DrivePods: A modular, steerable drivetrain for Gladiator

Vaishal Patel and Vikas Reddy

MAE 429

Prof. Garcia

Table of Contents

Executive Summary

A new drivetrain was designed from scratch for Gladiator. Gladiator is an all terrain humanitarian landmine detection robot designed by the student team Cornell Minesweeper. The drivetrain is responsible for providing locomotion for the Gladiator. It takes the rotary motion from electric motors and delivers it to the wheels.

The new drivetrain design utilizes a modular 'Pod' approach. Each of the four wheels is driven by the DrivePod and they are mechanically independent. This allows for easy swapping of broken DrivePods which is essential for a mine detection robot. The mechanical independence of each wheel also allows for more precise control and freedom to steer. The design utilizes an innovative dual motor drive which adds redundancy to the system and also lowers the total cost of the system.

The rendering of the DrivePod is shown in Figure 1.1 and the final DrivePod specs are listed in Table 1.1.

Figure 3.1: Rendering of Gladiator's DrivePod. Final Mass optimized version shown.

Table 1.1: Gladiator's DrivePod Specifications

Parameter	Specs
Quantity	4
Weight	2.3kg
Max. Dimensions	10"x6"x14"
Total Power	52W
Drive Torque	2.8Nm
Max Velocity	2m/s
Max. Load	10kg
Max. Acceleration	$1m/s^2$
Max. Incline	25%

Objectives

Cornell Minesweeper is designing robots for humanitarian landmine detection in countries such as Cambodia. The robots must be all terrain capable and must also be light and energy efficient. The team also is competing in the Intelligent Ground Vehicle Competition which test robots for their autonomous abilities. Both these tasks require a very stable platform that can easily and accurately controlled. However, the robustness requirements for both vary to a large degree. The IGVC is held on a grass field with minimal terrain disparities. The worst terrain condition at the competition is a sand pit and an incline of 15%. In a minefield, it could be a swampy clay pit or it could be oversized undergrowth. Due to the indeterminate nature of the requirements of the minefield, it was decided to design the DrivePods for IGVC. To ensure that Gladiator would be able to handle worse terrain to aid in the testing of the landmine detection sensors, the DrivePods were overdesigned to compensate for higher inclines, payloads and torque requirements.

At the same time, the DrivePods were also designed to have certain redundancy inbuilt to ensure longevity in a minefield. Also, the inbuilt modularity allows for quick fixes and overhauls of Gladiator in the event of a catastrophic loss of a wheel due to a landmine.

Design

Dynamics[1]

Gladiator's drivetrain was designed such that it can provide sufficient torque at the worst case scenario possible. Since this drivetrain was designed more for the competition than for landmine detection, the worst case scenarios at the competition were considered coupled with similar worst case scenarios for landmine detection. This scenario is depicted in Figure3.1 where the robot is on an incline with the friction coefficient of sand.

[1] Reddy

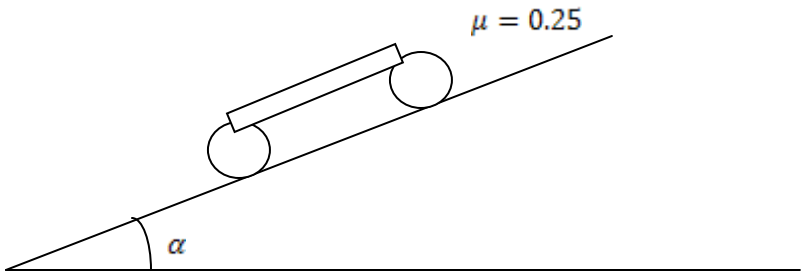


Figure 3.1: Robot on an incline of α degrees and coefficient of rolling friction of sand

The above situation was then converted into a free body diagram to aid in the determination of motor torque for each wheel. In the creation of the free body diagram, the following assumptions were made:

- Wheel deformation is negligible. This implies that the there is a point contact force with the ground which is easier to model.
- The system is in steady state. This allows us to ignore static effects such as static friction, rotor friction and also allows us to consider just the final torque the motor needs to generate.
- The mass is equally distributed on all four wheels. This affects the wheel traction force generated and also the rolling friction faced by each wheel.
- Assume wheel inertia effects are negligible compared to mass of the robot.

The resulting free body diagram for a single wheel is shown in Figure 3.2.

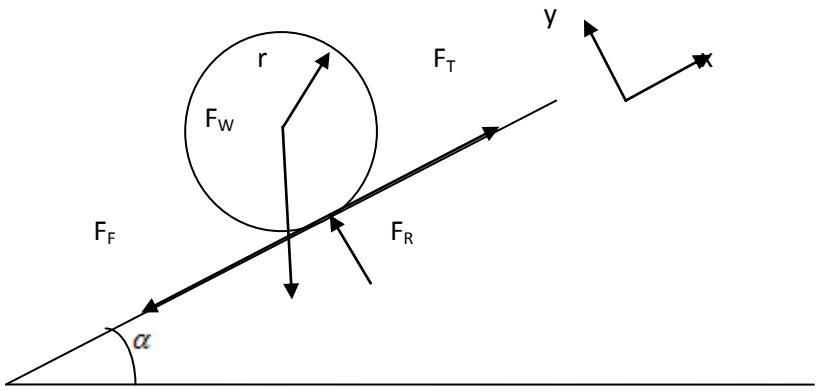


Figure 3.2: Free Body Diagram of a wheel

The variables listed in the above figure are explained in Table 3.1.

Table 3.1: List of Variables used

F_T	Traction Force. This is the driving force for each wheel and is related to the driving torque, T by F_T=T/r.
F_F	Friction. This is the rolling resistance for each wheel and it is related to the normal force by $F_F=\mu F_R$.
F_R	Normal Reaction Force
F_W	Weight loading on each wheel, F_W=mg
α	Angle of the incline
m	Mass supported by each wheel
a	Robot's acceleration

The summation of forces along the X and Y axes from the FBD result in equations 3.1 and 3.2

$$F_T - F_W \sin(\alpha) - F_F = ma \quad (3.1)$$

$$F_R = F_W cos(\alpha) \quad (3.2)$$

Combining and simplifying equations 3.1 and 3.2 by using the constitutive law of rolling friction, we get equation 3.3 which defines the traction force in terms of the coefficient of rolling friction, μ , the angle of inclination, α and the mass supported by each wheel, m.

$$F_T = ma + mg(\sin(\alpha) + \mu \cos(\alpha)) \quad (3.3)$$

The drive torque is obtained from the above equation using the torque constitutive law and is defined in equation 3.4.

$$T = \frac{mg}{r}\left(\frac{a}{g} + \mu \cos(\alpha) + \sin(\alpha)\right) \quad (3.4)$$

This design equation was then programmed in MATLAB and iterated upon for the different variables till a reasonable drive torque was achieved. The constants were m, μ and α and the design variables were r and a. Table 3.2 shows the final converged values. Figure 3.3 shows the iteration plots.

The Torque values were checked against the IGVC velocity requirements which specifies a maximum of v=2m/s. For this maximum velocity, the drivetrain has to provide torque T_{nom} at ω =15.75 rad/s. This implies that the drivetrain must be able to provide a constant power of atleast P=Tω=50W. The next step in designing the drivetrain is the selection of the motors and this value of power is the selection parameter.

Table 3.2: Final Values of Drivetrain variables

Variable	Value	Comments
m	7.5kg	This assumes that the robot's mass of 30kg is equally distributed on all four wheels.
g	9.81 m/s^2	Gravity
μ_{nom} μ_{max}	0.15 0.25	Coefficient of Rolling Friction of grass + loose gravel Coefficient of Rolling Friction of sand. Sand is the worst ground sinkage condition at the competition.
α	14 deg	This is a 25% incline. The maximum grade at the competition is 10%.
r	0.127m	This is a 10" wheel which provides sufficient ground obstacle clearance.
T_{nom} T_{max}	3.2 Nm 5.7 Nm	These values of the Torque satisfy all these parameters at the same time.
a	$1m/s^2$	This is the maximum acceleration possible with the above torque.
v	2m/s	Robot's velocity. IGVC requirement

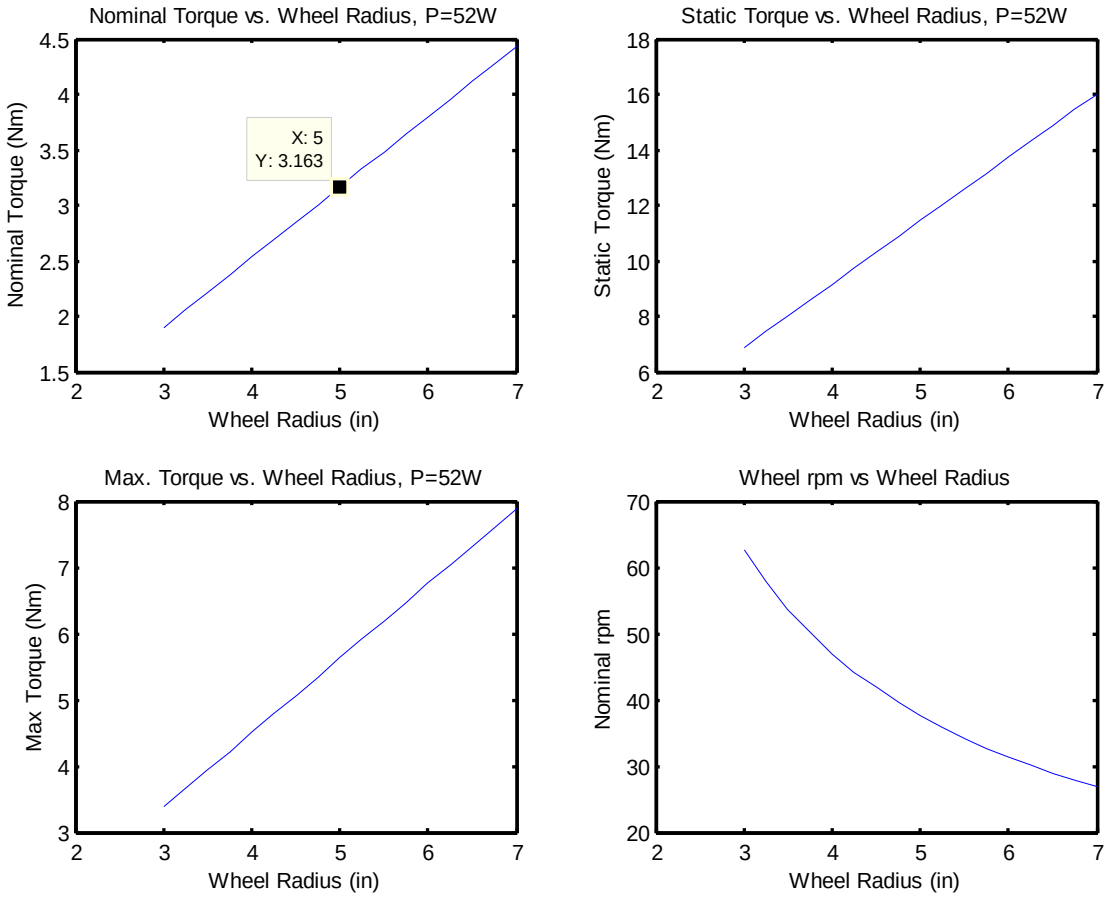


Figure 3.3: Drive Power and Torque Iterations Plot

Motor Selection

There are several types of motors: AC motors, Induction motors, DC Brushed motors, Brushless DC motors, Servo motors and Stepper motors. For our size and power demands, the DC motors have the best efficiency and weight. Brushless DC motors are a lot more efficient than brushed motors due to the lack of friction in the brushless commutation. However, they are harder to control electronically. Thus, the DC brushed motor was the motor of choice due to its control simplicity, relatively high power to mass ratio and decent efficiency.

