INTELLIGENT GROUND VEHICLE COMPETITION 2018

TEAM ABHIYAAN

KERNEL 2.0

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

Design Report

May 15, 2018

I hereby certify that the development of vehicle, Kernel 2.0, as described in this report is equivalent to the work involved in a senior design course. This report has been prepared by the students of Team Abhiyaan under my guidance.

Dr. Sathyan Subbiah
Faculty Advisor, Team Abhiyaan
Associate Professor, Department of Mechanical Engineering
IIT Madras

Team Leader - Sourabh Urade (uvw8275@gmail.com)
Team Members - Sourabh, Prashanth, Shreyash, Sriram, Kumud, Ashish, Veerendra, Fakheem, Athul, Favas, Fariz, Sajay, Anudeep, Hrishikesh, Nishant, Saikat, Ramana, Narendiran, Milind, Meenakshi, Satwik, Rachit, Bakthakolahalan, Karthik, Ravi, Vineeth, Rohith, Ujjwal, Ganesh, Sooryakiran, Kishore, Ankita, Shivani, Ashwin, Samar

Contents

Figure 1: Kernel 2.0

1 Introduction

Team Abhiyaan is a group of 35 interdisciplinary students enrolled in undergraduate as well as postgraduate engineering programs of Indian Institute of Technology (IIT) Madras. Fueled by common passion for autonomy, we work in Center For Innovation (CFI), a 24*7 Innovation Lab in IIT Madras. We have considered all the failures and shortcomings that Kernel had faced in IGVC 2017 and reassessed our thought process and brainstormed to come up with novel and innovative ideas. Kernel 2.0 is our second prototype manufactured for the purpose of participating in IGVC and this design report serves to document all the details.

2 Team Organization

The team is organized into 4 modules: Mechanical, Electrical, Software and Sponsorship & Design; based on the expertise and domain knowledge that is required in building the prototype and striking sponsorship deals for components and logistics. Module heads work with their team members and are responsible for each individual module. The team heads coordinate with all the module heads, organize team meetings and bonding sessions, track the progress of team and ensure that things are going smoothly as planned. Our design process involves a sweet spot based strategy where we ideated ways to achieve the design objectives in a cost-efficient manner without compromising on performance.

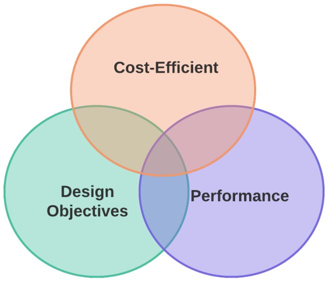

Figure 2: Design Process of Kernel 2.0

3 Innovations

- **Ramp detection**
  We use a vertical LiDAR to detect ramps, and remove them from the laser scan data so that ramps are not detected as obstacles.

- **Barrel detection**
  We use laser scan data from the horizontal LiDAR to identify and reconstruct barrels in 3D. This information is then used to remove barrels from the camera image. This is necessary to avoid white marks on barrels being detected as lanes.

- **RRT**
  We use a random-tree based path planner for quickly generating the optimal path given a set of waypoints considering the traversability of the path because of obstacles, lanes etc. We introduce several modifications and heuristics to make the algorithm better.

- **Pure Pursuit**
  We use a dynamically varying goal directed algorithm (Pure Pursuit) in the controller module. It computes short-term goals, plans a smooth path for the vehicle to reach it and generates the differential drive velocities based on the trajectory and vehicle kinematics.

- **Single Battery**
  Kernel 2.0 is powered up by a single battery entirely.

- **Efficient Stepping Down of Voltages**
  Stepping down voltages to the respective components has been done efficiently to increase the runtime of the bot.

- **Dual Quadrature Encoder Interface**
  The micro controller TIVA TM4C123GXL used this year supports dual quadrature encoder interface. So only a single TIVA Board is required instead of two Arduino Dues as was required last year.

- **Chassis Material**
  Hollow aluminum rods (ISO10799) are used for chassis fabrication which decreased the weight of the bot by 40%. Use of aluminum, at the same time, gave the bot considerable amount of strength.

- **Shrinkage reduction**
  Temperature change can be major problem for the teams having different weather conditions at home than that of the competition region. Tig welding of the aluminum was used which decreased the internal shrinkage to 0.17mm.

- **Ramp climbing**
  The bot can climb an inclination upto 15 to 20 degrees. Low weight of the bot and high torque from motor maintains the required speed of the bot.

4 Mechanical Design

4.1 Introduction

The primary objective of mechanical module of Team Abhiyaan is to build an efficient bot considering both mechanical and electrical aspects. While designing the bot we concentrated on bot's structural rigidity, weight optimization, low center of gravity for increased stability, accessibility to all the components and rules compliance.

