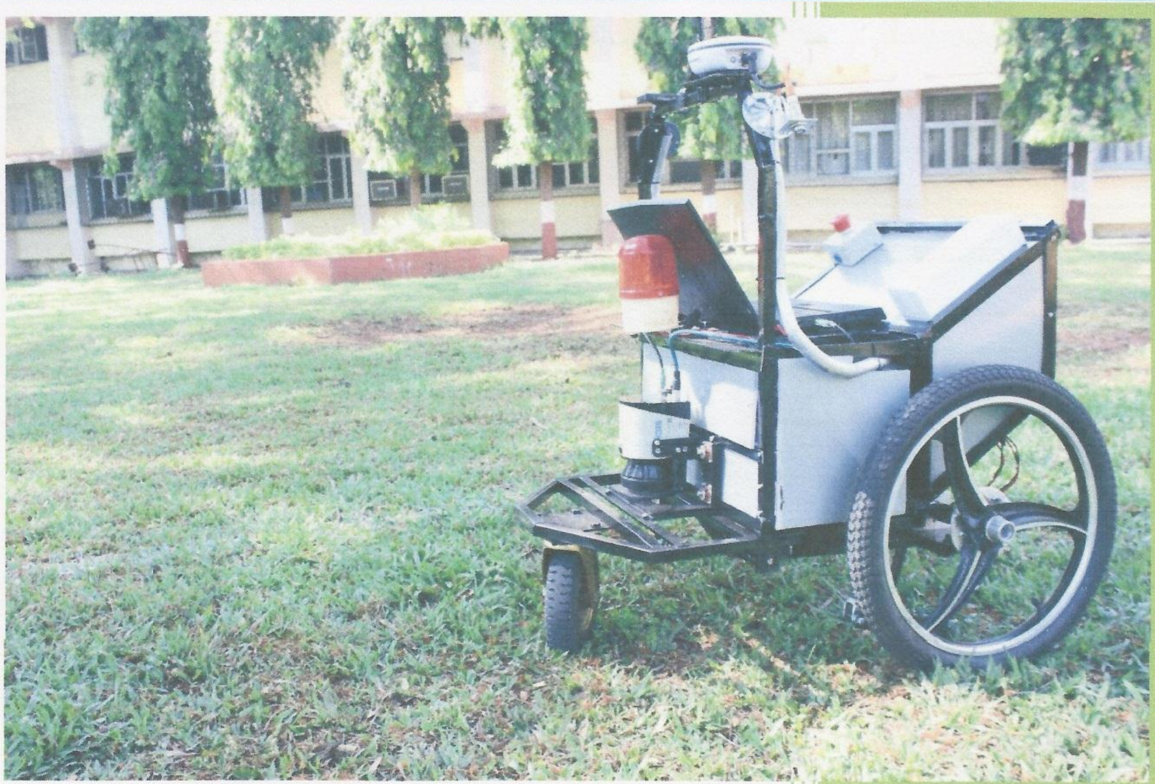
2013

Pushpak-2 IIT Bombay



Faculty Advisor Statement:

I hereby certify that the development of the vehicle, Pushpak-2, described in this report has been significant. This report has been prepared by students under my guidance.

Prof. Ramesh K Singh

सहायक प्राध्यापक/Asst. Professor
यांत्रीकी अभियांत्रिकी विभाग
Mechanical Engineering Department
भारतीय प्रौद्योगिकी संस्थान, बम्बई
Indian Institute of Technology, Bombay
पवई/Powai, मुंबई / Mumbai-400 076.

Prof. Ramesh K Singh
Prof. S. N. Merchant
Prof. Abhishek Gupta

Team IGVC
IIT Bombay

4/22/2013

CONTENTS

1 Team Overview

The Intelligent Ground Vehicle Competition provides an excellent opportunity for the student fraternity of the Indian Institute of Technology Bombay (India) to explore the design and implementation of advanced unmanned systems. The IIT Bombay team consists of students of various engineering disciplines, many of whom have worked on IGVC 2012 and the Autonomous Robot Design Challenge under Ministry of Defense, Government of India.

The team has 8 student members from various engineering disciplines. Considering the small size of the team, a flat team structure was opted for. The team has students involved in multiple roles as per requirement of the project. The team structure is as following

