INTELLIGENT GROUND VEHICLE COMPETITION 2014

Indian Institute of Technology, Madras, India

TEAM ABHIYAN

Faculty Advisor Statement:

I hereby certify that the development of the vehicle, ABHIYAN 1.0, described in this report has been significant. This report has been prepared by the students under my guidance.

Nitin Cha

Dr. *Nitin Chandrachoodan*
Associate Professor, Department of Electrical Engineering, IIT Madras

Dr. Nitin Chandrachoodan
Associate Professor
Department of Electrical Engineeri
Indian Institute of Technology Madı
Chennai - 600 036. India

Team Members : Sripriya Kalidoss , Ashish Sharma , Vinay Kale , Suraj Kashyap , Surya Pavan , Ekansh Verma , Arvind N , Aditi Singh , Ahmed Ansari , Vishwa Sai Prathyusha , Gagan Khandate , Manudeep Chowdary , Akhil LB , Saathvik DPKS

CΦ Centre For Innovation
IIT Madras

CONTENTS

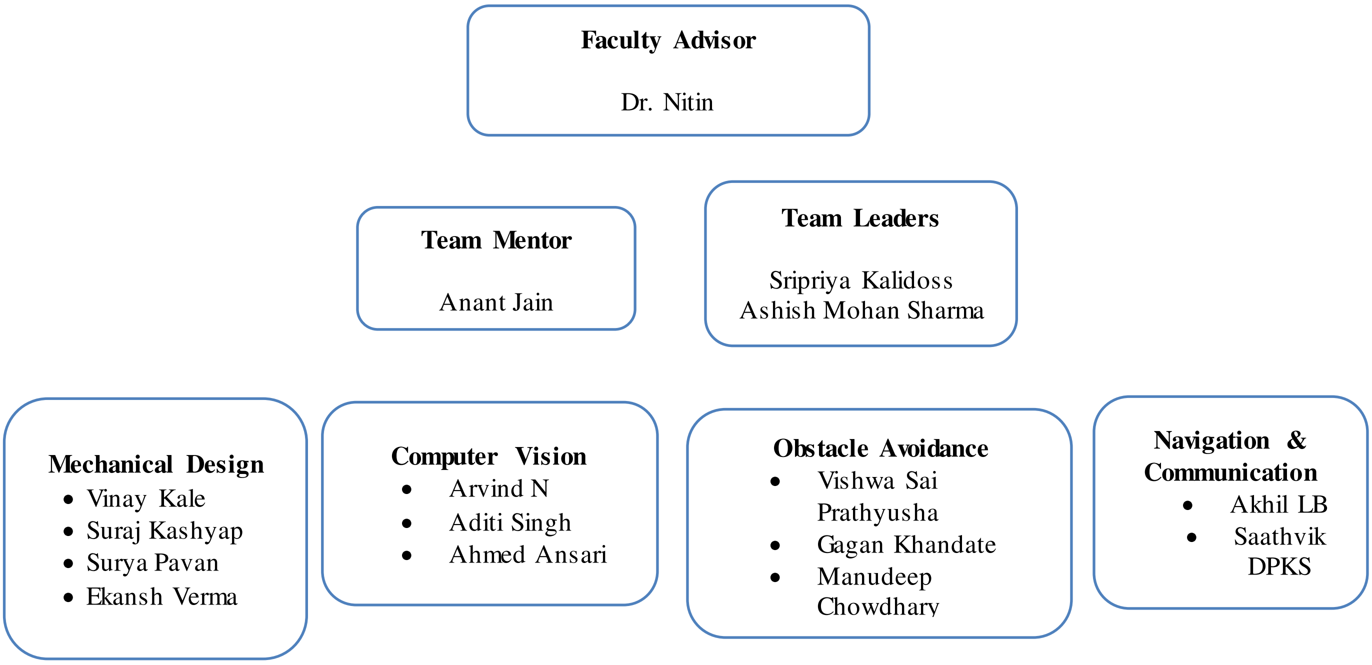
.

.

1. Team Structure

The team is being incubated in the **Centre for Innovation (CFI)** and is being supported by funds from the Alumni of IIT Madras. The project has taken a formal structure under the mentorship of Anant Jain, Member of DRDO student design project and Ashish Bajaj, Student head of the **Centre for Innovation**.

The team consists of junior undergraduates belonging to various academic backgrounds viz., Computer Science, Electrical and Mechanical Engineering. With this diversity, every member brings to the table different skills and opinions, which is vital considering the multi-disciplinary nature of the project.



Faculty Advisor

Dr. Nitin

Team Mentor

Anant Jain

Team Leaders

Sripriya Kalidoss
Ashish Mohan Sharma

Mechanical Design

- Vinay Kale
- Suraj Kashyap
- Surya Pavan
- Ekansh Verma

Computer Vision

- Arvind N
- Aditi Singh
- Ahmed Ansari

Obstacle Avoidance

- Vishwa Sai Prathyusha
- Gagan Khandate
- Manudeep Chowdhary

Navigation & Communication

- Akhil LB
- Saathvik DPKS

The task in hand is divided into five modules:

- Mechanical Design
- Computer Vision and Image Processing
- Obstacle Avoidance
- Communication
- Navigation System

2. Mechanical Module

Introduction

The weight and dimensions of the payload were the primal considerations during the course of the mechanical design. Being the heaviest part of the vehicle it was ensured low placement and CG as near to centre of vehicle as possible. The next step consisted of ensuring enough normal forces on suspensions to ensure sensors are free from vibrations by giving proper travel length for suspensions. The next prototype was made to ensure that wheels are always upright giving minimum camber possible to ensure high friction to move on wet surfaces. Also different suspensions were used for both wheels so that they are free to move on their own and can handle undulating terrains better while ensuring high ground contact to avoid vehicle getting stuck. For power transmission, chain transmission was used so that there are no axial forces on the motor and wheels are ensured same power for all travel distances of suspension.

