FINAL REPORT

EML 4552C – Senior Design– Spring 2013 Deliverable

Team 10 – CISCOR Autonomous Ground Vehicle
April 23, 2013

Team Members:

Donald Allex
Tye Buckley
Richard Komives
Cesar Mize

Department of Mechanical Engineering
Project Advisors
Dr. Chiang Shih

Department of Mechanical Engineering
Dr. Kamal Amin

Department of Mechanical Engineering
Reviewed by Advisor(s):
Dr. Oscar Chuy
Researcher, CISCOR

Department of Mechanical Engineering
Dr. Emmanuel Collins
Director, CISCOR

Vehicle WARNINGS
READ BEFORE OPERATING VEHICLE

Warnings:

- Check ALL emergency switches and make sure they are working properly
- DO NOT HOT PLUG/PLAY
    - Make sure all connections to M-Drive actuators are solid and USB is plugged into the computer before connecting power
- Small red switch on side of box controls power to M-Drives and Maxon Motor Controller and motor
- Make sure all actuators are in correct position before and after powering on
    - Throttle needs to remain against the lever but not depressing it
    - Brake should not be activated
    - Gear Select should be in park
    - Steering wheel should be straight forward
- Emergency Stop switches must be in the **extended** position before cranking Goliath
- Goliath must be in **park** and throttle must not be activated before attempting to crank

Cautions:

- Always follow Polaris recommended operating range
- Always follow safety precautions specified on Page 44

READ BEFORE OPERATING VEHICLE

Vehicle WARNINGS

Table of Contents

Acknowledgments

We would like to thank the Center for Intelligent Systems, Control, and Robotics (CISCOR) for not only providing our vehicle platform but also providing us with all the necessary funding and resources for this project.

We would also like to recognize and thank our project advisors Dr. Oscar Chuy and Dr. Emmanuel Collins for their countless support and advisement throughout this semester.

Additionally, the team would like to thank Ryan David-Reyes for helping the team develop and test the remote control software utilized in the initial testing of this vehicle.

CISCOR Introduction

The Center for Intelligent Systems, Control, and Robotics (CISCOR) uses engineering knowledge from the Mechanical and Electrical & Computing Engineering fields to develop new systems and implement technological innovations in the area of Intelligent Systems, Control, and Robotics. Their designs are used to solve practical problems in both industrial and governmental applications. CISCOR represents a cooperative approach for conducting interdisciplinary research in the automated systems area across two departments (Mechanical and Electrical & Computer) in the College of Engineering and the FSU Department of Computer Science. The Center's goal is to provide a means for the State of Florida to achieve national prominence in the area of automated systems and to assume a leadership role in the State of Florida's technology of the future. Established in 2003, CISCOR has become a leading center in Florida for the development and implementation of technologies related to intelligent systems, control, and robotics.

Project Introduction

Currently there is no off road vehicle platform for autonomous research and design in CISCOR's inventory. This team was tasked with modify an existing all-terrain vehicle (ATV) to be capable of full unmanned movement by designing, researching and manufacturing components to allow unmanned locomotion control.

To successfully complete this project, the resulting vehicle must be able to duplicate human-rider locomotion while being controlled solely unmanned. The vehicle must also provide mounts and protection for the various sensors that will be utilized by CISCOR to gain data about the vehicle's environment. An additional constraint from the project sponsor is that the vehicle must remain human-rider operable in case of a failure of the computer controlled systems. The design should not excessively deviate from the capabilities of the original vehicle and should remain aesthetically pleasing. The resulting vehicle has been named **G**as **O**perated **L**and **I**ntelligent **A**ll-Terrain Ve**H**icle or **G.O.L.I.A.T.H**.

To fully automate the operation of the GOLIATH, four basic locomotion systems must be altered to include a computer-controlled input. These systems consist of Steering, Braking, Throttle, and Gear Shifting. These designs must be rugged enough to withstand stresses and debris encountered in an off-road environment, while still maintaining the ability to reproduce human-rider like operation of the vehicle. These designs must also be very accurate due to the vehicles intended use as a research tested. Each locomotion system was tested for distance of travel required, force required, and reasonable response time for function. These values and the overall project constraints were used to design feasible systems to automate the locomotion. Input from the design group and the project sponsor was used to determine the final designs for each system. These designs, and their component selection are described in detail in this report.

Background

The increased use of autonomous vehicles both in the private and public sector has grown dramatically in the past forty years [2]. The wide platform usage of these autonomous vehicles makes them a favorite among the defense sector and plays an integral part in today's battlefield. However, the battlefield is not the only place where these autonomous vehicles are used. Today, many commercial vehicles are more intelligent and responsive. Many modern cars now have the ability to adapt to terrain and different weather conditions automatically, which in turn relates to the autonomous functionality of an autonomous vehicle [1].

An autonomous ground vehicle (AGV) is a vehicle that operates while in contact with the ground and without an onboard human presence. AGVs can be used for many applications where it may be inconvenient, dangerous, or impossible to have a human operator present. Generally, the vehicle will have a set of sensors to observe the environment, and will either autonomously make decisions about its behavior or pass the information to a human operator at a different location who will control the vehicle through radio communication or other teleportation devices [1].

Most recently, autonomous research and development has significantly grow and will continue to grow in the coming decade. In the coming decade, there will be a 4.6% increase in unmanned vehicle production in the military sector [3]. Researchers and institutions are finding innovate ways to have these vehicles compute and respond to different environments and scenarios. At CISCOR, researchers hope to advance the field of autonomous intelligence by developing cutting edge technology and algorithms in this field.

Locomotion Mechanism Introduction

There are four main locomotion controls on GOLIATH: Steering, Braking, Gear Select, and Throttle. In order to achieve unmanned movement, all four components must be retrofitted with devices to actuate the desired response. Figure 1 illustrates the four locomotion controls on GOLIATH. Each locomotion mechanism is denoted as a subsystem.

Four main locomotion mechanisms on GOLIATH

1) Steering
2) Braking
3) Gear Selection
4) Throttle

Figure 1 – Illustration of the locomotion controls on GOLIATH

Steering – Steering is the largest and most complex of all the subsystems. Multiple components go into the total assembly of this subsystem with many more micro-systems to each component. On GOLIATH, the steering motion is assisted by an auxiliary Power Steering Unit (PSU) that reduced the force required to turn the steering column. This feature is usually found on much larger vehicle and thus makes this a unique feature for an autonomous ground vehicle of this size. Figure 2 illustrates the steering component on GOLIATH.

Figure 2 – Steering Mechanism

The specifics of GOLIATHS steering characteristics are as followed:

Turning range: 162 degrees

Turning force required at end of handlebar: 32 lbf

Braking – The braking mechanism on GOLIATH works much like that of a bicycle brake system. When braking, a user pulls on the brake leveler that engages the brakes. The brake lever however does not pull on a pull string much like that in a bicycle brake system but instead the lever pushing on a cylinder which pushes brake fluid across the braking system. The master cylinder is what actually activates the brakes in this system. Figure 3 illustrates the braking component on GOLIATH.

Figure 3 – Braking Mechanism

The specifics of GOLIATHS braking characteristics are as followed:

Braking travel range of lever: 1.85 in.

Force required to activate brake lever: 32 lbf

Gear Select – Gear selecting on GOLIATH is very straightforward. There is a single lever arm that can be pushed or pulled on to selected the different gear positions (Park, Reverse, Neutral, Low, and High). Figure 4 illustrates the gear select component on GOLIATH.

Figure 4 – Gear Select Mechanism with arrow point to shift arm

The specifics of GOLIATHS gear select characteristics are as followed:

Force Required to shift gears: 30 lbf

Total travel of lever arm: 4.1 in.

Distance between different gears: max of .5 in.

Throttle – Throttle actuation on GOLIATH is also very straightforward. The throttle actuator is located on the right handlebar and is actuated by the right thump. If a user wishes to accelerate, he pushed on the throttle lever and the vehicle accelerates forward. Figure 5 illustrates the throttle component on GOLIATH.

Figure 5 – Throttle Mechanism

The specifics of GOLIATHS throttle characteristics are as followed:

Total throttle lever travel: 45 degrees

Force required to actuate throttle: 8 lbf at 2 cm from axis of rotation

Locomotion Actuator Designs

Braking

To meet our goal of automating all the locomotion systems of the ATV, a braking system must be devised that allows a computer input to precisely control the amount of braking force applied to the vehicle. Like the other locomotion systems on the ATV, the braking system requires accurate operation, fast response times, and the system must not be designed in such a way as to render the ATV inoperable by a human rider. Other factors, such as cost, simplicity, and aesthetics, factored into the final design of the unmanned braking system.

Design Selection

The current braking system operates by having the rider of the vehicle depress a hand-lever located on the left handlebar. This lever pivots about a pin on the handle bar and the rotating motion of the lever causes a piston in the master cylinder to be depressed. The depression of the piston inside the master cylinder pressurizes the fluid in the brake lines, causing the calipers located at all four wheels to engage the brake pads. To accurately determine the braking force being applied to the vehicle, the pressure of the fluid inside the braking lines must be known. The position of the piston in the master cylinder relative to its rest position can also be used to infer a braking force. The project advisor requested that the final system be able to record and report both data points.

Multiple designs were evaluated that met the basic requirements for successfully automating the braking system. One of these designs used to a linear actuator to directly depress the brake piston, obviating the need for the brake lever. Another design used the same method, but included another brake cylinder, spliced into the existing brake lines. This effectively created a parallel braking system. The final design concept and the one which was eventually selected with the project advisors approval is a design in which the linear actuator acts upon the brake-lever so as to depress the brake piston in much the same manner as the existing system is currently operated by a human rider. Figure 6 illustrates this design.

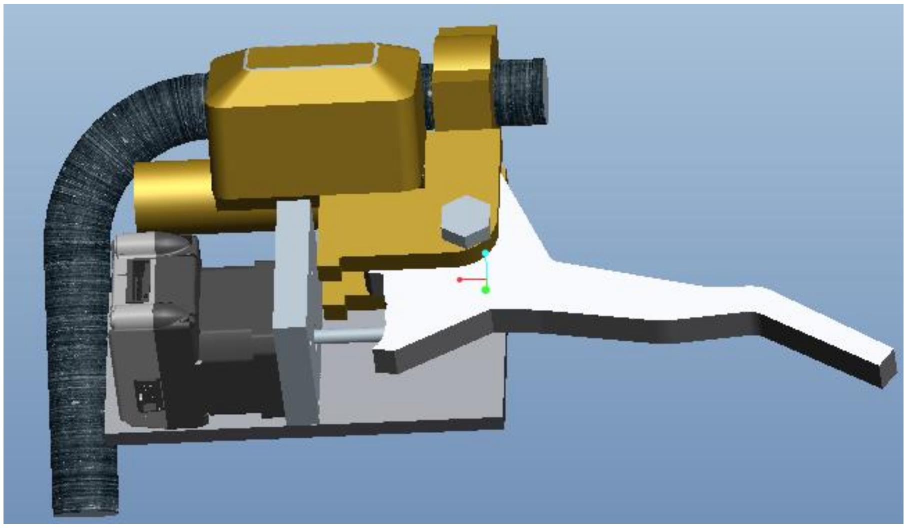


Figure 6 - Final Braking System Design

The CAD model above shows a linear actuator mounted in front of the existing braking system on the handlebars of the ATV. This actuator acts upon a protrusion of the brake-lever causing it to pivot about the pin and depress the brake piston. This design met the specified requirements for the locomotion systems of the ATV and does not interfere with the operation of the existing braking system by a human rider.

Component Selection

The most important component in this system is the linear actuator which is used to rotate the brake-lever. For the actuator to be mounted in the desired location in front of the handlebars, the actuator is of a small size so as to not interfere with other vehicle components mounted inside the handlebars which included the vehicles headlight. The mount on the front of the handlebars is in a very visible location to observers of the vehicle, which makes the aesthetics aspect of the design more prominent than other possible braking system designs. The current design is also significantly exposed to any dust, debris, or moisture which may be present in the environment. To enhance the aesthetics of the design and to combat damage due to debris, a plastic housing which encases a significant portion of the design will be later manufactured. This housing will be easier to design and will be more aesthetic the smaller it is, thus reinforcing the need for a small linear actuator. The housing is not shown in the CAD model above to make other components of the model clearer.

The actuator used in this system also fit the criteria for strength and accuracy. The force needed to depress the brake piston through its maximum travel was determined by measuring the force exerted on the brake lever at a certain distance, and then converting this force through the use of moment arm equations into a force directly acting upon the brake piston. After the calculations using Equations 1 in Appendix C, this force was determined to be ~35 lbf. To provide a small margin of error, it was determined that an actuator with a minimum of 50 lbs. of thrust force would be required to directly depress the piston. The final design does not utilize the actuator directly acting upon the piston, but instead uses the portion of the brake-lever acted upon as a moment-arm to then depress the piston. This means that the actuator actually required less than the 35lbf. to depress the piston. The actuator used in the braking system was also required to have some manner of measuring its displacement during operation. This is achieved with actuators using either a potentiometer or a type of encoder. After discussion with the project advisor, it was determined that a potentiometer would not be sufficient to achieve this constraint due to the high probability of failure after repeated uses. Thus, some type of encoder, preferably one integral to the actuator, would be needed to provide position feedback. The actuators which include encoders are much more expensive than those with more limited means of positional feedback. This was taken into account during the search for an adequate model, and every effort to find a sufficient, cheap actuator was made.

Due to safety concerns, and the effective operation of the vehicle in unknown terrains, the braking system must also be able to develop full braking force in a short amount of time. This reduces the chances of collision with objects which are not detected by the sensors until they are in close proximity to the vehicle. The response time of the system is dependent upon the total displacement of the linear actuator that is required for the maximum braking force to be applied and the speed of the actuator.

Finally, due to limitations imposed by the power supply from the operation of the vehicle itself, and the power consumption from the many other components of the final autonomous vehicle, the actuator must be operable with an input voltage no higher than 24 VDC. Using these criteria, the M-Drive 17 linear actuator manufactured by Schneider Electric, shown below, was chosen.

Figure 7 - Linear Actuator

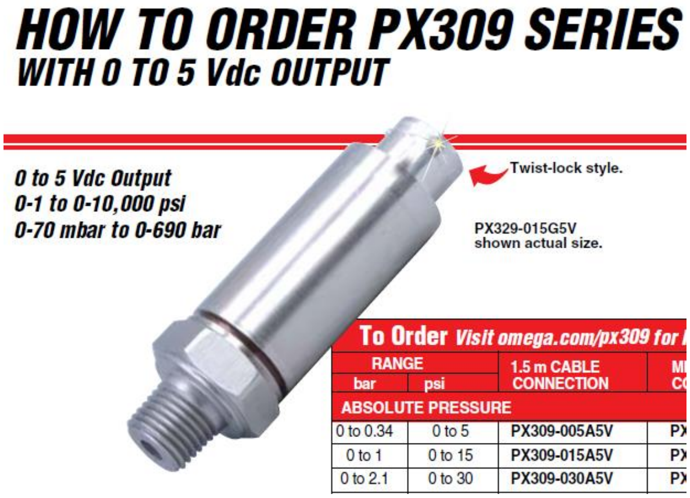
This actuator features an internal 512 line magnetic encoder which allows for a resolution of 2048 steps per revolution of the actuators drive motor. This actuator is able to generate more than 50 lbf. of thrust, exceeding the strength constraint determined earlier. The range of input voltage for this actuator falls between 12 and 48 VDC, encompassing the necessary low voltage range. Using the high load scenario of 50 lbf. of thrust and the travel length necessary to achieve full braking force, the system developed full braking force in less than one second from receiving the command signal. This actuators command and control systems were sufficiently compatible with the other systems of the autonomous vehicle. After presented with the technical specifications and the price, the project advisor approved the procurement and use of the actuator for the braking system.

The actuator also filled the required small size constraint. Shown in Appendix B, the rough dimensions for this actuator are 2.3” x 2.2” x 1.7”, which is easily mountable in the space in front of the handlebars without interference with existing vehicle components.

This actuator features a screw with a flat end which was placed against a surface on the brake lever. The revolution of the screw by the actuator causes the end of the screw to displace relative to the actuator, causing the screw to force the brake lever to rotate.

While the linear actuator possess an encoder to provide feedback about the operation of the braking system, the project advisor desired to have another, redundant feedback system that would be more accurate. This will be achieved by connecting a pressure transducer to the brake lines so that the pressure developed in the lines, which directly leads to the braking of the vehicle, can be measured. The transducer will be mounted on the brake line between the brake master cylinder and the distribution valve which allows the brake fluid to activate the brakes on all four wheels simultaneously. After discussion with the project advisor, it was determined that the criteria for selection of the pressure transducer was that it must be accurate, have a fast response time, be cheap, and produce a voltage output to determine the measured pressure. The use of a pressure gauge with a visual dial would not be necessary to achieve this portion of the system and would drastically increase the price of the transducer. Due to this item being more common and less critical to the overall operation of the braking system, an additional constraint to procure the item from a company on the previously approved procurement list if at all possible was added.

To begin selection of the transducer, the range of pressures developed in the brake lines during operation must be measured. Due to the fact that very little flow of the brake fluid occurs in the brake lines during operation, the static pressure of the fluid is the only necessary measurement. This was done by connecting the existing brake line to a small, analog pressure gauge and determining the maximum pressure that developed when the brake piston was fully depressed. This was approximately 730 psi. Using this pressure reading, and the other criteria, the companies listed as per-approved for procurement were checked for applicable products. It was determined that the PX309-1KG5V from Omega Engineering sufficiently met all the criteria. The selected transducer is shown in figure 8.



