Theory of Operations

P16241 Autonomous People Mover

Rochester Institute of Technology

May, 2016

Revision Sheet

Release No.	Date	Revision Description
Rev. 0	11/01/15	First draft of the user manual
Rev. 1	12/10/15	Revised completely by Phase II team
Rev. 2	5/10/15	Updating doc to account for Phase III changes. Retitled “Theory of Operations”

Theory of Operations
Authorization Memorandum

I have carefully assessed the Theory of Operations for the Autonomous People Mover. This document has been completed in accordance with the requirements of the multidisciplinary team P16241 and Dr. Raymond Ptucha.

MANAGEMENT CERTIFICATION - Please check the appropriate statement.

_____ The document is accepted.

_____ The document is accepted pending the changes noted.

_____ The document is not accepted.

We fully accept the changes as needed improvements and authorize initiation of work to proceed. Based on our authority and judgment, the continued operation of this system is authorized.

_____ _____
NAME DATE
Dr. Raymond Ptucha

_____ _____
NAME DATE
Joe Hudden

NAME
Michael Blachowicz

DATE

THEORY OF OPERATIONS

TABLE OF CONTENTS

1.0 GENERAL INFORMATION

1.0 GENERAL INFORMATION

Autonomous vehicles are currently being developed to improve roadway safety and optimize traffic flow. The Rochester Institute of Technology wishes to re-enter the field of autonomous driving by converting a low speed golf cart into an autonomous people mover. An autonomous people mover is a vehicle that transports people in a safe, fast, and comfortable way and requires no human interaction in normal operating circumstances. MSDII Team P16241 has just completed their work on a remote control, and autonomous golf cart with basic functionality.

The scope of this project includes creating a computer controlled vehicle that can take input from a remote control and also make it a fully-autonomous vehicle capable of driving a course and avoiding basic obstacles. The safety of passengers and bystanders are the greatest concern, thus the vehicle has been programmed to drive conservatively, and has emergency stop features.

1.1 System Overview

The autonomous people mover has a few major subsystems. The first major subsystem is the control system. The controls subsystem uses both mechanical and electrical components in order to make the golf cart move. The subsystem has three main categories, braking, throttle, and steering. Power is the next major subsystem. There are two different batteries and voltage converters that are used in order to power all of the other systems on the golf cart. The third major subsystem is the sensors. The sensors are used to collect data to help determine the position of the golf cart, the location of objects, and more. The main sensors that are used for the golf cart are LIDAR, ultrasonic sensors, GPS, encoders, and an IMU. The next major subsystem is the software. All of the golf carts decision making, and data acquisition is done using the Robot Operating System (ROS) and Arduinos. The last major subsystem is the electrical modifications. This includes the computer, PCB, Arduino shield, and the enclosure.

1.2 Organization of the Manual

The user's manual consists of eight sections: General Information, Controls Summary, Safety Summary, Power Summary, Sensor Summary, Software Summary, Electrical Modification Summary, Getting Started, and Reporting.

General Information section explains in general terms the system and the purpose for which it is intended.

Controls summary section provides an overview of how each system works. The documentation needed to understand both the hardware and software components used.

Safety summary section provides an overview of the various safety systems that are in place or should be in place to ensure that the passenger is always as safe as possible.

Power summary section provides an overview of how all of the electronics on the golf cart are powered. This documentation is needed to understand what components are being used and how they are connected.

Sensors summary section provides an overview of why each sensor was chosen, where they are mounted, and how each sensor works.

The software summary section provides an overview how the software (ROS and Arduino code) works and the overall flow of the program.

The electrical modification summary section provides an overview for the electrical components inside the enclosure.

Getting started explains how to run the APM. This includes how the GUI and other inputs of the golf cart are intended to be used.

The reporting section describes what information was collected throughout the project.

1.3 Project References

Golf Cart Documentation
https://www.yamahagolfcar.com/download/2005/golf_car/2005_-_Golf_Cars_-_GMAX_Electric.pdf

P15242 Edge Site
http://edge.rit.edu/edge/P15242/public/Home

P16241 Edge Site
http://edge.rit.edu/edge/P16241/public/Home

GitLab Code Repository
https://kgcoe-git.rit.edu/autonomous-golf-cart/golf-cart

1.4 Acronyms and Abbreviations

PWM	Pulse Width Modulation
APM	Autonomous People Mover
ROS	Robot Operating System

2.0 CONTROLS SUMMARY

2.0 CONTROLS SUMMARY

Controls summary section provides an overview of how each system works. The documentation needed to understand both the hardware and software components used.

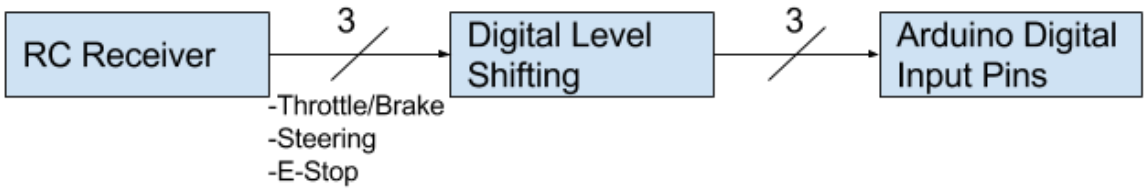
2.1 Subsystems Summary

The system is comprised of three subsystems, braking, throttle and steering.

2.2 Remote Control

In order to achieve remote control functionality, a wireless remote controller was used. The remote control system includes a remote control and a receiver. The receiver is mounted in the enclosure, and interfaces with one of the Arduino Dues. Signals for throttle, steering, brake, and emergency stop are received from the remote control.

2.2.1 High Level Overview

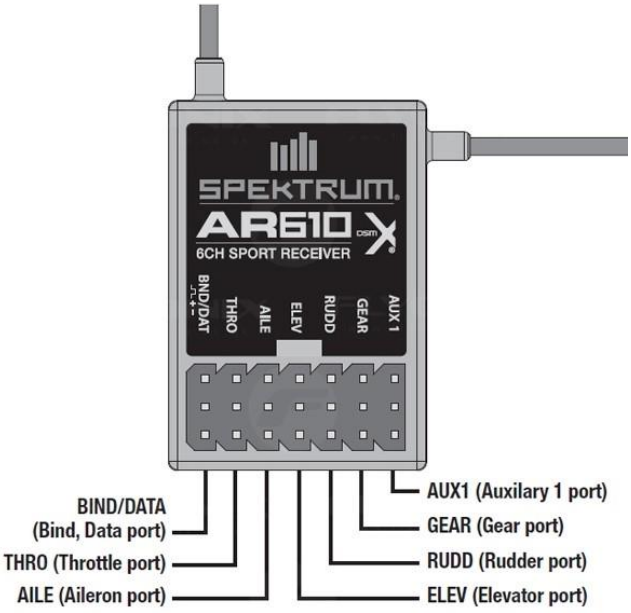


2.2.2 Hardware

The remote controller used on this project is the Spektrum DX6i. This remote controller is meant for RC planes. The user manual can be found here.

Spektrum DX6i

The receiver used for this project is the Spektrum AR610. The receiver gets 1ms to 2ms PWM signals from the controller. The receiver is pictured below with a description of what each output does. The output signals are 5V. Only one 5V input and one ground need to be plugged into power the receiver. All of the other wires just need to be the signal wire to the microcontroller.



THRO uses the left joystick and it controls the up/down motion
AILE uses the right joystick and controls the left/right motion
ELEV uses the right joystick and it controls the up/down motion

RUDD uses the left joystick and it controls the left/right motion
GEAR uses the gear switch on the left side of controller and is an on/off switch
AUX1 uses the flap/gyro switch near the left joystick and is an on/off switch

Currently, GEAR is used for the E-Stop, AILE is used for steering, and ELEV is used for the throttle/brake. On the jumpers, brown is ground, red is power, and yellow is signal. This information is critical when connecting the receiver to the PCB.

2.2.3 Software

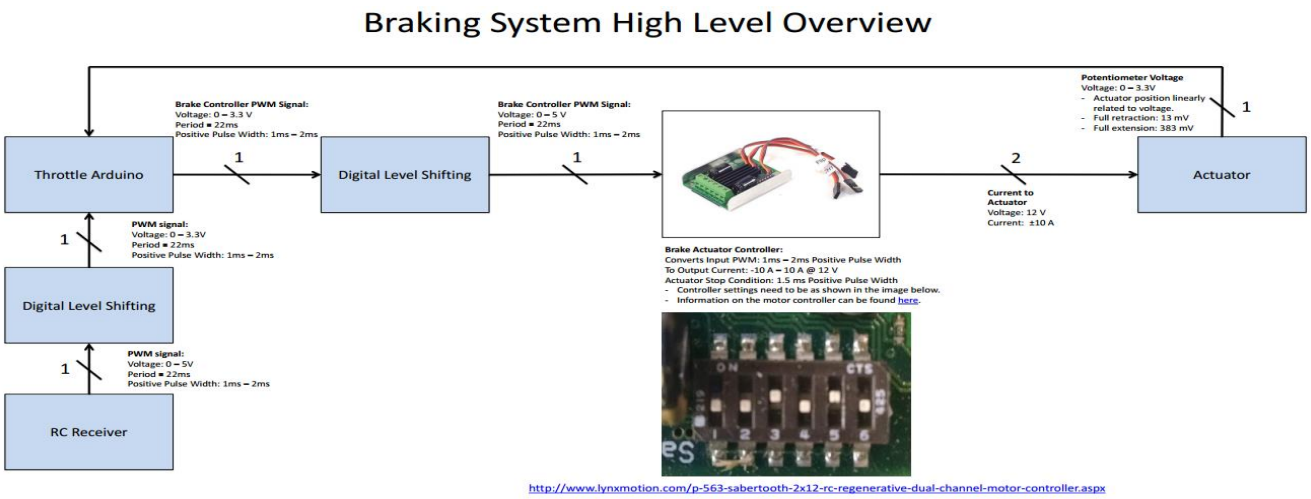
In order to interpret the PWM signal from the RC receiver, the corresponding Arduino pins were set to digital input, and interrupts were attached to each pin. Measuring the time between rising and falling edges allows for the respective duty cycle of the signal to be calculated. This duty cycle varies from 1ms to 2ms.

2.3 Braking

To implement computer-controlled braking an actuator was used to pull a steel cable that was connected to the brake pedal. Due to the size of the actuator it required a pulley to make a 90° turn before connecting to the brake pedal. The actuator and the pulley were mounted to the golf cart frame. This method allows for the passenger in the driver's seat to still be able to hit the brake in an emergency situation. For feedback, the internal potentiometer of the actuator was used. The potentiometer gives feedback to the Arduino so that it can control the position of the actuator. A Hall effect sensor was mounted to sense when the brake pedal was depressed and tells the APM to enter an emergency stop state, though it has since broken and has not been integrated into the braking system.

2.3.1 High Level Overview

The high level overview gives a basic break down on how each signal is sent to all of the components of the braking system.

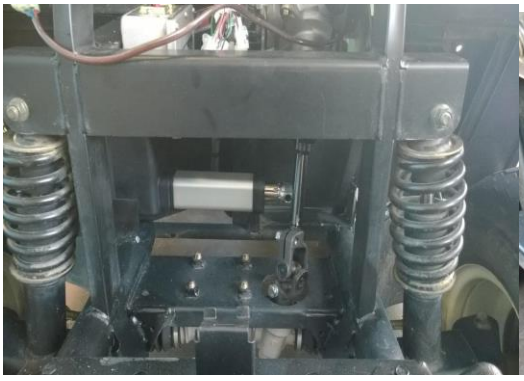


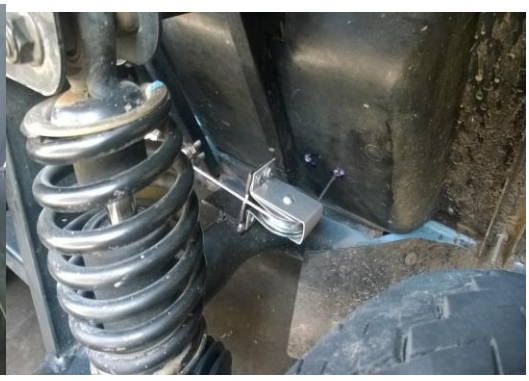
Braking System High Level Overview

2.3.2 Hardware

The braking system was designed to allow the brakes to be used as originally designed without adding any additional risks of failure. This was done by running a steel cable through a hole drilled in the steel pedal, through the plastic divider "firewall", around a pulley and to a linear actuator mounted under the hood of the cart. The pulley has a sheet metal cover to prevent the cable from coming off when there is slack in the cable. Previous phases had attached a small piece of foam to the brake pedal which was intended to activate a switch when the brake was pressed manually. It is unclear how this was intended to work and is not currently being used. The E150 linear actuator user manual can be found here and is pictured below.

