Florida Institute of Technology

PETE Design Report

May 15, 2018

Team Captain: Jose Della Sala (jdellasala2014@my.fit.edu)

Name	Email
Gianluca Annoscia	gannoscia2014@my.fit.edu
Patrick Assaf	passaf2014@my.fit.edu
Dylan Cannisi	dcannisi2017@my.fit.edu
Adam Diagne	adiagne2014@my.fit.edu
Mounibe Diarra	mdiarra2013@my.fit.edu
Rachel Froling	rfroling2015@my.fit.edu
Chukwubuikem Idigo	cidigo2013@my.fit.edu
John Joo	jjoo2013@my.fit.edu
James Marte	jmarte2016@my.fit.edu
Mohammed Al-Musalmani	malmusalmani2014@my.fit.edu
Zachary Paryzek	zparyzek2012@my.fit.edu
Gaelen Trew	gtrew2014@my.fit.edu
Yiqi Xiao	yxiao2014@my.fit.edu
Zhiyi Ye	zye2016@my.fit.edu

Advisors: Dr. Matthew Jensen & Dr. Ken Gibbs

Table of Content

Introduction

PETE is an autonomous vehicle designed, built, and programmed by the Florida Institute of Technology (FIT) team consisting of electrical, mechanical and software engineering students. The purpose of building PETE is to present it at the Senior Design showcase at Florida Institute of Technology on April 6th and compete in the Intelligent Ground Vehicle Competition (IGVC) on May 31st.

This year's vehicle is a continuation of last year's effort to position FIT as a contender at the IGVC competitions. While in autonomous mode, the vehicle uses a combination of Global Positioning System, Inertial Navigation System (GPS/INS) technology, and a ZED Stereoscopic Camera for lane detection, obstacle detection and motion planning. A Jetson TX1 processes the images received from the camera for motion planning and mapping. Mapping and motion planning are performed by a series of Raspberry Pis that send instructions to the Roboclaw Motor Controller. During manual mode, a radio transmitter sends radio signals to interface with the motor controller using an Arduino microcontroller.