Design Methodology

1. Drivetrain:

The drivetrain has been designed to be simple and effective. The motors, being one of the heaviest components on the vehicle, have been added to the sprung mass, and a chain-sprocket configuration has been implemented for transmission to the wheels.

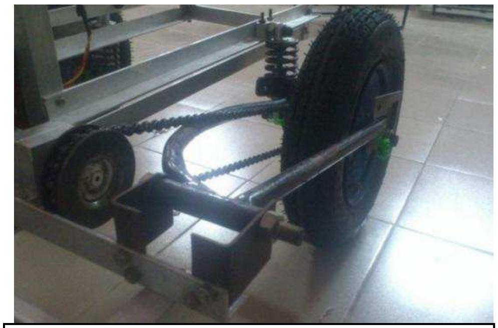


Figure 1: Drive train

a. Chain drive:

Out of the various options for the drive train, the chain drive mechanism was finalized. This was done so that, lower weight on the un- sprung mass would impart greater stability to the suspended system. Thus the motors are placed on the frame, and the transmission is done through a chain-sprocket assembly.
In the first prototype, it was observed that, due to slight slackness in the chain, it would come off the sprocket. To avoid slipping of the chain, the sprocket was modified by attaching circular MS plates on either sides of the sprocket. The plates used have 3 mm thickness which have been gas cut and welded on the sprocket by spot welds.

b. Tensioner:

To address the problem of chain slackness, a tensioner was designed and implemented in prototype -2. Although the idler served the problem of tensioning the chain, the design had other challenges. One of the major challenges was the functioning of the tensioner; the coil spring could provide force only in one direction, which meant that the drivetrain used to be slack when the vehicle would run in reverse. Another major reason was that the torsion spring that was being used, had to be pre-tensioned before mounting. This reduced the ease of assembly.

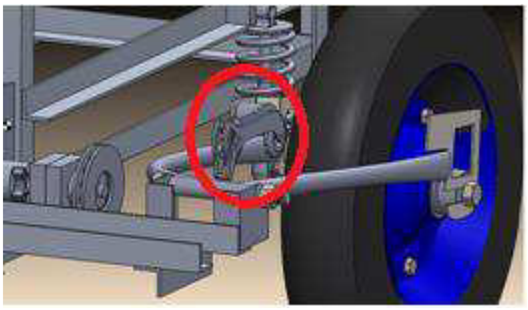


Figure 2: Indicating the tensioner used in prototype 2

2. Suspension: The precision sensors mounted on the vehicle demand a noise-free environment, to deliver the best results. To meet this requirement we have come up with a custom designed suspension system to counter the undulations in the terrain. The design has been analyzed to guarantee a smooth ride under anticipated ground conditions. The resulting forces are uniformly distributed to avoid any undue stresses on the chassis.

3. Steering: Being a completely autonomous vehicle, no manual input is allowed once the vehicle is on track. The differential drive was chosen as the steering methodology because of its simplicity and efficacy for our requirements. The vehicle has two powered wheels and one castor. The wheels are powered by one motor each. The RPM of the motors is controlled independently by the microcontroller. By varying the RPM of the motors the steering angle and the direction of heading of the vehicle can be changed. Encoders are being used for precise motor control to control the radius of turning as well.

4 Castor Positioning: The position of the castor plays a vital role in the dynamic orientation of the vehicle. Keeping the castor in the front would make the robot sway in the direction of the castor, which is theoretically controlled by the motion of the powered wheels. But after extensive testing it was observed that on rough surfaces the castor generates its own direction which makes the vehicle deviate from the desired path. Hence it was decided to use 2 front powered wheels with a rear castor.

Design process

1 CAD Model: We have used SOLIDWORKS by Dassault Systems for CAD modelling. It helped to reduce physical design iterations which saved us a lot of time. We also performed stress analysis on the design to avoid material failures.

2. Payload: The vehicle is required to run with a payload of 10 kg weight. The design and component distribution has largely been influenced by the presence of the payload. A low and centralized centre-of-gravity has been maintained to provide maximum dynamic stability to the vehicle.

3 Vehicle Configuration: After deep deliberation and analysis, it has been determined that two wheeled front powered vehicle with a single rear castor is required for locomotion, providing greater stability while turning. This weight distribution of the components and low speeds ensure ground contact and enough normal forces on suspensions at all times.

4 Torque Requirements:

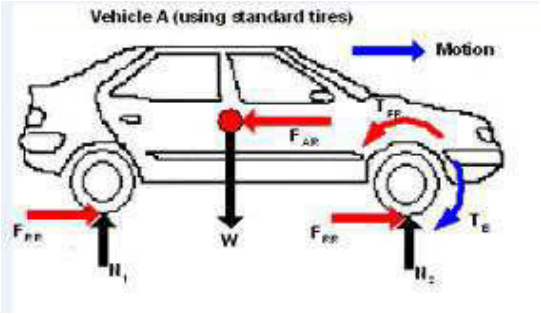
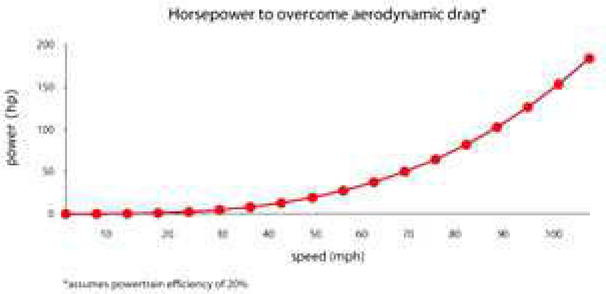


Figure 3: Free body diagram indicating the forces acting on the vehicle

The major forces acting on the vehicle are-

- Aerodynamic force: These are the drag forces that act on vehicle due to wind resistance. The forces are directly proportional to the square of velocity of the vehicle.

  –



*assumes powertrain efficiency of 20%

Figure 4: Plot of aerodynamic drag power vs speed.

From the plot and the equation it can be deduced that the air drag for our requirement is very low and can be neglected.