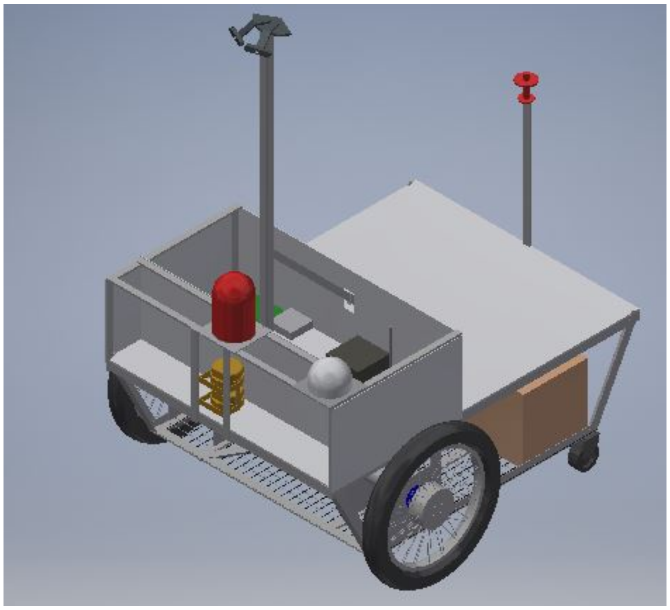

Figure 3: 3D model of the bot.

4.2 Chassis Design

Mechanical designing consists of chassis design and analysis, material selection, motor and gear combination, 3D modeling of parts and assembly. Bot has four wheel, two wheels connected to motor in the front and two follower caster wheels on the rear side.

- **Modularity**: In Kernel 2.0 we tried to overcome various problems faced by previous year's bot. The whole design of the bot was revamped. The chassis is made modular by dividing it in two parts, namely, upper and lower berth. Mechanical adhesives i.e. nut, bolts and L-clamps are used to join upper and lower berth. Keeping transportation constraints in mind, the lower birth is further divided into two parts. Allowance for sensor mounting and parts placement was considered according their requirements while designing. Total width of bot is 2 feet 2.5 inches and length is 3 feet 3.5 inches.

- **Modeling**: Autodesk Inventor was used for modeling of the bot and to create manufacturing drawings. The chassis, all the sensors, motor assembly, wheels and bot covering were modeled and assembled on the software. The upper berth and the lower berth are made from hollow squared aluminum rods, cut and welded according to the dimensions. We designed the bot in Autodesk Inventor according to the rules and then manufactured it. The berths are constructed precisely according to the dimensions. Hollow aluminum rods are used in order to give strength and reduce the weight.

- **Material Selection**: It was decided to use aluminum for the bot frame due to its relatively low weight for its strength and cheaper cost compared to other materials. The chassis is the structure which bears overall load of the bot and it needs to be strong and rigid enough to carry this load. It was found by analysis that the strains in the bot were low for the material AL6067. Initially it was planned to use L-shaped beams to manufacture the chassis but since the square beams have higher area moment of inertia which makes them difficult to bend, we chose AL6067 square beams of dimensions $20mm \times 20mm \times 2mm$ as an optimal solution for manufacturing the outer skeleton where they are welded together. The weight of the chasis was calculated to be 9kg on analyzing the model.

- **Wheel and Motor Assembly** The castor wheels at the rear of the bot are welded to the aluminum frame by means of an aluminum plate which provides a reliable connection for the castor wheels to the bot body. Aluminum (AL6067) was selected for manufacturing of motor mounts. The motor mounts carry a large percentage of stress from the wheels and house expensive motors making their structural integrity of paramount importance. Thus it was decided to use 5mm thick aluminum plates since it provides extra strength against larger bending torques even though it commands a higher weight. Two beams running along the length of the bot on both the upper and lower floors of the bot support the motor mounts. An extra beam is added across the bot on the lower level to counteract higher bending forces on the lower level.

- **Motor and Gear combination**: The possible motor types and gear ratio are selected by calculating the torque required for driving the vehicle at maximum required speed. Total effective effort can be given as:

$$T = \frac{TTE * D}{2 * n} \tag{1}$$

where, T = torque per wheel
TTE = Total tractive effort
D = Diameter of powered wheel

$$TTE = RR + GR \tag{2}$$

where, TTE = Total tractive effort
RR = Force necessary to overcome rolling resistance
GR = Force required to climb an inclination

$$RR = M * g * C_{rr} \tag{3}$$

where, M = Total weight of the vehicle
g = gravity
C_{rr} = Surface friction

$$GR = M * g * sin(\theta) \tag{4}$$

where, θ = Maximum inclination

$$\omega = \frac{2V}{D} * \frac{60}{2\pi} \tag{5}$$

where, ω = Angular velocity of wheels.