HOW TO ORDER PX309 SERIES
WITH 0 TO 5 Vdc OUTPUT

0 to 5 Vdc Output
0-1 to 0-10,000 psi
0-70 mbar to 0-690 bar

Twist-lock style.

PX329-015G5V
shown actual size.

To Order *Visit omega.com/px309 for*

RANGE		1.5 m CABLE CONNECTION	MI CO
bar	psi		
ABSOLUTE PRESSURE			
0 to 0.34	0 to 5	PX309-005A5V	PX
0 to 1	0 to 15	PX309-015A5V	PX
0 to 2.1	0 to 30	PX309-030A5V	PX

Figure 8 - Pressure Transducer

The selected transducer has the required voltage output, with a range of 0 to 5 VDC and has a range of pressure from 0 to 1000 psi. The transducer will work with the liquids like that found in the brake lines. This transducer has an accuracy of ± 0.25% of the static pressure and a response time of under a millisecond. The actuator will be connected to the existing brake lines using a 1/8" flare tee commonly used on brake lines and an adapter from the brake line tee to the NPT threads of the transducer. This will cause minimal interference in the normal operation of the braking system. After presented with the technical specifications and the price, the project advisor approved the procurement and use of the pressure transducer for the braking system.

The correct mounting of the autonomous braking system features complex geometries determined by the limited space, the angles of the handlebars relative to the braking master cylinder, and the need to account for the arcing motion of an object pivoting around a fixed point.

The actuator is mounted vertically on a base plate connected to the existing master braking cylinder with bolts. The actuator will be allowed to rest on this plate, since forces in the vertical direction will not develop during operation of the system. The actuator will be constrained from lateral movement by connection to a vertical plate connected to the base plate. The four threaded holes shown in the Appendix A – Brake Drawinings displaying the linear actuator dimensions will be used to connect the actuator to the vertical plate. The four bolts used to make this connection will also resist any torqueing or motion of the linear actuator perpendicular to its travel. The existing brake handle does not possess a convenient place for the linear actuator to act upon, so a new brake handle will be manufactured with an altered geometry to best achieve smooth travel during operation of the autonomous system. These components of the braking system will not be enclosed by the plastic housing which will protect the actuator, and thus will need to be able to resist rust and corrosion. The pieces may also require modifications after the initial manufacture to better fit the vehicle. To ensure that the pieces are cheap, easily machine-able, and corrosion-resistant, aluminum was chosen as the material to use for their manufacture. None of the designed pieces would be thicker than 0.45" or large in any dimension than 12" making procurement of the basic stock from any metal provider very fast, reducing lead time and price. The geometry of the initial design is simple to enable fast manufacture using water-jetting so that the mounting components can be fit-checked, and then adjusted if needed.

Current Status

This locomotion control mechanism is finalized and fully tested. This design has met and exceeded all design requirements and is ready for operation. Refer to Figure 9 for an illustration of the final design.

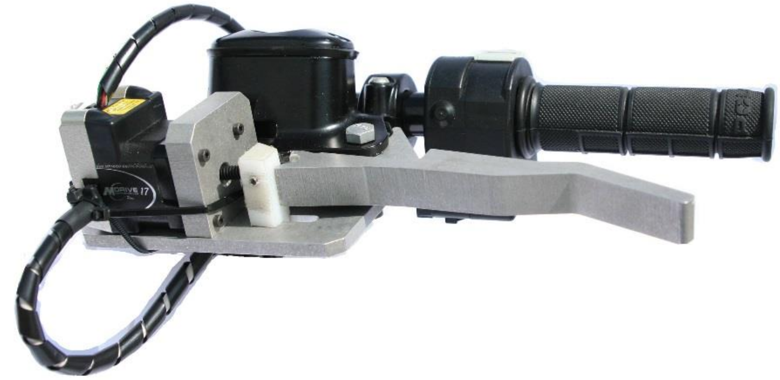


Figure 9 – Fully assembled Brake manipulation design

Steering

In order to make the ATV fully unmanned, a steering system must be designed that allows for full computer control. Our steering system must be able to utilize the steering's full range of motion, it must be able to withstand all feedback from the terrain, and the motor output must be powerful enough for any terrain and speed. The system must also be designed in such a way that allows for full user control when necessary. When assessing possible designs we also took into account other factors, such as cost, lead time, simplicity, and reliability.

Design Selection

The steering system of the ATV works by having a user turn the handlebars clock-wise or counter-clockwise. Attached to the handlebars is a steering column. The lower portion of the steering column is then connected to the power steering unit (PSU). When the user turns the handlebars, which turns the steering column, a torque sensor in the PSU determines which direction the steering column is turning and activates a brushless motor to aid in steering the vehicle. A secondary steering column extrudes from the bottom of the PSU which connects to linkages and tie-rods which physically makes the wheels turn. A basic drawing of the PSU and connected steering columns in shown below.

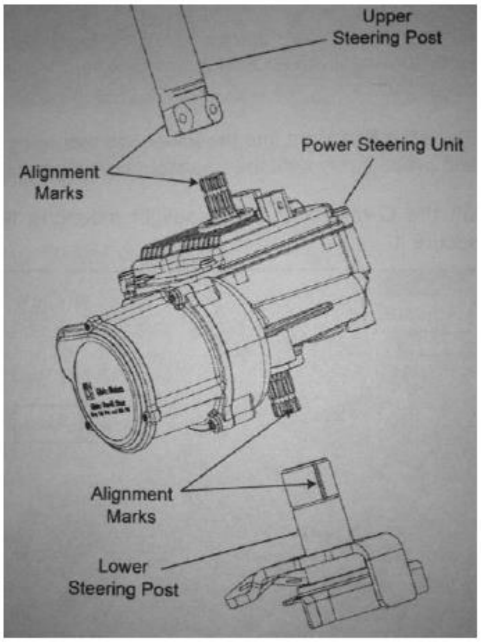


Figure 10 – Globe Motors power steering unit on GOLIATH

The initial design for the actuation of the steering system, as shown in the previous semester's final report, was to purchase a programmable power steering unit (PSU) from Globe Motors. This company designs and manufactures the original PSU in our Polaris ATV. We were told that a supplementary PSU could be purchased that had no proprietary firmware, was completely programmable, and contained an absolute encoder that provided position feedback. Until the beginning of this semester, this design choice seemed to be the most logical since the whole unit had the same mounting profile as the current unit and cost approximately $2000. At the start of Spring 2013 semester, our group spoke more with Globe Motors to find out that software needed to be purchased alongside the PSU in order to actuate it properly. This software cost approximately $5000. This was way above our allotted budget which caused us to design our own actuation system for the steering locomotion.

Before beginning to design our own actuation system we ran a simple experiment to determine the torque required to turn the steering column. We attached a spring scale to the end of the handlebars and pulled until the wheel began turning. At this point we took a reading to determine the force required. We then used a simple torque equation and found we needed approximately 30Nm of torque. This was used later on in the design process to determine a motor that would be sufficient to turn the vehicle.

Beginning to design the steering locomotion system, we first chose where the proposed system would be mounted in order to size the design appropriately. On the front end of the ATV was located a battery placed on a tray which was directly in front of the steering column. We relocated the battery out of this area and found that this would be suitable spot to mount the design. We then chose, out of the three proposed designs created in the concept generation portion of the project and chose a chain drive system due to the small space constraints. A mount was then designed where the motor would be placed on a top plate (motor shaft pointing downward) which would be angled at 25 degrees to match the angle of the steering column relative to the body of the ATV. Below the motor (on the opposite side of the motor mount plate) would be a small sprocket. This sprocket would connect to a chain which would then connect to another sprocket on the steering column of the vehicle. The sprocket on the steering column would be held in place, rigid to the steering column, using a neck coupler which would screw on either side of the column and have a through hole which would allow us to put a bolt through the coupler and steering column. This design was finally chosen due to manufacturability, fairly low cost, and ease of mounting. The design also fit within the space constraints without having to remove or move any major components of the vehicle. A CAD model of the designed system can be seen on the following page.

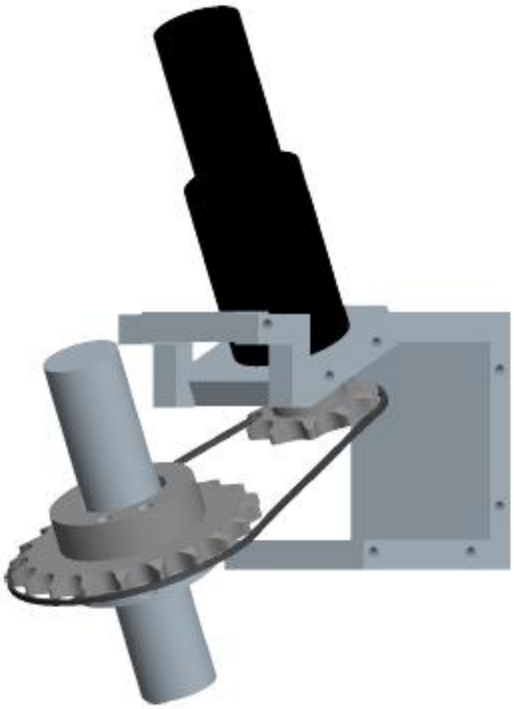


Figure 11 – Fully assembled Steering manipulation design

Component Selection

Once the design was finalized, the process of purchasing components, materials, and hardware was the next step. The first component to be chosen was the motor. A 24V 100W Maxon motor was provided to us that was able to output approximately 15Nm of torque. We decided to use this motor to cut down on the overall cost of the project. This torque rating, however, was not sufficient to turn the wheels of the ATV. This lead to the realization that using a small sprocket on the motor and a larger sprocket on the steering column would allow us to raise the torque felt by the steering column. Since the motor was outputting 15Nm and we required 30Nm, an approximate ratio of 2:1 was chosen. This lead us to choose, due to space constraints from the motor mount and vehicle itself, a small sprocket with a radius of 2" and a large sprocket with a radius of 4.3". Because the vehicle was required to weather off-road environments and the system would be handling a large amount of torque, stainless steel sprockets utilizing a #50 ANSI roller chain were chosen. These sprockets were ordered from McMaster Carr as well as the chain. The chain purchased was also constructed out of stainless steel. The final portions of the design that needed to be chosen involved the material used for the mount and the steering column sprocket coupler. 6061 aluminum was chosen due to its low cost, low weight, and ease of manufacturing.

Current Status

This locomotion control mechanism is finalized and fully tested. This design has met and exceeded all design requirements and is ready for operation. Refer to Figure 12 for an illustration of the final design.

Figure 12 – Fully assembled Steering manipulation design

Gear Select

In order to have full unmanned motion of the ATV (GOLIATH), we must implement a mechanism to control the gear or drive selection. Specific measurable objective must be met in order for full control. The mechanism must be able to achieve the full range of motion from first to last gear, Park to High-forward, while having the ability to stop at each gear in between. Along with being able to perform this operation with accuracy and repeatability, the system must also overcome the force required to move the gear selector. Other, more general factors, such as cost, durability, and aesthetics will be accounted for in our final design.

Design Selection

The Polaris Sportsman EPS 550 comes with five different gear selections: park, reverse, neutral, low-forward, and high-forward, in order from the driver to the front of the ATV. To select different gears the rider pulls/pushes on the shift arm till the correct gear is selected. There is a digital feedback to let the rider know what gear they are in. Shown in Figure 13, the shift arm is mounted about a pivot joint. The arm is coupled to the motor gear input selection by a connecting rod.

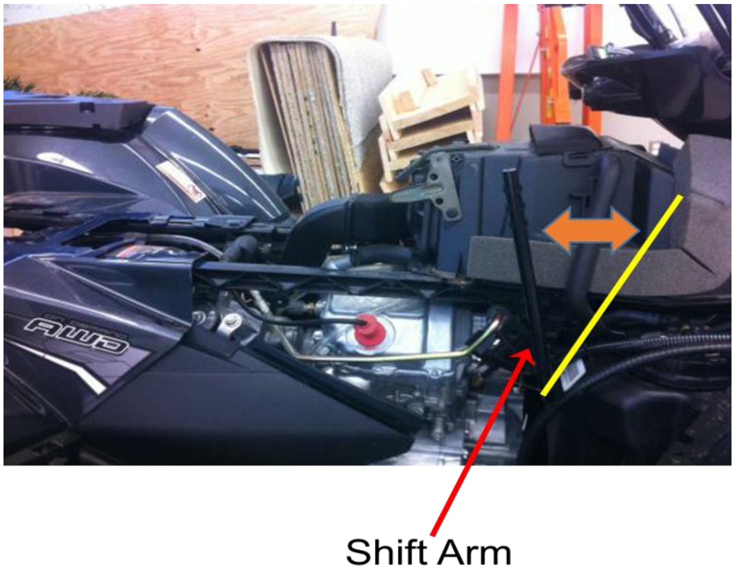


Figure 13 - Shift Arm Location and Motion

Two main design options were assessed, both of which could accomplish all gear locations. One option was a servo directly mounted to the motor. This eliminated moving parts simplifying the process. Mounting options were one of the biggest limiting factors with this design. Also, another constraint was placed on the project by our advisor which required the vehicle to still be able to perform standard user operation. This eliminated a servo mounted design option. The second and selected design was to use a linear actuator to perform the required motion. The actuator can be placed in line with the shift arm, and when activated, could move the shift arm to the specified location. GOLIATH has an aluminum plate located in the front that would allow for the linear actuator to be mounted in-line with the shift arm. In order to still permit user interaction, the coupler from the linear actuator will be designed to allow for a quick disconnect. By using a linear actuator design with a coupler link, all of the criteria for complete control of the shifting mechanism are satisfied. The design mount and location is illustrated in the figure below.

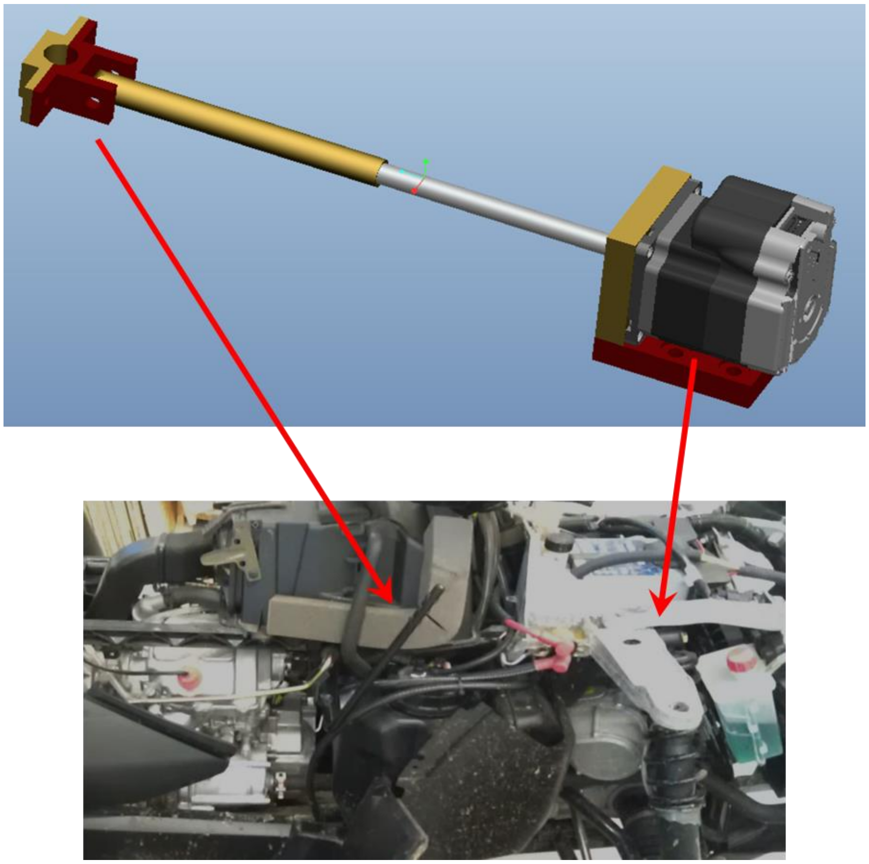


Figure 14 - Mounting Design and Location

Component Selection

After the design concept of using a linear actuator to select the different gears of GOLIATH was chosen, an actuator was selected that met very specific design criteria. Some of the requirements are solely due to performance specifications while others are due to design choices by the team.

Due to their high importance, performance specifications were the main driving force behind the selection of the linear actuator. One of the main specifications is the actuator needs to be able to overcome the 30 pound-force required to move the shift arm between the locations. This force was found by using a mass spring scale with a specific spring constant. By connecting the springs to the shift arm and measuring the displacement of the spring required to move the shift arm, a total force required for movement could be calculated. In addition to the force requirement, the actuator must have a total travel of at least 4 in with the ability to stop at different locations. From park to high-forward the shift arm travels a distance of 4 in. Gears from park to low-forward are separated by 0.75 in each while 1.5 in separates low-forward to high-forward.

Other criteria placed on the selection process by the team, included an accuracy requirement of 0.01 in along with a total time from park to high-forward of 5 sec. Also, for programming purposes a location feedback is need. The location need to be transmitted by a non-contact encoder. One of the final things considered in the selection process is the actuator must be heavily resistant to vibrations. Dust and debris effects were considered and will be avoided by the use of a housing to encompass the actuator.

In addition, due to the performance ability of GOLIATH and the components that it will have to support, the power supply is limited to a maximum of 24 VDC. Thus the linear actuator must be able to perform all of its required specification with the allotted 24 VDC.

Using these criteria, the team selected the M-Drive 23 non-captive linear actuator by Schneider Electric. This actuator is shown in figure 15.

