DESIGN AND FABRICATION OF INTENTION BASED UPPER-LIMB EXOSKELETON

Thesis

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Master of Science in Electrical Engineering

By

Manoj Kumar Sharma

UNIVERSITY OF DAYTON

Dayton, Ohio

May, 2016

DESIGN AND FABRICATION OF INTENTION BASED UPPER-LIMB EXOSKELETON

Name: Sharma, Manoj Kumar

APPROVED BY:

Raúl Ordóñez, Ph.D.
Advisor Committee Chairman
Professor, Department of Electrical and Computer Engineering

John S. Loomis, Ph.D.
Committee Member
Professor Emeritus, Department of Electrical and Computer Engineering

Dave Myszka, Ph.D.
Committee Member
Associate Professor, Department of Mechanical and Aerospace Engineering

John G. Weber, Ph.D.
Associate Dean
School of Engineering

Eddy M. Rojas, Ph.D., M.A., P. E.
Dean, School of Engineering

ABSTRACT

DESIGN AND FABRICATION OF INTENTION BASED UPPER-LIMB EXOSKELETON

Name: Sharma, Manoj Kumar
University of Dayton

Advisor: Dr. Raúl Ordóñez

Exoskeletons, a wearable robot that intelligently augments the physical power of a human being. Lately these robots are finding their way towards the military and consumers as well. Generally speaking, our body has its own skeleton that helps in maintaining the posture. Often times fatigue becomes an important issue, especially those who regularly carry heavy loads; one solution to this is to attach a structure that can cling to a human body that can bear the load on its own. One of the biggest challenge is to design a structure that can fit snugly and can be actuated/commanded in way that feels natural and without inhibiting to many natural functionalities. The approach proposed here, focuses on a simple 3DoF upper limb exoskeleton; to actuate the exoskeleton, the intention is read from the pilot, which arguably, is one of the best way to control. The designed structure has some compliance to it, and this helps in reading the intention, which is then parsed through Dynamic Intention Filter (DIF) and feedback loops that eventually controls the torque of the motors. As a result, the final design feels so natural that there is, almost, no learning curve to its operation.

Dedicated to my family.

ACKNOWLEDGMENTS

Thanks to Dr. Raúl Ordóñez, for being so kind and supportive, Dr. John S. Loomis and Dr. Dave Myszka for their commitment and valuable guidelines that made this design possible.

Special thanks to:

George Sutton (Chief Engineer, Yaskawa Motoman Inc.), for providing his valuable feedback and guidelines.

Timothy G. Klopfenstein (Chief Research Machinist, University of Dayton), for machining the parts used in the fabrication.

Yaskawa Motoman Inc. for letting me use some of the components.

And finally, all my friends for being supportive.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

BACKGROUND

Exo-skeleton is essentially a framework that can compliment an existing skeleton, as needed. Exoskeleton is a vague term and hence may refer to many different things. In this case we are going to focus on exoskeletons for the a human being, that are active in nature, instead of being passive.

1.1 Human skeleton anatomy

Credit: *MOVDATA [1]*

Figure 1.1: Human skeleton and muscle anatomy

Human body consists of a structure made up of bones, called skeleton. This skeleton (figure 1.1) is fundamentally responsible in maintaining a correct body structure/posture. It helps in transferring any external load and self body weight, acting on it, to the ground. These bones are connected to one or many other bones in a wide variety joints and then secured by muscles and tendons. Muscles are responsible for providing push or pull as per the requirements.

1.2 Motivation

Humans have always been pushing the limit, which is inevitable as the technology advances. When it comes to carrying loads, like the soldiers [2] do, fatigue becomes an important issue. Moreover, in the medical field, like people who cannot the move their limbs or may be unable to maintain their balance by its own or people suffering from stroke, Parkinson's disease, Multiple Sclerosis [3] or simply old aged people can be benefited from an exoskeleton, that can assist them in various tasks. A study [4] published in International Conference on Robotics and Automation in 2011, proposes a design of $3-$Dof upper-limb exoskeleton using force sensor array at more than one locations (arm and cuff), proposing intentional reaching direction (IRD). Further research [5] was carried forward and published in October 2015 with fixes in performances issues. Reading intention requires a finite compliance between the exoskeleton and the pilot, which allows the force sensor to work. On the other hand, researchers [6] also use the Electromyograph (EMG) signals, which are essentially electrical signals. The advantage with EMG is that, if filtered properly, a precise control of exoskeleton can be executed, but the problem is that, it is very hard to filter and the non-invasive methods requires a proper calibration and placement as well.

1.3 Proposed solution

Although the solution proposed here is entirely new, but it is still in its embryonic state, lately a lot of research has been going on in different parts of the world. For now we are going to focus on

this approach wherein the exoskeleton intelligently augments its own torque with that of a human being (or the pilot). Not to mention, like any other robot, like this one, requires some sort of input to actuate the motors, which in this case is the pilot's intention, or in other words, it (exoskeleton) is compliant to the pilot, wherever the pilot wants to move the arm, the exoskeleton follows. Theoretically any load on the exoskeleton can not be experienced by the pilot's arm as it (load) gets shared by the exoskeleton. Figure 1.2 shows a typical exoskeleton [7], being operated by the user (or pilot), as we can see that the external load gets transferred to the ground through exoskeleton and that is why the pilot might not feel anything similar to carrying a load.

Credit: *Popular Science [7]*

Figure 1.2: A typical exoskeleton

This design has one of the caveats that it requires a healthy person to operate. Even an important thing to keep in mind is that designing an exoskeleton is a complex task and (obviously) as the number of degrees-of-freedom (DoF) increases, the exoskeleton starts feeling more natural, which makes it difficult to manufacture and control as well. Here lies a balance where the decision has

to be made on the design which should be suitable for a given application. For simplicity, a 3DoF, upper limb exoskeleton design is proposed here and fabricated. One of the first things needed prior to the design is to understand the human skeletal anatomy, at-least the basic range-of-motions (RoMs). This RoM data taken from Orthopedic Reviews[[8]], is shown in figure 1.3 and becomes the foundation for the further design. The angles are tabulated in degrees, the column N_a shows the safe [8] range of motions for respective joints, which is less than that of the actual range a normal human can exhibit. This is good because we definitely do not want to max out the limit and must provide some factor of safety and thereby providing some room, just in case anything goes wrong.

Joints Movement	Na	Median	Normal R
Shoulder Flexion	108	119 (30-153)	150-180
Shoulder Abduction	117	98 (75-130)	180
Wrist Flexion	109	55 (30-75)	60-80
Wrist Extension	116	30(8-64)	60-70
Elbow Flexion	107	135 (110-150)	140-150
Elbow Extention	106	-27 (-51-10)	0-85

Credit: *Orthopedic Reviews [[8]]*

Figure 1.3: Safe joint angles

CHAPTER II

MECHANICAL ARCHITECTURE AND ANALYSIS

In this section we will focus on the design perspective, mechanism used to facilitate the motion and computer aided analysis to validate the design. The design focuses on three degrees of freedom which is explained (in part) in figure 2.1, actuated by a brushed DC motor through a spindle-drive mechanism, which will be discussed later in this section.

2.1 Design requirements

Designing a robot plays an important role in achieving a desired goal, stability, efficiency and etcetera and especially, when we think of an 'exoskeleton' as a robot clung to a human body, then utmost propriety is to protect the human being by minimizing the risk associated with operating the exoskeleton. Isaac Asimov's "three laws of robotics [9]" says that:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.

2. A robot must obey orders given it by human beings except where such orders would conflict with the First Law.

3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.

2.2 Design overview

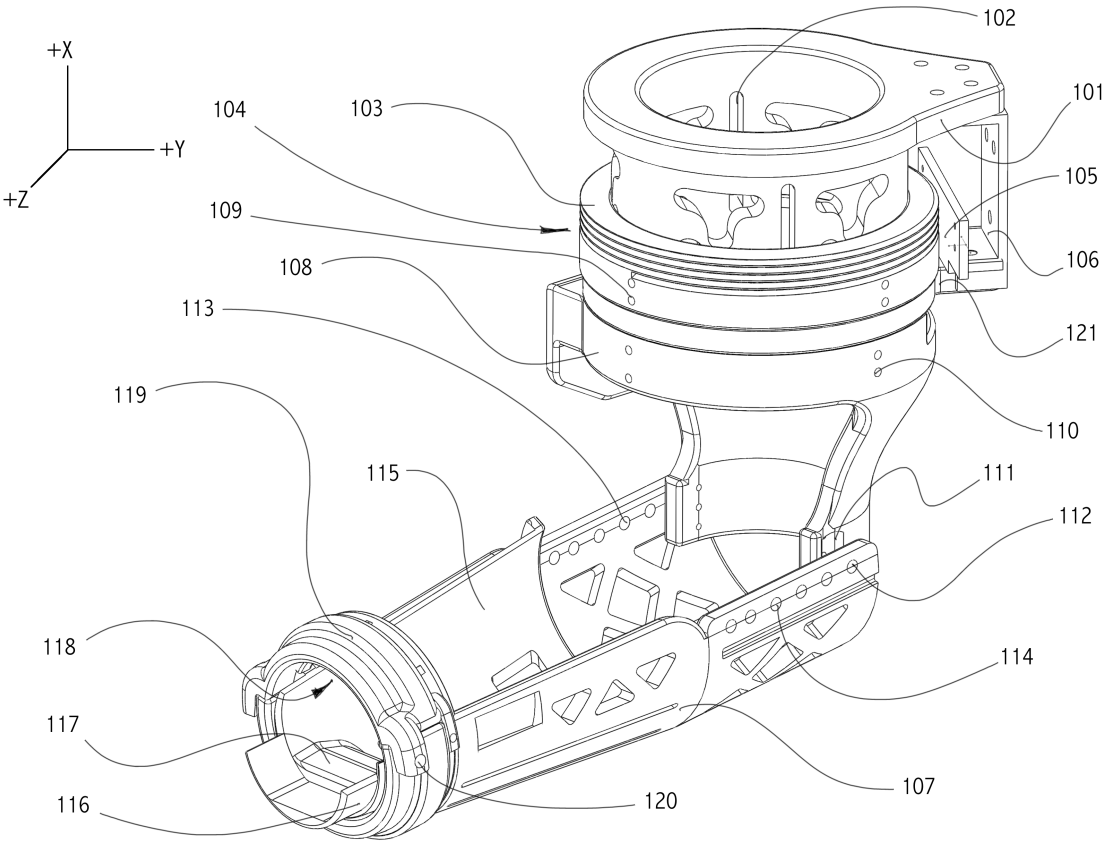


Figure 2.1: Design overview

Any stable design is a result of research and many iterations and so is this design. Considering the safety aspects and an appropriate power dense mechanism, this design, arguably, has so many advantages over something similar. Going back to the design of interest, figure 2.1 shows a wire diagram of the proposed structure. Here, as we can see, the whole structure is anchored on 106, which then explodes into a cylindrical structure (101). This laterally encloses the pilot's cuff portion. On top of this two cylindrical members (103 & 108) are allowed to glide smoothly; 103 and 108 are independent members and, after the assembly, its locked in place using a segmented-curved

aluminum plate at several in-line holes, 109 & 110. Member 103 has threads on it (104) through which a push-pull cable runs, for about 3 turn through the plate 105, and allows the other end of the cable to be actuated at some other location; this subsequently forces 108 to move accordingly about X-axis, let's call it yaw. At the end of 108, a pair of ball bearing support (111 & 112), another structure, 107, responsible for arm support. This very joint is driven by a spindle-drive which can be assumed as a linear motor with one end pivoted 108, while the other end at 113 or inline with 114, which will cause different ranges of motion and torque augmented to the elbow (pitch). Next is a member (115) that glides freely within the arm support (107). 115 is designed in such a way that it makes a direct contact with human arm and plays a crucial role in determining the pilot's intention using a unique placement of sensors and design (discussed later in fugure 2.2). 107 and 115 contains straight and oblique slots respectively which allows 115 to glide up-to 60°. This very motion extends up-to 170° in human body but in this case, due to design constrains and safety issues, it is allowed to rotate up-to 60° only, corresponds to roll about Z-axis, in normal configuration.

This structure is predominantly 3D printed using M-30 grade ABS plastic, capable of bear the overall load, however, the structure is strongest at its weakest point, especially the locations where the cuff and arm is hinged (112) & actuated (114). To overcome this, a portion of 107 containing the features like 113, 112, 114 are reinforced with $Al\ 3003 - H14$ grade, 1.8 mm aluminum sheet. For now the entire structure is clamped (at 106) to a test rig that holds the structure and transfers the load to the ground; this also allows us to conduct various testings and calibration before actually augmenting it on-to a pilot.

Now that we've seen how the structure behaves and its capabilities, let's focus on the sensor placement, a key component in determining the intention of a pilot. As we have discussed earlier, members 115 glides within 107, shown in figure 2.2 with a special ring configuration; The exploded view shows 118 is held securely with 115 (using a string that passes through the internally carved

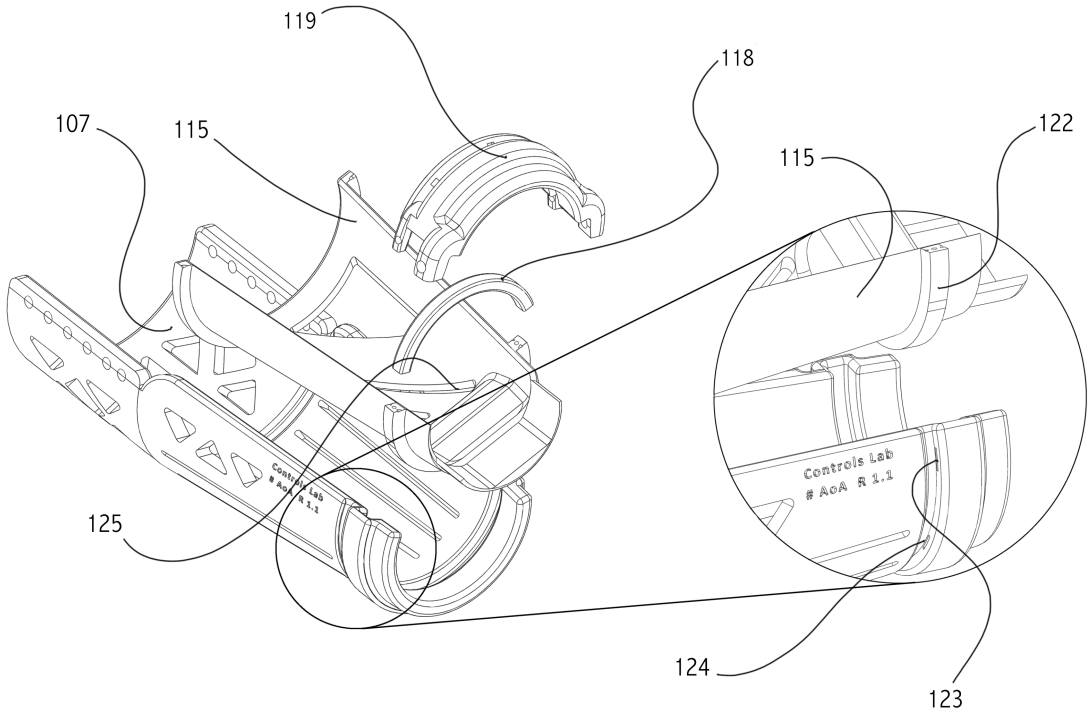


Figure 2.2: Arm's exploded view, shows sensor placement

thrugh-hole) and this assembly (115 & 118) remains stationary with respect to the pilot's arm. once 115 and 118 is in place, the final cap 119 is attached on 107 and then secured in place using four side screw mounts. The magnified view of this arrangement is shown in figure 2.2, here the slots (on 107), arranged in a circular fashion are shown, 124, 123 and the rest on the other sides. These slots allows the force sensor resistors (figure 3.1(a)) to be slid into a slot where the protrusion 112, (on 115), comes in a direct contact with the FSRs. Once 115 is disturbed by a pilot, the resulting force/ intention is quantized by the sensors and the associated electronics, as shown in figure 3.2.

2.3 Mechanisms

Now that we have discussed about the design requirements and its overview, let us focus on how those motions are achieved. Figure 2.3 shows all the three motions (Roll, Pitch and Yaw) in the design and uses different mechanism, chosen on-purpose, for the sake of convenience. The section below explains the mechanisms that are responsible for those motions.

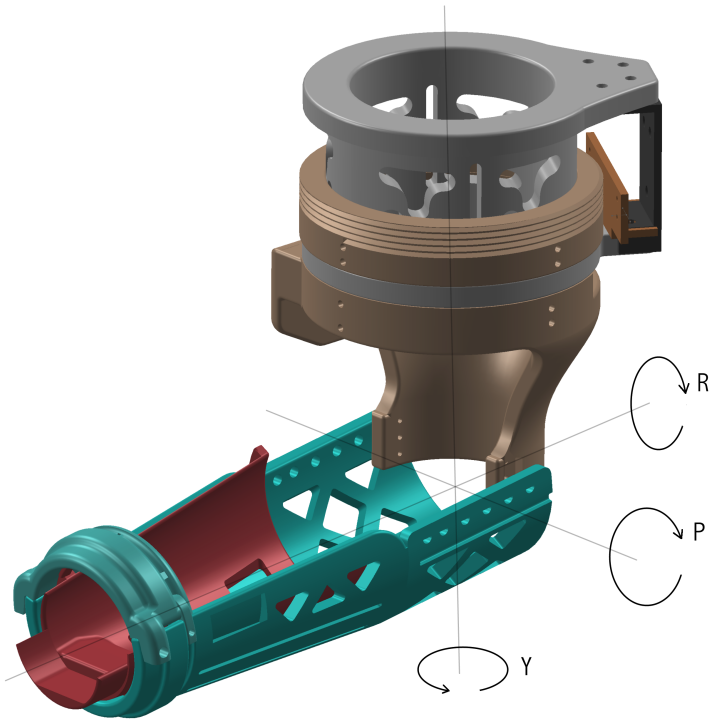
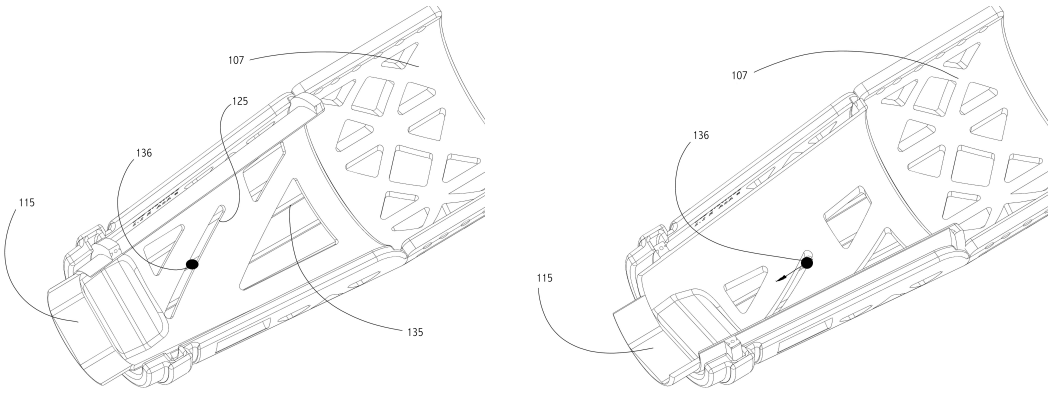


Figure 2.3: Degrees of freedom in human arm

2.3.1 Roll

Roll, as shown in figure 2.3 involves slider-pin mechanism. Figure 2.4 shows the working of this, slider-pin, mechanism. Considering 107 to be temporarily fixed, 115 is then allowed to glide over 107. A pin, 136, is engaged between a vertical slot, 135, and an oblique slot, 136, on the gliding



(a) Position A (b) Position B

Figure 2.4: Slider pin mechanism, roll

portion, 115. The oblique angle of the slot (136) dictates the range of motion, with kinematic and various other limitations. Figure 2.4(a) shows the pin at one extreme position (Position A) and as it moves towards its another extreme (Position B), the roll angle of 115 (arm potion secured to human wrist) changes, as shown in 2.4(b).