E150 Linear Actuator





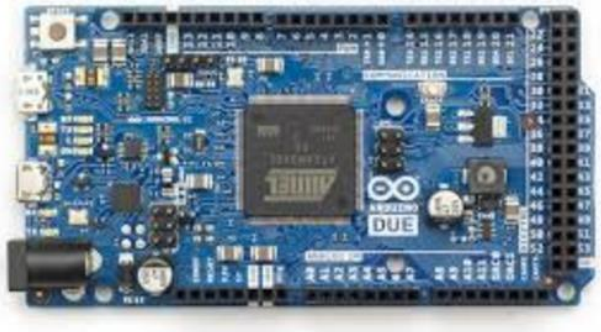
Actuator and Pulley

Brake Pedal

The braking system is controlled using a Sabertooth 2x12 R/C Regenerative Dual Channel Motor Controller. This system requires a 12V input as well as a 1ms to 2ms-pulse width input signal provided by the Arduino. A user manual for the Sabertooth R/C Regenerative Dual Channel Motor Controller can be found here.

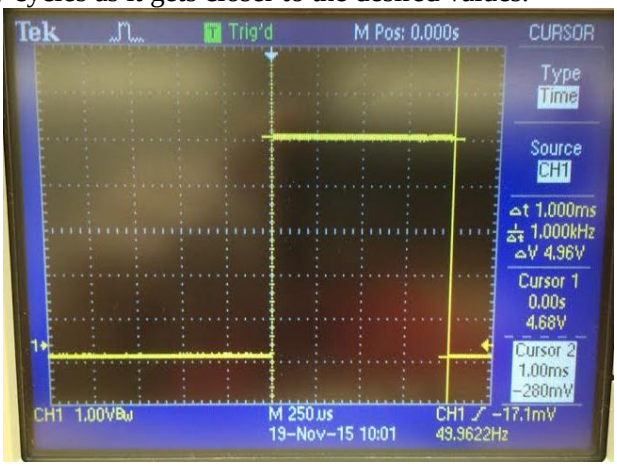
Sabertooth 2x12 R/C Regenerative Dual Channel Motor Controller

The Sabertooth Motor controller receives all of the PWM signals from an Arduino Due. Any information on the specification of an Arduino Due can be found here. An important note about the Arduino Dues is that they cannot output or receive 5V signals. This is important for the braking system due to the fact that the Sabertooth uses a 5V PWM signal to control the braking actuator. A level shifting circuit is being used in the main PCB board to account for this and it is noted above in the high level overview.

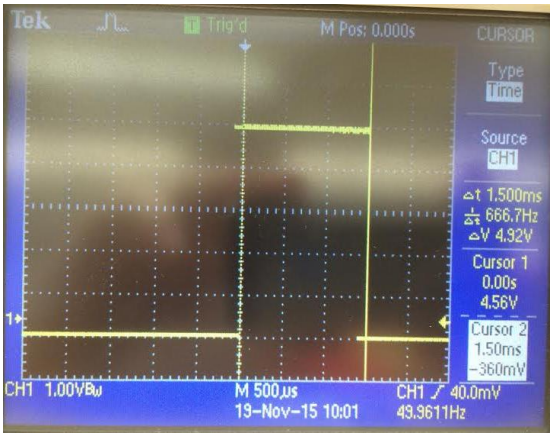


2.3.3 Software

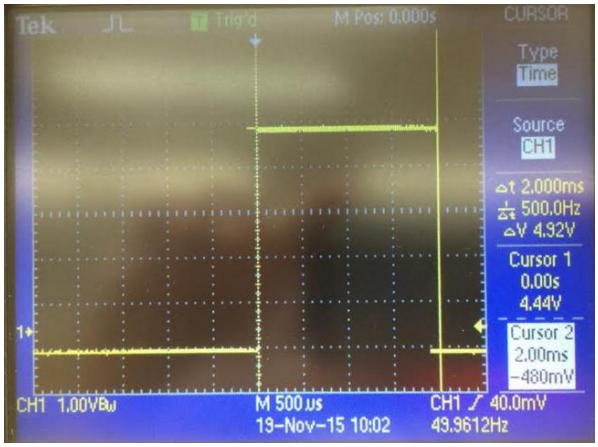
The software for braking is implemented like all of the subsystems as a library which can be imported into a program requiring control of the braking. Examples of its use can be seen in any of the braking test programs, or ‘main_due1’. How it works is by sending a pulse-width signal to the Sabertooth 2x12RC between 1ms and 2ms, for extend and retract respectively. A 1.5ms pulse is used to do nothing. By taking input from the internal potentiometer in the brake actuator, the braking system decides to extend or retract the actuator to reach its desired position. To avoid over and undershoot causing oscillations, the subsystem introduces delay cycles as it gets closer to the desired values.



PWM output to extent actuator



PWM output to do nothing



PWM output to pull the actuator

These signals are controlled in the Arduino code by using the built-in 'servo' object because the 50Hz 1-2ms pulse is how a servo is driven.

2.3.4 Mounting

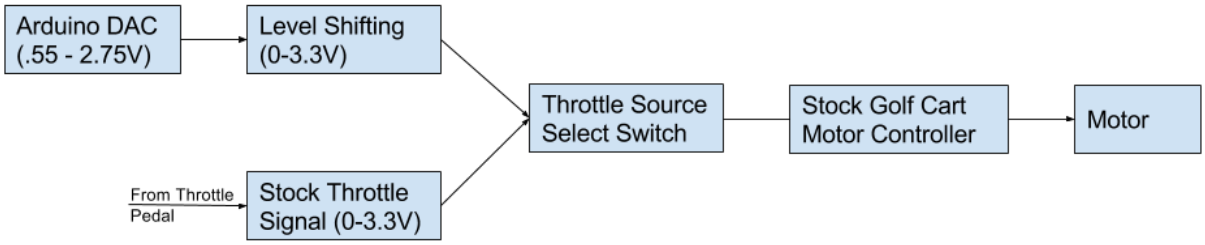
The brake actuator bracket and pulley were welded onto the frame of the APM. Unfortunately, the linear actuator was larger than expected. This created two issues with the current braking system. The actuator does not have enough room to extend fully and will crash into the bracket for the pulley. This is currently mitigated by limiting the range of travel of the actuator in software. The other issue is that the front right tire comes into contact with the brake actuator mounting bracket when the wheels are turned fully to the right. Again, this is mitigated by limiting the range of motion of the steering in software. A long term solution to these problems would be to re-mount the actuator in a position or orientation with more space.

2.4 Throttle

In stock configuration, the golf cart uses a potentiometer on the throttle pedal to generate a signal to the motor controller. The potentiometer varies from 0V (no throttle) to 3.3V (full throttle). In order to control this signal with software rather than the golf cart pedal, this signal was passed through the PCB. Two 23-pin Ampseal receptacles were added to the PCB so that the stock connector would plug into the PCB, and the PCB would pass through most of the other signals, while substituting the throttle signal with the desired value determined in software. This output comes from one of the DACs on an Arduino Due. Additionally, a switch was added to the PCB so that the throttle could operate in stock mode as well as software controlled mode. This allows the cart to be driven in entirely stock mode with the flip of a switch.

2.4.1 High Level Overview

The high level overview gives a basic breakdown on how each signal is sent to all of the components of the throttle system. Functionality can be broken down into two main modes: stock mode, and software controlled mode. Stock mode simply passes the throttle signal through as if the PCB didn't exist. In software mode, the throttle signal can come from one of many sources, depending on the mode that the cart is operating in. In manual mode, the Arduino reads in the encoded position of the throttle pedal, and outputs the correlated voltage on the throttle DAC. This allows the cart to be driven in manual mode without opening up the enclosure and flipping throttle source select switch. In remote mode, the throttle value is determined by relating the percentage of throttle stick displacement on the remote control to the desired throttle percentage. For example, when the remote control throttle stick is pushed 50% forwards, the Arduino outputs 50% of the throttle signal which is 1.65V. Finally, in autonomous mode, the desired throttle value is determined by the high level software in ROS and sent serially to the Arduino Due.



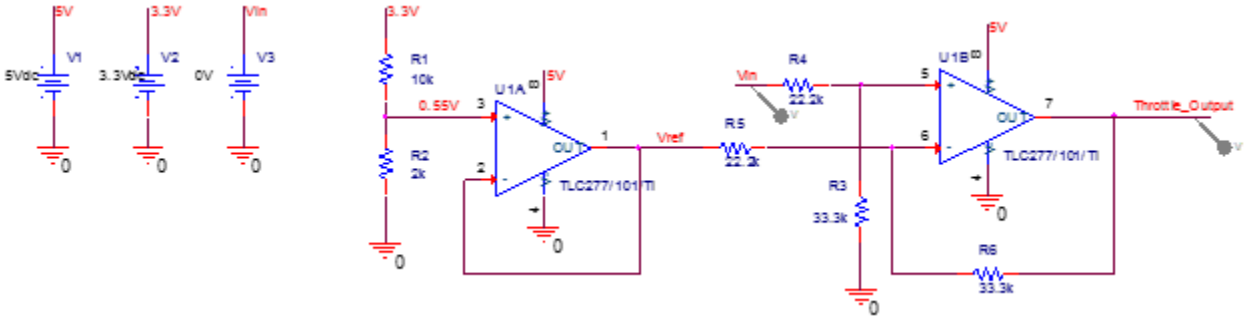
Throttle System High Level Overview

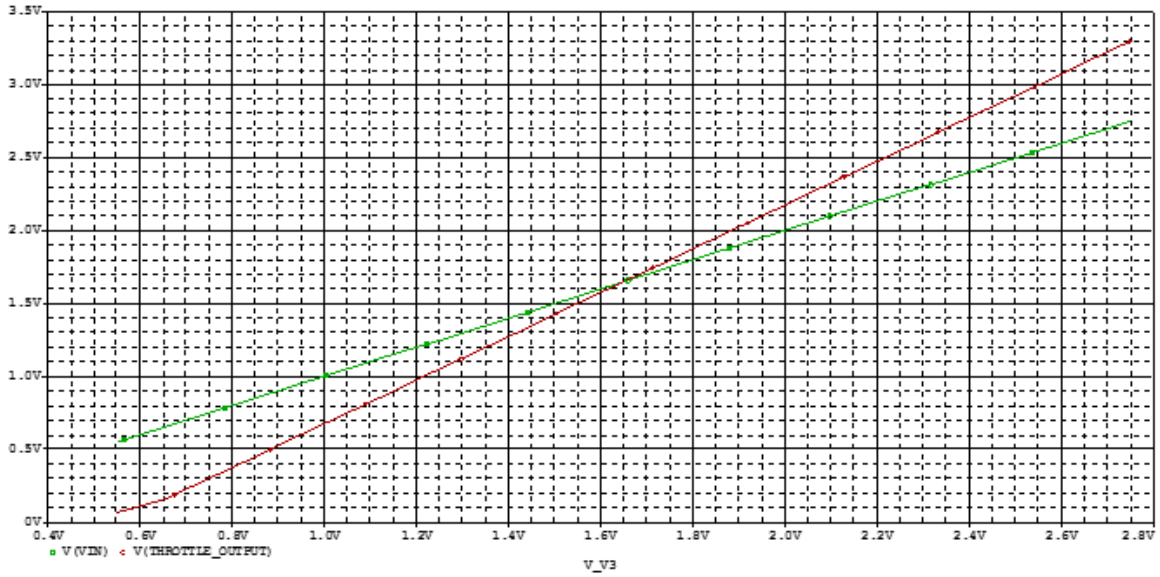
2.4.2 Hardware

The golf cart motor is located in the back of the golf cart. This is the stock motor that comes with the Yamaha G22E golf cart. It is a JU2-H1890-10 Advanced D.C. Motors motor. It works at 48 Volts DC and it is has a class H rating. The motor has a rating of AU25000 3.5HP and a 2.6KWatt per 30 minutes.

Golf Cart Motor

For the throttle subsystem, Arduino Dues are used in order to control the speed. The outputs used from the Dues are the DAC outputs. An issue with this setup is that the output of the Arduino Due DACs only range from 0.55V to 2.75V, and the stock system expects a signal from 0-3.3V. The following op-amp circuit was designed to scale the voltage from the DAC to the required range of the throttle system. It consists of a voltage divider and buffer, as well as differential configuration op-amp circuit. Please see the APM_voltage_shifting_overview.pdf document for a full explanation of this circuit including transfer equations.





Level Shifting Circuit for Throttle

2.4.3 Software

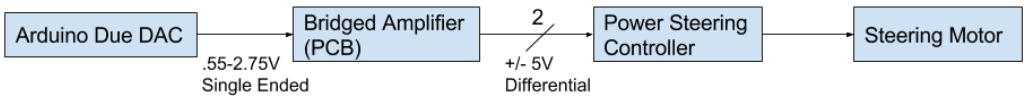
The software for throttle is implemented like all of the subsystems as a library which can be imported into a program requiring control of the braking. Examples of its use can be seen in any of the throttle test programs, or 'main_due1'. How it works is by being given a value between 0 and 100 representing 0% throttle and 100% throttle. This then becomes an output from a DAC on due1 which gets shifted as described above.

2.5 Steering

In order to achieve computer controlled steering functionality, a power steering system was installed on the cart, and the electronics were modified to allow computer control. The standard configuration of the power steering system contains a motor to turn the steering column, a torque sensor to detect torque on the wheel (normally by the driver), and a controller to interpret the torque sensor signal and drive the motor in an assisting manner to the driver. This control loop was modified so that the torque sensor output was replaced with a computer controlled signal, allowing computer control of the power steering system.