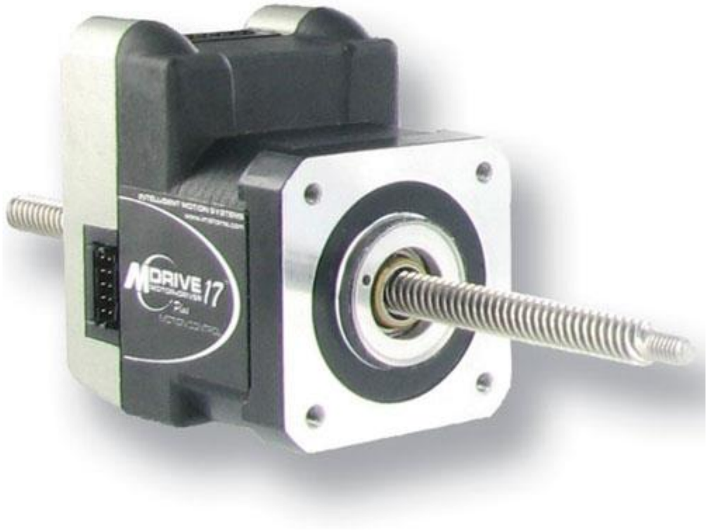


Figure 15 – Schneider Electric M-Drive 23 Linear Actuator

The M-Drive 23 by Schneider Electric is capable of producing up to 200 lbs of thrust. The non-captive shaft has a specified 7.5 in total length, which encompasses the required 4 in of travel. With the selected thread option of 0.001 in per step and running at a speed of 2000 step per second, the M-Drive 23 can complete the required 4 inches in 2 sec, theoretically. The M-Drive can operate at 24VDC, complying with the power output from GOLIATH. A 512 line magnetic encoder allows for 51200 steps per revolution.

As depicted in the figure 15, the shaft will have a threaded end, UNC ¼-20 thread, which will serve as the mount for the coupler link. The coupler will serve as a connecting rod from the linear actuator to the shift arm. By not rigidly connecting the actuator and shift arm, a user could easily disconnect the two if power were loss to the system and resume normal operation. Figure 16 shows how the coupler will be connected to the shift arm.

Coupler Design

One main specification for the coupler is that it needs to be able to handle the motion of the shift arm. Because the shift arm is located about a pivot point, the motion of any point along that arm is a two dimensional planar motion. The coupler must then be able to transform the two degrees of freedom output of the shift arm to the one degree of freedom input from the actuator. This can be accomplished by combining multiple joints with different degrees of freedom. By joining a slider joint and a pivot joint, we are able to translate two degrees of freedom into one. Therefore, when the linear actuator is in motion the pivot joint will account for the angle of the shift arm and the slider joint will account for the change in height of the connection point.

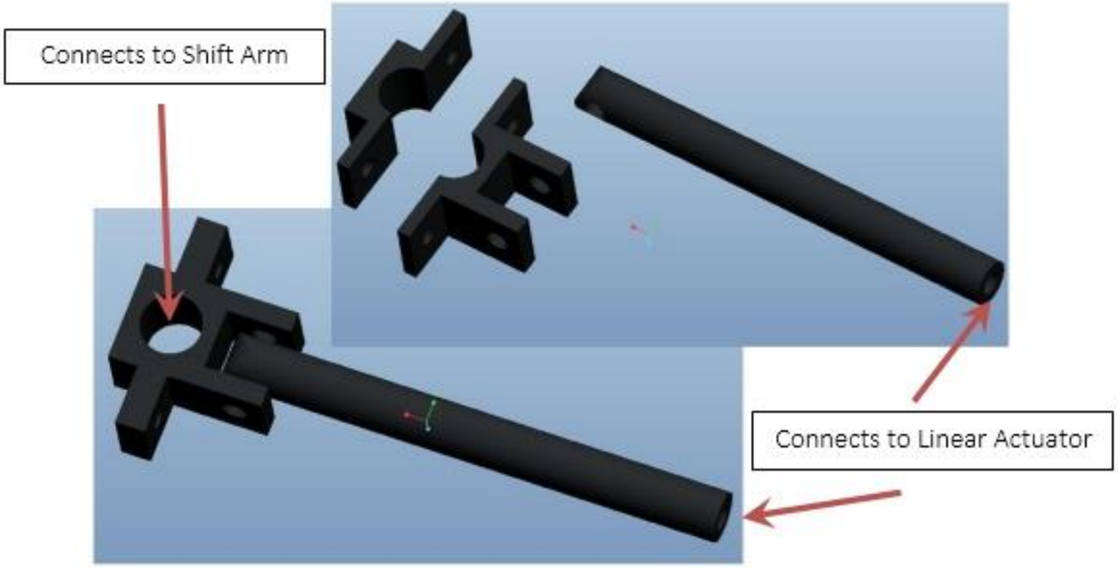


Figure 16: Coupler from Actuator to Shift Arm

Mounting Design

In order to mount the linear actuator to the metal arm, shown in figure 17, a simple L-shaped bracket will be used. Four screws will mount the actuator to the bracket. This will limit the actuator from slipping back and forth while under the forces required to move the shift arm. An L bracket was used to hold the actuator on the mount. This was decided by the team for several reasons. One is the multiple communication ports that we need to have access to. Another reason is the simplicity of removing the actuator if necessary. Due to minimal forces in the vertical direction, it was determined that the four mounting screws and L-bracket would suffice in holding the linear actuator in place during operation. Figure 14 shows the mounting configuration for the linear actuator.

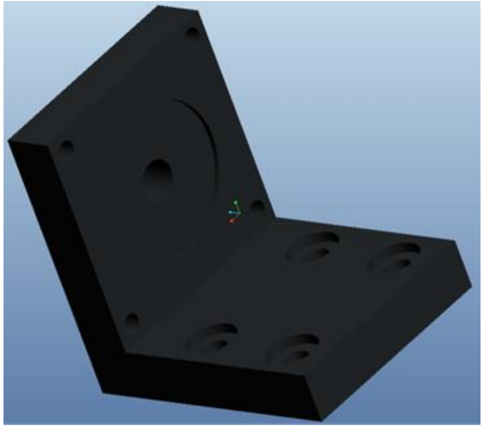


Figure 17 - L-Shaped Bracket for Actuator

Current Status

This locomotion control mechanism is finalized and fully tested. This design has met and exceeded all design requirements and is ready for operation. Refer to Figure 18 for an illustration of the final design.

Figure 18 – Fully assembled Gear Select manipulation design

Throttle

In order to have full autonomous motion of the ATV (GOLIATH), we must implement a mechanism to control the throttle. Specific measurable objective must be met in order for full control. The mechanism that will be used to control the throttle must utilize the whole range of throttle capabilities of the ATV. In other word, the system must be able to accelerate from zero to its top speed of roughly 60 mph. Furthermore, the system must be accurate and responsive. We would like the system to be able to be able to have the accuracy to specify the exact speed it so desires. The response time of this system is also to be considered. The system must meet or exceed the response time of a human actuating the throttle mechanism. Other, more general factors, such as cost, durability, and aesthetics will be implemented in our final design.

Design Selection

To accelerate on the Polaris Sportsman 550 a user utilizes a thump throttle lever located on the right handlebar as depicted in Figure 19. When exploring initial design concepts, the group came to the conclusion that simplicity was key in designing a system to actuate the throttle. We also considered in our design selection the ability to have the throttle actuator system seamlessly be able to be user driven without disconnecting of components. That is, a user can mount onboard GOLIATH and actuate the throttle without any modification to the unmanned actuator throttle system.

Figure 19 – Factory installed throttle actuator mechanism

With all these design consideration, the following is the final throttle actuator design for GOLIATH. This throttle actuator system will mimic a rider's thumb pushing on the throttle. This will be done by mounting a high torque stepper motor with the output shaft turning a mechanical arm. This mechanical arm will be coupled to a slider that will push the throttle lever when the stepper turns. Figure 20 illustrated this design. This design not only allows for a user to directly manipulate the throttle freely but also satisfies all measureable objectives when coupled with the Schneider Electric 23 M-drive stepper motor.

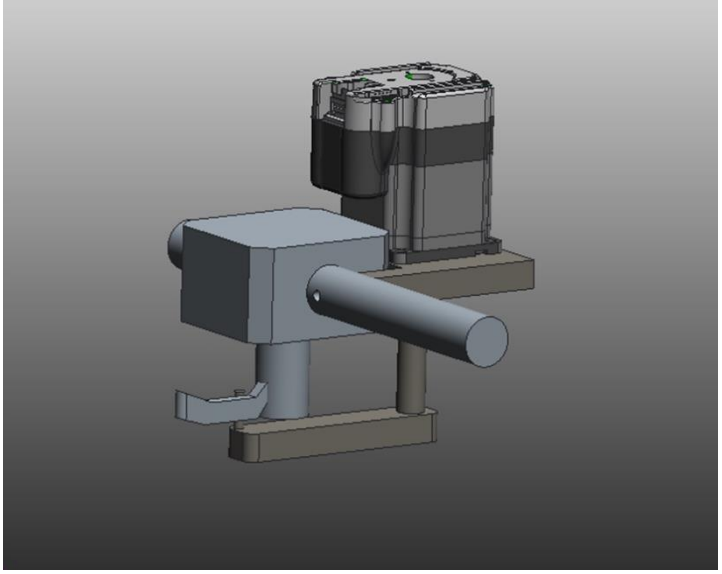


Figure 20 – Throttle actuator for unmanned locomotion

Component Selection

After the design concept of using a stepper actuator to actuate the throttle mechanism of GOLIATH was decided upon, a stepper motor was selected that met very specific design criteria. Some of the requirements are solely due to performance specifications while others are due to design choices by the team.

Due to their high importance, performance specifications were the main driving force behind the selection of the stepper motor. One of the main specifications is the actuator needs to be able to overcome and hold the .84 Nm required to actuate the throttle mechanism. This numerical value was computed by calculating the torque at the center of the throttle lever using a spring mass system and a perpendicular distance. Equation 1 in Appendix C was used for this. This torque was found by using a mass spring scale with a specific spring constant. By connecting the springs to the shift arm and measuring the displacement of the spring required to move the throttle arm at a known distance from the center.

Accuracy was also a major criteria for motor selection. Due to the relative small angle produced as the throttle travels from 0 to max, which is roughly 40 degrees, the motor must have a large resolution to allow such small travel and still maintain accuracy.

In addition, due to the performance ability of GOLIATH and the components that it will have to support, the power supply is limited to a maximum of 24 VDC. Thus the stepper motor must be able to perform all of its required specification with the allotted 24 VDC.

Using these criteria, the team selected the M-Drive 23 stepper motor by Schneider Electric. This motor is shown below in Figure 21.

Figure 21 – Schneider Electric M-Drive 23 Stepper Motor

This motor provides the following specification as they relate to the throttle manipulation:

Holding torque: 1.60 Nm

Resolution: 51,200 steps per revolutions

Speed: Operating limits from 0-4000 RPM’s

Mounting Design

The mounting design for this locomotion component is very simple. The mount will be directly bolted onto the handlebar by two M5 screws. The mount will be placed perpendicular to the throttle assembly and will not affect any factory installed components of GOLIATH. Figure 22 illustrates the mount for the M-Drive motor.

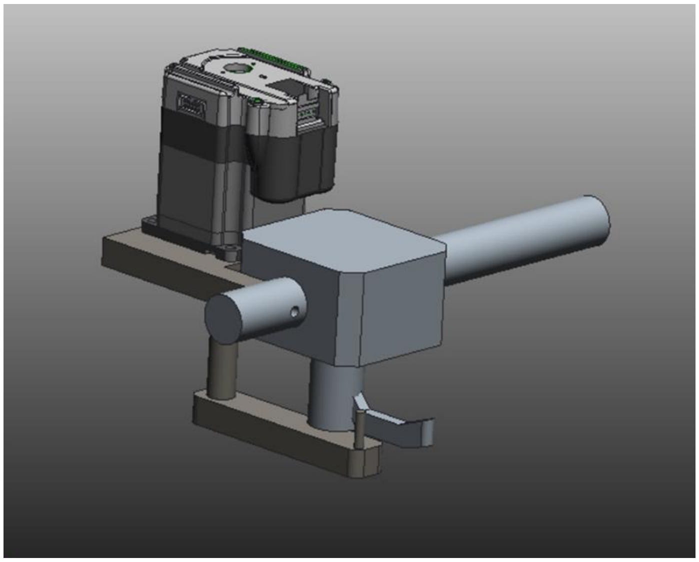


Figure 22 – Mounting for stepper motor. Manufactured parts are shown in dark brown coloring

Current Status

This locomotion control mechanism is finalized and fully tested. This design has met and exceeded all design requirements and is ready for operation. Refer to Figure 23 for an illustration of the final design.

Figure 23 – Fully assembled Gear Select manipulation design

Operation Functional Analysis

A simple functional analysis was used to lay out and explain the full operation of Goliath. After all systems have been initialized and Goliath is mission ready, the user will be able to input commands through the remote control. The remote control will allow the user to input all commands that would be needed as if they were a rider on the system.

The next process is getting the inputs transmitted to the actuators. The remote control will transmit the user inputs wirelessly to the remote computer. Through a wireless router the remote computer will send the signal to the computer onboard of Goliath. Once received by the onboard computer the commands will be sent out the proper actuators. It is also through this private network that the computer onboard Goliath will be able to send data back to the remote computer to update the user on Goliath's current condition.

Since a DC motor controls the steering column, the commands that are sent to it must pass through a motor driver first. In order to reduce the number of ports required the Schneider M-Drive actuators have been Daisy-Chained (connected in series). Each drive has a corresponding name. This way when the signal is transmitted to the individual actuators it will be preceded with the correct name. The user will be able to check the status of operation from feedback from the onboard computer.

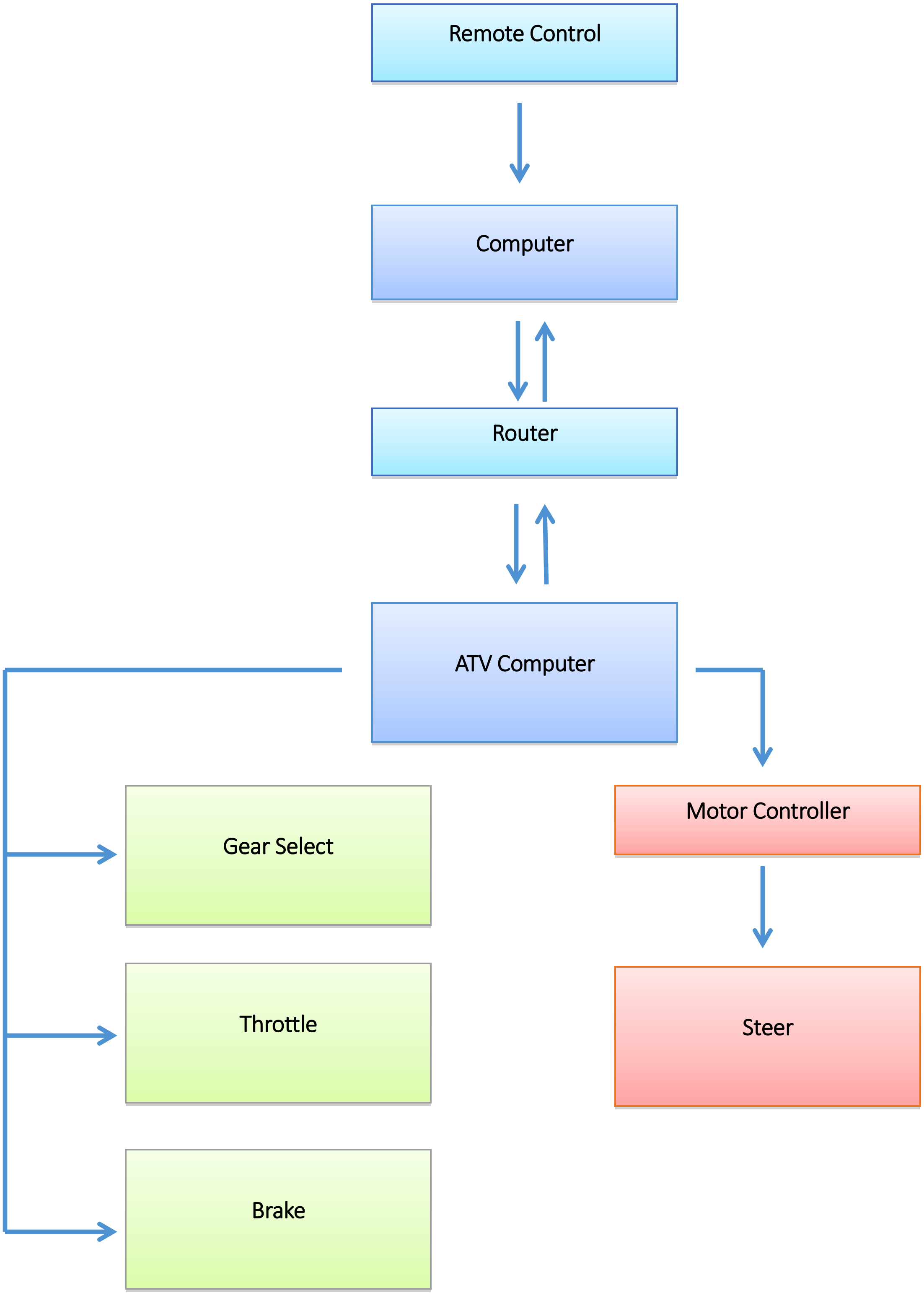


Figure 24 - Goliath Flow Chart

Operating Procedures

1. Connect Toughbook (Server) onboard Goliath to M-Drives and Motor Controller
2. Flip switch to power M-Drives and motor controller
3. Make sure remote control dongle is plugged into base station computer
4. Connect both the server and client to the LAN
5. Run the code blocks program “Goliath Server” on the Toughbook on board Goliath (should respond with waiting for client)
6. After “waiting for client” run the code blocks program “Goliath Client” from the base station computer
7. If “initialization complete” go to step 8, if not the check LAN connection and start again
8. Turn key to the on position, DO NOT FULLY CRANK YET
9. Test actuators to make sure they are responding correctly
10. Once actuators have been tested, make sure they all return to their correct starting positions
11. Crank Goliath
12. Re-test throttle and brake while Goliath is in the park position
13. Engage gear select

Manufacturing Process Selection for Various Components of the Design

Many of the components produced for this project were manufactured using the water-jet available in the machine shop to cut outlines out of aluminum plate. This process was used to cut the plates used to mount all four systems to the vehicle. It was also used for the plates that constrained the actuators and motors to the mounting plates. The water-jet was chosen as the best process to cut these outlines due to the fact that it cut the plate faster than either milling out the pieces or cutting them with a saw. It was also more accurate than using a saw. Using a saw or a mill to cut out the pieces was initially advantageous solely due to the fact that group members were capable of operating these machines, saving time from the overall manufacturing process by eliminating the need to wait for the machine shop to run through other projects higher in the queue. This advantage was negated by the training of one group member, Cesar Mize, in the operation of the water-jet. The final advantage of the water-jet was its ability to drill clearance holes in the part outline in insignificantly more time than cutting the blank outline itself. This saved time spent at the drill press on later pieces.