This pin is secured by a tension spring on 107 (arm) and thereby forcing the floating arm portion (115) at it one of the extreme. This natural tension (by the spring) on the pin is countered by a pull cable (similar to 133, 134; figure 2.8). The cable, within the sheath, is pulled by a linear motor, which in turn pulls the pin towards the other extreme and thereby allowing 115 to roll.

2.3.2 Pitch

The flexion/extension of arm is facilitated by a simple three link mechanism. The structures, 108 and 107 (figure 2.5) serves as two of the three links and the third link is a linear motor, 127. Figure 2.7 provides a detailed view of this linear motor and its placement. 107 is hinged to 108 at 112 (figure 2.1). As the motor is actuated, the coupled reducer head (126) drives a spindle screw

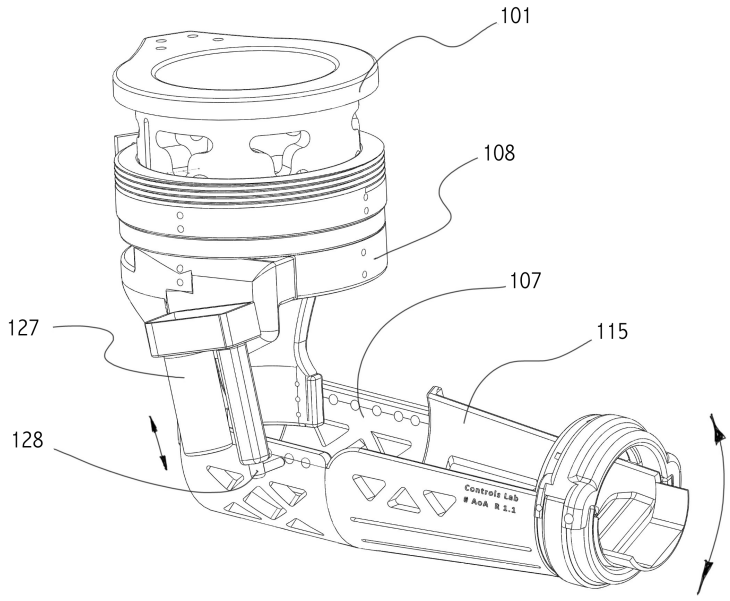


Figure 2.5: Expanding link mechanism, contracted position

mechanism which in turn allows the pin (128) to reciprocate. Figure 2.5 shows the pin at one extreme, as the motor is actuated, the pin reciprocates linearly and forces 107 to rotate about the pivot (112), and eventually reaches a new position as shown in figure 2.7, 21.00 - 81.00 degrees.

Let us compute the theoretical pitch angular velocity ω_p as a function of linear velocity along $AC(v_{AC})$. Figure 2.6 shows a simple link arrangement responsible for the pitch, here the link length AC is a variable, while AB, BC and BD are known and fixed as well.

$$\omega_p = \frac{v}{BC}$$

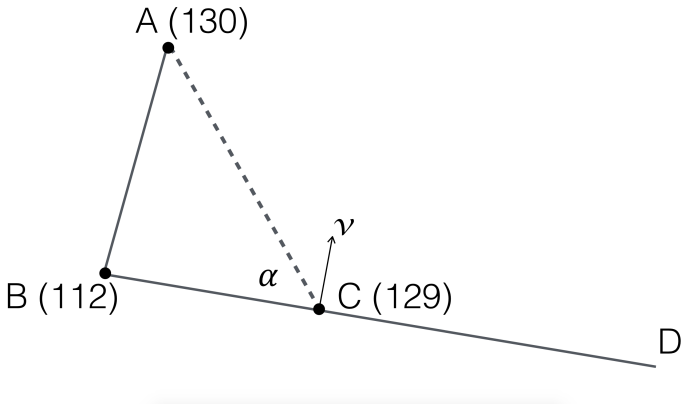


Figure 2.6: Pitch linkage design

$$\omega_p = \frac{v_{AC} \cdot \sin\alpha}{BC}$$

Using cosine law,

$$AB^2 = BC^2 + AC^2 - 2 \cdot BC \cdot AC \cdot \cos\alpha$$

$$\alpha = \cos^{-1}\Big(\frac{BC^2 + AC^2 - AB^2}{2 \cdot BC \cdot AC}\Big)$$

$$\omega_p = \frac{v_{AC}}{BC} \sin\Big(\cos^{-1}\Big(\frac{BC^2 + AC^2 - AB^2}{2 \cdot BC \cdot AC}\Big)\Big)$$

Substituting $AC = 0.081m$, $BC = 0.060m$, ($v_{AC\ max} = 0.0127m/s$)

$$\omega_p = 16.67 v_{AC} \sin\Big(\cos^{-1}\Big(\frac{0.01016 - AB^2}{0.00972}\Big)\Big)$$

However at 12V, the maximum average angular velocity (computed by the encoder) is $0.056\ rad/s$ or $0.0046\ rad/sV\ :\ 4 \leq V \leq 12$. As per the data sheet provided by the manufacturer, the maximum linear force the actuator can impart is $490N$. Thus, torque about 112 (τ_{112}) is given by:

$$\tau_{112} = 490N \cdot BC = 490 \cdot 0.06\ = 29.4\ Nm$$

Maximum force available at D (F_D)is given by:

$$F_D = \tau_{112} \cdot \frac{1}{BD} = \frac{29.4\ Nm}{0.27\ m} = 108.8\ N$$

The force is a function of the input voltage, with a maximum of $108.8\ N$ at 12 V; this is considered while designing and in the following sections, we will see the static load analysis for the arm at $120\ N$ load is within the fracture limit.

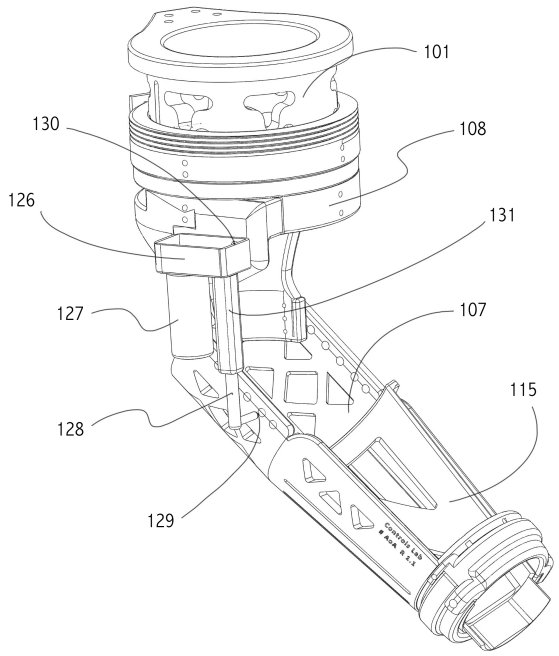


Figure 2.7: Expanding linkage mechanism, relaxed position

2.3.3 Yaw

The above figure 2.8 shows the mechanism involved in the yaw motion. The cylindrical section 104 contains circular grooves, spaced at 3.5mm, wrapped around $1440°$, that translates to 4 turns. A (pull) cable (133) is wrapped around this helix and then guided, through the cable sheath(134), to another helix, 132, again wrapped around it and secured using screws to restrict any relative motion between the cable (133) and the driver helix (132). This driver helix is coupled to the outer spline of

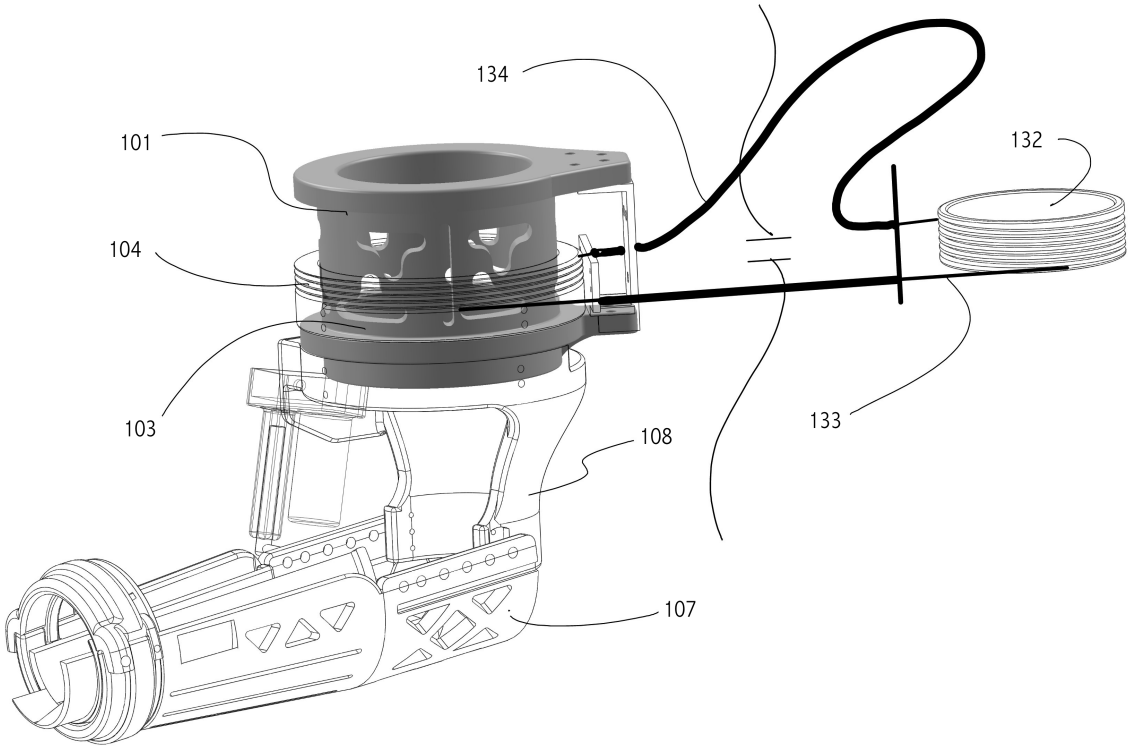


Figure 2.8: Pull cables, yaw

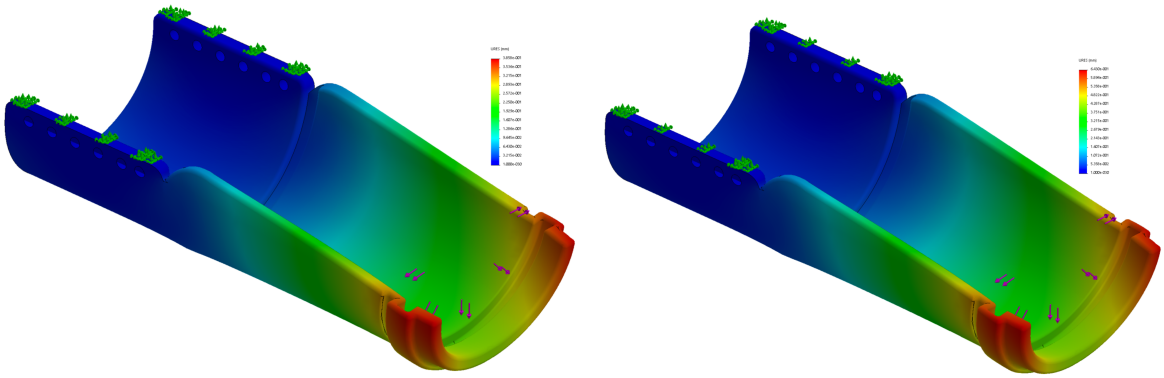
a harmonic reducer [10], capable of providing a reduction ratio of $160:1$. A PMDC motor (10Nm @ 6000RPM at 12V) drives the inner spline of the harmonic reducer, then reduced to a maximum of $6000/160 = 37.5$RPM. Since the reducer (132) is mechanically coupled to 104, it becomes the final drive in this mechanism. Multiple turns on 104 allows a larger surface are and thus a larger friction, which is good in this case, and allows a better clutch. Gear reduction happens two times, 160, followed by $(160/100 = 1.6)$, therefore from initial drive is reduced by $160 \cdot 1.6 = 256$ times. The motor is rated at $150\ W$, $12\ V$, $0.24\ Nm$, @6000 RPM; at its final drive the torque increases by a factor of 256, and thus we get $61.44\ Nm$ torque for the yaw motion at $37.5/1.6 = 23.44$ RPM or $2.45\ rad/s$ at its designed maximum for a range of 44.00 - 121.00 degrees.

2.4 Static Load Analysis

For the above design, a relatively new technique, 3D-printing is used to fabricate/print the components which is fast and cost effective as well. The material used here is ABS-M30 for sparse typed printing, rather than solid-printing. Sparse printing reduces the weight and the material consumed, significantly. Most of the designing is done in Solidworks (student version) followed by a static load analysis.

As mentioned earlier, the components 3D printed out of ABS plastic, one of the approach taken to reduce the amount of material used and enhance the strength, was to include several features, like circular or triangular voids. These added features make the deign utterly complex, which isn't a problem for 3D printing, however makes it substantially difficult for Finite-Element-Analysis (FEA). Computer-Aided-Designing (CAD) program needs to mesh the component, that is being analyzed, and the presence of features requires a painstaking process of segmented meshing in vicinity of those features. A study [11] shows a stress analysis using HyperMesh, which is a novel approach has a potential to be used in this research. As mentioned earlier that the part designed within the CAD being solid, while the printed part being sparsed. Given these factors, the only quick and reliable way for FEA is to assume the part being solid and without the features. A best judgement is made to evaluate the weakest part in the whole design, which in this case is the arm (107). To conclude, a displacement and stress analysis is performed on this arm portion for 120N and 200N, applied radially at the end of it. Figure 2.9 shows the result of displacement analysis done on the arm section for various loads. (Solidworks student version supposedly has this caveat that forbids the simulation to be exported as a full resolution, hence the legends are little hard to read; a better resolution will be added at the end of the report for reference though).

As per the inferred result, in the case of 120 N and 200 N radial load, the max displacement was around $3.580\ e-001$mm and $6.430\ e-001$mm respectively. In both the cases, the displacements



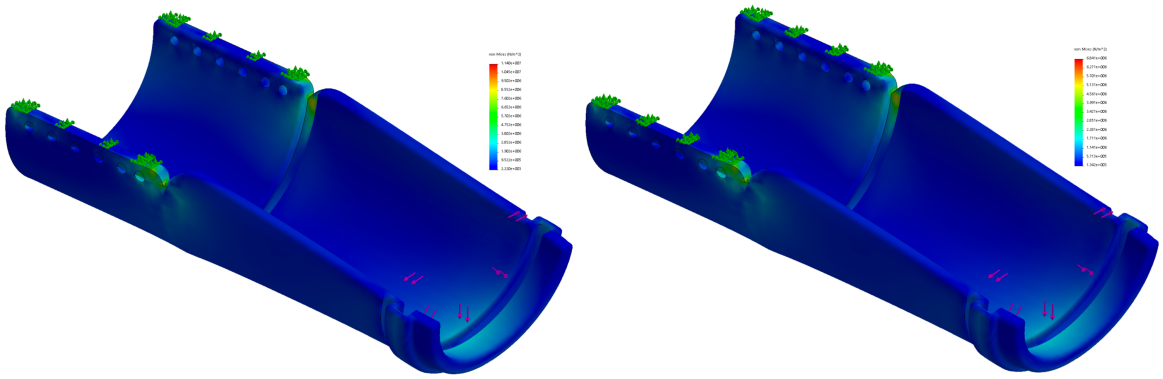
(a) 120 N (b) 200 N

Figure 2.9: Displacement analysis for different radial loads

are normalized, that is, the maximum displacement regions are depicted by red color and the least displacement regions are blue in color. Theoretically, these displacements are not high in magnitude, but it does provides the characteristics of the component, that helps in approximating its fracture point and its likelihood. Figure 2.10 shows the induced stresses as a result similar radial loads. Subfigures (a) and (b) shows the maximum stress induces are $6.84\ e+006 N/m^2$ and $1.40\ e+007 N/m^2$ for 120 N and 200 N respectively. As we can see that in both the figures, there are almost no regions marked with red, which means that (theoretically) the component can be safely loaded with 200 N without fracturing the structure.

However, in practice, the aforementioned values does not hold true, for the fat that so many assumption made during the analysis. Another important fact to consider here is that the material strength of the 3D printed parts are not consistent, predominantly because of the fact that, every successive layer laid by the machine is, sometimes not consistent. Especially the events like the filament replacement or similar events that requires and thereby significantly delaying the successive

layers. (Usually the inbuilt oven helps to keep the ongoing part warm enough that the next layer bonds to the previously laid, even after some delay, but never works perfectly).



(a) 120 N (b) 200 N

Figure 2.10: Stress analysis for different radial loads

CHAPTER III

ELECTRICAL ARCHITECTURE

In the previous chapter we have discussed about the mechanical structure and their design architecture which results in the desired motion. Now, the actuators are the members which completes the bridge from electrical to mechanical subsystem and vice-versa. In this section, therefore, we will focus on the electrical components like sensors, actuators, controllers and power distribution, and then we will see how we all of these can be interfaced to a micro-controller which could be processing and then signaled for the actuation.

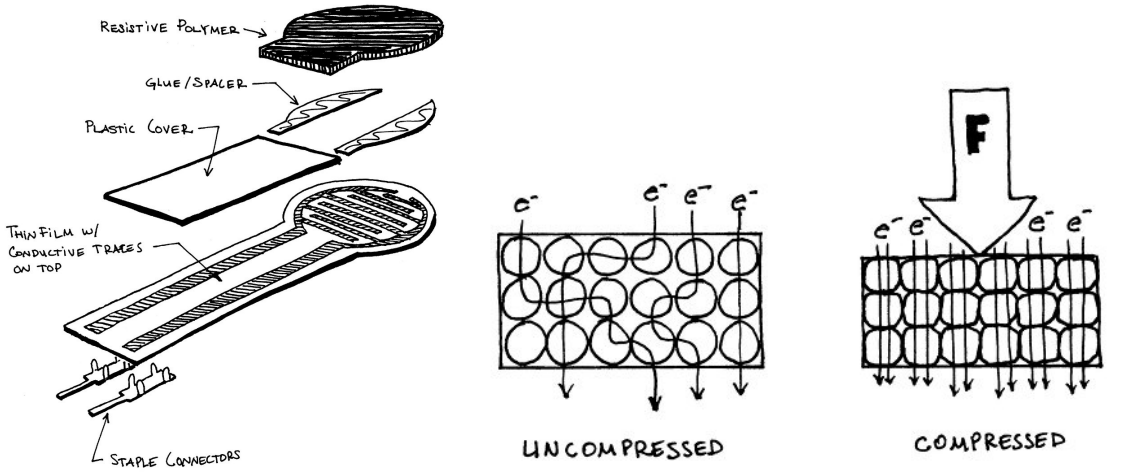
3.1 Sensors

Sensors are responsible for measuring the output of a system which is a crucial component of a closed-loop control system design . In practice its very rare to rely solely on the system's model and have open-loop design to control a system, as it drastically worsens the capability to cope up with any disturbance(s), simply because the output is not measured or fed back into the controller. This section deals with variety of sensors being used to design the system, their working, interfacing with a micro-controller and etcetera.