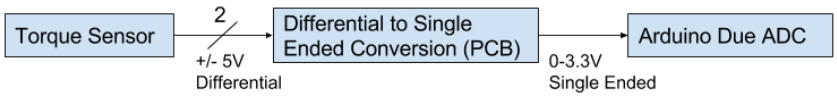
2.5.1 High Level Overview

The following figure shows the high level signal flow for the steering subsystem. The desired position for the steering system is determined in software, and is sent out by an Arduino DAC.



Steering Control Circuit

The output of the torque sensor is also conditioned and sent to an Arduino. Currently, nothing is done with this signal. Eventually, it may be monitored to detect human interaction with the steering wheel. This could be a useful safety feature that would switch from autonomous to manual mode when a passenger interacts with the steering wheel. The following figure shows the signal flow for the torque sensor circuit.

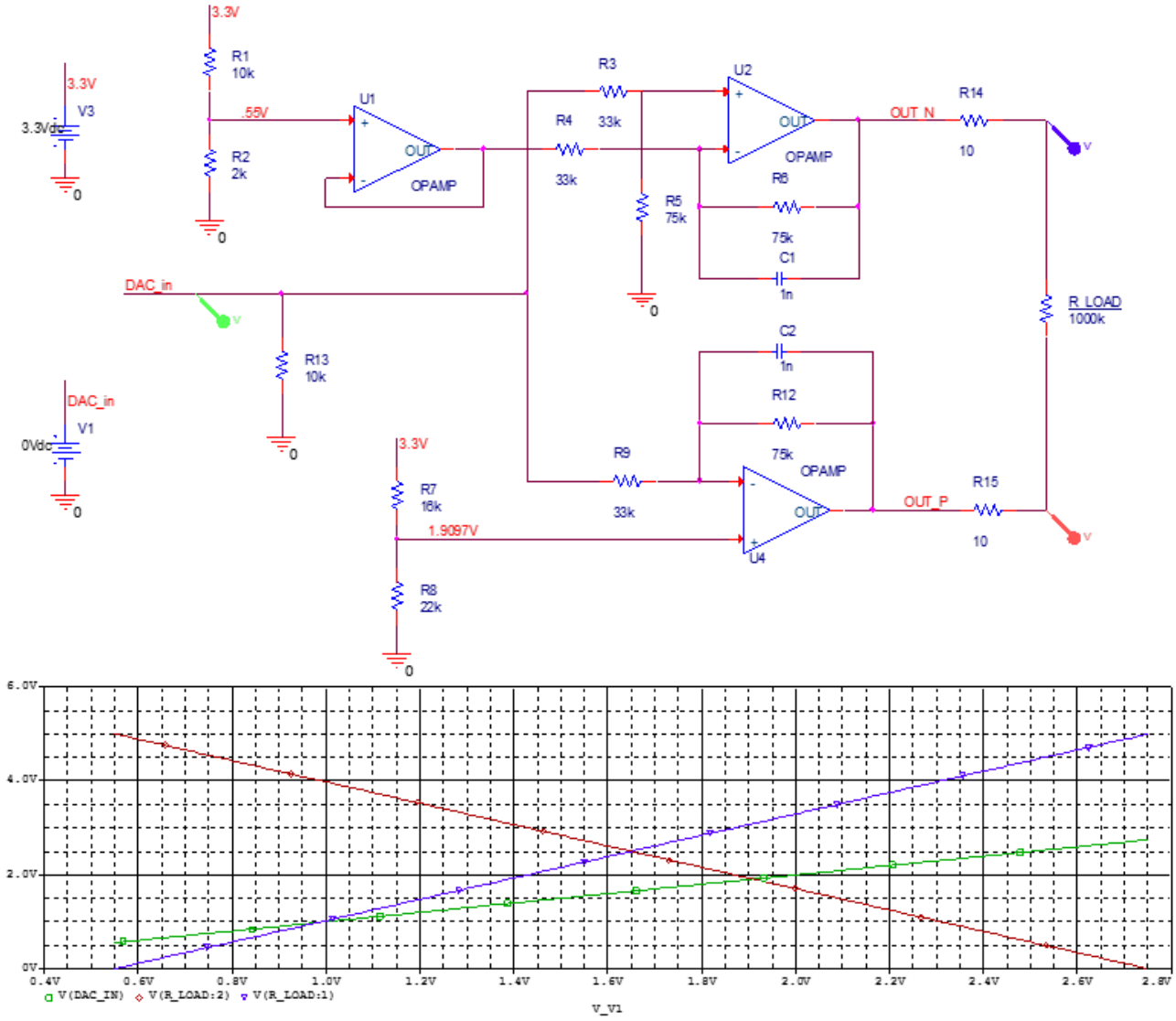


Torque Sensor Circuit

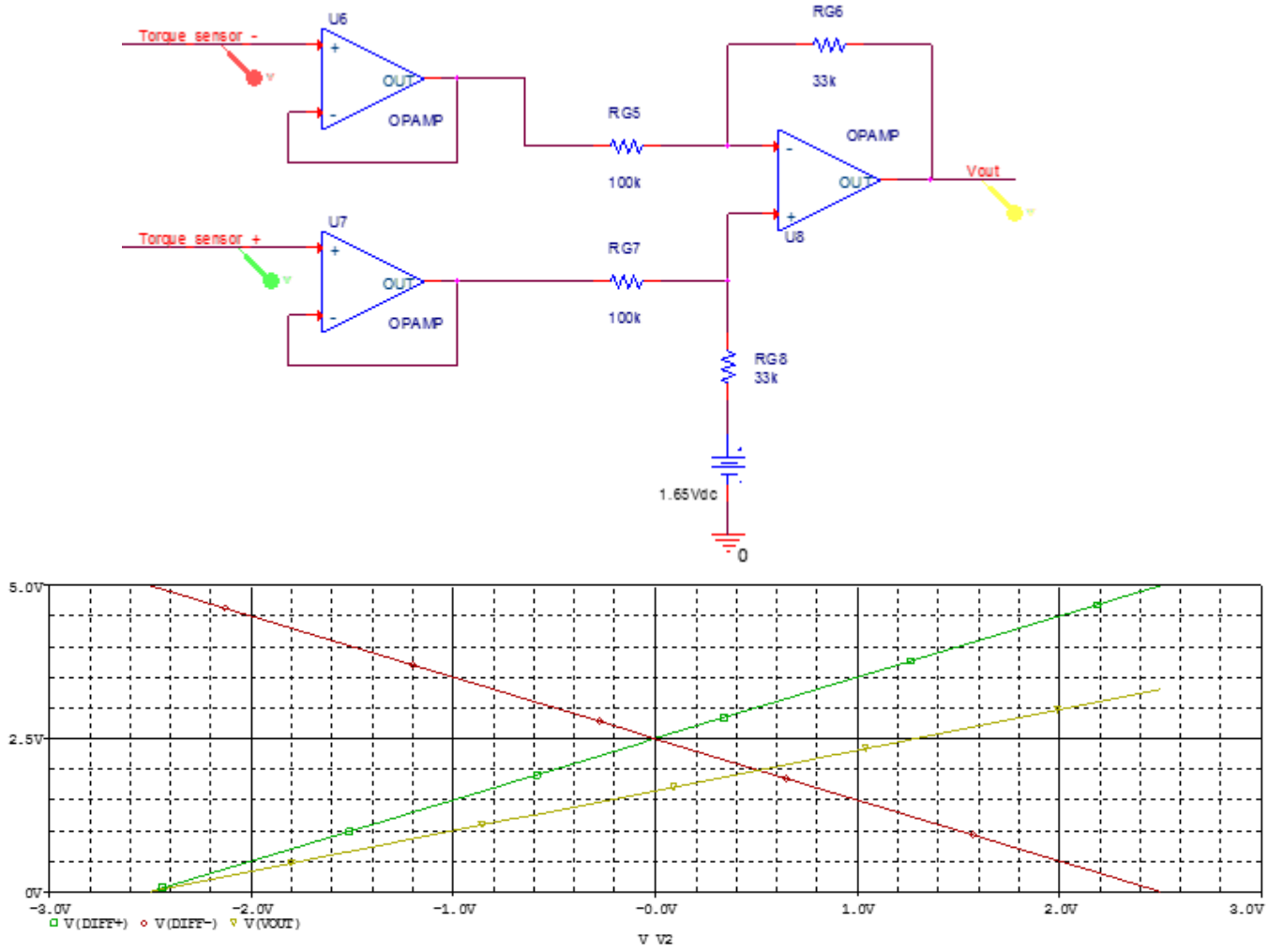
2.5.2 Hardware

The hardware used for this electric power steering setup comes from Wickedbuilt. Their UTV retrofit system utilizes 5V differential voltages to control motor torque and direction. Using the built in torque sensor and adding a chain driven potentiometer, steering shaft angle can be tracked at all times and torque can be managed via a PID feedback loop to achieve desired steering angles with appropriate amounts of smoothness.

The following figure shows the bridged amplifier circuit used to convert the single ended output of the DAC to the differential signal required by the power steering controller. As the DAC varies from .55V to 2.75V, the output of the amplifier varies from 5V,0V, to 0V,5V.



For the torque sensor circuit, a modified instrumentation amplifier style circuit was designed. This was required because during testing, it was noticed that the torque sensor circuit was sensitive to the impedance at the sensing end. The following figure shows this circuit.



Torque Sensor Circuit

2.5.3 Software

The software for steering is implemented like all of the subsystems as a library which can be imported into a program requiring control of the steering. Examples of its use can be seen in any of the steering test programs, or 'main_due1'. How it works is by being given either a desired potentiometer value, desired steering angle, or desired steering percentage. Due to noise on the line in addition to the low-pass filter on the physical line there is a software-implemented averager on the potentiometer position.

2.5.4 Mounting

Mounting brackets for the power steering motor and potentiometer were welded onto the frame of the APM. The motor control box was mounted directly to the frame. The potentiometer is linked to the steering column with a small bicycle chain.

2.6 Computer

The golf cart contains a computer from the Computer Engineering department to function as a ROS host computer and provide processing power for autonomous functions.

2.6.1 Hardware

The computer has several components that are mounted on the cart, described below. Questions about hardware can be directed to the Computer Engineering Lab Manager, Rick Tolleson (rateec@rit.edu).

- Motherboard - This is the main processing board on the computer, mounted in the upper left of the enclosure. It is powered from the Computer PSU using a 4-pin CPU power cord and a 24-pin ATX power cord. It has connections to the SSD, LCD monitor, Arduinos, RIT internet, internal LAN, Wi-Fi Adapter, and the power switch.
- LCD Monitor - This is the screen for the computer, mounted in the dashboard. More information is detailed in the Dashboard section. It connects to the motherboard with HDMI and USB
- Arduinos - The Arduinos connect to the motherboard through a USB hub on the back of the panel.
- SSD - The computer's storage drive is a solid state drive mounted on the side of the enclosure. It is powered with a SATA power cable from the computer PSU and communicated with a SATA data cable to the motherboard.
- RIT Internet - The motherboard has an onboard Ethernet connection next to the headphone jacks that is configured to be connected to RIT's internet exclusively. Inside Cube 2, there is one wall outlet with Ethernet jacks. It is configured to be plugged into the left jack on the wall.
- Internal LAN - An additional network card is present on the computer for communicating with the LIDAR and cameras on an internal LAN through a switch. This cannot be connected to the RIT internet.
- Ethernet Switch - Connects the motherboard, LIDAR, and eventually cameras together into one network so that they can communicate. The order in which Ethernet cables are connected to the switch is irrelevant. It is powered through a 12V supply
- Computer PSU - This PWR-M4-ATX power supply is a 250W 12V DC power supply for the computer, which converts the 12V input to several 12V, 5V, and 3.3V rails for the computer. It is powered on through the motherboard. For testing purposes, the power supply can be turned on by unplugging the 24-pin ATX power supply and jumping the green wire to ground. It provides a USB diagnostics tool for the motherboard, which connects to one of the USB headers on the board. Details about implementation can be found on MiniBox's website.
- Power Button - The Computer Power button connects directly to the +PW- header pins on the motherboard. It provides the same wiring function as the power button on the front

of a standard computer case.

- Wi-Fi - The Wi-Fi adapter is connected to the motherboard via USB headers, and is mounted on the left of the enclosure.