Power is the critical selection factor over torque for a motor since gear reduction can be employed to gain the desired Torque and rpm. Besides power, the motor's weight, voltage and rotor inertia are important selection parameters. The rotor inertia is critical in determining the motor's acceleration and deceleration profile. This is important is if the motor has to respond really fast to differing input signals and the effectiveness of this property is highly determined by the robot's control logic.

Based on the above parameters a thorough internet search was conducted and the Faulhaber 2657CR motor with the 26/1, 66:1 reduction gearbox was determined to be the best. The specs are shown in Table 3.3.

Table 3.3: Faulhaber Motor + Gearbox specs

Parameter	Value
Power	47.9W
Weight	0.296 kg
Voltage	24VDC
Stall Torque	18.9Nm
Gear Reduction	66:1
Efficiency	70%
Price	$750 (retail), $450 (academic)

The previous drivetrain on Spongebob utilized the Faulhaber 2342CR motors which performed excellently, increasing the team's faith in the current motor selection. However, the price was deemed to exorbitant and an alternative design solution was created.

The BH31 gearmotors from The Robot Marketplace (www.robotmarketplace.com) were found to have the weight and torque characteristics the drivetrain needed. Moreover they cost only $25 a piece. However, they were only rated for P=26W which was half the requirement. The motor's Torque and Power curve is shown in Figure 3.4.

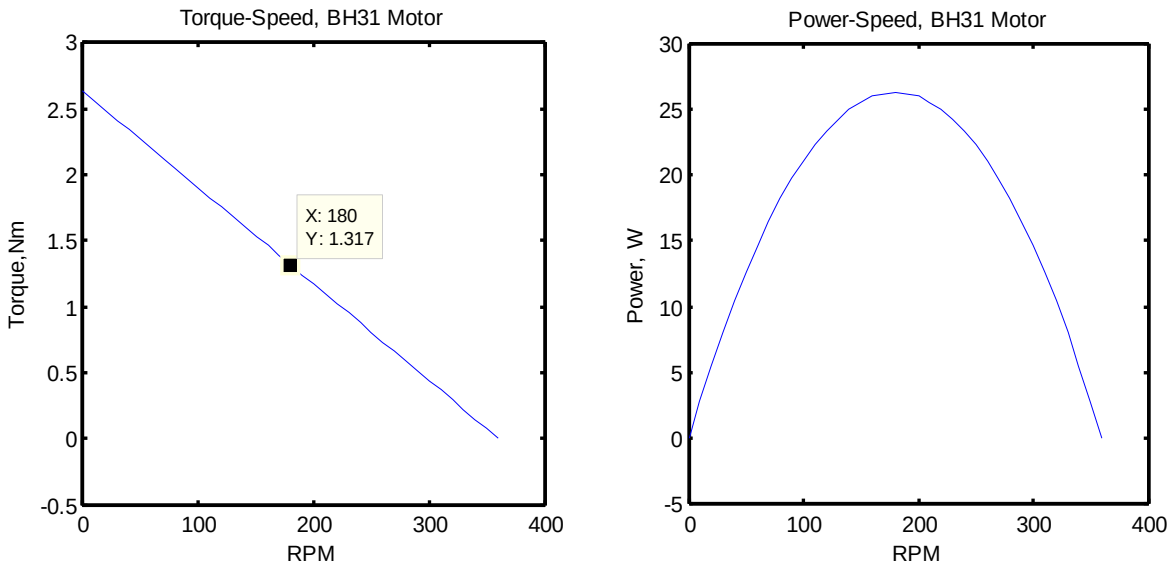
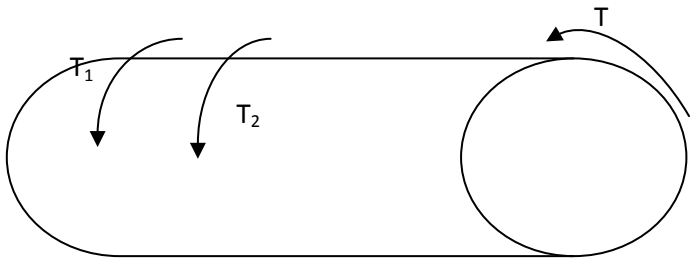


Figure 3.4: Torque and Power curves for BH31

Based on the cost and weight initiative, an ingenious solution was devised. It was proposed that two motors be used per wheel to gain the total power needed to meet the specifications. A similar drivetrain configuration was used by the Cornell Hybrid Electric Vehicle team but the motors weren't used in parallel. A higher power motor was used when the vehicle had to negotiate a higher incline or velocity which the lower power motor was used during in stop and go city driving. Since literature documenting this exact design configuration was not found, this design was consulted upon with Prof. Jack Thompson. The following argument was made and it was agreed upon by Prof. Thompson.



$$T = T_1 + T_2$$

$$P_1 = T_1\omega_1; P_2 = T_2\omega_2$$

Since the shaft is physically constrained to rotate at a single speed, $\omega = \omega_1 = \omega_2$, then

$$P=P_1+P_2=(T_1+T_2)\omega$$

Thus, the total shaft power is the sum total of the individual power inputs provided the torque is added at the same rpm.

The initial drive configuration had each motor connect directly to the drive shaft such that if one motor failed, the other motor could still provide half the power. This parallel configuration is ideal for mine detection applications where reliability and redundancy are critical.

However, for the competition, the robot will use a serial configuration since this configuration reduces the number of moving parts and also simplifies the fabrication process. But this also implies that the design has lesser tolerance for error. If the single power transmission component fails, the robot loses an entire drivetrain. Considering the running lifetime of the robot at the competition, the probability of this failure is negligible. Thus, the final drivetrain power and torque configuration is in Table 3.4.

Table 3.4.: Gladiator's Motor Drive Configuration

Parameter	Value
Power	52W (26Wx2)
Weight	0.420kg (0.210kg x2)
Voltage	24VDC
Stall Torque	2.79Nmx2
Gear Reduction	33:1
Efficiency	?
Price	$50

Transmission[2]

Gear Ratio

From figure 4.4, the BH31 motor's nominal torque is found to be 1.3Nm at 160rpm. It was decided to incorporate a factor of safety in this number and the motor's drive torque and maximum efficiency was assumed to be 1.0Nm. Two of these motors result in a total drive torque of 2Nm but the required drive torque is 3.2Nm. Thus, a gear reduction is necessary.

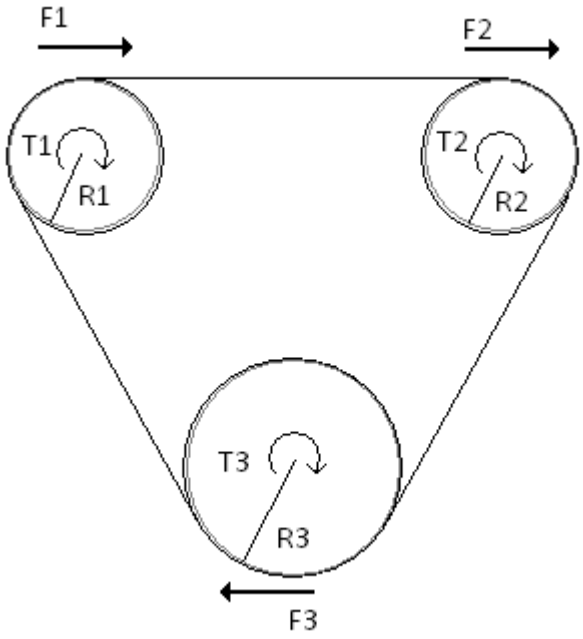
The obvious gear reduction to choose would 1.6:1, however a few more assumptions were made and it was decided to use 1.4:1 instead. The 3.2Nm drive requirement would drive the 30kg robot at 2m/s up a 10% incline. But since this an overestimate of the driving terrain, it was decided to lower the gear ratio to increase the drivetrain's efficiency. The 1.4:1 ratio provides a driving torque of 2.8Nm and is 10% more efficient than the 1.6:1 reduction.

Methodology

In keeping with the modularity aspect of the design, each wheel is powered by an in-wheel drivetrain instead of a single motor delivering power via transmission to all wheels. This makes each wheel independently operated and thus, more likely to succeed. In the event of failure, a stand-alone drivetrain module is easier to replace.

Each drivetrain utilizes two 24 Volt DC motors, acting in series, to provide power to the driveshaft. The following free body diagram outlines the mechanical advantage:

[2] Patel



Two motors act on pulleys 1 and 2 driving pulley 3; the following analysis shows the mechanical advantage:

Evaluating the forces acting on the belt via the motors:

$$F_1 = {T_1}/{R_1}; \quad F_2 = {T_2}/{R_2} \tag{3.5}$$

The forces must be in static equilibrium:

$$\sum F = 0 = F_1 + F_2 - F_3$$
$$F_3 = F_1 + F_2 \tag{3.6}$$

Substituting the previous result:

$$F_3 = T_1/R_1 + T_2/R_2 \tag{3.7}$$

Writing F_3 in terms of the torque and radius:

$$\begin{aligned} T_3/R_3 &= T_1/R_1 + T_2/R_2 \\ T_3 &= R_3 \cdot \left(T_1/R_1 + T_2/R_2 \right) \end{aligned} \tag{3.8}$$

Given that R_1 and R_2 are equal, we find:

$$T_3 = R_3 \cdot \left(\frac{T_1 + T_2}{R_1} \right) \tag{3.9}$$

As we can see, we final torque (T_3) acting on the driveshaft is the sum of the motor torques (T_1,T_2) multiplied by the gear ratio $\left(R_3/R_1 \right)$.