The linear actuators that are used in the braking and gear shifting systems require precise axial alignment of the threaded rod to operate. Due to the short distance separating the actuator from the point of thrust application in the braking system and the already present necessity for a plastic sliding block, the simple expedient of slotting the mounting plate for the plastic slider to translate along solved the axial misalignment problem. This allowed all the braking system components to utilize lower tolerance methods of manufacture. This was not the case with the gear shifting system which featured a long range of travel and a long threaded rod extended well beyond the actuator. Because of the distances involved, and the very small amount of allowed axial deflection, the mounting plate for the actuator and the bearing block employed to ensure axial linearity required high tolerance manufacturing. So that high tolerances could be achieved, the mounting holes in this piece were manufactured by the CNC machine. Using this method, the distances and relations of the holes could be guaranteed to match the design to a very high tolerance.

The CNC machine would have been capable of milling and drilling all of the components necessary for this project to a high tolerance. This would have not have allowed the project to stay inside the original schedule however due to the long time faced by the CNC process. The CNC machine in the shop is only operated by one employee, which greatly reduces the amount of man-hours per day that go towards CNC production. Also, as none of the group members were proficient in operating a CNC machine, the manufacturing would have been further limited by waiting for other project components higher in the queue to be manufactured first. Finally, for a small number of holes per component, the amount of time actually emplacing and drilling a component is often greater for a CNC machine than a drill press. Because the remaining components of the locomotion systems did not require the high tolerances of the gear shifting system and because members of the group were familiar with the operation of the drill press, the drill press was used for the drilling, milling, and tapping required for all other parts.

Due to machine shop preferences and the desire to make all components easy to remove and reinstall, all welding was avoided during this project.

Mount Material Selection

For selection of mounting material, careful consideration was taken to account for the life span of this project and also the environments in which this project will operate in. We want to maximize the reliability of all aspects of this project, thus we want to select material that will not fail under any operating circumstance. Also, we considered that the material selected might be exposed to corrosive elements such as water or humidity. After careful research, we concluded that stainless steel was the best option. Stainless steel not only is corrosive resistance but is also strong than 2024 aluminum [7]. The following Figure 25 compares multiple metals in a stress vs. strain graph. Clearly stainless steel is the best material for our application.

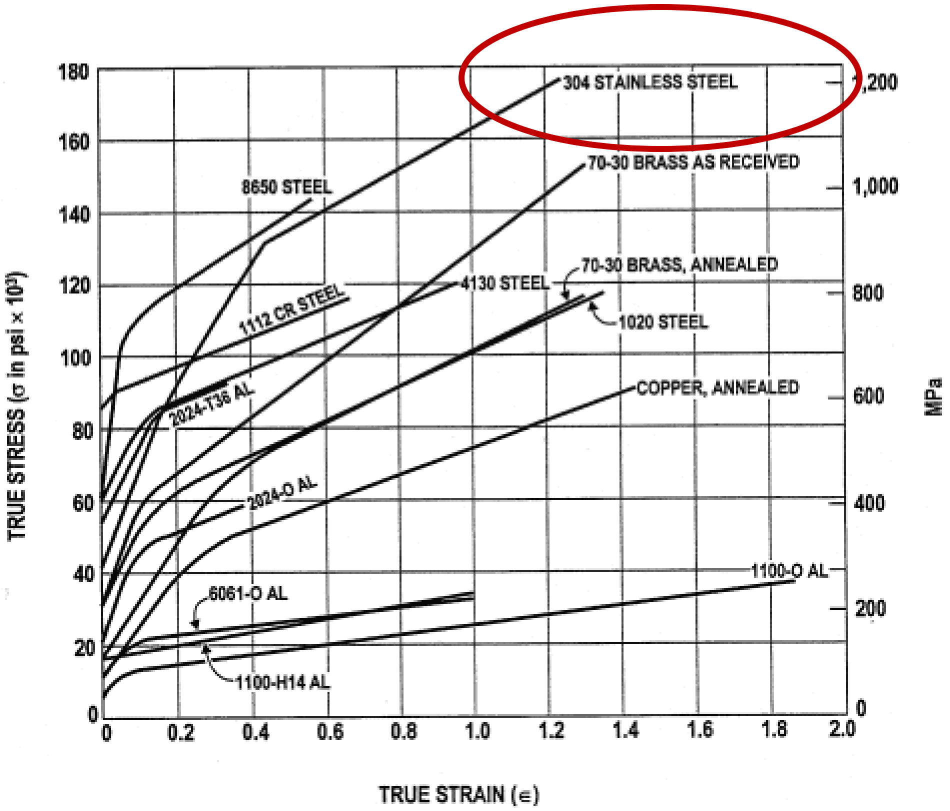


Figure 25 – Stress vs. Strain graph of multiple metals

Design for Reliability

Various components of each locomotion system (steering, throttle, braking, and gear shifting) required special care to ensure reliable performance. The determination of the particular components was obvious in the initial design stages or, at worst, in the initial testing stages. The steering system components that required special care focused primarily on the sprockets and chain used. The concerns were primarily based on loading and corrosion. These concerns were addressed by material selection. Stainless steel was chosen for its high load capability and its resistance to corrosion. The throttle system component that required special care focused on deflection of the mount due to the weight of the stepper motor and the load produced when actuating the system. This concern was remedied with appropriate material selection, proper hardware, and shims to add extra support. 6061 aluminum was used due to its strength in comparison to the loading brought on by the motor weight and actuation force, 1/4-20 bolts were used to secure the mount properly to the handlebars, and rubber shims were used between the motor mount and throttle lever casing to add extra support. The braking system component that required special care involved the nylon slider block that attaches to the motor shaft and is the part of the system that pushes against the brake lever. The concern was about deflection of the motor shaft that would cause inaccurate feedback. This issue was addressed by redesigned the nylon slide and cutting a groove in the brake system mount to allow the nylon slide to move along this path without concern of deflection. The gear shifting system component that required special care involved the manufactured shaft attached between the linear actuator and the gear shift bar. The concern was also about deflection. If the shaft deflected at all, the system could potentially jam. This issue was addressed by extending the linear actuator mount and adding a slider block on the end closer to the gear shift lever. This would allow the manufactured shaft to slide through this block eliminating any possible deflection.

Design for Economics

Because this project is only intended to produce one vehicle, the costs of manufacturing are limited to the individual component, and can ignore mass production considerations. Also, to reduce cost to the project budget, it was determined that all machining would occur in facilities at the school of Engineering so that the project would not be directly billed. To achieve this, the project was designed to be manufactured using relatively simple techniques, making the project members capable of manufacturing almost all the components, which equates to using less skilled machine shop labor.

The raw materials for this project were selected with cost and ease of machinability in mind. To this end, almost all of the components are made of either aluminum or nylon, both of which are simple to machine. Aluminum is also less expensive than other possible metals such as steel and stainless steel. Components were designed to account for standard raw material sizes to reduce purchasing costs, such as the elimination of a rare 6” solid aluminum cube, and the substitution of bolted .5” plates.

To mount the systems to the vehicle, and to assemble the systems themselves, the use of bolts and nuts were preferred over welding. Welding uses more machine shop time, and more experienced labor than the drilling and tapping required for nuts and bolts, which leads to higher costs of manufacture. The nuts and bolts were also selected from standard sizes and lengths.

Cost Analysis

The table below shows the current cost estimate for the completion of this project. The budget for this project is entirely provided by the project sponsor, CISCOR. Our current budget from CISCOR is $9,000. Our final cost estimate is $8,687 which is within our budget. Much of the budget was used to purchase the components to successfully automate the vehicle and provide feedback of the vehicles operation. A smaller portion was used to buy the mounting materials, such as aluminum, and the plastic to prevent axial misalignment of the actuators. The rest of the budget will be retained by CISCOR for use on future projects.

Components	Cost	Quantity	Total Cost
Throttle Stepper Motor	450	1	450
Gear Shift Linear Actuator	670	1	670
Braking Linear Actuator	590	1	590
Steering Motor	2,000	1	2,000
Brake Line Pressure Transducer	225	1	225
Aluminum Stock (Plate and Rods)	120	n/a	120
Mounting Hardware (Bolts, Nuts, etc.)	40	n/a	40
Plastic (For bearing blocks)	80	n/a	80
Driveshaft Encoders	418	4	1672
Encoder Hardware (Pulleys, Belts)	125	4	500
Panasonic Toughbook	200	1	2,000
Logitech Remote Control	45	1	45
Wireless Router	75	1	75
Wiring Components	75	n/a	75
Auxiliary Batteries	40	2	80
Electronic Storage Box	165	1	165
Total Cost of Project			$8,7 87

Table 1 – Budget breakdown

Safety Consideration

Safety is top priority on this project. With a top speed exceeding 60 mph and a weight of 700 pounds, GOLIATH is no small robot. Mechanical fail safes, electrical kill switches and redundancies have been implemented to ensure the safety of working personnel as well as the safety of GOLIATH. With this in mind, a safety supervisor position was created at the conception of this project in early Fall 2012. The safety supervision ensured the safety of the project members during the construction and the operation of the vehicle. The safety coordinator will ensure that non-group members were not endangered during the operation of the vehicle.

The following rules and regulations were implemented from the start of this project:

1. READ WARNINGs before operating vehicle
2. Only authorized personnel can operate vehicle
3. Authorized personnel operating the vehicle must wear appropriate safety equipment
    a. DOT approved helmet
    b. Close toed shoes
    c. Long pants
    d. Gloves
4. While making modification the vehicle must be in the PARK position with the engine OFF and battery UNPLUGGED
5. All flammable liquid must be kept in SAE approved flammable liquid receptacles
6. All toxic liquids must be disposed of in a safe and legal manner
7. No outside night time operation of vehicle

Environmental Consideration

Environmental safety is also a top priority for this group. Before purchasing and acquiring our all-terrain vehicle platform, we ensure that the vehicle complied with both federal and state laws for gas emissions. GOLIATH has a 550 cc gasoline powered engine, and thus releases CO2 gas emissions during operations. However, GOLIATH's manufacture, Polaris claims it has the lowest emissions and green house effects for its class ("Polaris Sportsman 550").

Furthermore, all hazardous liquids are disposed of in safe, dedicated waste receptacles. We follow all local, state and federal laws to ensure proper hazardous waste disposal.

Future Work

While G.O.L.I.A.T.H. is currently operable by remote and considering to be completed according to the scope of the project for the 2012-2013 academic year, there are many ways in which the current vehicle can be improved to be more compatible with intended future uses. Existing components of the vehicle can be optimized to ensure better, more reliable, operations. This includes but not limited to, more rigorous testing of the Schneider actuators for heat dissipation during operation. Testing on the current vehicle has shown that the Schneider components, when operated continuously at larger amperages, like the throttle, tend to heat up. Testing of the amount of heat the actuators reach under operating time periods and loads could determine whether this heat could cause the components to enter a shutdown safe mode. Additionally, testing determined that accidental loss of communication or power to the locomotive components caused issues with the telecommunications control of the components and risked possible damage of the components. This could be remedied through more secure fastenings which ensured the communication and power wires did not become disconnect from the components during any conceivable operations. Finally, the vehicle is envisioned as being an off-road platform, capable of many different terrain types. Currently, the locomotion components are not fully protected from the elements, such as dust and water, which could cause damage. The manufacture and emplacement of weatherproof housings capable of encompassing the components would greatly increase the ruggedness of the vehicle.

During design and testing, it became apparent that some existing components may require alteration or additional hardware to adequately meet future needs of the vehicle. Future versions of this platform will include a much greater amount of processors and other electronics that will need to be stored in the electronic storage box. This will create a greater heat load inside the box, necessitating the installation of a weatherproof system to ventilate the box. The current version of a wireless kill switch used by the remote could be replaced with multiple, dedicated kill switches that duplicated the function of the current wired kill switches located on the vehicle. It was also determined that a system which enable the encoders located on the locomotion components to determine a fixed, constant, reproducible zero point would allow for more reproducible results and better application of safety interlocks.

Much of the needed work to make the platform more responsive and reliable in the future will involve alterations to the telecommunications and coding of the vehicle components. Currently, the vehicle is run through a Windows platform, which leads to relatively long lag times and very sluggish vehicle responses.

Migrating the vehicle to a Linux platform would reduce the sluggishness, while also making the platform more reliable. The programming of the current telecommunications is not robust enough to ensure reproducible results. The command signals generated at the remote are not always received in the expected manner at the vehicle. The vehicle was also observed to sometimes receive benign, but unintended, commands from the remote. The base vehicle naturally senses and reports the RPM's of its motor. The vehicle also includes an electronically controlled power steering unit. Both of these use canned communications that could be cracked to provide sensor feedback concerning the vehicle, and, in the latter's case, provide an alternate method for steering the vehicle.

While this vehicle is intended for use in many future configurations, some additional sensors can be added to the base platform that should prove useful to each configuration at a minimum. A pressure transducer mounted on the brake line leaving the master cylinder could monitor the increase in the brake line pressure, giving a highly accurate measurement concerning the amount of braking force the vehicle is experiencing. Rotary encoders mounted to the four individual drive shafts could provide data about wheel slippage and an accurate vehicle speed. These sensors have already been purchased for the vehicle. Additionally, the encoder mounts, pulleys, and belts have already been purchased and machined. To ensure the final vehicle is able to operate autonomously, the mounting of two SICK laser sensors on the front of the vehicle will be required to identify obstacles in the vehicles path. The addition of an integrated camera on the vehicle will allow better post-test analysis of the vehicles performance and decision making.

Conclusion

Currently, our ATV is fully functional in both unmanned and user modes. All actuators are performing properly and exceeding all requirements. All mounts, brackets, and couplers are fully attached/mounted and are proving to be robust enough to withstand all maximum requirements set by the project goal. The communications system set up for remote control use of the ATV is demonstrating itself to be sufficient while testing of systems continues. The current code which maps all actuator motion onto our Logitech controller has some bugs which need to be dealt with, but overall is proving to also be sufficient while testing continues. As stated in our project objectives, our unmanned ATV is fully able to turn, accelerate, brake, and change gears without physical interaction and its locomotion controls and mounts are durable and able to withstand off-road environments. The ATV still retains its ability to be human operated and all on-board computers can be easily removed and re-mounted.

The end-goal of this project is to provide CISCOR with an autonomous ground vehicle that can withstand off-road environments. Several concepts have been generated to make this platform useful for further research. The current desire is to set up a system that will allow it to scan the terrain in front and determine the best path to take; that is to say the path of least resistance. At certain points in off-road environments, a sufficient distance in front of the ATV cannot be scanned and analyzed. There is currently a proposal to mount a “launch area” on the vehicle from which a quad copter can take off. When situations arise where a sufficient distance ahead cannot be scanned, this quad copter will take off in front of the ATV and scan the terrain and provide feedback to which path should be traversed. A second proposal, which can be used in conjunction with the previous, is a GPS movement system. A user can pinpoint an end goal destination for the ATV and it will traverse a path to that location. The user can also be able to draw out the path on a digital map and the ATV will follow the line, with some autonomous movement around debris and obstructions.

The completed platform also has many capabilities for civilian applications. One possibility for our platform is search and rescue. In 2007, five people went missing within a single hiking season in an area of 180,000 acres in Bankhead National Forest in Alaska [9]. This type of disappearance, especially in Alaska, is extremely common. With multiple platforms similar to ours, autonomously roaming the forests scanning the terrain, this number of disappearances could potentially be zero. Once the missing person(s) are found, they would also be able to mount the vehicle and be taken back to safety. This is especially important if a member of a hiking party is injured in any way. A second possibility for our platform could potentially help hikers, hunters, and other outdoors-men not disappear in the first place. Our platform could have the capability of traverse a large acreage and mapping the terrain. This would not only let hikers know of set trails but also warn of dangerous sections of debris or wet sections of land that should be avoided.