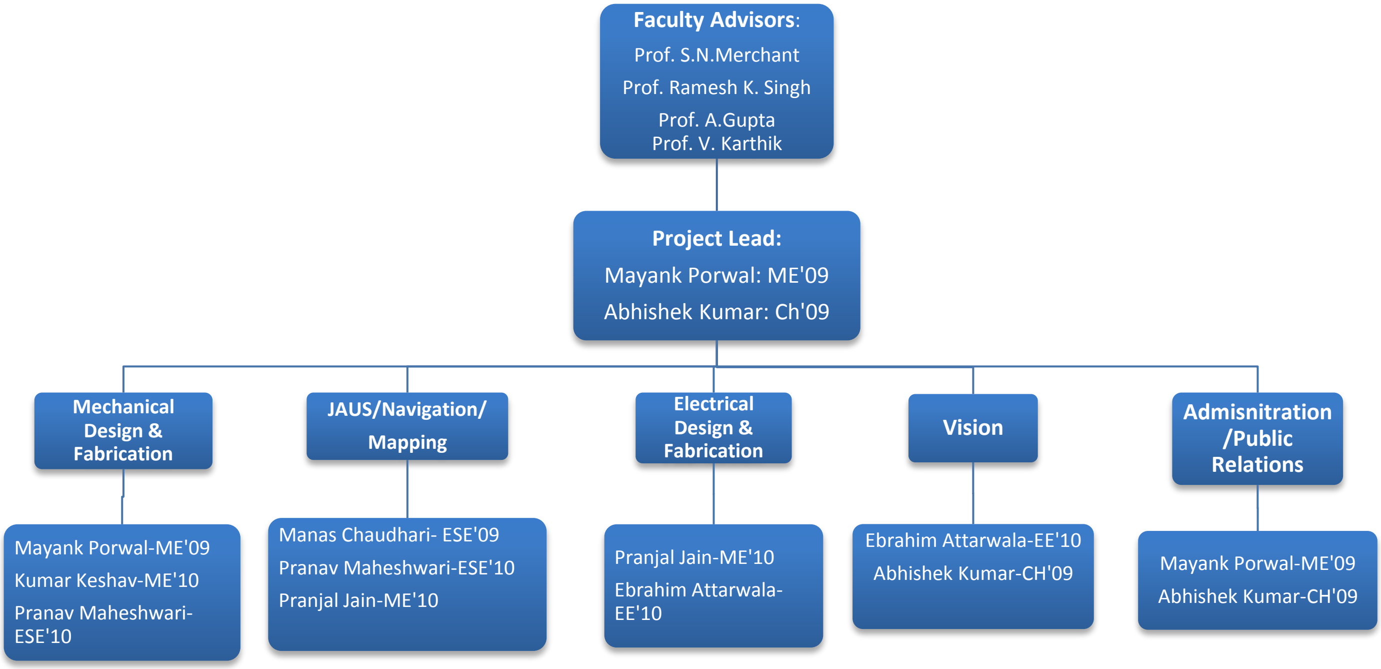
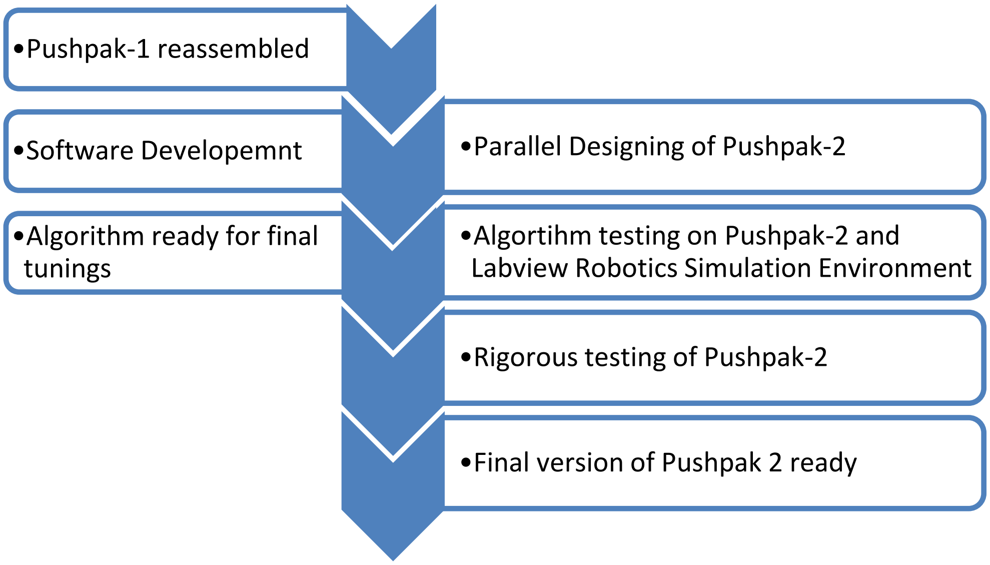


Figure 1-1 : Team Structure

2 Design Process

The design process has been described in figure 2. Since this is IIT Bombay's second year in the competition, several learnings from the past year experiences were taken and hence this methodology of working on parallel development was developed. Since a working prototype (last year's model) was already available, work was started on the software end while new mechanical structure was parallelly developed.



- Pushpak-1 reassembled
- Software Developemnt
- Algorithm ready for final tunings
- Parallel Designing of Pushpak-2
- Algortihm testing on Pushpak-2 and Labview Robotics Simulation Environment
- Rigorous testing of Pushpak-2
- Final version of Pushpak 2 ready

Figure 2-1 : Design Process

3 Design Specifications

Pushpak-2 measures 38 inches by 26 inches. The machine is designed for easy maneuverability in crunch spaces and hence we opted for a simple 3 wheeled differential concept with two driving motors and a free castor at the front. The height of the machine can be adjusted as per the requirement of the camera. The machine weighs around 28 Kilograms. This is a little heavier than the last year's version, considering that we have focused on the robustness of the machine. The machine has two 20 inch alloy wheels and a 6 inch castor. We have incorporated the used of suspensions and dampers as we felt their need from our past experience to ensure smoother ride and reduce equipment failure due to shock.

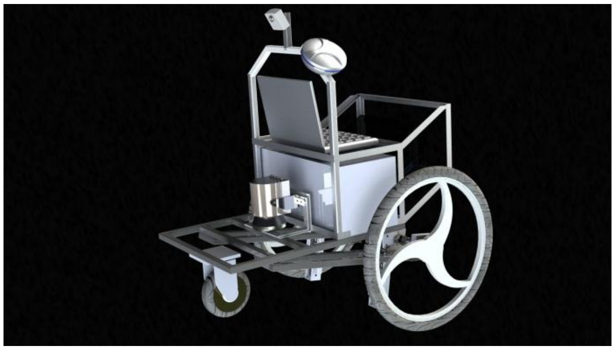


Figure 3-1 : CAD Rendering of Pushpak-2

The entire machine was designed using Solidworks, CAD modeling software from Dassault Systems. This reduced the prototyping time as many problems could be foreseen and avoided before actual fabrication started. The net weight, the center of mass and the moment of inertia of the machine were known beforehand.

