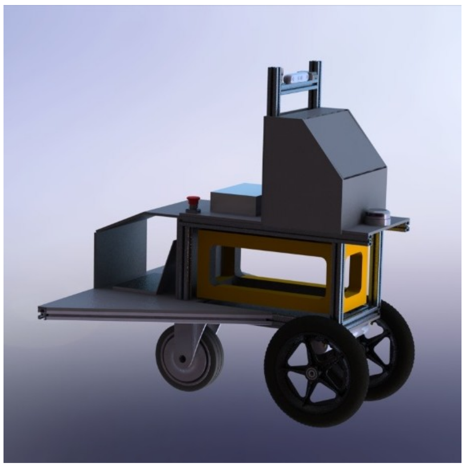
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur Village, Rohini
Delhi, India

UGV-DTU

ASHWINI



Date - 15 May, 2025

Lakshay (Team Leader)	lakshay_co21a4_72@dtu.ac.in
Kartik Walia (Team Captain)	kartikwalia_co22a4_66@dtu.ac.in
Vedant Singh	vedantsingh_23ce143@dtu.ac.in
Darshan Solanki	darshansolanki_23ce042@dtu.ac.in
Hriday Goel	hridaygoel_23en093@dtu.ac.in
Gatik Gupta	gatikgupta_23ce048@dtu.ac.in
Yuvraj Gupta	yuvrajgupta_23me298@dtu.ac.in
Akshat Rajwansh	akshatrajwansh_23ch008@dtu.ac.in
Theodore Regimon	theodoreregimon_23bt108@dtu.ac.in
Daksh Panchal	dakshshaileshpanchal_cs24a03_003@dtu.ac.in
Narayan Mishra	narayanmishra_en24b18_099@dtu.ac.in

I hereby certify that the design and development of the vehicle **ASHWINI**, described in this report is significant and equivalent to what might be awarded credit in a senior design course. This is prepared by the student team under my guidance.

[illegible]

Prof. S. INDU
Dean, Digital Education
Delhi Technological University
Shahbad Daulatpur, Bawana Road, Delhi-42

Prof. S. Indu
Faculty Advisor
Dean (Digital Education)
Delhi Technological University

1 Conduct of Design Process, Team Identification, and Team Organization

1.1 Introduction

UGV (Unmanned Ground Vehicle) is a team of passionate and driven students from Delhi Technological University, India, dedicated to developing autonomous ground vehicles for practical applications such as delivery systems and self-driving cars. Our goal is to conduct hands-on research in autonomous navigation and pass on this knowledge to our juniors to ensure long-term innovation and continuity. For IGVC 2025, we have built an autonomous ground vehicle named **'ASHWINI'**. It also originates from the Sanskrit word **'ashwa'**, meaning **horse**, representing **agility, strength, and readiness**. The name reflects the nature of our robot - agile, resilient, and engineered for intelligent performance in the AutoNav Challenge.

1.2 Organization

We believe that teamwork is at the heart of any great achievement. Our goal was not just to develop an autonomous vehicle but to build an environment which nurtures team and individual's growth simultaneously.

Name	Major/Year	Software	Mechatronics	Hours
Lakshay	CSE/4th	✓		450+
Kartik Walia	CSE/3rd	✓		400+
Vedant Singh	ME/2nd	✓		500+
Darshan Solanki	AE/2nd	✓		500+
Hriday Goel	EN/2nd		✓	400+
Gatik Gupta	AE/2nd	✓		450+
Yuvraj Gupta	EE/2nd		✓	450+
Akshat Rajwansh	ME/2nd		✓	450+
Theodore Regimon	BT/2nd	✓		400+
Daksh Panchal	CSE/1st	✓		500+
Narayan Mishra	EN/1st		✓	300+

1.3 Design Assumptions and Design Process

Our last participation in IGVC was in 2023. Since then, we have significantly enhanced the technological capabilities and reliability of our autonomous system. Drawing from past experiences, we embraced a pragmatic and iterative design methodology for the 2025 competition. This approach became central to our development strategy, guiding us to set achievable and effective design objectives rooted in real-world application and continuous improvement.