2.6.2 Software

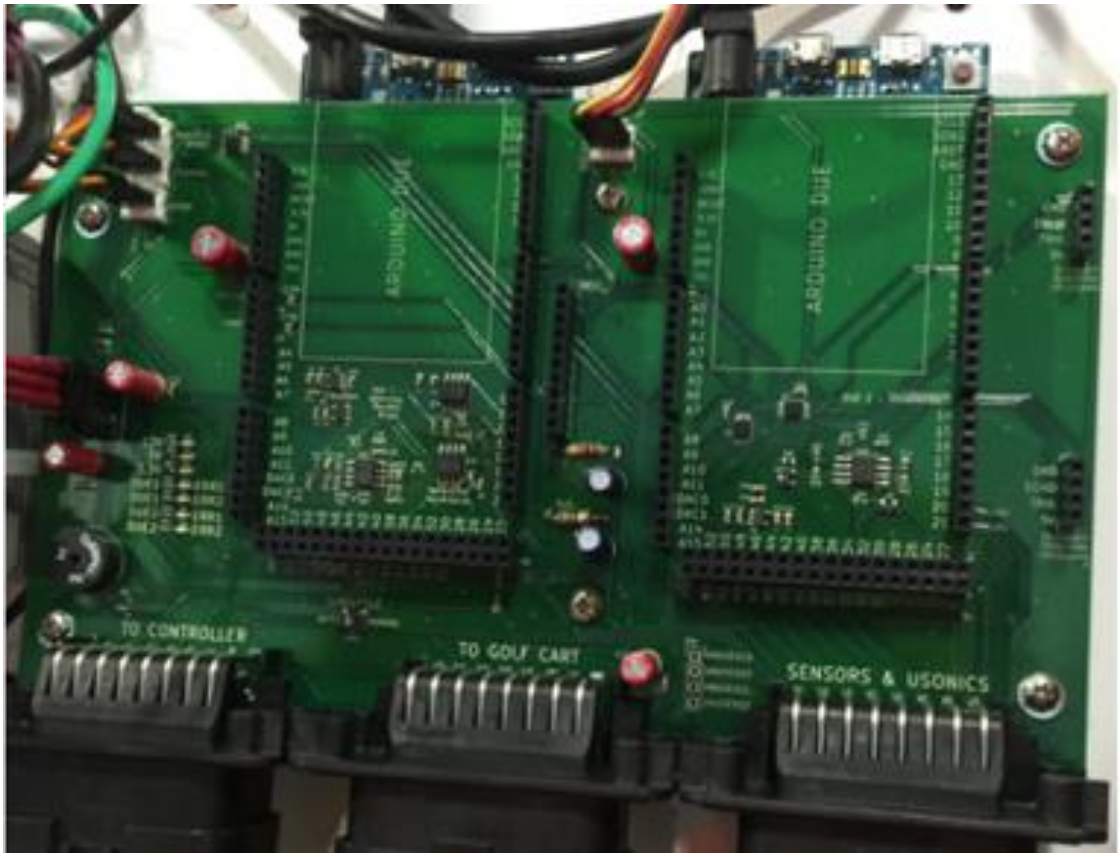
The computer runs Ubuntu 14.04 for an operating system. As the computer is a KGCOE lab computer, the system is supported by the system administrators. Contact the Computer Engineering systems administrator, Emilio Del Plato (ehdeec@rit.edu) for software support. If there is an issue with the system, the solution will usually be to reformat the computer and start fresh. This can be done by the system administrators. In this case, the computer will need to be reconfigured for the purposes of the cart. In the git repository, the Computer/Config file describes the various configuration and installation steps, as well as several pitfalls.

2.7 Printed Circuit Board (PCB)

A custom printed circuit board was designed and fabricated to accomplish all digital and analog signal conditioning, as well as providing the interface from hardware to software. Various voltage shifting and scaling circuits were required for steering, brake, throttle, remote control, ultrasonics, and safety systems. All circuits were first designed and simulated. Design considerations as well as simulation results can be found here: VoltageShiftingAndScaling.

2.7.1 Hardware

In order to interface with the stock golf cart, two 23-pin Ampseal receptacles were added to the PCB so that the stock connector would plug into the PCB, and the PCB would pass through most of the existing signals, tap onto the VTACH signal, and optionally replace the throttle signal. An additional 23-pin Ampseal connector was added to monitor and drive the remaining systems through the PCB. The following image is a top-side view of the finished PCB.



Printed Circuit Board

Combination male/female header pins are used to mount the Arduino Due's onto the bottom of the PCB. This allows for a solid connection to be maintained between the PCB and the Arduino, and enables easy probing of inputs and outputs. Currently the Arduino on the left is referred to as Due 1, and is used for throttle, steering, brake, and E-stop. The Due on the right is referred to as Due 2 and controls the ultrasonics and IMU. For further reference, and for the actual pinout, please see the PCB schematic in the KiCAD directory. This directory contains all documentation and design files used in the development of this board, including custom created KiCAD modules for several of the components.

2.7.2 Software

The PCB contains a single sounder, and four error LEDs. The sounder is connected to pin 23 of both Arduinos in a current sink configuration. So to enable the sounder, either Arduino can drive its pin 23 low. Each Arduino then has two error LEDs driven by pins 25 and 27. This enables a quick feedback mechanism for indicating issues or for debugging Arduino code.

3.0 SAFETY SUMMARY

3.0 SAFETY SUMMARY

3.1 Subsystems Summary

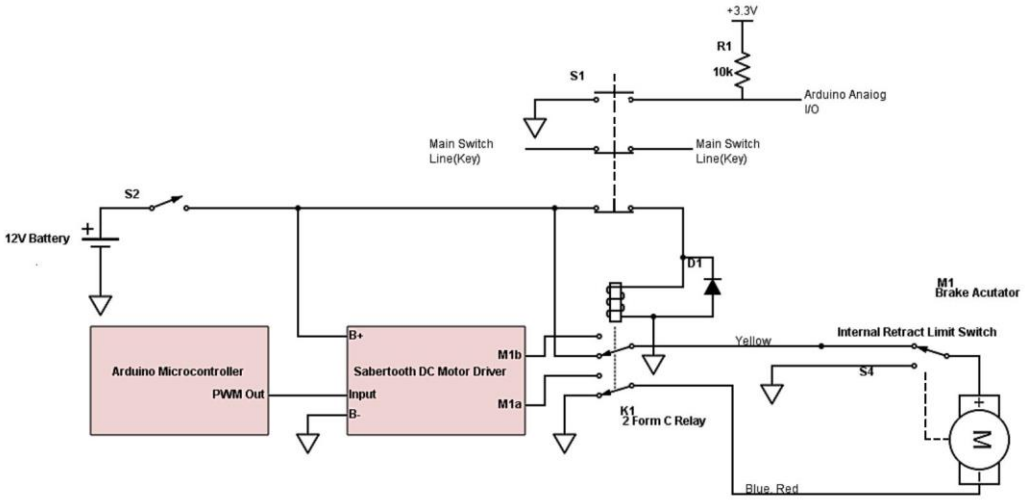
The objective of the safety systems is to ensure the wellbeing of the passenger, APM and surrounding pedestrians. The current system includes an emergency stop, pedestrian notification and occupant monitoring. The emergency stop system provides the passenger a fail-safe method of stopping the vehicle. Pedestrian notification utilizes proximity information from the LiDAR to alert surrounding pedestrian using lights and a sounder. Occupant monitoring system utilizes dual ultrasonics to detect whether there are occupants in the vehicle. This system permits the APM to stop at the disappearance of a passenger and allow re-entry.

3.2 Emergency Stop

3.2.1 High Level Overview

The Emergency Stop (E-stop) system provides to the passengers a fail-safe method of stopping the cart. The design of the system utilizes a push button to trigger three actions. The first action signifies to the system that an emergency stop has been activated. The second action causes a break in the "Main Switch Line" (Key) which disconnects power to the APM motor. The third action engages the brake actuator to stop the cart and prevent any further movement. These actions ensure that the APM and its passengers are in a safe state in the case of software error or hardware failure. The design of the E-Stop utilizes a hardware-only approach to engage each action. This approach minimizes reaction time and isolates the system such that failure in other components of the cart won't affect the reliability of the E-Stop.

3.2.2 Wiring Diagram



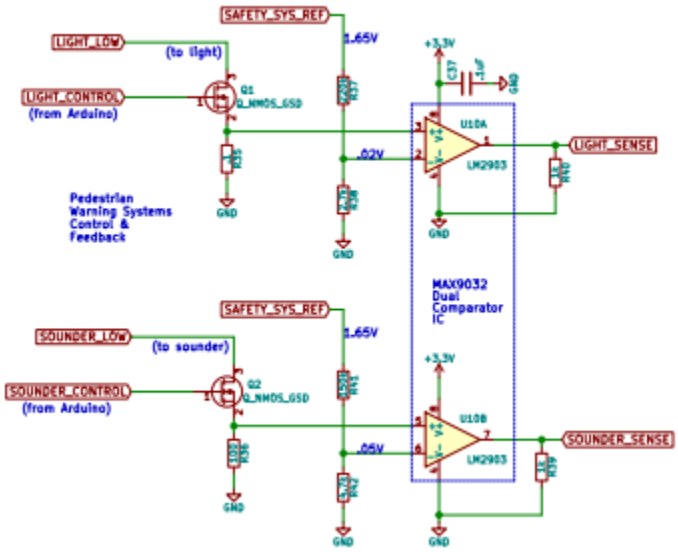
The above figure shows the a wiring diagram of the E-stop system. S1 is the emergency stop button(dashboard) which controls 3 mechanical relays. Two are normally closed while one is normally open. S2 is a switch which control power to the steering and brake actuator. This switch can be found in the rear panel. K1 is a power relay which switches control of the brake actuator from the Arduino Due to +12v resulting in the activation of the brakes.

3.3 Pedestrian Notification System

3.3.1 High Level Overview

The Pedestrian Notification System was added to notify pedestrians of the APM's presence when it is driving in autonomous mode. The system consists of a light and sounder to notify pedestrians when they are within a certain distance from the APM. Currently the light is being triggered whenever the APM is in autonomous mode. The sounder is being triggered whenever an object is within the stop field. Ideally the light or sounder would turn on once a pedestrian is detected close to the APM. Since there are not currently cameras on the cart, there is no way to differentiate a pedestrian from any other object detected. The flash pattern of the light can be changed by touching the blue wire to ground for a second.

3.3.2 Wiring Diagram



The figure above shows the circuits used to control the light and sounder. Since both run off of 12V and the Arduinos cannot supply 12V, a MOSFET was used as a switch to turn on and off the light or sounder. The gates of the MOSFETs were driven by digital I/O pins on the Arduino. There was also a feedback circuit to determine if the light or sounder were turning on when they should. The resistors on the source side of each MOSFET were chosen so that the intensity of the light or sounder would not decrease too much, and would produce a large enough voltage for the input to the comparators.

3.4 Occupant Monitoring System

3.4.1 High Level Overview

The ability to monitor the occupants in the APM allows for safe and efficient operation of the cart. This system provides feedback to the control system regarding whether there are passengers in the cart and allows the vehicle to halt when certain unexpected scenarios are encountered. The scenarios can include passengers exiting while vehicle is in motion. If this was to occur, the APM can briefly pause operation to allow re-entry or proceed to next task if re-entry doesn't occur. The occupant monitoring system on the APM utilizes dual ultrasonics aimed at the seat of the vehicle. The ultrasonics are able to cover the entire span of the seats from the far left to the far right. The control system will periodically poll the ultrasonics to read the distance of objects from the sensors. The sensors are calibrated by obtaining readings when the seats are empty. A threshold value is included to ensure bags or boxes are not interpreted as a passenger. Statistical values including average and standard deviations are used in the interpretation algorithm.

This system is currently incomplete and required further work. Some progress have been made towards the implementation. Two ultrasonic sensors (JSN-SR04T) have been selected and purchased for this purpose. Multiple mounting solutions have been tested. However, the readings from the sensor are inconsistent. Further testing is needed to determine the best mounting position for the sensors.

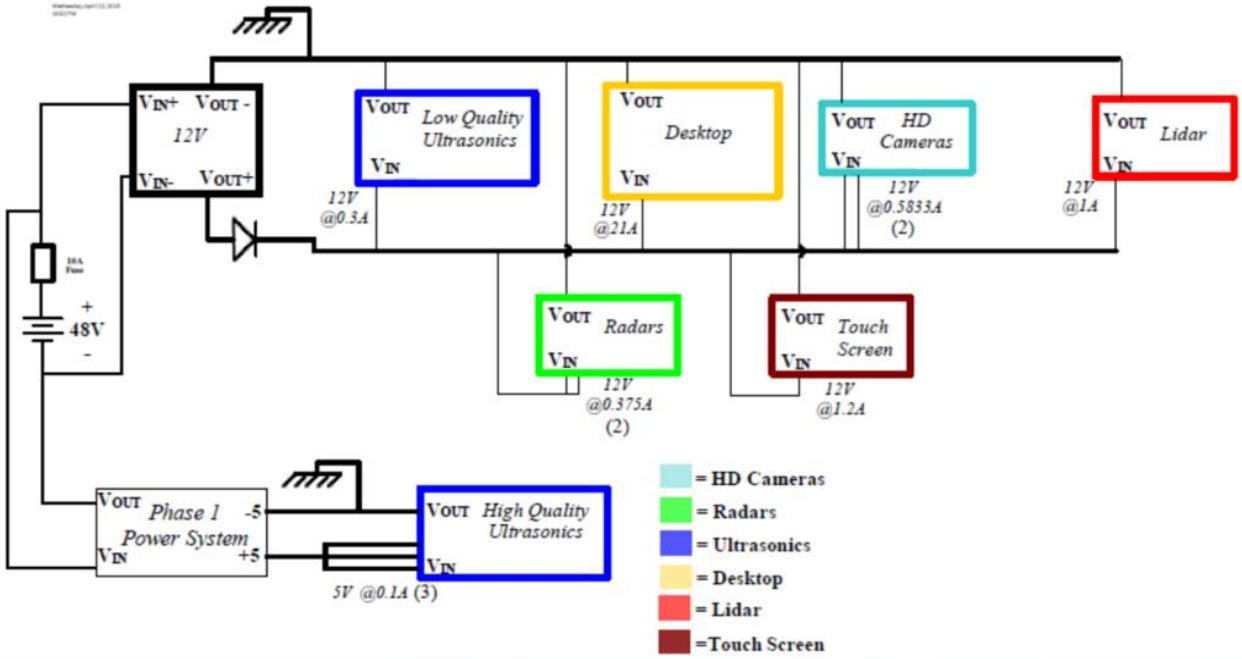
4.0 POWER SUMMARY

4.0 POWER SUMMARY

The power system provides 12 volt, 5 volt, and 3.3 volt power to the electrical components on the cart. This is done by the use of various voltage converters and two different battery setups.