Item	Actual Cost (USD)	Team Cost (USD)
Maxon DC Geared Motors	2300	2300
Wheels & Castor	100	100
Roboteq HDC2450 Motor Controller	645	645
Li Polymer Batteries	300	300
SICK LMS 111 Laser Range Finder	4300	0
Hemisphere A100 Crescent GPS Antenna	1900	1900
VectorNav VN-100 T IMU	1300	650
Allied Vision Pro Silica GT-1290	1300	1300
Sick UM-18 Ultrasound Sensors	600	600
FTDI-2232H USB-Serial Modules	35	35
Starcom PCI to 2 Ethernet express card	200	200
ThinkPad w530 laptop with Slice Battery	2300	2300
Hobbyking 2.4Ghz Wireless Remote + Receiver	25	25
Raw Material for Chassis & Fabrication	1000	1000
Miscellaneous Hardware	300	300
Total	**19616**	**10655**

Table 1 : Cost Analysis

3.1 Design Innovation

The key focus in design of Pushpak-2 has been its robustness. With the motive of product development, it has been ensured that the vehicle has a robust mechanical structure and is not vulnerable to factors like terrain, temperature, humidity and other factors.

The weight of vehicle without the payload is 28kg. The main chassis weighs around 6 kg. To ensure minimum shocks to equipment, suspensions and dampers were incorporated into the design. Since suspension for 15 kg load is not available in market, a unique idea of multiplying the force on the suspension was implemented.

As seen in the picture on next page , the distance of suspension (l1) and wheel mount (l2) are different hence the force on the suspension changes by a factor of (l1/l2). This enabled tuning the suspension by choosing proper l1 and l2.

With 90 % efficiency of the driving motors, the machine takes less than 80 watts during its normal run. Thus high energy density Lithium Polymer batteries are used that weigh only 1.5 Kg in total but can provide the required energy for 6 hours.

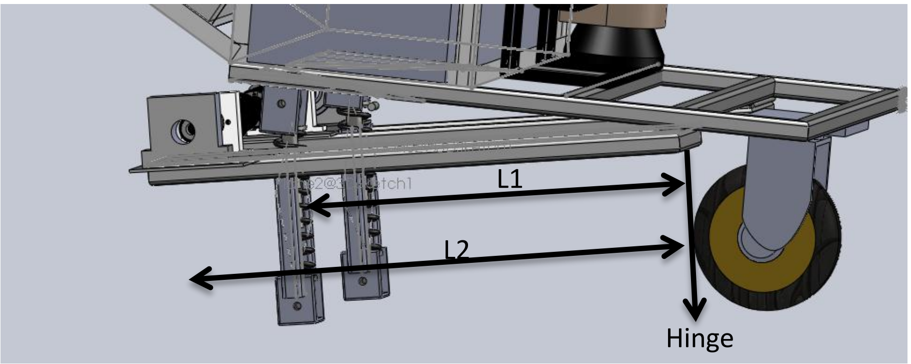


Figure 3-2 : Suspensions

The vehicle also has custom designed anti vibrating mounts for laptop, LIDAR and Camera to ensure minimum noise in the sensor readings and to reduce equipment failure. We developed in house silicone gel dampers to cater our needs. As a result of our design, there has not been a single failure this year as compared to several hard disk failures last year and camera reading errors.

Pushpak-2 is completely covered using polycarbonate sheet. The sheet provides a very high strength to weight ratio and also prevents any unwanted material from entering the machine. The electronics are placed in IP-65 boxes while all other sensors are also IP-65 compliant. This makes machine capable of running even is rainy and sandy environment.

One of the biggest advantages of the Pushpak-2 design is its modularity. The entire machine is made in 3 major parts, each connected to each other with 4 nut-bolts. It came out to be extremely useful when individual changes were to be made in the electrical or mechanical aspect of machine in a short span of time. Also this design comes in very efficiently in transportation, the three blocks can be fit to occupy very less volume and can be engaged-disengaged in about 10 minutes.

A very simple but effective method of isolating the sensor circuit from the high power drive noise has been created. The low weight of the batteries enabled easy usage of different power sources for the motor drive and the sensors. Since these power supplies are completely isolated, the mutual noise coupling is very low.

3.2 Fabrication and Assembly

The entire machine has been designed in SOLIDWORKS CAD modeling software. This increased the ease of manufacturing the machine. The number of iterations also went down as compared to last year. The machine’s lower part, that takes the major load has been made of SS-302 steel, whereas the upper part of the machine, that does not take major load is made of Aluminum allot 6061. We thus optimized the machine as per requirements of load bearing capacity. The frames have been welded using Gas metal arc welding.

The wheels are fit on a pair of deep grove ball bearings that take the load, which is then connected to the motor using a Universal (UV) Joint. The UV joint acts as a coupler and ensures that no force is transmitted to the shaft of the motor even if the axes of wheel and motor get mismatched.

The various sensors on the machines have been bolted so as to provide flexibility of shifting. The camera and the ultrasonic sensors have been mounted on custom designed stands that allow movements in both angular degrees of freedom. Pushpak -2 is completely covered using polycarbonate sheet. The sheet provides a very high strength to weight ratio and also prevent any unwanted material from entering the machine.

The box containing batteries and other electrical component has been attached to the back part of the machine through a linear guide. This facilitates easy sliding out of the box and hence easy changing of batteries without hampering other parts of the machine.

3.3 Mechanical Design

The maximum load on the machine is during a 15^{0} incline. Assuming a total weight of 40 kgs (with payload), and a wheel radius of 11 cm, the load torque on each motor is

$$\zeta = \frac{40 \times \sin 15 \times g \times 0.11}{2} = 5.6\; N - m$$

The machine has to achieve an average speed of 0.44m/s. The machine is designed to have a max speed of four times this value. Hence, the speed of the motor at rated load is

$$\omega = \frac{0.44 \times 4}{0.11} \times \frac{60}{2 \times \pi} = 152\; rpm$$

A Maxon DC brushed geared motor is used whose rated speed is 7800 rpm and stall torque is 0.58 N-m. Adding at 31:1 reduction gearhead ensures matching of our requirements.

