prev\_state == 0)

{

hand\_state = 3;

prev\_state = hand\_state;

}

else

{

hand\_state = prev\_state;

}

return hand\_state;

return prev\_state;

}

void tripleTap()

{

nolight();

if(prev\_state == 0)

{

hand\_state = 4;

prev\_state = hand\_state;

}

else

{

hand\_state = prev\_state;

}

return hand\_state;

return prev\_state;

}

void fourTap()

{

nolight();

if(prev\_state == 0)

{

hand\_state = 5;

prev\_state = hand\_state;

}

else

{

hand\_state = prev\_state;

}

return hand\_state;

return prev\_state;

}

void hold()

{

nolight();

hand\_state = 0;

prev\_state = 0;

return hand\_state;

return prev\_state;

}

void tapAndHold()

{

nolight();

if(prev\_state == 0)

{

hand\_state = 2;

prev\_state = 2;

}

else

{

hand\_state = prev\_state;

}

return hand\_state;

return prev\_state;

}

void hand\_calibration()

{

//Serial.println("Calibrating");

while(millis() < 5000)

{

temp = analogRead(analogpin);

if(temp < sens\_min)

{

sens\_min = temp;

}

if(temp > sens\_max)

{

sens\_max = temp;

}

}

}

# Hand virtual training mode processing Code:

import processing.serial.\*;

import java.awt.event.KeyEvent;

import java.io.IOException;

Serial myPort;

String data="";

//float roll, pitch;

finger activate;

finger deactivate;

int x =0;

float val = 0;

float state = 0;

void setup() {

size(800,600);

rectMode(CENTER);

activate = new finger(color(255,0,0), 1.02);

deactivate = new finger(color(255,255,255), 1);

frameRate(30);

myPort = new Serial(this, "COM17", 115200); // starts the serial communication

myPort.bufferUntil('\n');

}

void draw() {

noStroke();

pushMatrix();

translate(150,-20);

textSize(22);

text("state: " + int(state) + " Pitch: " + int(val), 240, 300);

if (state == 0){

deactivate.pinky();

deactivate.ring();

deactivate.middle();

deactivate.index();

deactivate.thumb();

}

else if (state == 1){

activate.pinky();

activate.ring();

activate.middle();

activate.index();

activate.thumb();

}

else if (state == 2){

deactivate.pinky();

activate.ring();

deactivate.middle();

activate.index();

deactivate.thumb();

}

else if (state == 3){

deactivate.pinky();

deactivate.ring();

deactivate.middle();

activate.index();

deactivate.thumb();

}

else if (state == 4){

deactivate.pinky();

deactivate.ring();

deactivate.middle();

activate.index();

activate.thumb();

}

else if (state == 5){

deactivate.pinky();

deactivate.ring();

activate.middle();

deactivate.index();

deactivate.thumb();

}

fill(255);

arc(240,width/2+20,330,350, radians(0), radians(180), CHORD); // Palm

popMatrix();

}

// Read data from the Serial Port

void serialEvent (Serial myPort) {

// reads the data from the Serial Port up to the character '.' and puts it into the String variable "data".

data = myPort.readStringUntil('\n');

// if you got any bytes other than the linefeed:

if (data != null) {

data = trim(data);

// split the string at "/"

String items[] = split(data, '/');

if (items.length > 1) {

state = float(items[0]);

val = float(items[1]);

}

}

}

class finger {

color c;

float zoom;

finger(color tempC, float tempzoom) {

c = tempC;

zoom = tempzoom;

}

void pinky()

{

pushMatrix();

//stroke(1);

fill(c);

//scale(zoom);

rect(100,width/2-40,50,80);

rect(100,width/2-125,50,80);

arc(100,width/2-170,50,100, radians(180), radians(360), CHORD);

popMatrix();

}

void ring()

{

fill(c);

rect(170,width/2-60,50,120);

rect(170,width/2-185,50,120);

arc(170,width/2-250,50,100, radians(180), radians(360), CHORD);

}

void middle()

{

fill(c);

rect(240,width/2-75,50,150);

rect(240,width/2-230,50,150);

arc(240,width/2-310,50,100, radians(180), radians(360), CHORD);

