Intelligent Ground Vehicle Competition 2024

Indian Institute of Technology Madras

Team Abhiyaan

Vidyut

Design Report

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

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

SATHYAN SUBBIAH, Ph.D.
Professor
Department of Mechanical Engineering
Indian Institute of Technology Madras
Chennai 600 036, India

Team Leaders

Aahana Hegde
Aayush Agarwal

Team Members

Electronics

Adithyaa R G
Arun Krishna AMS
Ashwaat Tarun T S
Dhaksin Prabu K
Kesava Aruna Prakash R L
Mouna Rajesh G
Nitin G
Prithvi P Rao
Shakti Anil Sharma
Varun Rajesh V
Vamsi Krishna Chilakamarri

Mechanical

Advait Abhijeet Kadam
Aneesh Bhandari
Ankit Yadav
Arvind P
Hridyansh Aggarwal
Nipun Nair
Perabathini Bhavana
Priya
Sarvesh Mane
Sudharsan Senthilraja

Software

Alan Royce Gabriel
Ashmitha
Ibrahim Kaif
Joel George Kallarackal
S Vishal
Saiharan Rajakumar
Tamoghna Saha
Trisha Wadhwani
Varinderpal Singh
Yash Purswani

*Names are hyperlinks for emails.

Contents

1. CONDUCT OF DESIGN PROCESS, TEAM IDENTIFICATION AND TEAM ORGANIZATION

1.1. *Introduction*

Team Abhiyaan consists of 50 interdisciplinary undergraduate engineering students from the Indian Institute of Technology (IIT) Madras. We share a passion for developing autonomous vehicles in India. We are guided by Prof. Sathyan Subbiah from the Department of Mechanical Engineering and receive support from the Centre for Innovation (CFI), a student-led facility at IIT Madras. The team is pleased to introduce Vidyut, a substantially improved iteration of last year's vehicle, Vikram. This design report thoroughly documents the conceptualization and development process behind Vidyut.

1.2. *Organization*

The team is split into four modules: Software, Electronics, Mechanical, and Business & Design. Each module is overseen by dedicated module leads responsible for managing module-specific tasks and operations. Team leaders oversee overall team coordination and make final decisions based on collective input from all members. The team emphasizes robust inter-module collaboration to facilitate effective knowledge sharing. Additionally, the team adopts a flexible hierarchy that encourages members to contribute across multiple modules. It fosters a dynamic and open environment where members are empowered to learn, question, and implement their ideas.

Senior members are pivotal in transferring their technical expertise to new recruits, enabling the team to leverage and expand upon its collective experience. Each member worked 3-4 hrs per day for the past year (Feb 2023 - May 2024). Regular team meetings are held to ensure alignment among all members. The use of *Notion* has been continued this year too due to good management and productivity improvements it gave during the previous year.

1.3. *Design assumptions and design process*

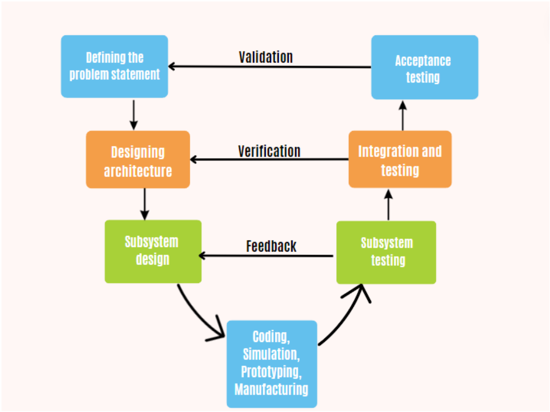

Figure 1: V-Model Design Process

The design process starts with getting familiar with the rules and regulations of IGVC. These, along with some assumptions made about the environment of the competition, various criteria regarding physical dimensions, performance metrics, core functionalities, and operating design domain (ODD) are decided. Assumptions include weather conditions, ambient lighting, and GPS line-of-sight. The next stage of the process is to define the problem statement. All the subsystems work simultaneously during a rigorous ideation phase, and the proposed design is discussed and agreed upon. The feasibility of the design is checked through simulations and numerical or analytical methods. Special care is taken in the ideation phase to resolve the previous year's failures (In our case, ramp detection and turning radius).

Following this, components are chosen. Cost, performance, industry standard compliance, and competition requirements are all considered. Each subsystem develops and tests individual prototypes, followed by final integration and testing. Based on testing results, the design is tweaked to optimize the performance. Care is taken to document each change for inference and future reference.

2. EFFECTIVE INNOVATIONS IN OUR VEHICLE DESIGN

2.1. *Innovative technology implemented on the bot*

2.1.1. *Mechanical*

1. **Independent Suspension:** A novel trailing arm-inspired suspension system, designed and produced in-house, offers enhanced wheel suspension tailored for autonomous bots. See 3.3
2. **Smart Cooling System:** A specialized cooling system controls the temperature inside the component housing. It uses temperature sensor data and adjusts fan speed based on the results of CFD simulation. See 3.5 and See 8
3. **Adjustable Positioning for Stereo Camera:** An innovative camera mount fabricated through 3D printing technology offers versatile adjustments for both elevation from ground level and camera pitch, facilitating precise positioning to achieve optimal height and angle for enhanced performance. See 3.4

2.1.2. *Electrical*

1. **Automotive Ethernet:** Using the popular, well-standardized, and fast Ethernet to transmit Odometry and feedback data from MCU directly to host ROS nodes in order to prevent transmission errors, decentralize processing of feedback data, and reduce CPU usage consumed by the Electrical ROS stack. See 4.4
2. **LoRa Mesh Network based E-Stop:** In-house designed LoRa Mesh network to increase the reliability of Emergency Stop signals in uncertain weather conditions. See 4.5.2
3. **Capacitive Collision Avoidance:** A novel and inexpensive method to detect objects in the blindspots of the bot, which the Zed Stereocam or the LiDAR will not detect. See 4.5.3
4. **PDS:** A Power Distribution System developed in-house that handles monitoring of battery state and health, wireless E-stop, and power distribution to all vehicle sections, all in one compact PCB. See 4.2

2.1.3. *Software*

1. **Augmented Reality:** Innovatively developed augmented reality (AR) models to enhance the testing capabilities of our bot systems. These AR tools provide an immersive and visual approach to testing, allowing for interactive simulation and real-time analysis of robot behaviors. See 5.5
2. **Behavior Trees:** Behaviour trees have been used to enhance modularity and reusability. Vidyut implements replanning and recovery utilizing behavior trees. See 5.4.1
3. **Lane Following:** Carried out lane detection by taking the camera feed and then perform Inverse Perspective Mapping(IPM) to obtain the world coordinates of the lanes. Nav2 is used to follow lane goals. See 5.4.3
4. **Ramp Detection:** Developed a robust split and merge algorithm specifically tailored for ramp detection using LiDAR data. This model efficiently identifies and delineates ramp structures by analyzing point cloud information, enabling accurate and real-time recognition of ramps within complex urban environments. See 5.4.5

3. DESCRIPTION OF MECHANICAL DESIGN

3.1. *Overview*

The mechanical module of Team Abhiyaan is focused on creating a robust vehicle that integrates seamlessly with the software and electrical modules. Building upon the previous year's successes and conducting a critical analysis to identify areas for improvement, the team has developed the mechanical design of Vidyut. This design incorporates a novel trailing arm suspension system, hub motors for the drive train, and optimized cooling systems, which have been designed, manufactured, and assembled in-house. The design process has considered several critical factors, including the vehicle's structural integrity, weight optimization and distribution, modularity, aesthetics, and compliance with the rules. These features are discussed in greater detail in subsequent sections.