Our ATV platform can also have many capabilities in the military sector. Currently, Israel has autonomous ground vehicles (cars) navigating themselves along the country's borders scanning the area in order to warn military personnel of incoming danger or unwanted intruders [8]. Our platform has the capability of doing the same tasks and producing similar, if not better, results. The ATV has some benefits over using a car because of its size. Its smaller size could allow it to traverse tighter areas that the car may not be able to fit in. The United States military is currently using a Northrop Grumman RQ-8 autonomous helicopter for real-time reconnaissance, surveillance, and battle damage assessment [10]. Our platform has a similar capability of navigating into potentially dangerous situations and getting a closer look from the ground. This would allow the military to have not only a closer view of potential enemy cover but also of IEDs placed along roads and pathways.

References

[1] "DARPA Learning Applied to Ground Robots (LAGR) Project (Concluded)." DARPA Learning Applied to Ground Robots (LAGR). N.p., n.d. Web. 07 Dec. 2012.

[2] Bennett, Brian. "Police Employ Predator Drone Spy Planes on Home Front." *Los Angeles Times*. Los Angeles Times, 10 Dec. 2012. Web. 15 May 2013. <http://articles.latimes.com/2011/dec/10/nation/la-na-drone-arrest-20111211>.

[3] "Global Unmanned Aerial Vehicle Payload Market to Reach $68.6 Billion by 2022." *Global Unmanned Aerial Vehicle Payload Market to Reach $68.6 Billion by 2022*. N.p., n.d. Web. 15 May 2013. <http://www.spacewar.com/reports/Global_Unmanned_Aerial_Vehicle_Payload_Market_to_Reach_68_6_billion_by_2022_999.html>.

[4] *Remote Sensing and Mapping*. N.p., n.d. Web. 15 May 2013. <http://www.isprs.org/proceedings/XXXVII/congress/1_pdf/203.pdf>.

[5] "FAMU-FSU College of Engineering :: CISCOR Center for Intelligent Systems, Control and Robotics." FAMU-FSU College of Engineering :: CISCOR Center for Intelligent Systems, Control and Robotics. N.p., n.d. Web. 05 Dec. 2012.

[6] "Sportsman ® 550." 2013 Polaris Sportsman 550 Sage Green ATV: Features. N.p., n.d. Web. 07 Dec. 2012.

[7] Brandt, Daniel A., and J. C. Warner. Metallurgy Fundamentals. Tinley Park, IL: Goodheart-Willcox, 2009. Print.

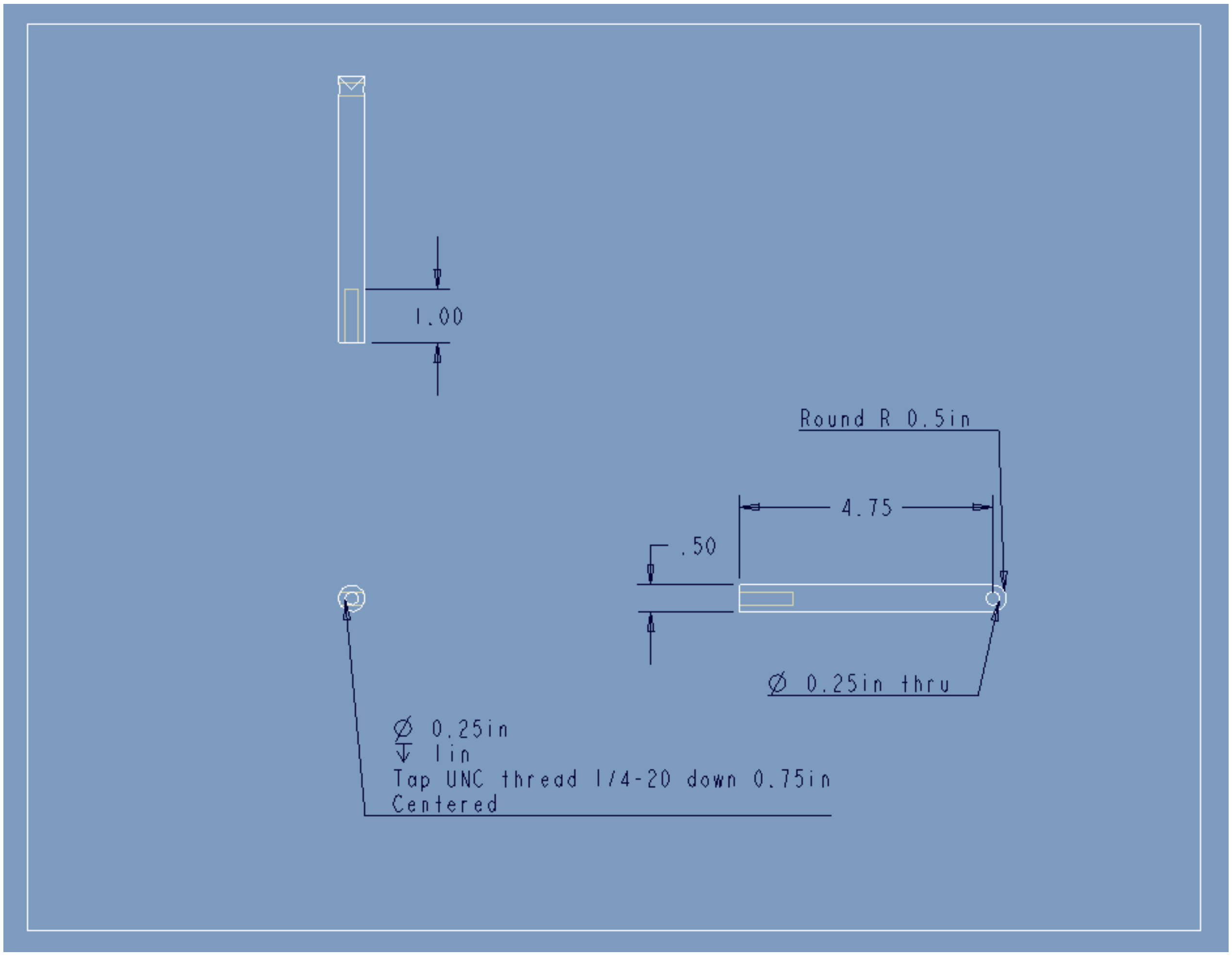
[8] "Autonomous Cars Patrol Israeli Border." *TechNewsDaily.com*. N.p., n.d. Web. 18 Apr. 2013. <http://www.technewsdaily.com/17350-self-driving-cars-israel.html>.

[9] "Lost Hikers." *USATODAY.com*. N.p., n.d. Web. 16 Apr. 2013. <http://usatoday30.usatoday.com/news/nation/2007-09-03-hikers_N.htm>.

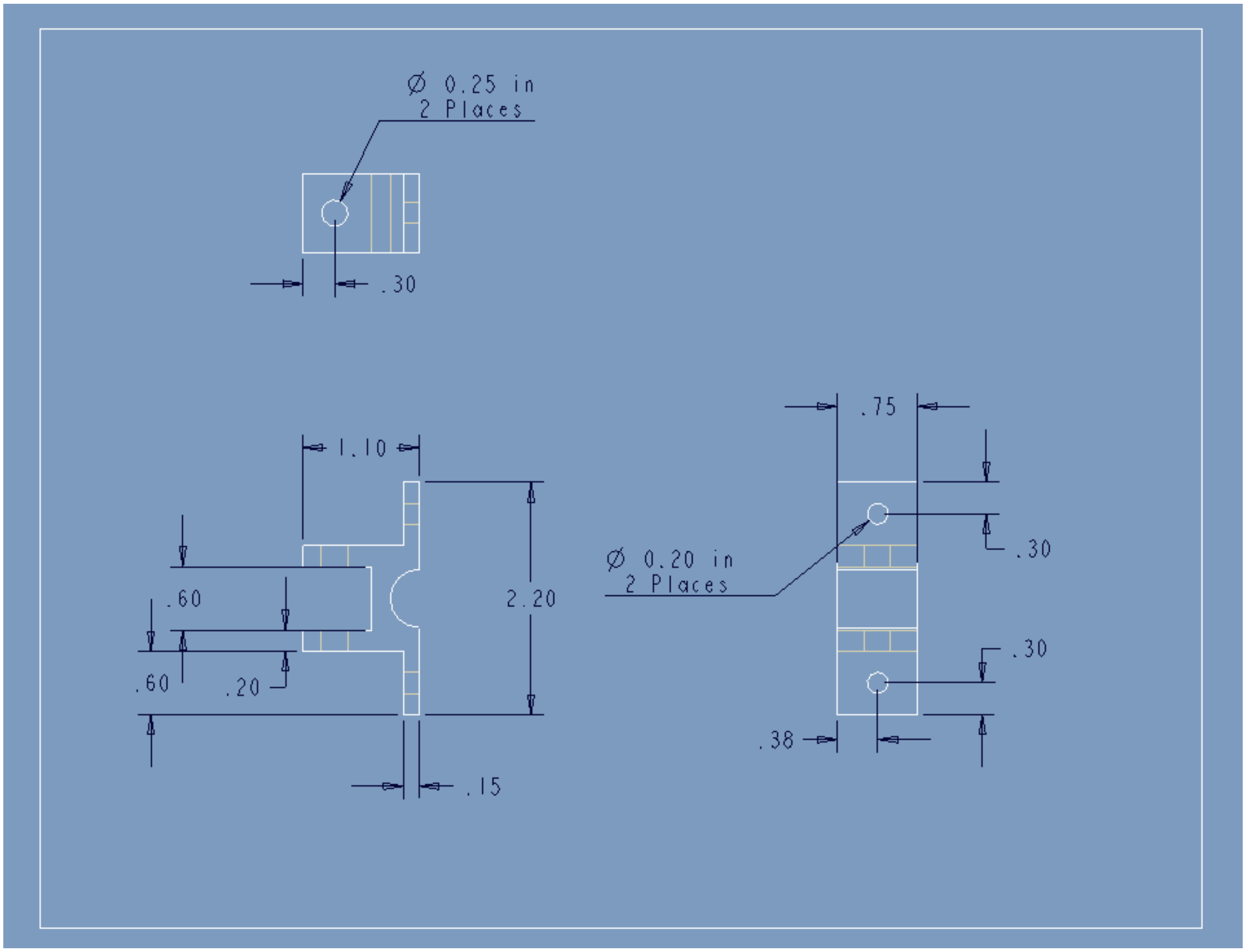
[10] "Northrop Grumman RQ-8/MQ-8 Fire Scout / Sea Scout Unmanned Autonomous Helicopter (2009)." *Northrop Grumman*. N.p., n.d. Web. 18 Apr. 2013. <http://www.militaryfactory.com/aircraft/detail.asp?aircraft_id=889>.

Appendix A – Engineering Drawing of Parts

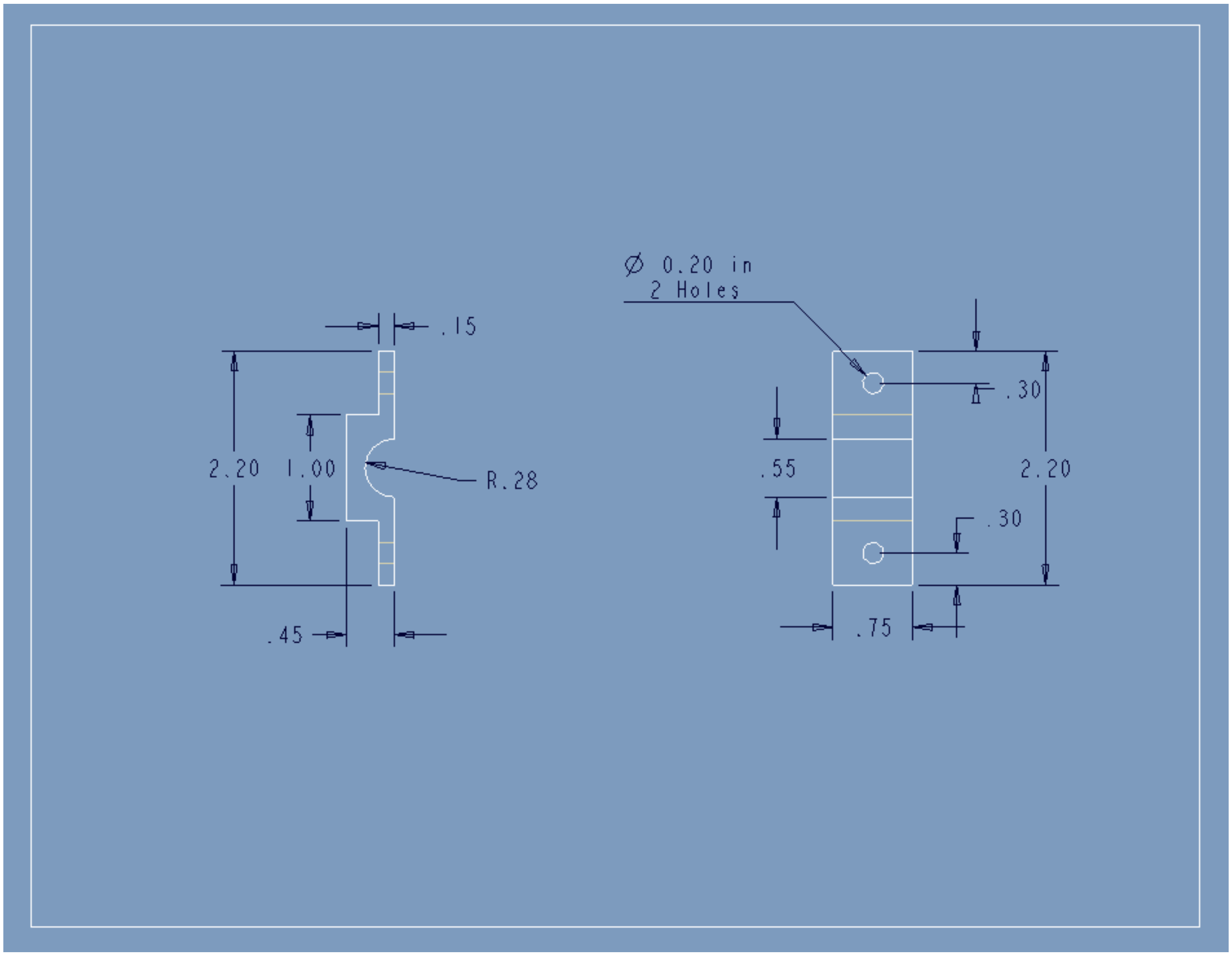
Connector Rod



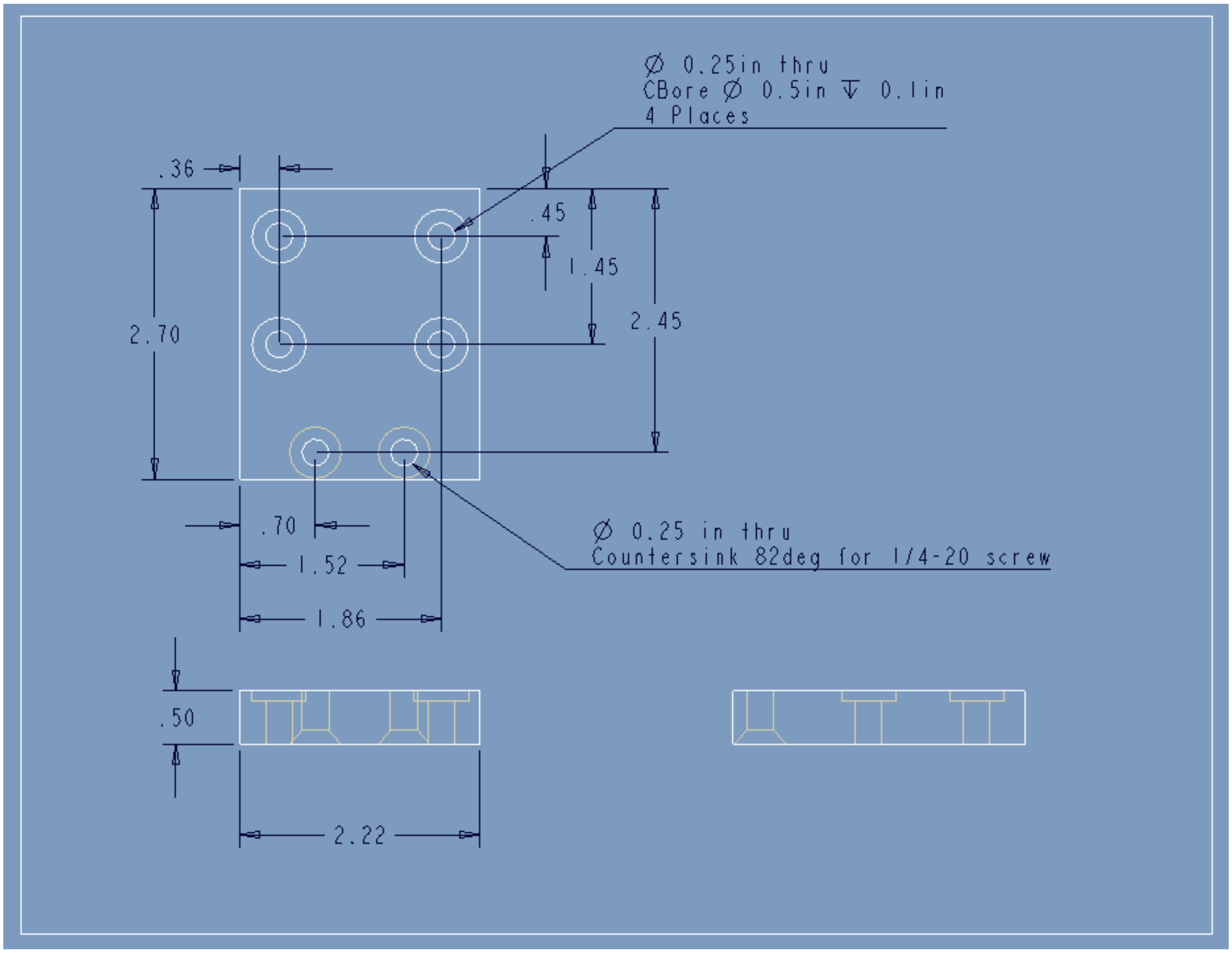
Pin Block


Ø 0.25 in
2 Places
.30
1.10
.60
.60
.20
2.20
.15
.75
Ø 0.20 in
2 Places
.30
.30
.38