Timing Belt Selection

In designing the drivetrain, selection of the belt and pulleys are critical. The following outlines its technical considerations:

- belts last approximately 3000 hours
- frame mounts must be rigid to prevent variations in belt tension
- minimize backlash during forward and reverse operation
- minimize pulley misalignment
- achieve desired gear ratio

We use the technical library at Stock Drive Products/Sterling Instruments (SDP/SI) to design the drivetrain to meet the above specifications. SDP/SI provides performance information on a range of belt types, the following figure plots speed verses horsepower for various belts:

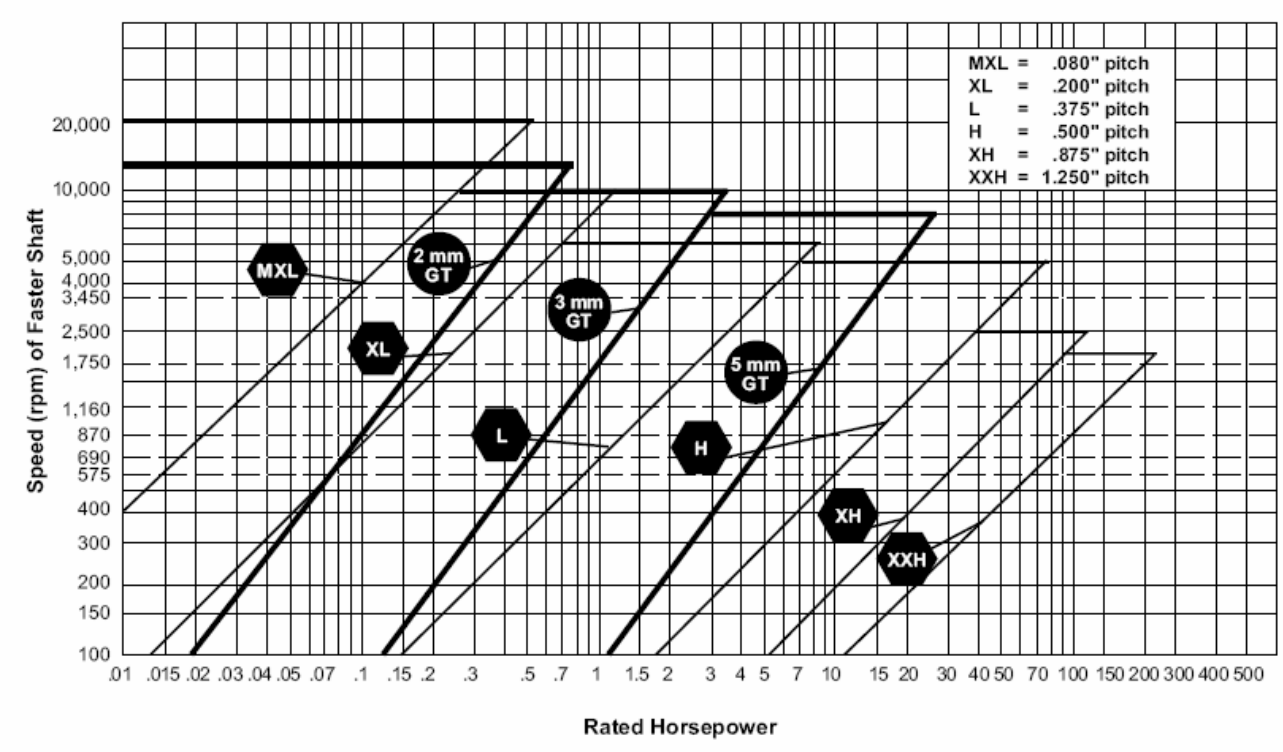
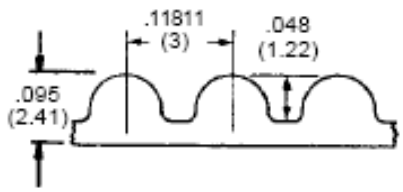


Figure 3.5: Timing Belt Selection Chart

The 3mm GT belt is most suitable for our application. We use the GT series because it is a proven design by the Gates Corporation. The following figure shows the tooth and pulley contact cross-section view

Figure 3.6: Powergrip GT2 Belt Profile

The high contact area reduces shear stress experienced by the belt, greatly increasing belt life. The curvilinear tooth profile provides minimum backlash when the drivetrain changes its direction of rotation. The GT series belts are also specifically engineered to mesh cleanly resulting in quieter and efficient operation. The following figure gives a detailed tooth profile:



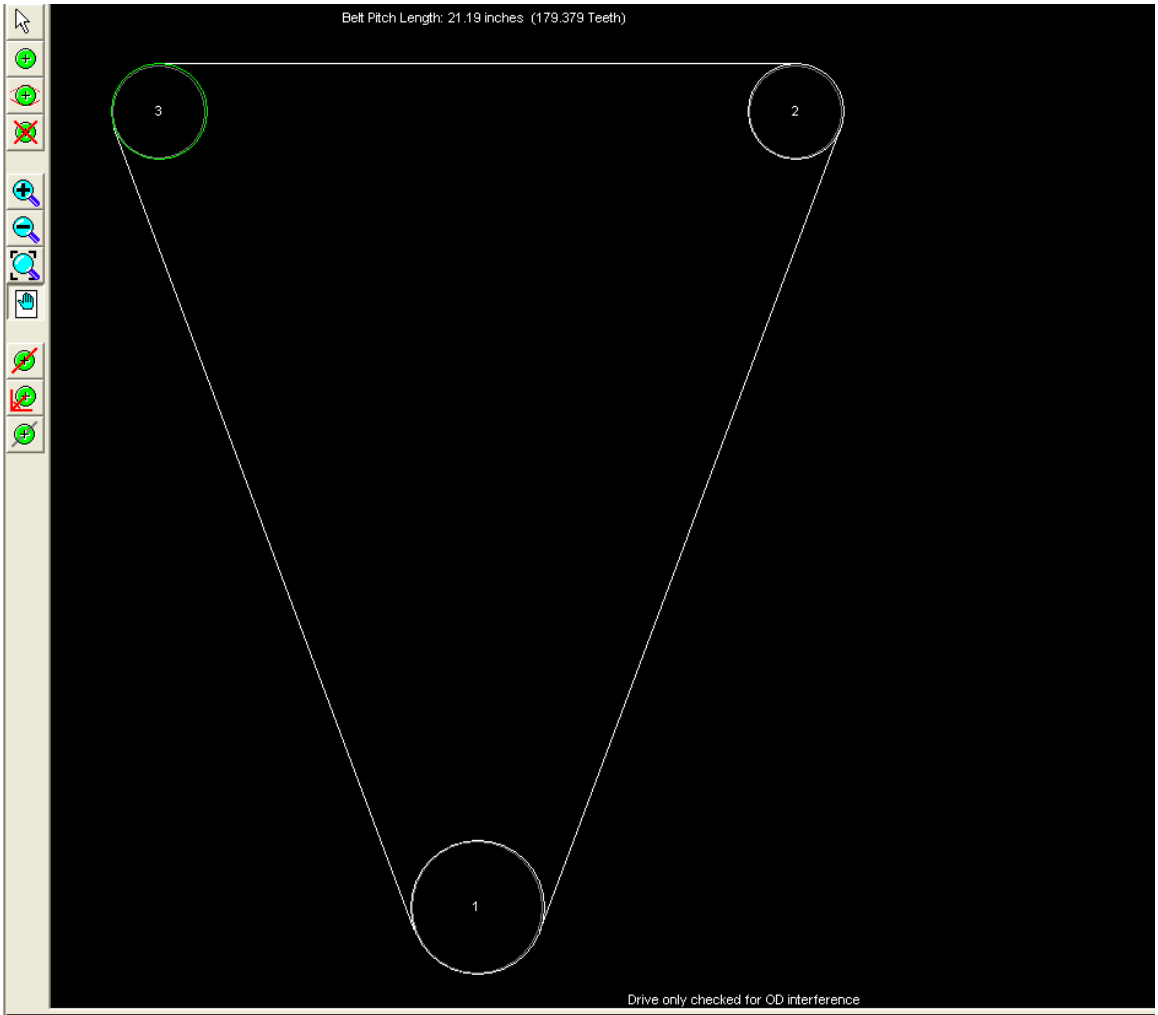
Pulley Selection and Positioning

The GT series makes various sized pulleys to accompany the 3mm belt. We have previously determined to use a 1.4:1 gear ratio in our design. The GT series makes pulleys as small as 16 teeth and as large as 80 teeth. To select the proper pulleys, we consider the following

- minimum teeth-in-mesh (TIM) criteria of 6
- pulley diameter must be much greater than belt diameter
- pulleys should be flanged to prevent belt creep
- minimum overall dimensions, compact

In order to minimize cost, we use pulleys of standard number of teeth. For example, a 17 teeth GT pulley costs $15.43 while a 16 teeth pulley costs $11.20. Also, we want to minimize the overall dimensions due to space limitations at each wheel. Therefore, we choose a 20 teeth pulley on the motor side to power a 28 teeth pulley on the drive side.