3.2. *Drive System Analysis*

Vidyut employs a differential drive configuration. It consists of two caster wheels, one at the front and one at the rear, and two drive wheels in the middle of the platform. Each drive wheel is powered by a separate 350W BLDC hub motor coupled to an encoder, which is more than sufficient for the bot to maintain the speed limit of 5 mph.

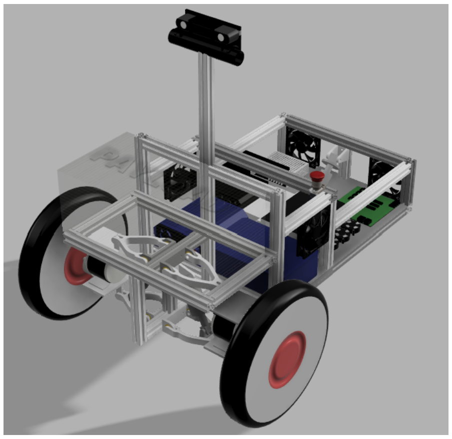

Figure 2: Vikram (2023)

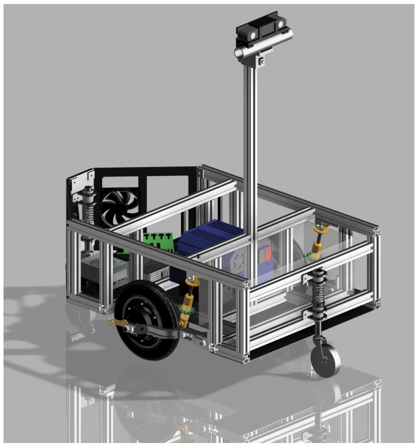

Figure 3: Vidyut (2024)

Motor and Encoder Assembly: The BLDC hub motors used in Vidyut provide a space-saving, cost-effective, lightweight, and easily mountable solution. They reduce mechanical losses compared to the Brushed DC Motor used in the previous vehicle, which decreases mechanical failure points and enhances vehicle power efficiency.
The compact design of hub motors allows the wheel to be positioned in the middle of the body, reducing the turning space by 34% compared to previous vehicles with front-wheel drive.
Moreover, the BLDC motor has been modified and fitted with a 3D-printed helical gear to enhance the reliability of the sensing.

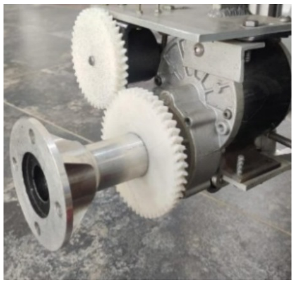

Figure 4: Vikram Drive

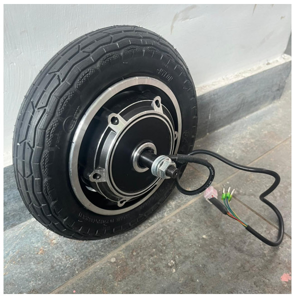

Figure 5: Hub motor

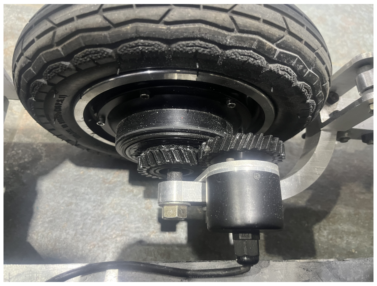

Figure 6: Vidyut drive

3.3. ***Suspension and Articulation***

The achievement of a stable camera feed in our previous vehicle has inspired us to enhance our current project by introducing independent suspension systems for both the drive and support wheels. This development aims to enhance the vehicle's performance, stability, and functionality. By incorporating independent suspension, we expect improved ride quality, increased traction, and better control, setting a new benchmark for maneuverability and efficiency in our vehicle design.

1. **Drive Wheel Suspension**(Enhanced suspension system for precise navigation): In response to the limitations of last year's double wishbone suspension system, which effectively reduced vibration but proved ineffective in addressing undesirable camber-related effects on our odometry-based localization.
Due to camber, our encoder reading was affected, as given by the below relation:

$$\theta_{normal} = \frac{(\omega_R - \omega_L)R_{wheel}}{L}t \ , \ \theta_{tilt} = \frac{(\omega_R - \omega_L)R_{wheel}}{L + R_{wheel}\sin\alpha}t$$

So, we designed a trailing arm-inspired suspension design. This adjustment has not only preserved the notable vibration reduction capability but also provided a more compact, lightweight, and adaptable solution for our differential drive robot and has effectively resolved the camber-induced inaccuracies in localization.

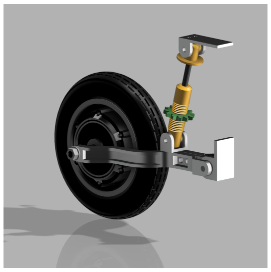

Figure 7: Trailing Arm Suspension

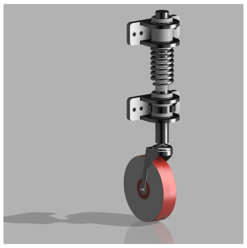

Figure 8: Caster Suspension

2. **Caster Wheel Suspension:** Uniform weight distribution needs a caster in both the rear and front for better stability. So, two wheels and a simplified suspension design instead of a wishbone make it easy to assemble without compromising vibration issues.

3.4. ***Chassis Design***

1. **Material Selection and Modularity:** The chassis must be strong enough to support the entire load yet light enough for better maneuverability. Hence, the aluminum frame uses highly modular standard extrusion bars. They enable efficient assembly, disassembly, and compact packaging to aid our logistics.

2. **Space Efficiency and Weight Distribution:** The strategic placement of the payload compartment at the same elevation as the component bay, unlike the previous design, where it was positioned above the bulky suspension, has effectively lowered the vehicle's center of gravity. This design improvement eliminates potential toppling tendencies when navigating ramps or inclines, enhancing the overall running stability. Proper weight distribution is crucial for vehicle stability, and by positioning the center of gravity close to the middle and in line with the drive axis, the design has optimized traction at the drive wheels, further contributing to stability and maneuverability.
3. **Vehicle Enclosure and Platform:** The body panels are made using **acrylic sheets**, which were easier to fabricate and gave our bot an excellent finishing look. Due to their low weight and high strength, the base plate, designed to withstand all electrical components, is made of **Fibre Reinforced Plastic (FRP)** sheets. The FRP sheets are fabricated with glass fiber mats and polyester resin. The acrylic and FRP sheets are precisely cut and assembled by the team members.
4. **Adjustable Positioning for Stereo Camera:** A simple yet innovative stereo camera mount allows for variations in pitch and elevation (2 degrees of freedom) from the ground if required. Last year's model inspired this design and has been further improved. An additional mount for a LiDAR sensor has also been incorporated into the camera pole.
5. **Weatherproofing:** The panel gaps and edges have been equipped with rubber seals and foam tape to mitigate water ingress. Additionally, 3D-printed fan covers have been implemented to minimize water infiltration through the cooling fans.

3.5. *Smart Cooling System*

The Jetson Orin is a highly efficient unit that can generate significant heat. Other heat-sensitive components in the bot, such as the Li-ion battery and the Power Distribution System, also require effective cooling.

