International Journal of Advance Research and Innovation
Vol. 9(2), Apr-Jun 2021, pp. 97-102
Doi: 10.51976/ijari.922115
www.gla.ac.in/journals/ijari

Article Info

Received: 13 Feb 2021 | Revised Submission: 28 May 2021 | Accepted: 05 Jun 2021 | Available Online: 15 Jun 2021

Design and Development of an Intelligent Ground Vehicle for Constrained Environments

Abhishek, Gurkirat Singh**, Harsh Raj**, Hrithik Tanwar** and A. K. Madan***

ABSTRACT

The Intelligent Ground Vehicle Competition (IGVC) is one of three, unmanned systems, student competitions that were founded by the Association for Unmanned Vehicle Systems International in the 1990s. The IGVC challenges engineering student teams to develop, test, and compete with their intelligent vehicles, focusing on a series of autonomous mobility objectives pertaining to advanced control theory, machine vision, vehicular electronics, and mobile platform fundamentals to design and build an unmanned system. The IGVC offers a design experience that is at the very cutting edge of engineering education. Autonomous robots are becoming prevalent in many aspects of modern society and the IGVC, while being in a controlled environment, provides a great example of how self-navigating vehicles could be used, such as going into areas that may pose risk to human life. Our goal is to design and construct a ground vehicle with autonomous capabilities. LIDAR, camera (computer vision), IMU sensor, and GPS enable the vehicle to function autonomously. The vehicle is capable of detecting and following lanes, avoiding obstacles, and tracking GPS coordinates. The entire system is run on the ROS (Robot Operating System), with sensors and actuators acting as nodes that interact with one another.

Keywords: *Autonomous systems; Control algorithms; Mobile robots; Motion planning; Obstacle detection.*

1.0 Introduction

In this age of automation, robotics is a significant area of research. A computer-controlled robot, unlike humans, can operate quickly and accurately without being tired. In a dangerous world, a robot may also perform pre-programmed tasks, reducing human risks and threats. A mobile robot, for example, can quickly detect, identify, locate, and neutralize a range of threats, such as enemy force action, chemical and biological agents, and impossible terrain or impassable routes or roads. While there has been great progress in dealing with individual scenarios like lane detection, obstacle detection, collision avoidance, there is still a lot of ground to be covered to achieve an entirely integrated autonomous driving. Our aim is to design and build an autonomous ground vehicle with the ability to navigate through a constrained environment and a set of GPS waypoint targets while detecting and following lanes and avoiding obstacles. The paper has been divided into mainly three sections namely Mechanical design, Embedded system and the Software architecture.

2.0 Mechanical Design

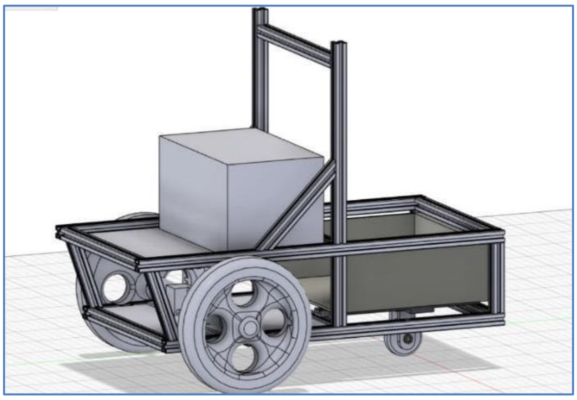
The vehicle's body is designed to be compact and easy to transport, as well as modular and space-efficient, with minimal adverse heating effects on electronic components. The CAD model of the vehicle was created on Solidworks. The frame is made of T-slotted aluminum members, which makes the design highly flexible and allows for easy assembly and disassembly on the working model. T slot braces, gussets, and plates may be used to link the members rigidly.