Prior to initiating the design and build phases, it was essential to establish a clear execution strategy encompassing every stage - from conceptualization to real-time testing. To achieve this, we adopted a star-model development approach. This model effectively outlines the interdependencies between each phase of development and its corresponding validation step, ensuring a structured and reliable workflow throughout the lifecycle of the project.

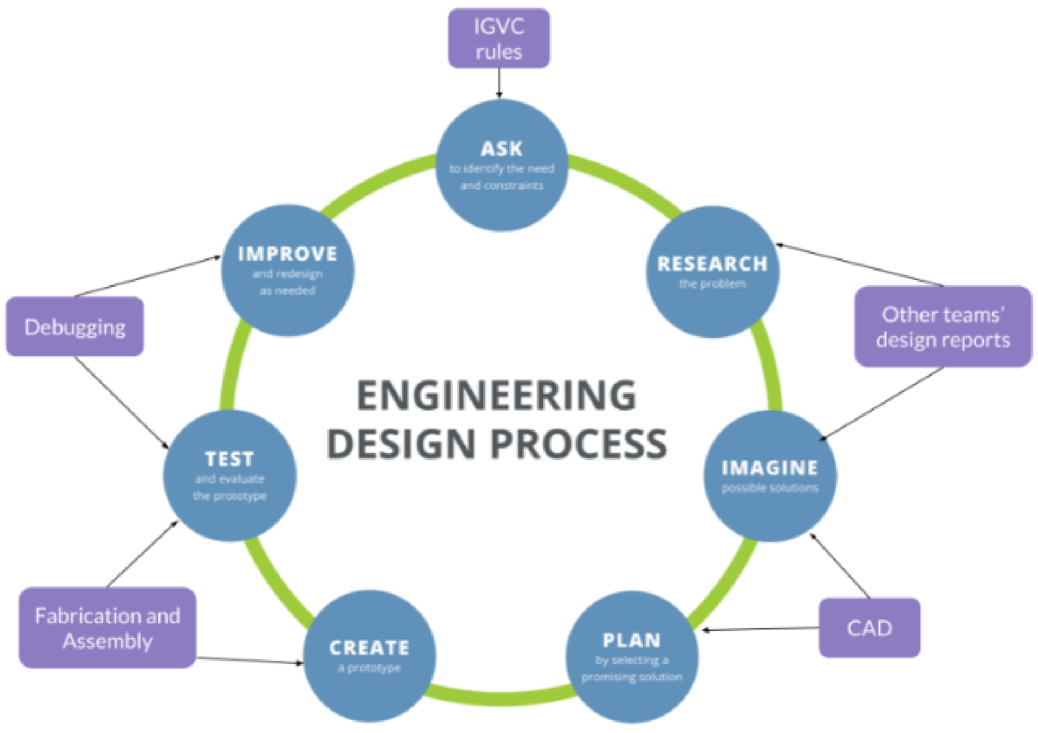

IGVC rules
ASK
to identify the need and constraints
RESEARCH
the problem
Other teams' design reports
IMAGINE
possible solutions
CAD
PLAN
by selecting a promising solution
CREATE
a prototype
Fabrication and Assembly
TEST
and evaluate the prototype
Debugging
IMPROVE
and redesign as needed
ENGINEERING DESIGN PROCESS

Figure 1: Engineering Design Process

2 System architecture of your vehicle

2.1 Identification of Mechanical, Power, and Electronic Components

Item	Specification	Quantity	Total Cost
Battery	Lead Acid 12V 7000mAh	2	35.74$
Lidar	RP Lidar AM31	1	875.37$
Motor Controller	Cytron MDD20A	1	40$
Motors	RHINO Mega Torque Geared DC Motor	2	100$
Camera	Intel RealSense D455	1	500$
GPS Module	u-blox C099-F9P-1 GNSS	1	389$
Wireless Transmitter and Receiver Module	TP-LINK CPE210	2	45$
Kill Switch	Kill Push Button Switch	1	2.43$
Status Light	NeoPixel RGB Led Strip	1	8.51$
Microcontroller	Arduino UNO	2	16$
Fan	Cooling Fan	1	2.43$
Buck Converter	Step Down Buck Converter	2	12.15$
Laptop	ASUS TUF Laptop	1	1000$
Aluminum T Slots	6061 Aluminum Extrusion	-	121.58$
Acrylic Sheet	Acrylic Sheets	-	48.62$
Wheels	Wheels	3	60.79$