4.1 High Level Overview

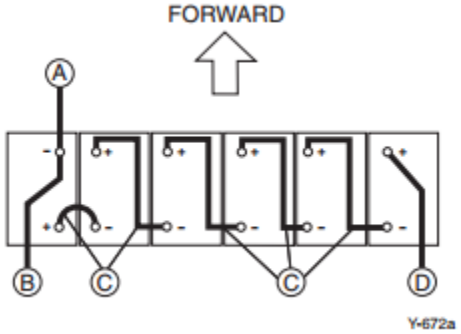
The cart is powered by a 48 Volt battery bank which is reduced to 12 Volt, 5 Volt, or 3.3 Volts as needed. Between the sensors and the desktop, 202 watts are required at 15 Volts, with the desktop requiring 145 watts. This works out to a current draw of 17.12 amps. The CUI Inc. VFK600 Series 48 Volt to 12 Volt DC-DC converter provides 50 amps at 12 Volts, more than enough to run the existing equipment with room to expand in the future. Live wires have red insulation exclusively; ground wires have black insulation exclusively. The 12 volt battery is used to power the braking actuator and the steering. There were no tests completed to test the life of the battery. The APM was on for approximately 8 hours at Imagine RIT and the battery lasted the entire time without being recharged.



High Level Overview of Power System

4.2 Hardware

Golf Cart Batteries



http://www.digikey.com/product-detail/en/PYB20-Q24-S3-U/102-3246-ND/4477504

http://www.digikey.com/product-detail/en/PYB30-Q24-S5-U/102-3262-ND/4477520

http://www.cui.com/product/resource/pyb20-u.pdf

http://www.digikey.com/product-detail/en/VFK600-D48-S12/102-2463-ND/2770681

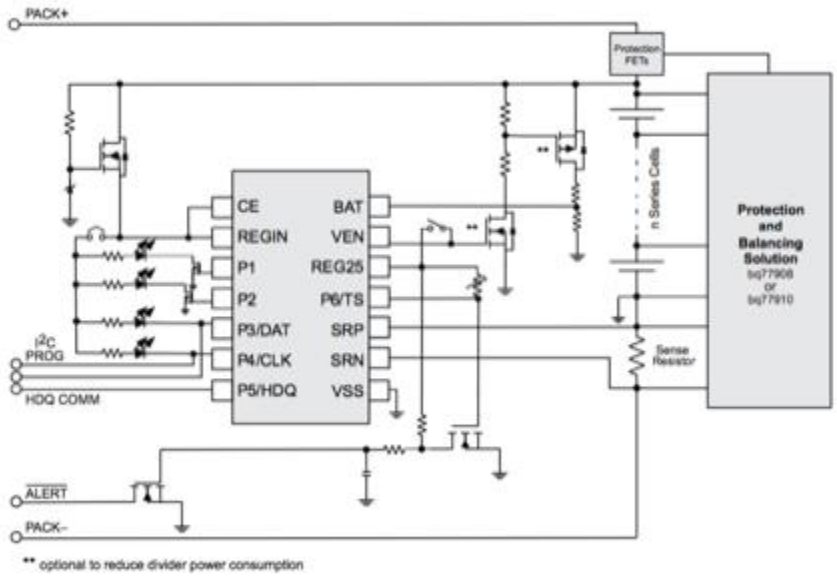
Terminal blocks, 4 inputs, 16 outputs – 2 inputs go to the top 8 outputs, 2 inputs go to the bottom 8 outputs. Top row is used for 12 volt, bottom row is used for ground.

4.3 Mounting

The 12V battery is mounted under the seat of the golf cart. It is resting between the two main chassis rails on a piece of 1/8in aluminum sheet. Holes at the corners of the plate are used by four 3in pipe clamps. To remove the likelihood of longitudinal movement of the 12V Battery, four pieces of aluminum scrap were epoxied to the rails in front of and behind the two forward clamps. Combined with the mass of the battery, these enable the battery tray to remain fairly secure.

4.4 Energy Monitoring

It is recommended that the future team implements an energy monitoring system that is capable of measuring the battery capacity and estimating the time remaining for the batteries on the cart. Two parts have been purchased by the phase III team that can be used to monitor the battery. Notes on these methods can be found here. The first part that could be used is the TI Impedance Track Standalone Fuel Gauge (BQ34Z100-G1). This fuel gauge estimates the capacity of the battery using a patented algorithm that accounts for battery aging and self-discharge. The device can be read from or written to using the Arduino DUE and also has many configurable parts which would allow for information on the battery capacity to be sent to the Arduino where the information can be processed. The simplified hardware implementation for this device is shown in the figure below.



Another energy monitoring part that was purchased was the TF01N. This device is a coulometer that measures battery voltage, current and power. The voltage and current measured have +/- 1% accuracy. The measured values are displayed on a small LCD screen. The screen displays the remaining battery capacity, battery voltage, battery current and remaining time of charging/discharging. This device also

has an output pin so the information can be sent into an Arduino for additional monitoring. For more information info on the device go to: TF01N

5.0 SENSORS SUMMARY

5.0 SENSORS SUMMARY

5.1 Subsystems Summary

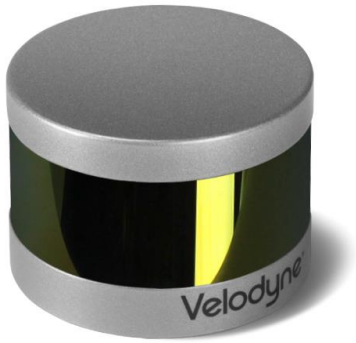
The sensors included in this project to date are the Velodyne VLP-16 ‘Puck’ LiDAR unit, the Hikvision IP Bullet cameras, and the MaxBotix outdoor waterproof ultrasonic sensors. These sensors have been tested independently of the APM control systems and have not yet been installed on the APM or integrated with the APM systems.

5.2 LIDAR

The LiDAR module used was the 16 channel VLP-16 module by Velodyne, referenced at: http://velodynelidar.com/vlp-16.html.

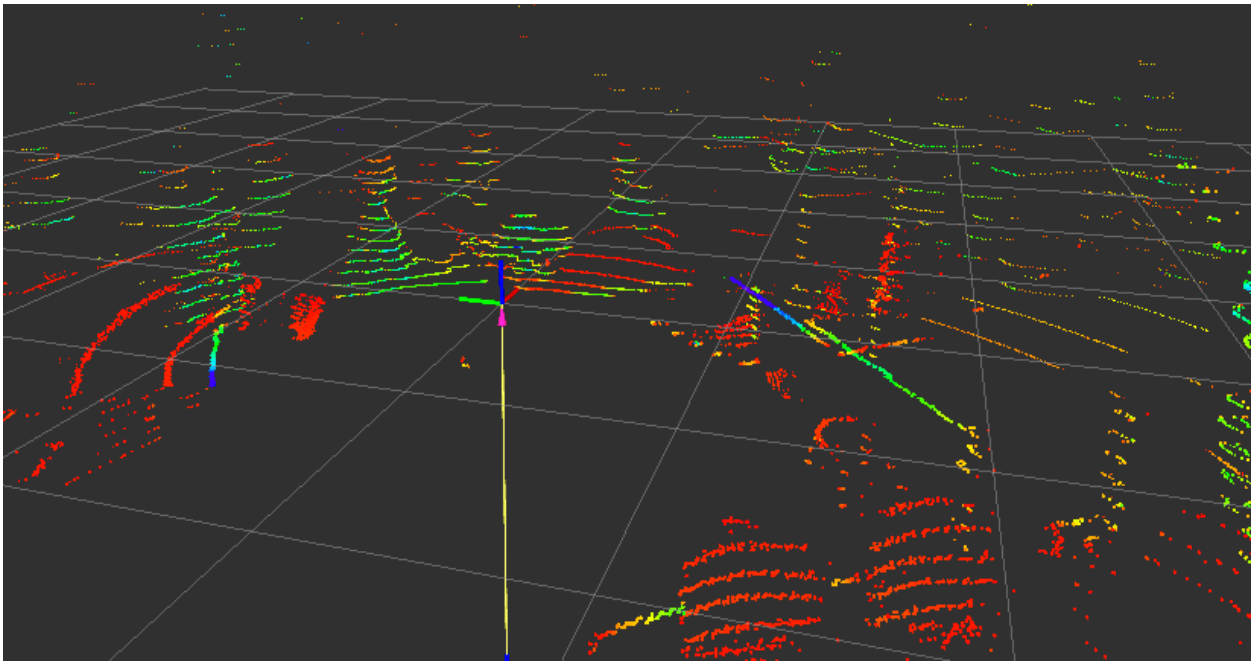
5.2.1 High Level Overview

The VLP-16 scans its surroundings and outputs a three dimensional pointcloud The resolution of the device allows for the ability to obtain visual odometry information and object information for both tracking cart movement and detecting obstacles in the environment.



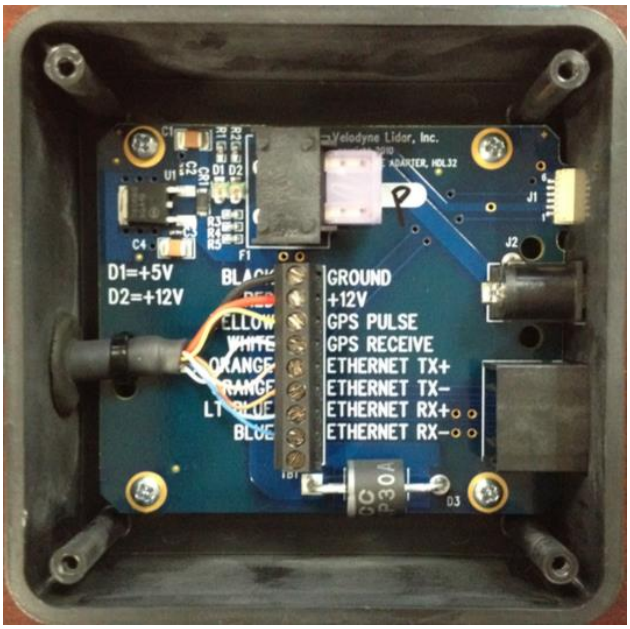
5.2.2 Software

The LiDAR raw data can be observed using the VeloView software that was provided with the module, independent from ROS. Additionally, a ROS package exists to capture the LiDAR output and feed it into ROS as a topic that publishes PointCloud2 data. The primary tool used to visualize the LiDAR point cloud data was the RVIZ visualizer in ROS. Using the ROS formatted PointCloud2 data, a number of packages exist to utilize the LiDAR data. The Velodyne ROS node is responsible for starting up the PointCloud2 topic in ROS. This topic is called “velodyne_points” and can be started using the cart.launch launch file located in the repository.



5.2.3 Wiring Diagram

The LiDAR unit connects to its Velodyne interface box via Ethernet cable, and the interface box connects to the computer unit via another Ethernet cable.



5.2.4 Mounting

The VLP-16 is mounted on the front of the APM approximately 38” above the ground and angled down (negative pitch) about 9°. The angle is adjustable in 5° increments with 10° increments on the upper connection and a single 5° increment on the lower connection.

5.3 Ultrasonic Sensors

The ultrasonics acquired for the APM were MaxBotix MB7001and MB7363 long range weather resistant sensors. Currently there are seven ultrasonics mounted on the front of the APM (six of the MB7363 and one of the MB7001).