The intelligent cooling system includes a custom fan controller developed and interfaced with the ROS stack. This controller actively monitors the ambient temperature inside and outside the bot, as well as the surface temperature of the Power Distribution board. The fan speeds are adjusted as needed to cool the target locations as calculated.

Additionally, **the outlet fans** were considered to be redundant and a waste of power, so **air vents** were used instead, which proved to be effective according to the **CFD simulation**. See 8

4. ELECTRONICS AND POWER DESIGN

4.1. *Overview*

This year's design philosophy emphasizes efficient performance and reliable operation through extensive testing and simulations for all critical systems.

Some of the highlights of this year's design are:

1. Active Battery Monitoring System incorporated with Power Distribution System
2. Capacitive Proximity Sensing based Emergency Stop
3. Automotive Ethernet

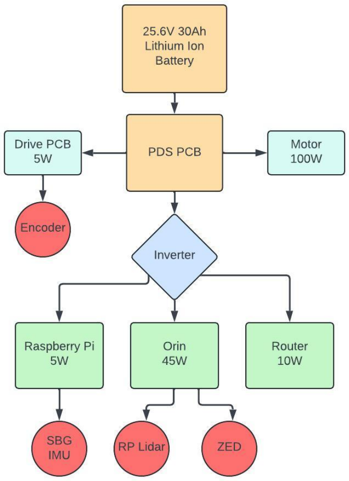

Figure 9: Power flow

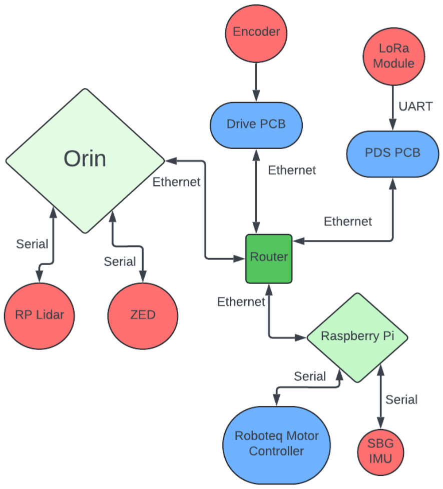

Figure 10: Communication flow

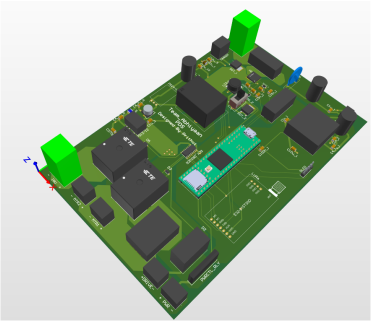

Figure 11: 3D model of PDS

4.2. *Power Distribution System*

Building up on last year's Battery Monitoring System, is the Power Distribution System. It handles the power supply to all parts of the bot, monitors the battery SoC, and houses the LoRa E-Stop. It also provides overvoltage, undervoltage, and overcurrent protection.

This year's Power Distribution System has two major advantages:
Individual control of power to each electrical subsystem which results in simpler electrical wiring in the vehicle.

Coulomb counting is performed using a Hall Effect-based Current Sense IC(TLI4971) instead of last year's LTC4151 to calculate the State of Charge of the Battery. It offers the following benefits:

LTC 4151	Hall Effect
Measures the voltage drop due to the current across a small resistance.	Measures the magnetic field generated by the current flow.
Lower sampling rate of 17 Hz.	Higher sampling rate, up to 5000Hz.
Digital signal with 12-bit accuracy.	Output is a differential analog signal, which is less susceptible to noise.
The resistance introduces a small voltage drop in the supply voltage.	Measurement is isolated and does not interfere with the current flow.
Only senses current in one direction, requiring other workarounds to calculate SoC.	The Hall Effect can sense current flow in both directions, which is helpful for calculating SoC.

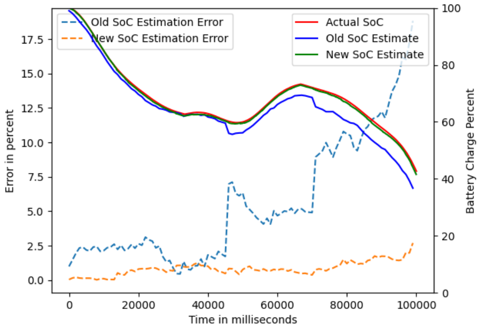

Figure 12: SoC calculation with different sampling rates

This PCB is at high risk of heating due to the high current flow in it. For this, a surface temperature measurement IC is used as feedback to control fan speed.
Fig:12 Compares the actual and estimated SoCs as well as the absolute percentage error in the estimated SoCs between the old and new current sense modules.
The plot shows how the SoC calculation accuracy has improved due to our increased sampling rate. This is because small surges and spikes in current might not be picked up, and this error accumulates over time.

4.3. *Motor Control, Drive PCB & Encoders*

To control our Brushless DC Hub motors, we use the Roboteq SBLG2360T motor controller with many useful features like Undervoltage Protection, Overvoltage Protection, Configurable Current Limits, Configurable RPM and Acceleration Limits, Anti-cogging for smoother commutation at lower speeds, etc.
The motor controller is directly connected to the Raspberry Pi through an isolated USB connection and communicates with the ROS stack through an in-house developed driver that sends commands to the controller and gets important data like RPM reading of the integrated Hall sensor and current consumed by each motor.
For wheel feedback, we have two sources:

1. BLDC Internal Hall Sensor
   (a) Speed is determined by measuring the time between Hall sensor transitions, which is very accurate, provided the sensor and motor are well-integrated.
   (b) The Hall speed measurement from Roboteq does not report wheel speeds below 20 RPM. We require a far higher resolution for our control loops.
2. Quadrature Encoder

(a) The encoder operates at 12V, so the Signal-to-Noise ratio is higher than standard 5V encoders. The Drive PCB isolates the signal and steps it down to microcontroller voltages.

(b) The Drive PCB is designed in-house to power the 12V encoder, acquire and process the encoder signal, and send Odometry data to the RPi through Ethernet.

A count of N pulses from the encoder actually implies that the wheel rotated somewhere between N to $N + 1$ samples in that time frame. This results in a $\frac{1}{N}$ error in the measurement of velocity from encoder pulses. Fewer pulses are counted at lower speeds in every interval, implying a higher error. Here are two methods to resolve this issue:

1. **Dynamic encoder frequency:** The frequency of encoder data collection would be inversely dependent on the vehicle's speed. This would ensure that the % error in the velocity calculation is kept constant. Since there is no requirement for such high precision, a simple binary model, wherein at speeds below 0.5 m/s, we step down the encoder frequency to 50Hz, and at higher speeds, we maintain the standard 100Hz sampling rate.
2. **Mixed mode Frequency/Period measurement:** A mixed mode measurement is used, where the number of pulses in a given time period is measured just like in traditional frequency measurement methods, but it also measures the time between the last pulse of the period and the end of the period and uses this to reduce the error in the reading.
   Referred from here.