For M = 70kg, g = 9.81 m/s^2, C_{rr} = 0.055 (grass), θ =15 and D = 457.2mm (18 inch):
$RR = 37.76N$
$GR = 178.545N$
$TTE = 216.31N$
$T = 24.72Nm$
Therefore,

$$V_{max} = 2.23m/s(5mph) \Rightarrow \omega_{max} = 93.24RPM$$

$$V_{min} = 1.34m/s(3mph) \Rightarrow \omega_{min} = 56.03RPM$$

According to above results, following combination of motor and gear chosen:
Motor:

S No	Specification	unit	Value
1	Nominal Voltage	V	24
2	Nominal Current	A	10.8
3	No load speed	RPM	5950
4	Nominal Torque	mNm	405

Gear:

S No	Specification	unit	Value
1	Reduction	-	66:1
2	Max. continuous torque	Nm	30
3	Mass inertia	gcm^2	16.7

4.3 Manufacturing of the bot

- **Process Planning**
  The bot is divided into two parts, upper and lower berth. Due to logistical constraints the lower berth is further divided into two parts. The upper berth carries LiDAR, PCB, NUC, camera, router and IMU sensor whereas the lower berth carries motors, wheels, payload, E-Stop and wheels. The upper berth was manufactured first followed by lower berth and motor to wheel coupler.

- **Manufacturing** The aluminum chassis was manufactured at the IIT-M Central Workshop. For high precision, CNC machines were used. Wherever aluminum plates were involved, tig welding was carried out using highly skilled labor. A strict timeline for manufacturing was followed to devout sufficient time for testing and rectify any problems.

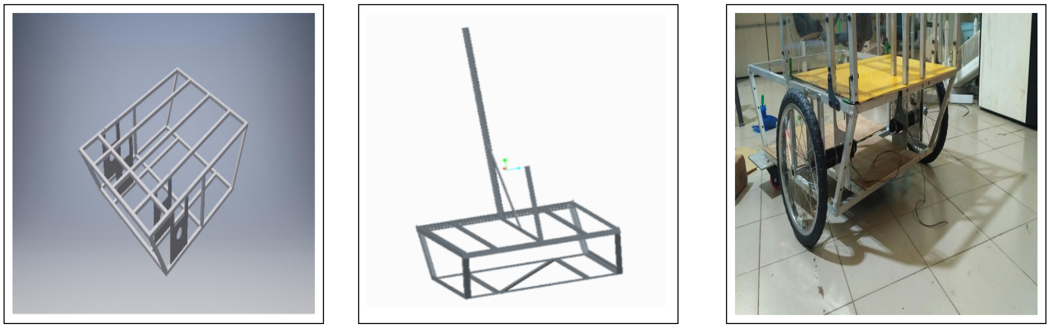

Figure 4: Assembly of the bot.

5 Electrical Design

5.1 Introduction

We have designed the electric circuitry of Kernel 2.0 in such a way that it is safer than the previous and runtime of the bot has been increased considerably. We implemented a distributed power system this year that uses a single battery. We also developed a compact and customized Printed Circuit Board to cater to all features of the bot. TIVA TM4C123GXL is the micro controller used. Motors and motor drivers have been calibrated in a better way to reduce odometry errors to a large extent.

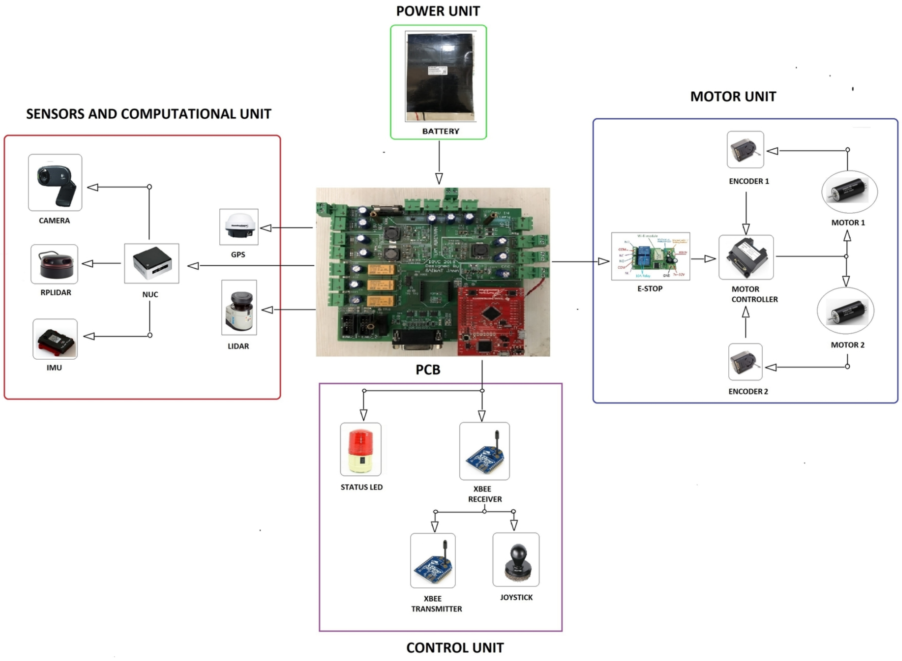

Figure 5: Electrical Flow diagram.

