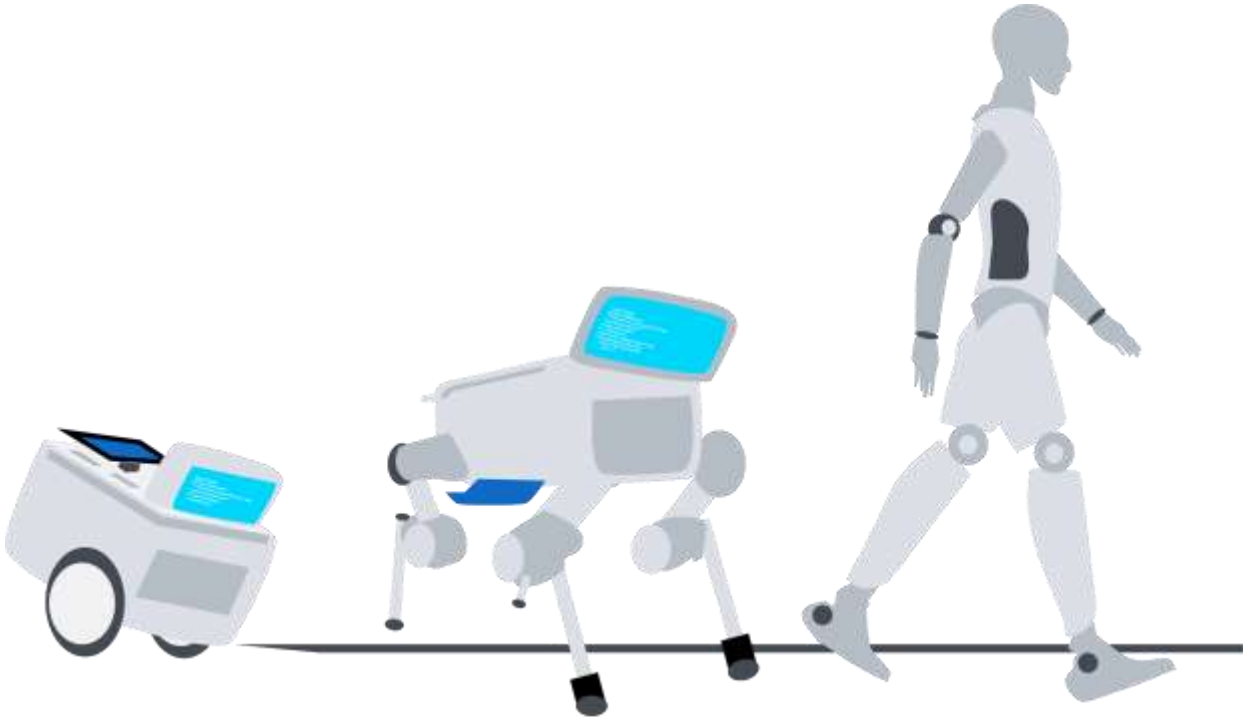
SVKM'S NMIMS
Deemed to be UNIVERSITY

MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT & ENGINEERING™



Team **D.A.R.V.I.N.**

Vehicle Name: **CAMAZOTZ**

Faculty Mentor: **Prof. Kashyap Joshi**
(Electronics and Telecommunication Dept.)

Team Captain: **Saksham Gupta**

Name	Role
Kashyap Joshi	Faculty Mentor
Saksham Gupta	Team Captain
Abhishek Agarwal	Team Manager
Mit Desai	Team Mechanical
Vaibhav Mishra	Team Mechanical
Sarthak Mishra	Team Programing
Vaibhav Raheja	Team Programing
Yash Desai	Team Programing
Mihir Momaya	Team Electronics
Param Nagda	Team Electronics
Anjnya Khanna	Team Electronics
Devansh	Mechanical Volunteer
Tejas Jindal	Mechanical Volunteer
Alfaz	Mechanical Volunteer
Mayur Satpute	Programing Volunteer