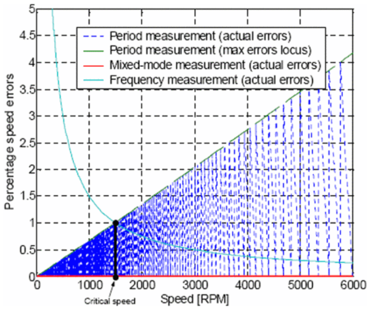

Figure 13: Comparison of Encoder Algorithms

Our Control algorithm is a classic PID controller running on the Host. Our testing has shown that the PID responds well to disturbances like inclines of up to 35 degrees, road surface changes, and minor load changes.

4.4. *Automotive Ethernet*

This year, we decided to use Ethernet to communicate between our microcontroller and the RPi, replacing last year's Serial-through-USB communication. This change was because we noticed encoder data being dropped occasionally, even at the highest baud rate allowed by our hardware.

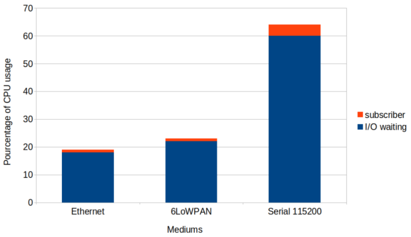

Figure 14: Microcontroller CPU Usage v/s Transports

We use **Micro-ROS** to transmit feedback data from the Drive PCB to the host ROS2 nodes. This made it easy for us to switch to Ethernet, as Micro-ROS supports Ethernet as a transport natively for some microcontrollers.

On the Microcontroller side, most of the CPU time is used for I/O operations, as seen in the graph below. As Ethernet is theoretically about **100x faster** (100 Mbps v/s Serial's 921600 baud (1Mbps)) the CPU spends less time on the I/O and has more free time to perform other tasks.

The CPU usage in the Host PC has reduced by a significant amount, as seen in our Systems Test 9.

4.5. *Safety & Protection Mechanisms*

4.5.1. *Encoder Redundancy*

The Quadrature Encoder is the main feedback to the velocity control algorithm. However, an issue was faced last year where the encoder gear disengaged, resulting in uncontrolled behavior.

To mitigate this issue, the motor driver also gets the wheel RPM reported by the inbuilt Hall sensors of the BLDC motors and publishes it to the ROS stack. This data is compared with the Encoder data, and if a significant difference is observed between both data points, the vehicle is brought to a safe stop, and the error is logged and raised.

4.5.2. *LoRa Mesh Network for Emergency Stop*

The LoRa based Emergency Stop has been implemented on the Power Distribution board. It handles the Emergency Stop and the wireless periodic Heartbeat to the vehicle. On losing the heartbeat (when the remote moves out of range or likewise) the bot comes to a swift stop.

The range of a point-to-point LoRa connection with line of sight has been tested to be about 4km. To increase the effective range and reliability of connection in non-ideal conditions like loss of line of sight, extreme weather, etc. an in-house LoRa based mesh network has been designed that uses a modified version of the Destination Sequenced Distance Vector (DSDV) routing algorithm with address filtering, encryption and ACK messages. In case the remote is far from the vehicle itself but is within the range of the mesh network, the message will be transmitted through the mesh network nodes to reach the destination.

4.5.3. *Capacitive Proximity Sensing based Collision Avoidance*

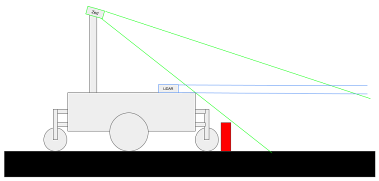

Figure 15: Side view of Field of Vision of Sensors on Vidyut

The LiDAR and Zed camera have a minimum sensing distance of 0.15m and 0.2m respectively. From fig:15, it is evident that our sensor placement results in a blind spot right in front of the vehicle. The problem with sensors like LiDAR's, Radar's and Ultrasonic Sensors is that they operate from a point and detect objects in Line of Sight. The requirement is for a sensor that can detect any object near the surface of our bot.

For this purpose, we use a capacitive sensor. It measures the capacitance between the surface of the bot and any nearby object. When this value rises above a tolerable limit, an emergency stop is issued, and the vehicle is brought to a halt.
Humans and metals have very high dielectric constants. This makes it an excellent technique to prevent collisions with human beings and other vehicles when on the road. The distance and shape data from the wave-based sensors can be combined with capacitive data to determine the dielectric constant of the object.

5. DESCRIPTION OF SOFTWARE STACK AND ARCHITECTURE

5.1. *Introduction*

Software architecture is structured around three key modules: perception, mapping and localization, and path planning. Recently, our team migrated from **ROS to ROS 2** as it has better real-time performance, and enhanced security and unlike ROS, does not have a single point of failure. This year, **LiDAR** technology and mapping capabilities were integrated into our software stack. To facilitate testing, augmented reality was used to simulate obstacles.

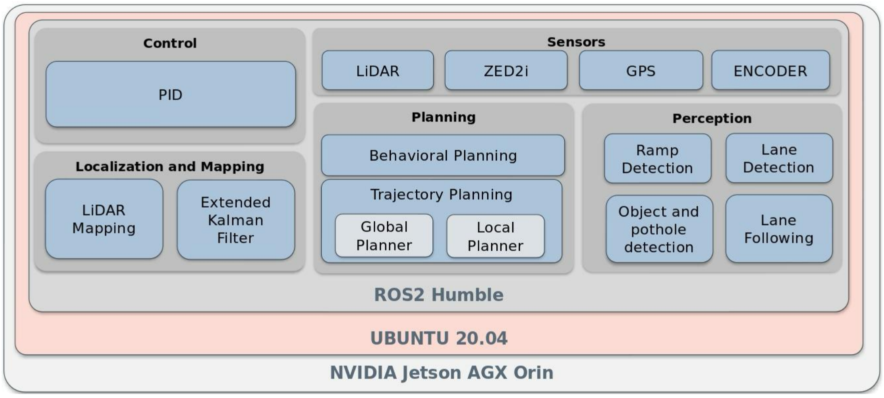

Figure 16: Software Architecture

5.2. *Obstacle detection and avoidance*

5.2.1. *Sensors*

- **RPLiDAR A1:** It is used to detect objects and can estimate the distance of the obstacles. This is used for map generation and costmaps.
- **Zed 2i Stereo Camera:** It provides RGB, depth image, and point cloud stream, mainly used for lane detection. It comes with an inbuilt **IMU**, which is used to navigate the ramp.
- **Septentrio Mosaic-X5 Development Kit:** Provides satellite-based location data for accurate geospatial positioning and helps us navigate the waypoints.

5.2.2. *Ground Layer Perception*

The LaserScan data obtained from the RPLiDAR and PointCloud data generated after performing Inverse Perspective Mapping(IPM) on detected potholes and lanes are processed in the following layers:

Obstacle and Lanes Layer	Potholes Layer	Inflation Layer
Lanes detected from the perception stack are converted to LaserScan to merge with LiDAR data to facilitate mapping.	Potholes detected through the stereo camera are integrated into the costmap.	This layer adds additional cost around the obstacles to maintain safe clearance.