The minimum TIM criteria determined experimentally by the Gates Corporation says that at least 6 belt teeth must be in contact with the pulley for efficient torque transmission. This condition will guide our pulley positioning. Gates makes a design software tool, DesignIQ, that includes a database of timing belts and pulleys. The tool calculates the overall required belt length and number of TIM given pulley type and position. The motor side pulleys and drive side pulley must be at minimum a wheel radius apart to ensure no interference. The following screenshot from DesignIQ shows the overall pulley locations, TIM for each, and overall belt length:



No.	Pulley Name	X Coord	Y Coord	Side	Grooves	Units	Type	Diameter	Moveable?	Arc Length	Ratio	Wrap Angle	Arc Length	Span Length
1	1	2.50	0.00	Inside	28	---	Grooves	1.02 OD	Fixed		1.00	139° (10.8 Teeth)	1.28	6.73
2	2	5.00	6.25	Inside	20	---	Grooves	0.72 OD	Fixed		0.71	110.5° (6.1 Teeth)	0.73	5.00
3	3	0.00	6.25	Inside	20	---	Grooves	0.72 OD	Fixed		0.71	110.5° (6.1 Teeth)	0.73	6.73

Figure 3.7: DesignIQ Results

As we can see from the Wrap Angle, the TIM criteria is met for all pulleys and the overall center-to-center distance between the motor and drive pulleys is 6.73 inch, larger than wheel radius. The total number of teeth in this design is 179.38. The extra 0.38 teeth allows for slack within the assembly.

Wheels

The following outlines the criteria we considered in choosing wheels to incorporate into the drivetrain:

- high traction, high rolling resistance
- minimize weight
- low damping so wheel absorbs vibrations

We considered three wheel design options for the drivetrain. First, we looked at a wheel produced by Northern Industrial:

Figure 3.8: Wheel from Northern Tools

The wheel is 8 inch in diameter and 1.5 inch wide. It is made of solid rubber with polyurethane spokes and aluminum hub. The wheel and tire assembly were light at .25lbs but it was not chosen since the wheel hub was designed for a free rolling wheel. The conversion of this hub to a driving hub was not possible due to material and size restrictions.

Next, we considered a wheel made by Xootr:

Figure 3.9: Xootr Wheel

The wheel is 7 inch diameter, 1 inch wide, and weighs 1 lb. It is made of solid polyurethane with all aluminum spokes and hub. The Xootr has very low rolling resistance, and thus has very little traction. The solid rubber provides little damping and is designed to be used in an urban setting. The wheel has

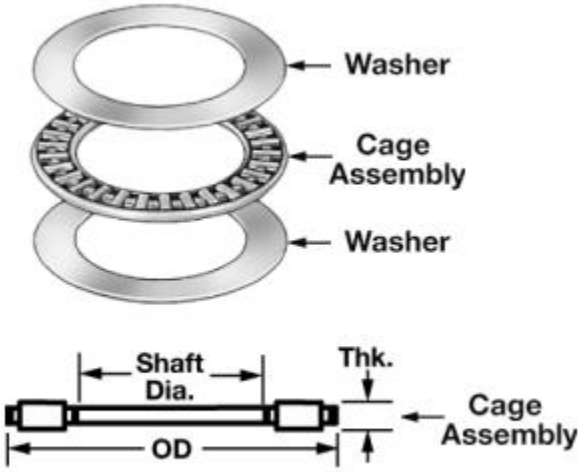
no treads and has a smooth finish indicating a low rolling resistance. This wheel is not an optimal choice for an all-terrain application because of the likelihood of slip and little damping.

Finally, we looked at wheels designed by Radio Flyer. These wheels are specifically designed for children's wagons for all-terrain applications. The wheel is 10 inch diameter, 3.5 inch wide, and weighs 2 lb. It is a pneumatic wheel made of natural rubber with nylon spokes and hub. The wheel is designed for outdoor use and thus is treaded. The pneumatic feature of the wheel provides low damping and absorbs vibrations making it appealing to our application.

Ultimately, we choose the Radio Flyer wheel due to its all-terrain design and ability to damp vibrations.

Bearing Selection

Bearings help reduce friction between two moving parts. The drivetrain consists of many moving parts, and thus we utilize bearings to reduce power dissipation to friction. Between the rotation of the pulleys and the surface of the V-plate, we implement needle roller thrust bearings. The following image outlines a thrust bearing:



To constrain the driveshaft, we utilize self-aligning bearings at the two ends. These self-aligning bearings are from Igus and are shown below.

The bearings allow the driveshaft to freely rotate but do not rigidly constrain it in the radial direction. Self-aligning bearings allow for upto 21 degrees of misalignment and still allow rotary motion. This provides a very high tolerance for machining error which reduces the overall cost of the robot. Also, these bearings were obtained for free via the Y.E.S program at Igus.

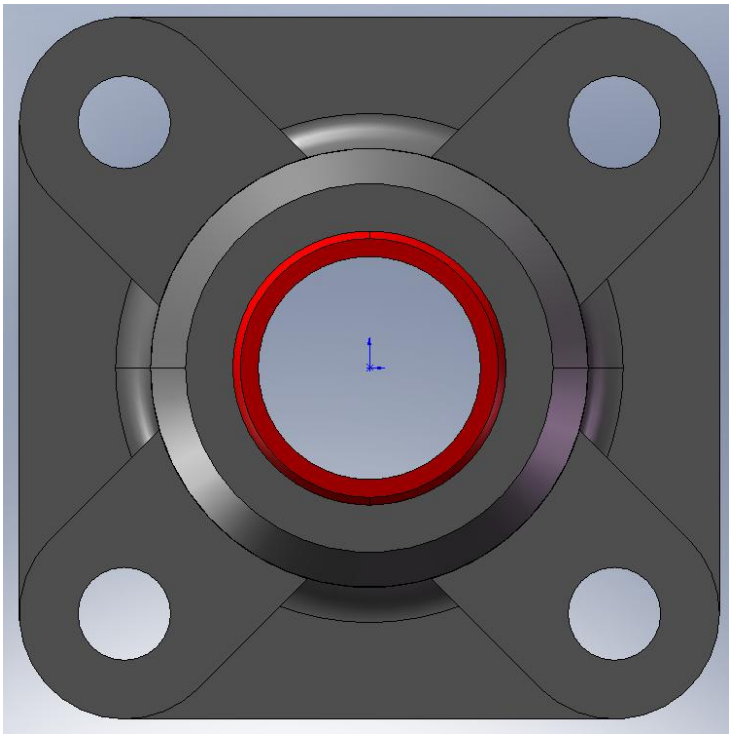


Figure: Igus Spherical Bearing

With the finalization of the all the DrivePod components, we proceeded to design the various parts needed to hold the components in place and transmit the optimum torque. The design has been through several iterations and only the final configuration is discussed.

Parts Design and Analysis

Mounting Plate Design[3]

The two main criterions for this fixture, the V-plate, were mass optimization and a factor of safety against yielding. The following CAD image shows the mounting plate:

Figure 4.1: V Plate Rendering

The dimensions found from DesignIQ are applied to the V-plate. The one side of the plate has rigid motor mount locations, while the other side is slotted to account for tolerance stack-up. The slotted feature also allows for tuning in belt tension and ease of assembly. The following figure shows the assembly of the pulleys, motors, and V-plate:

[3] Patel

Figure 4.2: V Plate Rendering with attached Motors and Pulleys

The V-plate is optimized for weight reduction, while maintaining a factor of safety against yielding. The following stress analysis from COSMOS shows that the design has low stress concentrations and a factor of safety close to a 100. This FOS is not a good indication since boundary conditions are not accurate.:

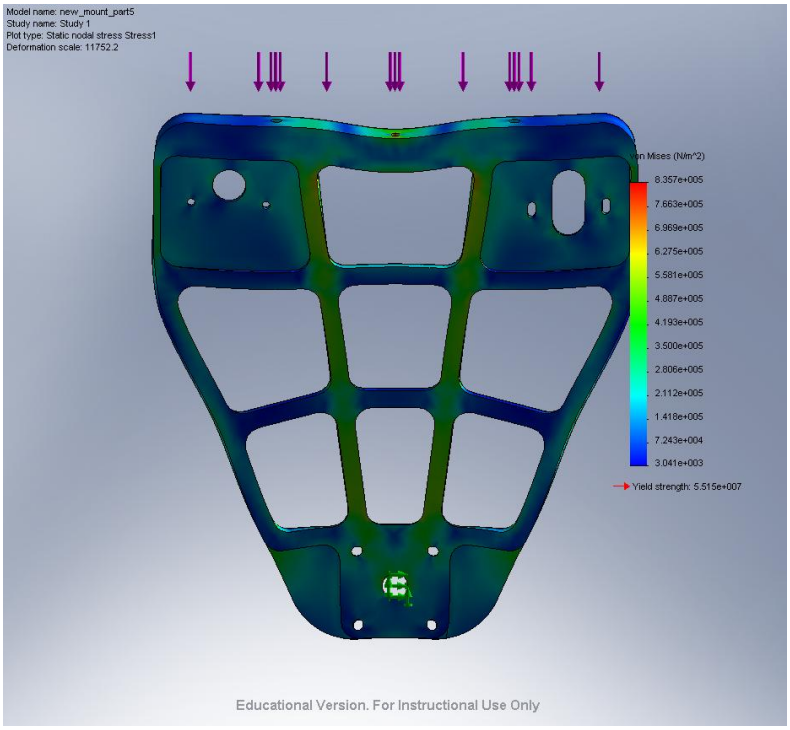


Figure 4.3: COSMOS Stress Analysis of the V Plate

Mounting Top Plate[4]

The mounting top plate connects the steering shaft to the drivetrain. It acts to transfer the steering torque to the drive assembly. The following CAD image shows its design:

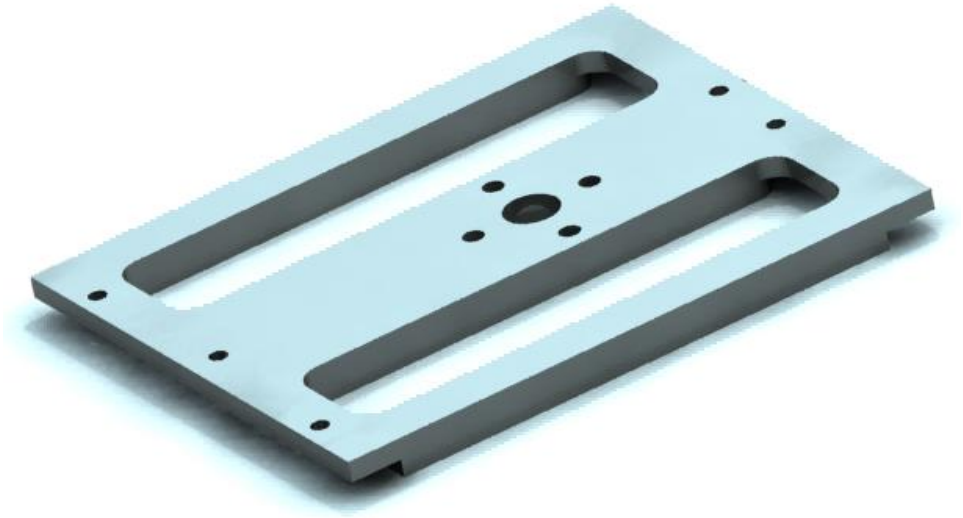
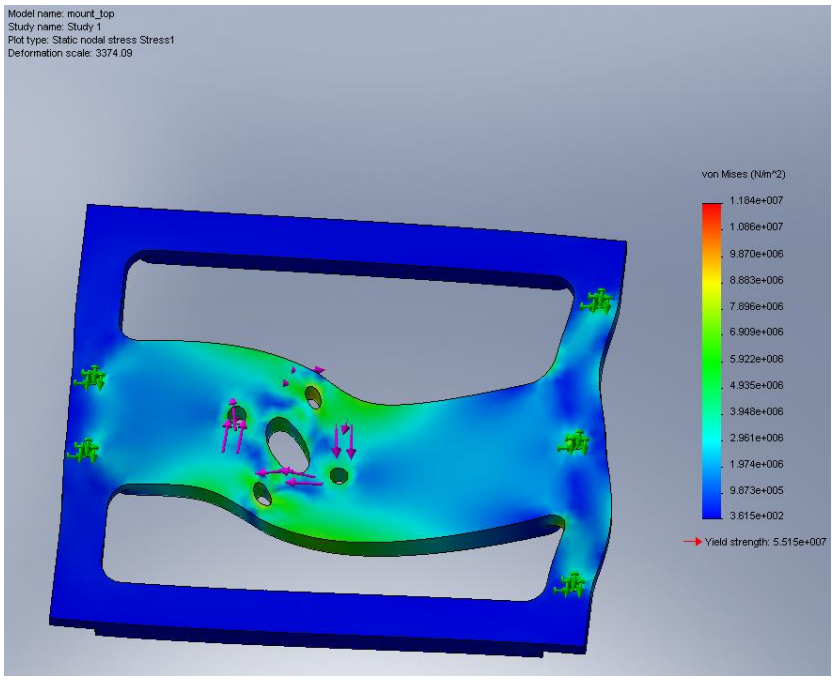


Figure 4.4: Rendering of the Mounting Top Plate

As we can see, the holes for connecting the V-plate to the mounting top plate are offset. This helps eliminate any lateral or rotational play within the assembly. It ensures the structure will be rigid. The overall design is mass optimized and the following COSMOS image shows the stress analysis in this part:



[4] Reddy

Figure 4.5: COSMOS Stress Analysis of the Mounting Top Plate

The Other Plate[5]

The other plate provides structural support to the assembly. It provides the mounts for constraining the driveshaft in the axial direction. Also, the flat surface of this part provides a location to mount encoders. These sensors measure the angular rotation of the driveshaft for control purposes. The following CAD image shows the mass optimized support plate:

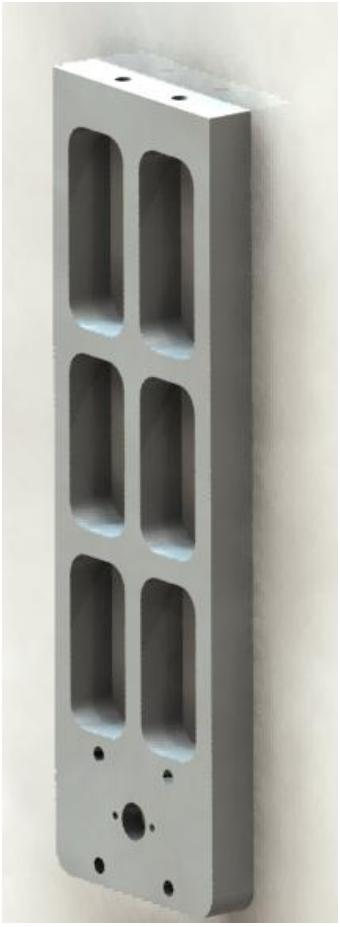


Figure 4.6: CAD Rendering of the Other Plate

This part has been mass optimized. A design check was performed using COSMOSWorks and the stress analysis results are shown in Figure 4.7. The design suffers from minimal deformation and minimal stress concentrations.

[5] Patel

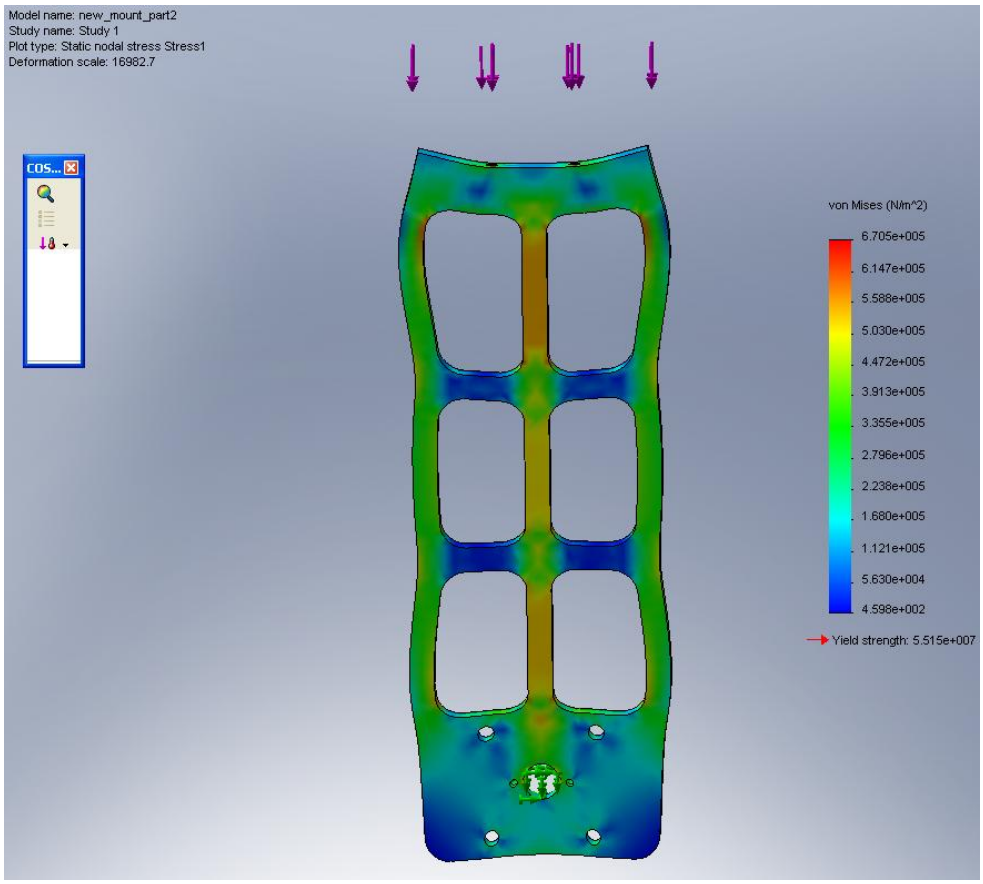


Figure 4.7: COSMOS Stress Analysis of the Other Plate

Drive Shaft[6]

The drive shaft performs two roles: it transmits the rotary power from the transmission to the wheels and it supports the robot's weight. It also undergoes the most number of cycles of varying loads and is subject to fatigue failure. Due to the fatigue, radial and torsional requirements, steel is the best material for the shaft largely due to its infinite lifetime and high strength.

The shaft was designed to support the robot's weight, provide the driving torque and also provide mounting locations for the wheel, drive pulley, bearings and the drive encoders. The shaft is shown in Figure 4.8. The shaft is .516" in diameter and is rated at W1 Tool Steel which has good machinability characteristics and has a yield strength of 650MPa. The shaft is 7" long and has ¼" tapered ends which allow for the drive pulley and the optical encoders to be mounted. The shaft also has two diameter reductions which measure 0.5" so that it fits in the bearings well.

[6] Reddy

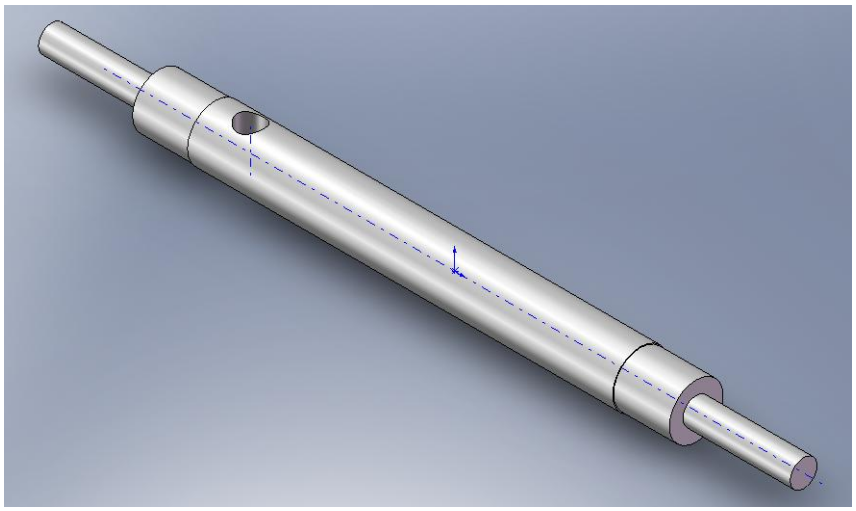


Figure 4.8: Gladiator's Drive Shaft

The unconventional 0.516" diameter was chosen to allow for a press fit with the Radioflyer wheel hub. The press fit is not reliable since the hub is made of plastic and plastics tend to creep. So a positive engagement was designed to link the drive shaft and the wheel. A quarter inch ball nose mill was used to mill a notch in the wheel hub. Then a .246" hole was drilled in the shaft at a predetermined location and a half-inch steel pin was pressfit in the shaft. This steel pin engages with the notch in the wheel hub and provides a non-slip wheel engagement. To further ensure that this linkage does not fail, a two piece shaft collar (6436K48 from McMaster) was used to keep the wheel hub in compression. This assembly is shown in figure 4.9.