- Rolling Resistance: The rolling resistance is the friction that the vehicle experiences due to hysteresis in the runner tires. The Rolling resistance is a constant value and the computed by

$$R_r = \mu_r \times N$$

$$\tau = r \times R_r$$

  - R_r = Rolling Resistance on Rear wheels
  - μ_r = Coefficient of rollong friction between rubber and ground
  - N = Normal
  - *r = Radius of the driving wheel*

Variable	Value
μ_r	0.35
N	180 N
R_r	24.5 N
r	203 mm

5. Motor specifications:

To achieve the required average speed of 1 mph (1.6 kmph), motor specifications were calculated based on a maximum speed of 3.9 kmph and an average speed of 2.4 kmph.

No load voltage = 24 V

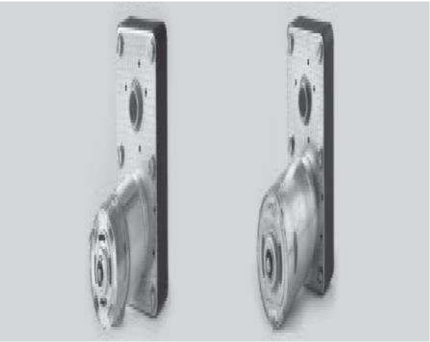


Figure 5 : Pic Source: Mechtex India

Gear ratio= 40:1
No load RPM- 125
Stall torque-23.96 Nm
Rated speed (for 3.9 kmph) = 50
Rated torque =13.95 Nm
Stall current = 14.3 A
Efficiency = 65%

6. Detailed design of the compression spring

The springs are designed in such a manner that after installation between the chassis and the wheel, the springs compresses by 20 mm because of the preload force due to the weight of the vehicle. The attachments have also been designed such that the chassis remains horizontal when the springs are compressed with the preload.

Force on each spring	70N
Length of the spring under the applied load	11cm
Compressed length of the spring, under initial pre-load	2cm
Spring stiffness	3500 N/m
wire diameter (d)	6.5 mm
coil diameter (D)	54 mm
correction factor	1.06
Shear stress	79.396539 MPa
Ultimate tensile strength	2432.5 MPa
Factor of safety	30.5
N (no. of turns)	8
Natural length of spring	11cm

Table 1: Suspension spring parameters

Fabrication and Assembly

AL 6061 material L-channels have been used to build the chassis. All L-channels are connected to each other by reliable fasteners which make it easy to assemble. The motors were secured to chassis using vertical clamps to reduce the vibrations and proper functioning of transmission.
The rear wheel assembly is innovatively designed to operate stronger wheels using chain transmission. The wheel is attached to center plate using 5 equidistant bolts to ensure centering and alignment.

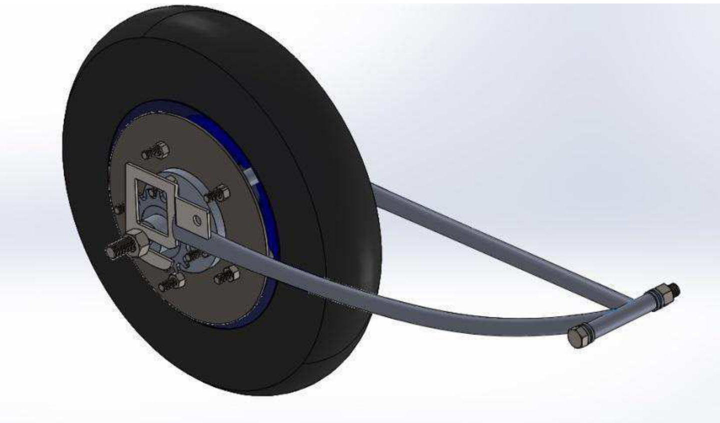


Figure 6: CAD model of the rear wheel assembly.

Custom-designed Steel connecting rods have been used to clamp wheels from both sides which reduce possible camber and toe in-toe out problems. Parts like bearing-casing, couplers, intermediate shafts, rear wheels assembly were manufactured using lathe and milling machines.

.

Design Optimization:

Based on the testing carried out on prototype 1 and prototype 2, several modifications have been made to reduce the number of dimensions, and components of the vehicle, and to facilitate modularity.

Reduction in number of parts:

- The design had a significant decrease in the number of components between prototype 1 and 2. This was majorly in the drive train subsystem. One of the major changes made was to reduce the misalignment in the drive train due to a number of co-axial components used. The connecting rods (two independent rods) were pivoted at the shaft using ball bearings. The assembly also had two bearings and casings to attach them. This assembly as a whole was coupled to the motor shaft using a rigid coupling. Due to the number parts that had to be aligned and the intrinsic eccentricity of the shaft, the motor shaft was experiencing cyclic loading. Also, the clearance between the bearing and the shaft was giving rise to eccentricity.
- To eliminate the problem, all bearings in the assembly were replaced by sleeves with solid lubricant. The sleeve has the advantage of providing a revolute joint and at the same time occupying much lesser space.

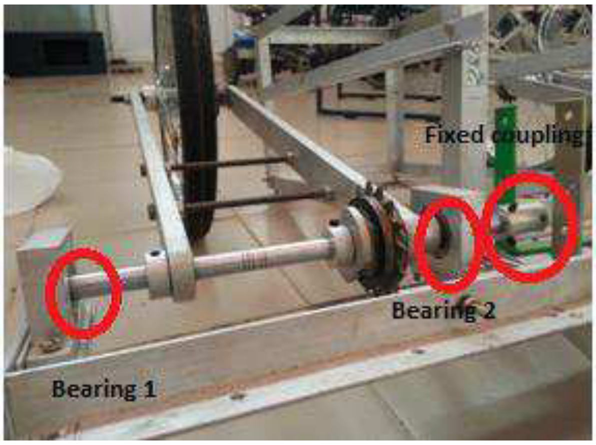


Figure 7: Rear wheel assembly. Clearly the shaft used is very long. The numbers of components aligned co-axially are many.

Reduction in dimension of the chassis:

- The width of the chassis was deliberately reduced. Since the vehicle uses only one rear castor hence reduction of the chassis dimension was vital, to increase stability and prevent any toppling due to cornering forces. The width of chassis was reduced by 110 mm between the first and second prototype.