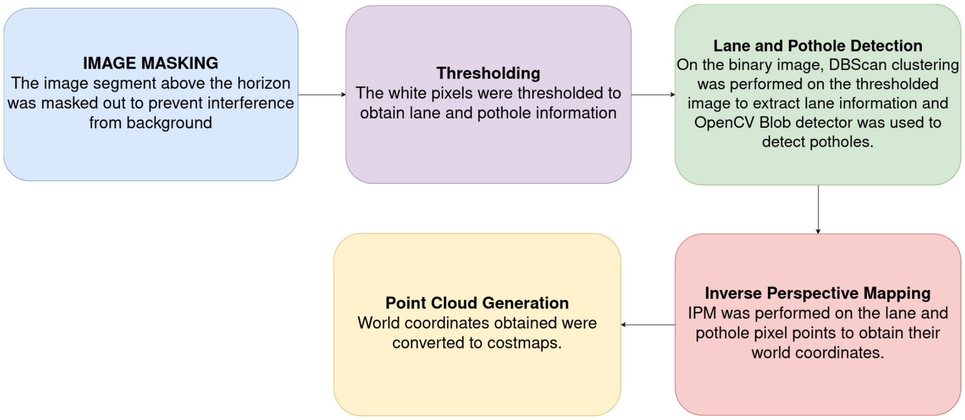
IMAGE MASKING
The image segment above the horizon was masked out to prevent interference from background
Thresholding
The white pixels were thresholded to obtain lane and pothole information
Lane and Pothole Detection
On the binary image, DBScan clustering was performed on the thresholded image to extract lane information and OpenCV Blob detector was used to detect potholes.
Inverse Perspective Mapping
IPM was performed on the lane and pothole pixel points to obtain their world coordinates.
Point Cloud Generation
World coordinates obtained were converted to costmaps.

Figure 17: Perception Pipeline

A costmap is an occupancy grid map where each cell is assigned a cost based on the probability of an obstacle occupying that particular cell. Inverse Perspective Mapping(IPM) plays a key role in generating costmaps of lanes and potholes. It is used to transform the detected potholes and lanes to the world coordinates.

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \frac{h}{\mathbf{n_c}^T \mathbf{K}^{-1}(u, v, 1)^T} \mathbf{K}^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$$

where (X_c, Y_c, Z_c) is the position vector in world coordinates, (u, v) is the image pixel coordinate, h is the height of the camera, K is the intrinsic camera matrix, and n_c is the normal to ground plane matrix.

5.3. ***Mapping and Localization***

Vidyut relies on highly detailed and up-to-date maps to understand its position and plan routes. Simultaneous Localization and Mapping (SLAM) techniques are used to build and update maps in real-time while simultaneously determining the vehicle's location within the map.

5.3.1. *Map generation*

To generate the map of the environment, two types of SLAM Algorithms were tested and implemented:

1. **LiDAR-BASED SLAM:** It uses laser beams to generate highly accurate 2D maps of the surroundings, while simultaneously locating the position of the sensor. This technology is highly effective in environments where GPS signals are weak or non-existent.

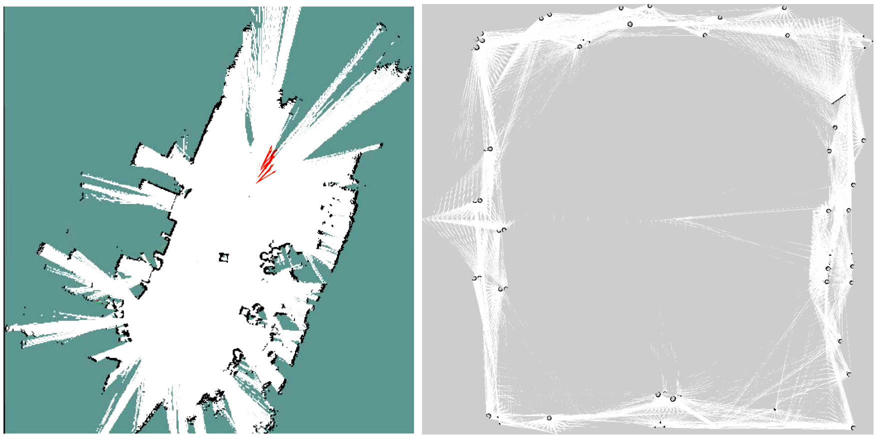

Figure 18: LiDAR Mapping (a) In our workspace (b) In Simulation

2. **vSLAM:** Visual simultaneous localization and mapping refers to the process of calculating the position and orientation of a camera, with respect to its surroundings, while simultaneously mapping the environment. The process uses only visual inputs from the camera.

Due to the simplistic nature of the obstacle course, LiDAR was chosen for mapping.

5.3.2. *Localization*

Extended Kalman filtering is implemented to integrate sensor data for precise state estimation. Through this process, both global and local odometry are refined by combining encoder odometry, GPS odometry, and IMU data. By fusing these sources of information, the filtering technique enhances the accuracy and reliability of the robot's odometry measurements, enabling more robust navigation and localization capabilities.

5.4. ***Software Strategy and Path Planning***

Our software strategy for Vidyut revolves around a modular architecture that helps us organize and manage the flow of data efficiently.

5.4.1. *Behavior Trees*

A behaviour tree is a tree of hierarchical nodes that control the flow of decision-making. Behavior Trees are leveraged to make the system more modular and reusable, utilizing Nav2's architecture, based on the BehaviorTree CPP V3 library. The tree has the following main segments:

1. **Navigation with Replanning:** This sequence handles continuous path planning and following. It includes selecting controllers and planners, replanning at a rate of 1/3 Hz, and executing the path through selected controllers. If a planning or execution failure occurs, specific recovery sequences activate.
2. **Recovery Sequence:** This fallback sequence activates if both controller and planner recoveries fail. It involves checking for updated goals, performing environmental clearing, spinning, waiting, and backing up as sequential recovery actions.

5.4.2. *Lane Detection*

Lane detection was approached in two different ways.

1. **Machine Learning:** Used YOLOP for lane detection to classify the lanes. Binarizing the output extracts lane pixel information. DBSCAN clusters lane pixels based on spatial density. Then, polyfit is applied to calculate coefficients for each curve, which are published as a point cloud and added to the global map.
2. **OpenCV:** The goal was to create lightweight code for fast and efficient lane detection. Image thresholding is done to obtain a binary image with white pixels representing lane markings. DBSCAN is implemented to form clusters of white pixel coordinates. Only the largest two clusters, assumed to be the lanes, are displayed on the binary image.

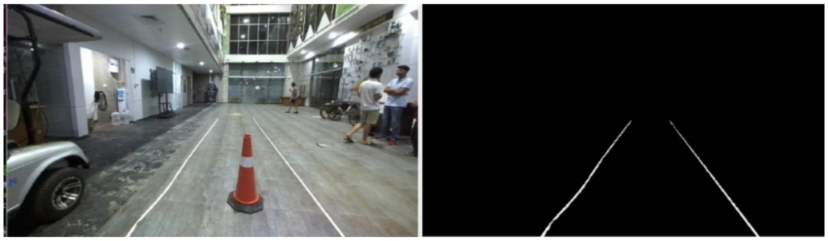

Figure 19: Lane Detection using OpenCV

OpenCV was preferred over Machine Learning for Lane Detection due to its lightweight nature, offering a more efficient solution compared to machine learning algorithms, which often demand substantial GPU processing power.

5.4.3. *Lane Following*

Lane points are located by initiating a search from the centre of the binary image obtained after lane detection and utilizing OpenCV's connected components method to track them. The midpoints of these lane points are chosen as the required point and converted to world frame using IPM. When only one lane is detected the point with an offset of 15 cm to the lane is chosen as the required point. These coordinates in the world frame are then transmitted as Nav2 goals, enabling Nav2 to devise paths while taking obstacles into account.