5.2 Features of PCB

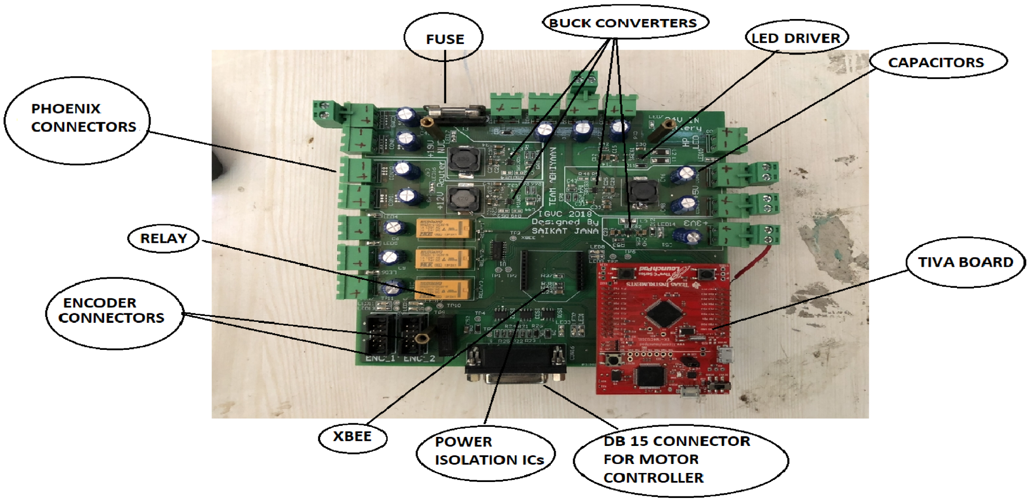

Figure 6: Components of PCB.

5.2.1 Power Distribution

The primary power for all systems is provided by the designed Printed Circuit Board. All the components in the vehicle are powered by a single source of 25.9V, 46.8 Ah Lithium ion battery. The PCB supplies 3.3V, 5V, 12V, 19V and 24V to the required components. Different buck converter ICs are used from efficient stepping down of voltages. These ICs turn on and off using PWM. Thus by using the desired duty cycle, we get appropriate power supply with an efficiency of more than 95% in conversion of voltages. This is better than the Linear Voltage Regulators used last year which were not energy efficient. ICs used for power distribution are

1. TPS54202 - For converting 24 volt input to 5 volt output. This 5 volt is used for powering other IC's and for powering SC189
2. TPS54202 - For converting 24 volt input to 12 volt output. This 12 volt is used for powering relay and router
3. TPS54302 - For converting 24 volt input to 19 volt output. This 19 volt is used for powering NUC.
4. SC189 - For converting 5 volt input to 3.3 volt output. 5 volt is output of TPS54202 and output 3.3 voltage is used to power XBEE and SI8621.

5.2.2 Signal and Power Isolation for Roboteq

No path must be created between the ground terminal of the Input Output DB15 Connector of Roboteq and ground terminal of the main power supply. In case the controller's ground terminal is disconnected but power terminal is still connected, high current may flow through controller which may damage the controller. Thus the signal and power should be isolated. We use RE0505S to power isolation and to create Isolated continuous power supply for input/output connector of

Roboteq Thus they create an electrically isolated supply from the main power supply from the battery and works like a secondary battery. SI8621 is used to isolate 3.3 V signals data transmission signals. SI8620 isolates 5V encoder channel signals.

5.2.3 LED Driver

LEDs are current controlled devices. Hence the LED driver AL8860 maintains constant current. The IC can take upto 1.5A and the inductor used regulates it to 300mA. The input Supply is up to 40V. By changing the frequency, the LED can be made blinking or solid depending on manual or autonomous mode of Kernel 2.0.

5.2.4 Evaluation of PCB

Several test points have been placed around the PCB for troubleshooting any problem. LED indicators green in color has been used to indicate power transmission whereas red LEDs are used to indicate data transmission.