2.2 Identify the Safety Devices

2.2.1 Emergency Stop System

Ashwini features both mechanical and wireless emergency stop systems. A physical E-stop button allows immediate shutdown, while a wireless LoRa-based system using Arduino and a relay enables remote stops from up to 1km away, ensuring reliable long-range safety.

2.2.2 Safety Indicator Light

Ashwini includes a visual safety indicator using an LED light strip. When powered on, the LED remains solid, indicating the system is active. During autonomous operation, it blinks continuously to alert bystanders and operators that the robot is navigating independently, enhancing situational awareness and safety.

2.3 Significant Software Modules and System Integration

Key software modules include ROS Noetic middleware, EKF and VIO for localization, gmapping for SLAM, LiDAR and RealSense for perception, move_base for path planning, motor control via Arduino, waypoint navigation, simulation with Gazebo, and safety through E-stop and LED nodes. ASHWINI's components are integrated in a modular architecture where sensors, actuators, controllers, and software communicate through a structured ROS-based framework. Here's how they interact:

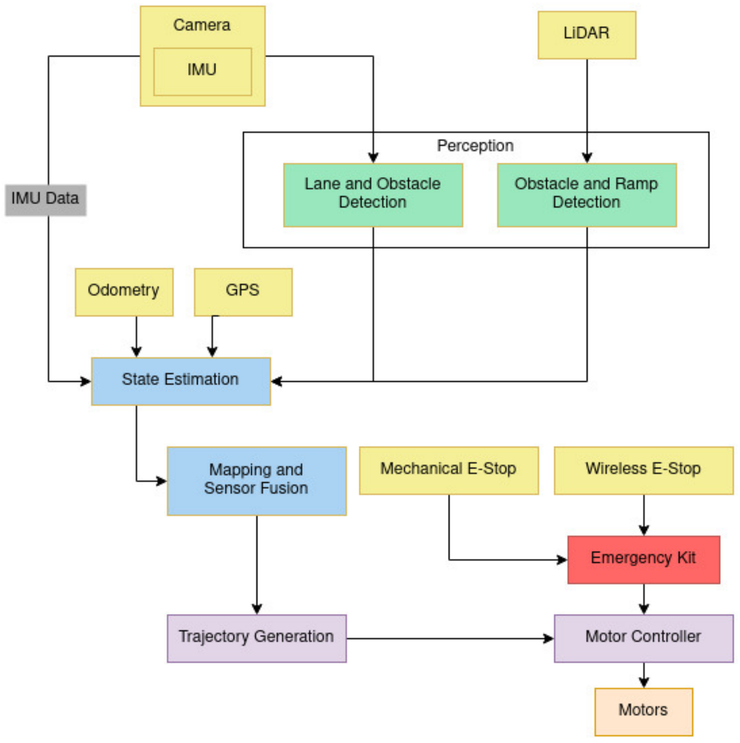


Figure 2: Vehicle System Architecture

3 Effective innovations in our vehicle design

1. **Complete Insulation:**
   All wiring is protected using heat-shrink tubing to prevent short circuits and electrical hazards.
2. **Organized Layout:**
   Clearly marked and simplified wiring ensures quick and error-free connections.
3. **Battery Optimization:**
   The system uses parallel cell configuration to expand battery capacity, allowing for extended operational time without recharging.
4. **Stability-Focused Design:**
   Heavier components are positioned at the base to lower the center of gravity and improve balance.
5. **Easy Maintenance:**
   External and internal lids provide fast access to electronics, motors, and batteries without disassembly.