The outer body is covered with Bakelite sheets. It keeps its shape and is resistant to heat, scratches, and solvents that can harm it. It is also electrically resistant and valued for its low heat conductivity. Because of its low coefficient of friction and robust surface, it is the preferred material. The electronic bay is covered with transparent acrylic sheets. It

**Corresponding author; Department of Mechanical Engineering, Delhi Technological University, Delhi, India*
(E-mail: abhishek_2k18me005@dtu.ac.in)
***Department of Mechanical Engineering, Delhi Technological University, Delhi, India*

helps the controller to see through it and detect any errors. Underneath the motors, silicon-Germanic rubber is used to increase friction between the foundation and the motors, which absorbs vibrations. The LiDAR mounting framework has an extra rubber sheet that acts as a suspension to keep it stable when in motion. The laptop is kept in place by a separate suspension on the castor. The two main drive tires are made of a durable rubber polymer that absorbs damage. The vehicle's suspension is made up of these, as well as the PU caster wheel, which combines the elasticity of rubber wheels with the stiffness and resilience of metal wheels. To minimize movement and movements from the vehicle when it is in motion, the camera is fixed to a rugged and durable mast. Vibrations are often filtered out, and vehicle movement over rugged terrain is taken into account by the camera program.

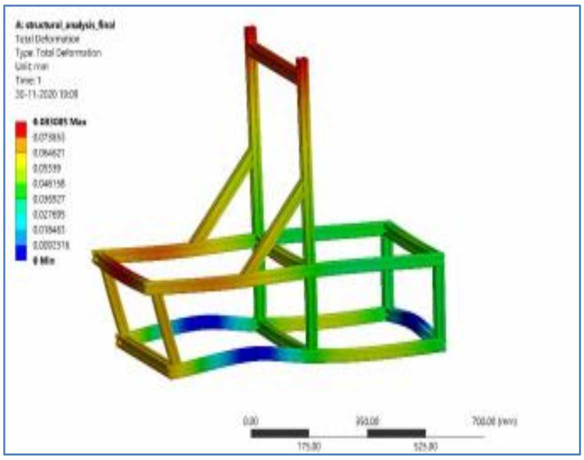
Figure 1: Side-View of the UGV



To choose the vehicle's drivetrain, a distinction was made between a four-wheeled and a three-wheeled (3rd wheel – a caster wheel) bot, with the relative advantages of each noted. It was decided to go with a drivetrain with two powered wheels on the front and a free caster wheel on the back due to simpler manufacturability, quicker assembly, and lower cost and complexity of the three-wheeled drivetrains. A highly symmetric weight distribution and the differential drivetrain provide sharp turning radius [1]. During the design process, a compact lightweight frame with a low centre of gravity was successfully achieved.

In Ansys, a static structural analysis of the chassis was performed using the predicted loads and constraints. The working stress was well below Aluminium's yield stress, and deformation was minimal.

Figure 2: Vehicle Structural Analysis Deformation Distribution



3.0 Embedded System

Embedded system in the autonomous vehicle deals with collection of data from the environment, curating it and passing it to the devised algorithms. Further, it is used to execute the actuation commands produced by navigation algorithms. The electrical system design of the vehicle can be categorized into three parts: the processing unit, sensors and control, and actuation and power supply. The processing unit is a laptop which takes input from different sensor modules, analyzes and produces the desired output commands. Sensors and control include the camera, LIDAR, IMU, GPS, and Arduino microcontrollers. The power distribution system includes the batteries, motors, relays, and power circuit design.

3.1 Processing unit

The purpose of the processing unit is to:

- To analyze the inputs from peripherals hardware units.
- Compute the desired output.
- Implement control of execution parameters.

The primary processing unit is a laptop with 3 GHz CPU, 16GB RAM and GTX-1050Ti and 5 USB Ports. Not all the sensors are directly connected to the laptop but need a microcontroller for interfacing as there are multiple protocols, additionally using microcontrollers for low level filtering will save some clock speeds on the Main Processing Unit.

BeagleBone Blue is used for this reason due to its AM335x 1GHz ARM Cortex-A8 processor, 512MB DDR3 RAM, 4GB 8-bit eMMC flash storage and inbuilt 9 axis IMU.