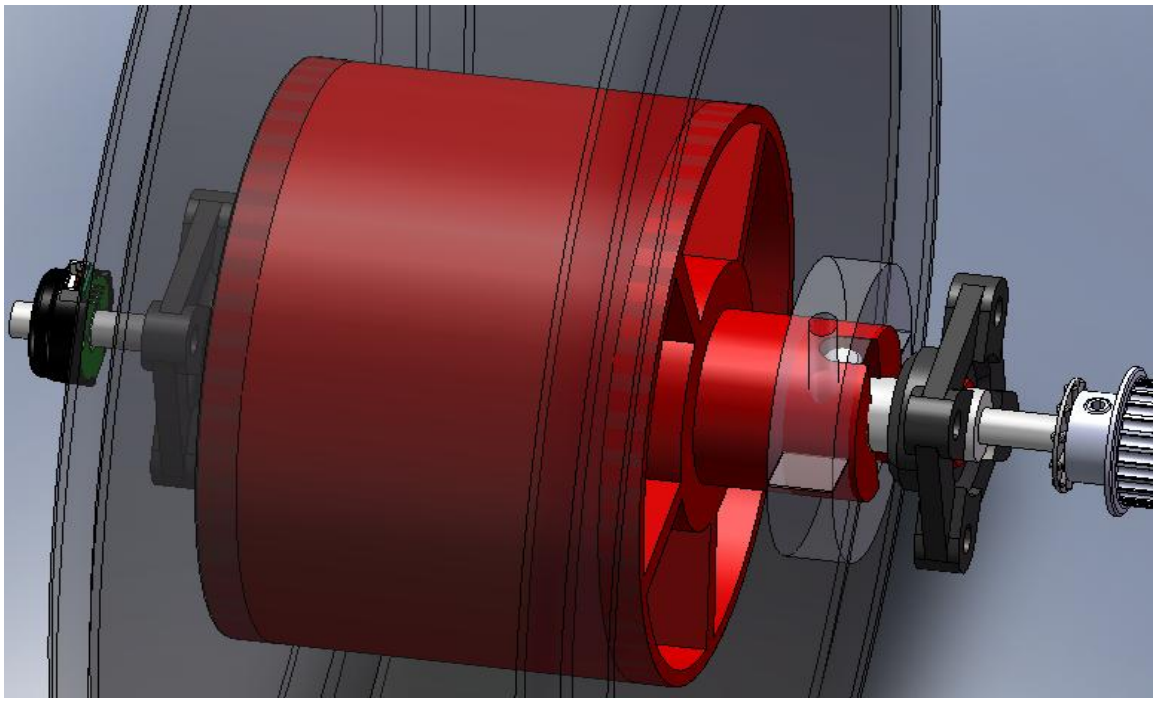


Figure 4.9: Drive Shaft Assembly

Fabrication

The parts were designed for manufacturing and assembly. They have simple 2D profiles and easy access points. All the parts were initially designed to be machined on the conventional mill and lathe. But with the recent availability of the CNC, the designs were mass optimized in a more geometrically unconstrained manner. The fabricated DrivePod is shown in the following figure. This was entirely fabricated on the conventional mill and lathe and took close to 12 man hours due to improper tooling.

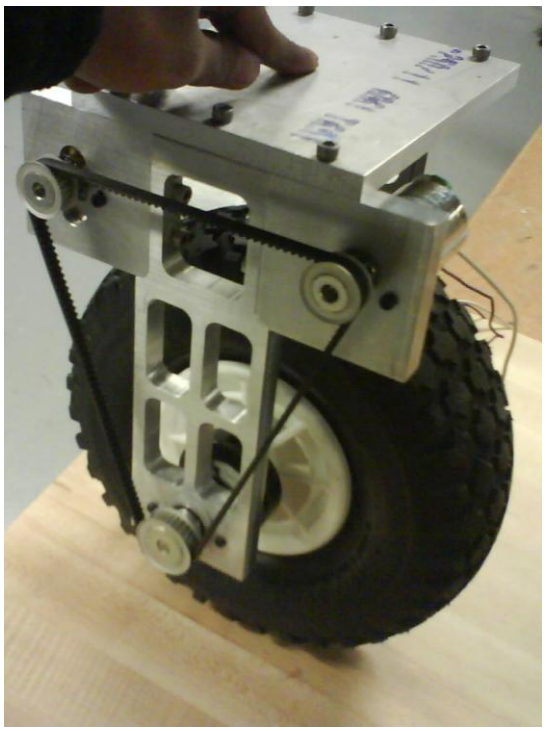


Figure: Fabricated DrivePod. Initial Version

Conclusion

The DrivePods were successfully designed and their manufacturing feasibility was demonstrated. The DrivePods were also tested statically to determine if the belt engagement works according to design and they performed well under initial tests.

Future Work

The DrivePods are yet to undergo field testing and this will be done in Jan, 2008. The control system for velocity control of the robot has already been thought out and the control block diagram is shown below. The velocity feedback is provided by a US Digital E4p optical encoder and the controller is implemented on an Atmega32 microcontroller. The feasibility of the controller and the control system was tested on Spongebob and velocity control was achieved using open loop duty cycle estimation. Once the closed loop PID controller is implemented, the Drivepods will be subject to complete field testing before being handed off to the CS team.

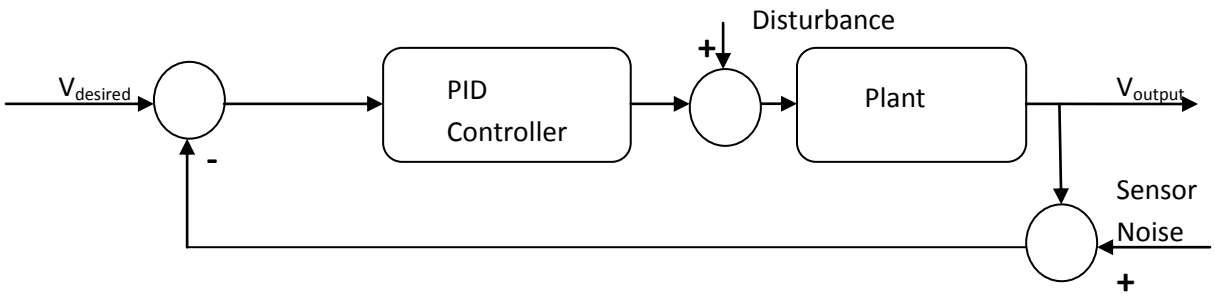


Figure: Proposed Control System

References

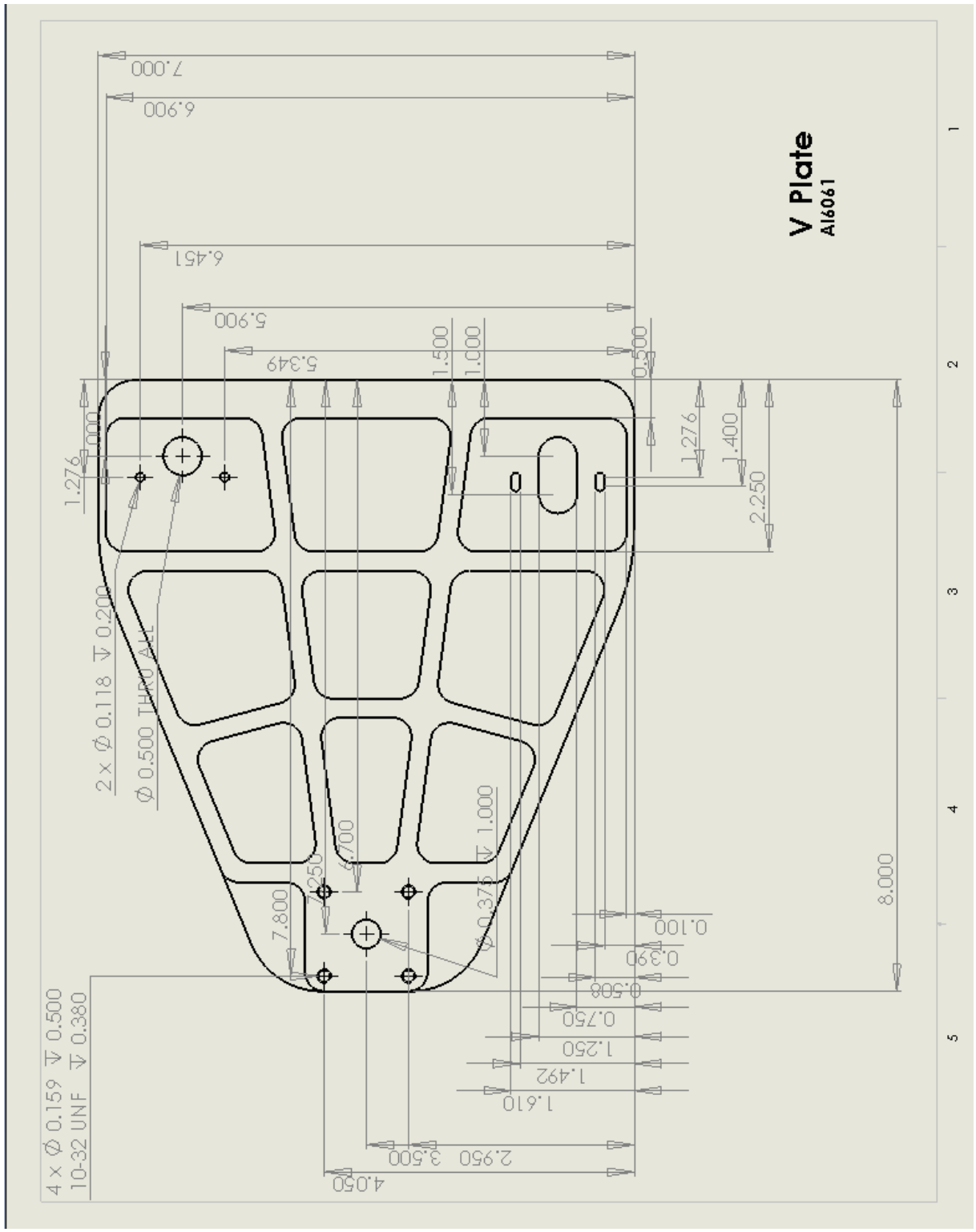
1. Cornell MineSweeper Design Reports: Fall 2006, Spring 2007
2. Bekker. Off Road Locomotion
3. Stock Drive Products: Technical Library
4. Control and simulation of a DC motor driven hybrid electric vehicle / by Ryan Abraham McGee.