5.2.5 EMI Radiation minimizing

1. As the LED Driver switches on and off at a higher frequency, an alternating magnetic field is created which leads to production of electromagnetic radiation. These waves affects the IMU data, NUC or any sensitive electronics. Thus the magnetic field is shielded by using ferromagnetic material.

2. Loops in the PCB act as inductor coils and enhance electromagnetic interference. Thus the area of loops is minimized to reduce the EMI radiation.

3. Thin ground wires have been used to prevent back flow of EMF and flow to other components since it has higher resistance.

4. Noise generation during communication has been reduced by keeping the communication lines away from noise generation circuits and keeping them near ground.

5. Schottky diode is used for fast switching of relays and it also has very less voltage drop. Ferrite bead is used to reduce the electromagnetic interference generated due to fast switching of relays

5.2.6 Safety

1. Resettable Fuses are installed in the PCB which provides resettable overcurrent protection. Once the circuit is broken for higher currents, it makes the circuit again after a small time interval.

2. Buck Converters have short-circuit protection. Thus the maximum current won't go above a specified limit in any case in PCB.

3. A diode is placed to prevent plugging input terminals in the reverse polarity.

4. If the input voltage is less than 24 volt, the power circuit is cut off which provides protection against low voltage.

5.2.7 Miscellaneous

1. Relays are operated by providing signals from TIVA through ULN2003 Darlington Transistors.
2. Output voltages of three terminals can be chosen between 5V, 12V, 24V by populating the corresponding resistor.
3. The Tiva board can be powered either through PCB or through USB while priority is given to USB by using a diode.
4. The battery voltage levels will be sent as messages to the ROS system from PCB. So in case the battery levels go below 22V, all controls from software would be stopped to prevent damage to the components.

5.3 Power Consumption

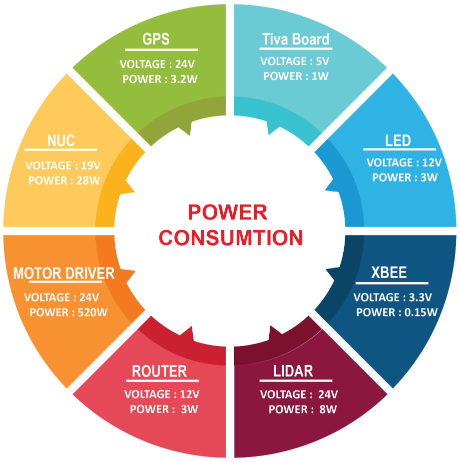

Figure 7: Power Consumption Chart.

Kernel 2.0 uses a Lithium-ion battery of rating 25.9V-46.8Ah with total capacity of 1200Wh. The Battery has a built-in battery monitoring system (BMS) for cell balancing, over-voltage, under-voltage and short-circuit. As the figure shows, the Lithium-ion battery provides 24V to the LiDAR, Motor Controller, GPS and Buck converter that converts 24V to 19V to power up the NUC, 12V to power up the Router and LED, 5V to power up the Tiva-board and 3.3V to power up the XBEE. It also provides the required voltage for powering up the DC to DC buck converter ICs, Relays, Led Driver IC etc. Having a single source for everything makes Kernel 2.0 more compact and simple. The maximum estimated power consumption including consumptions by each component, DC-DC conversion losses and technical losses (including Heat losses) is 680W.

$$\begin{array}{l}\text{Estimated Runtime considering}\\ \text{Maximum power consumption}\end{array} = \begin{array}{l}\text{Total Capacity divided by Max}\\ \text{estimated power consumption}\end{array} = 1.76\text{ hr}$$

The runtime of 1.76 hours is when the bot is under maximum power consumption. In normal running cases, we had a runtime of over 4 - 5 hours during testing period. The batteries takes about 2 hours to be completely charged. Kernel 2.0 uses Lithium ion batteries having more volumetric as well as specific energy densities compared to last year's Li-Po batteries. This reduces payload as well as storage space. All battery connections are made by using XT-60 connectors with appropriate power ratings.

5.4 Motor Interface

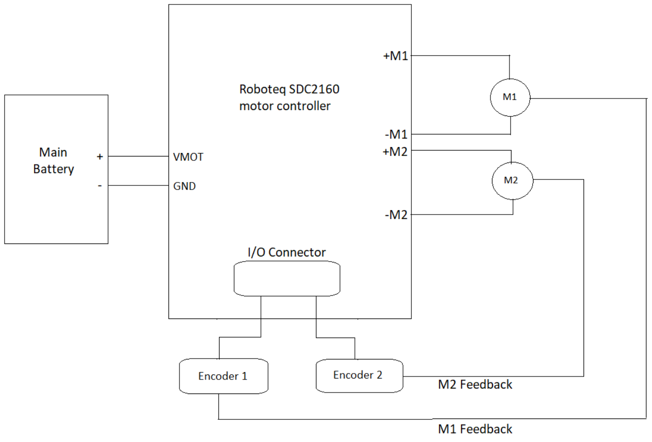

Figure 8: Motor Interface.