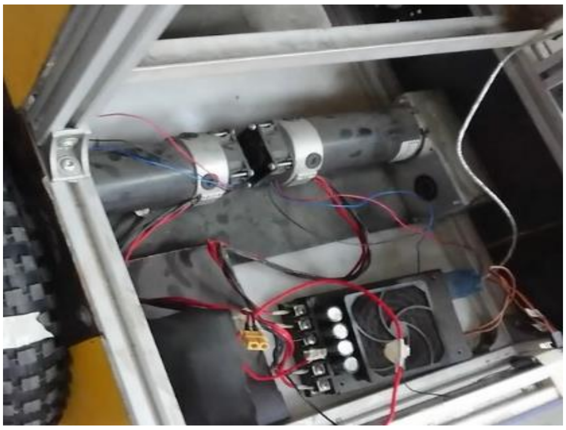
3.2 Sensors and control

The electrical schematics of the vehicle have been planned so that it depicts insight, long working hours, safe design, and sturdiness. A Global Positioning System (GPS) antenna, Inertial Measurement Unit (IMU) sensor, and a front view camera are mounted on the chassis to get raw data which is sent to the processing unit for additional handling. MPU6050 MEMS has been utilized to acquire the inertial vectors. Multiple IMU's will provide the 6 axis values with some uncertainty and a lot of drift, this can be settled by using either complementary filters or some flavor of Kalman filter [2]. The platform is designed with a fully modular electronics bay. This efficient design allows for the main power systems to be separated from the chassis via series of easy disconnects. The removable hardware allows for easy off-chassis integration, diagnostics, and testing. RTK based GPS technology is utilized to take advantage of the correction feedback mechanism of base stations. Sparkfun Zed-f9p RTK-2 module is used for the same. 2D Hokuyo LIDAR is used for detecting obstacles. The LIDAR readings are interpreted as point clouds and processed for noise removal. This point cloud is then thresholded by a specified distance, and all points lying at a greater distance are left out. Remaining points are considered as obstacles. Lanes and potholes are additionally treated in the same way as obstacles, as a point cloud. A stereo camera is used to capture a large part of the environment and the image processing camera extracts the lane from the image. Camera is needed for detection of lanes and potholes using OpenCV. SJCAM SJ5000 camera has been used for this purpose. The field of view of the camera is 170 degrees. The wide FoV enables us to get a better picture of the lanes on both the sides of the track. The camera is also water-proof which contributes to the waterproofing of the robot.

3.3 Actuation and power supply

AmpFlow AF160 motors are controlled by AmpFlow Dual Motor Speed Controllers that can output up to 42 Volts and can handle 160 Amps per channel. An inbuilt PID controller is used for generating actuation commands. The input consists of linear and angular velocities of the vehicle and outputs the motor commands.

Figure 3: Actuation System: Motors, Motor Driver and Battery



The main power source for the vehicle is a 22.2 V, 16000mAh Lithium-Polymer Battery. It can give great instantaneous discharge current upto 400A. It weighs just 2120 grams and is smaller in size compared to alternatives such as NiCd, Ni-MH and Lead acid batteries.

Table 1: Power Requirement of the Electrical Components

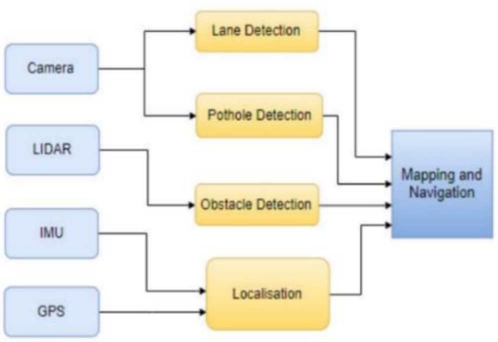
Electrical component	Max power consumption	Operating voltage	Source
GPS Sparkfun GPS-RTK2 module	297mW	3.3V	Li-Po Battery
Camera SJCAM SJ5000 WIFI Action Waterproof	10W	10V DC	Li-Po Battery
Lidar Hokuyo's URG-04LX detectable range is 20 mm to 4000 mm	2.5W	5V	Laptop (via USB)
Motors AMPFLOW FAN COOLED 24V A28-400-F24-G	700W	24V	Li-Po Battery
Microcontroller Arduino Mega	0.1W	5V	Laptop (via USB)
Cooling fan(x2) MAA-KU DC12025 4.72" inches,2400rpm	3.36	12V	Dedicated battery

Our UGV has two E-stops for emergencies: a mechanical E-stop and a wireless E-stop. The mechanical E-stop is implemented through hardware. It is a red push button placed in the back of the vehicle and is connected to the relay. When the mechanical E-stop is enabled, it cuts off the relay to the motors. Whereas the wireless E-Stop is based on FlySky Transmitter-receiver paired up with an Arduino which can be activated through the transmitter in case of emergency.