5.4.4. *GPS Waypoint Navigation and Path Planning*

Vidyut autonomously navigates through a sequence of GPS coordinates by converting them into world coordinates and establishing them as goals. Upon receiving each goal, it plans a path using both global and local planners.
A highly optimized fully re-configurable grid-based **A*** implementation is used for our global planning. It is slightly slower but provides increased quality in paths. For local planning, we use **DWA(Dynamic Window Approach)** algorithm which finally produces velocity commands to move the robot. When the vicinity(2 meter radius) of a waypoint is reached, the subsequent waypoint is passed as the next goal, and this iterative process continues until the final waypoint is reached. The bot can arrive at its goal within an accuracy of 15 cm.

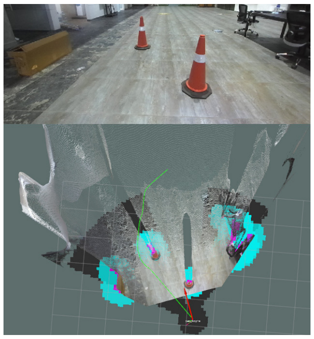

Figure 20: Path Planning

5.4.5. *No Man's Land and Ramp Strategy*

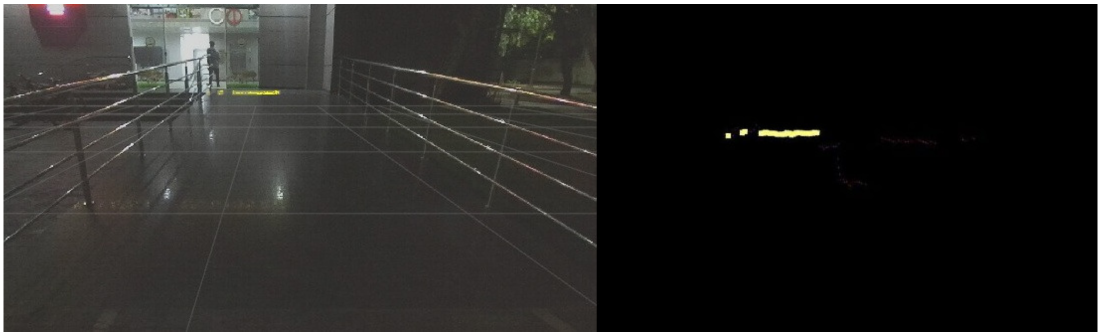

Figure 21: Ramp Detection

In No Man's Land, Vidyut depends entirely on GPS waypoints for navigation. Ramp can be detected from LiDAR data as it will be a straight line in it. The implementation of the Line extraction algorithm uses a split and merge technique to find distinct lines from the lidar data. Since the length of the ramp is known, a length-wise thresholding is done to detect the ramp. As Vidyut starts climbing the ramp, the LiDAR system is deactivated. The LiDAR system is reactivated only after Vidyut completes the ramp, a transition detected by the IMU. This deactivation is essential because the LiDAR misinterprets the ground as an obstacle when it is at an angle. The climb is detected using an IMU which tells the angle of the bot with respect to the ground plane.

5.5. ***Augmented Reality for Vehicle Testing***

Since a testing facility to carry out testing was not available, in order to help us with this problem, the development of an AR obstacle course that can be developed in an empty space was started. AR obstacles and lanes course are created, harnessing augmented reality technology's power to replicate real-world scenarios, and providing a safe environment for testing autonomous vehicles without risking damage. There are three different approaches to the AR course. They are:

- **Unity with ROS 2**: Built the obstacle course in Unity game engine, and used the ROS_TCP_Endpoint and Connector to integrate Unity with ROS 2 to get AR projected images on a ROS2 topic. This image is passed to a stereo depth estimation model to get a depth image which is then processed into a point cloud.

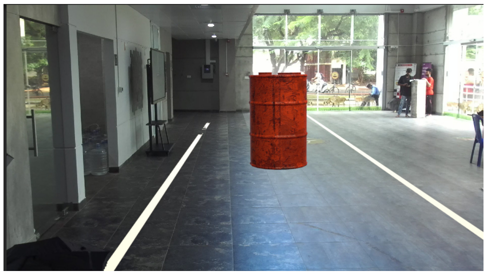

Figure 22: AR Barrel and Lanes in Unity

- An **OpenCV model was used to detect ArUco markers**, and the intended homography matrix is used to project 3D objects into the image. This then is passed to a depth estimation pipeline.

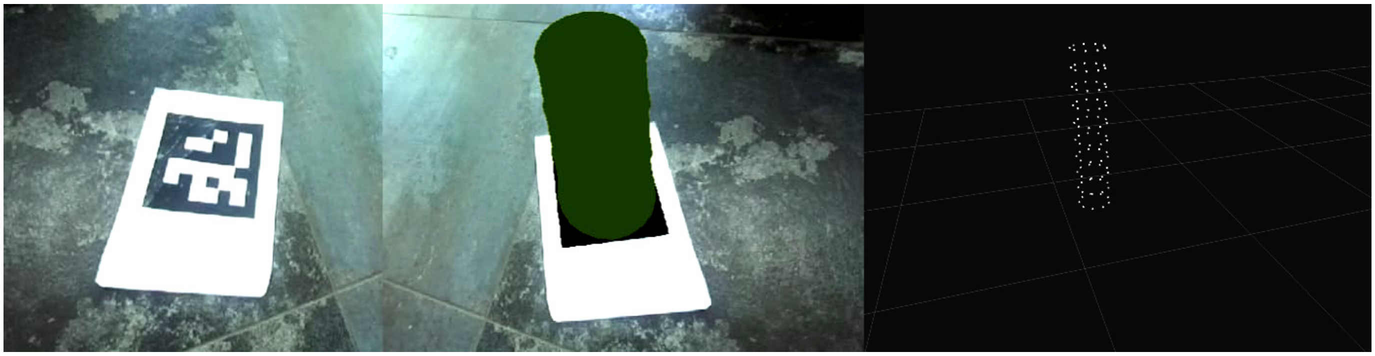

Figure 23: AR Barrel with OpenCV

- This is similar to the above method but instead of projecting a 3D object into the image, the model finds the midpoint of the ArUco marker, performs IPM on this point and **directly publish a cylindrical pointcloud** centred around this point.

5.6. *Computation Unit*

NVIDIA Jetson Orin is used as the compute unit, which is the industrial-grade unit for robotics applications because of its low power consumption, CUDA based on Ampere architecture and 10 GB/s ethernet. cuDNN, which is a GPU-accelerated library of primitives for deep neural networks, is used.

5.7. *Parallelization with GPU*

The GPU in Orin allows us to parallelize our software stacks, such as depth image to point cloud and IPM. The CUDA runtime API is used to communicate and manage processes in our GPU. This provides us with a significant reduction in the processing of depth images (roughly around 300 times than serialized code).

6. CYBER SECURITY ANALYSIS

6.1. *The NIST RMF Process*

Most security issues are caused by negligence on the developer's part. To prevent this, a systematic process should be followed to ensure security before deploying the product. Using NIST RMF the amount of risk that the system can tolerate is analyzed and appropriate protections are enabled.

6.2. *Implementing the NIST RMF Process*

6.2.1. *Modeling and Analyzing the threats (Prepare and Categorize)*

The case of a rival team trying to disrupt our robot in the pit area is considered. The functional requirements and the information that needs to be used for proper functioning and their impact if confidentiality, integrity or availability is compromised are as follows.