Sensor selection is not a trivial task; a same output can be measured by so many different ways but as a design engineer one has to make a wise decision based on the factors like accuracy, ease of use, cost and so on, which are discussed in this sub-section.

3.1.1 Force Sensor

For every robot, some sort of signal is needed to actuate the actuators. The idea behind the intention based exoskeleton is that the intention of the uses is interpreted physically and then converted into electrical signal, for further processing. This can be done by allowing the design to have a little compliance, which allows the force sensor to sense the force being applied on to it.



(a) Exploded view, FSR (b) Working

Credit: *Open music labs [12]*

Figure 3.1: Force sensing resistor concept

Technically these sensors are known as Force Sensing Resistors (FSR); shown in figure 3.1(a) as an exploded view. We can see that there are base lining of a conductive material, like tin-oxide, coated with the resistive polymer with a pair of spacer sandwiched in between. The resistive polymer contains carbon atoms, which are conductive in nature, with some space around them, as shown in figure 3.1(b). When the polymer is not in stress, then electrons have to find a path through randomly placed carbon atoms, now as a force is applied, these atoms starts to compress and as a result

the electron makes their way from one end to another quickly, parameterized by the amount of force applied.

Now that we know that sensor essentially exhibits a change in resistance, whenever subject to an external load, One of the easiest way to interpret this is to convert this change in resistance to a proportional change in voltage, which then can be fed to analog to digital converter (ADC). The micro-controller being used here has sixteen inbuilt ADCs capable of providing resolution up-to ten bits ($2^{10} = 1024$).

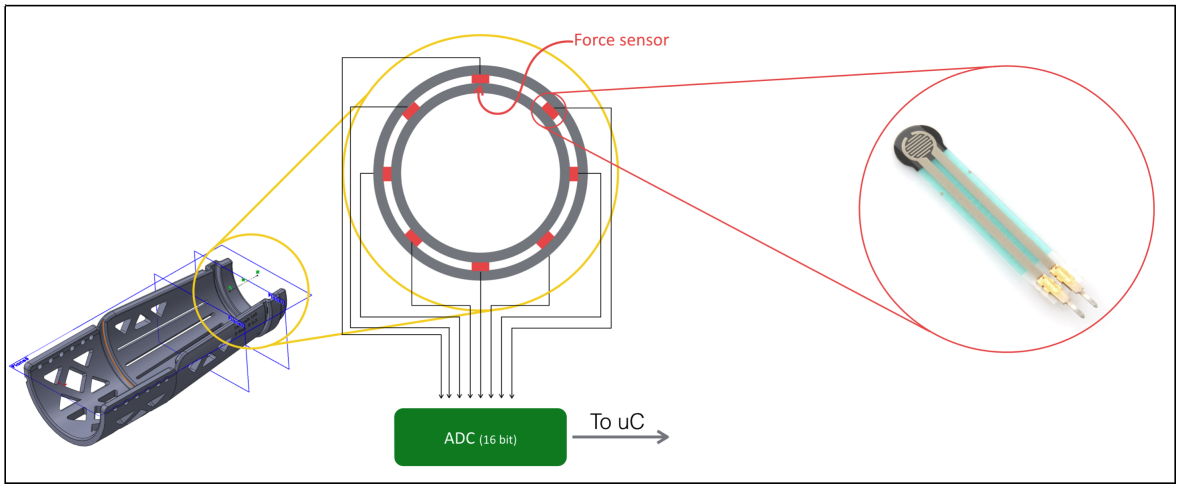


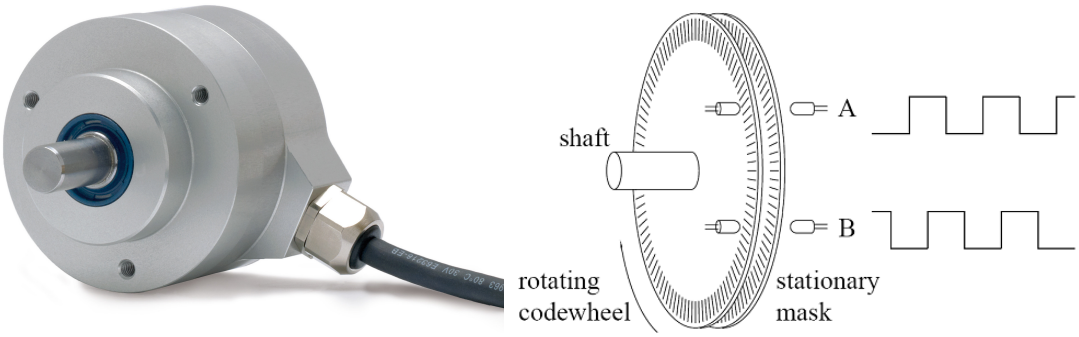
Figure 3.2: Force sensor array design and placement

Figure 3.2 shows how the FSRs (shown in red) are placed/fixed on the outer ring, facing inwards, in a circulation array within the exoskeleton arm design. The inner ring almost touches the FSR's open surface and remains stationary with respect to the pilot's arm; as a result, any pilot's movement (direction and magnitude) is recorded by this FSR array which is essentially the intention. Mathematically a transfer function which compensates for the spatial orientation is generated in real time which then effectively mapped to the respective actuators, thereby augmenting the force to the pilot.

3.1.2 Encoder

Generally speaking, closed loop control has so many advantages over its counterpart. The brushed motors, which we discussed earlier, needs some sort of feedback. To achieve this, we have so many options available which can be broadly classified under Relative encoder and Absolute encoder.

Absolute encoder, as the name suggests, outputs the absolute angular position; in essence no matter what happens, the angular position is read and available on demand. Simply a (rotary) potentiometer is an example of such encoder whose output is absolute, always, assuming the resistor is behaving ideally. This assumption is really not the case in reality and affects the output immensely due to addition of noises (however, its way less costly than any other encoders). Some absolute encoder uses a combination (in full or a part of) relative encoder (discussed below), storage, a control circuit and a power source to keep track of any movement (These are sometimes available in a single package for quick setup and are pretty easy to implement).



(a) Quadrature encoder (b) Differential output

Credit: *(a) Metronics [13] (b) Neuroscience and robotics lab [14]*

Figure 3.3: Quadrature encoder concept

Rotary optical encoder or rotary encoder or quadrature encoders, figure 3.3(a), are used interchangeably, and refers to a type of encoder wherein every discrete change in angular position, based on its resolution, is followed by change in the electrical outputs, as shown in figure 3.3(b). These encoders are precise and can easily provide resolution up-to 16 bit and almost insusceptible to noise(s). A typical quadrature encoder outputs six different signals which are commonly labeled as A, B, I and their complementary signals $\bar{A}, \bar{B}, \bar{I}$. A minimum of A and B is required to read the encoder's output.

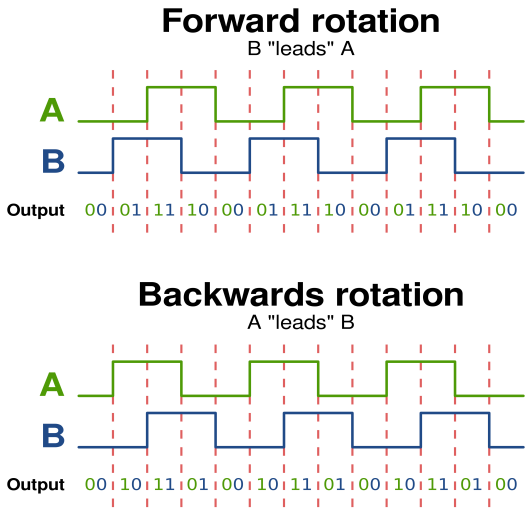
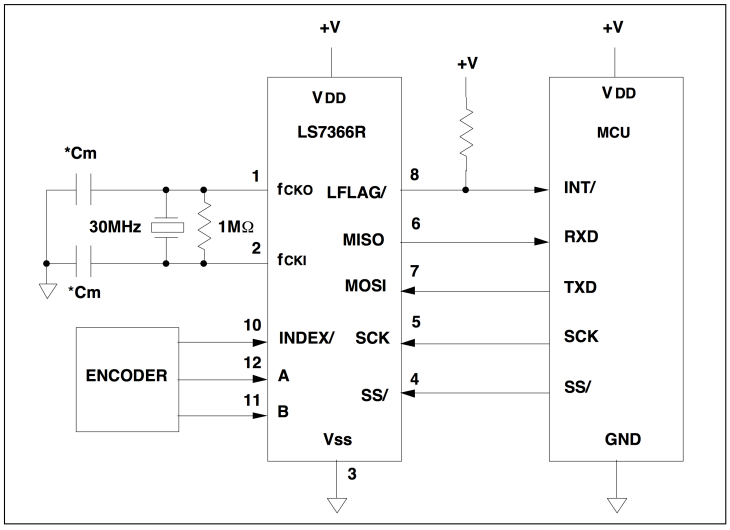


Figure 3.4: Quadrature encoder directional output

Credit: *Wayne and Layne [15]*

We can see that way in which A and B outputs the signal based on the direction of rotation and the frequency depends on rotational speed and encoder's resolution. The rate of change in states determines the angular speed, the number of times the state has changed dictates the angular position, and the way these output changes depends on the direction of rotation as seen in figure 3.4. As we can see that the fact that the counting starts when the controller is powered on and thus

picks up from 0 (and hence the name 'relative'). Now to read the change in the states of A and B, a specific hardware pin, called, 'interrupts' are used which have an added advantage of reading the pins at its highest level of priority. Ideally the controller ($8\ MHz$) executes the process at every 1.25×10^{-7} seconds or 8×10^{6} process in a second. The quadrature encoder has a resolution of 10 bits, that means it changes its output 1024 times per revolution. Now if the encoder is rotating at 7812 revolutions per second, it would, at-least, require $7,999,488$ to measure these changes. In reality this limit (7812 revolution per second) is way less, probably because of the interrupt handling routine and other background process and as a result the controller fails to count the change and ends up presenting a false data. To fix this issue, its good to use a dedicated encoder counter chips (IC). One such example is LS7366R, 32 bit programable counter and clocked at $30\ MHz$. The counts are stored locally and hence doesn't consumes main controller's precious clock cycle; a circuit diagram is shown in the figure 3.5 below.

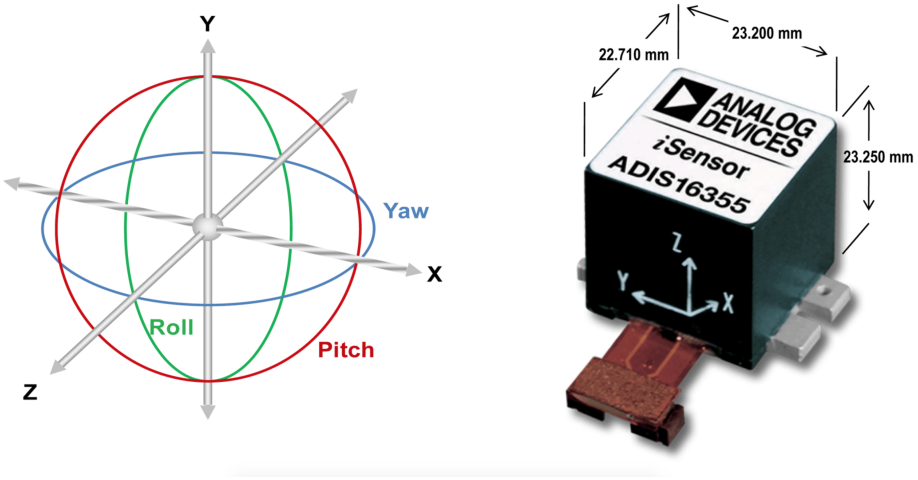


Credit: *LSI/CSI [16]*

Figure 3.5: Circuit dragram for LS7366R encoder interfacing

3.1.3 Inertial Measurement Unit (IMU)

Inertial measurement unit is a sensor package consists of usually 3-axis accelerometer, 3-axis magnetometer and 3-axis gyroscope of Microelectromechanical System (MEMS) type. Accelerometer measures acceleration action on all three axes; gyroscope measures the rate of change of angular position in all three axes; and magnetometer detects the magnitude of earth's magnitude on all of its three imaginary axes. Figure 3.6 shows an IMU (ADIS16355) by Analog devices. Spacial disorientation can be overcome using the IMU and running algorithm like Attitude Heading Reference System (AHRS) which yields Roll, Pitch and Yaw, commonly found in aircraft navigations (HUDs).



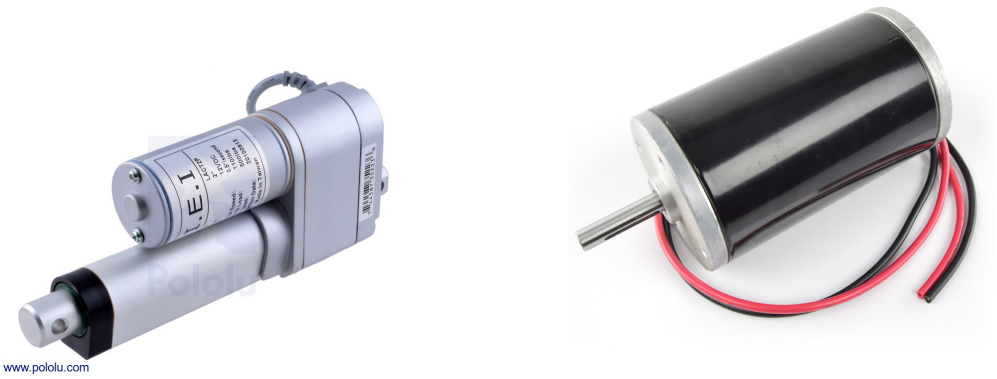
Credit: *Analog Devices [17]*

Figure 3.6: IMU by Analog devices

3.2 Motors / Actuators

Motors (or actuators) are the elements that convert the electricity into mechanical work, thus it is quiet hard to categorize them into a pure electrical or mechanical domain. Nevertheless, every

degree of freedom uses a different mechanism (as discussed earlier) and therefore different motors. All three motors used here are Permanent Magnet Direct Current (PMDC) motors, which are relatively easy to control.



(a) Linear Motor (b) BLDC motor

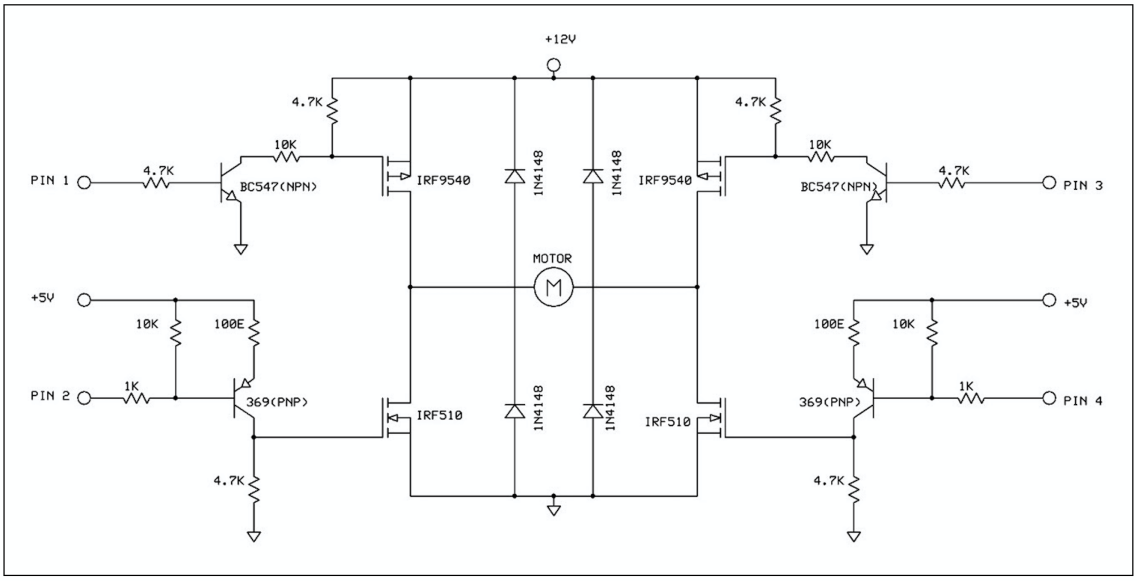
Credit: *(a) Pololu electronics [18] (b) BaneBots [19]*

Figure 3.7: Actuators

For the roll, the linear motor shown in figure 3.7(a), is directly used to drive the pin as discussed in the earlier sections. As evident from the pitch design mechanism, a linear motor is used for the purpose, which is different from the linear motor shown above; rather, this motor was disassembled and a set of custom machined parts are secured on its existing spindle drive. This helped in reducing the weight significantly and fitting the motor in the given space. For yaw, the BLDC motor (figure 3.7(b)) is directly coupled to the input spline of the harmonic drive.

3.3 Motor Driver

The DC brushed motor, discussed earlier, can draw up-to 5 amperes of current at its peak load and about an ampere with no load; while the micro-controllers usually operate within few micro-amperes (to be precise, < 50 mA) there is no way a micro-controller (on its own) can provide enough current to run this motor. To overcome this issue, often a motor driver (or amplifier) is used which accepts the (low magnitude) command signals and fires a combination of metal oxide semiconductor field effect transistors (MOSFETs) to provide a current of desired magnitude and direction to the motor.



Credit: *Next electronics [20]*

Figure 3.8: Simple H-Bridge circuit

In essence, an H-Bridge consists of four MOSFETs as shown in figure 3.8. The direction of current flowing through the motor can be switched by triggering appropriate set of MOSFETs; for

example, if PIN 1 and PIN 4 is triggered, the motor starts rotating (say) clockwise and on the other hand when PIN 2 and PIN 3 is triggered we get a counter-clockwise rotation. These MOSFETs can be triggered by the micro-controller directly, as it requires a few milli-amperes of current, however in this design the H-Bridge has an onboard controller to handle some complex functions and reduce the number of pin requirement to just two. An important thing to note here is that, at any given time none of the two pairs (PIN 1,2 or PIN 3,4) should be triggered, as it results in short-circuit! The aforesaid onboard controller has an added advantage that it eliminates this very conflicts.

The above section explains how the direction of a brushed DC motor can be controller using H-Bridge circuit; rotational speed of the motor can be controlled simply altering the voltage across the motor as the rotational speed of a brushed DC motor is (predominantly) a function of voltage and torque applied on the rotor as well.