Kernel 2.0 uses the Roboteq SDC2160 2x20A high performance dual channel brushed DC motor controller with HEDS-5540 quadrature encoders. The controller is operated by Robot Operating System (ROS) through serial communication. It includes an elite 32-bit microcomputer and quadrature encoder input to perform motion control algorithms in closed loop speed mode. Closed loop speed modes ensure that the motor(s) will run at a precise desired speed. It collects the motor's feedback from the encoder through digital input pins of the controller which utilizes PID loop control to improve the precision of speed. This controller has internal memory which is able to hold configuration settings that controls operation of motors.

5.5 Wireless Joystick

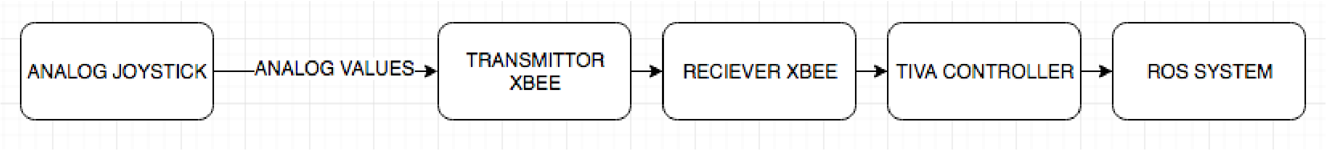

Figure 9: Flow of Data.

A wireless joystick has been designed for controlling the bot manually when it is not in 'autonomous' mode. The joystick is an analog 2-axis thumb device. Data is transmitted through XBEE S2C which has an operating range of 2000 feet. The wireless Joystick can also switch between manual and autonomous mode. It could also stop the bot in case of an emergency.

5.6 Emergency Stops

The motor power is connected through the RF module on the bot. This RF Module helps to control the circuit wireless by switching it ON/OFF using a remote control and has a range up to 100 meters. This serves as the Wireless E-stop, added to the Mechanical E-stop (Push-button) on the vehicle. Both the Wireless E-Stop and the Mechanical E-Stop turn off the power of only motor instead of turning off the entire vehicle and it's components which may cause damage to NUC, Router and other electronic devices. Both these devices are hardware controlled. The Wireless Joystick stops the bot through software control.

6 Software Strategy

Kernel 2.0's software stack is built in Robot Operating System (ROS) because it offers a simple interface with sensors through *nodes*, *topics* and *services* to help manage the hardware abstraction and low-level control. Due to its popularity and largely active community, there are ever-growing contributions and the APIs are well documented which accelerated the development of our software stack. It is scalable, provides graphs of processes for easy debugging and allows control of the robot through multiple networked machines.

6.1 Introduction

We designed our software stack to be modular so that any component of the stack can be replaced independent of the other components. This allows for major changes and replacements in the code-base with minimal change to the existing code. We also developed our own algorithms that are robust and can be customized to suit our needs.

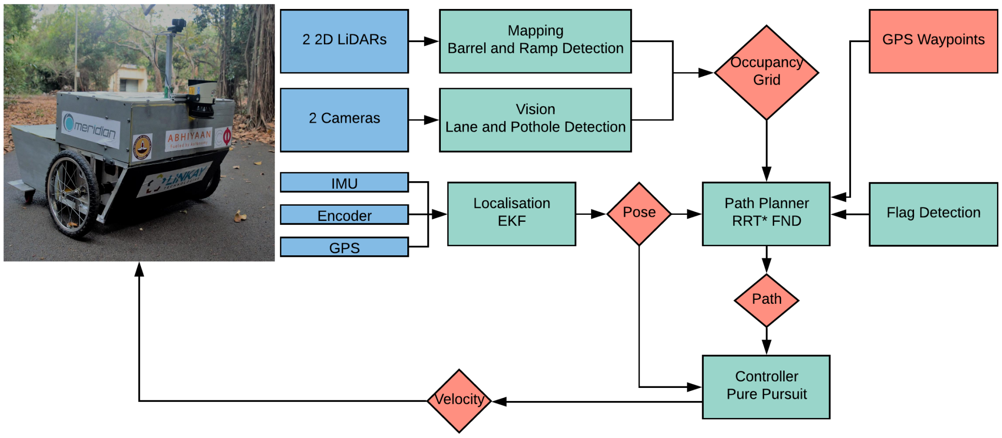

Figure 10: Software flow

6.2 Mapping and Obstacle Avoidance

For indoor regions, the GPS data is insufficient for accurate localization. So we use a 2D LiDAR to perform Simultaneous Localization And Mapping (SLAM). It scans the environment horizontally