Organization

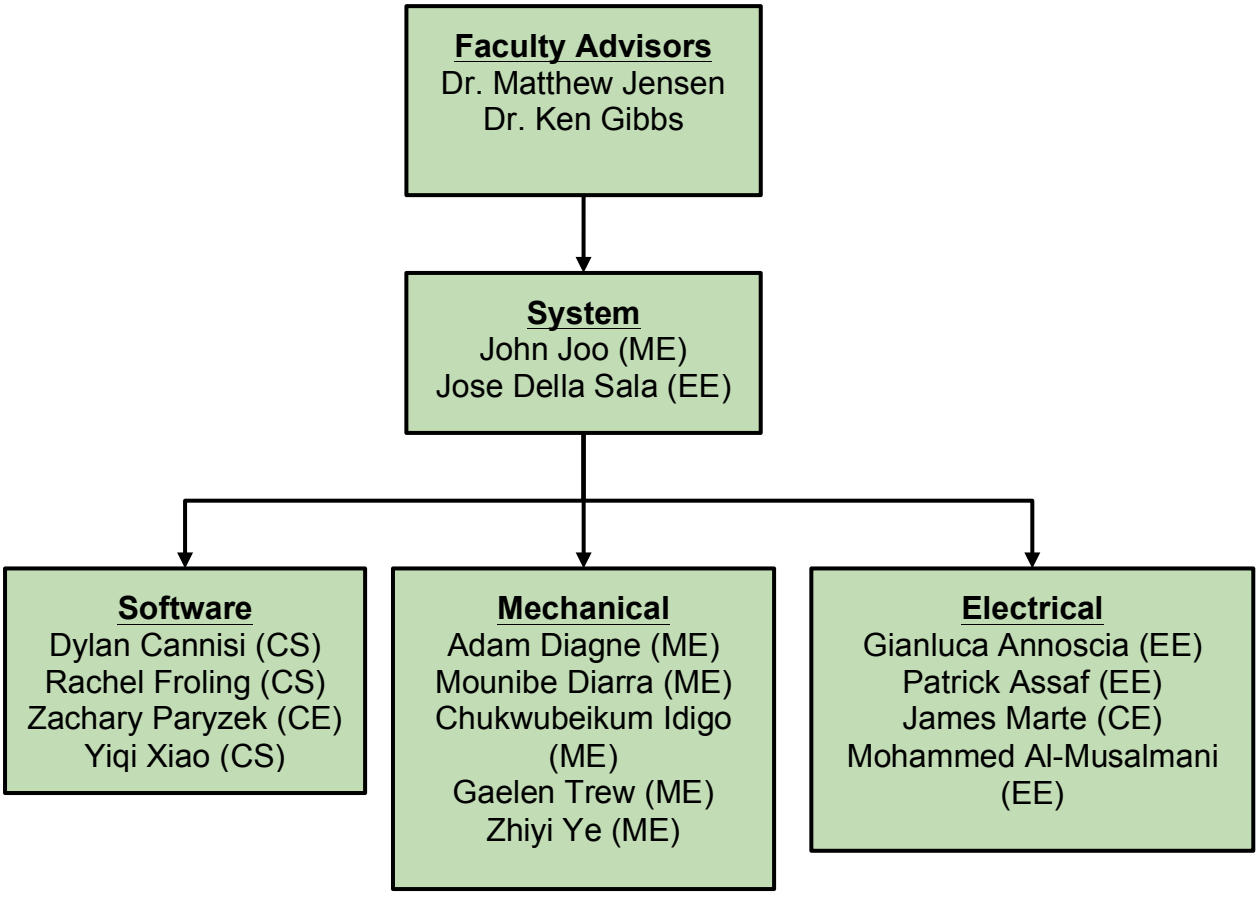

Figure 1: Team organization structure

The team is comprised of software, electrical, and mechanical sub teams. The mechanical team is tasked with designing a light and waterproof chassis, assembly

for the motors and wheels, and mounts for electronics placed outside the vehicle. The electrical team is in charge of designing and implementing the circuit inside the vehicle. The circuit had to take into account several aspects like: power distribution, communications, safety and usability. The software team is in charge of producing the algorithms and control of the communication, vision, position estimation, and motion planning. The duty of the systems team is to organize meetings, facilitate communication between subsystems, and manage documentation for the competition.

Design Process

The team utilized 6 major strategies in the design of PETE. The first step is to analyze the problem and define requirements the vehicle needs to meet. The team looked into the design and documentation of the previous year's vehicle to identify areas for improvement. Ideas where then generated to address the problem statement and requirements for the vehicle. These ideas are explored through further research and discussed with the team. From the various alternatives, an approach is selected. The idea and design undergoes multiple iterations before manufacturing and integrating the components. The last step is to test and refine the model. The performance of the vehicle will be determined and when problems surfaces, they are taken back to the first step of the design process to develop a solution that conforms to the new requirements.

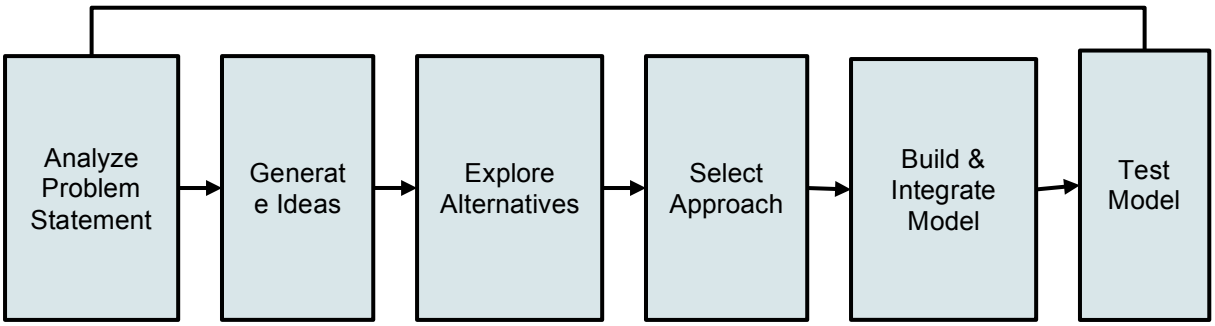

Figure 2: PETE Design Process

Mechanical Design

Overview

The design of the vehicle focused on manufacturability, weight, and ease of use. The vehicle was designed to have sufficient space for the payload, motors, and electronics to be placed inside. It was determined that maintaining and repairing the vehicle during testing and competition is an important element to consider for the team to make changes or replacements when necessary. The team considered the effects of the terrain and weather conditions that the vehicle will be running in during competition as outlined in the competition rules.

Frame Structure and Housing

The vehicle chassis is made entirely of aluminum square tubes. Aluminum square tubes were chosen because of its strength to weight ratio, making the vehicle light and rigid enough to withstand collisions and other physical abuse. Its rectangular shape provides the space for the electronics, motors, and the payload inside the robot. Its shape also makes it easier to manufacture body and lid to protect the inside of the vehicle from the environment. The tubes are welded together, making the frame rigid. The frame should be capable of withstanding collisions at the maximum speed the vehicle will be moving. The frame was designed to keep the center of gravity close to the ground to prevent from tipping when moving up slopes and uneven terrain.