4.0 Software Architecture

The vehicle uses the combination of inputs from camera and LIDAR to navigate through the lanes and detect obstacles. The vehicle uses two cameras for lane detection (one for each lane) which helps in detection of wider lanes and gives a comparatively closer view of the lanes i.e. where the vehicle is currently located with respect to the lanes and hence we can know are when the vehicle is deviating from its desired path at a faster rate. The use of two cameras thus gives an advantage over a single wide angle camera. The 2D Hukoyo LIDAR is used to map the obstacles and find the distance from them.

Figure 4: Flow Chart of Sensor Control

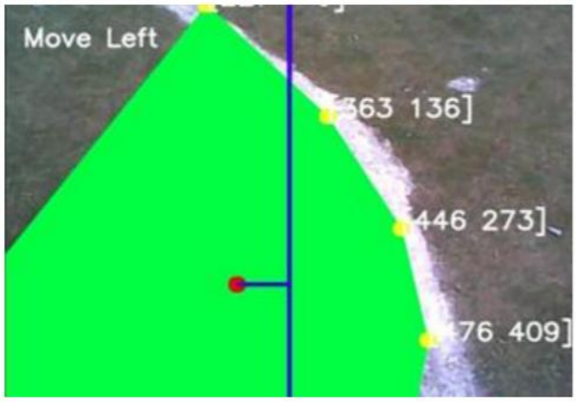


4.1 Software strategy and path planning

4.1.1 Lane detection

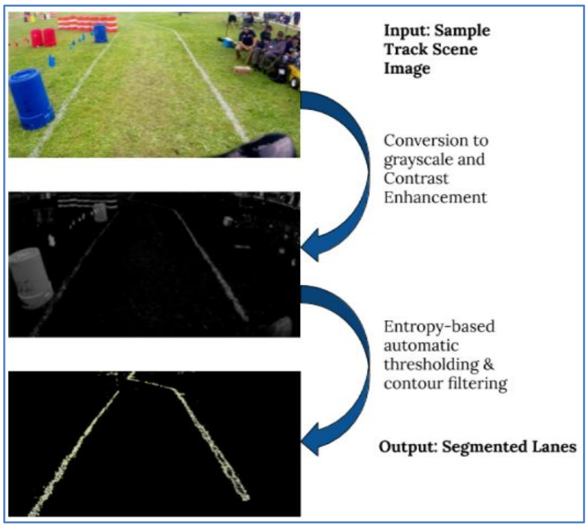
To detect lanes initially the noise in the image is removed through Gaussian blurring. After blurring adaptive thresholding is used to detect the pixels of white color. The HSV(Hue, Saturation, Value) color format was used as by applying a threshold to only hue and saturation we can make the detection light intensity invariant (since 'Value' 11 defines the intensity of the color) [3]. This would give a binary image with white pixels representing the lane. Further noise is removed using 'Opening' morphological operation i.e. erosion followed by dilation. A histogram is generated that shows high intensity points hence showing white color in the image. Then the lane is detected from the white pixels by using polynomial approximation as shown below.

Figure 5: Polynomial Mapped Over the Detected Points



A polynomial is mapped over the detected points hence giving us the lane. Another advantage of using polynomial approximation is that when the view of some part of lane is obstructed by some obstacle even then we can get the whole lane by approximating the lane using the visible lane points [4].

Figure 6: Main Stages in Image Processing Pipeline



Once the lane was detected, multiple points were taken from the lane and the slope was calculated giving the direction of the lane. Then the motors are given commands accordingly to keep the vehicle at a fixed distance from the lane and parallel to it.