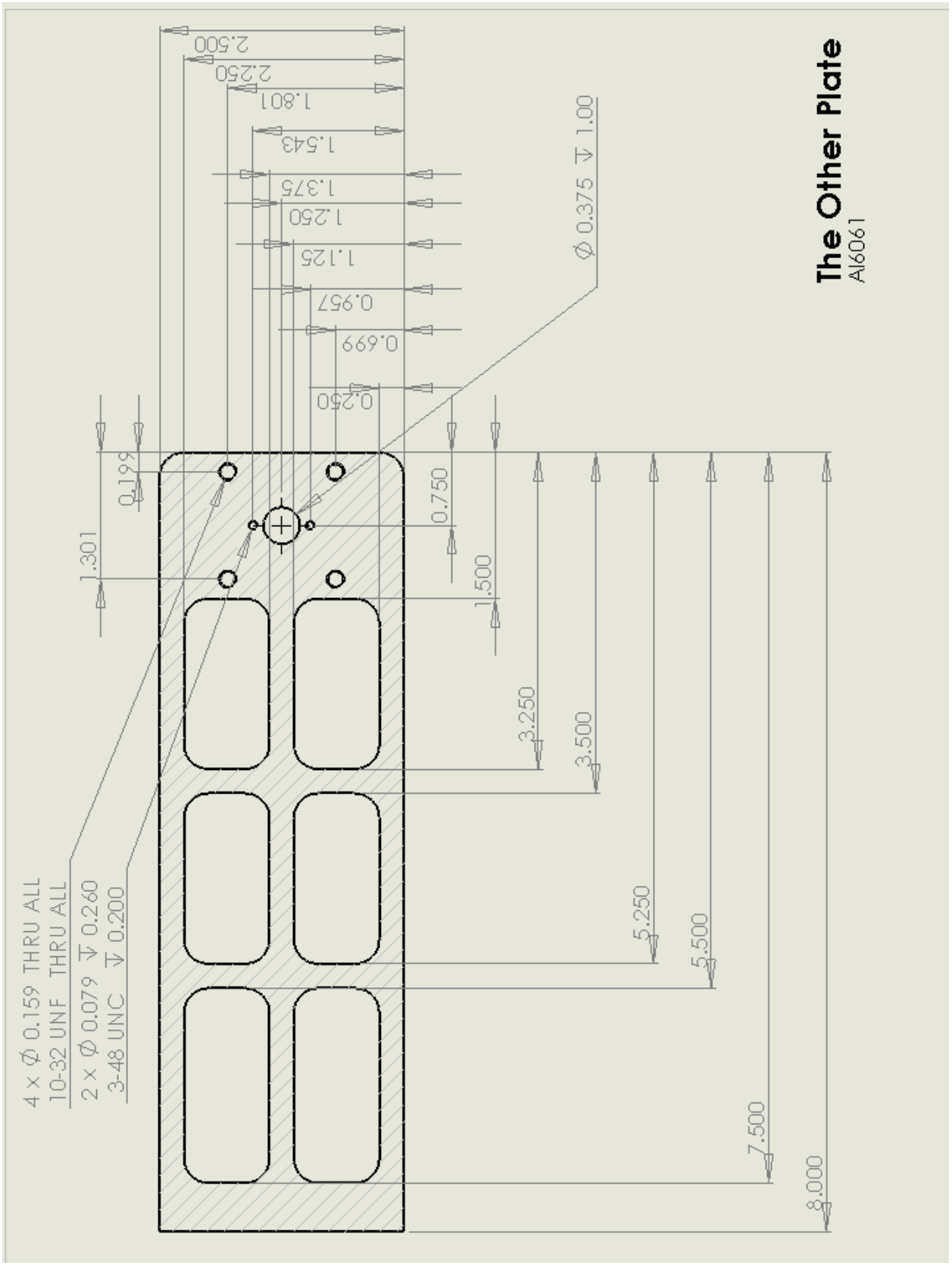
Appendix

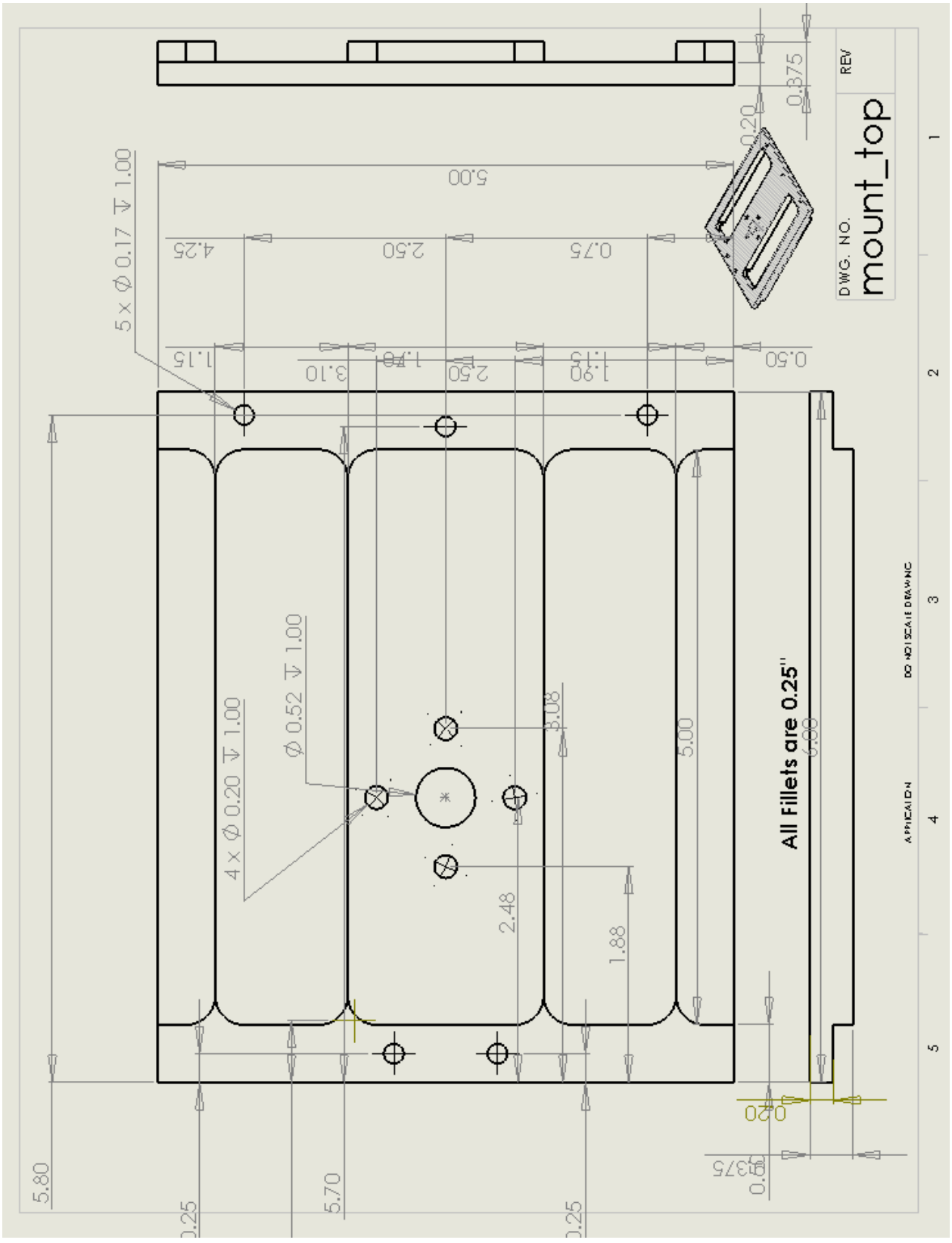
Appendix A: Bill of Materials

Part	Item Number	Vendor	Quantity	Price
V Plate	61f.5x8	Speedy Metals	4	$56.28
Mounting Top Plate	61f.375x6	Speedy Metals	4	$22.20
The Other Plate	61f.5x2.5	Speedy Metals	4	$29.76
Drive Shaft	8890K89	McMaster	4	$7.86
Spherical Bearings	EFSI08	Igus	8	Free
Thrust Bearings	-	MSC Direct	12	-
Motors	BH31	The Robot Marketplace	8	$200
Optical Encoders	E4P-360-250-H-PKG2	US Digital	4	$86