Modularity of the system:

- The vehicle has been made in a fashion that it is highly modular. The entire vehicle can be dismantled into separate units, the drive trains and the chassis. These are attached only using fasteners and thus can be easily disassembled.

Power systems

Since we are working with different types of sensors of varying specifications, we require to carefully design the power systems of the UGV. Among all the various sensors/components, the LiDAR and the motors consume the maximum power.

LiDAR: 24V/1A rating

Motors (DC52a): No load current=500mA
Stall current=14300mA
Stall torque= 5984g-cm
Operating torque= 2750g-cm.

Hence, we project a requirement of 15A at 24V. We considered the following:

-Lead-Acid batteries

Low cost, easily available but maximum current is very low & also, if used, will drain out in around 5 to 10 minutes. Also, they are really heavy.

-Li-Po batteries

It was finally decided to use ZIPPY Flightmax 8000 mAh 3S1P 30C (Total cost=17555INR) with a continuous runtime of 38 to 45 minutes.

3. Obstacle Avoidance

One of the main objective of the this ground vehicle is to perceive the environment, and negotiate outdoor obstacles. Obstacle avoidance is, hence, the most fundamental and essential part of the vehicle's navigation system.

The crucial component for this is the sensor, which is chosen to be a LiDAR (Light Detection and Ranging Device). LiDAR is a laser based ranging device that scans the surroundings in 2D plane in front of it to detect objects and evaluate their positions. It, basically, gives a polar profile through which the distance of the nearest obstacle at any angular position can be measured.

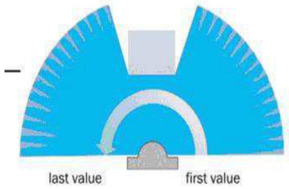


Figure 8: A sample LIDAR scan

The initial algorithm that was planned was based on Vector Field Histogram Method. The simulations were performed using various software (MobileSim of ARIA), the results for which have been summarized in the following section.

Moving on to an advanced stage, it has been decided to use ROS (Robotic Operating System) and the in-built algorithms available with its packages for the navigation of the robot. The current control and operating system in Abhiyan robot is based majorly on ROS.

Vector Field Histogram method:

In the vector field histogram method, a polar histogram of the vehicle's environment is created using a suitable sensor. A polar histogram is the plot of Max. Range [Distance of obstacle] vs. Scan Angle as shown below.

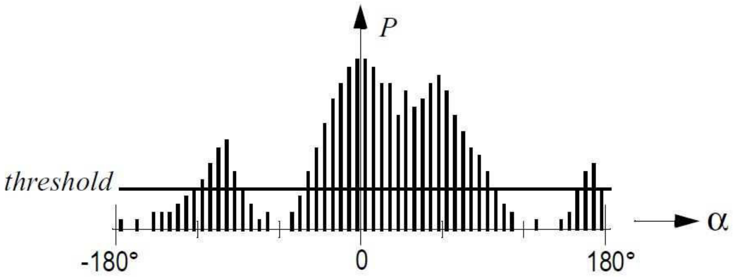
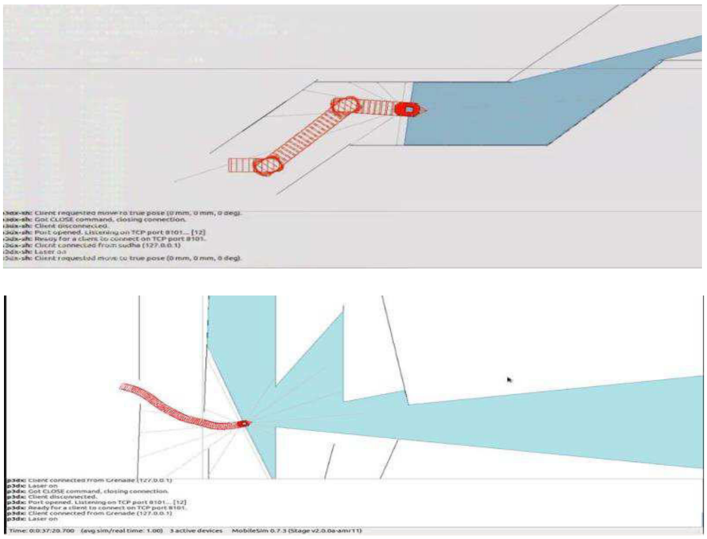


Figure 9 Graph

The algorithm used here is a basic one. The figure above shows the data obtained in a LaserScan. The whole region is segmented into different regions called as valleys and peaks. Valleys represent no obstacle region and peaks represent the region in which obstacle is present. Hence, at any instant the algorithm looks for all the valleys and chooses a valley closest to the goal position with a constraint that the valley width is greater than the robots width. MobileSim is a software on ARIA (Advanced Robotics Interface for Applications) platform of mobile robots that is being used for simulating the basic algorithm.



```
3dx-sh: Client requested move to true pose (0 mm, 0 mm, 0 deg).
3dx-sh: Got CLOSE command, closing connection.
3dx-sh: Client disconnected.
3dx-sh: Port opened. Listening on TCP port 8101... [12]
3dx-sh: Ready for a client to connect on TCP port 8101.
3dx-sh: Client connected from sudha (127.0.0.1)
3dx-sh: Laser on
3dx-sh: Client requested move to true pose (0 mm, 0 mm, 0 deg).
```

```
p3dx: Client connected from Grenade (127.0.0.1)
p3dx: Laser on
p3dx: Got CLOSE command, closing connection.
p3dx: Client disconnected.
p3dx: Port opened. Listening on TCP port 8101... [12]
p3dx: Ready for a client to connect on TCP port 8101
p3dx: Client connected from Grenade (127.0.0.1)
p3dx: Laser on
Time: 0:0:37:20.700   (avg sim/real time: 1.00)   3 active devices   MobileSim 0.7.3 (Stage v2.0.0a-amr11)
```

Figure 10 Simulation on MobileSim software

Limitations -

* The present mapping software cannot be used for complicated environments as required for testing

* The motion control and other functions are all in-built, which cannot be customized as per our requirements

* Direct integration with other modules and ROS is not possible.