(a) Original Image
(b) Thresholded weighted combination of red and blue channels
(c) Lanes interpolated using regression

and laser scans are fused with odometry data to map the surroundings. In outdoor regions, localization is performed using Extended Kalman Filter(refer Section 6.3) and the obstacles are detected using laser scans. An occupancy grid is constructed using this information. We also perform ramp detection with another 2D LiDAR in the vertical direction and remove the cost of the ramp from the grid to ensure that ramps are not detected as obstacles. Further, the path planner module subscribes to this occupancy grid and constructs path.(6.5)

6.3 Localization

Localization is essential for getting accurate pose estimates of the robot. The robot must be able to locate itself in a particular frame of reference and must be able to keep track of its state as it moves. Since we have to localize in a sparsely populated outdoor environment, SLAM is not a good choice as the features are sparse. Hence we resort to using GPS coordinates (latitude and longitude) as the major source of location data. We also get pose estimates from the wheel encoders which track the odometry of the robot. However we observe much more drift and increase in covariance of the wheel odometry data and hence we use the GPS as the primary source of position data. The IMU helps in getting accurate orientation and angular acceleration data which can be fused together with the GPS position data to get an accurate pose estimate of the robot. The IMU also offers us data about linear acceleration however we observe this to be noisy and hence discount this data. We fuse all the sensor data using Extended Kalman Filter which uses a non linear model to propagate the system state. Although vanilla KF which uses a linear model has been theoretically proven to converge unlike EKF, the latter performed much better in our case.

6.4 Computer Vision

The robot has to detect white lanes on green grass. To tackle this problem, we analyzed the absorption spectrum of a typical photosynthetic plant which has peaks in the red and blue area as expected. So red and blue channels are chosen for detection of lanes. Both are taken separately and denoised using Gaussian blur. They are combined using a weighing mechanism which helps in getting the maximum contrast of the lanes with respect to the grass and other objects in the environment. To ensure that white colored barrels are not detected as lanes we use LiDAR information to remove the barrels from the camera image and then apply a Gabor filter mask. We interpolate lanes using polynomial regression to counter the problem of barrels hiding the lanes and dashed lanes. Inverse perspective transformation is done at the final stage compared to the usual process of doing it first. This is because pixels representing farther points contain more noise to useful information compared to near ones. Doing perspective transform before processing inflates these noise filled points. The occupancy grid is updated with point cloud data obtained which treats lanes as obstacles.