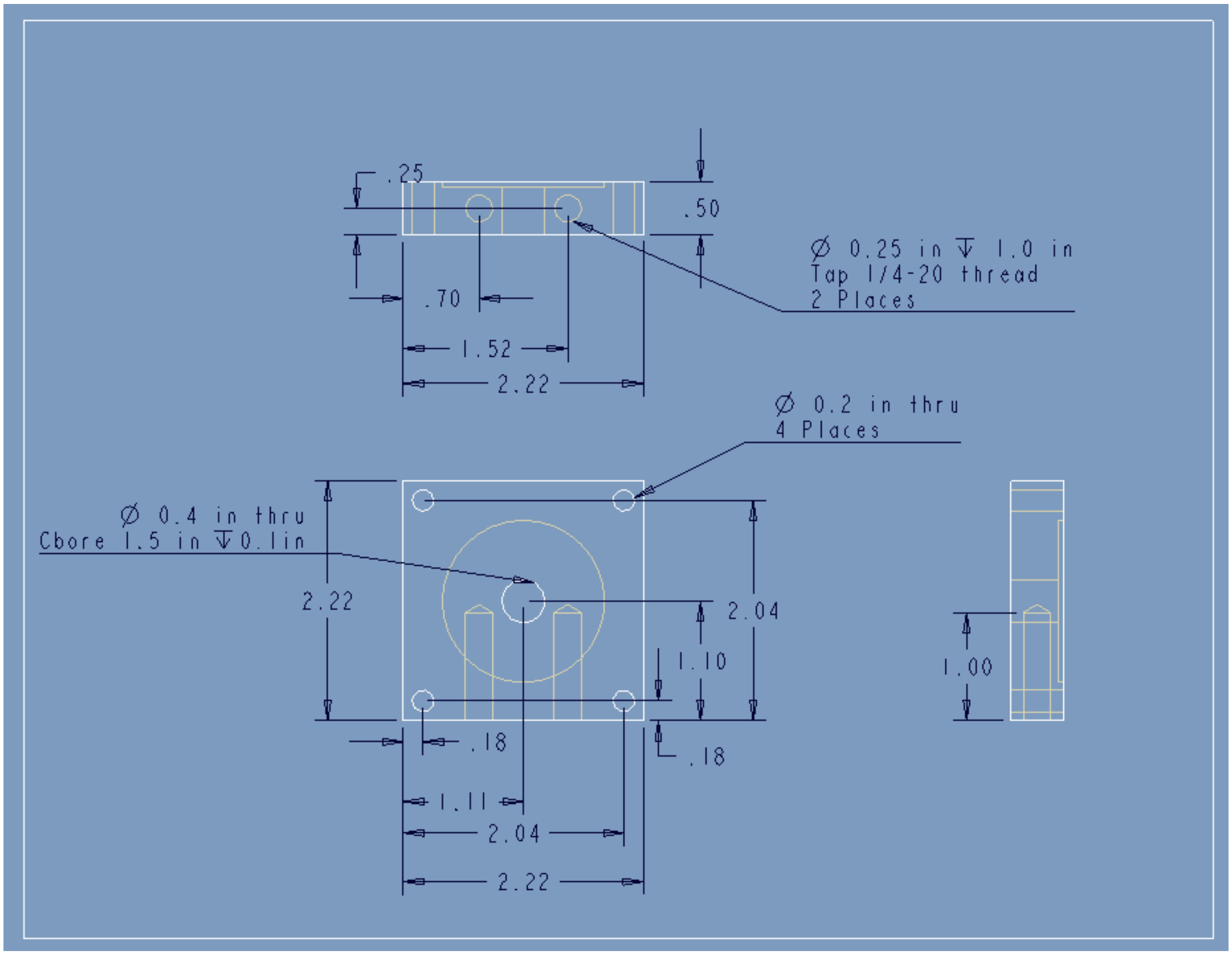
Pin Block 2



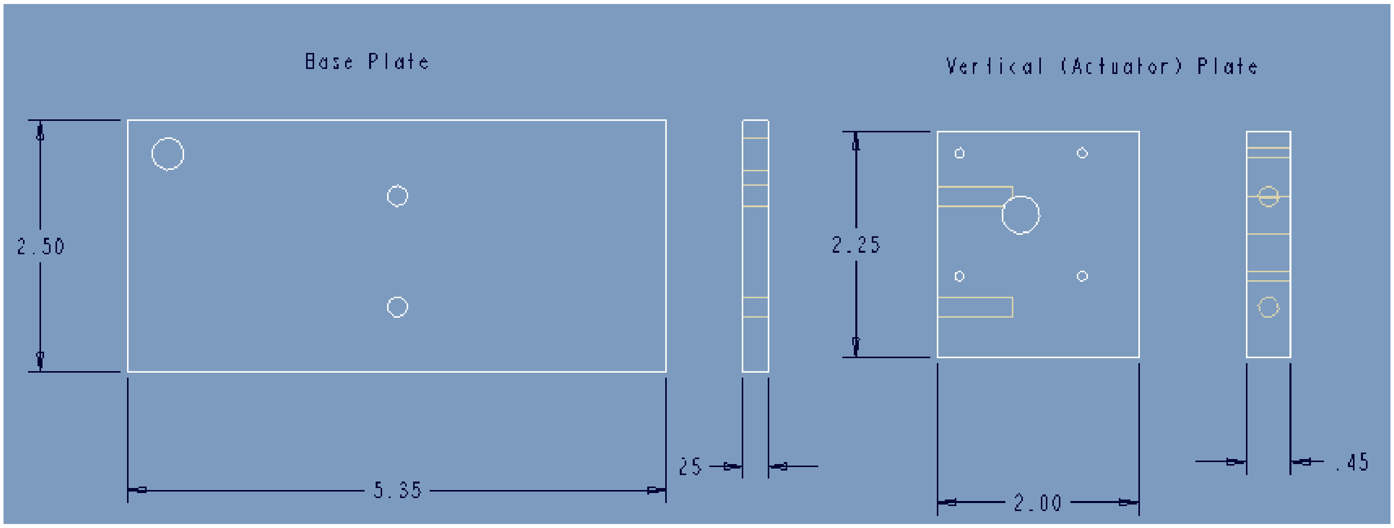
Base Mount


Ø 0.25in thru
CBore Ø 0.5in ↧ 0.1in
4 Places
.36
.45
1.45
2.45
2.70
.70
1.52
1.86
Ø 0.25 in thru
Countersink 82deg for 1/4-20 screw
.50
2.22

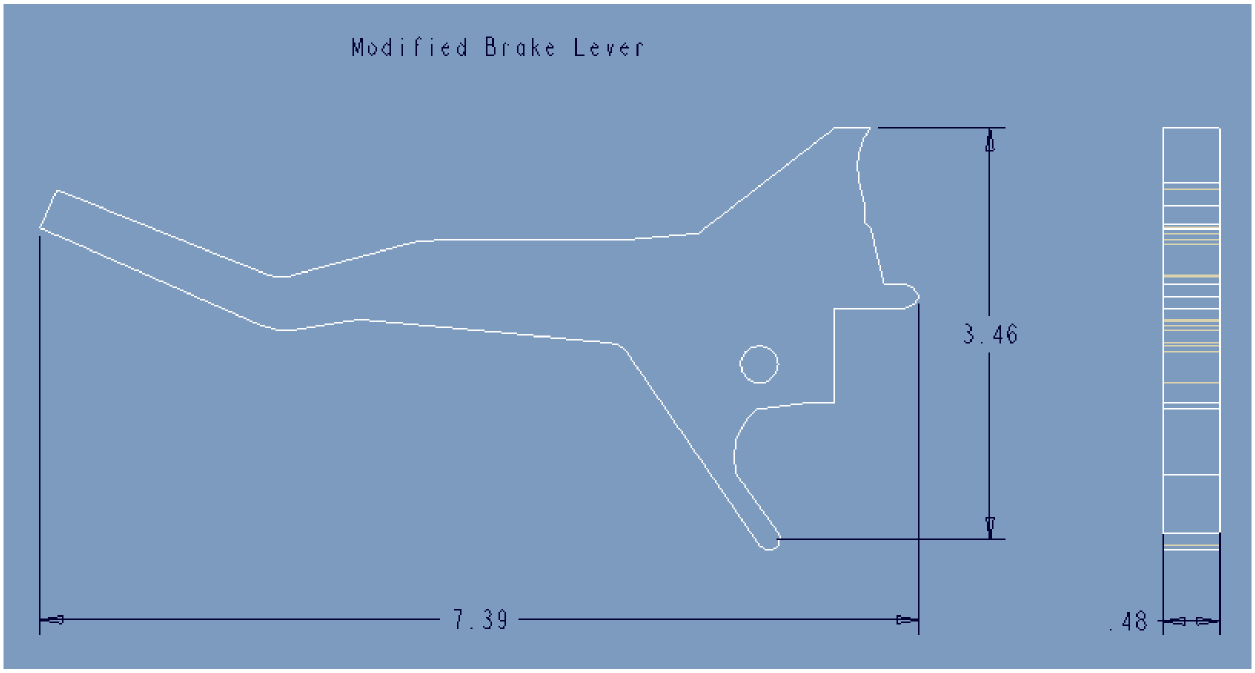
Support Mount



Brake Mounting Plate

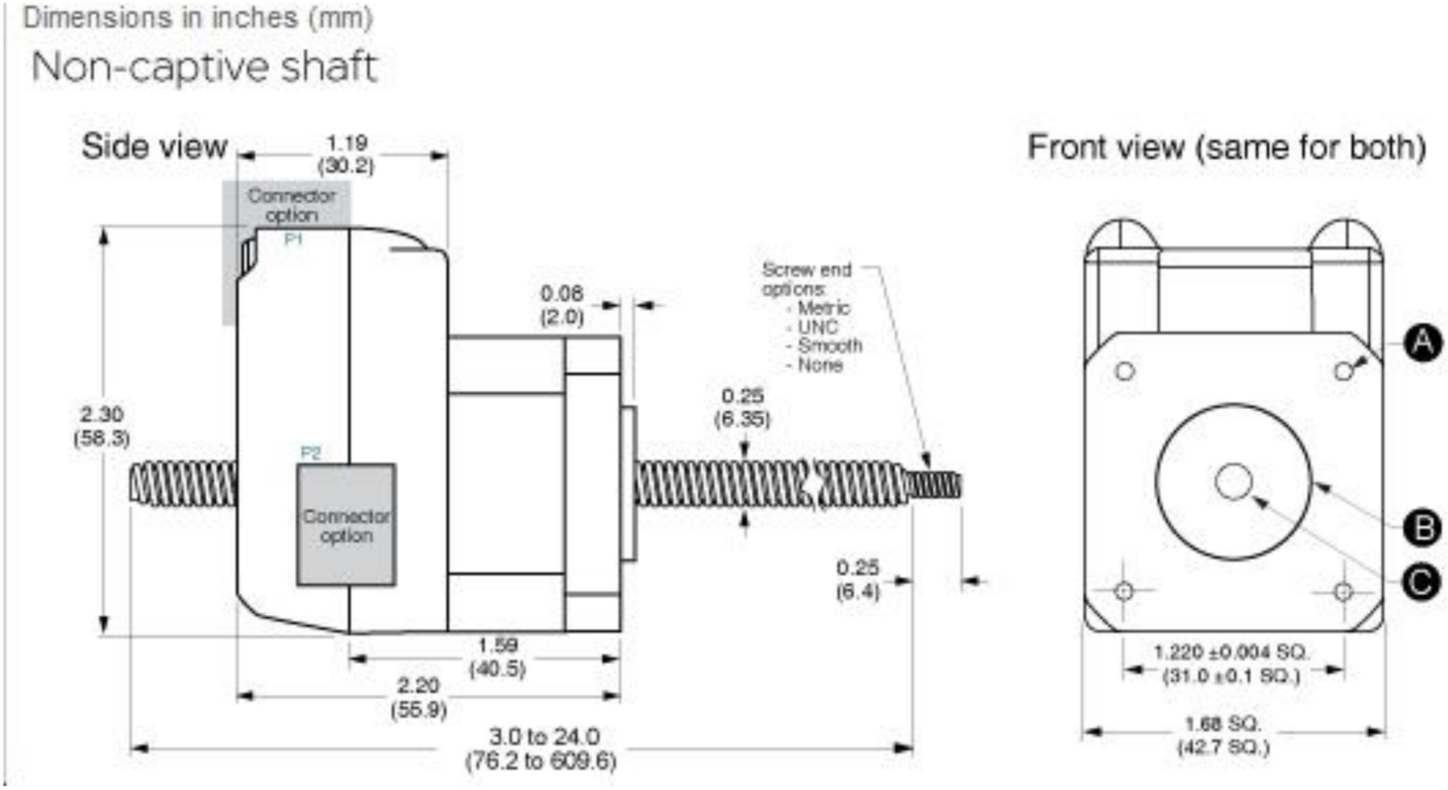


Modified Brake Lever

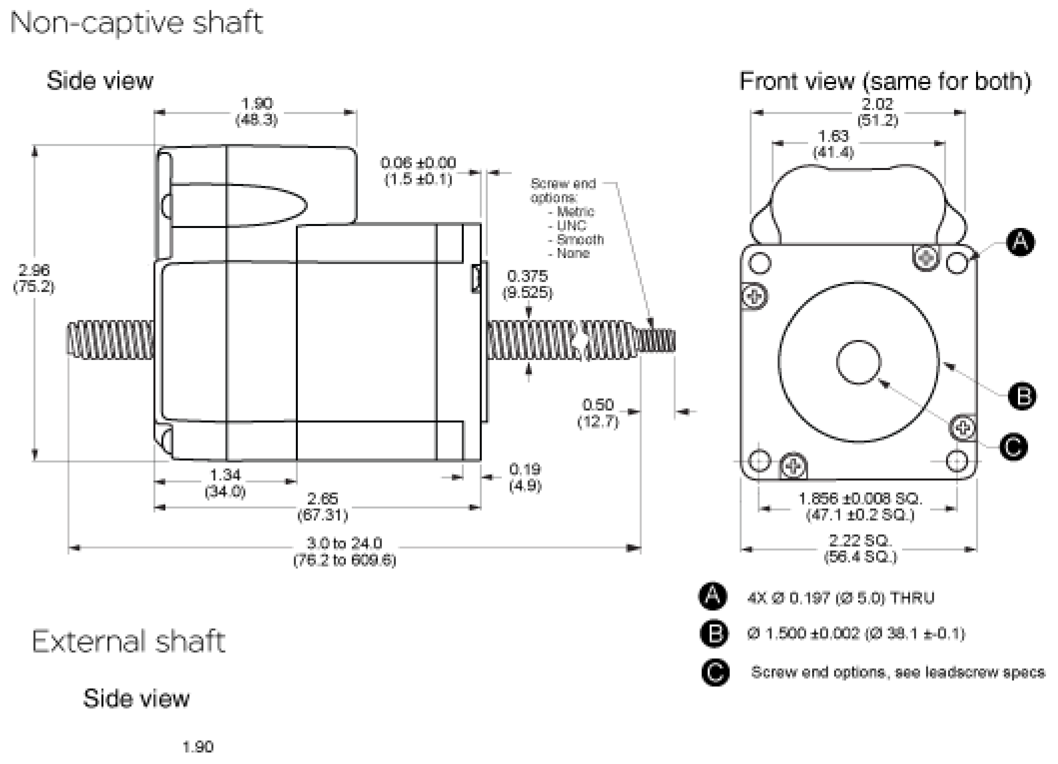


Appendix B – Component Dimensions

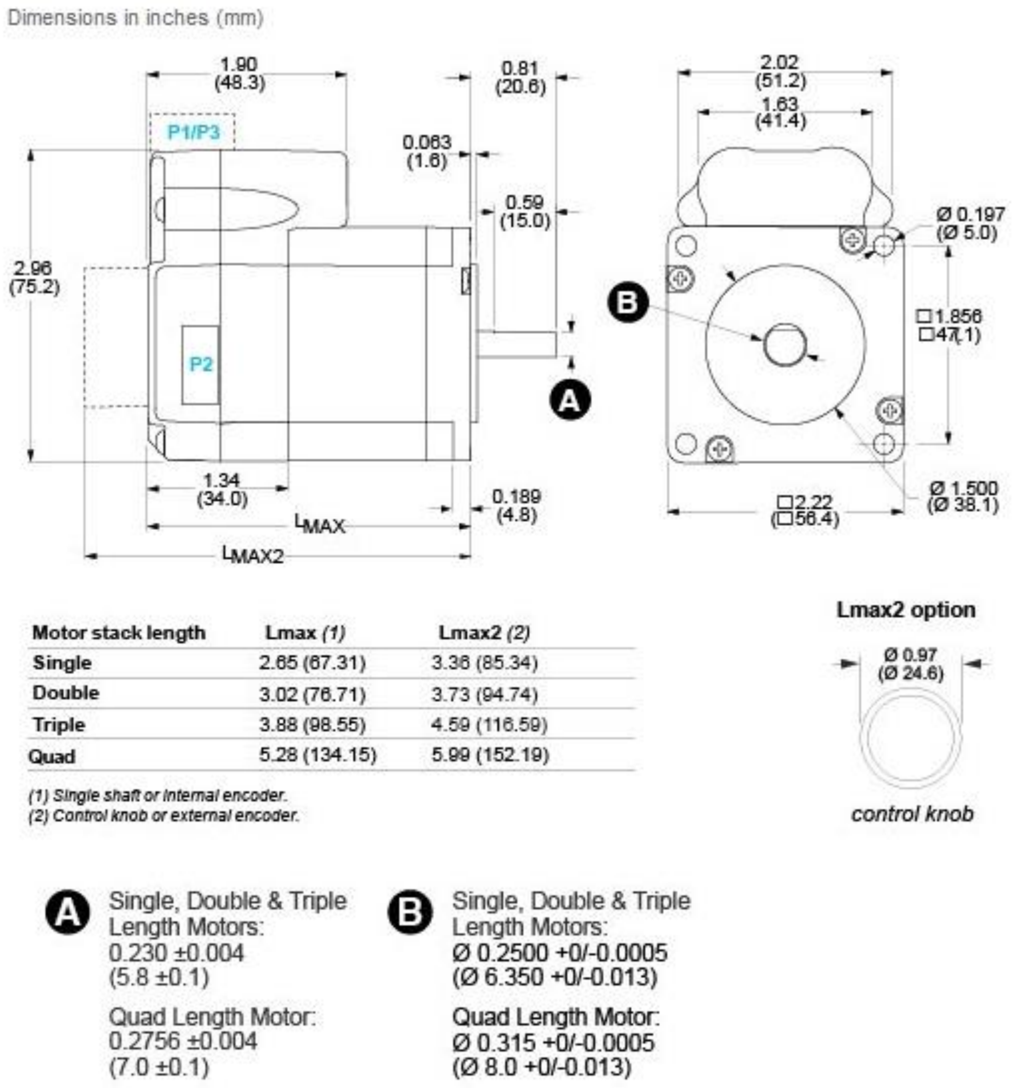
Schneider Electric 17 M-Drive Linear Actuator



Schneider Electric 23 M-Drive Linear Actuator



Schneider Electric 23 M-Drive Stepper Motor



Dimensions in inches (mm)

Motor stack length	Lmax *(1)*	Lmax2 *(2)*
Single	2.65 (67.31)	3.36 (85.34)
Double	3.02 (76.71)	3.73 (94.74)
Triple	3.88 (98.55)	4.59 (116.59)
Quad	5.28 (134.15)	5.99 (152.19)

(1) Single shaft or internal encoder.
(2) Control knob or external encoder.