Contents

1 Introduction 4

2 Team Organization 4

3 Mechanical 5

3.1 Overview 5

3.2 Frame Design 6

3.3 Wheel Assembly 6

3.4 Sensor Stand 7

3.5 Weather Shield 7

4 Electronics 8

4.1 Overview 8

4.2 Electronic Suite Description 8

4.2.1 Microcontroller Unit 9

4.2.2 Main Controller Unit 9

4.2.3 IMU 9

4.2.4 LiDAR 10

4.2.5 INS 10

4.2.6 Stereo Camera 10

4.2.7 Controller Area Network (CAN) 11

4.2.8 Motor Driver 11

4.2.9 Zigbee 11

4.3 Circuit Operation 12

5 Software System 14

5.1 Overview 14

5.2 Main Software Pipeline 14

5.3 Lane detection Pipeline 15

5.4 Object Detection Pipeline 16

5.5 No Man's Land detection and navigation 18

5.6 Arduino communication 19

5.7 Arduino motor driving 19

1 Introduction

Team D.A.R.V.I.N is the student organization for competitive robotics at the NMIMS university, India that has designed its first iteration of autonomous vehicle Camatoz to compete in the 28th Intelligent Ground Vehicle Competition. With integration of an improved control system and robust optical localization system, we have developed an Intelligent ground vehicle that pushes the boundaries of robotics. Camatoz includes new cutting edge capabilities including but not limited to its robustness, design, swiftness and software architecture. Our vehicle Camazotz focuses on accessibility for users, safety improvements to the electrical system, and effectiveness of our software and is a representation of our team's effort.

2 Team Organization

The team is organized into four modules: Mechanical, Electronics, Software and Management; based on the expertise and domain knowledge that is required in building the prototype and striking sponsorship deals for components and logistics. Modules have clear cut roles and responsibilities and many team members contribute to multiple modules. The team is overseen by a faculty advisor and team head. The team head coordinates between each module, organizes meetings, evaluates progress and reports to the faculty advisor. Each module has a head and is responsible for progress within the module. Module heads report progress to the team head.

The Management team looks after all non-technical tasks including talent acquisition, public relations, strategic industrial relations etc. The Mechanical module has members working in Design, Simulation and Manufacturing of the prototype. It overlaps with the Software and Electronics team in the area of Control systems and Mechatronics. The Electronics team deals with the electronic circuitry, sensor data acquisition & signal processing and actuation. They overlap with the Software team in the area of embedded systems and microcontroller programming. The software team works with the development of software solutions for Autonomous Navigation and Interoperability.

3 Mechanical

Figure 1: Frame with Basic Wheel Assembly

3.1 Overview

Right at the start while going ahead with the mechanical design we wanted our vehicle to be extremely modular with smooth assembly and disassembly process and efficient use of space. As we wanted to go ahead with modular design we went ahead with an aluminum profile to ensure modularity. Our vehicle is a three wheeled vehicle with two DC motors attached to the wheels via custom coupling at the front and a castor wheel at the rear. Aluminium profile were 30mm*30mm cross-section. The mild steel brackets at the bottom, houses the 142kgF DC motor.

3.2 Frame Design

The frame design started from finalizing the wheels and wheelbase and steering system after which we started to design the chassis around our system and the payload. To ensure that our vehicle is modular we designed our frame using aluminium frame and used T bolts to ensure easy assembly and disassembly. We kept our battery bay and payload bay very close to the ground to maintain a very low centre of gravity. Camera members were kept modular to change the pitch angle and the height of the stereo camera module that we are using.

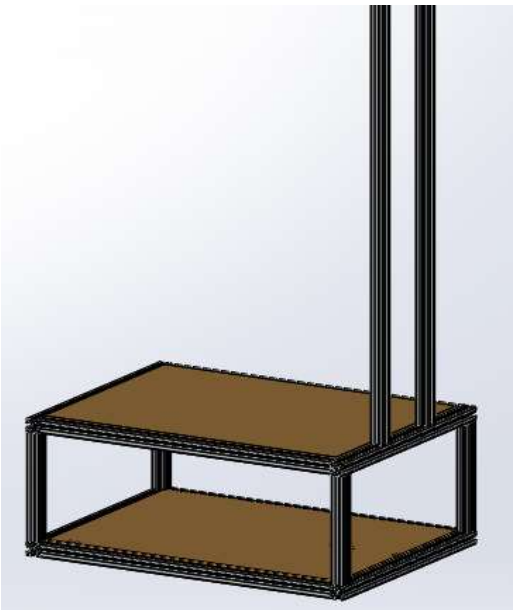


Figure 2: Naked Frame

3.3 Wheel Assembly

The wheel assembly comprises 142kgf DC motor which is rested on a custom designed mild steel motor mount, wheel, wheel hub and a jaw coupling. The wheel is assembled with our in house manufactured wheel hub. One end of the jaw is meshed with the shaft of the motor and the other with our custom wheel hub , This coupling transmits the torque through compressionion of an elastomeric spider inserted between two intermeshing jaws.