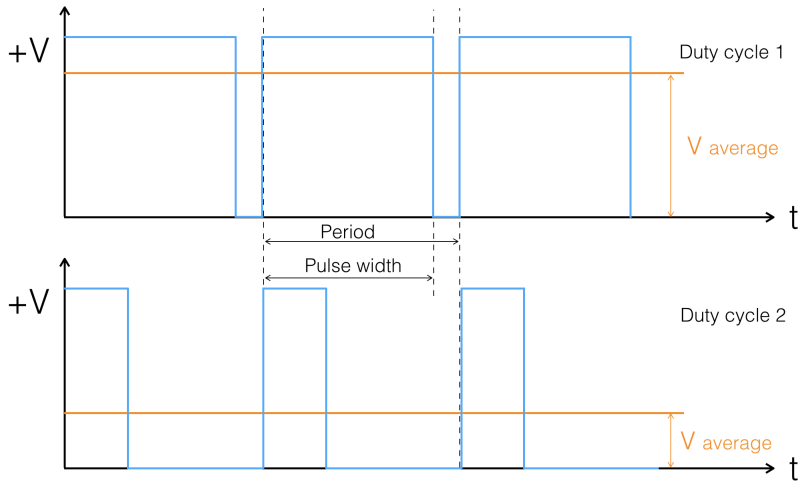


Figure 3.9: Pulse width modification (PWM) and averaging

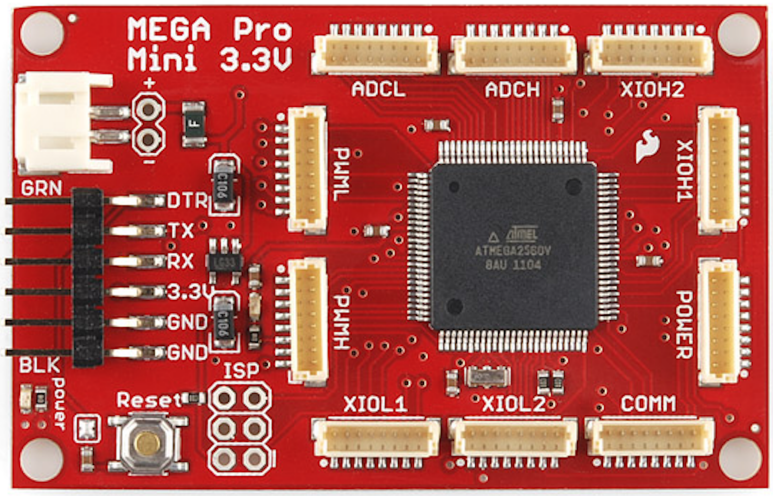
Pretty much every robots are powered by a constant voltage DC source (including this one), therefore to tap an arbitrary voltage (V_{av}) where, $0 \leq V_{av} \leq +V_m$. Pulse width modulation [21]

(PWM) technique is implemented. The controller generates a pulse at every predefined intervals, in this case it is 20 KHz, this means a pulse is generated every 5×10^{-5} seconds. These pulses can be forced to be $0V$ or $+V_c$ (V_c is controller voltage, $+5V$ in this case) or in general, these pulse can be maintained at $+V_c$ for any length of time within the cycle, that is from $0\ s$ to $5 \times 10^{-5}\ s$ and this fractional duration is termed as duty cycle. A 100% duty cycle refers to a pulse which is on for the entire cycle and for every successive cycles which ultimately means that the average (V_{av}) equals to the main voltage (V_m); consequently a 50% duty cycle means that a pulse is on for 50% of its cycle period $(2.5 \times 10^{-5}\ s)$ for every successive cycles and as a result $V_{av} = V_m/2$. The same is true for any duty cycle ranging from 0% to 100% and subsequently a proportional fraction of V_{av} can be fed to the motor (which ultimately changes the magnitude of current flowing through the motor), however in practice the motor has some threshold duty cycle to start rotating. At this point its fair to say that changing duty cycle changes the motor's RPM proportionately (for a constant torque), as shown in figure 3.9 how two different duty cycles (duty cycle 1 and duty cycle 2) results in different respective V_{av}, and this very property is used to design a feedback controller for position control.

3.4 Micro-controller

Micro-controller (uC) can be treated as the brain for any robot as these are responsible for commanding the associated components to work in a desired manner (pretty much what a brain does to our body). All the measurements are sent to the micro-controller for processing and since uC has direct control over all the actuators, it can then control these actuators based on the controller algorithm(s).

The micro-controller used here is Arduino Pro Mini, figure 3.10, runs at $V_c = 3.3V$ logic, uses ATMEGA 2560 chip, clocked at $8\ MHz$. This uC features too many essential features (discussed below) in a single package. All features these can be utilized by uploading desired 'sketch' using its proprietary open-source integrated development environment IDE. The overall design has more than



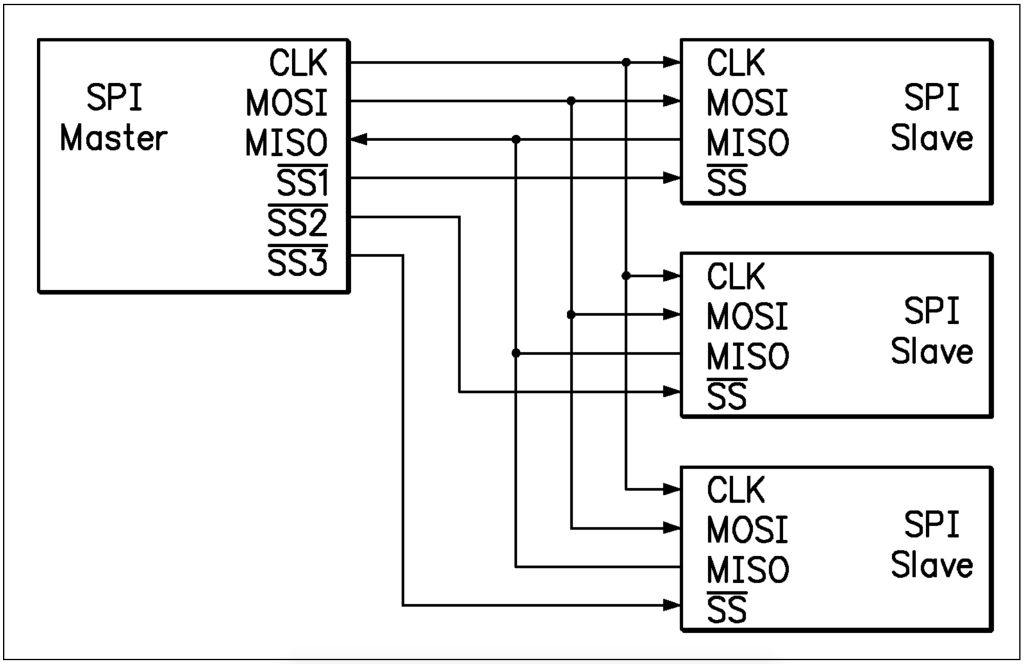
Credit: *SparkFun electronics [22]*

Figure 3.10: Arduino mega pro 3.3V

one uCs running simultaneously and exchanging data back and fourth over a predefined protocols like SPI, I2C or UART.

ADC: Any analog voltage $(0V \leq V \leq V_c)$ can be fed directly into one of the 16 available pins for its digital conversion at a resolution of 10 bits. Apparently there is only one analog to digital converter (ADC) on the chip, while the architecture has multiplexer which allows only one (out of 16) conversion at a time, however this process is so fast that in practice its fair to assume that the conversion happens in real time.

Communication: As the complexity of any system increases, its quiet possible that a single uC simply can not handle all the workload or even sometimes its the only way a component communicates (like the quadrature encoder counter IC) as set by the manufacturer. Whatever the case it, there are three well known communication protocols used by the uCs to exchange the data; the main uC is usually called master and all other treated as slaves.



Credit: *Google patents (US 20120072628 A1),* *modified [23]*

Figure 3.11: Serial Peripheral Interface (SPI) administration

SPI: Serial Peripheral Interface, is a synchronous serial communication which means that all the data are transferred serially between the master and slave upon synchronizing the clock. This usually requires four physical signal wire (along with a common ground for reference) namely master-in-slave-out (MISO), master-out-slave-in (MOSI), shared-clock (SCK) and chip/slave-select (SS). The data exchange only happens over MISO and MOSI, while per the nature of this communication they (master and slave) share a common clock (SCK) pulsed by master for synchronization. The advantage is that the data transfer is very fast (in fact could be the fastest among the three) while a disadvantage of having a separate wire. The chip select (SS) allows the master to wake up any slave selectively to exchange the data. Figure 3.11 shows how the SPI is administered. The number of

SPI slaves are limited by the number of available digital output pins available on the uC which can act as SS, one for every SPI slave.

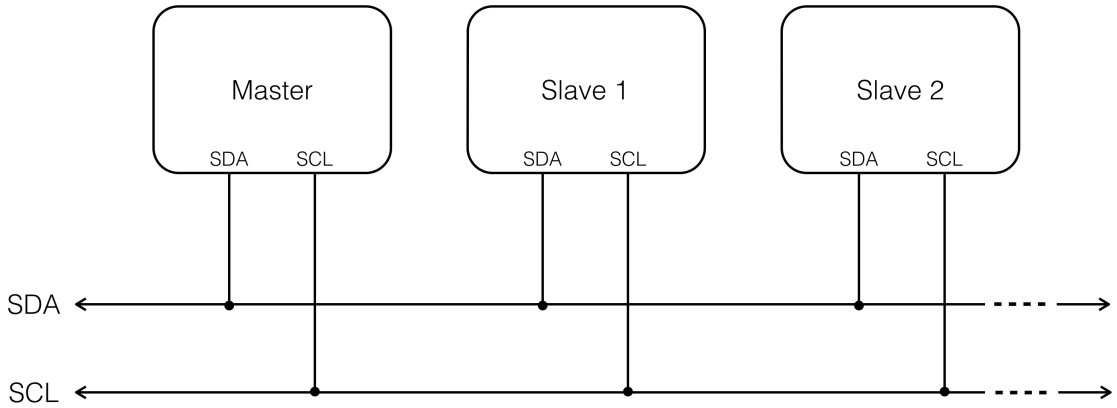


Figure 3.12: Inter Integrated Circuit (I2C / IIC / TWI) communication

I2C: Inter Integrated Circuit also known as two wire interface (TWI) is another type of synchronous communication which requires fewer number of pins than the SPI which comes at a compromised data transfer rate. As shown in figure **??**, like SPI the I2C's administration has master and slave(s) with a clock shared by the master (SCK), but unlike SPI there is no slave select that means all the devices remain active over the line; each slave as an unique address (typically assigned by the manufacturer) and that is how the devices recognize each other; furthermore the data in and out is multiplexed into a single data line which means, at a given instant, the data can only be transmitted or received and this is why it is relatively slow. Nonetheless this type of communication is pretty effective in devices which does not require to communicate very often.

UART: Universal Asynchronous Receiver/Transmitter is an asynchronous type of serial communication wherein the devices have their own clock and agree to operate at a common pre-determined frequency and thus eliminating the need for a separate clock line which comes at a cost of chances

of data loss at higher transmission rate, on the other hand, it fits the best for wireless communications if operated within an appropriate baud rate. Figure 3.13 shows a typical UART communication which require Receive (R_x) and Transmit (T_x) pins (along with a ground for reference which may or may not be common).

Figure 3.13: Serial / UART communication

UART communication is peer-to-peer (P2P) and unlike SPI or I2C a channel cannot be used for more than one device at a time. As mentioned earlier that this is a serial communication where the data, in the form of an character, is (must be) transmitted from T_x and subsequently read by R_x and vice versa. This R_x and T_x pair is termed as (say) UART1 and often times predefined by the manufacturer and cannot be changed (while FPGAs have this exception).

In this design SPI is being used predominantly by the uC which communicates to the quadrature encoder counter and to another uC that handles the array of force sensors and some vital safety tasks like arming the system and etcetera, for a safe operation. Along with this some data are transmitted over UART to a computer to visualize the desired parameters and results, and for diagnostic purposes.

3.5 Block Diagram

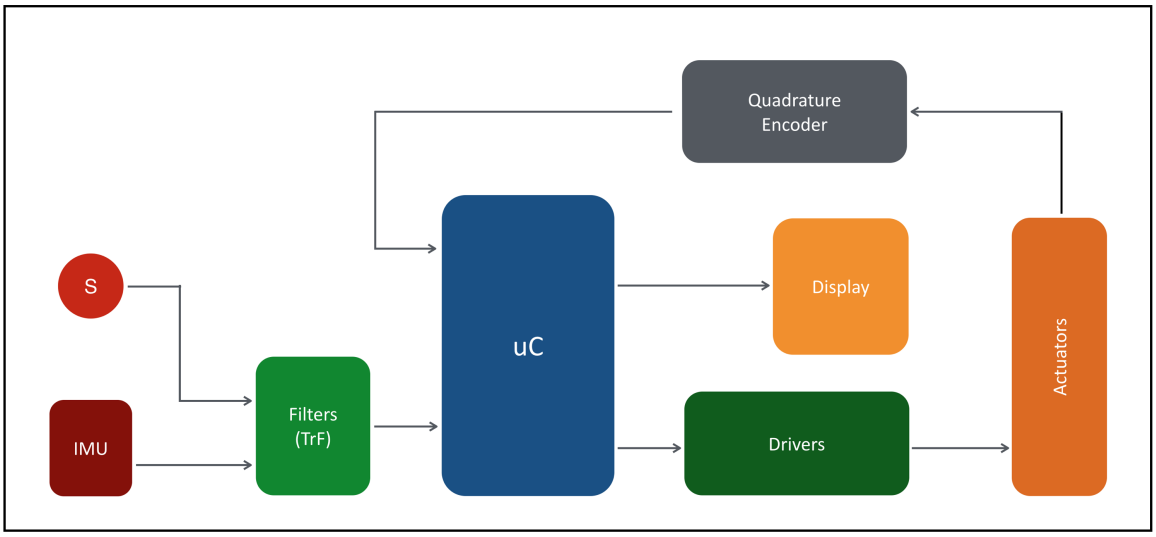


Figure 3.14: Electrical block diagram

Now that we have seen all the necessary elements needed for the system, let's see how architecture looks like in a simple block diagram as shown in figure 3.14. The sensors feed the intention related data to the Dynamic Intention Filter (DIF) along with the inertial data, to the micro-controller. Based on the system's model, the micro-controller triggers the motor driver. These motor drivers amplifies the current, sufficient enough to drive the motors. All the encoder positions are fed back to the micro-controller for a feedback-control. A small display including indicator LEDs and segmented displays interfaces the controller to show some relevant state informations.

CHAPTER IV

CONTROLS

In this section we will examine the control algorithms that are responsible for the actuation of the motors. This algorithm accepts the force input, absolute angular position and various other inputs, and then intelligently maps/computes the magnitude and direction of current for the respective motor.

4.1 Block Diagram

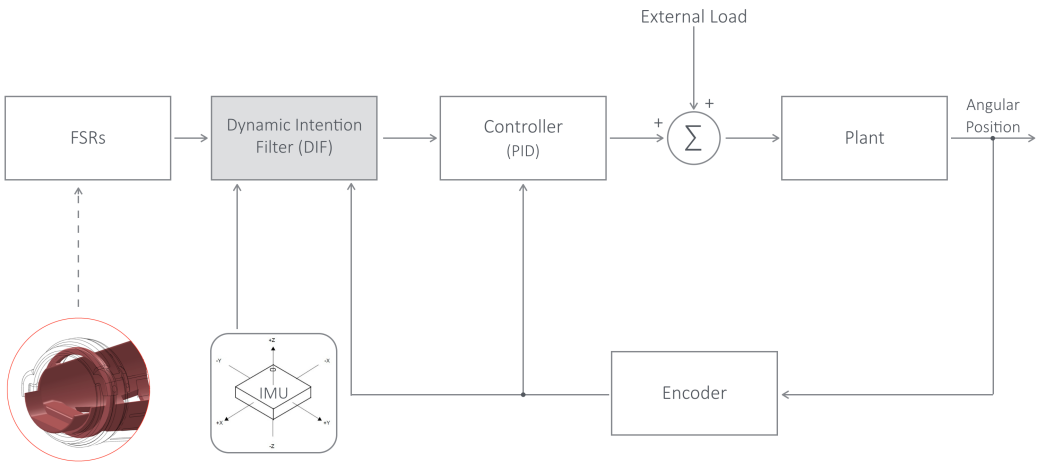


Figure 4.1: Feedback controller

Here the plant consists of three DC motors; input for each motor is an 8 bit pwm value where, $-255 \leq pwm \leq 255$. At $pwm = -255$ the motor runs at its maximum RPM (0.056 and $2.45\ rad/s$ for pitch and yaw respectively) in one direction, similarly for $pwm = +255$ it runs (at its maximum) in another direction and ~~stalls~~ stops at $pwm = 0$. Figure 4.1 shows the block diagram of the feedback controller. As mentioned earlier, our upper limb has a finite range of motion and hence the exoskeleton is designed in a way that it (exoskeleton) operates within that limit. For now (only) the motors' torque is being controlled and this leaves the exoskeleton venerable to structural damage and (obviously) and human body as well. To overcome this, a PID feedback loop runs at the plant's core and thereby confirming that the motors only get actuated within the limit discussed earlier. Although, having a feedback loop provides an added advantage of implementing a position control, but for simplicity and due to lack of testing, only torque control is implemented.

All the force sensors' values are fed into the Dynamic Intention Filter (DIF), along with the exoskeleton's angular position and feedback from inertial sensors (not taken into consideration, for now) and then fed through a weight matrix which eventually controls the torque of the motors.

4.2 Dynamic Intention Filter (DIF)

The way force sensors are mapped to the torque is not as simple as one-to-one mapping, rather, to make an exoskeleton intelligent and behave the way a human does, these mapping becomes dynamic and therefore, theoretically each force sensor should have the control over the motors. The term dynamic refers to the fact that the their (force sensor's) control must change in a fuzzy manner. Again, for simplicity, this dynamic updating feature is not fully implemented, however the control structure is designed in such a fashion that the it would eventually lead to the proposed goal. Consider the figure 4.2, all six force sensing points are clubbed in three different pairs. Broadly speaking, FS1 and FS4 maps to the pitch motion, FS2 and FS6 maps to the yaw motion and FS3 and FS5 maps to the roll motion.