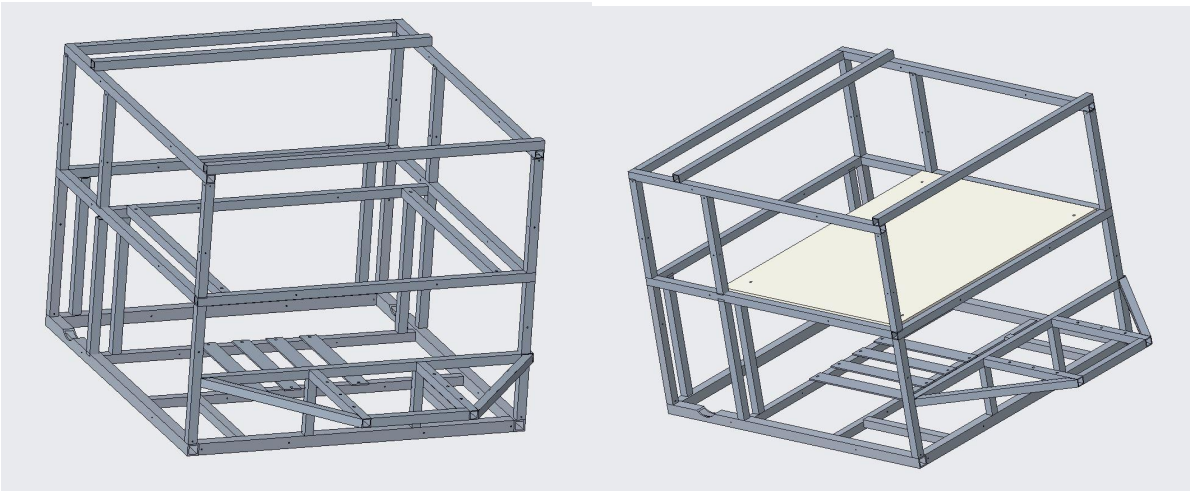

Figure 3: Aluminum Chassis of the Vehicle

The frame is covered with composite panels and aluminum sheets. Most of the coverings are made from composite panels to reduce weight and aesthetics. Composites are lightweight and provide the necessary strengths to hold the weights that will be placed on it. Aluminum sheets is used where rigidity and strength is important. Electrical sub team had given the requirement that the vehicle must not be completely enclosed in metal or any other electrical insulator. Carbon fiber and fiberglass were found to inhibit the radio and wireless signals that the vehicle will be using and so, a particular type of fiberglass that do not insulate the signals was used for the lid.

Drive System

For this year's design, two PG27 Planetary Gearbox with RS775 Motors from AndyMark are used to provide the torque for the wheels. Based off the given stall torque of 6.3 ft-lbf and the situation that vehicle should be capable to climb a 15 degree incline. From the calculation results based on the formula for torque at an incline, an idealized range for the radius of the wheel between 12 inch to 16 inch has been computed. With this information, two 13-inch wheels were chosen. To maneuver the vehicle, two options were considered: two-wheel drive and differential steering. The two-wheel drive system would add complexity to the robot, as a separate steering capability will be required. The differential steering on the other hand provides both steering and driving capabilities through the difference in angular velocity of the two wheels. The motors from the previous year are used to reduce the

cost for manufacturing the vehicle, giving the team more flexibility to purchase other parts and materials.

Weatherproofing

To prevent rain from entering the vehicle and damaging the electronics, the lid covering the top of the vehicle uses a foam seal. The wires for the electronics placed outside of the vehicle such as the camera, GPS antenna, and the emergency stop run through 3D printed covers to prevent rain from entering the vehicle.

Suspension

A dedicated suspension system is not designed as it was determined that one was not needed for the relatively slow speeds and terrain the vehicle will be running at. Rubber washers are utilized to dampen vibrations on the electrical platform and the front caster wheel.