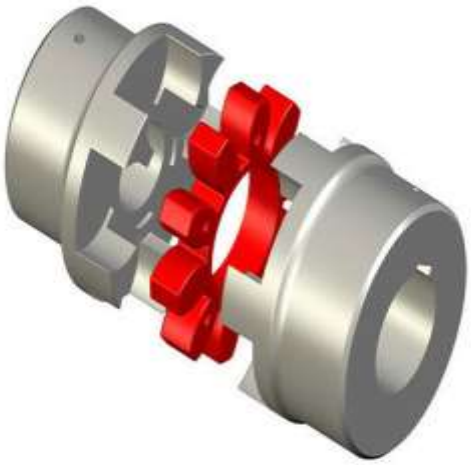


Figure 3: Jaw Coupling

Figure 4 : Custom Wheel Hub

3.4 Sensor Stand

Our sensor stand comprises of a 3D LiDAR and a stereoscopic camera ZED by stereolabs.There is enough modularity to customize both the sensor according to the various scenarios, We have carefully placed shock absorbers beneath our sensor stands to reduce distortion while maneuver on uneven surfaces and for the safety of our sensors .

3.5 Weather Shield

To protect our electronics from water damage we have 3-D printed a custom weather shield using PLA that will avoid the water and dust coming in,on the top of that we have painted a coat of silicone resin that will ensure heat resistance and weatherability. To hold the shield we have used silicone tapes which also add extra water resistance.

4 Electronics

4.1 Overview

The vehicle is powered by a 24V LiPo battery, 16000 mAh. There are 2 auxiliary supplies 12V and 5V respectively. The main processing is done on Jetson TX2 and the motors are driven by microcontroller ATMega 328. The communication between microprocessor and microcontroller is made through a controller area network (CAN bus). For safety purposes the vehicle has two kill switches one on the vehicle and another is E-kill. The vehicle can be turned on/off using both these switches. The vehicle is driven by two 142 kg-cm torque DC motors. For lane detection and object detection combination of LiDar and stereo vision camera.