3.4 Electrical Design

It was noticed that energy required for driving motors is fairly large than the other components. Also the time required to replace to battery was being bottle-necked by laptop and motors. Adding to it there was another issue of isolating other sensors from these components to make sure any noise in motors should not hamper the other equipment.

As a result of the above scenario, power supply systems for motors, laptop and sensors are kept separate. A 12V 8Ah Li-Polymer battery for sensors, a 22.2V 8.8Ah battery for motors and a 95 Watt-hr slice battery for laptop is used.

Power requirements for all the electrical components used is analyzed below

Average Power Drawn (W)	Peak (W)	Idle (W)
Driving Straight (60% Duty Cycle)	43.2	0
Integrated Turning	13.2	0
Climbing (15^{o} Incline)	43.2	0
Drivetrain Weighted Average	34.8	0
Hemisphere A100 GPS	2	2
VectorNav VN-100 IMU	0.2	0.2
SICK LMS111 Lidar	25	25
ThinkPad W530 Laptop	80	80
UM-18 Ultrasonic Sensors	3.6	3.6
Allied Vision GT 1290 Camera	10	10
Miscellaneous Electronics	10	10
Total	**137**	**102**

Table 2 : Power Requirements

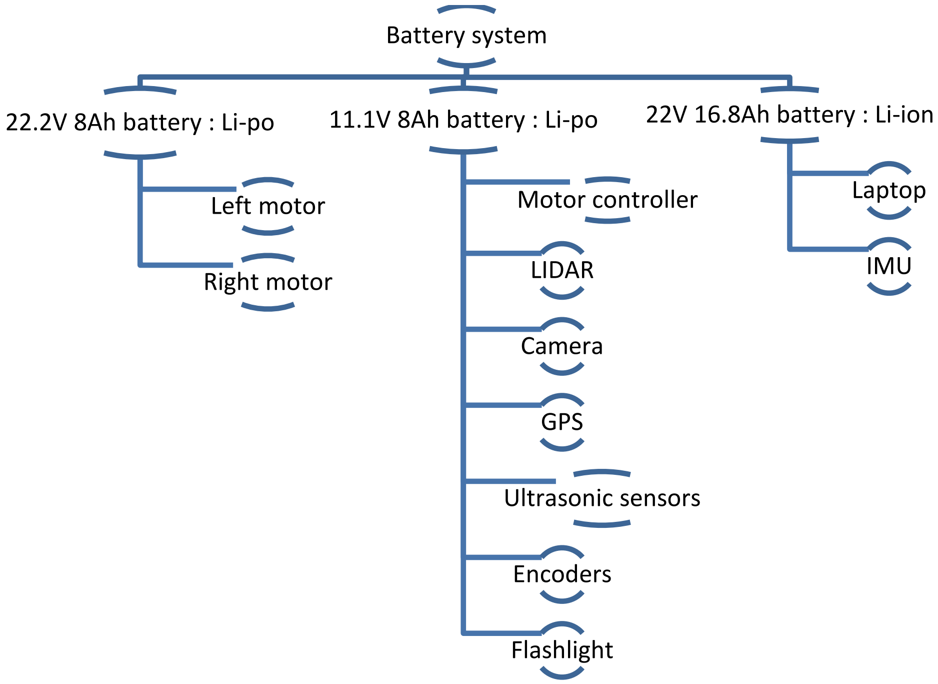


Figure 3-3 : Power system

3.5 Sensor Interfacing and Computation

Pushpak-2 consists of 7 different sensors which are interfaced using different protocol. The previous version (Pushpak) had a Lenovo Thinkpad X220 laptop for computation, but during our algorithm development, it was found out that the amount of data generated by interfacing sensors and then using them in the path planning algorithm created a lag, which resulted in poor performance. Since more computing power was needed, a Lenovo ThinkPad W530 has been used which has a Quad Core i7 processor and 8 GB RAM.

The Hemisphere A100 DGPS antenna is interfaced using a USB to serial Development Module (FT2232). It has an accuracy of 60 cm in its differential mode. A Prosilica GT 1290 camera has been used in machine. A P-iris lens with auto aperture control is attached to the camera to make sure that that light entering the camera does not vary much in changing ambient conditions.

A Sick LMS 111 is fit in the front of the machine to provide a 2-D picture of obstacles in front of the machine. It can measure any obstacle's distance within 270 degree and 20 m of its position and. The LIDAR and camera both use Ethernet port for passing data via TCP/IP protocol. Since the laptop has only one Ethernet port, a PCI to Ethernet Express Card is used. The express card works at 400 Mbps and hence ensures no lag in data acquisition.