Current Control and Operating System

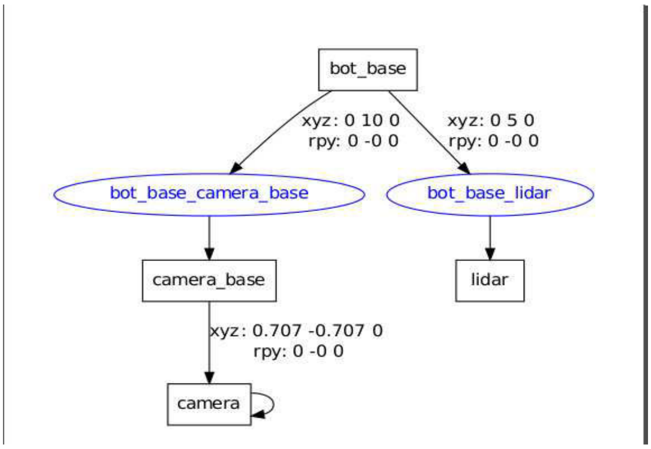
ROS is an operating system with a set of libraries and software for various robotic applications. It also covers various algorithms for many programs, provides visualization software for simulations, device drivers for various sensors, and also unique opportunity of synchronization with various applications at a time. That allows for the better and easier ways of integration of different modules.

In such huge set of packages available, the package called 'Navigation Stack' is being used for this robot. Navigation stack takes in input data from all the sensors attached to the robot, processes them in different appropriate nodes simultaneously or sequentially, and gives the values of the velocities to the robot to follow as an output.

Working of Navigation Stack

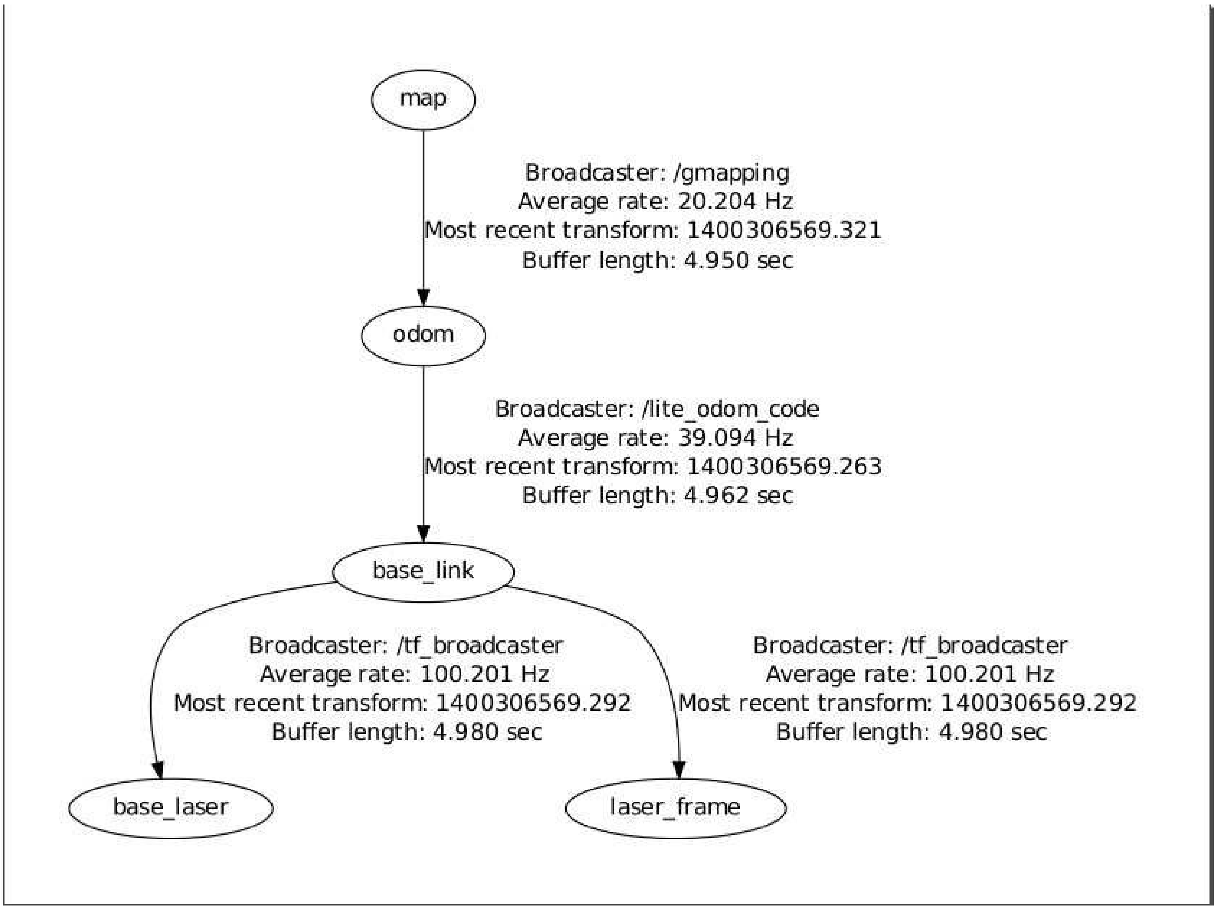
Input:

Robot Configuration - This includes transform configuration and robot setup, wherein the physical shape, size and dimensions of the robot are set up. With this information about the robot, all the frames available on are identified. This transform considers all the coordinate frames and uses this transform to interpret data from various sensors appropriately.