4.2 Electronic Suite Description

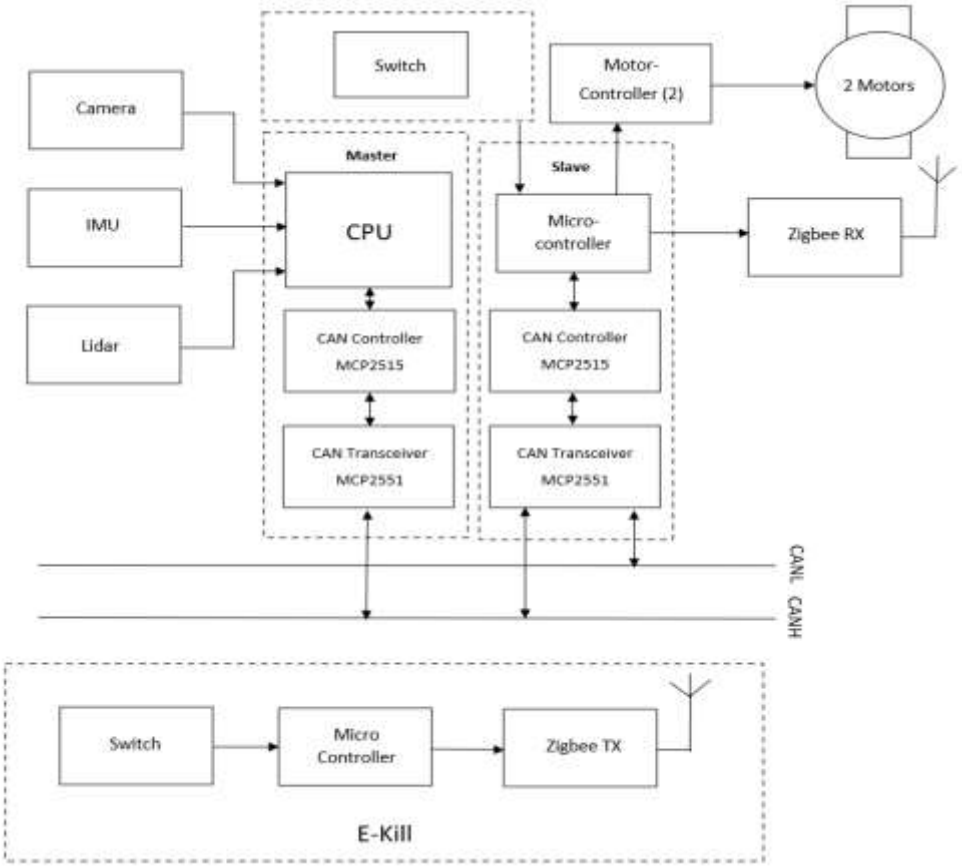


Figure 5: Block diagram of the designed system

4.2.1 Microcontroller Unit

ARDUINO UNO:

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins,6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button.

ARDUINO MEGA:

The **Arduino Mega 2560** is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

4.2.2 Main Controller Unit

NVIDIA JETSON TX2:

The NVIDIA Jetson TX2 Developer Kit is a full-featured development platform for visual computing. It is ideal for applications requiring high computational performance in a low power envelope. It comes pre-flashed with a Linux environment, includes support for many common APIs. The board exposes many standard hardware interfaces, enabling a highly flexible and extensible platform.

4.2.3 IMU

SPARTON IMU AHRS 8P:

It is capable of sensing the dynamic heading, dynamic pitch & roll and its acceleration with an update rate(sample/sec) of 100 and a baud rate of 115200 baud rate. It also transmits data over the serial bus.

4.2.4 LiDAR

VELODYNE LIDAR:

Lidar is an acronym for "light detection and ranging." The technology uses eye-safe laser beams to create a 3D representation of the surveyed environment. It is a remote sensing method that uses light in the form of a pulsed laser to measure ranges (variable distances) to the Earth. These light pulses—combined with other data recorded by the airborne system— generate precise, three-dimensional information about the shape of the Earth and its surface characteristics.

4.2.5 Inertial Navigation System

XSENS MTi-7:

Xsens MTi-7 Global Navigation Satellite System (GNSS)/Inertial Navigation System (INS) module is a miniature motion tracking module with multiple GNSS receiver support. This module uses advanced sensor fusion algorithms. The MTi-7 GNSS/INS module features a miniature SMD form factor (12.1mm x 12.1mm) and consumes low power. This module is ideal for upcoming technologies including drones, Unmanned Aerial Vehicles (UAVs), smart farming, unmanned control, Internet of (Moving) Things, and robotics.

4.2.6 Stereo Camera

STEREO LABS ZED:

ZED is a camera that uses dual lenses and through triangulation, it provides a three-dimensional understanding of the scene it observes, allowing the vehicle to become space and motion aware. It captures high-definition 3D video with a wide field of view and outputs two synchronized left and right video streams in side-by-side format on USB 3.0.

4.2.7 Controller Area Network (CAN)

A Controller Area Network (CAN bus) is a robust vehicle bus standard designed to allow microcontrollers and devices to communicate with each other's applications without a host computer. It is a message-based protocol, designed originally for multiplex electrical wiring within automobiles to save on copper, but can also be used in many other contexts. For each device the data in a frame is transmitted sequentially but in such a way that if more than one device transmits at the same time the highest priority device is able to continue while the others back off. Frames are received by all devices, including by the transmitting device. For this CAN controller MCP 2515 and CAN transceiver MCP 2551 are used. Using this CAN controller 124 controllers can communicate with each other.

4.2.8 Motor Driver

Hercules Motor Driver:

Hercules 6V-36V, 16Amp Motor Driver can take up to 30A peak current load and can be operated up to 3 KHz PWM. Motor driver can be interfaced with 3.3V and 5V logic levels. Motor driver has built-in protection from under / over voltage, over temperature and short. The Motor driver has a terminal block as power connector and 7 pin 2510 type relimate connector for the logic connection.

4.2.9 Zigbee

ZIGBEE S2C PRO:

The XBee/XBee-PRO Zigbee RF Modules provide wireless connectivity to end-point devices in Zigbee mesh networks. Using the Zigbee PRO Feature Set, these modules are inter-operable with other Zigbee devices.