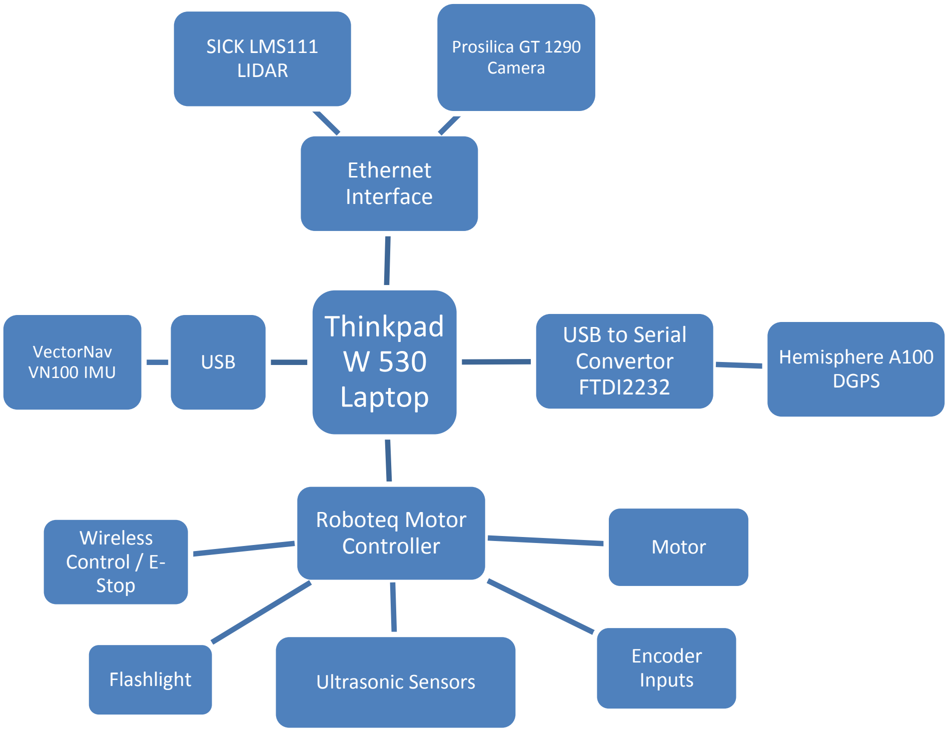


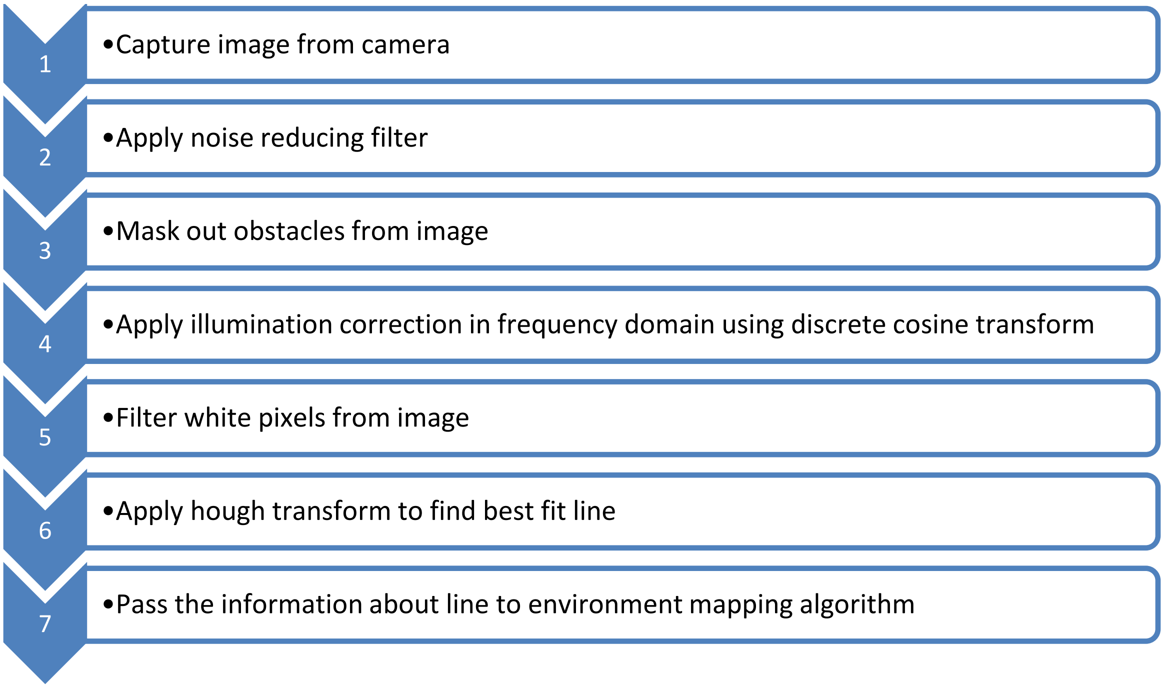
Figure 3-4 : Electrical Hardware Network

The IMU is powered by USB port and it also uses the same Port to transfer the data. It can provide the accelerometer, gyroscope and compass reading along all 3 axes. The motors are controlled by ROBOTEQ HDC 2450 Motor Controller. The device comes in with an inbuilt 24 bit microcontroller which has a number of analog/digital inputs and digital outputs. It also has the facility of directly connecting the motor encoders as well as the RC remote inputs to the microcontroller. Hence ultrasonic sensors and the flash light are interfaced using the motor controller. The Roboteq controller can be connected to laptop using USB and programmed using its proprietary software or serial port commands. This very feature eliminated the requirement of adding an external microcontroller.

4 SOFTWARE DESIGN

The complete software for the vehicle has been made using LabVIEW on Windows 7 operating system. Image processing algorithm is written in C/C++ using OpenCV library and the corresponding dynamic link library (dll) files are generated. These dll files are later used in main LabVIEW project, thus integrating the whole code in one module. LabVIEW also provides separate environment named Robotics Module which is used to simulate the motion of vehicle. JAUS code is also written in C++ and dll files are linked with LabVIEW.