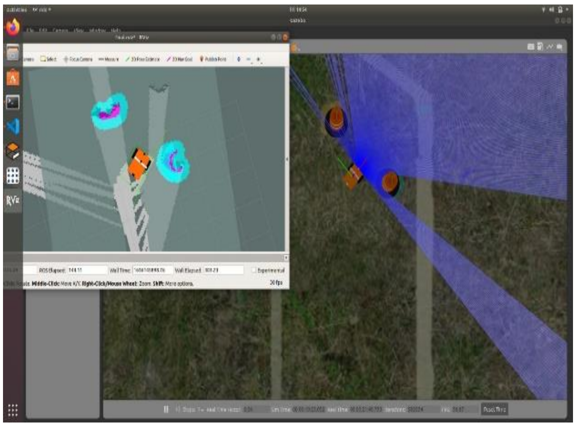
4.1.2 Obstacle detection and avoidance

LIDAR is used to detect obstacles, which provides a point cloud. Importing this PointCloud into the Robot Operating System (ROS) [5], the machine visualizes the hard limits of its environment and gives the points of detected objects a high cost. After filtering and giving the costs to points, a cost map is generated to avoid the areas with high cost.

A threshold distance is set and when the vehicle is closer than the threshold distance, command is given to avoid the obstacle. A combination of lane and obstacles position information is required for obstacle avoidance to make sure that the vehicle stays within the lanes while avoiding the obstacles.

To avoid obstacles, the position of the lane and the obstacle is noted and then the possible path with maximum gap is followed. For example in the case where an obstacle is placed in the middle of the lane, the algorithm would measure the pixel distance between lane and obstacle and either side and will choose the larger distance.

Figure 7: Simulated Run on Gazebo



4.2 Software integration

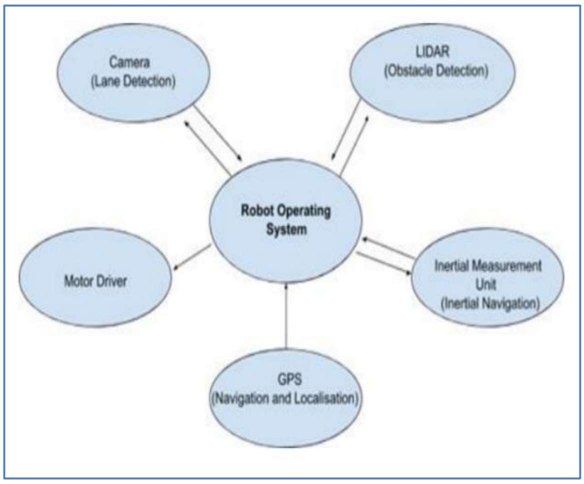
4.2.1 Robot operating system

For efficient resource management, and to enable seamless communication of data between modules the open source Robot Operating System developed by Willow Garage has been used.

Building the system over an Operating system, helps maintain real time constraints and avoids resource conflicts.

The Lidar driver, Object Detection, Image Processing Stack and the microcontrollers used for reading sensor data and generating pwm for motor direction and speed control are each assigned separates nodes within the system and share data with each other using different ROS topics which use Multiarray messages.

Figure 8: State Map of the System



5.0 Conclusions

In this paper, we have presented the design and implementation of an intelligent ground vehicle. A robust design combined with heavily tested software has allowed the vehicle to navigate through GPS waypoints, follow lanes and avoid obstacles while steering through sharp turns. It can easily navigate in a low speed - low traffic regions (having appropriate guidance marks). With some modifications this system can be easily implemented in a light weight electric vehicle such as a golf cart, for in-campus transportation.

References

[1] SV Shirsath, KR Jadhav, RV Patil, AV Mohite, DD Patil. Zero turn vehicle. International Research Journal of Engineering and Technology, 2016.

[2] T Moore, D Stouch. A generalized extended kalman filter implementation for the robot operating system. In Emanuele Menegatti, Nathan Michael, Karsten Berns, and Hiroaki Yamaguchi, editors, Intelligent Autonomous Systems 13, 2016, 335–348,

[3] A Anwer, A Baig, R Nawaz. Calculating real world object dimensions from Kinect RGB-D image using dynamic resolution. 12th International Bhurban Conference on Applied Sciences and Technology (IBCAST), 2015.

[4] https://doi.org/10.1109/IBCAST.2015.7058504.

[5] P Soni, P Tanwar, KS Raju, SA Akbar. Multi obstacle identification using image processing technique. International Conference on Advanced Electronic Systems, 9, 2013, 160–165.

[6] M Quigley, K Conley, BP Gerkey, J Faust, T Foote, J Leibs, Rob Wheeler, Andrew Y Ng. Ros: an open-source robot operating system. volume 3 (1), 2009.