4.3 Circuit Operation

The heart of the system is Jetson TX2 processor board. Input from camera and Lidar sensor is given to the processor. The processor processes this data and takes necessary actions for the vehicle. The actions include signal to the Arduino which ultimately drives the motor which is a 24V high torque DC motor. The signal is sent to the Arduino through the CAN Controller(MCP 2515) and CAN Transceiver(MCP 2551). Camera and Lidar are used for object detection, lane detection and detection of the potholes.

For E-kill, Zigbee is used along with a microprocessor Atmega328p. When switch 1 is pressed the transmitter transmits hexadecimal number “AA”. At the receiver when “AA” is received it sends a signal to stop the motor. When switch 2 is pressed the transmitter transmits hexadecimal number “CC”. At the receiver when “CC” is received it sends a signal to start the motor again after it is stopped by pressing switch 2.

For the Feedback system the DC motors are fitted with encoders which count the values for the number of rotations. Using the count of the encoders the speed of the vehicle is controlled. The motors are controlled by using the data from the encoders, camera and Lidar sensor.

For mechanical kill the supply to the motor driver is cut-off through a 24V electromechanical relay. The whole vehicle is powered using a 24V DC Li-Po battery. The power distribution board is used to provide power to all the sub-systems. Two auxiliary supplies of 12V and 5V are used to supply power to Jetson TX2 and Microcontroller Atmega328p respectively.

The Controller Area Network (CAN) is used for communication between the microcontrollers to increase the efficiency of the system. As CAN being 2 wired communication protocol it is easy for the microcontrollers to communicate through this protocol.