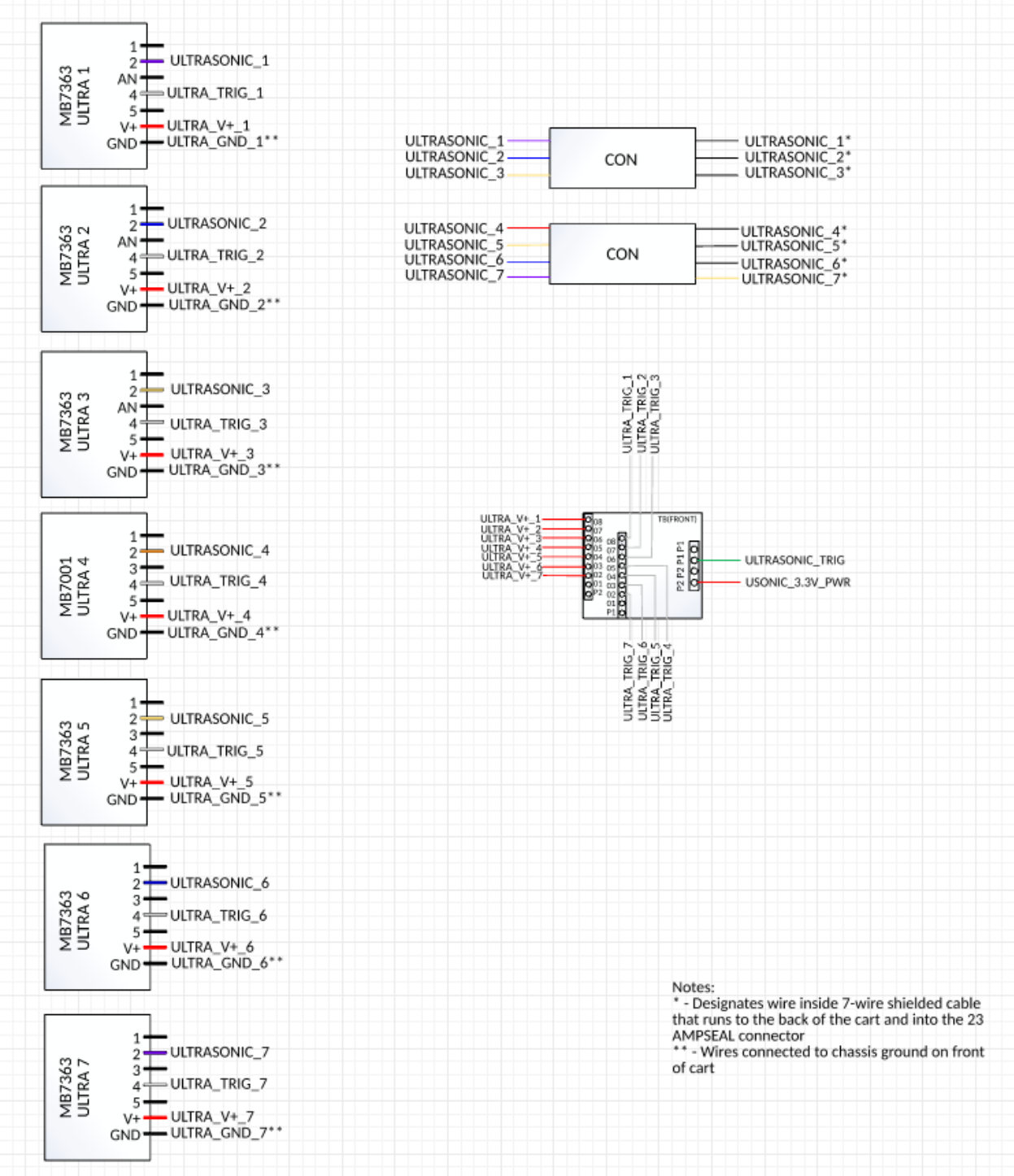
5.3.1 High Level Overview

The ultrasonic sensors were acquired for this project as an additional fail-safe system to the LiDAR unit. All seven ultrasonics are triggered at the same time with a high signal from the Arduino. Each ultrasonic sends out a sound wave to determine the distance of the nearest object. This distance is calculated on the Arduino by measuring the length of the pulse width that is sent from pin 2 on each ultrasonic. This distance is sent to a ROS node and is used to determine if there are objects within a predetermined distance in front of the cart. The ultrasonics are used in a stop zone node. This node looks for multiple ultrasonics to “see” an object within the given threshold. Currently, the ultrasonics are functioning but are noisy. Additional testing should be done to determine the cause of this noise and to determine the ideal positioning of the ultrasonics. Once this is determined, the ultrasonic data should be added to the point cloud. Due to the mounting of these devices, the wires connected to the ultrasonics are twisted which creates strain on them. It is recommended that a future team purchases connectors that would prevent the wires from being forcefully twisted.

5.3.2 Software

The distances each ultrasonic sensor is seeing is published to ROS from due 2. Due 2 reads pulse-width signals from each of the ultrasonic sensors and uses a conversion factor to publish distances to objects in meters as Float32 messages in ROS.

5.3.3 Wiring Diagram



5.3.4 Mounting

The ultrasonic sensors are mounted to the front of the cart using an adjustable and configurable PVC mount. The position, angle, and number of ultrasonics sensors can be changed easily with this design. The

PVC mounts to the frame via rubber-isolated clamps which are designed to be easy to remove for transport.

5.4 Encoders

5.4.1 High Level Overview

The brake encoder is not operational.

The Speed encoder signal enters the system on **Pin 14 (NOT 16)** of the 23-pin connector. It is a 5V square wave signal with 50% duty cycle. The frequency of the square wave is directly proportional to the frequency, with off being 0Hz and full throttle being around 210Hz. The sensor is directly connected to the wheel speed, presumably a Hall Effect Sensor.

The steering encoder is a potentiometer connected to the steering column. The slider of the potentiometer is connected to the control PCB in the back of the cart.

5.5 GPS

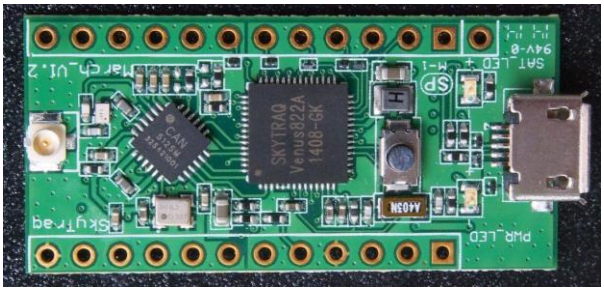
There are currently two GPS modules installed in the cart. One module is a combined GPS/GLONASS, and the other is an Real Time Kinematic (RTK) capable GPS module. The goal here was to obtain both reliable and accurate GPS measurements. The GPS/GLONASS module is like a standard GPS receiver that provides a fix with normally <2.5m accuracy. And the RTK GPS is a unique module that with additional corrections and additional software, is able to generate a fix accurate to about 10cm.

5.5.1 High Level Overview

The GPS modules are not currently being used as a source of localization data.

5.5.2 Hardware

The GPS/GLONASS module is SkyTraq's NavSpark-GL device. The product page can be found here at the NavSpark store: NavSpark-GL. The benefit of using a combined GPS/GLONASS receiver is that by receiving information from both the GPS and GLONASS constellations, satellite coverage is essentially doubled. This is a huge benefit when operating in urban areas where satellites at lower elevations can be obstructed by buildings. This in turn increases the chance that there will be adequate satellite coverage to achieve a fix, as well as increases the accuracy and reliability of that fix.

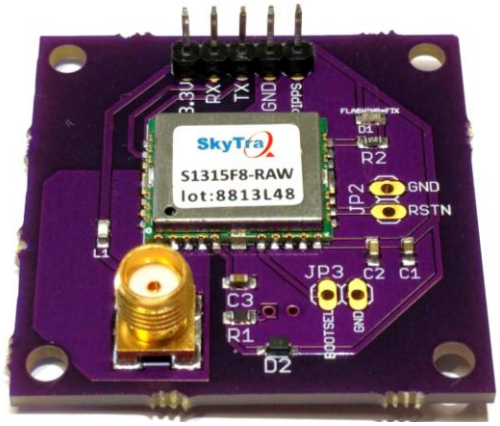


NavSpark-GL GPS/GLONASS Module

The Tallysman TW4320 antenna was purchased for this module due to its low noise figure as well as its support for both GPS and GLONASS signals. The antenna is an enclosed, magnetic mount, ceramic patch antenna. As per the datasheet, optimal results are obtained with the use of a 100mm ground plane.

Tallysman TW4320 GPS/GLONASS Antenna

The RTK capable GPS module is SkyTraq's S1315F8-RAW. This module allows for carrier phase raw measurement output. With this data, and additional corrections from a base station, software such as RTKLIB is able to generate a 10cm accurate GPS fix. The output of the module is standard UART serial, so a UART-to-USB adapter is used to both power and communicate with the device. During development, this was the cheapest RTK capable GPS module on the market at $25 (for just the module).

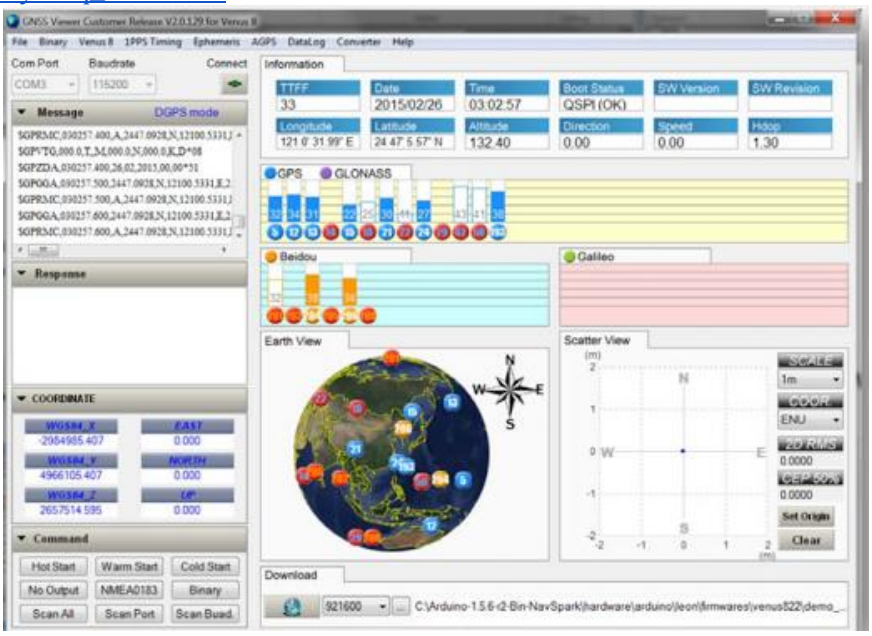


S1315F8-RAW RTK Capable GPS Module

The Tallysman TW4020 antenna was purchased for use with the S1315F8-RAW GPS module. Initially, a cheaper antenna was used, but it was found that the noise figure of the previous antenna was too high. This resulted in a poor signal to noise ratio, and did not allow for a fix to be obtained with RTKLIB. A ground plane of 100mm is also used with this antenna.

5.5.3 Software

As both GPS modules are made by SkyTraq, the SkyTraq GNSS Viewer software is used to configure them both. This program only runs on windows, but it allows configuration settings to be stored in FLASH memory so that they can be persistent through power cycling. Examples of settings set in this program would be power mode, output frequency, binary or NMEA output, elevation mask, etc.. The following image is a sample of the GNSS viewer software running. The software and user guide can also be found here: [SkyTraq_Resources]



Once the desired settings are configured, the output can be set to NMEA (this is also the default setting). The GPS receiver then operates like any off-the-shelf GPS module, tracking satellites and sending out information in the form of NMEA messages.

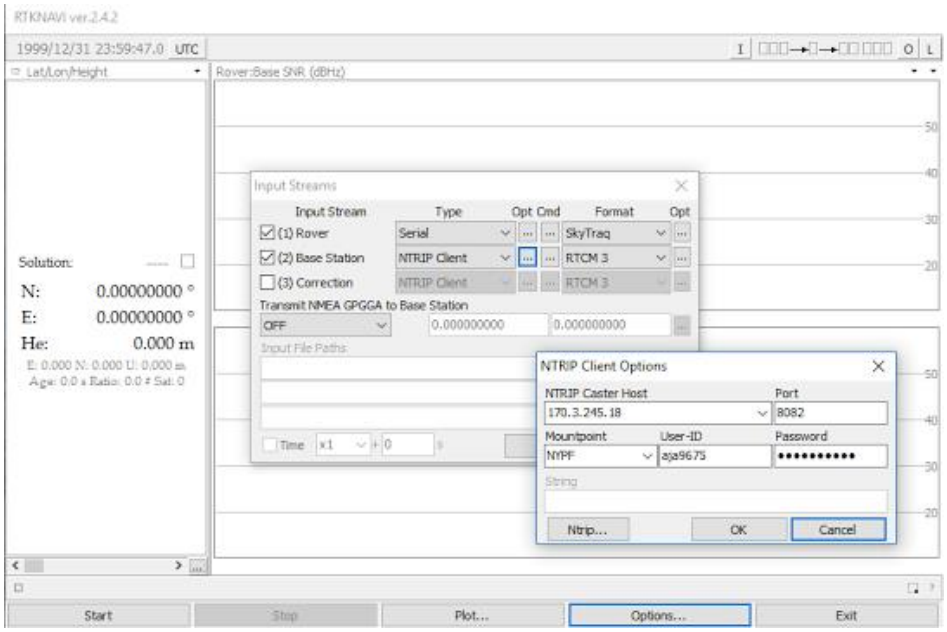
To use this NMEA information in ROS, a node exists that is called nmea_serial_driver, which is part of the nmea_navsat_driver package. Documentation for this package can be found here: nmea_navsat_driver. The nmea_serial_driver node will take a port and baud rate for a GPS module as arguments, and then connect to that module, parse the incoming NMEA messages, and publish ROS sensor_msgs/NavSatFix messages. Example command line syntax is:

$ rosrun nmea_navsat_driver nmea_serial_driver port:=”dev/ttyUSB0” baud:=”115200”

With the future addition of an IMU, this NavSatFix message, combined with local odometry and IMU data, can be fused together by the navsat_transform_node in the robot_localization package. This localization solution could then be used as an input source to one of the navigation filters in the robot_localization package. The robot_localization documentation can be found here: robot_localization. A current configuration file for setting up an Extended Kalman Filter with the robot_localization package can be found in the repository directory catkin_ws/src/robot_localization/launch/odom_gps.launch.