}

void index()

{

fill(c);

rect(310,width/2-65,50,130);

rect(310,width/2-200,50,130);

arc(310,width/2-270,50,100, radians(180), radians(360), CHORD);

}

void thumb()

{

fill(c);

pushMatrix();

translate(380,width/2-60);

rotate(radians(20));

rect(15,-5,50,120);

popMatrix();

pushMatrix();

translate(420,width/2-125);

rotate(radians(20));

arc(0,0,50,100, radians(180), radians(360), CHORD);

popMatrix();

}

}

**CHAPTER 5 RESULTS AND DISCUSSION**

The prototype prosthetic hand, shown in Figure 3.4, has been tested for each finger separately. The test included several parts. First, the movements of each finger were tested while imitating a human hand’s movements. Second, the force was measured for each finger during the grasping process. Third, the EMG signals were determined for each finger. Furthermore, the feedback signal from each finger was tested as was the ability to control the vibration motor. Fifth, the total cost and weight of the prosthetic hand was presented for all components and materials.



Figure 8.1 The movements of middle finger. Figure 8.2 The movements of index finger.



Figure 8.3 The movements of pinky finger. Figure 8.4 The movements of ring finger.

From table 8.1 it is clear that the maximum x coordinate for pinky, ring, middle, index, and thumb are 7, 8, 9, 8, -6.57 respectively. The minus sign means that the coordinate lies opposite to the direction of the axis of the base. The table shows the results for only three different angles for each joint. Figures 8.1 ~8.4 show the space that each finger can move in during closing and opening operations.



Figure 8.5 Show the hand when holding a tennis ball in different situations.

From above table and figures, it is clear that the fingers can move in enough space to cover almost the same as the human’s movements. Where, the structure of the fingers is flexible, that’s means the finger’s joints can move all at the same time, or just one or two joints can move together to hold different shapes of objects. This structure will create the ability to grasp different objects as shown in the figures below:



Figure 8.6 Precision open. Figure 8.7 Tripod grip. Figure 8.8 Hook grip.



Figure 8.9 Mouse grip. Figure 8.10 Relaxed grip. Figure 8.11 Power grip.



Figure 8.12 Show different gripping patterns.

## 8.2 Signal Results

The second part of this chapter will discuss the signal results. First, it will show the EMG signal detected from the electrodes without any filters or treatment (except amplifying). Then, this section of chapter will show the effect of filters and amplifiers on this signal. Finally, it will show the final signal for each finger.

When the electrode detects the EMG signal, this signal will be very small and has a lot of noise as shown in Figures 8.13 and 8.14 below:

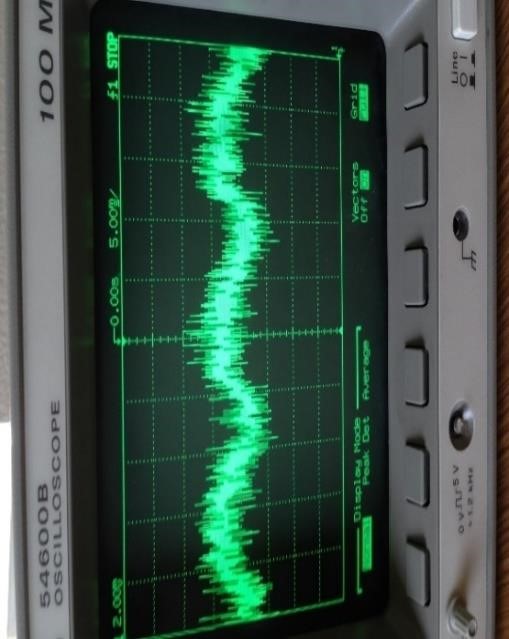
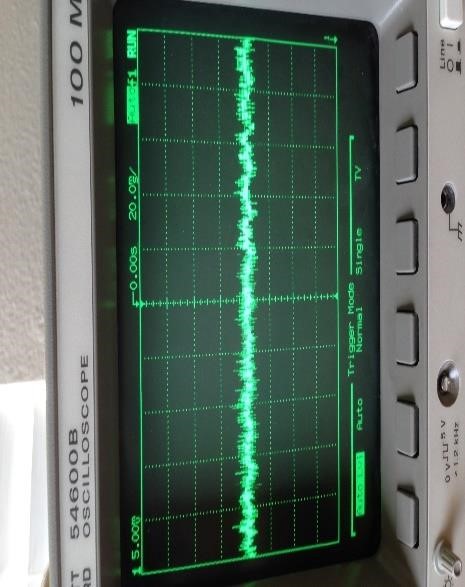
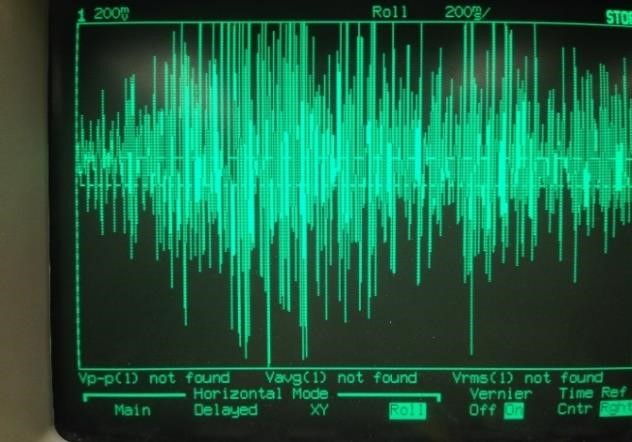


Figure 8.13 Signal when the muscle at Figure 8.14 Signal when the muscle at rest.

contraction.

Whereas, the signal in pictures 8.13 and 8.14 is measured from pin 6 at AD620 (instrumentational amplifier) that means this signal is amplified by about 100 times. Then, the signal will pass through pre-amplifier stage, low pass filter, high pass filter, and amplifier stage. The final signal of the circuit is shown in Figures 8.15 ~ 8.18 below.



Figure

8

.15

Two short contractions.

Figure

8

.16

Mu

scle at contraction

.

Figure

8

.17

Long contraction.

Figure

8

.18

Musc

le at rest.

From Figures 8.15~8.18 we can note that the signal has positive and negative values. The problem now is that we should remove the negative values to be readable by the Arduino. To do that, a diode will be used to remove the negative part of the signal as shown in Figures 8.19.



Figure 8.19 Signal after cutting the negative part.

When the Arduino receives this signal, it will sample this signal, and it will take the average of samples. In the second part of this section, the signal of each finger will be presented in pictures to show the amount and the frequency of this signal at contraction and rest.

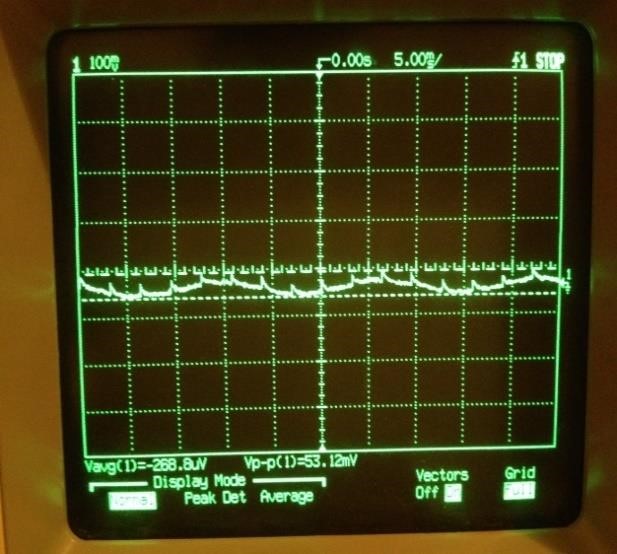


Figure 8.20 EMG signal of pinky finger.