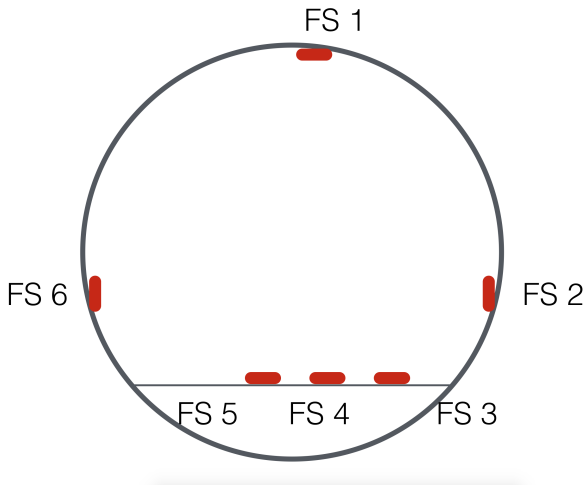


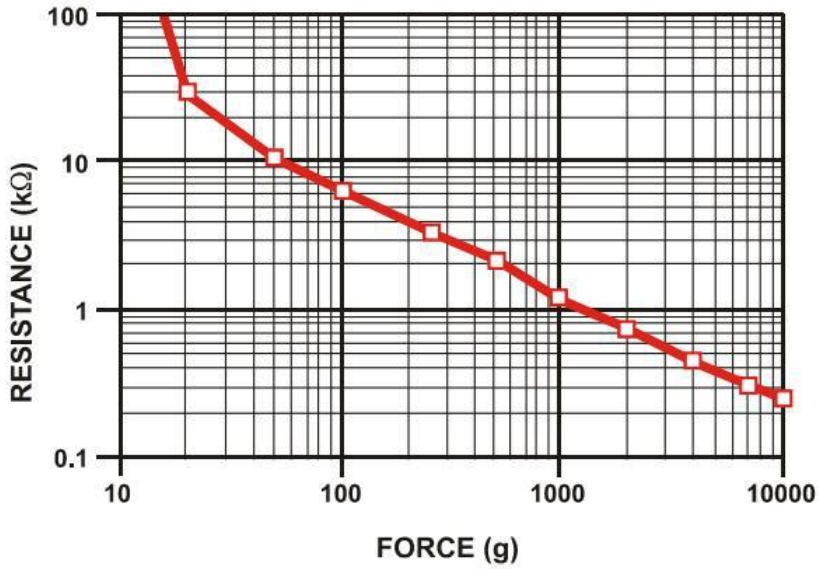
Figure 4.2: Discreet force sensing points

Now before any input is fed in-to the controller, a pre-filtering is done to gain the required sensitivity. In this case the sensitivity is determined empirically and relies on the way it feels natural to operate. The way the electrical connections are made, the output is given by the 10 bit ADC is varies linearly $[1023 \;\; 0]$ for load within $[0 \;\; 60N]$. For simplicity, we invert this signal, so that no load is represented by 0 and full load $(\leq 60\ N)$ is represented by 1023.

$$fs_j = 1023 - fs_j \qquad\qquad j = 1, 2, ..., 6 \tag{4.1}$$

In reality, the output might not be perfect zero, for no load. Moreover, even if there is a load we sometimes do not always want the system to respond, for instance, if pilot's arm is resting on the exoskeleton and due to the gravity the respective force sensors (fs_3, fs_3 and fs_5) might experience some load, which might be unintentional (figure 4.4). Along with this, empirically it has been found that feeding the force sensors output (to the controller) of same sensitivities does not feels natural and to deal with this, a unique sensitivity is assigned to every force sensors. Figure 4.3 shows a

relationship between the force applied on the force sensor and the resulting resistance, which is pretty much linear.



Credit: *Interlink Electronics [24]*

Figure 4.3: Force vs restance, FSR

As discussed earlier, to deal with varying sensitivities and thresholding, dead band truncation and percentage matching is applied within the DIF. First, this force sensor value ($0 \leq fs_j \leq 1023,\ j = 1, 2, ..., 6$) is subtracted with a threshold vector (ϕ) and continuously updated as

$$fs = fs - \phi \tag{4.2}$$

$$fs = \begin{bmatrix} fs_1 \\ fs_2 \\ fs_3 \\ fs_4 \\ fs_5 \\ fs_6 \end{bmatrix} - \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \\ \phi_6 \end{bmatrix}$$

As shown in the figure 4.4, as the pitch angle changes, the (reaction) force experienced by the respective force sensors, changes due to the miss-match of reaction vector (ζ) and the downward force due to gravitational pull on the arm.

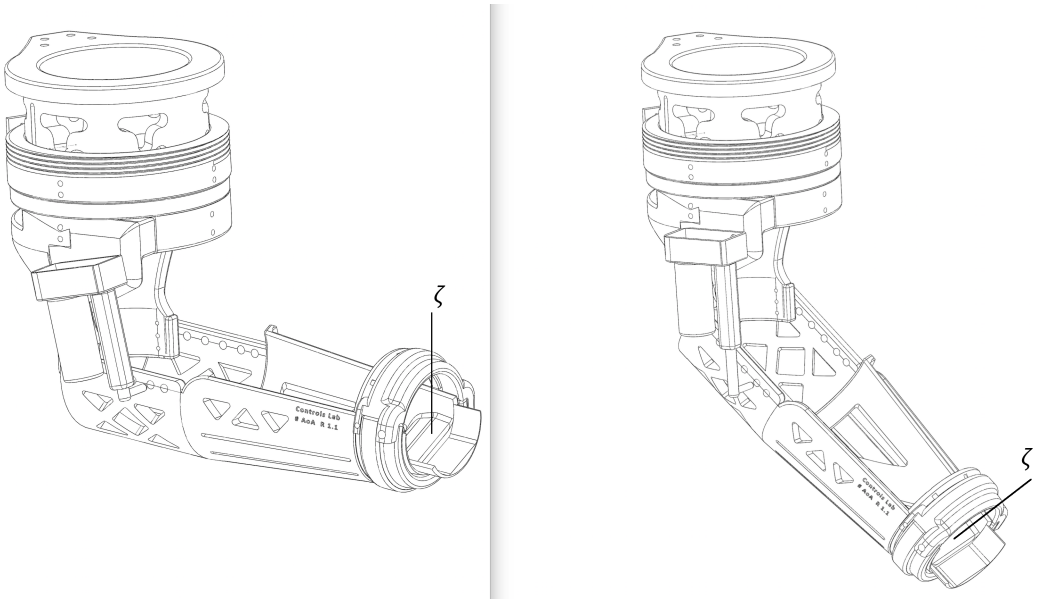


Figure 4.4: Varrying reaction force on FSR

To deal with this gravity compensation is to dynamically change the thresholds (ϕ). For simplicity, ϕ_4 is chosen for allowed to change dynamically based on θ_p. These thresholds entirely depends

on the pilot's body and how sensitive the exoskeleton is ought to be. In my case, we choose:

$$\phi = [10 \;\; 20 \;\; 30 \;\; \phi_4 \;\; 20 \;\; 20]^\top$$

where,

$$\phi_4 = \begin{cases} |\kappa_4 \cos\theta_p|, & \forall \; fs_4 \leq \kappa_4 \\ \kappa_4, & \text{otherwise} \end{cases} \tag{4.3}$$

The κ_5 in the equation 4.6 is a constant value, in this case it is set to 70. As long the product of κ_4 and $\cos\theta_p$ is less than κ_4, the threshold for fs_5 is computationally assigned, otherwise, it is set to κ_4. Alternatively, ϕ is computed as a product of spatial position weights (ψ) and the spatial position (angular positions of the exoskeleton links, $\theta = [\theta_p, \; \theta_y, \; \theta_r]$). ψ is a 6 by 3 matrix that allows the threshold (ϕ) to adjust based on θ. To do so, θ^* is computed which is,

$$\theta^* = \begin{bmatrix} \sin(\theta_P) \\ \sin(\theta_Y) \\ \sin(\theta_R) \\ \end{bmatrix}$$

and then the final threshold (ϕ) is computed as,

$$\phi = \psi \; \theta^* \tag{4.4}$$

In this case, ϕ has been manually set to, $\phi = [10 \;\; 20 \;\; 30 \;\; \phi_4 \;\; 20 \;\; 20]^\top$ and, only ϕ_4 gets updated dynamically as per the equation 4.4, where ψ is given by

$$\psi = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 70 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

ψ contains lots of zeros, because of the fact that, not all the angles are affecting the threshold (ϕ), as of now. More research is needed to study the desired behavior which might lead to a non-zero ψ. ϕ is computed as:

$$\phi = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 70 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \sin(\theta_P) \\ \sin(\theta_Y) \\ \sin(\theta_R) \end{bmatrix}$$

Moving on to the sensitivity, as mentioned earlier, selecting different sensitivities for each sensor (which may or may not not same) makes it feel more natural. To do so, first we need to determine a input lower bound (lb_{ij}), input upper bound (ub_{ij}) and corresponding output lower bound (lb_{oj})

and output upper bound (ub_{oj}) for j^{th} sensor $(j = 1, 2, ..., 6)$. Then we determine the input percent (ξ) for the given $(ub_{ij} - lb_{ij})$ range.

$$\xi_j = \frac{100 \cdot fs_j}{ub_{ij} - lb_{ij}}, \qquad j = 1, 2, ..., 6 \tag{4.5}$$

$$\xi_j = \begin{cases} \xi, & \forall\, \xi < 100 \\ 100, & \text{otherwise} \end{cases} \tag{4.6}$$

Once ξ is computed, the fs_j is updated again with its new value that is a representation of ξ % for the range $(ub_{oj} - lb_{oj})$ as shown in the equation 4.8

$$fs_j = \frac{\xi_j \cdot (ub_{oj} - lb_{oj})}{100} \tag{4.7}$$

where,

$lb_{ij} = lb_{oj} = 0, \qquad \forall\, j = 1, 2, ..., 6$

$ub_{oj} = 1023, \qquad \forall\, j = 1, 2, ..., 6$

Thus, the only variable left in this sensitivity mapping is ub_{ij}, that can be chosen empirically. For instance, if an input range for j^{th} sensor is picked to be between 0 and 200 and fs_j reads (say) 100, then from equation 4.5, $\xi_j = 50\%$. Now using equation 4.8, the updated fs_j for the full range $(\text{say from, } 0\ to\ 1000)$ is computed, which is,

$$fs_j = \frac{50 \cdot (1000 - 0)}{100} = 500 \tag{4.8}$$

The above example shows, how a value from once range is mapped to some other range, which essentially alters the sensitivity. One advantage of this approach is that it provides the flexibility of choosing non-zero bounds, which otherwise is very difficult in simple multiplication. In the Arduino IDE, the core library has an inbuilt function [25] that eases the above mapping process. This can be simply evaluated by,

$$fs_j = map(fs_j,\ lb_{ij},\ ub_{ij},\ lb_{oj},\ ob_{oj})$$

$$fs_j = fs_j : (\ lb_{ij},\ ub_{ij}) \longmapsto (\ lb_{oj},\ ub_{oj})$$

At this point, all the fs_j are updated with their respective intentions. The goal is to compute the pwm_X values that can drive the motors to produce proportionate torque. Three different pwm values are computed for every motor, $(-255 \leq pwm_X \leq +255,\ X = P, Y, R)$

$$pwm = w\ fs \tag{4.9}$$

$$\begin{bmatrix} pwm_P \\ pwm_Y \\ pwm_R \end{bmatrix} = \begin{bmatrix} w_{11} & 0 & 0 & -w_{14} & 0 & 0 \\ 0 & w_{22} & w_{23} & 0 & -w_{25} & -w_{26} \\ 0 & 0 & w_{33} & 0 & -w_{35} & 0 \end{bmatrix} \begin{bmatrix} fs_1 \\ fs_2 \\ fs_3 \\ fs_4 \\ fs_5 \\ fs_6 \end{bmatrix}$$

Where,

w_{ij} : weights, that defines the mapping and sensitivity of the device.

fs_i : digitized force value by i^{th} sensor; $0 \leq fs_j \leq 1023,\ j = 1, 2, ..., 6$

pwm_X : pwm values for the motors; $-255 \leq pwm_X \leq +255,\ X = P, Y, R$

$$w_{23} = \begin{cases} 0.65, & \theta_P \leq 21.00^\circ \\ 0, & \text{otherwise} \end{cases} \tag{4.10}$$

$$w_{25} = \begin{cases} 0.65, & \theta_P \leq 21.00^\circ \\ 0, & \text{otherwise} \end{cases} \tag{4.11}$$

$$w = \begin{bmatrix} 0.88 & 0 & 0 & -0.80 & 0 & 0 \\ 0 & 0.88 & w_{23} & 0 & -w_{25} & -0.80 \\ 0 & 0 & 0.45 & 0 & -0.45 & 0 \end{bmatrix}$$

As we can see that, w has so many zeros, that essentially means that only selected force sensors are allowed to contribute to a particular motion (Roll, Pitch or Yaw), the remaining force sensors are forced to zero (or contribute nothing). A differential of these sensors are computed based on the

(tuned) weights, which, in turn, is a function of the gravity compensation of the arm, sensitivities and et-cetera. This differential helps in normalizing the force values, which could be an (undesirable) outcome of the way sensors are worn. The final pwm values are parsed through the feedback loop, which makes sure that the exoskeleton operates safely within range-of-motion limits and in case of $|pwm| > 255$, the value is then forced to 255 with proper signs.

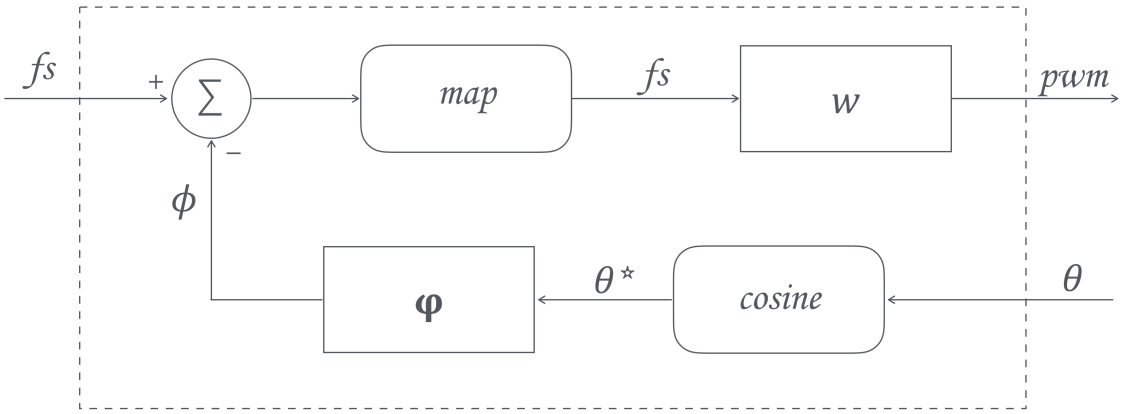


Figure 4.5: DIF block diagram

The components of DIF are shown in figure 4.5; it (DIF) is a black box that accepts motors' position (θ) and the force inputs (fs). Based on the algorithm, discussed above, a final pwm value for each motors are generated. This pwm is again passed through a feedback controller, wherein software limits are embedded which allows the exoskeleton to operate within the safe limit.

CHAPTER V

RESULTS AND DISCUSSIONS

5.1 Motion Analysis

After designing the simple torque control algorithm, the force sensor values along with the angular pitch and yaw positions are sent over serial to the Matlab for the sake of data visualization. Figure 5.1 shows the pitch motion for a mock inputs from fs_1 (dash-dot blue), fs_4 (dotted red) and the absolute pitch angle (solid black) θ_p.

The zero for pitch is set at 21.00° and as the arm is moved up, about elbow, the angle increases to a maximum of 81.00°; this is the limit within which the arm can move (not to mention that the hardware is capable of going beyond this software limit up-to the limit which is safe for a normal human). Focusing on the figure 5.1, θ_p starts at 21.00° and remains steady for a while, for $fs_1 = fs_4 = 0$. A little before 500^{th} sample fs_1 experiences some force ($fs_1 > 0$). It is easy con infer that higher the magnitude of fs_1, higher is the rate of change of θ_p. It continues to behave like this, till θ_p attains its (predefined) maximum of 81.00° (around 2000^{th} sample) and at this point, no matter what fs_1 has to offer, θ_p can not increase. A little before 3000^{th} sample, force is applied on fs_2 (shown in dotted red), and as a result, θ_p starts decreasing proportionately with fs_2's magnitude. The way differential algorithm works, is depicted by the samples over 3500; here the controller receives the value from fs_1 and fs_2 as well (can be possible, depends on how the sensor is placed and also varies from person to person). A differential is computed out of fs_1 and

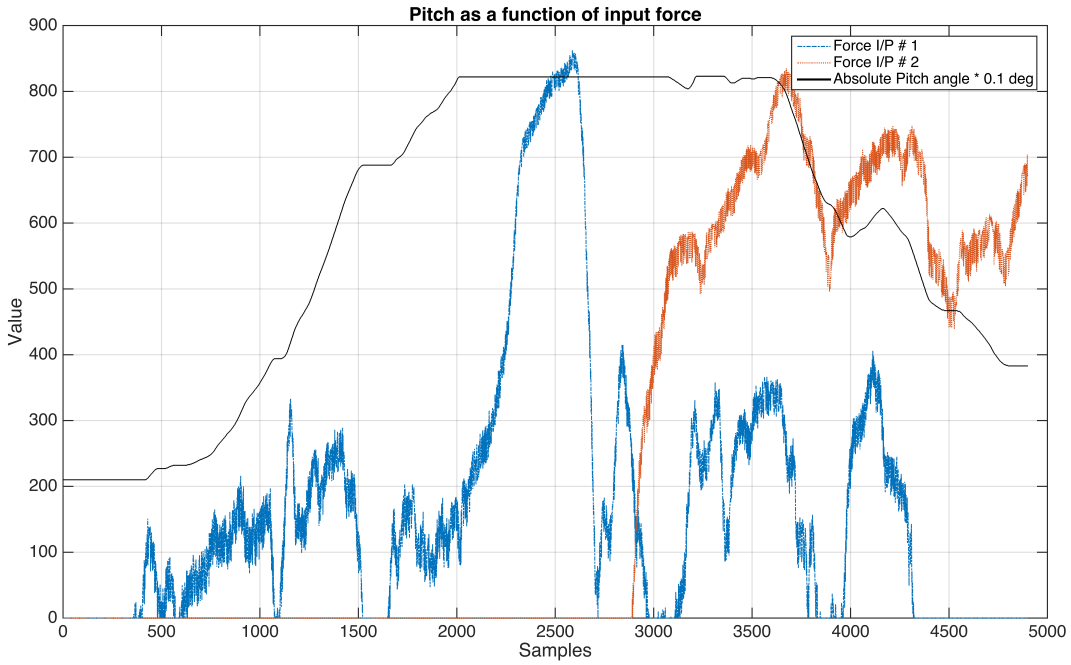


Figure 5.1: Pitch motion

fs_2 and as turns out in this case, it is forced to be biased towards fs_1, that is, the value from fs_1 gets prioritized over fs_2, as shown around 4000^{th} sample; although the $fs_2 > fs_1$, the effect of fs_1 wins and θ_p starts increasing (for a while) and as fs_1 decreases significantly, the effect of fs_2 starts to dominate and consequently θ_p starts to decrease proportionately with fs_2.