Kind of information	**Confidentiality**	**Integrity**	**Availability**	**Description of threat**
Cryptographic keys	High	High	High	Access to cryptographic keys might lead to swapping of components with malicious ones.
Bot Source code	Moderate	High	High	If the integrity of code is violated, malicious code execution can be achieved. This results in a total takeover of the bot.
E-Stop	Low	High	High	Malicious actor might be able to stop the bot at any moment and end a run if integrity is compromised, or endanger the bot if availability is compromised.
Odometry Data	Low	High	High	Inaccurate odometry data would affect localisation and path planning making it behave in unpredictable ways.
GPS Data	Low	High	High	Lack of GPS data will hamper the bot from being able to go through the No Man's Land.

6.2.2. *Selecting and Implementing the Cyber Controls*

IA-3 (Device Identification and Authentication)

Implementation	**Testing strategy**	**Information Protected**	**Threat Eliminated**
Authenticated communication between the Microcontroller (Teensy) and Computational Unit (Jetson Orin) is established by a secure 3-way handshake protocol.	On connecting a rogue Teensy, the Orin throws an error and does not further interact.	Microcontroller Authenticity	Hackers with physical access cannot easily replace the Teensy to give incorrect data and manipulate the bot into unexpected actions. Replay attacks can be eliminated.

SC-8 (Transmission Confidentiality and Integrity)

Implementation	**Testing strategy**	**Information Protected**	**Threat Eliminated**
All communication from the microcontroller to Orin and from the E Stop remote are encrypted and authenticated by HMACs with pre-shared keys. Rolling keys are used.	It can be seen that any attempt to change the data in the message packet will fail as the HMAC won't be correct.	Odometry data and E-Stop.	Hackers with physical access to the communication medium won't be able to perform a man-in-the-middle attack to send inaccurate encoder or odometry data.

AC-17 (Remote Access)

Implementation	**Testing strategy**	**Information Protected**	**Threat Eliminated**
Allow access to SSH using public keys of sufficient length.	Observe that trying to SSH into Orin using a password always fails.	Bot source code and Cryptographic Keys.	Hackers cannot access the Orin remotely by guessing passwords.

SC-13 (Cryptographic Protection)

Implementation	**Testing strategy**	**Information Protected**	**Threat Eliminated**
The SSD on the Orin is encrypted to prevent someone with physical access to the system from harvesting keys.	Boot up the Orin and observe that it asks for a decryption key.	Bot source code and Cryptographic Keys	Someone with physical access to the Orin cannot view code and data unless they have encryption keys.

SI-7 (Software, Firmware, and Information Integrity)

Implementation	**Testing strategy**	**Information Protected**	**Threat Eliminated**
By enabling the Secure mode and disabling the JTAG port on Teensy we ensure only the code that is encrypted by our key will run on it. The flash memory is also encrypted	Observe that any code upload to the Teensy fails unless it is correctly signed. Also, observe that the device won't be identified in openOCD.	Odometry Data, Bot source code and cryptographic keys	Someone with physical access to the microcontrollers cannot upload rogue code and mess with the data sent to the computing unit. Flash cannot be read to expose secrets.

7. DESCRIPTION OF FAILURE MODES, FAILURE POINTS, AND RESOLUTIONS

7.1. *Mechanical Failure Modes & Resolutions*

Failure point	**Cause**	**Resolution**
Undesirable lateral movement in the trailing arm	Improper tolerancing of bearing housing	Change dimension of bearing housing
Loosening of the linear bearing of the caster wheel suspension	Loosening of circlips	Press Fit linear bearing in the corrected bearing housing along with circlips
Loss of traction while climbing ramp	Insufficient length of spring strut and interference of encoder and battery compartment	Increase the length of the spring mount Reduce the size of the battery compartment

7.2. *Electrical Failure Modes & Resolutions*

Failure Point	**Cause**	**Resolution**
Power Surges damage connected components	Inrush currents and Electrostatic Discharge creates sudden voltage and current spikes	Power and signal isolators and fuses so that the discharge and inrush currents are not propagated.
Encoder gives wrong readings	Mechanical disengagement, or electrical damage to the encoder or PCB	Detect the error with the BLDC Hall sensor and stop the vehicle
Loss of connection with the Wireless E-Stop	Communication faults or going out of range of the transceiver	Heartbeat on the wireless E-Stop to ensure communication is always possible, and stop the bot if the connection is lost.
Sudden movement of the bot after the push and release of Estop	After the mechanical estop is pushed, the bot goes to rest, but the setpoint is still at a non-zero value	Heartbeat on the bot's velocity and its setpoints, which identifies the inconsistency in setpoint and velocity for a long enough time and publishes a software estop

7.3. *Software Failure Modes & Resolutions*

Failure point	**Cause**	**Resolution**
No valid path is found by Nav2.	This arises due to the large inflation radius of the costs.	Nav2 recovery behavior, which sets the bot to spin until a valid path is found.
Falsely detecting ground plane as an obstacle	While climbing the ramp, LiDAR detects the ground as an obstacle and gets incorporated in costmap	IMU is used to detect if the bot is on the ramp and then switch off the costmap from LiDAR.
Loss in a data stream	Unaccounted errors in code or sensor driver issues could lead to a certain node stopping publishing data	Heartbeat has been deployed to ensure appropriate measures are taken when a certain data stream is lost.

7.4. *All failure prevention strategy*

Heartbeat: The *heartbeat* is a master node that keeps a check on all other nodes that sense the bot's surroundings. These nodes are defined as ROS2 nodes (written in ROS2 Humble). The nodes process the bot's parameters like velocity, current drawn from motors, temperature, state of the estop, etc. Heartbeat issues warnings for non-fatal errors and stops the bot for fatal errors. This is done using a publisher-subscriber interface. In regular intervals, Heartbeat processes the current data for errors and warnings. It is a very modular code that can include any number of error checks.

8. SIMULATIONS EMPLOYED

1. Structural Analysis of the Suspension Components:

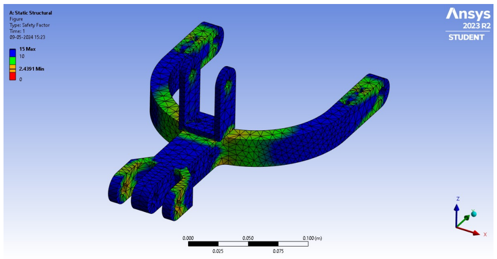

Figure 24: Static Trailing Arm Analysis

The Trailing Arm is an integral structural component of the suspension system and is susceptible to the highest loading. All structurally critical components have been developed with a minimum safety factor of 2 and optimized for overall rigidity. Finite Element Analysis tools have been used along with CAD Modeling software to design the optimized parts.