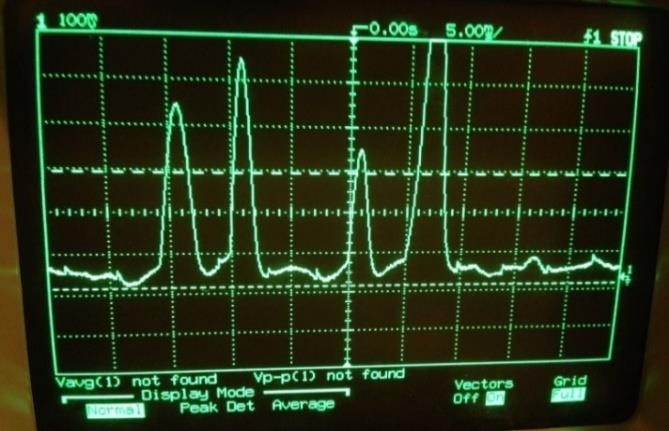


Figure 8.21 EMG signal of ring finger.

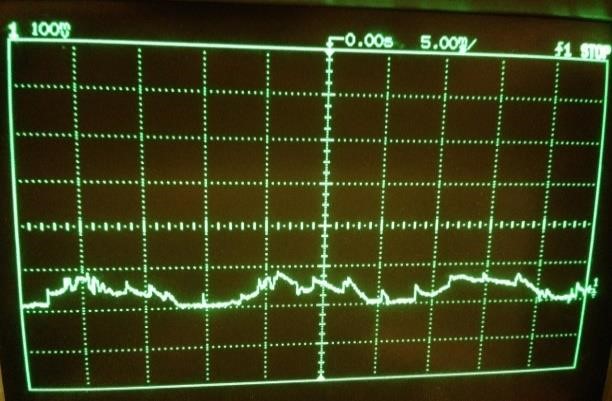


Figure 8.22 EMG signal for middle finger.



Figure 8.23 EMG signal of the index finger.



Figure 8.24 EMG of thumb.

**CHAPTER 6 ADVANTAGES, DISADVANTAGES AND APPLICATIONS**

**6.1 ADVANTAGES**

* Easy to Use
* Inexpensive
* Affordable to the Masses
* Light Weight
* Accurate Grips
* Repeatability and Mass Production is possible
* Less Training is Required
* Since Threshold is set few noise signals are also avoided

**6.2 DISADVANTAGES**

* Takes time to Train
* Many Small Tiny parts are required which are hard to assemble
* Assembling is Difficult
* Accuracy takes more time to Achieve
* Precision needs to Fine Tuned

**6.3 APPLICATIONS:**

* Prosthetic Hand acts as an Extension Of One’s Hand
* People Who have lost A Part of their Arm can use this Prosthetic Hand
* It can sometimes be used to reach places where a normal hand cant reach
* It can be used in Environments where a regular Arm cant be used
* It can be a substitute arm for Amputees

**CHAPTER 7 CONCLUSION**

This thesis project processes the design of a prosthetic hand controlled by the surface EMG signal. This cost-effective hand has 14 degree of freedom, and is made from aluminum sheet to be light and strong. The hand has five fingers, and each finger has three links except the thumb which has two links. Each finger is actuated separately by using servomotor which is controlled by Arduino. The Arduino controls the angle of the servomotor and the servomotor will pull a tie wrap to open or close the finger.

This prosthetic hand is controlled by the surface EMG signal which comes from the rest of the amputee’s arm. The surface EMG signal should pass through many steps to be readable by the Arduino. The first step is detecting the EMG signal from the muscle by using three electrodes. Electrode one is connected to the top of the muscle, electrode two is connected to the end of muscle, and the third electrode is a reference electrode which is connected to the bone.

The reference electrode could be common between many muscles. That means if there is a circuit to test five muscles, it will need eleven electrodes. One will be common between all muscles and two electrodes for each muscle. After detecting a signal from the muscle, the next step will be to subtract the signal of electrode two from the signal of electrode one to remove the common noise. Then the signal should amplify several times to be large enough to pass through the next step.

In this pre-amplifying step, the signal will amplify for several times because it has noise and we don’t want to amplify these noise. The next step is the filtering step where the signal should pass through the low pass filter then high pass filter to extract the EMG signal and remove almost all noise.

The final step in EMG signal processing is the amplifying step where the signal will be amplified to be big enough to read by the Arduino to make control for the hand’s fingers.