Figure 5.2 shows the filtered output out of the fs_1 and fs_4. As we can see that the input to the DIF is from 0 to 1023, and after applying the spatial threshold and the weights, the pwm_P is computed, where $(-255 \leq pwm_P \leq +255)$. This is then passed through the weights and sent to the motor driver that controls the current to the motor. Now here is an assumption made, because of no feedback for position control, only torque is the controlled, this means that if there is any external load on the exoskeleton, the rate of change of the θ_p might not be consistent. A similar study [26] published in 2010 used a partially open loop to control the torque of the motors and obtained some

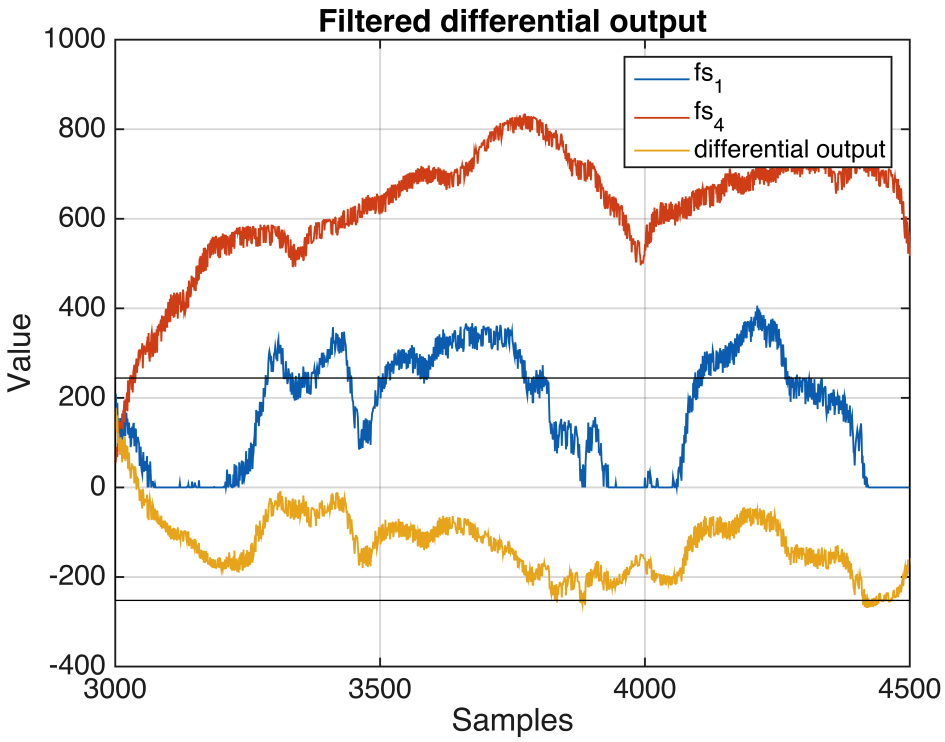


Figure 5.2: Differential filtered output, pitch

better results. Even without this the tracking seems to work against the disturbance rejection, as shown in figure 5.3. As long as the magnitude of differential input in greater than zero, then θ_p also (in this case) increases.

Pretty much the same holds true in the case of yaw motion (θ_y) (solid black), which is a function of fs_2 (dash-dot blue) and fs_6 (red blue) as shown in figure 5.4. In this case only, fs_2 and fs_6, pair drives θ_y, for all $44.00° \leq \theta_y \leq 121.00°$. As fs_6 experiences force, the controllers commands the motor (the one responsible for yaw) consequently a proportional torque is applied on the arm till $fs_6 \neq 0$ (to be precise, $|w_{22} \cdot fs_2 - w_{26} \cdot fs_6| \neq 0$). It is clear that, at around 1200^{th} sample, $fs_6 = 0$ and during this, the rate of change of $\theta_y = 0$, in other words, the arm stays wherever it was left, as long as $fs_6 = fs_2 = 0$. At around 2000^{th} sample, it is clear that (θ_y) has already saturated

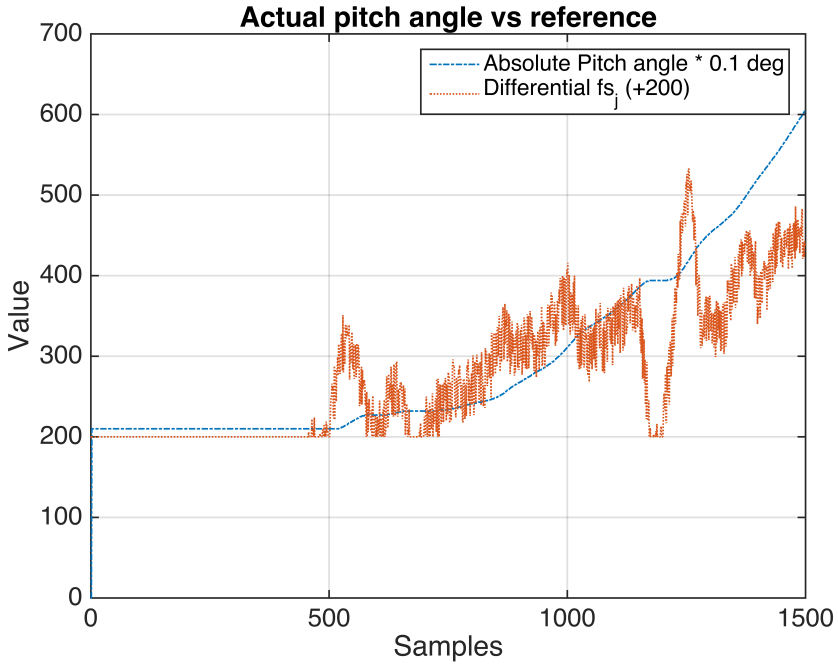


Figure 5.3: Actual pitch angle and the reference

to its maximum allowable value and hence does not responds to (in general) fs_6. As fs_2 increases, as expected (θ_y) starts to decrease till it reaches its minimum of $44.00°$. After 4000^{th} sample, the effect of differential input can be seen clearly; here both (fs_2 and fs_6) are non zero and therefore based on their respective weights, one of them supersedes and drives the torque in the respective direction.

Analyzing the differential filtered output from fs_2 and fs_6 from figure 5.4, a closed look is shown in the figure 5.5. We can see that, between samples 3000 and little before 4000, $fs_6 = 0$ and thus the output is approximately the difference of the fs_2 and fs_6. At around 4000^{th} samples, as the magnitude of fs_6 becomes greater than that of fs_2, as expected, the differential output decreases below zero. This translate that the intention is on the other direction and thus the yaw motor's torque direction reverses, thereby reversing the direction of the yaw motion. Once the force inputs and the

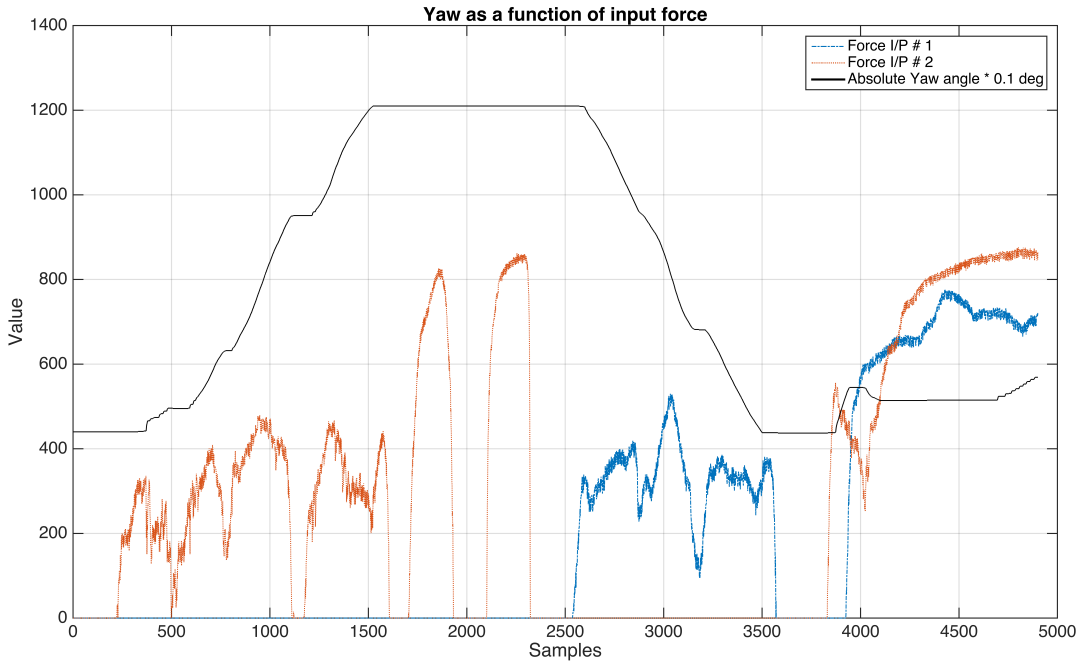


Figure 5.4: Yaw motion

spatial positions enters the DIF, the output pwm for yaw is determined (shown in yellow). This pwm is parsed through the PID controller which then takes over the torque control on the motor responsible for yaw. A gain, a generic yaw motion is shown in the figure 5.4.

In case of roll, the linear actuator/motor has an inbuilt mechanism that cuts off the torque at the extreme positions and thereby prevents any permanent damage to the motor. With this safety feature, the values from fs_3 and fs_5 are mapped directly to the motor's torque.

5.2 Conclusion

The system seems fairly responsive and reacts pretty much the way it is supposed to. One of the goal we had during this design was to test various ways to transmit power from one place to another and hence we tried two different types of cable based mechanism and one link based.

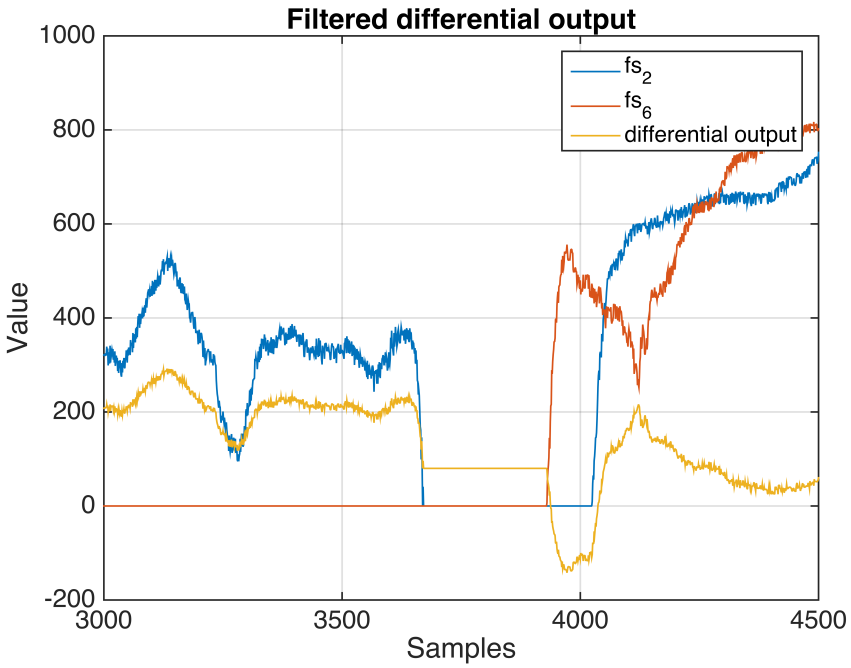


Figure 5.5: Differential filtered output, yaw

The cable mechanism should offer offer a near perfect linear force transmission, while in this case it so happened that the usage of springs added non-linearities to a significantly. The other two mechanisms seems to respond relatively well. One of the root cause cause for these issues is that fact that plastic is used to fabricate these parts and poor quality of printing helped to increase the already existing relatively high friction. Other than that the force sensors are able to read the intention, with a room still available for better tuning. Overall this has been a great experience and we will continue this research for further improvements.

5.3 Future Work

As a continuation of this research, the goal is to implement position control, rather than a simple torque control. Position control has several advantages, as usually it adapts to the varying load and as per the study published in International Conference on Intelligent Robots and Systems [27] it

is even considered safe. As mentioned in earlier sections, where DIF is discussed, most of the elements of the weight matrix are zero, while in reality, these may not be. A detailed study will be done on how a normal human being moves their limbs in free space and thus a mathematical analysis can be done, a similar study [28] published in the International Conference on Information and Automation for Sustainability provides some useful guidelines to conduct this type of study. At the University of Dayton, the Department of Physiotherapy has a state of art Vantage Motion Capture Camera (VICON) that will be helpful in studying the human biomechanics. As far as next design is considered, the approach will be to use metal (or equivalent) to fabricate the design that preferably will be naked, to minimize the weight and volume so that higher range of motions can be achieved. This approach might allow to install the motors right at the links and have the successive links coupled to a harmonic reducers, similar to a design concept published by Dongyong Jia [29]..

5.4 Limitations

Being a prototype, this design has many limitations, however it does provides a proof of concept which allows us to understand the world of exoskeletons even better. ABS plastic is being used to fabricate the prototype by 3-D printing the parts out of it, now, 3-D printing itself is not a best way to fabricate, especially when strength matters. Due to uneven cycles involved in laying the successive layers, the strength might get compromised and it did happen in this case. To minimize the material usage, sparse technique is used to print, while a solid design is used in computer aided analysis, which obviously does not provide actual results, but we still stick with the computer generated results.

The main controller that hosts the control algorithm is a microprocessor that is clocked at just $8\ MHz$, which is relatively slow for this very application, as the dynamic computation becomes more complex, the microcontroller might just fail to process everything in real time, thereby rendering it useless; in spite of this, for now the controller seems to perform just well.

BIBLIOGRAPHY

[1] "Movdata." [Online]. Available: http://www.movdata.net/human-skeleton-with-muscles.html

[2] H. Kazerooni, "Exoskeletons for human power augmentation," *Intelligent Robots and Systems, 2005. (IROS 2005).*, pp. 3459 – 3464, 2005.

[3] A. Salarian, "iTug, a Sensitive and reliable measure of mobility," *IEEE Transaction on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 3, pp. 303 – 310, 2010.

[4] L. Weiguang Huo Huang Jian Wang Yongji Jun Wu Cheng, "Control of upper-limb power-assist exoskeleton based on motion intention recognition," *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 2243 – 2248, 2011.

[5] Y. Jian Huang Weiguang Huo Wenxia Xu Mohammed S. Amirat, "Control of upper-limb power-assist exoskeleton using a human-robot interface based on motion intention recognition," *Automation Science and Engineering, IEEE Transactions on*, vol. 12, no. 4, pp. 1257 – 1270, 2015.

[6] N. C. M. Lenzi, T De Rossi S.M.M. Vitiello, "Intention-based emg control for powered exoskeletons," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 8, pp. 2180 – 2190, 2012.

[7] "Typical exoskeleton," April 2008. [Online]. Available: http://www.popsci.com/scitech/article/2008-04/building-real-iron-man

[8] B. L. L. L. de Camargo Pinto Roberto Giugliani James Edmond Wraith Nathalie Guffon Elke Eich and M. Beck, "Orthopedic manifestations in patients with mucopolysaccharidosis type ii (hunter syndrome) enrolled in the hunter outcome survey," *Orthop Rev (Pavia)*, vol. 2, no. 16, 2010.

[9] "Mit tech review." [Online]. Available: http://www.technologyreview.com/view/527336/do-we-need-asimovs-laws/

[10] "Harmonic drive." [Online]. Available: http://harmonicdrive.net/products/actuators/sha-cg/

[11] X. W. H. S. J. Jiang, "The fatigue strength analysis of the pump shaft of rcv charging pump," *International Symposium on Fluid Machinery and Fluid Engineering*, vol. 6, no. 11, pp. 1 – 6, 2014.

[12] “Open music labs.” [Online]. Available: http://www.openmusiclabs.com/learning/sensors/fsr/

[13] “Metronics.” [Online]. Available: http://www.metronics.net/pr/2013/02-2013_New-Robust-Absolute-Rotary-Encoders.php

[14] “Neuroscience and robotics lab.” [Online]. Available: http://hades.mech.northwestern.edu/index.php/Rotary_Encoder

[15] “Wayne and layne LLC.” [Online]. Available: https://www.wayneandlayne.com/bricktronics/design-and-theory/

[16] “LSI/CSI datasheet.” [Online]. Available: http://www.lsicsi.com/pdfs/Data_Sheets/LS7366R.pdf

[17] “Analog devices.” [Online]. Available: http://www.eetimes.com/document.asp?doc_id=1305107

[18] “Pololu electronics.” [Online]. Available: https://www.pololu.com/product/2303/specs

[19] “Banebots electronics.” [Online]. Available: http://www.robotshop.com/en/banebots-first-cim-motor.html

[20] “Next electronics, motor driver schematics.” [Online]. Available: http://www.next.gr/circuits/H-Bridge-Motor-Driver-using-MOSFETs-and-Transistors-l28177.html

[21] Jordandee, “Learn pwm sparkfun.” [Online]. Available: https://learn.sparkfun.com/tutorials/pulse-width-modulation

[22] “Arduino mega pro mini 3.3 sparkfun electronics.” [Online]. Available: https://www.sparkfun.com/products/retired/10743

[23] “SPI communication, google patents (us 20120072628 a1).” [Online]. Available: http://www.google.com/patents/US20120072628

[24] “Interlink electronics.” [Online]. Available: http://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/2010-10-26-DataSheet-FSR400-Layout2.pdf

[25] “Inbuilt mapping command in arduino.” [Online]. Available: https://www.arduino.cc/en/Reference/Map

[26] X. Chen, “Design and experiment of an open control system for a humanoid robot,” *International Conference on Automation and Logistics (ICAL)*, pp. 367 – 372, 2010.

[27] W. A, “Force control strategy for a hand exoskeleton based on sliding mode position control,” *International Conference on Humanoid Robots*, pp. 4615 – 4620, 2006.

[28] K. MS, “An adaptive complementary filter for inertial sensor based data fusion to track upper body motion,” *International Conference on Information and Automation for Sustainability*, pp. 1 – 5, 2014.

[29] D. Jia, “Mechanical design of a light weight and high stiffness arm for humanoids,” *International Conference on Humanoid Robots*, pp. 337 – 342, 2009.