2. CFD Simulations for Cooling Systems:

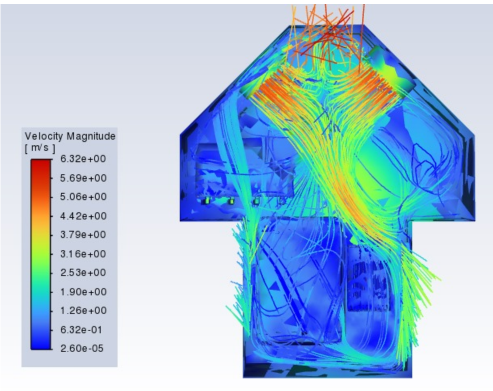

Figure 25: CFD simulation

Two high-volume flow rate axial fans (85cm each) have been used to circulate air inside the main compartment. Airflow was modeled using CFD analysis tools and simulations were analyzed for both fans having different speeds to visualize the airflow distribution in the vehicle. Airflow and components were placed accordingly, optimizing airflow requirements for each component and weight distribution of the vehicle. The system is also designed to have automatic control based on the vehicle's internal temperature and the surface temperature of the Power Distribution board. At peak fan speed, it is found to be able to dissipate about 100W of heat and lower the operating temperature of Orin Jetson by 4 °C. (Fig. 15)

3. PCB Current Density Simulation:

The Power Distribution board was designed with Altium Designer. As it is a board that handles high currents, the Power Analyzer extension of Altium was used to perform current density simulations.

The simulation enabled the identification of current hotspots, allowing us to place the Temperature sensor on the hotspot and design the board with appropriately sized traces and vias.

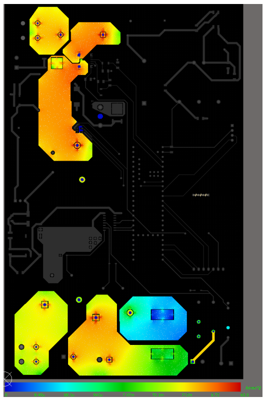

Figure 26: Top Layer

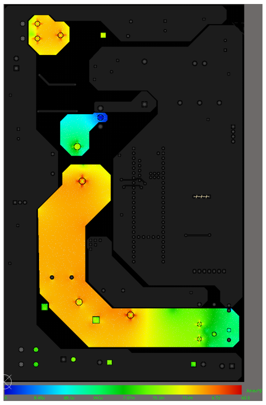

Figure 27: Bottom Layer

4. Integrated Simulation using Gazebo and RViz:

RViz is a 3D visualization tool that allows us to visualize our robot's model and sensor data in a simulated environment. Gazebo, on the other hand, provides a physics engine that complements RViz in simulating the physical properties of the robot model and its environment, enabling us to test and refine the control algorithms while simultaneously testing the ideal performance of Vikram in various scenarios. Combining these two allowed us to achieve a great head start on our first real-life prototype. Very few major changes were required on the actual vehicle because of extensive testing on the simulated environments.

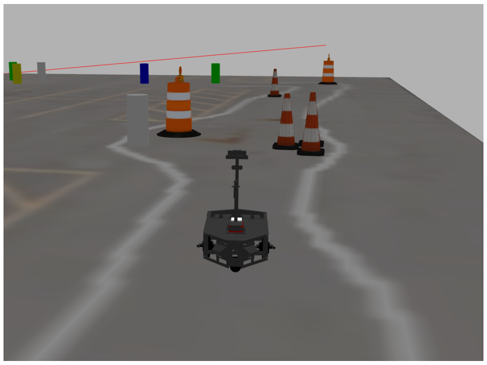

Figure 28: Ground view of the simulation

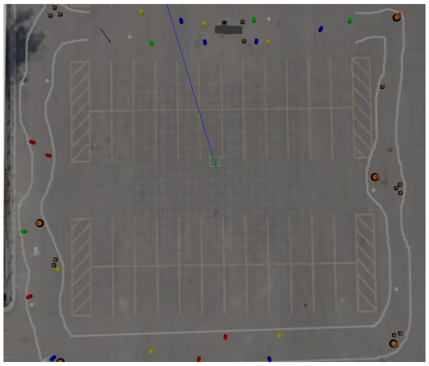

Figure 29: Top view of the simulation

9. COMPONENTS AND SYSTEMS TESTING

9.1. *Suspension Testing*

The vehicle underwent multiple repeated rumble strip tests, with and without a suspension system to prove improved stability. An evident difference in IMU readings was observed, with the flattening of jerk variations.
The graphed data below illustrates this contrast.

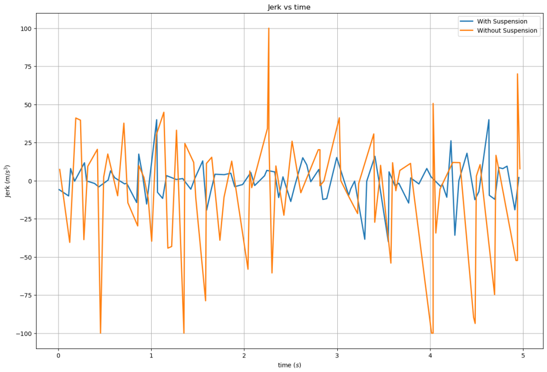

Figure 30: Jerk vs time

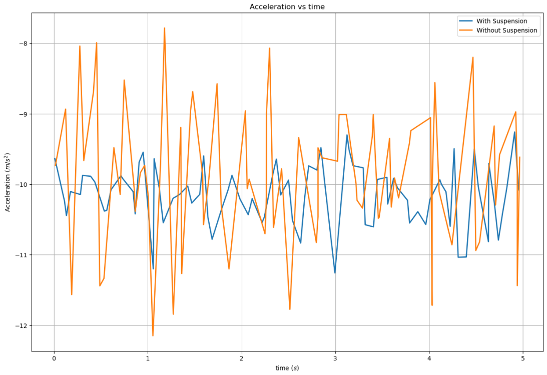

Figure 31: Acceleration vs time

An appreciable change in the standard deviation of the acceleration and jerk readings:

Evaluating Element	**Extreme analysis**	**Nominal Analysis**
Acceleration	1.15g	1.03g
Jerk	4.07g/s	0.92g/s

9.2. *PID Testing*

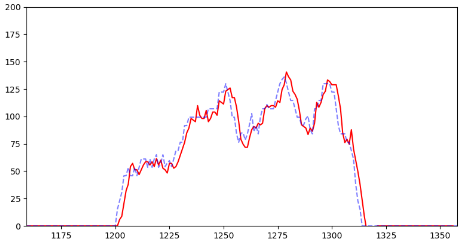

Figure 32: Speed Tracking Performance of PID

The observed wheel velocity is plotted against the setpoint. Under rapid changes to setpoints, the graph indicates a close following of the velocity with the setpoint. We recorded a starting acceleration of 0.82 m/s^2 and a stopping deceleration of 0.72 m/s^2. At a velocity of 0.5 m/s, we recorded a stopping distance of 0.13m. The average reaction time to reach set point is 0.7 secs

10. INITIAL PERFORMANCE ASSESSMENTS

Evaluating Element	**Extreme analysis**	**Nominal Analysis**
Computing Unit	45W	30W
Runtime	1.75 hours	3 hours
Motor Currents (Mean)	4A + 4A (5 mph)	2A + 2A (1 mph)
Motor Currents (Peak)	10A + 10A	6A + 6A
Recharge Time	3 hours	6 hours
Power Consumption	330 Watt	210 Watt

Part Name	**Price In USD**	**Part Name**	**Price In USD**
NVIDIA Orin	2,199.00	Roboteq SBLG2360T	625.00
RPLiDAR A1	101.00	Mean Well 24V 300W Inverter	79.99
Zed Camera	499.99	24V Li-Ion battery	419.99
Septentrio GPS module	723.92	Raspberry Pi 4 Model B 8GB	74.99
Router	49.99	Hub Motor x2	159.99
Autonics E50S8-1024-3-T-1 x2	99.99	**Total**	**5033.85**