After designing the hand and getting the EMG signal, the project focused on getting a feedback signal from each finger. The feedback signal is the signal that tells the amputees if there is any object that is touching any finger. To do that, there is a force sensor connected to each fingertip which senses if there is any force applied on the finger. The way that the force sensor tells the amputee about this force, is by turning on the vibration motor which is connected to the amputee’s arm, the speed of the vibration motor related with the amount of the force that is applied to the fingertip. The design of the hand has been simulated by using the v-rep program. This design has good functions that make it very good and useful for amputees. It is affordable, has light weight, easy to fix, good finger control, and very good EMG detecting. I hope this project will make good difference in many amputees’ lives.

**CHAPTER 8 FUTURE SCOPE**

After successfully designing and testing this project there is some recommendations for future work on this project. First of all, the feedback sensing ability should be improved.

As mentioned before, each finger has a force sensor attached to the fingertip that can sense if there is something in touch with the finger. To improve the ability of sensing it is very important to increase the ability of the hand to sense in any part of the hand or fingers not just in fingertip. To do that that, I recommended creating an artificial skin which covers all hand and has a good sensing ability and good friction to catch the object.

On the other hand, this design can tell the amputee if the object touches the finger hardly of softly. But it cannot tell the amputee more properties about this object such as the temperature, if it is soft or not, if it is big or small, the shape of this object ...etc. Some of these properties are very important for blind people. To make this project very useful for blind people it is important to activate some feedback which tells the amputee about these properties.

* Brain Signals in Addition of EMG Signals can be used as inputs to power the upcoming prosthetic arms
* A mini Solar Panel can be used as a Substitute Power Source
* Electrodes Insertion can be made simpler and more Aesthetic

**REFERENCES**

* 1. C

1. Jessica Badger, “Meet the real-life bionic woman: Earlsfield fitness fanatic proud owner of hand built with F1 technology,” 2015.
2. M. Seo, D. Yoon, J. Kim, and Y. Choi, “EMG-based Prosthetic Hand Control

System Inspired by Missing-Hand Movement,” no. Urai, pp. 290–291, 2015.

1. Department of Physics and University of Illinois at Urbana-Champaign, “Q &amp;

A: Twisted Wires,” 2007.

1. S. K. Choudhary, D. Chakraborty, N. M. Kakoty, and S. M. Hazarika,

“Development of cost effective EMG controlled three fingered robotic hand,” Proc. 2012 3rd Int. Conf. Comput. Commun. Technol. ICCCT 2012, pp. 104–109,

2012.

1. P. Geethanjali and K. K. Ray, “A Low-Cost Real-Time Research Platform for

EMG Pattern Recognition-Based Prosthetic Hand,” IEEE/ASME Trans.

Mechatronics, vol. 20, no. 4, pp. 1948–1955, 2014.

1. X. Jing, X. Yong, Y. Jiang, H. Yokoi, and R. Kato, “A low-degree of freedom

EMG prosthetic hand with nails and springs to improve grasp ability,” Proc. - 2014

7th Int. Conf. Biomed. Eng. Informatics, BMEI 2014, no. 25249025, pp. 562–567,

2015.

1. A. L. Crawford and A. Perez-Gracia, “Design of a Robotic Hand With a

Biologically-Inspired Parallel Actuation System for Prosthetic Applications,” Vol. 2 34th Annu. Mech. Robot. Conf. Parts A B, vol. 2, no. PARTS A AND B, pp. 29–

36, 2010.

1. C. Scott, G. Sen Gupta, and L. Tang, “Sensing and processing of bio-metric signals for use in low cost bio-robotic systems,” 2014 IEEE Sensors Appl. Symp.

SAS 2014 - Proc., pp. 283–288, 2014.

1. V. Kumar and E. Todorov, “A low-cost and modular, 20-DOF anthropomorphic robotic hand: design, actuation and modeling,” 2013 13th IEEE-RAS Int. Conf.

Humanoid Robot., pp. 368–375, 2013.

1. N. Federico, “Metrological Issues Concerning Low Cost EMG- Controlled

Prosthetic Hand,” pp. 3–8, 2012.