6. **Adjustable Vision System:**
   The Intel RealSense D455 camera is mounted on a bracket allowing vertical and horizontal adjustment.
7. **Compact Footprint:**
   Designed slightly above the IGVC minimum size, the bot offers improved clearance and maneuverability.
8. **Dedicated Compartments:**
   Isolated sections for battery, payload, and electronics enhance safety and structural integrity.

4 Description of Mechanical Design

4.1 Overview

The mechanical module of **Team UGV-DTU** focuses on developing a robust, adaptable unmanned ground vehicle integrated with software and electrical subsystems. Building on last year's insights and identifying key improvement areas, the team designed and developed the mechanical structure of **Ashwini**. Its compact, 3-wheeled design enables sharp turns and improved obstacle avoidance. This year, the team drew inspiration from the Indian Air Force's Dassault Rafale, incorporating high performance, precision engineering, and aerodynamic efficiency.

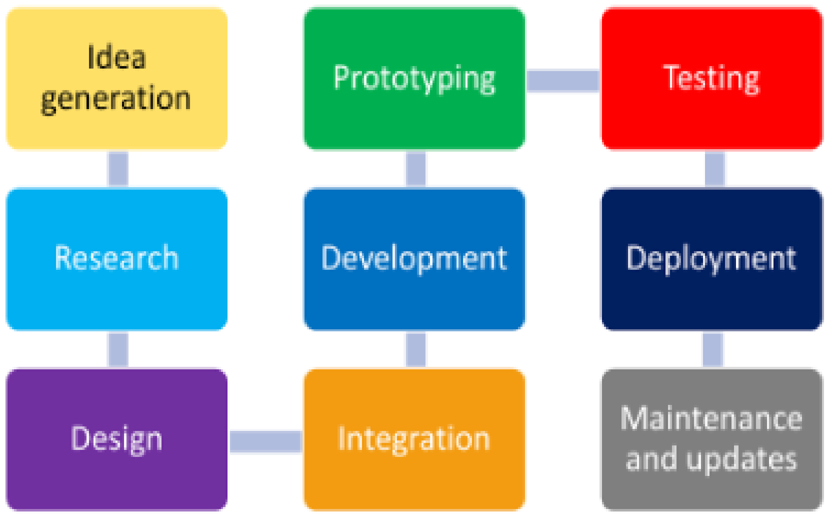


Figure 3: Bot Crafting Blueprint

4.2 Chassis Design

Ashwini's chassis is constructed from 3030 Aluminium 6061 T-slot extrusion, offering an optimal strength-to-weight ratio while ensuring durability. The modular assembly, secured with angle brackets, nuts, and screws, allows for easy disassembly, facilitating compact transport and avoiding excessive cargo charges. Designed to comply with IGVC 2025 regulations, the vehicle measures slightly larger than the minimum required dimensions (3 ft + 2.3 in length, 2 ft + 1.8 in width), maintaining a compact footprint for enhanced maneuverability at high speeds.
The structure is divided into three key compartments: a trapezoidal base housing heavy components like motors, gearbox, and battery for a low center of gravity; a cuboidal payload compartment; and a rear section for heat-sensitive electronics, ensuring thermal and electrical isolation. Since the competition terrain is primarily asphalt, Ashwini omits a full suspension system, instead using silicon rubber padding under critical components (motors, LiDAR, cameras) to dampen vibrations. This design prioritizes stability, weight distribution, and ease of maintenance while minimizing unnecessary complexity.