A Single, Double & Triple Length Motors:
0.230 ±0.004
(5.8 ±0.1)

Quad Length Motor:
0.2756 ±0.004
(7.0 ±0.1)

B Single, Double & Triple Length Motors:
Ø 0.2500 +0/-0.0005
(Ø 6.350 +0/-0.013)

Quad Length Motor:
Ø 0.315 +0/-0.0005
(Ø 8.0 +0/-0.013)

Appendix C – Bill of Materials

Braking System

Label	Part	Material	Qty.
A	Modified Brake Lever	Aluminum	1
B	Mounting Plate	Aluminum	1
C	Actuator Plate	Aluminum	1
D	Handlebar Clamp	Aluminum	1
E	Plastic Slider	Nylon	1
F	Linear Actuator	N/A	1

*Nuts and bolts used to connect components not mentioned in BOM or assembly drawing.

Steering System

Label	Part	Quantity
A	Lower Mount Spacers	2
B	Mount Profile	2
C	Lower Mount Plate	1
D	Front Mount Plate	1
E	Top Mount Plate	1
F	Motor Mount Plate	1
G	Upper Mount Spacers	2
H	Upper Mount Plate	1
I	Motor/Gearbox/Encoder	1
J	Small Sprocket	1
K	#50 ANSI Roller Chain	1
L	Large Sprocket	1
M	Steering Column Coupler	2

*Top mount plate (E) not shown in assembly drawing. Currently not in use on the design.

** Lower mount spacers (A) and upper mount spacers (G) not shown in assembly drawing. Varying thicknesses for proper mounting and fitting.

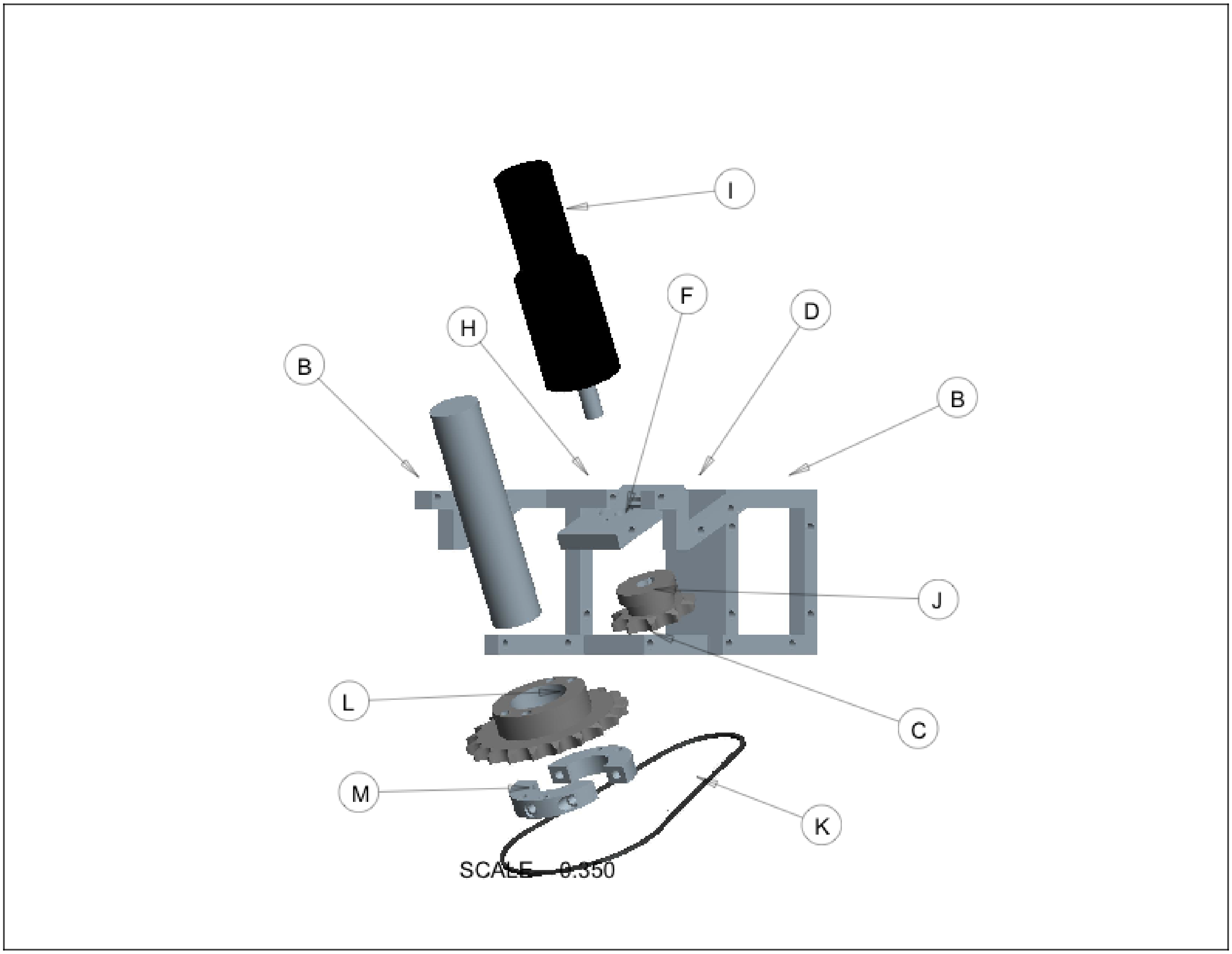
***All hardware not mentioned in BOM or assembly drawing. All hardware needed is mentioned in installation instructions.

Throttle System

Label	Part	Quantity
A	Main Mount Plate	1
B	Shaft Extender	1
C	Lower Throttle Extend Arm	1
D	Throttle Manipulator	1

*Design uses metric screws for all mounting.

Appendix D – Assembly View and Instruction

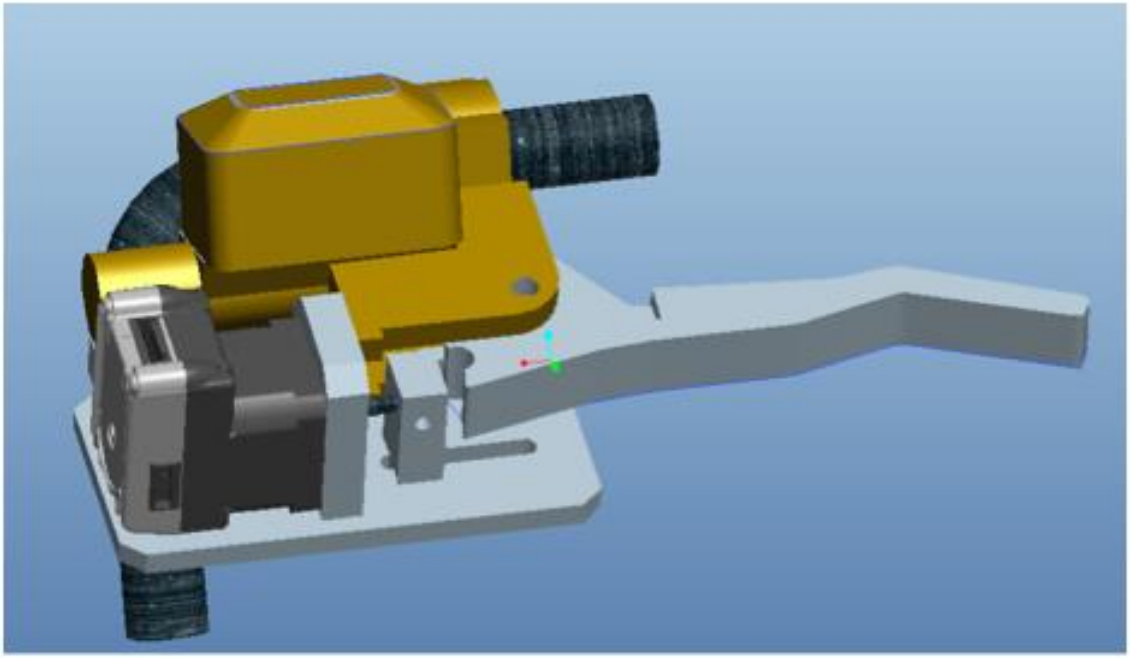


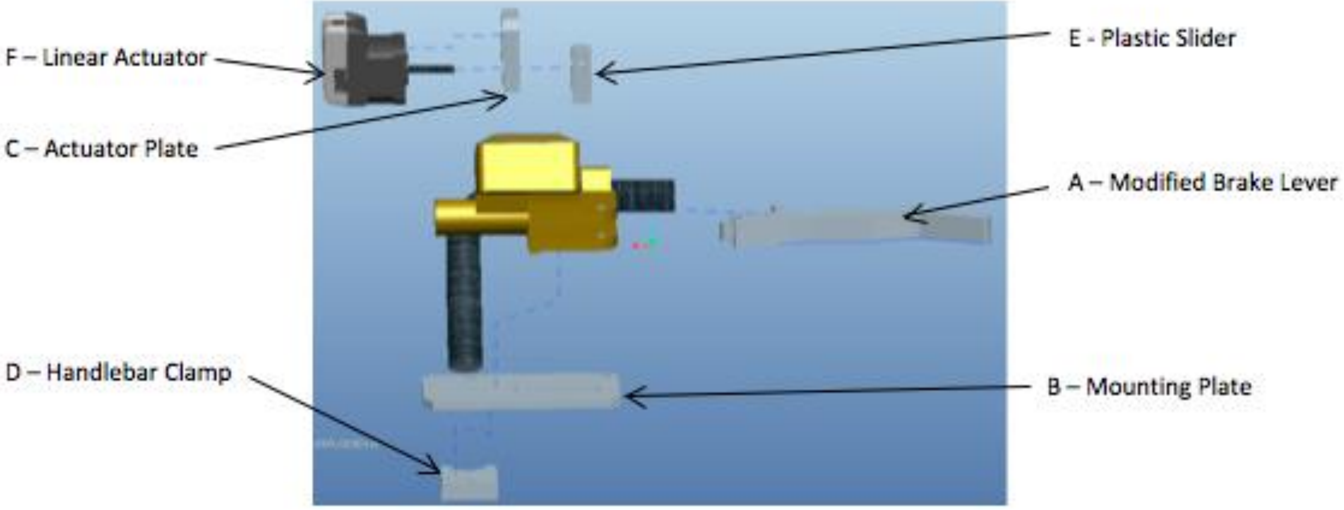
Steering System Assembly Diagram

Steering System Assembly Instructions

1. Remove steering column from ATV by removing both pinch bolts from the bottom of the column near the PSU and the bearing block approximately half way up the column. Pull straight up.
2. Hold large sprocket slightly above PSU and slide steering column down through the large sprocket back onto the PSU input. Reattach bolts mentions in step 1.
3. Take both halves of steering column coupler and attach them on either side of the steering column below the large sprocket. Fasten with 1” M6 bolts.
4. Take a 3” 1/4-20 bolt and slide through first half of coupler, through drilled hole on the steering column, and through other side of coupler. Fasten with nylon insert 1/4-20 nut.
5. Use four 1.5” 1/4-20 bolts and nylon insert 1/4-20 nuts to attach large sprocket to steering column coupler.
6. Away from the ATV, construct the motor mount using the mount profile, lower mount plate, front mount plate, motor mount plate and upper mount plate as seen in the assembly using 0.75” 1/4-20 bolts.
7. Attach motor to motor mount plate as seen in the assembly using M4 screws.
8. Slide small sprocket onto motor shaft below the motor plate and tighten attached set screw.
9. Using the necessary amount of spacers below the lower mount plate and upper mount plate, place mount onto battery tray and upper support bar. Attach the lower mount plate to the battery tray using two 1.5” 1/4-20 bolts with corresponding nylon insert 1/4-20 nuts. Attach upper mount plate to upper support bar using a 1/4-20 nylon insert nut.
10. Wrap #50 ANSI roller chain around both sprockets and link using provided #50 chain connector links.