Electronic and Power Design

Overview

Several distinct types of hardware have been specifically chosen to fulfill the required result:

Table 1: Important Hardware Used in the Vehicle

Jetson TX1	The Jetson TX1 is the main computer vision embedded system. It receives raw data from the ZED camera and performs the image processing algorithms needed. It then sends the processed signals to the Raspberry Pi responsible for Motion Planning.
Vectornav 200 INS	This powerful GPS/INS module will feed the Raspberry Pi with the location of the vehicle.
AC1300 Dual-Band Gigabit Wi-Fi Router	The router will serve to create a LAN between the Raspberry Pis and TX1 in order to exchange information.
Tolako 5V Relay Module	This module contains all the relays in the system.
Fuses	Different fuses are used across the board to protect the electrical circuit from overcurrent.
ZED Camera	This camera is the primary vision sensor for the system. It will provide the images to the Jetson TX1 for further processing.
Turnigy 2.4 Ghz 8 channel receiver with radio controller	This radio controller-receiver pair is in charge of switching between the Autonomous and Manual Modes. When in Manual Mode, it allows us to remotely

	control the robot's motions using radio signals. It also grants the ability to wirelessly stop the vehicle in case of emergency.
RoboClaw 2x45A Dual Channel Motor Controller	This dual-channel Motor Controller is in charge of controlling the DC servomotors' motions through PWM command signals sent from the Motion Planning Raspberry Pi.
Raspberry Pi	Vision Processing: This Raspberry Pi is in charge of communicating with the INS to determine the vehicle's GPS location on the map. Motion Planning: This Raspberry Pi analyses and interprets the data gathered from the sensors in order to plan and dictate the vehicle's motions. Mapping: This Raspberry Pi will contain all the GPS/INS mapping that will allow to use and interpret the signals coming from the GPS/INS.