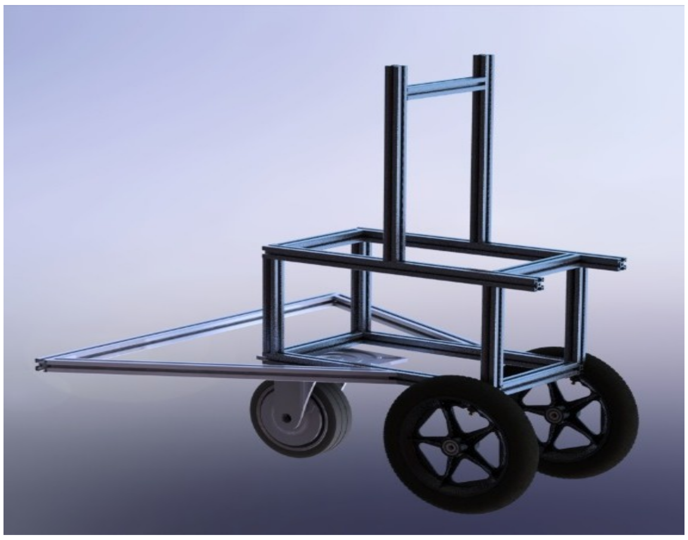


Figure 4: Chassis Design

4.3 Drive System Analysis

Ashwini uses a differential drive with two independently driven wheels and a rear castor wheel for passive support. This setup ensures excellent maneuverability and simplifies control, ideal for IGVC's obstacle-rich environment. Each drive wheel is powered by a RHINO High Torque Planetary DC Geared Motor, delivering high torque for challenging terrains.

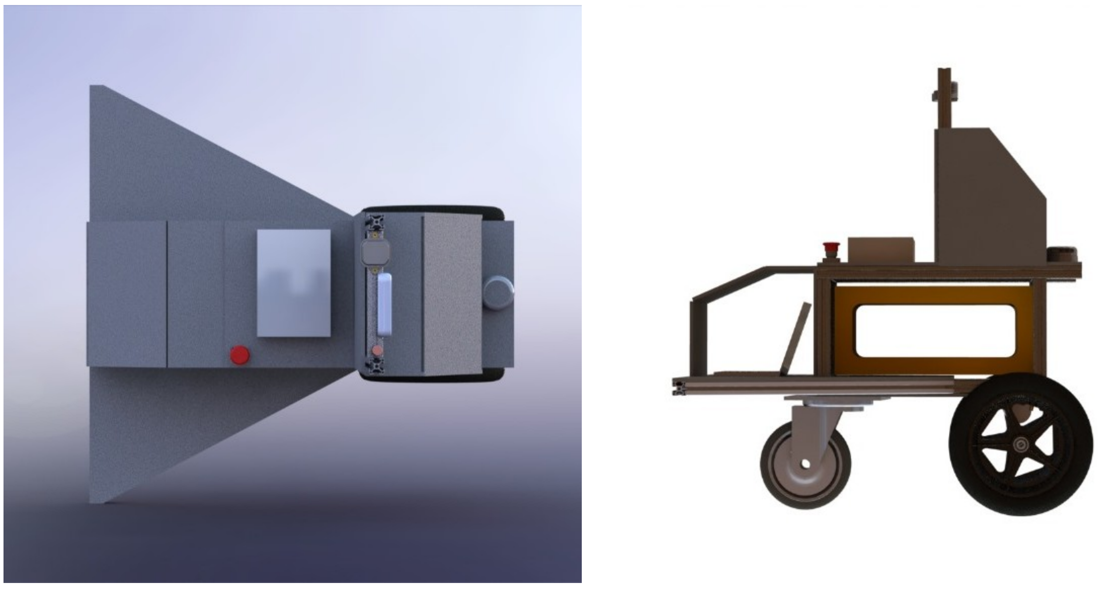


Figure 5: Top and Side View of Ashwini

4.4 Weatherproofing

The vehicle's outer body is made of 4mm acrylic sheets for their light weight, waterproof nature, and ease of fabrication. Since no load acts on the exterior, thicker material wasn't required. Joint gaps are sealed with waterproof rubber and acrylic tape to prevent moisture ingress and protect internal components.

4.5 Weight Distribution

ITEM	Total Weight (Pounds)	Weight Ratio (%)
Aluminum Extrusion	19.8	22.43
Motors and Gearbox	35	31.10
Wheels	12.1	11.40
Battery	2.9	2.73
Payload	20.5	18.85
Other Components	5.3	3.11
Outer Body	10.5	10.38
Total	**106.1**	**100**

Excluding the weight of Aluminium frame chassis which constitutes 22.43% of the total weight of the vehicle, it can be concluded that around 64% of the total weight lies at the lower part of the vehicle. This lowers the center of gravity of the vehicle which increases the stability of the vehicle.