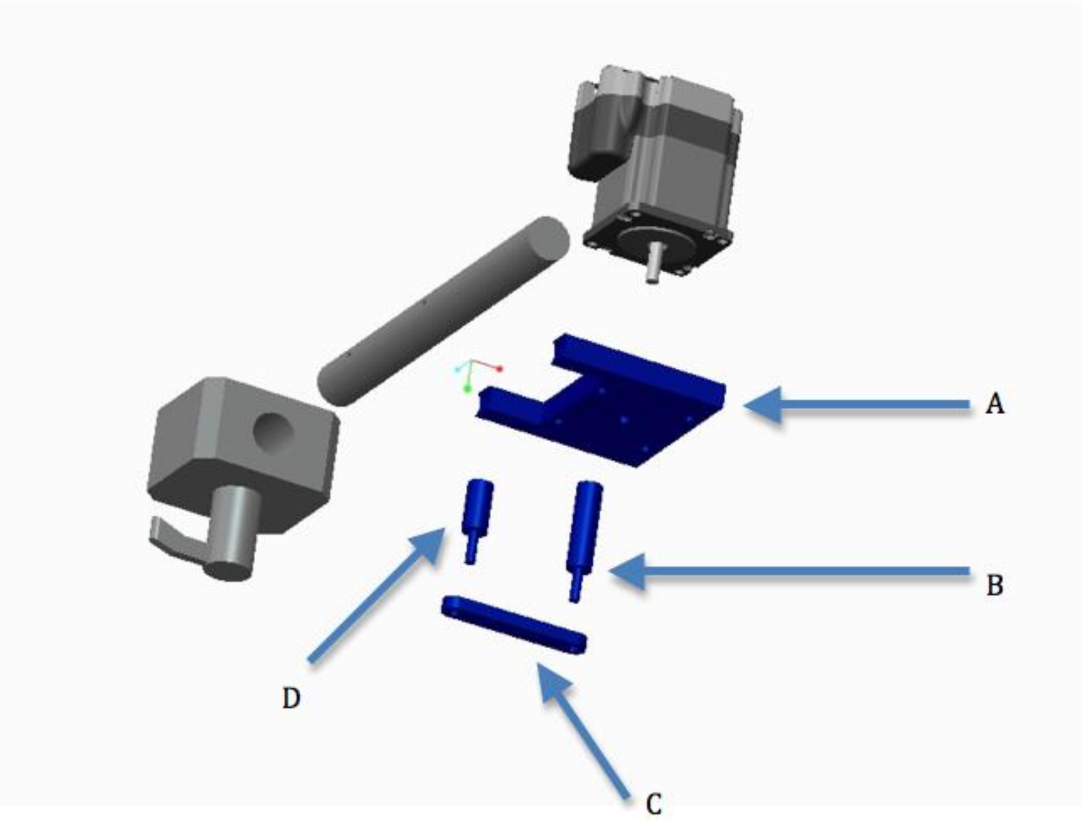
Brake System Assembly Diagram





Braking System Assembly Instructions

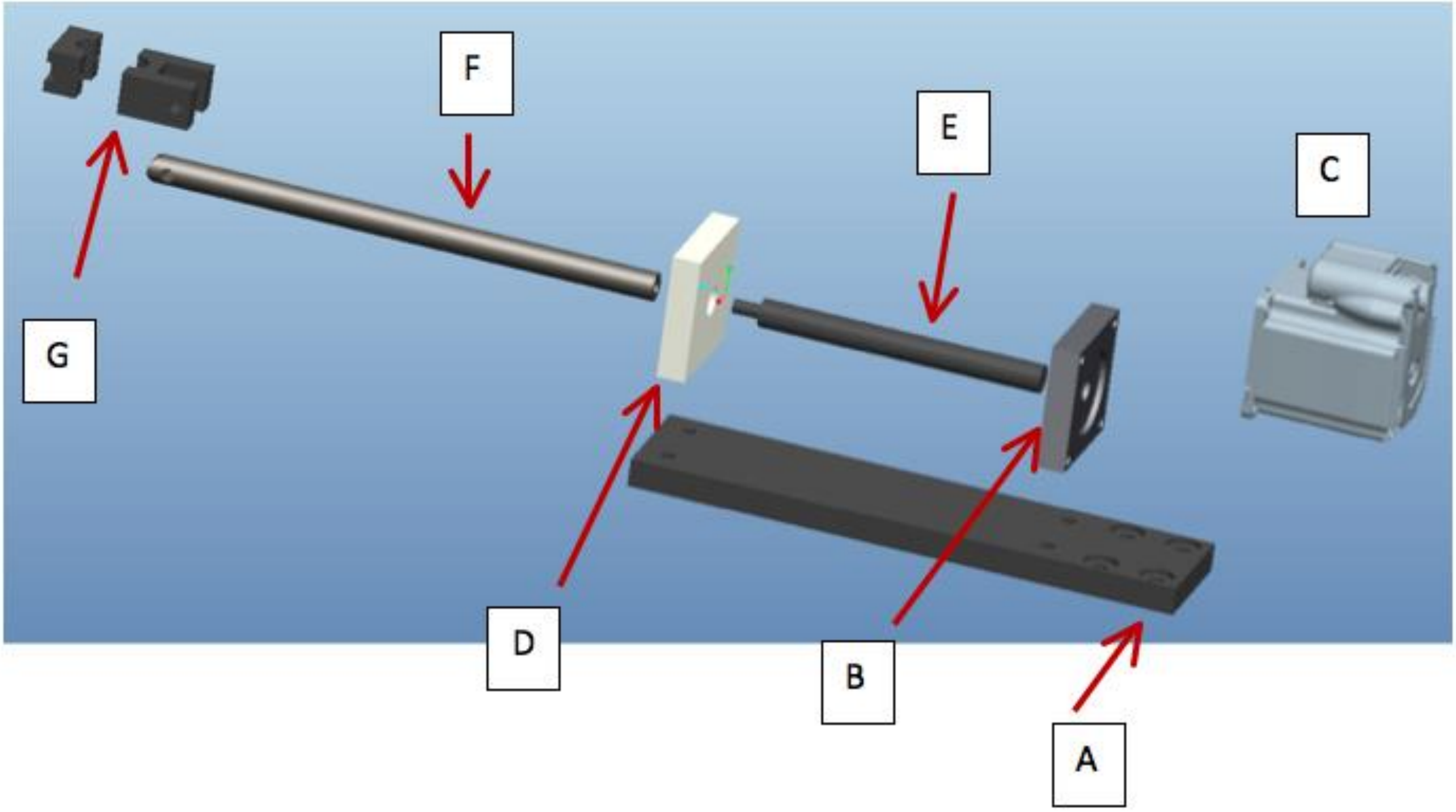
1. Mount the Modified Brake Lever to the existing Brake Master Cylinder.
2. Mount the Handlebar Clamp to the handlebar below the Brake Master Cylinder.
3. Install the Mounting Plate by placing it below the Brake Master Cylinder, and connecting to the Handlebar Clamp which it will rest upon.
4. Mount the Linear Actuator to the Actuator Plate.
5. Mount the Actuator Plate upon the Mounting Plate.
6. Emplace the Plastic Slider in the slot in the Mounting Plate.
7. Thread the threaded rod from the Linear Actuator into the Plastic Slider.
8. Adjust setscrew in Plastic Slider to mate to Actuator threaded rod.



Throttle System Assembly Diagram

Throttle System Assembly Instructions

1. Mount Main Base Plate (A) onto handle with 2x 2’’ M5 screws
2. Mount Schneider M-Drive 23 Actuator to main base plate
3. Attach Shaft Extender (B) to output motor shaft with 2x M1 set screws
4. Attach Lower Mount Plate (C) to Shaft Extender (B)
5. Attach Throttle Contact (D) to Mount Plate (C)



Gear Select System Assembly Diagram

Gear Select System Assembly Instructions

1. Assemble Mount
   - Aluminum Base Plate
   - Plastic Support: ¼-20 with Friction Lock Head Screw (2)
   - Actuator Mount: ¼-20 Pan Head Screw (2)

2. Attach Mount to Goliath
   - Place mount on top on Goliath arm and plastic on bottom
     - ¼-20 Button Head Screws (4)
     - ¼-20 Friction Lock Nut (4)
   - Make sure that mount is aligned with shift arm

3. Attach Actuator to Mount
   - M-5 Thread Screws (4)
   - M-5 Nut (4)

4. Install Connector Link
   - Thread M-Drive Screw into M-Drive Actuator
   - Thread ¼-20 nut onto end of thread
   - Screw Black connector link onto end of actuator screw and tighten with nut

5. Install Connector Joint
   - ¼-20 Black Hex Head Screw

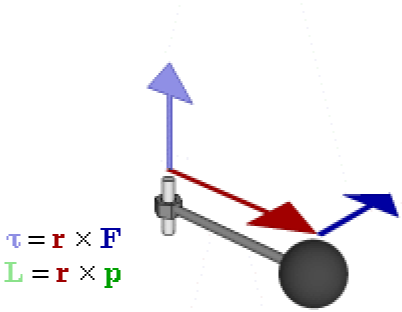
6. Connect connector pin to connector link
   - ¼ in pin/screw

7. Connect Actuator to system
   - Connect parallel communication system
   - Connect to power

Appendix E – Equations

Equations 1:

Torque = Force * Perpendicular Distance



```
//Appendix F

//GOLIATH CLIENT CODE
//C Programing

/*
    Author: Ryan - David Reyes
*/
#include <iostream>
#include <cmath>
#include "TCPIPWIN32.H"
#include "GamepadWin32.h"

const int LSTICK_DEFAULT = 32767;
const int LSTICK_DEADZONE = 200;
const int LSTICK_MAX = 65535;

const int TRIGGER_DEFAULT = 32767;
const int TRIGGER_MIN = 134;
const int TRIGGER_MAX = 65401;

const int START_BUTTON = 0x80;

const int RIGHT_BUMPER = 0x20;
const int LEFT_BUMPER = 0x10;

//////////////////////////////MOTOR VALUES///////////////////////////
const int MAX_STEER = 20000; //Maximum absolute position for the steering:
(-MAX_STEER TO MAX_STEER)

const int MAX_THROTTLE = 100; //Maximum absolute position for the throttle
range :(0 TO MAX_THROTTLE)
const int MAX_BRAKE = 6656; //Maximum absolute position for the brake
range :(0 TO MAX_BRAKE)

const int GEAR_POSITIONS[5] = {0, -2048*8, -2048*13, -2048*17, -2048*25};
//Gear positions for park, reverse, neutral, low, high, in that order.
enum gearPos {PARK, REVERSE, NEUTRAL, LOW, HIGH};
//////////////////////////////////////////////////////////////
const char STEERMSG = 'S';
const char GEARMSG = 'G';
const char THROTMSG = 'T';
const char BRAKEMSG = 'B';
const char STOPMSG = 'X';

const int PORT = 65534;
const char * ADDRESS = "192.168.11.26";

using namespace std;

bool initClient(TCP & tcpConnection, Gamepad &pad1);
void sendData(TCP & tcpConnection, char messageType, int data);

int main()
```

```
{
      int LStickValueX =0, buttonState = 0;
      int triggerAxis = 0, throttleVal = 0, brakeVal = 0, steerVal;

      int gearSetting = PARK;
      int gearVal = 0;

      bool running = true;
      char buttonPressed;
      //char buffer[50];

      Gamepad pad1;
      TCP tcpConnection;

      if(! initClient(tcpConnection, pad1)) //if the initialization fails
            return 1;

      ////////////////////MAIN
LOOP/////////////////////////////////////////////////
      while (running)
      {
            pad1.CheckGamepad();

            ////////////////////CHECK THE X
AXIS//////////////////////////////////////////
            if (LStickValueX != pad1.getX())
            {
                  LStickValueX = pad1.getX();

                  //if the left stick value is out of the deadzone
                  if ( abs(LStickValueX - LSTICK_DEFAULT) >
LSTICK_DEADZONE)
                        steerVal = (LStickValueX - LSTICK_DEFAULT) *
static_cast<double> (MAX_STEER) / (LSTICK_MAX - LSTICK_DEFAULT);
                  else
                        steerVal = 0;

                  sendData(tcpConnection, STEERMSG, steerVal);
                  //cout << "      " << LStickValueX << "     " << steerVal
<< endl;
            }

            ////////////////////CHECK THE
TRIGGERS/////////////////////////////////////////

            if (triggerAxis !=  pad1.getZ())
            {
                  triggerAxis = pad1.getZ();

                  if(triggerAxis > TRIGGER_DEFAULT) //if the left trigger
is pressed
                  {
                        //conversion
```

```
                       brakeVal = (triggerAxis - TRIGGER_DEFAULT) *
static_cast<double> (MAX_BRAKE)/(TRIGGER_MAX-TRIGGER_DEFAULT);
                       sendData(tcpConnection, BRAKEMSG, brakeVal);
                 }

                 else if (triggerAxis < TRIGGER_DEFAULT) //if the right
trigger is pressed
                 {
                       //conversion
                       throttleVal = (triggerAxis - TRIGGER_DEFAULT) *
static_cast<double> (MAX_THROTTLE)/(TRIGGER_MIN-TRIGGER_DEFAULT);
                       sendData(tcpConnection, THROTMSG, throttleVal);
                 }

                 else  //else either both triggers are pressed or not
pressed
                 {
                       throttleVal = 0;
                       brakeVal = 0;
                       sendData(tcpConnection, BRAKEMSG, brakeVal);
                       sendData(tcpConnection, THROTMSG, throttleVal);
                 }

                 cout << "brakeVal: " << brakeVal;
                 cout << "\tthrottleVal: " << throttleVal << endl;
                 //I separate the read/writes so that there's less lag and
less information
                 //needing to be sent
                 //cout << "trigger: " << triggerAxis << endl;
           }

     ///////////////////////////////////////////////////////////////////
///////////
           ///////////////////////CHECK THE
BUTTONS////////////////////////////////
           if (buttonState != pad1.getButtons())
           {
                 buttonState = pad1.getButtons();

                 if (pad1.getButtons() & START_BUTTON)
                 {
                       buttonPressed = STOPMSG;
                       cout << "Start Pressed!" << endl;
                       running = false;
                       sendData(tcpConnection, buttonPressed,
START_BUTTON);
                 }

                 //if the bumber buttons are pressed
                 if (  ( (pad1.getButtons() & RIGHT_BUMPER) ||
                       (pad1.getButtons() & LEFT_BUMPER) )  &&
                       (throttleVal == 0) &&
                (brakeVal > 0) )              //can only shift gears when
the throttle is zero and brake is on
```

```
                  {
                        buttonPressed = GEARMSG;

                        if ((gearSetting < HIGH) && (pad1.getButtons() &
RIGHT_BUMPER))
                              gearSetting++;

                        else if ((gearSetting > PARK) && (pad1.getButtons()
& LEFT_BUMPER))
                              gearSetting --;

                        gearVal = GEAR_POSITIONS[gearSetting];
                        sendData(tcpConnection, buttonPressed, gearVal);
                  }
            }


      ////////////////////////////////////////////////////////////////////
///////////////////////////////////////
            /*
               cin >> messageType >> value;
               cout << messageType << "\t" << value << endl;
               tcpConnection.sendData(tcpConnection.getSocket(), (char *)
&messageType, sizeof(messageType));
               tcpConnection.sendData(tcpConnection.getSocket(), (char *)
&value, sizeof(value));
               */
      }

      cout << "Terminating.." << endl;
      return 0;
}

bool initClient(TCP & tcpConnection, Gamepad &pad1)
{
      if (pad1.InitGamepad() == 0)
            cout << "Gamepad Initialized!" << endl;
      else
      {
            cout << "Gamepad Not Found!" << endl;
            return false;
      }

      if (tcpConnection.connectToHost(PORT, ADDRESS))
      {
            cout << "Connection success!"<< endl;
            return true;
      }
      else
      {
            cout << "Connection Failed!" << endl;
            return false;
      }
}
```

```
void sendData(TCP & tcpConnection, char messageType, int data)
{
      tcpConnection.sendData(tcpConnection.getSocket(), (char *)
&messageType, sizeof(messageType));
      tcpConnection.sendData(tcpConnection.getSocket(), (char *) &data,
sizeof(data));
}
```

```
//Appendix G

//GOLIATH SERVER CODE
//C Programing

/*

   Author: Ryan - David Reyes

   Co-Author: Dr. Oscar Chuy (maxonDriver.h)

*/

#include <iostream>

#include <cstdlib>

#include <string>


#define STEERING_MOTOR

#define RUN_MOTORS


#include "TCPIPWIN32.h"

#include "SerialCom.h"


#ifdef STEERING_MOTOR

   #include "maxonDriver.h"

#endif // STEERING_MOTOR



using namespace std;


const int PORT = 65534;


char * SERIALPORT = "com6";

const string PARITY = "9600";
```

```
const char CTRL_J = 10;
const char SPACE = ' ';
char MA[2] = {'m','a'}; //move absolute

const int STEERING_CENTER = 0;

void initConnections(SerialCom & serialConnection, TCP & tcpConnection, SOCKET & clientSock);
void motorControl(SerialCom &connection, char messageType, int value);
void resetMotors(SerialCom &connection);

int main()
{
   int steerVal = 0, throttleVal = 0, brakeVal = 0, gearVal = 0, tempVal;
   bool running = true;
   char messageType;

   SOCKET clientSock;  //client socket for TCPIP connection
   TCP tcpConnection;
   SerialCom serialConnection;

//INITIALIZATIONS:
   //Initialize the connections:
   initConnections(serialConnection, tcpConnection, clientSock);

#ifdef STEERING_MOTOR
   //Initialize the steering motor:
   Init_Maxon_Motor_Driver();

   if (ftStatus != FT_OK)
```

```
    {
        cout << "Steering Motor Failure!" << endl;
        return 1;
    }

    Enable_Maxon_Motor_Driver();
    //Set_Traj_Params();
#endif

/////////////////////////////////////////////////////////////////////////

    while (running)
    {
        tcpConnection.recieveData(clientSock, (char *) &messageType, sizeof(messageType));

        //Checks to see if the client has disconnected
        if (tcpConnection.recieveData(clientSock, (char *) &tempVal, sizeof(tempVal)) ==-1 )
        {
            running = false;
            cout << "Client Disconnected!" << endl;
        }

        switch (messageType)
        {
            case 'G':
                gearVal = tempVal;
                break;

            case 'T':
```

```
        throttleVal = tempVal;
        break;

    case 'B':
        brakeVal = tempVal;
        break;

    case 'S':
        steerVal = tempVal;
        break;

    case 'X':
        running = false;
        break;

    default:
        running = false;
        break;
}

cout << "Msg: "     << messageType  << "\tThrottle: "   << throttleVal;
cout << "\tGear: "  << gearVal      << "\tBrake: "      << brakeVal;
cout << "\tSteering: " << steerVal << endl;



if (running)    //make sure we don't write garbage data to the motors.
{
    #ifdef STEERING_MOTOR
        if (messageType == 'S')
```

```
                Move_Motor_Abs(tempVal);
            else
        #endif

        #ifndef STEERING_MOTOR
            if (messageType != 'S')
        #endif
                motorControl(serialConnection, messageType, tempVal);
    }

}

cout << "Resetting motors..." << endl;

#ifdef STEERING_MOTOR
Close_Maxon_Motor_Driver();
#endif

resetMotors(serialConnection);
tcpConnection.closeSocket(clientSock);

cout << "Terminating..." << endl;
return 0;
}

void initConnections(SerialCom & serialConnection, TCP & tcpConnection, SOCKET & clientSock)
{
    cout << "Waiting for client..." << endl;
```

```
    tcpConnection.listenToPort(PORT);
    clientSock = tcpConnection.acceptConnection();

    if(clientSock != SOCKET_ERROR)
        cout << "Connection Accepted!" << endl;

    else
    {
        cout << "Accept Failed!" << endl;
        tcpConnection.closeSocket(clientSock);
        exit(EXIT_FAILURE);
    }
    #ifdef RUN_MOTORS
    serialConnection.init(SERIALPORT, PARITY, 0,0,8);
    #endif
}
void motorControl(SerialCom &connection, char messageType, int value)
{
    #ifdef RUN_MOTORS
    char motorvalue[10] = {0};
    itoa(value, motorvalue, 10);

    connection.SendData(CTRL_J, 1);
    connection.SendData(messageType, 1);

    for (int i = 0; i < 2 ; i++)
        connection.SendData(MA[i], 1);

    connection.SendData(SPACE, 1);
```

```
    for(int i = 0; motorvalue[i] != '\0'; i++)
        connection.SendData(motorvalue[i], 1);

    connection.SendData(CTRL_J, 1);
    #endif
}

void resetMotors(SerialCom &connection)
{
    #ifdef RUN_MOTORS
    motorControl(connection, 'G', 0);
    motorControl(connection, 'T', 0);
    motorControl(connection, 'B', 0);
    #ifdef STEERING_MOTOR
    Move_Motor_Abs(STEERING_CENTER);
    #endif
    #endif
}
```