4.1 Image Processing



1 •Capture image from camera

2 •Apply noise reducing filter

3 •Mask out obstacles from image

4 •Apply illumination correction in frequency domain using discrete cosine transform

5 •Filter white pixels from image

6 •Apply hough transform to find best fit line

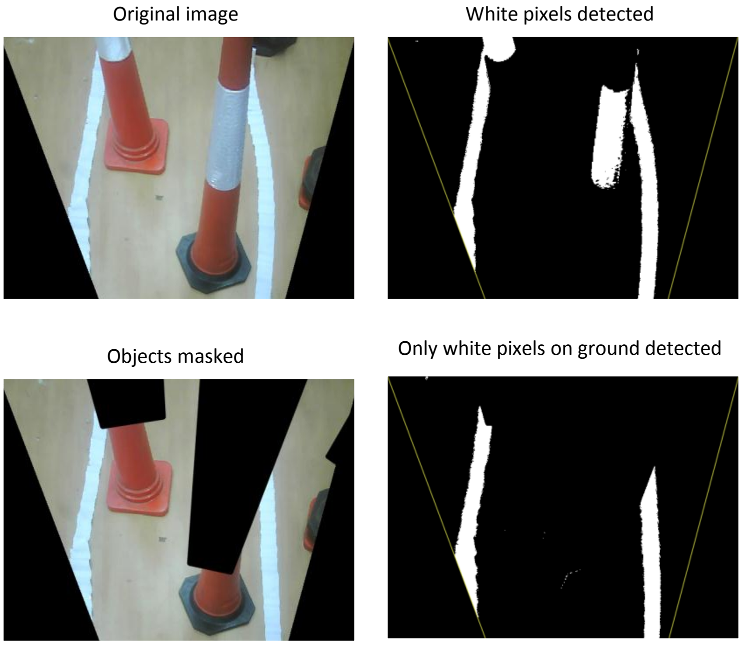
7 •Pass the information about line to environment mapping algorithm

Figure 4-1 : Image Processing Algorithm

Image processing is used to detect white lane marking on the ground. The camera in front of vehicle sends images to the laptop. Images are then processed to extract information about white lines in front of vehicle. This information is further used to construct local map around vehicle.

4.1.1 Object masking

The image is solely used for detection of white lines, but presence of white obstacles made the task difficult. So the method of masking of obstacles was carried on from last year's code. Data from LIDAR is used to check each white pixel from image. If that pixel is part of an obstacle, it is no longer considered for fitting line in Hough transform.



Original image

White pixels detected

Objects masked

Only white pixels on ground detected

Figure 4-2 : Object masking

4.1.2 Illumination correction

One of the innovations is usage of discrete cosine transform for illumination correction. Illumination correction was required to reduce the effect of sunlight being reflected from grass and other surfaces. The image is divided in several blocks and discrete cosine transform is applied on each block. In frequency domain, each block's average saturation value is pulled down to complete image's average saturation value. Next, each block is reconstructed and merged to make final image.

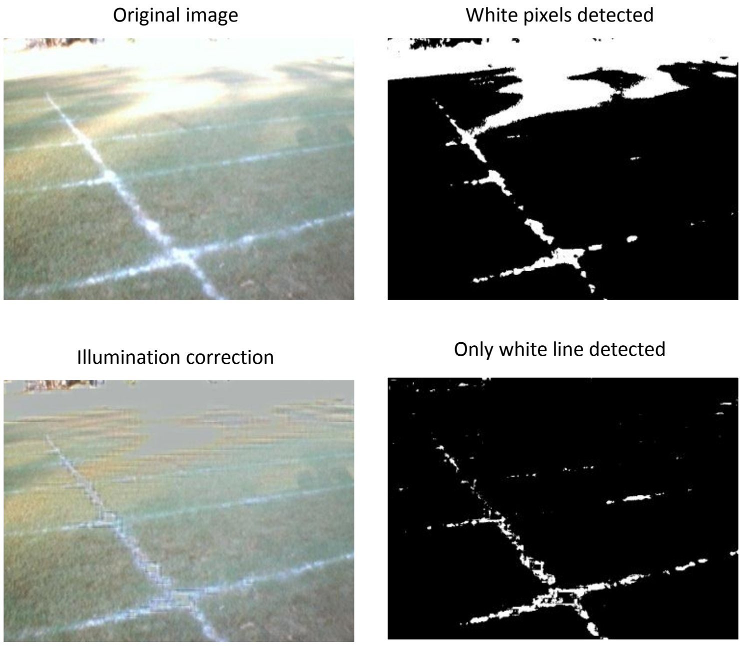


Figure 4-3 : Illumination correction

4.2 Environment Mapping

The LIDAR data is used to create a polar map of 0.5 degree resolution of its immediate surroundings. The image processing module sends information about white lines, which is also used to create a map of same kind. Both the maps are merged to create a common map. The span of this common map is stretched from 180 degrees to 270 degrees using ultrasonic sensors.