5 Description of Electronic and Power Design

5.1 Overview

Ashwini's electronic architecture is designed for safety, efficiency, and reliability with seamless integration. An ASUS TUF Gaming Laptop handles high-level tasks like sensor fusion, navigation, and real-time decision-making. Propulsion uses RHINO High Torque Planetary DC Geared Motor motors controlled by the Cytron MDD20A Dual Channel DC motor driver for precise control. A LoRa-based wireless emergency stop system allows remote shutdown from up to 1 km away. An Arduino manages low-level control and communication between components, while a safety LED strip visually signals the robot's operational status. Overall, the system balances advanced functionality with practical safety, ensuring robust performance in complex environments.

5.2 Power Distribution System

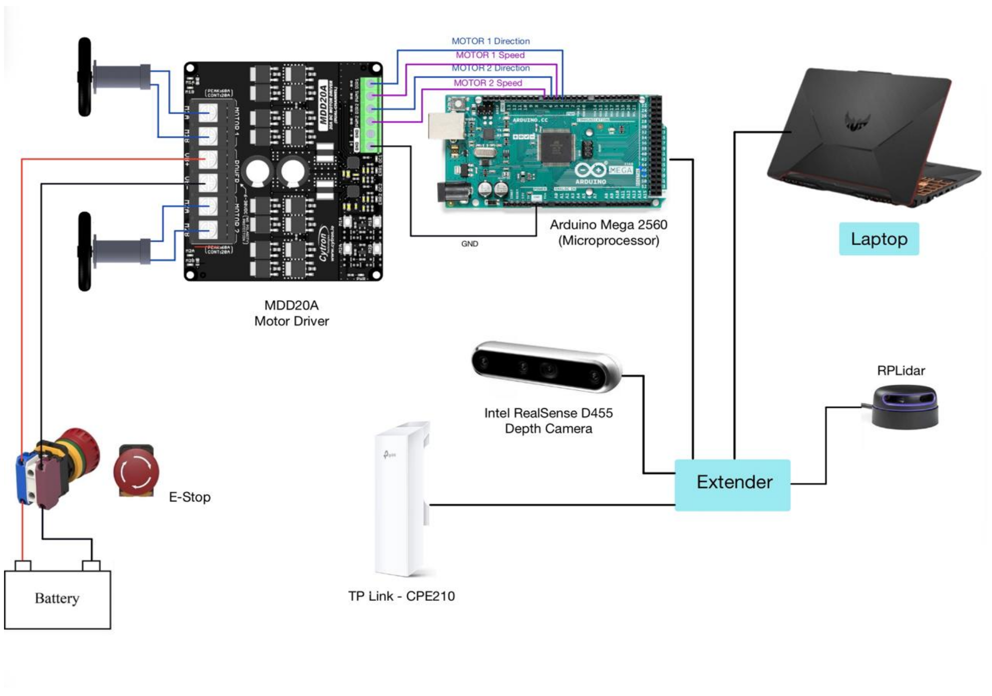


Figure 6: Electrical Design Suite

COMPONENTS	POWER CONSUMPTION	OPERATING VOLTAGE
RP Lidar	2.2 W	5 V
U-blox GNSS receiver	0.5 W	3.3 V
Depth Camera D455	1.9 W	5 V
Motor Controller	300 W	24 V
TOTAL POWER	**304.6 W**	

BATTERY SPECIFICATIONS	MEASURED VALUE
Battery Capacity	7000mAh
Operating Voltage	24V DC
Power Supplied (Ideally)	66.66 W
Power Supplied (Actually)	65.4 W
Battery Life	2Hr.
Recharge Rate	35 min.

5.3 Electronics Suite Description

5.3.1 Onboard Computing System

We replaced the Jetson Xavier NX with an ASUS TUF Laptop featuring a 10th Gen Intel Core i5, 8GB RAM, and an NVIDIA GTX 1080 GPU. Running Ubuntu 20.04, it executes the robot's core control logic via Python scripts. The enhanced CPU and GPU enable efficient real-time data processing, computer vision, and autonomous navigation.