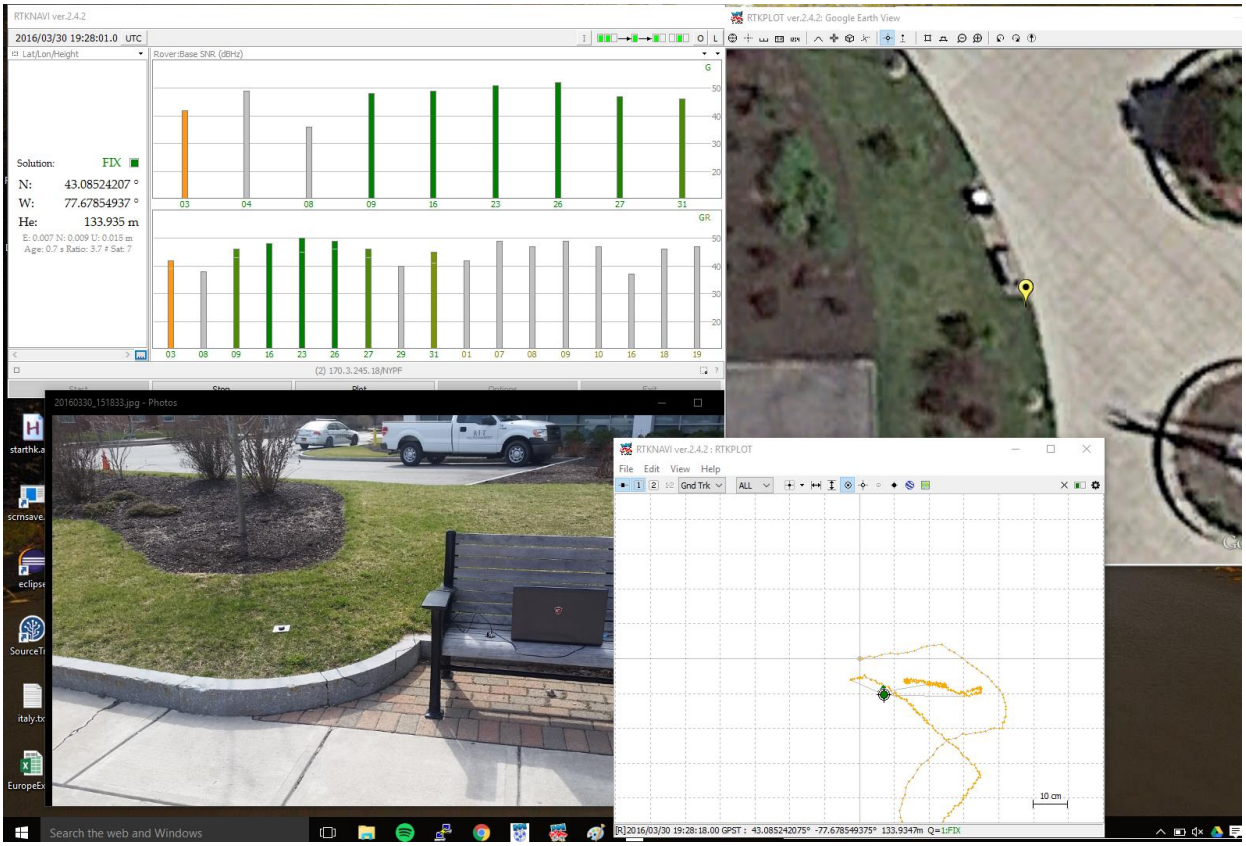
Alternatively, the S1315F8-RAW module can be configured for binary output and processed with RTKLIB. Currently, this has only been tested in windows. RTKLIB is open source and contains separate applications for use in windows and Linux. The navigation application is called rtknavi in windows and rtkrcv in Linux. Rtknavi provides a GUI interface whereas rtkrcv is command line based. RTKLIB software, documentation, and tutorials can be found at RTKLIB.

RTKLIB requires raw carrier wave measurements from the ‘rover’ receiver on the golf cart, as well as measurements from a static base station. Currently, base station data is being streamed from a Continuously Operated Reference Station (CORS) in Pittsford, NY. This data is provided as a free service. For instructions and to sign up, go to this site: NYSNet/CORS. You will receive a confirmation of subscription as well as the IP address and port of the broadcaster. This information is then used in RTKLIB to configure the base station input stream. The following image shows an example of this setup.



RTKLIB Windows Configuration

A configuration file currently exists that should set up the environment appropriately. It is called rtknavi_windows.config and is located in the repo at golf-cart/Old/Documentation/. This file can be loaded into RTKNAVI via Options…> Load > rtknavi_windows.config. These settings however, are not ideal. Significant improvements could be achieved by further tuning the software parameters. For example, modifying the elevation mask and signal to noise ratio mask so that only 'good' satellites are tracked, at the cost of not maintaining enough satellites for a fix or, adjusting the minimum ratio to fix ambiguity so that less false positive fixes are seen. With the current configuration, to achieve a fix there must be at least 7 common satellites between the rover and base station, all above 30dBHz SNR. Also, satellites below about 25 SNR should be blacklisted manually. To do this, you would first start up the process, wait for a fix, and record any satellites below 25 SNR. Then you would stop the process, go to Options…> Setting1 and add these satellites to the list of excluded satellites, and restart the process. The following image shows a test scenario with the physical position of the GPS antenna as well as the reported position by RTKLIB.



RTKLIB Test Setup and Results

6.0 SOFTWARE SUMMARY

6.0 SOFTWARE SUMMARY

The software system on the APM utilizes a ROS communication system to implement controls and data processing.

6.1 Robot Operating System

Robot Operating System (ROS) was used to implement the data processing for the sensors in addition to the controls on the Arduino microcontrollers. ROS nodes executed operations based on changes in the data in the ROS topics. ROS Topics serve as communication channels upon which messages can be sent. Messages can either be standard scalar values (Integers, Floats, Bools, Empty messages, etc.), or custom messages which contain multiple types.

6.2 Localization / Odometry

6.2.1 High Level Overview

The area of the cart that serves to have the most room for improvement is localization. Localization is what tells the cart where it is, and the odometry data is an integral part of that. Without good odometry and localization the cart is lost. Currently, the only working odometry for the cart is so called 'cart odometry' which is a model that was developed to take in the wheel speed and steering angle of the cart and outputs an estimated change in position. Previously an IMU was used to supplement this data, but high levels of noise made it less than useful. However, the ROS package robot_localization, given good data including covariance matrices, will be able to fuse the cart odometry, IMU data, GPS data, visual odometry, and whatever else can be leveraged to be combined in an extend Kalman filter and used to output a combination of the them all as filtered odometry.

6.2.2 Software

The cart odometry listens to the speed and steering angle topics and uses known parameters of the cart to plug into a three degree of freedom model which will output an estimated change in position. This is started by running the 'odom_gen' node in the 'odom' package.

6.3 Controls

Controls in ROS were implemented on Arduino microcontrollers which required some additions to the Arduino code and additional steps in the setup of the ROS system. A tutorial for setting up the system can be found at http://wiki.ros.org/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup.

6.3.1 High Level Overview

While there is an existing library, called rosserial, which provides a serial interface between microcontrollers and ROS, as of the writing of this manual there are specific issues with the Arduino DUE microcontroller, and while it can be made to work for short periods of time (less than 10 min or so), it is unreliable. To get around this a new serial communication framework was developed by P16241 called rosdue, which allows for the creation of publishers and subscribers in the Arduino code and communicates with a node running on the computer over serial. Examples of use of this library can be seen in test_rosdue.ino as well as both the main programs for the dues, and the public repo for rosdue: http://github.com/codysmithd/rosdue.

Given this connection to the computer, the two Arduino dues have differently. Due 1 is tasked with controlling the throttle, steering, and brake, as well as taking in the remote control input. Due 1 subscribes to the control_mode topic from the ROS system to decide if it should operate the controls in manual, remote, or autonomous mode. Due 2 also listens to the control mode topic, and publishes data from the IMU, Ultrasonic sensors, and controls the on-board light and sounder.

6.3.2 Software

The main programs that are loaded onto due 1 and due 2 for use are ‘main_due1.ino’ and ‘main_due2.ino’. These programs leverage all of the previously discussed libraries and operate in the manner described above.

7.0 ELECTRICAL MODIFICATIONS SUMMARY

7.0 ELECTRICAL MODIFICATIONS SUMMARY

7.1 Ampseal Connectors

Wires for pins 14 and 16 were reversed in the 'To Cart' and 'To Controller' Ampseal connectors so that the VTACH signal could be monitored properly. This was due to the original PCB schematic from Phase I switching these wires, and this error being carried over to the new design.

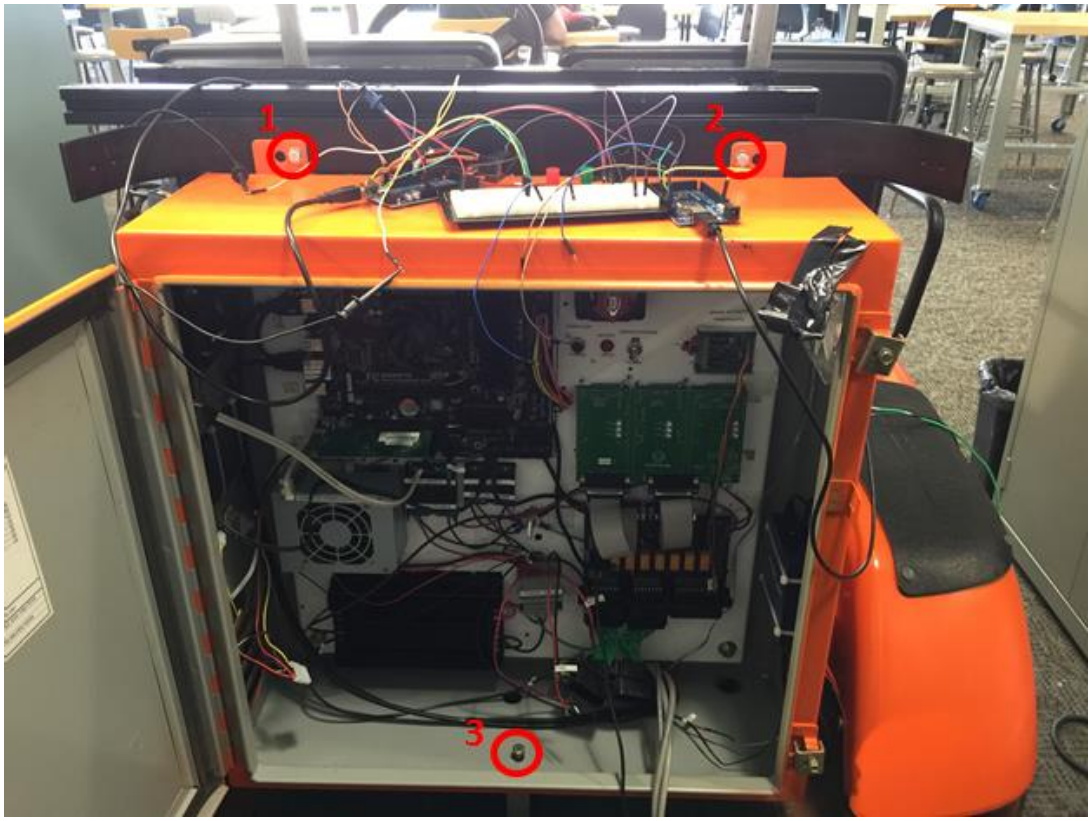
7.2 PCB

While debugging the VTACH signal mix-up, a trace on the PCB was burnt out, requiring a jumper on the rear side of the PCB to connect VTACH input and VTACH output.