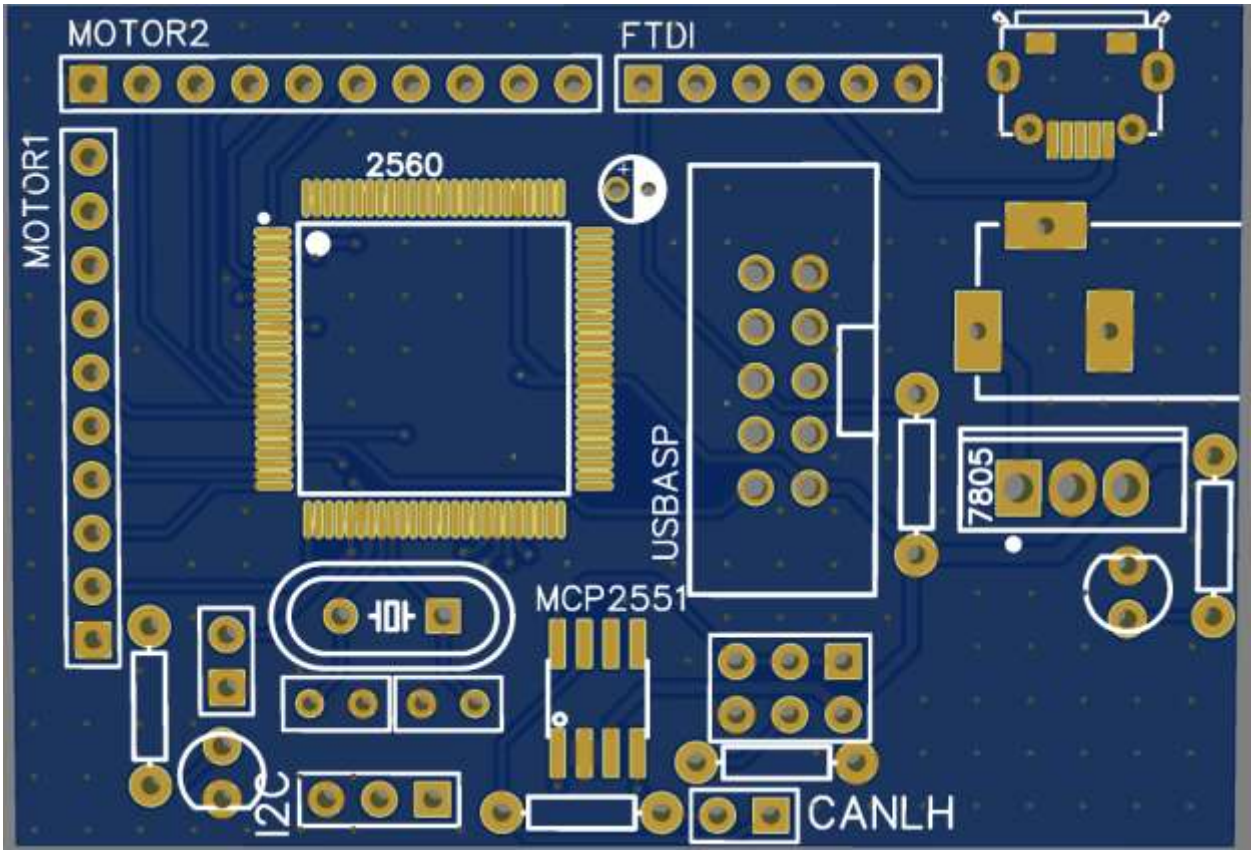


Figure 6: PCB of the designed circuit

5 Software System

5.1 Overview

Our software is written completely on python and is executed on the Nvidia Jetson tx2 running Linux. The tx2 is connected to an ATMEGA 2560 based custom PCB via can bus and I2C and is programmed to receive data from the tx2 and drive the motors. We have used the stereo labs zed dual camera which is capable of depth calculations. Originally this board was to handle the e-kill functionality with the help of Zigbee modules but during testing it was found that in certain cases the signal was ignored. Hence for safety we moved the e-kill functionality to a separate Arduino.

5.2 Main Software Pipeline

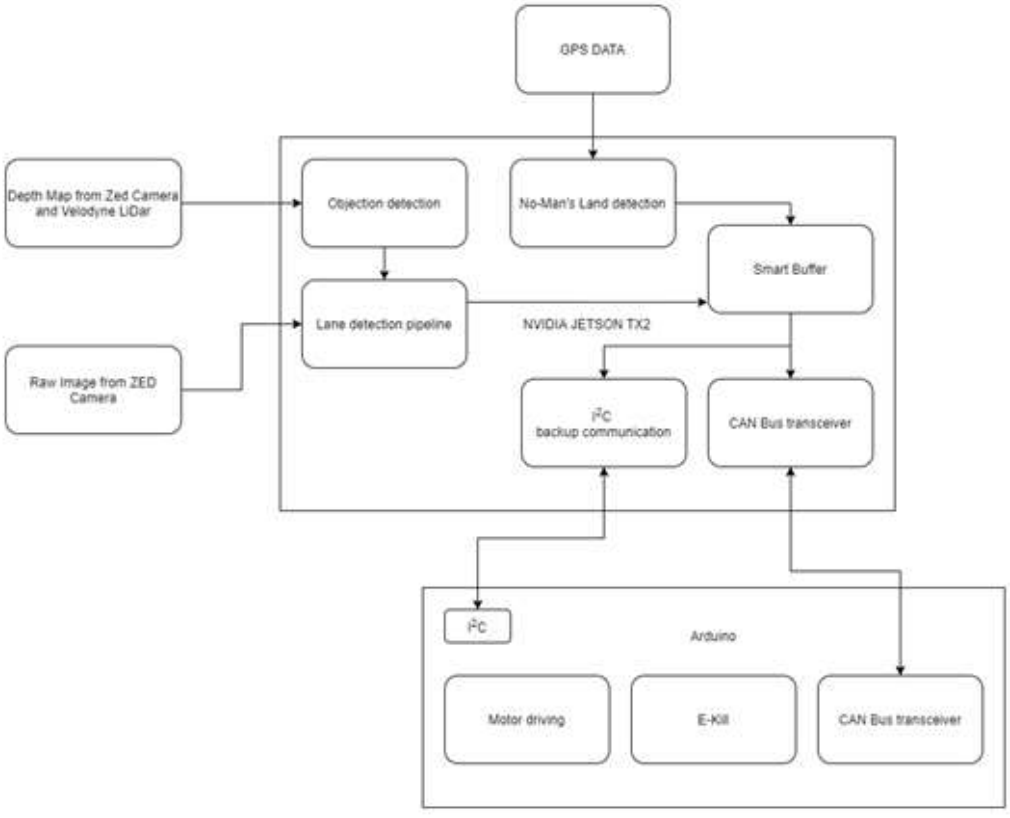


Figure 7: Software Architecture

Image, depth data are read from the camera and LiDar and processed by the various pipelines and functions to calculate location of objects and the direction of lanes. All these values are combined and given to the smart buffer. The GPS data is constantly monitored by the No man's land detection function and outputs data that tell the smart buffer which data to send to the Arduino. If the GPS coordinates of the robot lie within the given waypoints, the lane detection data is ignored. The object and GPS data is combined and sent to the Arduino.