Circuit Block Diagram

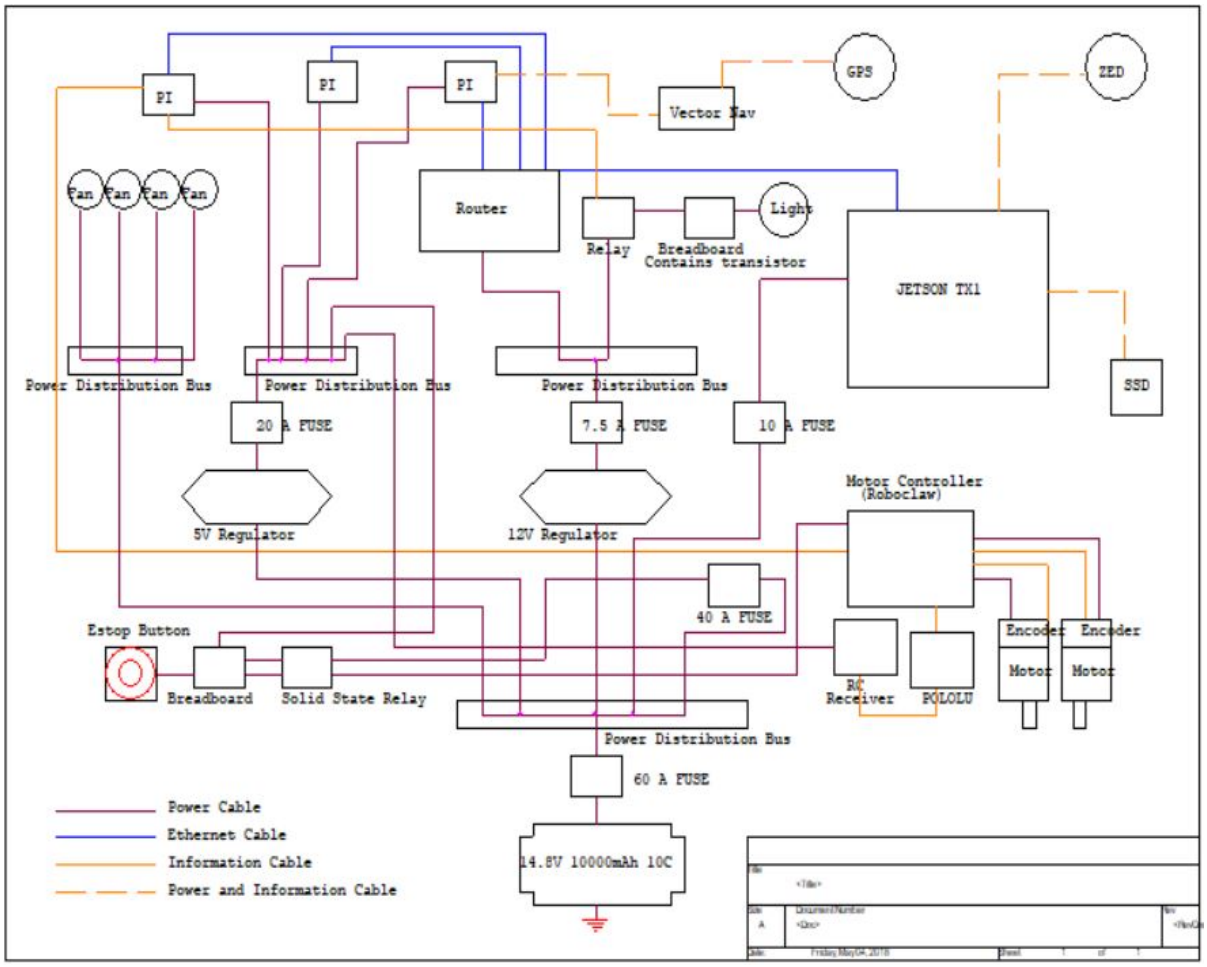

Figure 4: Final Block Diagram of the Electric Circuit

Power Distribution System

The circuit distributes power from the 4S 14.8V 10000mAh 10C lithium polymer battery to different subsections/branches by using a system of power distribution busses and voltage regulators/converters. The battery is connected to one central power distribution bus that transports its power to the other subsections. One subsystem uses a 5-volt step-down regulator, which distributes its power through a smaller power distribution bus, and distributes the power to the 3 Raspberry Pi and the mechanical emergency stop circuit. Another subsystem uses a 12-volt step down regulator to distribute power through another small distribution bus and powers the router and the indication light. Finally, the motor controller and the TX1 are wired directly to the main power bus.

To improve the stability and reduce noise we have outfitted the circuit with step-down regulators that maintain a constant current and voltage. The 5-volt regulator chosen has a higher current output because of the three Raspberry Pi's current requirement. The 12-volt regulator is chosen due to the fact that a more powerful router is needed to manage all of the extra traffic and communication between the devices. The TX1, which can handle up to 19 volts, is connected directly to the battery and is running on 14.8 volts. The motor controller is also connected directly to the battery because of its large current draw and high voltage requirements. This also helps to reduce noise since most of the other subsystems draw their power through the regulators.

Battery Life

The total power consumption is calculated to be roughly equal to 700 Watts. Using the specifications of our battery (4S 14.8V 10000mAh 10C), estimations of the battery life for different scenarios have been computed. Assuming that all systems are running simultaneously at their maximum capacity, the run time of the vehicle is estimated to be at around 10 minutes. However, if the vehicle is to stay on Standby, then the battery life would extend to approximately 1.08 hours.

Radio Controlled Mode

Following competition guidelines, the vehicle is equipped with a radio controlled mode allowing the user to fully control the movements of the vehicle. The components included in this module are: RC Controller, RC Receiver, Pololu RC switches, Raspberry Pi 3, Arduino UNO and the Roboclaw motor controller.

The implementation of the RC mode consists of three main steps: reading the signals from the receiver, sending the signals to motion planning, and executing commands. The receiver is the Turnigy receiver that is paired to its respective controller. The receiver has 4 analog channels and 4 digital channels. The analog channels are controlled by the left and right knobs on the controller while the digital channels have been mapped to the four back switches on the controller. The multiplexing of the 8 channels is achieved using Pulse Position Modulation. However, each channel has an individual PWM wave with a duty cycle that ranges from 5% to 10%. The controller is configured to facilitate two analog channels for driving and steering. The horizontal motion of the right knob is used to indicate the direction to turn and the vertical left knob is used as the throttle.

Two of the digital channels were enabled to indicate the change in modes between autonomous and radio controlled, and the triggering of the emergency stop. To process the radio signals coming from the receiver, two devices that entail different methods were used: an Arduino UNO and a Pololu RC switch. The Arduino uses two pins to read the PWM signals coming from the receiver. The PWM libraries on the Arduino Uno IDE were used to interpret the signals. The Arduino Uno is connected to the Raspberry Pi by a USB Type B cable to establish serial communication.

Two Pololu RC switches are used to obtain a Boolean signal from the channels on the receiver. These devices transform a radio signal into a Boolean HIGH or LOW signal. The voltage level for HIGH can be adjusted, yet the value used was 5 Volts for HIGH. The output of these switches goes to the GPIO pins on the Raspberry Pi for motion planning. The Pi is connected to the motor controller by USB. The Roboclaw is equipped with a firmware capable of receiving and executing USB commands that come from the raspberry Pi in order to move the motors in the desired direction and with a controlled speed.

Emergency Stop

Due to competition guidelines, the vehicle is equipped with an emergency stop mechanism that is independent from software. A two level emergency stop mechanism is designed to meet competition guidelines and, ensure effectiveness and safety. The first layer is the software soft stop; the second layer consists of a logic circuit and a relay to cut the power to the motors.