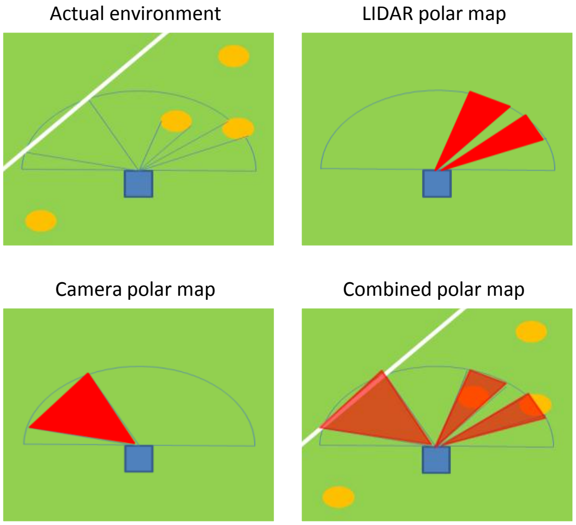


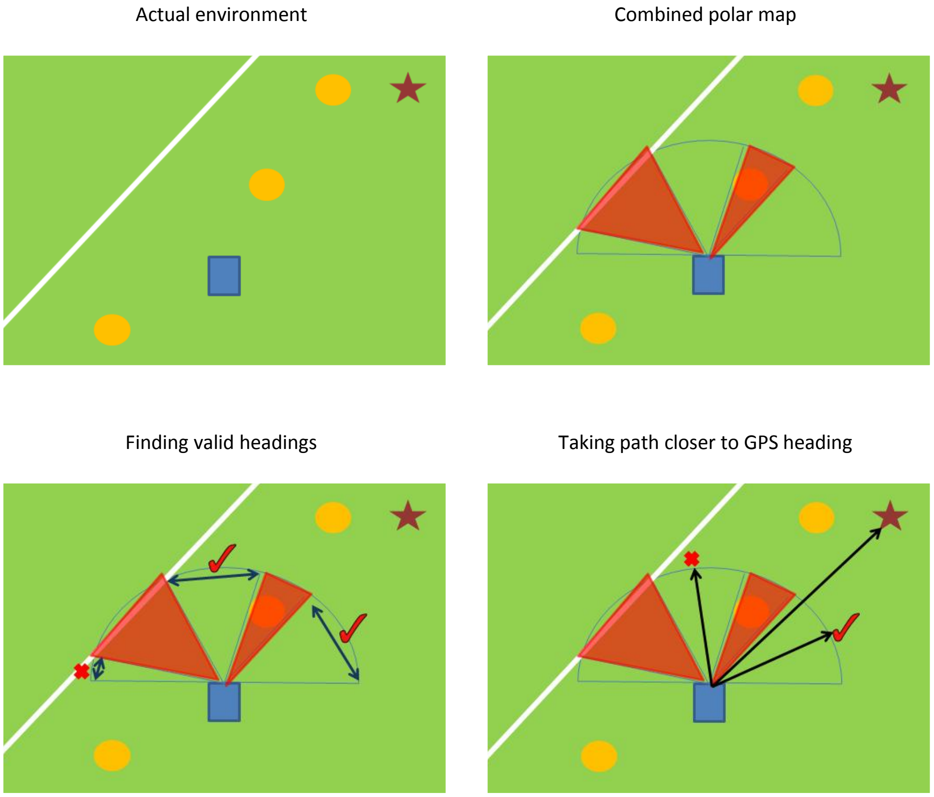
Figure 4-4 : Environment Mapping

In the illustration above; green represents the surface of lawn, orange represents solid obstacles, white represents lane markings and blue represents the vehicle.

4.3 Navigation Algorithm

The vehicle is given a target GPS waypoint. All the decisions taken in navigation algorithm ultimately aim to drive the vehicle towards this GPS point. Using the current GPS coordinates, absolute heading is calculated where the vehicle should move.

The local map obtained in previous part is digitized here. A threshold is set and all angles with distance lesser than this threshold qualify as obstacles (assigned value 0), while others qualify as open path (assigned vale 1). Thus sectors of 1 and 0 are obtained. Each sector is analyzed and if its width is greater than vehicle's width, it qualifies as valid path. This gives an array of valid paths that the vehicle can take safely. The vehicle is asked to follow the valid path closest to absolute heading obtained previously. The algorithm also stores all GPS points the vehicle has touched so far. Therefore, in case of being faced by a dead end, the vehicle retraces its trajectory until it finds out a new path to reach the goal.



Actual environment

Combined polar map

Finding valid headings

Taking path closer to GPS heading

Figure 4-5 : Navigation Algorithm

In the illustration above; red areas are invalid sectors while star represents target GPS point.

4.4 Feedback Control

The navigation module gives an absolute heading on which the vehicle is supposed to move. The vehicle has a feedback control based on digital compass and inertial measurement unit which helps the vehicle achieve the heading very quickly. The vehicle also has individual feedback control on both driving motors based on rotary encoders. This control loop ensures both the wheels are moving with the desired angular velocity.

An average wheel rpm has already been set, and the G_{c1} controller gives Δrpm which will be added to average wheel rpm for left wheel and subtracted from average wheel rpm for right wheel. Thus both wheels have a difference of 2*Δrpm between them which makes the vehicle rotate by angular velocity $\boldsymbol{\omega}$ given by

$$\omega = \frac{\mathrm{d}\Theta}{\mathrm{dt}} = \frac{R}{l} * 2 * \Delta rpm$$

R = Radius of wheel

l = distance between center of two wheels

To obtain closed loop parameters (K_p, K_d and K_i) of G_c, step change response of control variable in open loop was first obtained. The closed loop parameters were decided based on a set threshold in which the desired response should be obtained.

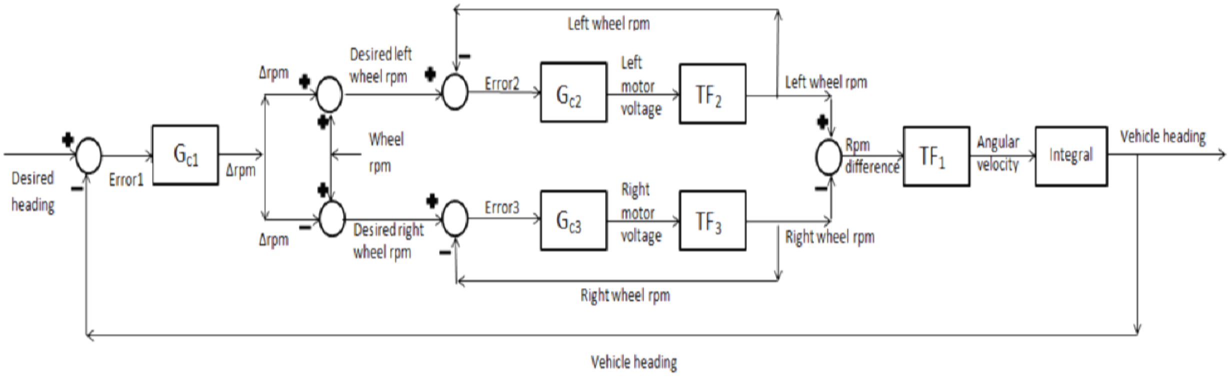


Figure 4-6 : Feedback Control

Controller transfer function

G_{c1} – for vehicle heading
G_{c2} – for left wheel rpm
G_{c3} – for right wheel rpm