5.3 Lane detection Pipeline

The image obtained from the zed camera is first converted to the HSV color space. This is done as different lighting conditions cause the original RGB values to change for the same color. This can cause issues later. HSV on the other hand separates the LUMA i.e. image intensity from CHROMA i.e. color intensity. Hence different lighting conditions have little effect on the image. The next step is color filtering. Since our lanes will be white in color, we use OPEN-CV functions to find pixels with values in a given range. This range is one of the factors that can be tuned later. We also apply a Gaussian Blur to smoothen the image. The next step is to remove less useful parts of the image. For this we use a region of interest mask and remove the rest of the image. This is also one of the factors that can be adjusted on field for better performance.

Next, we split the image into two parts-left and right. Each of these images are passed individually to the Hough transform line detector. The images are split into 2 to prevent mixing of lanes. The coordinates of lines found in each image are calculated. Lines with similar slopes are combined and rest are removed. The output after this stage will be 4 sets of x, y coordinates representing 2 lines hence the 2 lanes. The top points are used for finding the midpoint of the lane. The difference between this and the midpoint of the image is taken is the final output of the lane detection pipeline. This is combined with the other pipelines and sent to the Arduino.

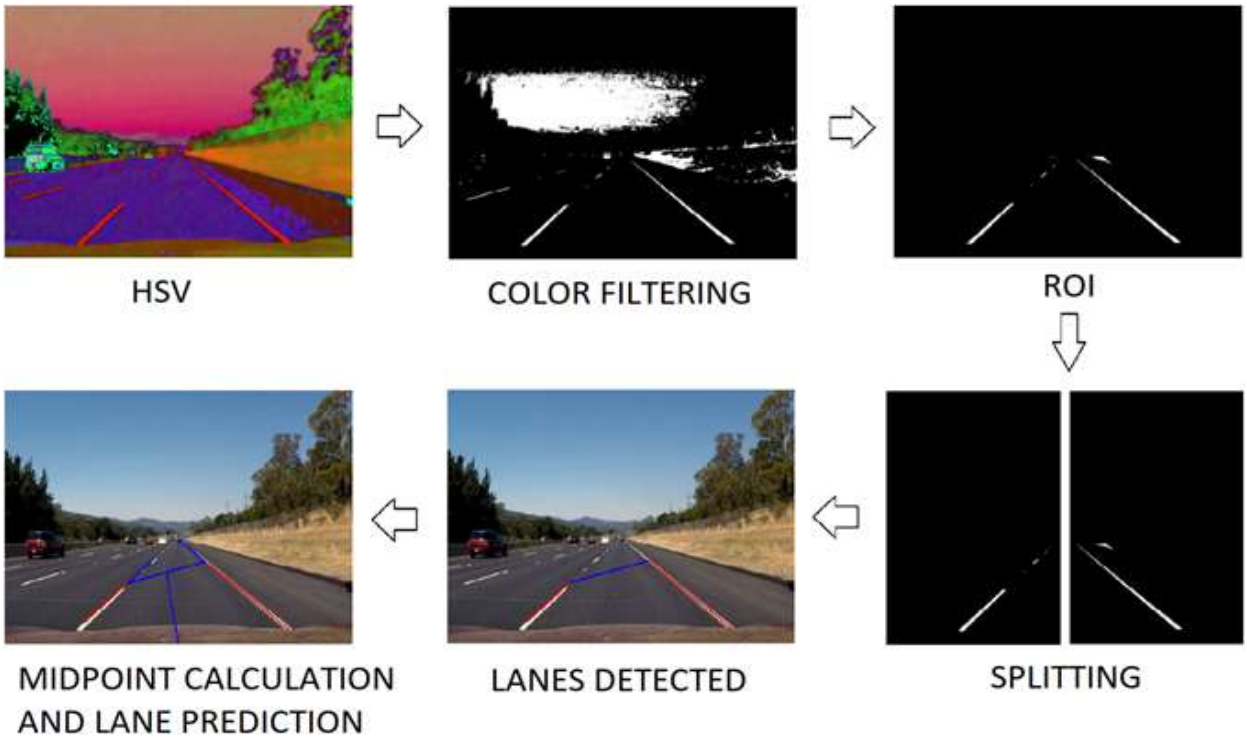


Figure 8: Lane detection pipeline test output

5.4 Object Detection Pipeline

For detecting objects in front of the vehicle we are using the depth capability of zed cameras and Velodyne LiDAR. The camera returns the depth value of each pixel and the LiDAR returns a complete map. Important data is extracted from both and combined. We use the same OPEN-CV function that we used before for detecting lanes but this time it will detect objects. Any object that appears within the given range will be detected by the pipeline. The pipeline will then calculate its location and return an offset that needs to be combined with the lane data. This offset can be positive or negative based on where the object is. Positive for right and negative for left. This value can then be subtracted from the lane data.

The reason we subtract this value can be made clear by the following cases:

Case 1: Lane is straight and object is on the left side

The lane is straight so the lane detection pipeline will output a value close to zero. Now the object is on the left side so a negative value will be returned by the object detection code. Therefore: lane detection value - (- object detection value) = lane detection + object detection value This causes the vehicle to move in the opposite direction as the object i.e. right.

Case 2: Lane is straight and object is on the right side

The lane is straight so the lane detection pipeline will output a value close to zero. Now the object is on the right side so a positive value will be returned by the object detection code. Therefore: lane detection value - object detection value = lane detection - object detection value This causes the vehicle to move in the opposite direction as the object i.e. left.

Case 3: Lane is right and object is on the left side