Soft Stop

The first layer of the emergency stop is a “Soft stop” conducted by the Raspberry Pi. When the emergency stop switch has been triggered by the RC controller, the receiver sends an RC signal to the RC switches that is later transformed into a readable Boolean signal by the Raspberry Pi. When the Raspberry PI detects a HIGH coming into its GPIO pin, it immediately sends ‘’STOP commands’’ via USB to the motor controller. The motor controller then receives these ‘’stop commands’’ and ceases to send current to the motors. This stop mechanism is designed to inform the motion planning that an emergency stop has been called to reduce the power entering the motor controller when the crowbarring of the power occurs.

Hard Stop

The second layer of the emergency stop is the crowbarring of the power going into the motors. This ensures that no current will flow to the motors when the mechanism is triggered; thus bringing the vehicle to a stop. It is important to note that this mechanism is independent from the first layer where the ‘’soft stop’’ occurs and contains no software interfaces or sequences.

The main component in the emergency stop is the D1D1000 solid-state relay. The input loop of this relay is controlled by a digital circuit with the output loop going directly to the motors.

The circuit shown on top consists of the following logic components: an inverter gate, and or gate, a 555 monostable multivibrator, and a positively triggered D-type flip

flop. The circuit receives a level signal coming from either the mechanical estop button or the RC switch. These level signals are passed through an OR gate and the output goes to a Resistor-Capacitor circuit that creates a short pulse from the level signals. This short pulse is enough to trigger the monostable multivibrator. This device creates a delay that will allow the Raspberry Pi to perform the ‘’soft stop’’. The delay is also a short square pulse that serves as a trigger to the clock of the D-type flip flop. Once the flip-flop is triggered, it will send a 5V output through its Q pin. This output is used to control an IRLZ34 MOSFET that is being used as a switch. In the presence of a HIGH output, the MOSFET will become saturated, hence behaving as a shot circuit that will ground the input loop of the relay, enabling current to flow through the output loop that feeds the motor controller. In the absence of an output from the flip-flop, the MOSFET will remain on cutoff mode, hence behaving as an open circuit blocking the path of current in the input loop of the relay and consequently its output to the motor controller. This way the digital circuit is able to decide whether or not the relay will enable current to the motors.

After the emergency system has been triggered upon runtime, and the flip-flop has been activated, the system needs to be brought back up to a known state. To do that the flip-flop has to be reset so that it looks for a trigger again. This is accomplished by a reset button attached to the reset pin on the flip-flop. This reset button is a normally closed button that sends a constant 5V signal to the reset pin in order to ensure proper operation. To reset the flip flop, the reset pin has to be brought low, and then back high again where it will stay that way until it is triggered again and a new reset of the flip flop is required.

Software Design

Overview

The overall software strategy for this project is to build individual subsystems, each with their own task that will communicate to perform the overall objective of the vehicle. The time constraint of the limited timeframe of a senior design project, and in depth knowledge needed for each component of the system is another constraint. After initial research and combing of the code from the previous year, the software subteam team divided the necessary tasks and each member focused on one component of the overall process.