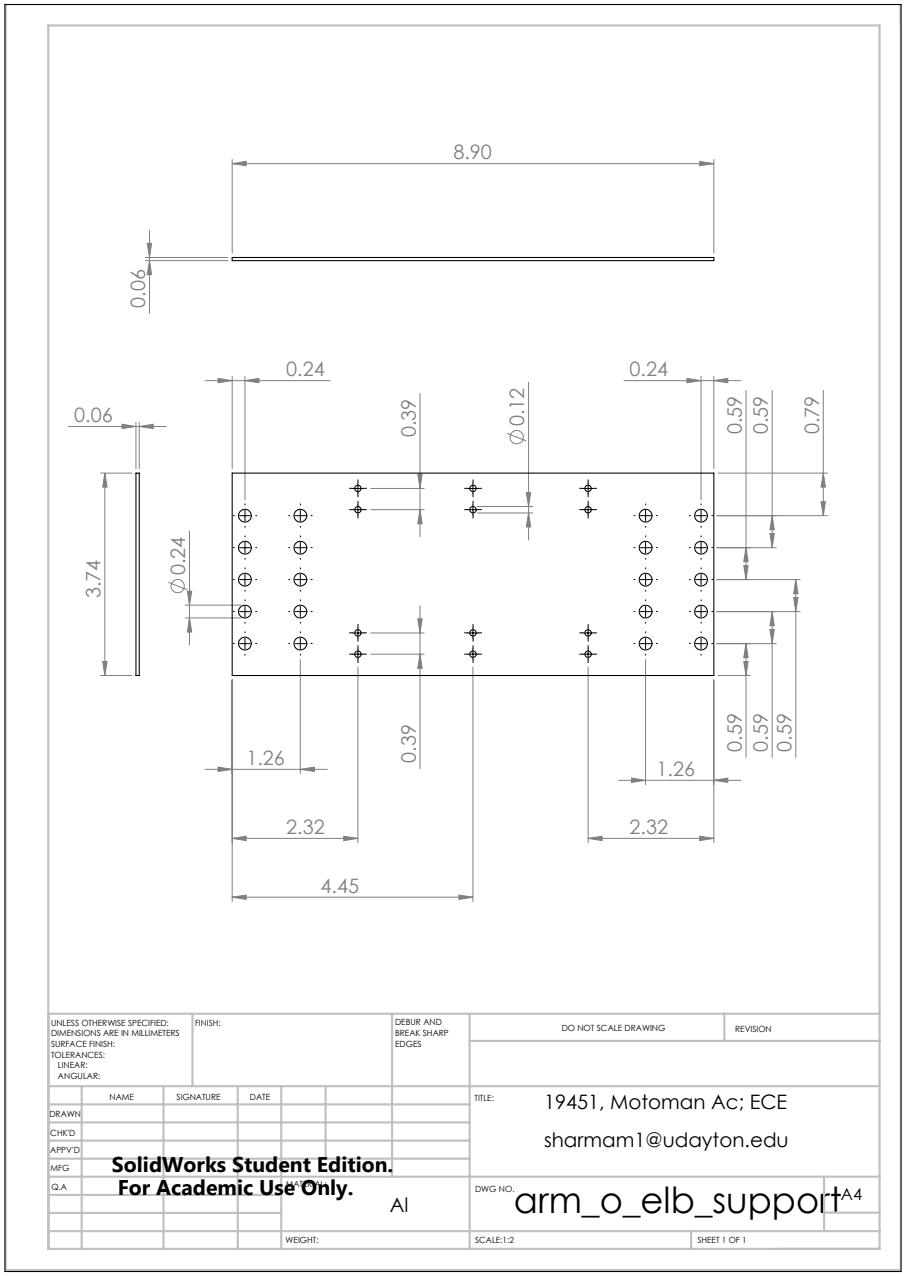
APPENDIX

Arduino script

```
#include <Adafruit_GFX.h>    // Core graphics library
#include <SPI.h>       // this is needed for display
#include <Adafruit_ILI9341.h>
#include <Wire.h>      // this is needed for FT6206
#include <Adafruit_FT6206.h>

// The FT6206 uses hardware I2C (SCL/SDA)
//Adafruit_FT6206 ctp = Adafruit_FT6206();

// The display also uses hardware SPI, plus #9 & #10
#define TFT_CS 48
#define TFT_DC 46
//Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC);

int capx = 0;
int capy = 0;

const int mainPowerPin = 38;
const int secondaryArmPin = 29;  // To be changed, foot
int mainSupplyState = 0;
int secondaryArmState = 0;

const int motorAdirPin = 25;
const int motorApwmPin = 2;
const int motorBdirPin = 23;
const int motorBpwmPin = 3;
const int motorCdirPin = 4;
const int motorCpwmPin = 5;
const int motorDdirPin = 6;
const int motorDpwmPin = 7;

const int joyStickXpin = A4;
const int joyStickYpin = A5;
const int fsrPin1 = A13;
const int fsrPin2 = A12;
const int fsrPin3 = A11;
const int fsrPin4 = A10;
const int fsrPin5 = A15;
const int fsrPin6 = A14;


int joyStickX = 0;
int joyStickY = 0;
int fsr1 = 0;
int fsr2 = 0;
int fsr3 = 0;
```

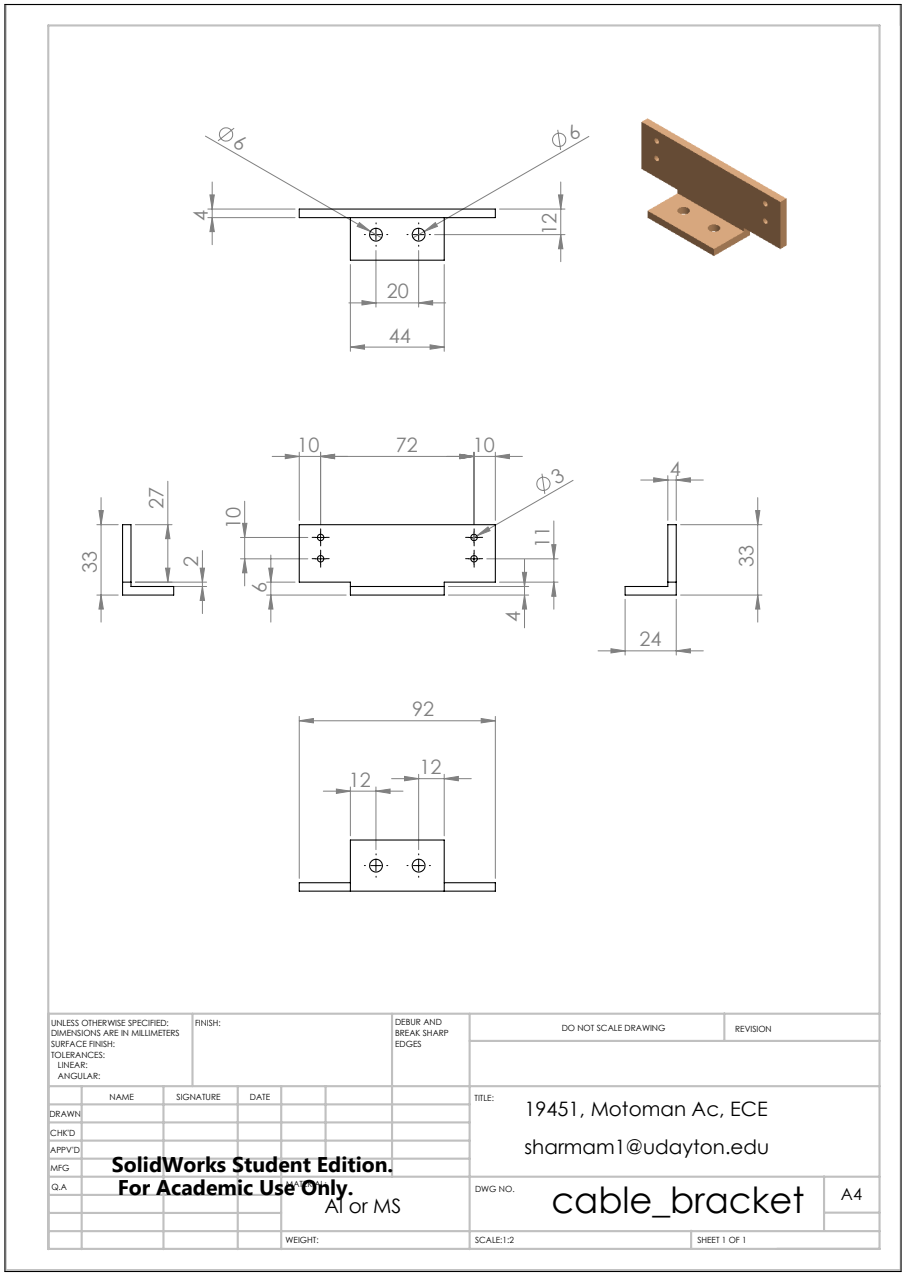
Matlab Scripts

```
% Extract the variables

clear all; close all;  clc;
global s;
%s = serial('/dev/tty.usbserial-AJ038PJU', 'BaudRate', 9600);
s = serial('/dev/tty.usbserial-A6026N7L', 'BaudRate', 57600);
fopen(s);

timeout_flag=0;
x = 0;
y = 0;
z = 0;

for i = 1:8000

    start_timer=tic;
    %%Data timeout
    out = fscanf(s);%fread(s,s.BytesAvailable);
    C = textscan(out, '%d %d %d', 'delimiter',',','EndOfLine','\n');%,'commentStyle','//');
    [xx,yy,zz] = deal(C{:});

    x = [x xx];
    y = [y yy];
    z = [z zz];
    disp(zz)
    end_timer=toc(start_timer);%End timer
    %plot_timer=plot_timer+end_timer;

%     altarray=[altarray;z];
%     presarray=[presarray;y];
%     temparray=[temparray;x];
    %timearray=[timearray;plot_timer];
    %figure(1)
    %plot(x)
    %drawnow;


end

fclose(s)
```

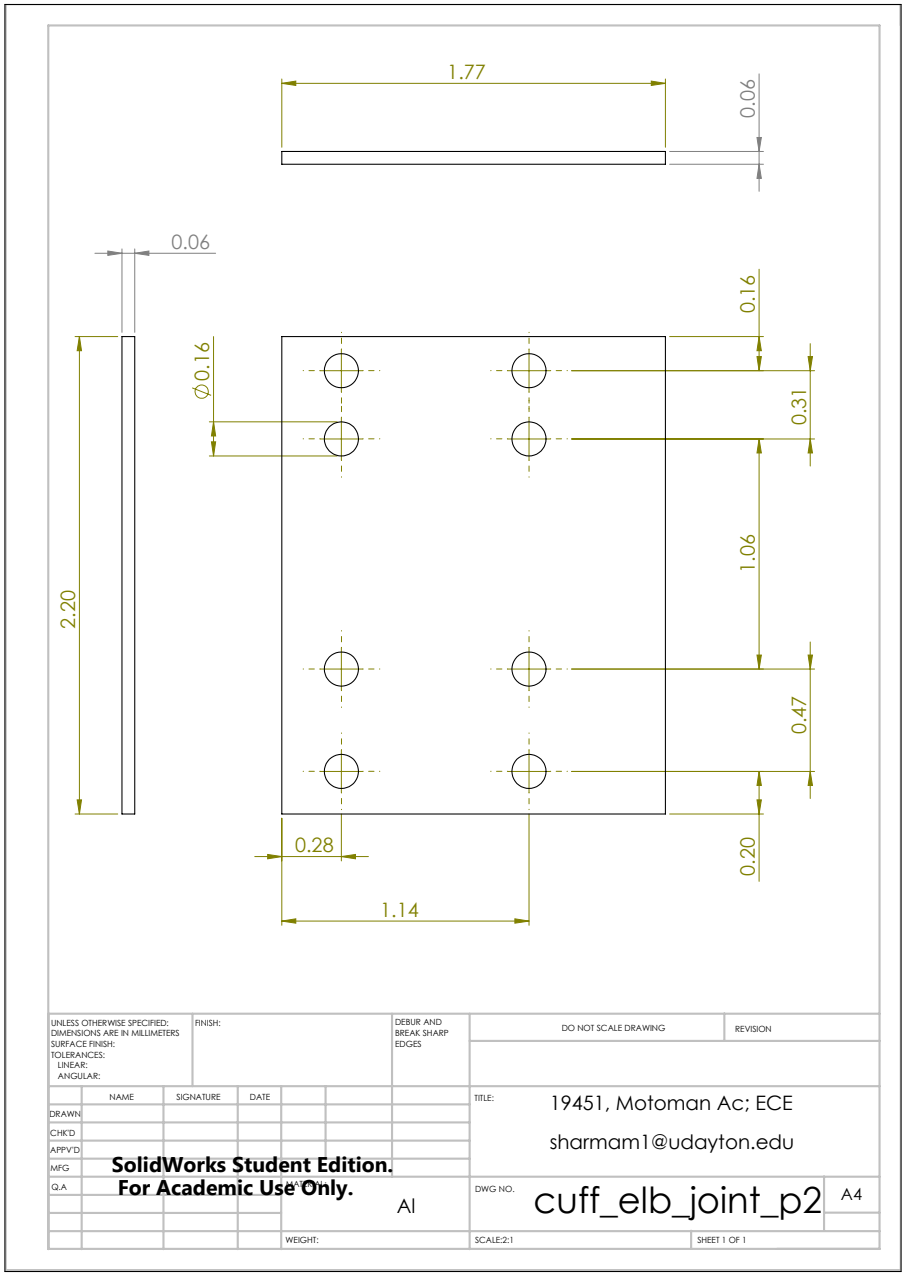


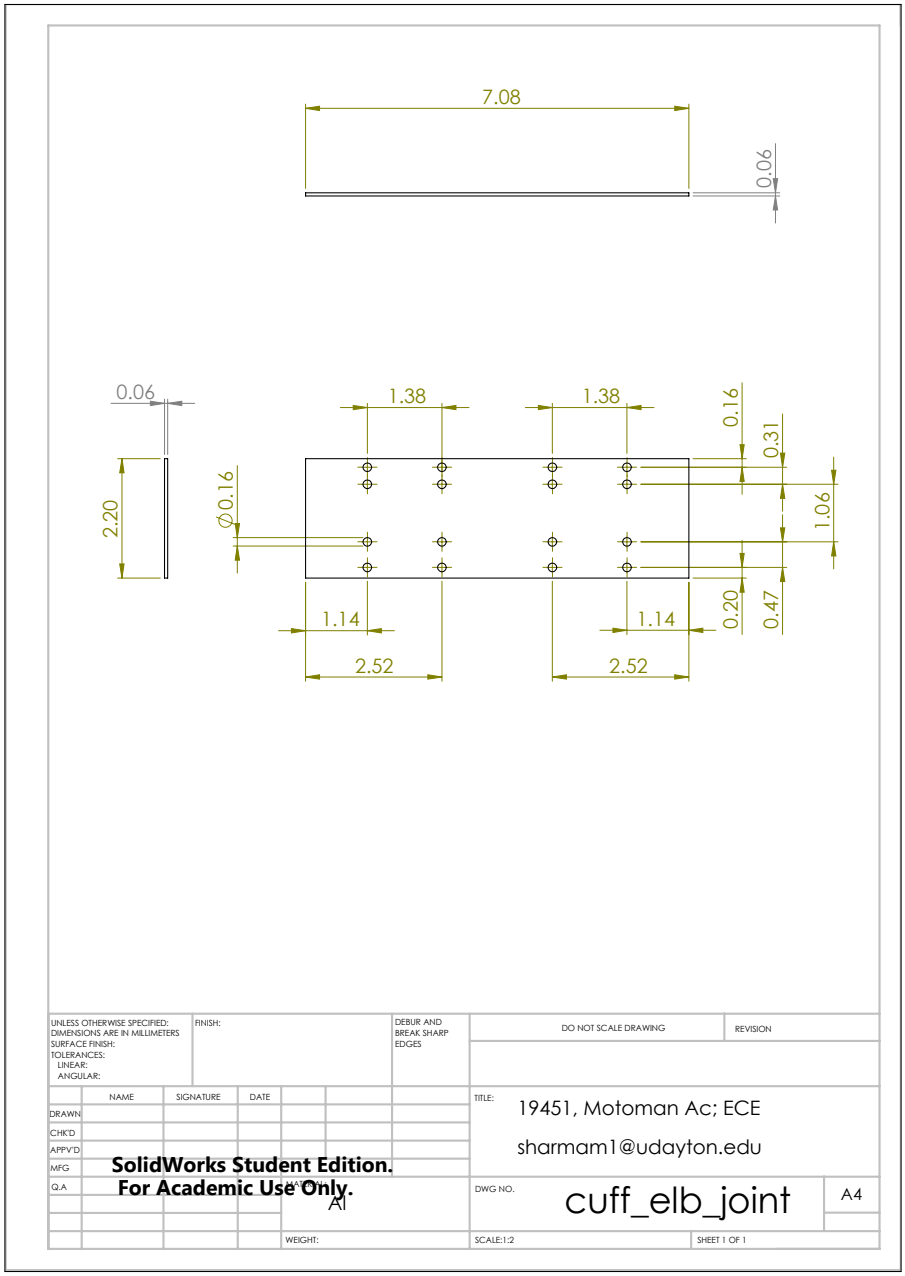
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:			FINISH:			DEBUR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE				TITLE:	19451, Motoman Ac; ECE
DRAWN								sharmam1@udayton.edu
CHK'D								
APPV'D								
MFG								
Q.A							DWG NO.	arm_o_elb_support
				MATERIAL				A4
				WEIGHT:			SCALE:1:2	SHEET 1 OF 1