The lane is right so the lane detection pipeline will output a value greater than zero. Now the object is on the left side so a negative value will be returned by the object detection code. Therefore: lane detection value - (- object detection value) = lane detection + object detection value This causes the vehicle to move in the opposite direction as the object i.e. right.

Case 4: Lane is right and object is on the right side

The lane is right so the lane detection pipeline will output a value greater than zero. Now the object is on the right side so a positive value will be returned by the object detection code. Therefore: lane detection value - (object detection value) = lane detection - object detection value This causes the vehicle to move in the opposite direction as the object i.e. left.

Case 5: Lane is left and object is on the left side

The lane is left so the lane detection pipeline will output a value less than zero. Now the object is on the left side so a negative value will be returned by the object detection code. Therefore: -lane detection value - (- object detection value) =-lane detection + object detection value This causes the vehicle to move in the opposite direction as the object i.e. right.

Case 6: Lane is left and object is on the right side

The lane is left so the lane detection pipeline will output a value less than zero. Now the object is on the right side so a positive value will be returned by the object detection code.
Therefore: -lane detection value - (object detection value) =-lane detection - object detection value This causes the vehicle to move in the opposite direction as the object i.e. left.

This technique avoids the obstacles while staying in the lane.

5.5 No Man's Land detection and navigation

The GPS and compass readings are constantly monitored by the No man's land detection software module. The current coordinates are compared with the waypoints and if found equal within the range of error then the lane detection pipeline data is ignored. The compass is used for calculating the offset angle to the next waypoint. This data is used to guide the vehicle towards the next waypoint. The object detection data is still being used. When the vehicle reaches the next waypoint, the whole process is repeated. This is done x number of times where x is no of waypoints.

5.6 Arduino communication

We use CAN bus for transferring data to the Arduino. For backup we also have used the onboard I2C functionality of the Nvidia Jetson tx2. CAN bus is used in modern cars for communication between the ecu and other devices such as power steering etc. It is a robust and reliable communication protocol. It also enables us to add other modules to the network in the future very easily to upgrade or add new features. If in a case where the can bus fails to function the I2C bus gets activated and all data transmission occurs via I2C.

As a security measure to prevent hacks, we have developed a custom encryption and decryption algorithm that runs on both the Arduino and Nvidia Jetson Tx2. This algorithm is very light weight and works in real-time.

5.7 Arduino motor driving

The Arduino receives the data and stores it. The data received is basically an integer value that determines the offset of the center of the robot to the center of the lanes. The Arduino calculates PWM values for each of the 2 motors with the help of this integer and then drives the motor. The Hercules motor driver provides diagnostics for each of the 2 motors. This is monitored by the Arduino in real time. The Arduino also runs a PID loop in accordance with the encoders to run both the motors at exactly the same speed. The PID constants have been tuned manually after testing vigorously on different terrains.