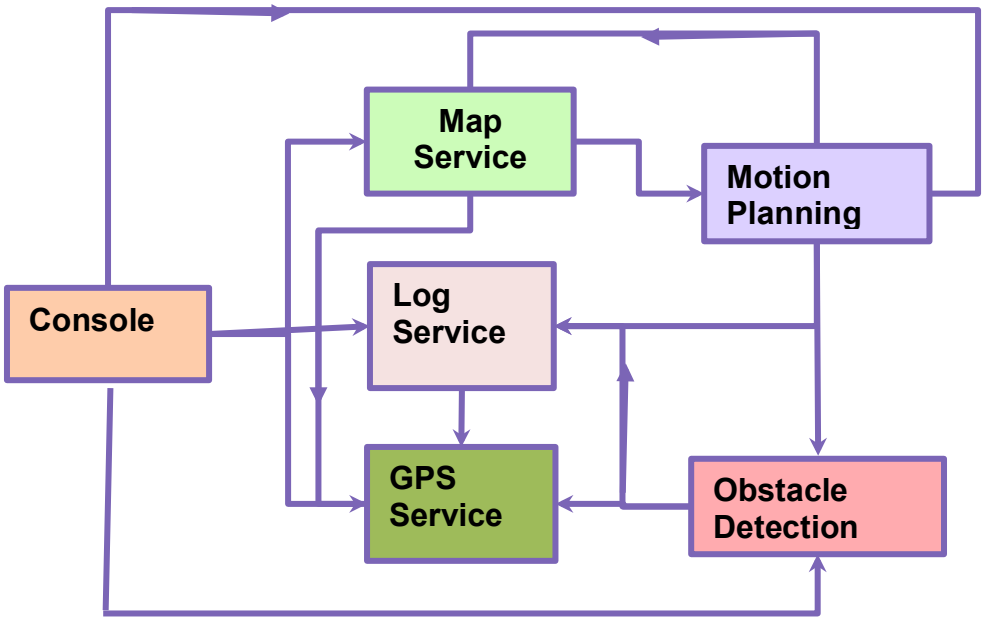

Figure 5: Communication Framework of Software Components

Obstacle Detection and Avoidance

The primary goal of the line and object detection system is to precisely calculate the location of lines and obstacles in a short amount of time. Parallelizing the processing of the image data is used as much as possible to achieve this. The hardware used for parallelization is the Graphics Processing Unit (GPU) provided on the NVIDIA TX1 board which operates as the brain of the vehicle. The NVIDIA GPU provides the speed necessary to process very detailed high-resolution images.

Based on the previous year's experimentation, the team decided to continue using the innovative stereoscopic camera called ZED to produce images that are then processed through the GPU. However, the team decided to also use the ZED camera for object and line detection. The ZED consists of two cameras and a powerful suite of libraries to assist in data analysis. The ZED's direct integration with the GPU allows the algorithms to be processed almost exclusively on the GPU where the calculations can be done in a much more optimized manner.

Implementation of line detection begins by capturing the image from the left camera of the ZED. To calculate the line it starts with a Gaussian blur that spreads out the pixel. Then canny edge detection is performed on the image followed by dilation to destroy more of the definition. Finally, probabilistic Hough line detection is performed which looks at the left and right neighbors to determine the highest contrast. This

reports the coordinates of the lines and combined with the point cloud returned from the ZED camera, the lines are mapped as orthorectified objects. The point cloud is used to determine distance from camera.

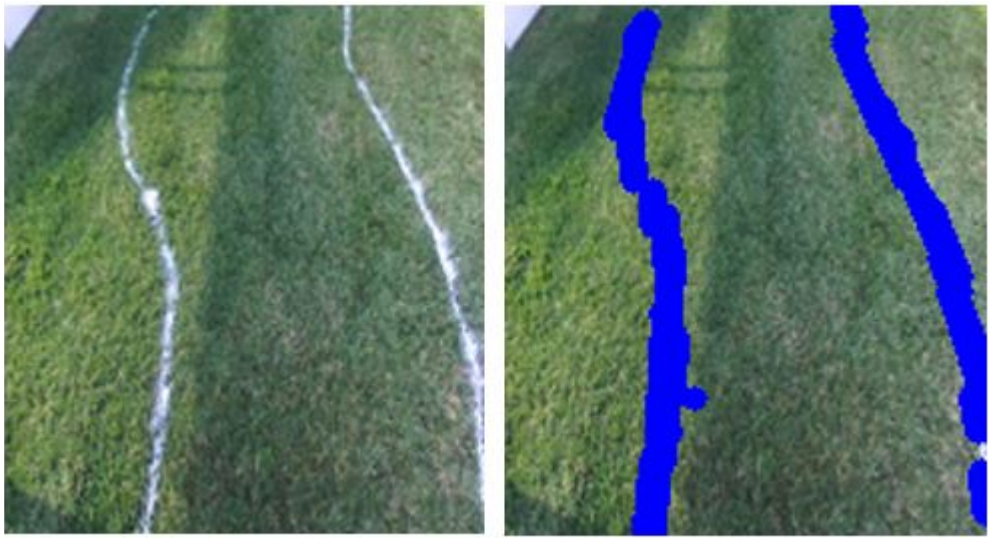

Figure 6: Example of Line Detection using Prepared Image

Object detection is performed using a depth map returned from the ZED Camera. The depth map image is then processed from the top to the bottom using raster lines. The raster lines detect an object when a depth jumps a set parameter forward from the normal depth line. This will allow detection of objects close to the camera while ignoring those farther away. Once an object it detected by a raster line, then the object is orthorectified and placed on a map using the point cloud.

Motion planning

The motion-planning module is responsible for receiving a premade map from the mapping module, calculating efficient paths to waypoints on the course (pathfinding), and generating a sequence of movements that are consistent with the constraints of the robot in question to such path. In order to ensure fast and reliable calculations, a Markov's Decision Process (MDP) is used to produce an adaptive pathfinding planner. It is required to dynamically calculate and recalculate the fastest way to get to the goal.