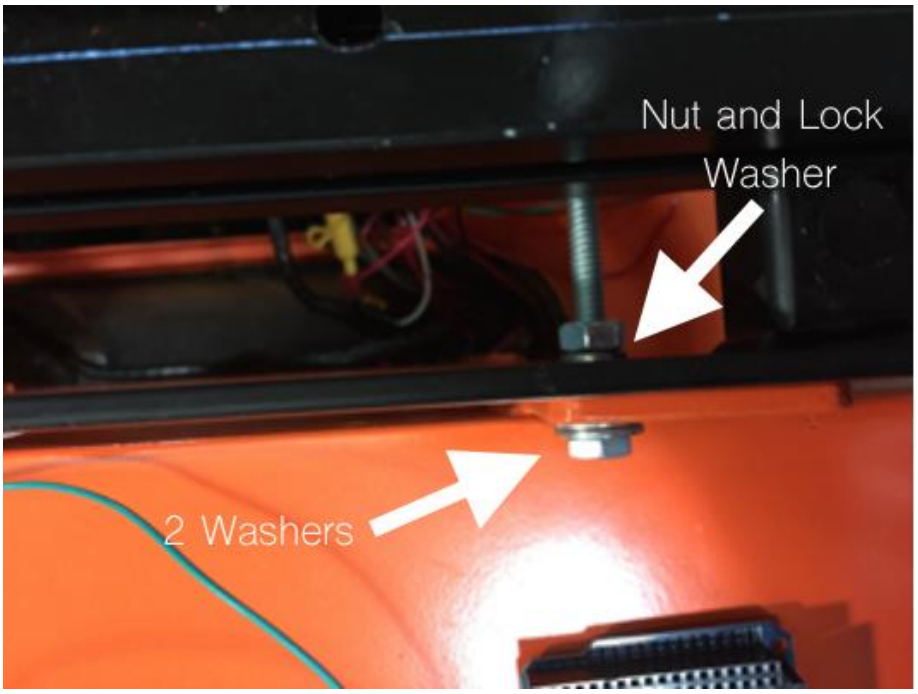
7.3 Enclosure

The enclosure is located on the back of the cart. To take the panel off, the fans need to be removed. To remove the fans make sure that the power to both fans is disconnected (wires TB03+ and TB03- and wires labeled R Fan) from the panel, the hard drive, HDMI, USB, LED lights, Wi-Fi adapter, wires labeled TB01+ and TB01- need to be disconnected. After these are all disconnected remove the four nuts and washers holding the PCB panel in place. Then carefully pull the PCB panel straight out a little way until just off the bolts. Once the PCB panel is off the bolts, tilt the top of the PCB panel toward yourself. There are wires with quick connects on the bottom backside of the panel that need to be disconnected. After all these wires are disconnected, the panel can be carefully removed from the enclosure. All wires and connections to and from the cart come into the enclosure through the larger hole at the bottom right of the enclosure.

To remove the enclosure from the cart, all connections coming from the cart into the panel, must be disconnected. Then all the mounting bolts must be removed.



The three mounting bolts can be seen above as 1, 2, and 3. The upper bolts (1, 2) go through the inside holes of the two tabs.

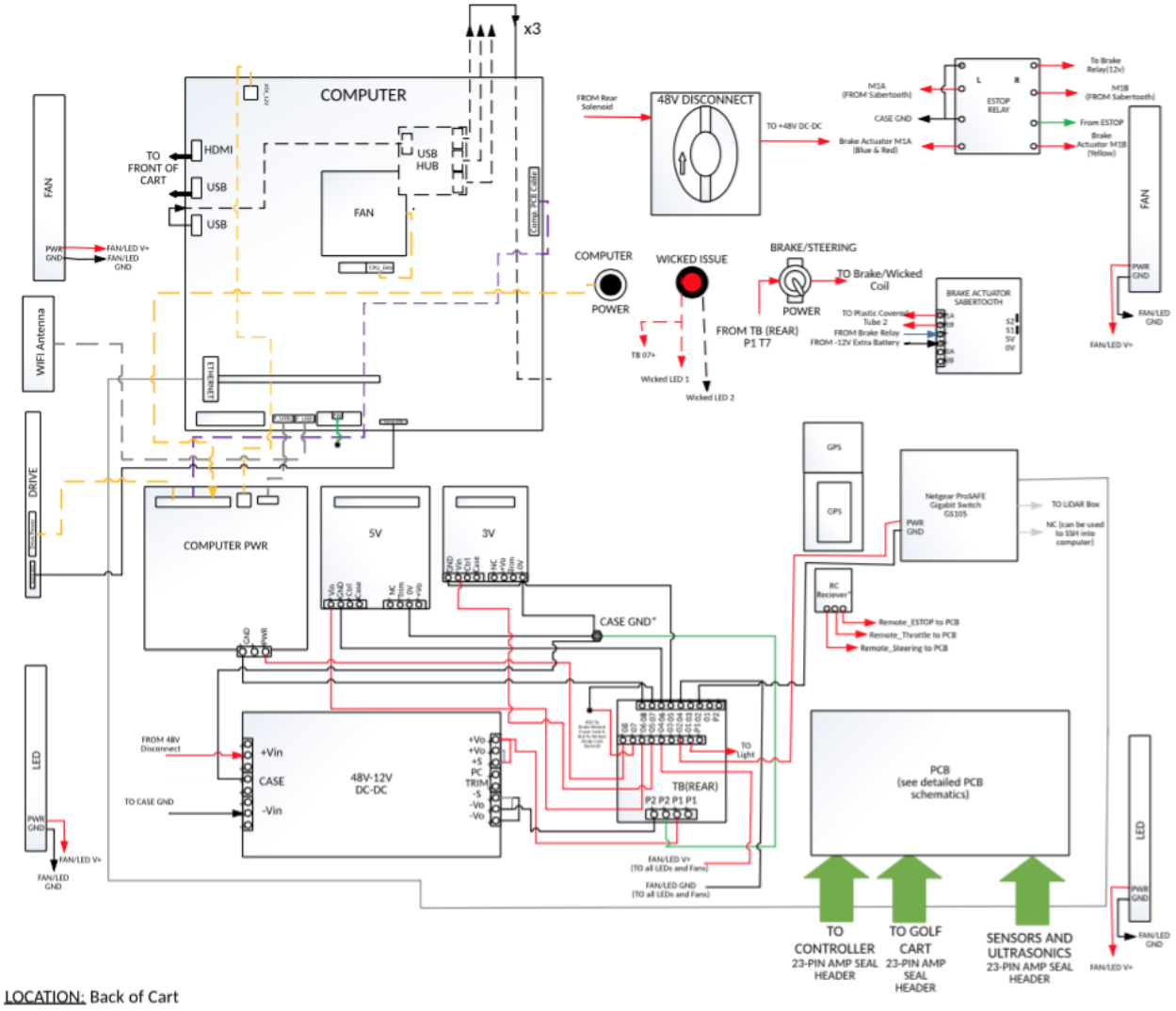


Top down view: How to place washers and nuts on bolts.

7.3.1 High Level Overview

The enclosure contains the majority of the hardware needed to make the APM autonomous.

7.3.2 Wiring Diagram



LOCATION: Back of Cart

7.3.3 Mounting

The primary mounting elements for the enclosure are the two full thread ¼ in bolts through the flanges at the top of the enclosure and the enclosure base at the bottom. The upper bolts should be used with a lock washer and standard washers on the front face. Bolts should be torqued to ~8lb-ft.

The enclosure base is secured with two full thread 5/16 in bolts. The upper bolt is secured from the top with a lock nut, goes through the lower plate of the enclosure, and threads into the upper section of the enclosure base. The lower bolt is secured from the bottom with a lock nut, goes through the lower frame rail of the golf cart, and threads into the bottom of the enclosure base.

When installing the enclosure base start by loosening the two upper bolts. Always secure the bottom bolt first making sure the lock nut is threaded all the way to the head of the bolt. To make fitment of the upper bolt easiest, torque the bottom bolt to ~12lb-ft to ensure the base is flush against the rear tray. Once torqued, swing the bottom of the enclosure forward carefully to align the top of the enclosure base and the hole in the bottom of the enclosure. Once aligned thread the lock nut all the way to the head of the bolt, insert the upper bolt, and torque to ~12lb-ft. (Specs: 1/2in thick threaded aluminum plate at each end, 2.7in long 1x1in aluminum square stock center section)



(Upper) (Lower)

7.4 Dashboard

7.4.1 High Level Overview

The Dashboard consists of wiring and switches/buttons required for the operator to interact with the control systems. The switches included in the dashboard are: a 10.1" LCD Touchscreen, an emergency-stop switch, a Forward/Reverse switch, the golf cart power key, and the accelerator stop toggle switch. The operation of the components is detailed below:

- LCD Touchscreen - This touchscreen communicates with the processing computer via HDMI and USB, with HDMI providing video signal to the monitor and USB providing touch inputs to the computer. The LCD has a status LED which changes color based on the status: Red when the monitor is powered off, Orange when it is powered on but there is no video input, and Green when it has video input. The LCD is powered by a 12V power supply, which is fed from the 48V converter in the rear of the cart.
- Emergency Stop Switch - The design of the emergency stop system utilizes a push button to trigger three actions. The first action signifies to the system that an emergency stop has been activated. The second action causes a break in the "Main Switch Line"(Key) which disconnects power to the motor of the APM. The third action engages the brake actuator to stop the cart and prevent any further movement. These actions are necessary to ensure that the golf cart and passengers are in a safe state in the case of software error or hardware failure.

    - When the E-Stop button is pressed, the following should occur:
        - A zero throttle signal is sent to motor.
        - Brakes are fully engaged.
        - Power is disconnected from the drive-motors.
    - When the E-Stop button is released, the following should occur:
        - Throttle control is returned to the Arduino Due.
        - Brake control is returned to the Arduino Due.
        - Power to drive-motor remains disconnected until the Throttle-Safety button is pressed.

- Forward/Reverse switch - This is a switch that switches the direction the cart is moving. This is a stock switch included on the standard golf cart.
- Key - This turns on the cart, which must be done for the throttle to drive. It does not switch the 48V input to the enclosure, nor does it switch the 12V power for the brake and steering systems.
- Accelerator Stop Toggle Switch - This toggle switch replaces the switch inside the accelerator pedal. On the stock golf cart, a switch is present in the accelerator pedal to prevent the cart from moving unless the pedal is physically pressed, and to ensure that the cart does not move if the pedal is pressed while the key is turned on. This button replaces this switch to enable driving autonomously without pressing the pedal. The button must be deactivated (out) when the cart is turned on, and then must be pressed in order to drive. A click and a humming should be heard when the cart is successfully activated.

The dashboard is not waterproof, and thus the golf cart should not be driven outside in the rain. In order for the dashboard to be waterproof, a new mounting solution will be required to prevent the LCD from getting wet and to protect the electronics inside.

7.4.2 Wiring Diagram

8.0 GETTING STARTED

8.0 GETTING STARTED

8.1 Starting the Golf Cart

To start the cart make sure the switch under the seat, on passenger side of the cart (the 'tow' switch) is in Run (down position). Make sure that the cart key is in the ignition, and the green button is out/'off'. Turn the key to on, and press the green button so it is in the pressed position. At this point you should hear a solenoid click. In the enclosure, turn the 48V disconnect on the panel to the on position to get power to the panel. Flip the Brake/Steering Power switch to send power to the steering Wicked system, brake actuator and panel when ready. We don't recommend flipping this switch until you have connected DUE 1 to the computer with ROS and rqt running.

8.2 Starting the Computer

To start the computer, after the 48V enclosure power is turned on, press the computer power button. At this point, the computer specific fans will start, and the computer should boot into Ubuntu 14.04. The user account is 'apm', and the password is 'gamma', the name for the account (for some GUI viewing purposes only) is 'admin'. Assuming you want to get the cart running autonomously, the following commands entered into the terminal are required. These can either be launched in separate terminal tabs, one per command, or backgrounded by appending an '&' character to the end of each command, viewed by running 'jobs' and brought back to the foreground by running 'fg [process number]'. Please note that the '~/.bashrc' automatically runs 'source /devel/setup.bash' for the catkin workspace.

First launch roscore:

roscore &

Then open the GUI for debugging and controlling the mode of the cart:

rqt &

Once rqt is open, you may need to go to plugins -> apm -> apm mode control to get the GUI for selecting the control mode

Then, to be able to visualize what is going on the with the cart, open up rviz

rviz &

Open rosdue connections to DUE 1 and DUE 2

rosrun rosdue serial_due.py /dev/ttyACM0 &
rosrun rosdue serial_due.py /dev/ttyACM1 &

Run the launch files for the LIDAR's pointcloud publisher and the pointcloud to laserscan.

roslaunch velodyne_pointcloud VLP16.launch &
roslaunch pointcloud_to_laserscan p2l.launch &

Launch the node that initializes the cart odom, it is suggested to run this in a new terminal tab so that is can be restarted easily.

rosrun odom odom_gen

Launch the autonomous node that converts cmd_vector to the cmd_throttle, cmd_steering, and cmd_brake for DUE 1.

rosrun apm_autonomous vector_to_command &

The artificial potential field is the layer of autonomous driving which prevents the APM from driving into anything. We developed new type of APF for the project called Modified Path-Weighted Artificial Potential Field (MPWAPF). Without this running, the desired vector will not get converted to the cmd_vector.

rosrun apm_autonomous artificial_potential_field &

The last step is to generate the desired vector from the waypoint path. At this point, autonomous mode is ready to use. It is suggested to run this in a new terminal tab - not backgrounded so that it can be stopped and restarted when the cart is being reset.

rosrun apm_autonomous local_driver_node <waypoint_file.txt>