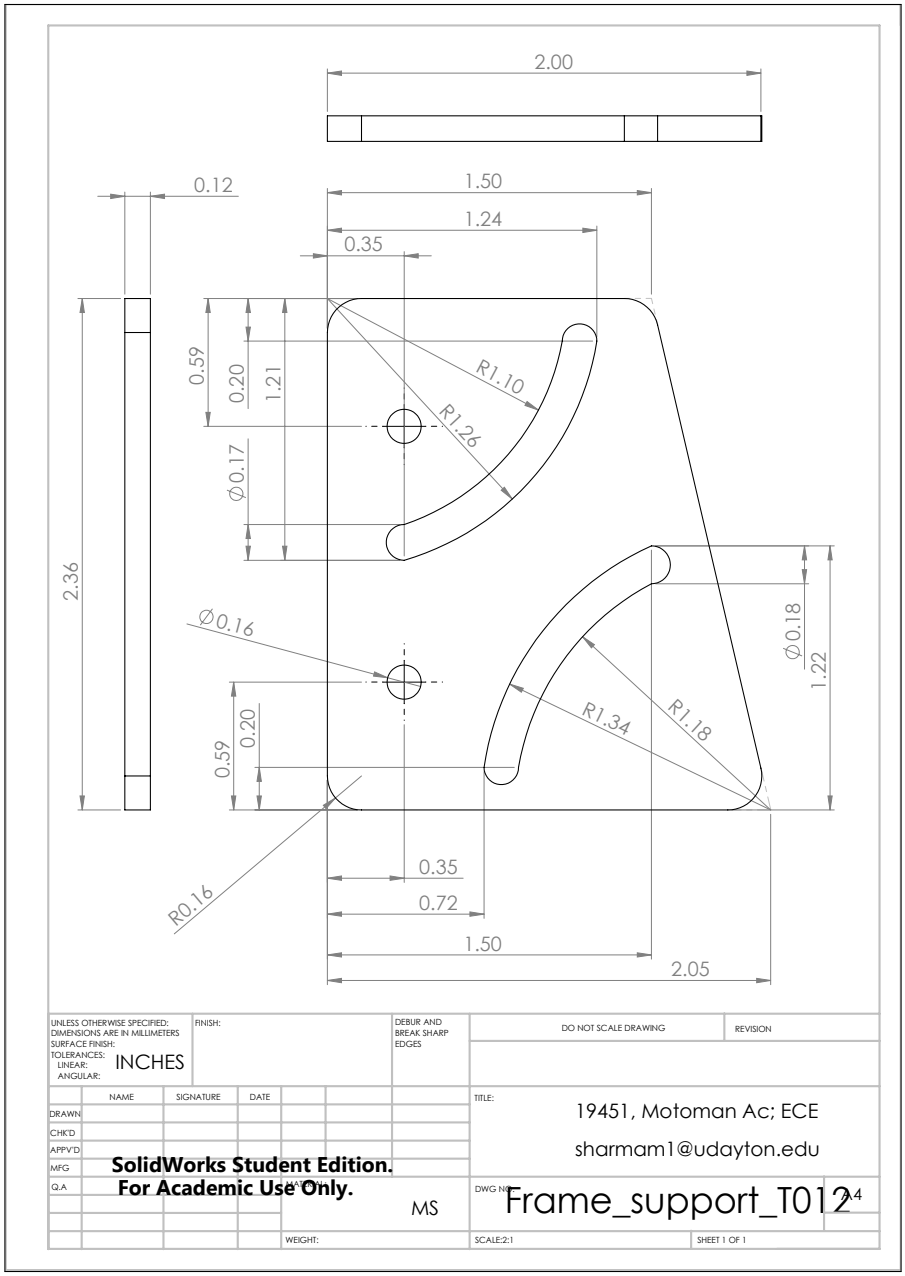
Al

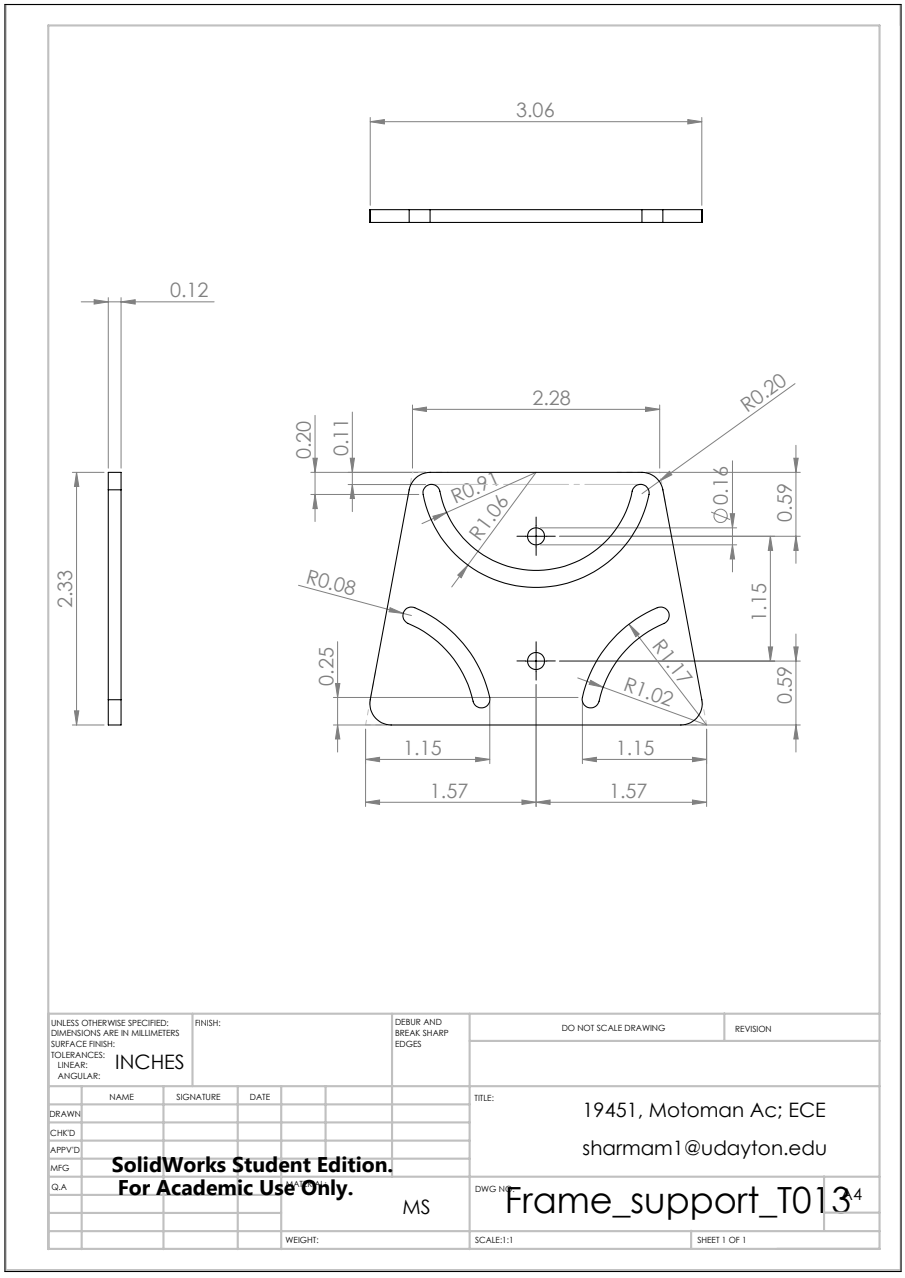


UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:		DEBUR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE	TITLE:	19451, Motoman Ac, ECE sharmam1@udayton.edu
DRAWN					
CHK'D					
APPV'D					
MFG					
Q.A				DWG NO.	cable_bracket
			MATERIAL: Al or MS		A4
			WEIGHT:	SCALE:1:2	SHEET 1 OF 1

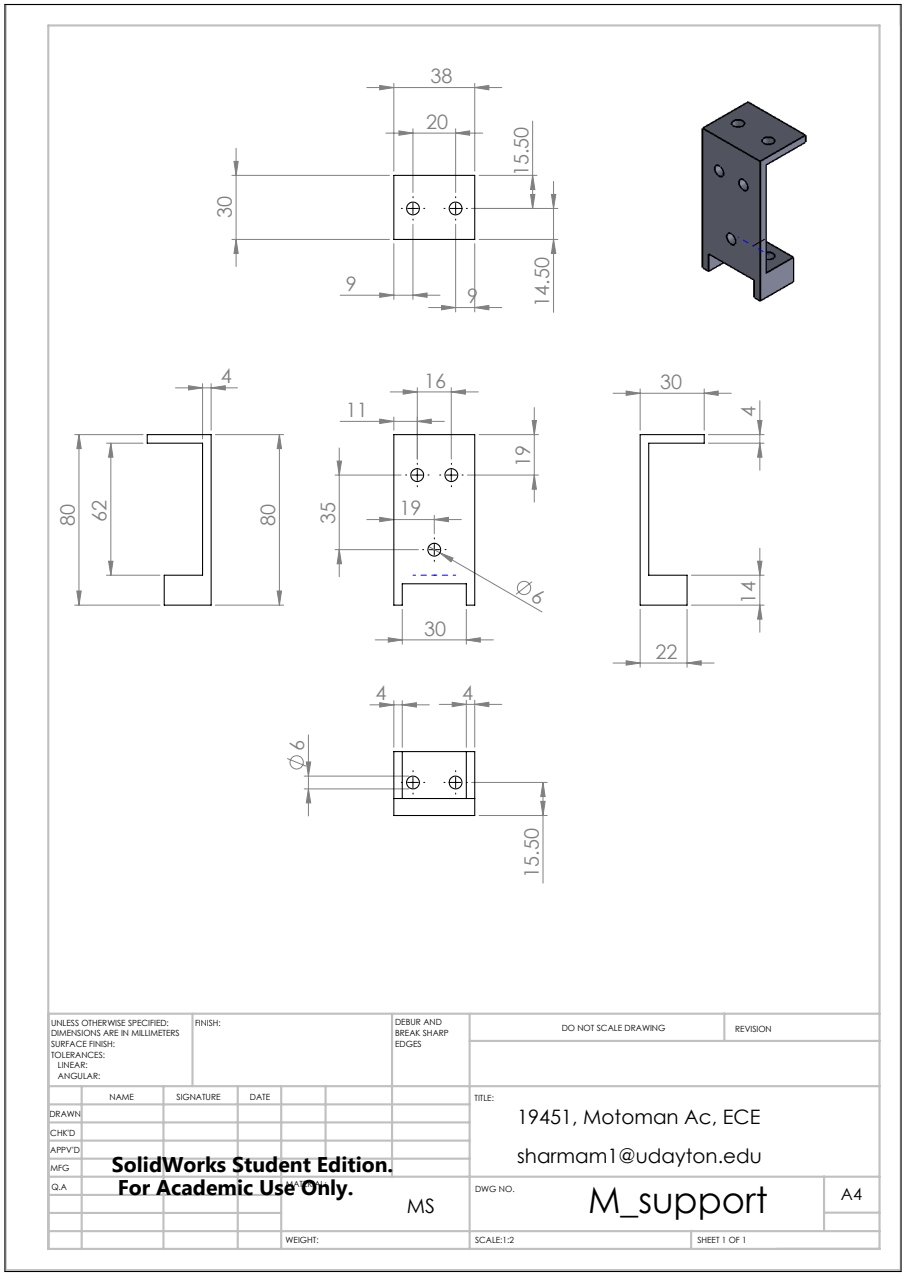

1.77
0.06
0.06
0.16
Ø0.16
0.31
2.20
1.06
0.47
0.20
0.28
1.14
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:
FINISH:
DEBUR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION
NAME
SIGNATURE
DATE
DRAWN
CHK'D
APPV'D
MFG
Q.A
TITLE:
19451, Motoman Ac; ECE
sharmam1@udayton.edu
SolidWorks Student Edition.
For Academic Use Only.
MATERIAL
Al
DWG NO.
cuff_elb_joint_p2
A4
WEIGHT:
SCALE:2:1
SHEET 1 OF 1

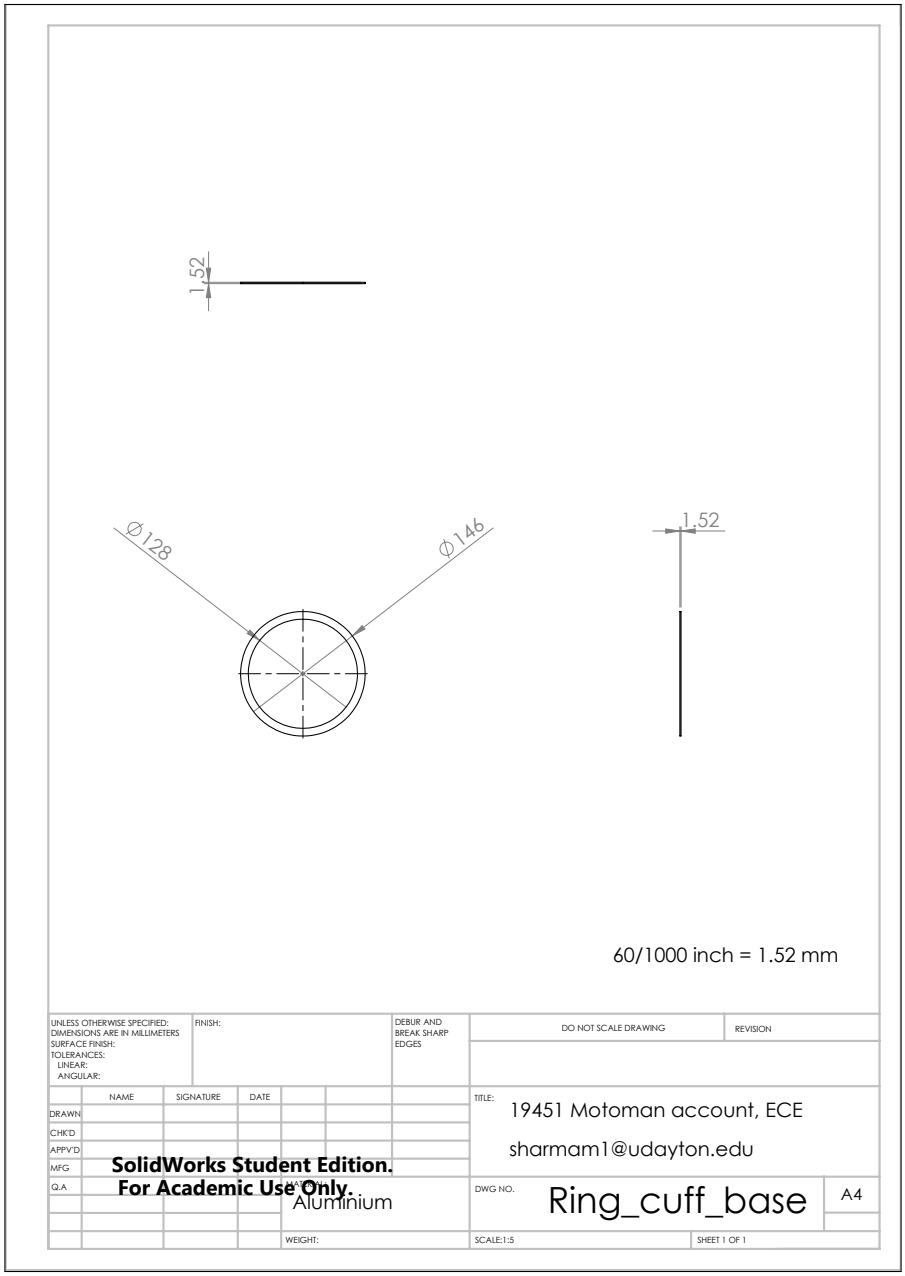

7.08
0.06
0.06
1.38
1.38
0.16
0.31
1.06
2.20
Ø0.16
1.14
1.14
0.20
0.47
2.52
2.52
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:
FINISH:
DEBUR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION
NAME
SIGNATURE
DATE
DRAWN
CHK'D
APPV'D
MFG
Q.A
MATERIAL
Al
WEIGHT:
TITLE:
19451, Motoman Ac; ECE
sharmam1@udayton.edu
SolidWorks Student Edition.
For Academic Use Only.
DWG NO.
cuff_elb_joint
A4
SCALE:1:2
SHEET 1 OF 1

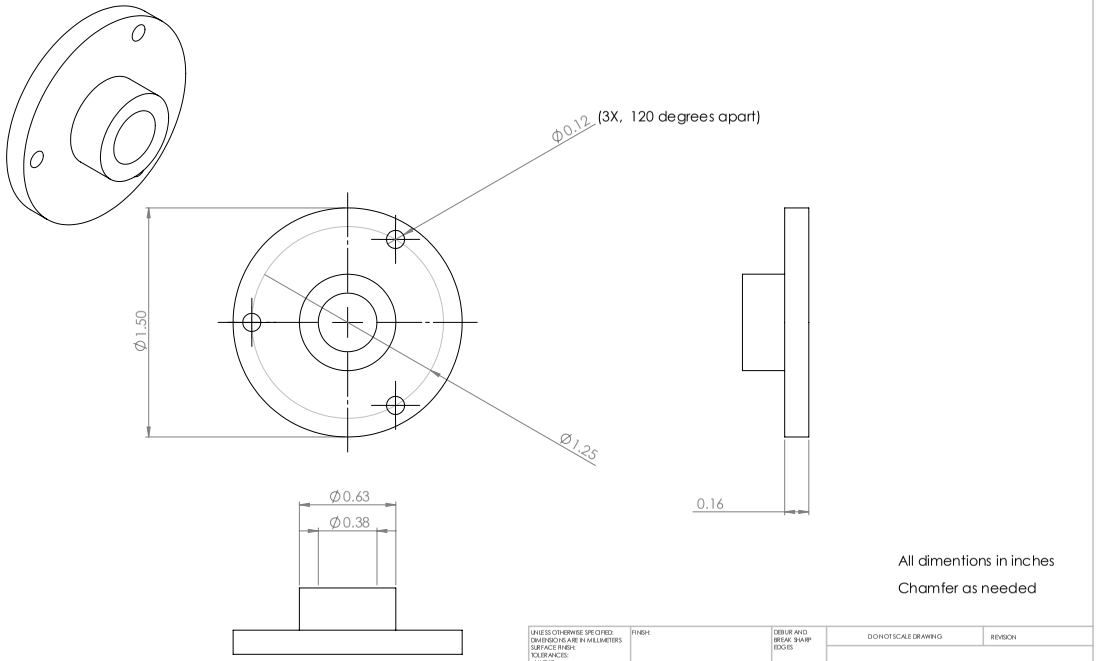

2.00
0.12
1.50
1.24
0.35
0.59
0.20
1.21
R1.10
R1.26
Ø0.17
2.36
Ø0.16
Ø0.18
1.22
R1.34
R1.18
0.59
0.20
R0.16
0.35
0.72
1.50
2.05
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:
INCHES
FINISH:
DEBUR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION
NAME
SIGNATURE
DATE
DRAWN
CHK'D
APPV'D
MFG
Q.A
MATERIAL
MS
WEIGHT:
TITLE:
19451, Motoman Ac; ECE
sharmam1@udayton.edu
SolidWorks Student Edition.
For Academic Use Only.
DWG NO.
Frame_support_T012
A4
SCALE:2:1
SHEET 1 OF 1



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR: INCHES	FINISH:		DEBUR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE	TITLE: 19451, Motoman Ac; ECE sharmam1@udayton.edu	
DRAWN					
CHK'D					
APPV'D					
MFG					
Q.A				DWG NO. Frame_support_T013	A4
			MATERIAL: MS		
			WEIGHT:	SCALE:1:1	SHEET 1 OF 1


38
20
15.50
30
14.50
9
9
4
16
11
19
35
19
Ø6
30
80
62
80
30
4
14
22
4
4
Ø6
15.50
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:
FINISH:
DEBUR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION
NAME
SIGNATURE
DATE
DRAWN
CHK'D
APPV'D
MFG
Q.A
MATERIAL
MS
WEIGHT:
TITLE:
19451, Motoman Ac, ECE
sharmam1@udayton.edu
SolidWorks Student Edition.
For Academic Use Only.
DWG NO.
M_support
A4
SCALE:1:2
SHEET 1 OF 1


1.52
Ø128
Ø146
1.52
60/1000 inch = 1.52 mm
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
SURFACE FINISH:
TOLERANCES:
LINEAR:
ANGULAR:
FINISH:
DEBUR AND BREAK SHARP EDGES
DO NOT SCALE DRAWING
REVISION
NAME
SIGNATURE
DATE
DRAWN
CHK'D
APPV'D
MFG
Q.A
MATERIAL
Aluminium
WEIGHT:
TITLE:
19451 Motoman account, ECE
sharmam1@udayton.edu
DWG NO.
Ring_cuff_base
A4
SCALE:1:5
SHEET 1 OF 1
SolidWorks Student Edition.
For Academic Use Only.

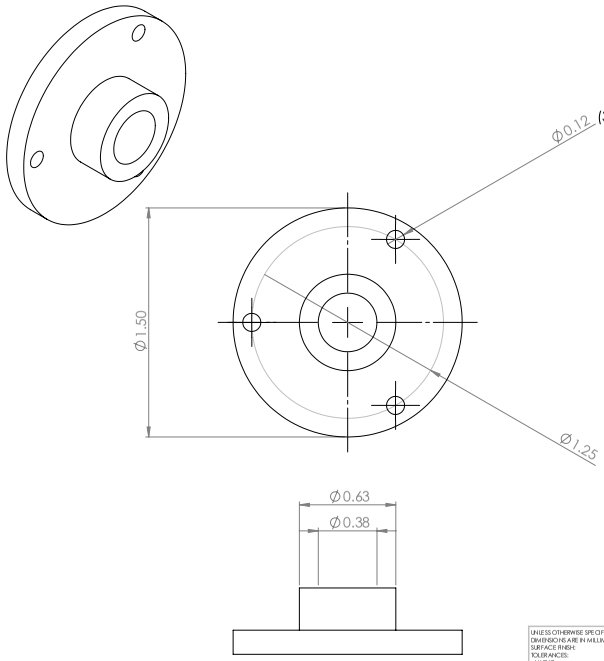


All dimentions in inches

Chamfer as needed

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:		DEBUR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE	TITLE: Motoman A/C	
DRAWN				19451	
CHK'D				sharmam1@udayton.edu	
APP'V'D					
MFG					
Q.A.					
			MATERIAL: Al	DWG NO. yawMotorCoupling	A3
			WEIGHT:	SCALE:2:1	SHEET 1 OF 1

SolidWorks Student Edition.
For Academic Use Only.



All dimentions in inches

Chamfer as needed

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS SURFACE FINISH: TOLERANCES: LINEAR: ANGULAR:	FINISH:		DEBUR AND BREAK SHARP EDGES	DO NOT SCALE DRAWING	REVISION
	NAME	SIGNATURE	DATE	TITLE: Motoman A/C	
DRAWN				19451	
CHK'D				sharmam1@udayton.edu	
APP'V'D					
MFG					
Q.A.					
			MATERIAL: Al	DWG NO. yawMotorCoupling	A3
			WEIGHT:	SCALE:2:1	SHEET 1 OF 1

SolidWorks Student Edition.
For Academic Use Only.