Robot configuration

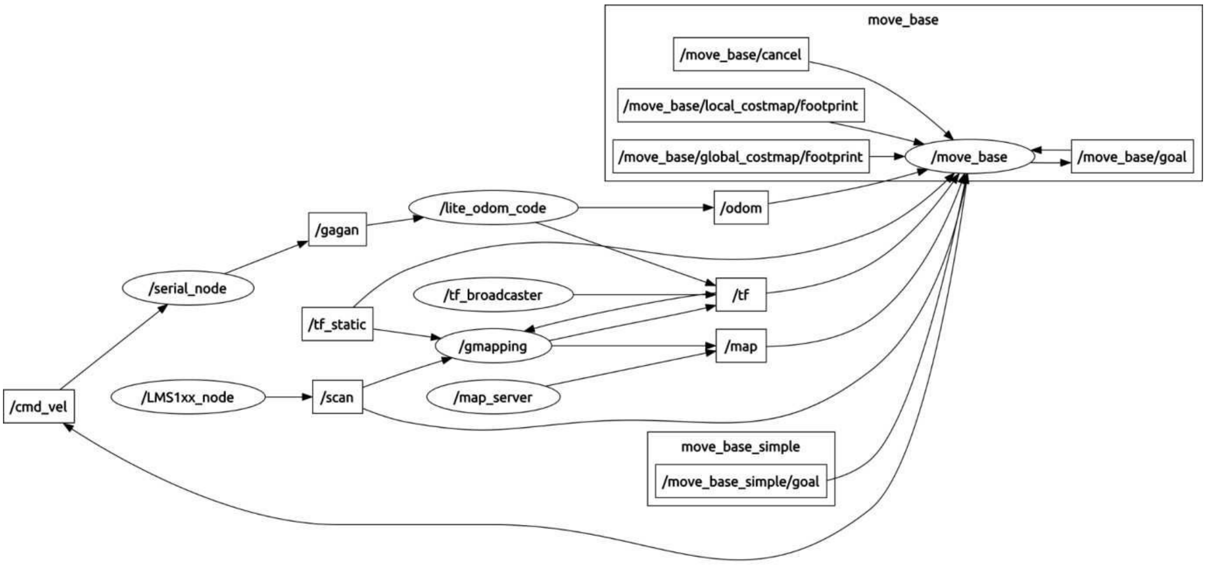
<u>Sensor</u> <u>Information</u> - Data from different sensors is obtained and taken in form of messages. There are a set of kinds of messages in which the data can be published and subscribed by different nodes whenever required.



A view of the frames of the robot, where LiDAR (base_laser), Camera (laser_frame) are attached to the main robot base (base_link) receiving odometry readings (odom). Also shows the parameters in which the sensor data is being received.

Process and Outputs:

The navigation stack with all the parameters in the .yaml files for robot configuration, with all the inputs from different sensors published and subscribed by different nodes, maps are generated with specified goals to reach. Then, by using SLAM(Simultaneous Localization and Mapping) in gmapping package on a dynamic map or by using AMCL(Adaptive Monte Carlo Localization), the pose of the robot is traced and it is localized. After the localization of the robot, according its position with respect to the target position, the robot is given velocity commands (/cmd_vel) to reach the goal.



Network of the topics published and subscribed as various topics between different nodes.

4. Image processing

A major challenge in the problem statement is the detection of lanes in the arena and navigating the vehicle accordingly. The goal of the Image Processing Module was to effectively isolate the lanes from the grassy arena and to feed the same to the path planner.

The basic algorithm adopted is as follows:

1. Capture image of the arena using a camera.
2. Compute the orthographic view of the image.
3. Convert the image to HSV color space and extract the white lines.
4. Transfer the lanes to the ROS Gmapping package for dynamic path planning.

A Logitech C310 camera (640X480) was used to capture the view of the arena. The camera allows adjustment of shutter speed and aperture size to capture the best possible image, even in full sunlight.

The orthographic view of the arena is computed using the formula:

$$A = HB$$

Where A is the orthographic/final computed image
B is the initial image captured by the camera
H is the homography matrix

The Homography matrix is calculated by training the camera for various heights and angles and optimizing the errors obtained. After a rigorous testing phase, it was finally decided to settle for a camera height of 90cm, with the camera facing horizontally ahead. A blind spot of 240cm (approx.) was observed in front of the vehicle. (The camera we are using has 60 degree field of view, so the blindspot should be of appox. 156cm, has this value of 240cm been calculated or practically tested)

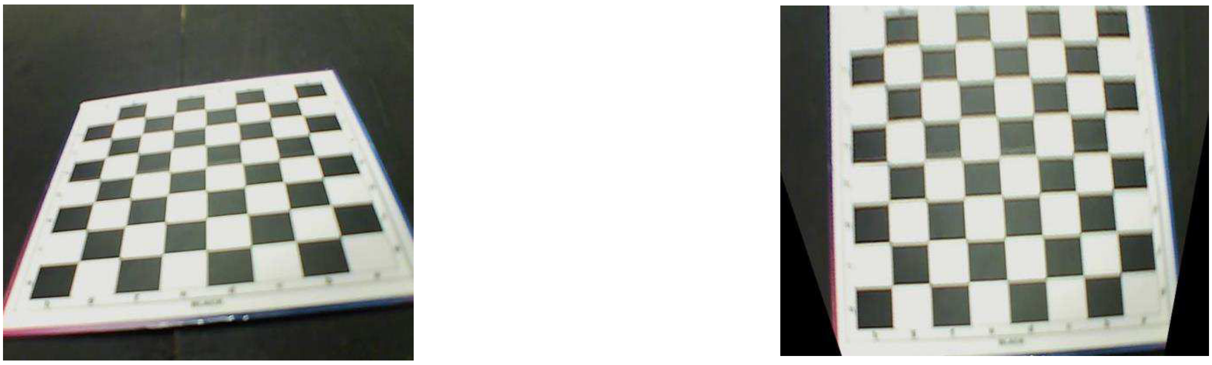


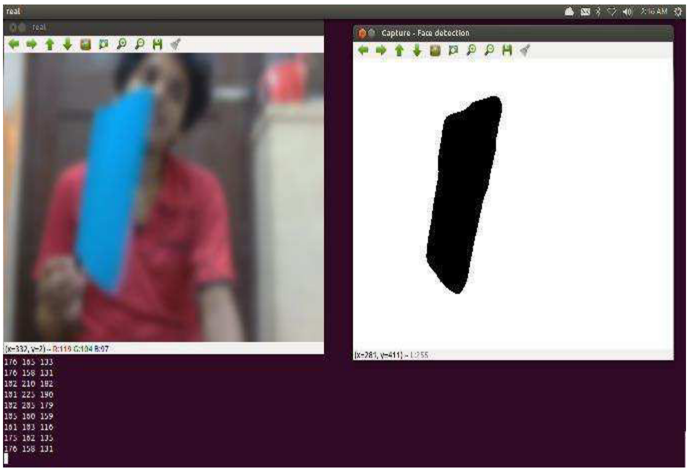
Figure 11 A sample perspective image and its orthographic view