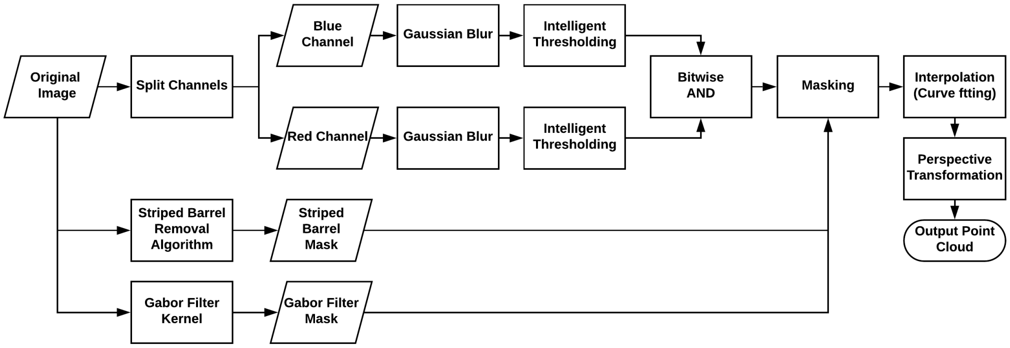

Figure 12: Lane Detection

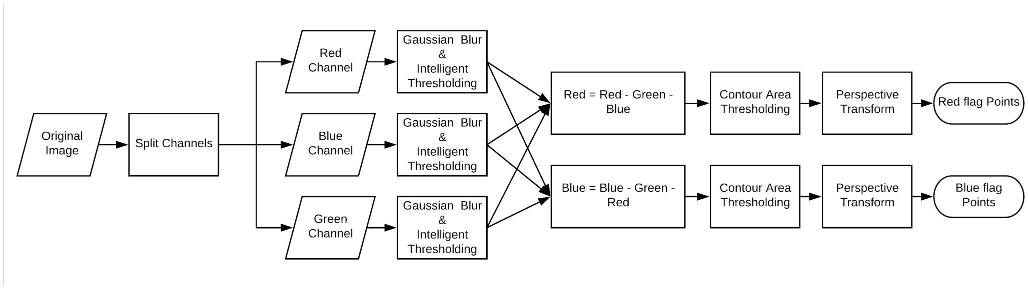

Figure 13: Flag Detection

6.5 Path Planning

The path planner sub-module computes a low-cost, near-optimal path using a GPS waypoint as the destination. We use a custom RRT based path planning algorithm that can efficiently deal with unseen obstacles without the need for re-running the algorithm again and again. The Rapidly exploring Random Tree (RRT) algorithm runs on a costmap, which represents the cost of each grid as the probability of it containing an obstacle. The map comprises of large costmap grids resulting in large number of edges in the graph. This makes finding the optimal path computationally intensive. Probabilistic Roadmap Models (PRMs) are used to overcome this challenge and RRT is one such algorithm. It involves a random tree rooted at the robot's current position and tree grows towards the goal position. The environment may have dynamic goals as well as robot may have changing goal points. Only part of tree is removed whenever an obstacle is encountered and the forest of trees thus formed are recombined into one tree by random growth of vertices. The code was written from scratch and several heuristics have been employed to improve its performance in terms of path optimality as well as time and space complexity.

6.6 Controller

The controller gives appropriate control inputs to the robot to ensure that the planned path is followed. We use a pure-pursuit based controller to move the robot smoothly while taking its non-holonomic constraints into account. The controller essentially tries to mimic the human driving nature by taking a point at a fixed distance in front of the vehicle as the immediate goal. This goal changes continuously as the vehicle moves on the planned path, traversing through a coarse set of waypoints. Once this look-ahead goal point is found, the controller algorithm plans a smooth curvilinear path from the current position of the vehicle to this point which is used to generate velocities to be given to the wheels for following the intended course.

6.7 Simulation

Gazebo, an open-source platform has been used for the simulations as it is highly integrated with the ROS communication framework and the codebase also uses ROS's latest communication framework. A simplified 3D model of the bot in the form of URDF file was then created that adds metadata to the meshes. The main aspects of metadata include collisions, joint limits, inertia. The sensor data collected from the plugins is made available through topics to other submodules that we have deployed. This allowed us to have an easy understanding of what the vehicle is seeing when it encounters a situation, which facilitates easier debugging and optimizations.

7 Initial Performance Assessments

- Lanes have been detected upto 10 feet with 140° field of view.
- RRT is able to plan at 10Hz along with dynamic planning and provides a stable path to the controller.
- Ramps and barrels are getting detected upto 10m distance and are getting removed from the camera images
- Outdoor localization is providing an accuracy of around 10cm using GPS, wheel odometry data and IMU
- The bot is able to move and turn as per the instructions from controller.
- Ramps of an inclination of 14 degrees are covered with average speed of $1.5 m/s$.
- Runtime of Kernel 2.0 with nominal load was found to be 4 hours and with full load is estimated to be 1.76 hours and .

8 Failure Modes

S.No.	Failure Mode	Resolution
1	Detection of white barrels as lanes	Detect barrels through Horizontal LiDAR scan and remove them from camera image
2	Detection of ramps as obstacles	Detect barrels through Vertical LiDAR scan and remove them from camera image
3	Mismatch of Sensors and device dev paths	Use *udev* rules to create symbolic links for device dev paths
4	Failure of one or more critical sensors	Stop vehicle for such critical cases
5	Incorrect plugging of wires	Phoenix Connectors used allow plugging in only on direction and also diode provided in input terminals for additional safety
6	Holes off-centering due to temperature change	Slotted L- clamps
7	Vibrations caused by loosening of the bolts	Stop the vehicle and fit them to avoid excessive damage

9 Cost Estimation

S.No.	Component	Retail Cost[1]	Team Cost[1]
1	Sick LMS1xx LiDAR	5000	0
2	RPLiDAR	600	600
3	Hemisphere A101 GPS	3000	0
4	Sparton AHRS-8P IMU	1500	0
5	Intel NUC	1000	0
6	Logitech Camera	150	150
7	Printed Circuit Board	70	70
8	Roboteq SDC2160	125	125
9	TIVA TM4C123GXL	15	15
10	XBEE S2C	15	15
11	Maxon Motors	937	629
12	Maxon Gears	738	508
13	Encoders	218	146
-	**Total**	**13,368**	**2,258**

10 Acknowledgements

We would like to thank our institute IIT Madras for providing us a platform to work on our own autonomous robot and giving us the chance to participate in IGVC 2018. We would like to thank the Center For Innovation (CFI) for supporting us through this journey to IGVC. We would also like to thank our faculty advisor Prof. Sathyan Subbiah for guiding us all along.

---

[1] All costs are in US Dollars