1. Y. Jiang, S. Sakoda, S. Hoshigawa, H. Ye, Y. Yabuki, T. Nakamura, M. Ishihara,

T. Takagi, S. Takayama, and H. Yokoi, “Development and Evaluation of Simplified EMG Prosthetic Hands,” 2014.

1. H. Ye, S. Sakoda, Y. Jiang, and S. Morishita, “Pinch - Force - Magnification

Mechanism of Low Degree of Freedom EMG Prosthetic Hand for Children,” pp.

2466–2469, 2015.

1. D. Van Der Riet, R. Stopforth, and G. Bright, “Control Considerations of the Low

Cost Prosthetic Touch Hand,” pp. 225–232, 2015.

1. S.-Y. Jung, S. Kang, M. Lee, and I. Moon, “~ " ~ ~~ I,” pp. 83–86, 2007.
2. J. K. Jacqueline Louise Finch,lyn Harvey Heath,Ann Rosalie David,

“Biomechanical Assessment of Two Artificial Big Toe Restorations From Ancient

Egypt and Their Significance to the History of Prosthetics,” 2012.

1. Robert H.Meier, “History of Arm Amputation, Prosthetic Restoration, and Arm

Amputation Rehabilitation.”

1. ISAAC PERRY CLEMENTS, “How Prosthetic Limbs Work.”
2. M. R. Al-Mulla, F. Sepulveda, and M. Colley, “A review of non-invasive techniques to detect and predict localised muscle fatigue,” Sensors, vol. 11, no. 4, pp. 3545–3594, 2011.
3. RYAN CLINGMAN, “EVALUATION OF A NOVEL MYOELECTRIC

TRAINING DEVICE,” University of Virginia, 2010.

1. C. J. De Luca, “SURFACE ELECTROMYOGRAPHY: DETECTION AND

RECORDING.”

1. “3-lead Muscle / Electromyography Sensor for Microcontroller Applications

MyoWareTM Muscle Sensor (AT-04-001) DATASHEET.”

1. Texas Instruments, “LMx24-N, LM2902-N Low-Power, Quad-Operational

Amplifiers,” 2015.

1. “Force Sensitive Resistor (FSR).”
2. Kjell Magne Fauske, “Example: Annotated manipulator,” 2006.
3. M. W. Spong, S. Hutchinson, and V. M., “Robot Modeling and Control,” Control, vol. 141, no. 1, p. 419, 2006.
4. M. Zahak, “Signal Acquisition Using Surface EMG and Circuit Design

Considerations for Robotic Prosthesis,” in Computational Intelligence in Electromyography Analysis - A Perspective on Current Applications and Future

Challenges, InTech, 2012.

1. “Neuromuscular Junction Lecture Notes,” 2011.
2. Algonquin Academic, “Action Potential Generation in Skeletal Muscle,” 2013. [Online]. Available: https://www.youtube.com/watch?v=WrV510gUlco.

[Accessed: 25-Jul-2016].

1. Erica Jacques, “What is an EMG, and How Does it Work?,” 2014.
2. “Lifelines Neuro-diagnostic.”
3. “What is EMG (Electromyography) and how does it work?,” 2015.
4. Dr. Rich Vogel, “Overview of EMG,” 2016.
5. A. Devices, “Low Cost Low Power Instrumentation Amplifier AD620.”
6. P. Konrad, “The abc of emg,” A Pract. Introd. to Kinesiol. …, no. April, pp. 1–60, 2005.
7. V. B. Kitovski, “Electronic devices and circuit theory, 6th edition, R. Boylestad and L. Nashelsky, Prentice Hall International Inc., 1996, 950 pp. A4 (paperback),”

Microelectronics J., vol. 29, no. 8, p. 574, 1998.

1. S. N.kale, “Fundamental of Robotic Manipulator,” 2013.
2. “Kinematics (Advanced Methods in Computer Graphics) Part 1.”
3. Winifred Warren, “Introduction to Concepts in Robotics In this lecture, you will learn: -Robot classification -Links and Joints -Redundant manipulator -

Workspace.,” 2015.