The algorithm for extracting the white lanes from the grass arena includes converting the color space to HSV for better color segmentation. The obtained image is then processed by the Canny Edge detection algorithm before running it through the Probabilistic Hough Transform. The voting procedure is set up to maximize the number of lines of a minimum length that can be generated in the image.

ROS Work:

The lanes, obtained above, are converted into the form of LaserScan data. This conversion is needed because the ROS gmapping package can accept data inputs only in the form of LaserScan data or Point Cloud data to generate maps. After converting to LaserScan data, the CV images are converted to ROS image messages using cv_bridge package, so that the images can be published, and accessed by the ROS gmapping package, which visualizes the lanes as walls/obstacles to be avoided. This data is fed into the navigation stack, along with the LiDAR, GPS and odometry data for path planning and autonomous navigation.

The second part of image processing subsystem is flag detection. The advanced course consists of flags placed at various positions in the course and the vehicle is expected to detect them and it should stay to the left of Red flags (Grainger 3LUK2) and right of Blue flags (Grainger 3LUK4). The approach here is to use color segmentation and filters to obtain the profile of the flag. The stem height is then calculated, compared and calibrated accordingly, to differentiate the flag from other red and blue entities.



```
176 165 133
176 158 131
182 210 182
181 225 190
182 285 179
185 180 159
161 183 116
175 162 135
176 158 131
```

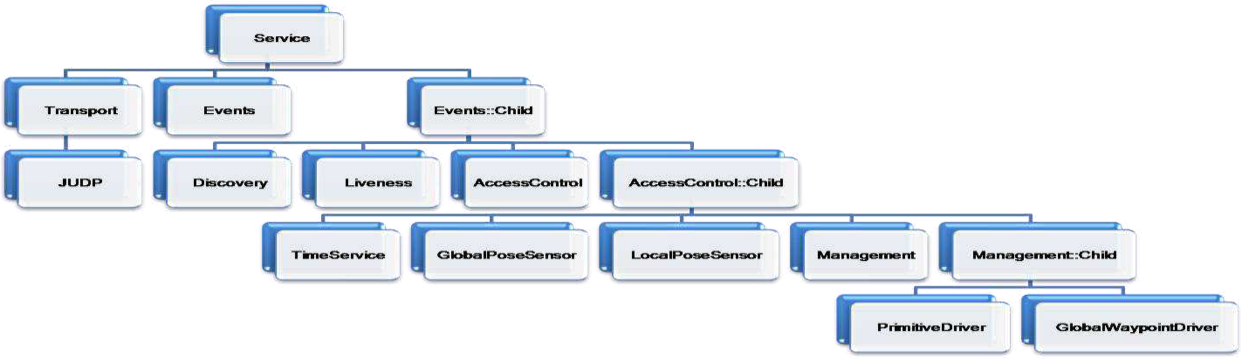
Figure 12 Flag Detection

The aforementioned algorithms face challenges in the form of ambient conditions of the track, processing capabilities of the camera and the computer, lightning conditions (illumination correctness), noise removal in video feed and orientation of the camera and the images obtained.

5. Communication

The JAUS challenge requires us to implement JAUS services on our machine and make it interoperable. We have used the JAUS++ SDK for this purpose. JAUS++ is an Open Source implementation of JAUS in C++. It implements the current versions of the SAE standard. JAUS++ includes complete message libraries for both the Core and Mobility service sets, and interfaces for the most common services. JAUS++ is Object-oriented in design and provides threads in the background to send and receive messages. Every JAUS component has a subsystem ID and a node ID. The assignment of these IDs is static. For transport, JAUS over UDP transport protocol will be used as specified in AS5669A.
The UGV first tries to discover the calling device and tries to establish a connection with that node. It keeps listening to that node for queries. As soon as a query is received, it processes the type of query it is and decides which action to take. If a query for velocity or GPS is made, the last value of the required component published from ROS is read and updated in the JAUS system navigation points and fed back to ROS. JAUS++ provides us an interface to update different components and since it runs threads to receive and send messages in the standards mentioned by SAE, we don’t have to worry about them while coding it. Since a reliable transmission is needed, JAUS is implemented on TCP.

The following diagram shows the class inheritance diagram for services within JAUS++. It shows the Core Services and a few Mobility Services available.



6. Navigation System

Introduction:

A vital system for any ground vehicle to make an accurate decision about its heading and navigate from initial position to final position. Here, we have used global coordinate system i.e Global positioning system (GPS) for localization in the vehicle's frame of reference. Apart from this, Inertial Measurement Unit (IMU) is also used to obtain localization in vehicle's frame. This provides another input for vehicle's movement to know its position and orientation accurately. To know the vehicle's heading with respect to the Earth's magnetic field, combination of triple axis magnetometer and triple axis accelerometer is used. Combination of these two sensors provides tilt compensated values of vehicle's heading.

GPS readings are obtained and converted to link vehicle's local coordinate system to the Global coordinate system using Spherical Polar coordinates. The heading angle of the vehicle, in local coordinate system, with respect to Earth magnetic field direction (N-S) is obtained from the compass. We have used GPS system and compass to determine the angular rotation of the vehicle .This angle of rotation can be fed to the wheel encoder using algorithm to rotate precisely with same amount as calculated. Here Inertial Measurement Unit (IMU) helps in tracking precise angular rotation of vehicle and its motion.