Open loop transfer function

TF_1 – for vehicle's rotational velocity
TF_2 – for left wheel rpm
TF_3 – for right wheel rpm

4.5 JAUS Challenge

JAUS functionality has been implemented in C++ using OpenJAUS 4.0 SDK which uses SAE AS5669A transport specifications. The SDK provides classes for services specified in AS6009 Mobility Service Set. The complete code is compiled into a dynamic linked library (dll) which is then used by central software in LabVIEW. Following functions can be called from LabVIEW –

- ***init*** - This function starts the JAUS component
- ***read_message*** - Messages received are stored in a stack of strings. This function returns the last received message. Returns a default value if no pending messages.
- ***update_local_pose*** - This function updates the vehicle position and orientation which is required by local pose sensor service. When any Query message like QueryLocalPose is received, the response will contain data received from the last call of this function. Thus, this function has to be called in a loop to keep the position, velocity data up to date.
- ***update_velocity_state*** - Similar to update_local_pose except that this function updates the information required for the velocity state sensor service.
- ***update_waypoint_status*** - Similar to update_local_pose except that this function updates the information required for the waypoint list driver service.

For simulating the competition environment, the vehicle's laptop and an external laptop are connected through a wireless router. A COP program runs on the external computer. It sends JAUS messages as specified in the JAUS challenge to the machine laptop and the response from vehicle is verified.

5 Simulation Environment

To reduce the efforts involved in testing the algorithm on actual vehicle in open environment, virtual simulation was tried. LabVIEW provides quite unique feature of testing the code in Robotics Environment Simulator. The 3D model of vehicle was imported in simulator and the sensor modules attached to the model. Environment was also built very much similar to IGVC terrain. The complete algorithm was easily implemented without any need of rewriting code in other language. This proved out to be very useful, given the fact that Mumbai receives heavy rain for approx. 4 months in a year.

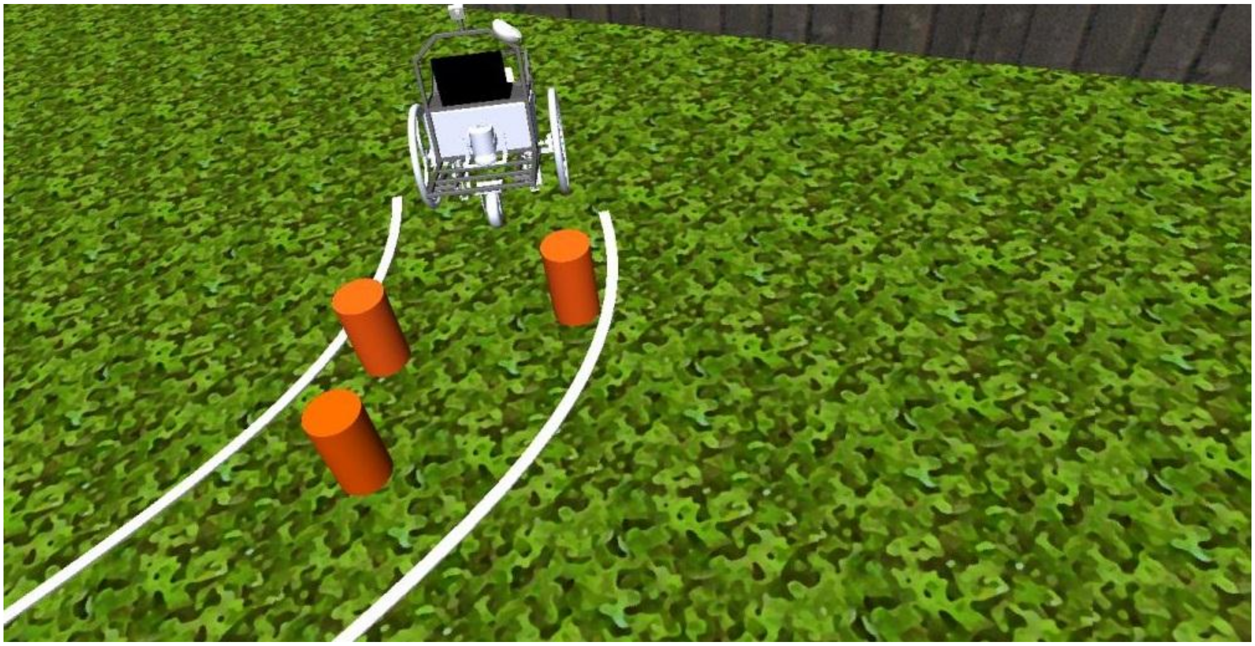


Figure 5-1 : LabVIEW Robotics Simulation Environment

6 CONCLUSION

Pushpak-2 performed magnificently during test runs and we expect the same in the main competition also. The robustness in both mechanical and software design has improved the reliability of Pushpak-2 over its earlier versions. We hope that this report clearly illustrates the innovations we have made and help readers develop better understanding of our work.

7 ACKNOWLEDGEMENT

We would like to thank IIT Bombay for being our chief patron and providing us major funding for vehicle construction, travel and logistics. We are thankful to our industry sponsors for providing us financial and technical support pertaining to usage of the sensors in the most efficient way. Apart from our faculty advisors , we would like to acknowledge support of Prof. Leena Vachhani, Prof. Parul Shah and Prof Sharat Chandran. We would also like to acknowledge never ending support from following alumni of IIT Bombay - Mr. Ashok Kalbag, Mr. Atul Gupta, Mr. Dhrumil Gandhi and Mr. Girish Desai.