MDP is a technique which gives the map a certain amount of reward to aid in getting the vehicle to the goal. Finding a path will always result in an appropriately generated trajectory without extra processing. Smoothing is needed to convert the original path from the MDP into a control for the motor controller. The result from the MDP will also give a correct action for each state that the vehicle is in.

Map Generation

The goal for mapping is to generate a dynamic map using localization data from navigation modular and the obstacle data from obstacle detection. The output of

mapping is a list of bit stream to show obstacle data and pass it to the motion planning modular. For example one instance of a map is a matrix of 0s and 1s with 1s representing an obstacle. Currently all obstacles are assumed to be static and in the shapes of a square or circle. The map is getting constant updates that replace the old map within a certain degree, for example, 10 maps will be saved in the Raspberry Pi and the 11th will replace the first map. This module does not manipulate data from obstacle detection because localization data is processed by the ZED camera. The module assumes that it can obtain the location of the robot and the obstacles to generate a map. Its main functionality will be storing maps and updating them to motion planning.

Challenge: Obstacle Detection modular will take care of shape of obstacle as well as some difficult situation such as a circle that appears rectangular when facing it in front. Apart from that, the limited memory (2GB) of a 3-B model Raspberry Pi could cause problems and affect the amount of maps stored at a certain time since the maps are updated in real time and there will be maps stores in the pi for comparison e.g: several comparison will be enough to determine an obstacle.

Goal Selection and Path Generation

The GPS waypoints shall be entered into the system prior to heading out for the run. The waypoints are then entered into a queue and when the vehicle is within an acceptable range they will be removed and the next waypoint will be activated. For generating the path, the goal will be activated and the policy map generated by the MDP will be traversed to come up with the optimal path. After the path is found It is then smoothed and the velocity for each wheel will be generated and sent to the motor controller.

Additional creative concepts

Communication Framework:

Because of the extensive amount of computation, we resorted to distributing the workload across many inexpensive machines. Micro services were created to perform each individual task required by the motion-planning client. A gRPC rpc framework is used to allow for all the different services to communicate. This allows the software to write network procedure calls over a network at a high level in all the three computer languages used.

Failure Modes/Points and Resolution

Table 2: Failure Mode/Point and Resolution for PETE

	Failure Point/Mode	Resolution
Mechanical	1. Vibrations disrupting	1. The camera pole is clamped to the

	images from the camera 2. Water seepage into vehicle 3. Loosening of bolts and nuts	chassis and bolted at the bottom 2. Layer of foam is used between the outer panels and the chassis 3. Lock nuts and washers are used to prevent bolts and nuts from loosening from vibrations
Electrical	1. Overheating the electronics 2. Overcurrent from the battery and motors 3. Loose wires that may disconnect	1. Four computer fans are used to create a constant airflow inside the vehicle 2. Fuses are placed in the circuit to prevent damage to the electronics from the overcurrent 3. All wire connections are soldered and tightened
Software	1. Communication error due to the different languages used 2. Overloading the memory of the Raspberry Pi with data 3. Incorrect software uploaded to the Raspberry Pi	1. A gRPC framework is used to write network procedures in all three languages used 2. The amount of data being stored, replaced, and deleted by the Raspberry Pi will be regulated and refined through testing 3. Software is modularized with each Raspberry Pi having its own software to run

Performance Test to Date

Manual Navigation: The manual control of the vehicle has been tested through the use of the RC controller. The test shows that the vehicle is capable of switching between modes and navigating a field using the RC controller.

Obstacle and Lane Detection: The obstacle and lane detection software tests have been performed under manual control. The camera and processors for these detections were run as the vehicle moved under RC control. The maps generated with the lanes and obstacles were viewed from a external computer using wireless connection. A member of the team stood within the visible range of the camera with the software showing the member's presence.

Wireless Emergency Stop: The wireless emergency stop from the RC controller is capable of stopping the vehicle when engaged. The vehicle was also unable to move while the emergency stop was engaged.

Mechanical Emergency Stop: The mechanical emergency stop subsystem has been built and tested.

Initial Performance Assessment

To date, PETE is capable of being controlled using the RC controller. Both the wireless and mechanical emergency stops are functioning so that the vehicle can be stopped when needed. PETE is able to transverse both grass and asphalt terrain. The vehicle has also been able to climb relatively flat slopes without the payload. The exact elevation of the slope that it can climb will need to be tested. White lines on the asphalt road were detected by the camera and viewed on the external computer. The vehicle is capable of detecting obstacles that are within the vision of the camera. From the current performance of the vehicle, PETE is ready to integrate and test the autonomous functionality of system.