Implementation in ROS:

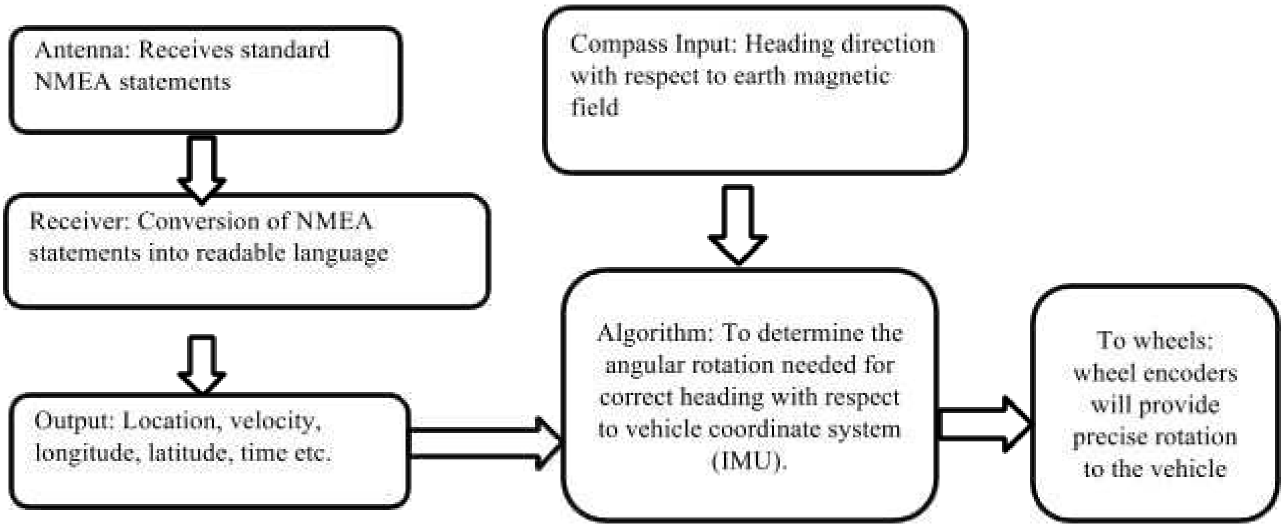
With respect to ROS, the navigation module includes some basic tasks to be done:
1) Giving the bot a direction toward next GPS waypoint
2) Sending goal points to navigation stack
2) Providing the remaining modules with necessary data (this includes providing odometry data to robot_pose_ekf package for estimating bot's current position).

How GPS values are obtained: The gpsd daemon running on the Ubuntu looks into device file onto which the values are written by the GPS. Using telnet we publish these values onto a port. The gpsd_client node running in ros listens to this port and takes in the value and publishes it by a topic named /fix.

How a direction is given: The gps_com_filter node implemented in the gps_t package takes in an input from both the compass (for calculating bot heading) and the GPS data (for calculating direction towards GPS waypoint) and depending on them calculates a direction which is published for the remaining modules to make a decision based upon.

Sending GPS goal points: Navigation stack requires goal points specified on base frame for making a movement. The gpsgoal node present in the package simple_navigation_goals sends
goal points in bots base frame to the navigation stack. These goal points are calculated based on the present GPS coordinates and the bots heading.

Odometry data: The robot_pose_ekf package requires odometry data for calculating present pose of the bot. This is calculated by the gps_common node present in gps_umd package
The odometry data calculated is published by the topic name
Block diagram for above mentioned steps:


Antenna: Receives standard NMEA statements
Receiver: Conversion of NMEA statements into readable language
Output: Location, velocity, longitude, latitude, time etc.
Compass Input: Heading direction with respect to earth magnetic field
Algorithm: To determine the angular rotation needed for correct heading with respect to vehicle coordinate system (IMU).
To wheels: wheel encoders will provide precise rotation to the vehicle

7. What makes the project unique?

The features, mentioned earlier can be easily adapted and its uses are multi-varied and distinct. The prototype when implemented on large scale can be used for sting operations by the military. The Image processing technology used for path following can be extended for many Human Computer interface technologies which can be used in the consumer market. On the whole the complete integration of hardware and software makes it very versatile for different applications.

The technology we are developing is an intelligent ground vehicle. It is a vehicle that can be used in semi –rugged terrains for military operations, mitigation, surveillance and sample collection. Capable of traversing over sand, gravel and grass, the vehicle realizes autonomous navigation and detects obstacles through laser ranging. Algorithms developed to automatically generate waypoints, will guide the vehicle in unidentified regions. Manual control of the vehicle is achieved using wireless communication. The vehicle can thus be controlled from a remote base station. Together with an image processing module in place, the vehicle is capable of independently navigating in a new environment. It is also capable of delivering a live video, recorded by the on-board camera at a remote location. The technologies involved in the project are those of emerging industries today. Among those applications are many with great opportunities for breakthroughs and innovation.

8. Cost Analysis

Part/Electronics	Company & Model No.	Description	Amount
Laser Range Sensor	SICK AG & LMS 111	2D Laser scan sensor	$3,417.50
Inertial Measurement sensor	Spark Fun & Razor	3 Axis-Gyro, Accelerometer and Magneto-meter	$209
Global Positioning System	U-blox & Neo-6Q-0-001	Standard GPS Module	$97.30
Digital Compass	Spark Fun & Compass LSM303	Tilt compensated Compass	$29.10
Camera	Logitech & C310	HD720P Webcam	$27.40
Wheel Encoder	Autonics & E30S4-500-6-L-5	Rotary encoder	$231.20
Motor Driver	Polulu & VNH5019A-E	H Bridge Motor Driver	$118.40
Fabrication	Prototype-1 and Final Vehicle	Aluminium Frame and Custom Made Steel Connecting Rods	$571
Motor	Mechtex DC 52A	24V-DC, 40:1 Gear Ratio	$257
Total			$4,957.9