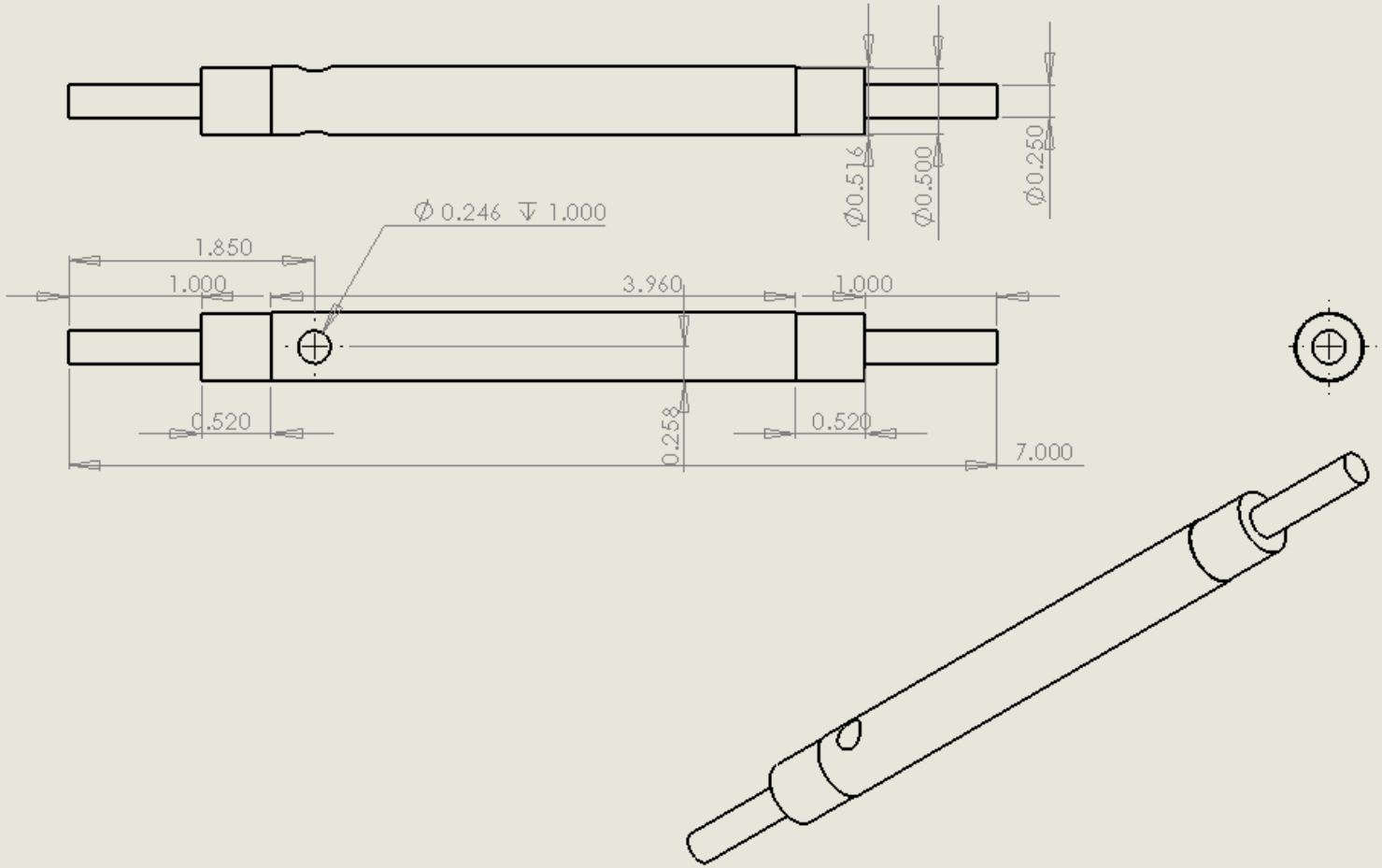
Appendix B: Part Drawings




The Other Plate
A16061
4 x Ø 0.159 THRU ALL
10-32 UNF THRU ALL
2 x Ø 0.079 ↧ 0.260
3-48 UNC ↧ 0.200
Ø 0.375 ↧ 1.00
2.500
2.250
1.801
1.543
1.375
1.250
1.125
0.957
0.699
0.250
0.199
1.301
0.750
1.500
3.250
3.500
5.250
5.500
7.500
8.000


mount_top
DWG. NO.
REV
All Fillets are 0.25''
DO NOT SCALE DRAWING
APPLICATION
5.80
5.70
5.00
6.08
2.48
1.88
0.20
0.375
0.50
0.75
2.50
4.25
1.15
1.90
3.10
1.78
0.25
5 x Ø 0.17 ↧ 1.00
Ø 0.52 ↧ 1.00
4 x Ø 0.20 ↧ 1.00

Shaft_Drive
Carbon Steel
Hole Tolerance Critical
Drill D



Appendix C: Drivetrain Design Code

```
% Cornell MineSweeper Drivetrain Torque and Power Calculator
% Vikas Reddy, ve23@cornell.edu
% Last Updated: 13 Oct 2007
clear
clc
close all

% Parameters (units: SI)
m=30;
v=2;

alpha=15;
alpha_nom = 5;

mu_s=1;
mu_r=.25;
mu_r_nom=.15;

a=1;
g=9.81;

r=5; %(in)
r=r*.0254;

% Equations
F_static=m*g*(sind(alpha)+mu_s*cosd(alpha))
F_dynamic_max=m*g*(sind(alpha)+mu_r*cosd(alpha)+a/g)
F_dynamic=m*g*(sind(alpha_nom)+mu_r_nom*cosd(alpha_nom)+a/g)
i=1;
P=F_dynamic*v/4

for R=4:1:6
    r(i)=R*.0254;
    T_static_wheel(i)=(F_static/4)*r(i);
    T_dynamic_wheel(i)=(F_dynamic/4)*r(i);
    T_dynamic_max(i)=(F_dynamic_max/4)*r(i);
    w(i)=P/(T_dynamic_wheel(i)*4);
    w_rpm(i)=w(i)*30/pi;
    i=i+1;

end
subplot(2,2,1)
line(r./0.0254,T_dynamic_wheel)
xlabel('Wheel Radius (in)')
ylabel('Nominal Torque (Nm)')
title('Nominal Torque vs. Wheel Radius, P=52W')

subplot(2,2,2)
line(r./0.0254,T_static_wheel)
xlabel('Wheel Radius (in)')
ylabel('Static Torque (Nm)')
```

```
title('Static Torque vs. Wheel Radius, P=52W')

subplot(2,2,3)
line(r./0.0254,T_dynamic_max)
xlabel('Wheel Radius (in)')
ylabel('Max Torque (Nm)')
title('Max. Torque vs. Wheel Radius, P=52W')

subplot(2,2,4)
line(r./0.0254,w_rpm)
xlabel('Wheel Radius (in)')
ylabel('Nominal rpm')
title('Wheel rpm vs Wheel Radius')
```

Appendix D: Motor Control Code

```
/*****************************************************************************

                        Cornell Minesweeper

                        Motor Control Program, v1.0

                        Written by: Vikas Reddy, ve23@cornell.edu

                        Last Updated: 1 December, 2007

*****************************************************************************/

#include <avr/io.h>
#include <stdio.h>

#include <util/delay.h>
#include <avr/interrupt.h>
#include "serial.h"
#include <string.h>

#define USART_BAUDRATE 19200
#define BAUD_PRESCALE (((F_CPU / (USART_BAUDRATE * 16UL))) - 1)

#define front 0b01010101
#define back 0b10101010
#define left 0b01100110
#define right 0b10011001
#define stop 0b00000000

//UDR variables
//int volatile Drive[5]={0,0,0,0,0};

int volatile Input[4]={0,0,0,0};
char volatile data;
char volatile data_buffer[50];
//char data_conv[5];

//counter variables
int volatile countInst=0;
int volatile i=0;
int volatile j=0;
int buffer_full=0;

//generate 4 pwm channels with frequency of 31kHz and 20%duty cycle
void init_pwm1(void){
```

```
	TCNT0=0;
	OCR0=50;
	DDRB|=_BV(3);
	TCCR0=_BV(WGM00) | _BV(COM01) | !_BV(CS01)| _BV(CS00);


}

void init_pwm2(void){
	TCNT2=0;
	OCR2=50;
	DDRD|=_BV(7);
	TCCR2=_BV(WGM20) | _BV(COM21) | _BV(CS20) ;

}

void init_pwm3(void){
//10 Hz signal
	DDRD|=_BV(4) | _BV(5);
	ICR1=255;
	TCCR1A = _BV(COM1A1) | !_BV(COM1A0)		//  Both PWM outputs set at TOP,
	 | _BV(COM1B1) | !_BV(COM1B0)		//    clear on compare match
	 | !_BV(FOC1A) | !_BV(FOC1B)		//  PWM mode, can't force output
	 | _BV(WGM11) | !_BV(WGM10);
	TCCR1B = !_BV(ICNC1) | !_BV(ICES1)		//  Disable input capture noise canceler,
					//    edge select to negative.
	 | _BV(WGM13) | !_BV(WGM12)		//  Fast PWM, TOP = ICR1
	 | !_BV(CS12) | !_BV(CS11) | _BV(CS10);   //  clk(i/o) / 1024
}




void move(int dCycle1,int dCycle2,int dCycle3,int dCycle4,char dir){
	OCR0=dCycle1;
	OCR1A=dCycle2;
	OCR1B=dCycle3;
	OCR2=dCycle4;
	switch(dir){
		case 'f':
			PORTA=front;
			printf("Forward\n");
			break;
		case 'b':
			PORTA=back;
			break;
		case 'r':
			PORTA=right;
```

```
                        break;
                case 'l':
                        PORTA=left;
                        break;
                default:
                        PORTA=stop;
                        break;
        }
        UCSRB|=_BV(RXEN);
}


/*void motion_parse(void){
        int dCycle1=Input[0];
        int dCycle2=Input[1];
        int dCycle3=Input[2];
        int dCycle4=Input[3];
        int dir=Input[4];
        move(dCycle1,dCycle2,dCycle3,dCycle4,dir);
}*/
void data_convert(void){
        //UCSRB|=!_BV(RXEN);
        int p=0,r=0;
        printf("j=%i\n",j);
        for(int t=0;t<20;t++)
        printf("Data_Buffer[%i]=%i\n",t,data_buffer[t]);
        char data_conv[3];
        char dir=data_buffer[0];

        for(int q=2;q<j;q++){
                if(data_buffer[q]!=','){

                        data_conv[p]=data_buffer[q];
                        p++;
                }
                else{
                        p=0;
                        Input[r]=atoi(data_conv);
                        r++;

                }
        }
        printf("r=%i\n",r);
        j=0;
        int dCycle1=Input[0];
        int dCycle2=Input[1];
        int dCycle3=Input[2];
        int dCycle4=Input[3];
```

```
		printf("dir=%c,dCycle1=%i,dCycle2=%i,dCycle3=%i,dCycle4=%i\n",dir,dCycle1,dCycle2,dCycle3,d
Cycle4);
		move(dCycle1,dCycle2,dCycle3,dCycle4,dir);

}

void init_INT(void){
		MCUCR=0;
		GICR=_BV(INT0)| _BV(INT1);
		GIFR=_BV(INTF0)| _BV(INTF1);
}
void terminate(void){
		PORTA=stop;
		UCSRB|=!_BV(RXEN);

}
/*ISR(SIG_INTERRUPT0){
		terminate();
		printf("Program terminated via E-stop\n");
}*/

/*ISR(SIG_INTERRUPT1){
		UCSRB|=_BV(RXEN);
		printf("Spongebob Reborn\n");
}*/
ISR(USART_RXC_vect){
		data=UDR;

		if(data!='&'){
			data_buffer[i++]=data;
			j++;
			}
		else{
			putchar('\n');				//use putchar to avoid overwrite
			data_buffer[i]=0x00;	//zero terminate
			i=0;
			buffer_full=1;
			UCSRB|=!_BV(RXEN);
			countInst++;
			printf("Instruction Count=%i\n",countInst);

		}

}

int main(void){
		_delay_ms(50);
```

```
	sei();
	init_uart();
	init_pwm1();
	init_pwm2();
	init_pwm3();
	//init_INT();

	printf("Program Starting.\n");

	DDRA=0xff;

	//int dutyCycle=6125;


	while(1){
		if(buffer_full){
			data_convert();
			buffer_full=0;
		}
		OCR1A=Input[1];
		OCR1B=Input[2];
	}

}
```