5.3.2 Motor Controller and Motors

Ashwini uses RHINO High Torque Planetary DC Geared Motors for their efficiency and compact design. They're controlled by a Cytron MDD20A dual-channel motor driver (6–30V, 20A continuous, 60A peak) with built-in protections against over-voltage, over-temperature, and over-current.

5.3.3 RP Lidar

The system utilizes the RP Lidar A3M1, a 2D scanning LiDAR sensor known for its efficiency and precision. It offers a full 360-degree field of view with a scanning range of up to 25 meters and operates at a frequency of 10Hz. Thanks to its compact design, low power requirements, and high-speed data acquisition, it serves as a reliable sensing solution for various autonomous navigation and mapping tasks.

Figure 7: RP Lidar

5.3.4 Intel RealSense Depth Camera D455

The Intel RealSense D455 stereo depth camera offers accurate 3D perception with a 90° field of view and up to 20m range. Capturing 1280×720 at 90 FPS, it provides high-quality depth and RGB data for real-time obstacle detection, mapping, and navigation, even in varied lighting.

Figure 8: Intel RealSense Depth Camera D455

5.3.5 GPS Module

We've used the u-blox C099-F9P-1 GNSS board with the ZED-F9P chipset for high-precision positioning. Supporting RTK corrections, it delivers 1–2cm accuracy over L1/L2 bands, making it ideal for autonomous navigation and mapping.

Figure 9: u-blox C099-F9P-1 GNSS

5.4 Safety-Devices

Ashwini is equipped with both hardware and software-based emergency stop mechanisms to ensure safe operation at all times. A **physical E-stop button** is prominently mounted at the top of the robot, allowing anyone nearby to immediately halt its motion by pressing the button.

In addition, a **wireless emergency stop system** has been implemented using Arduino microcontrollers, a relay circuit, and a pair of LoRa communication modules. One module serves as the transmitter, held by the operator, while the other acts as the receiver installed on the robot. This setup allows the robot to be remotely stopped from distances up to 1 kilometer with a single button press, offering a reliable long-range safety control method.

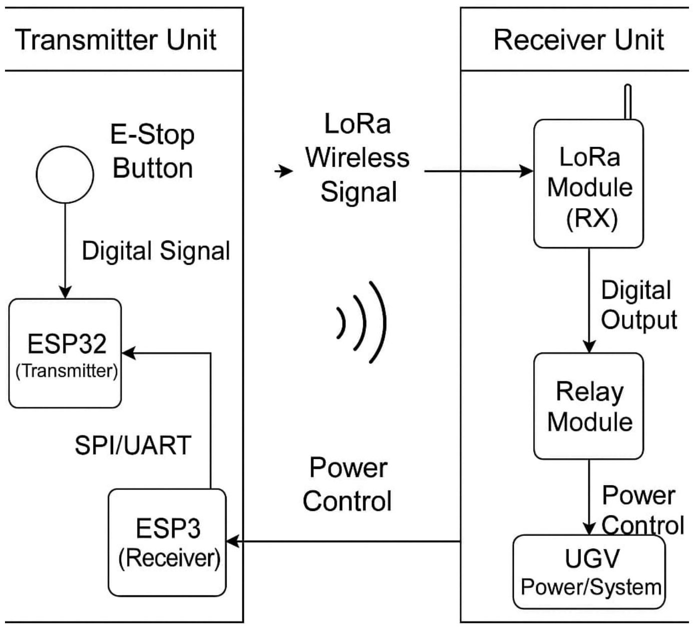


Figure 10: Wireless E-Stop System

6 Description of software system

6.1 Overview

Ashwini's onboard software manages perception, mapping, planning, and control for autonomous navigation in IGVC. Built on ROS Noetic with Ubuntu 20.04, it emphasizes modularity, simulation-driven development, and cost-efficiency. A high-performance laptop runs core tasks, using tools like Gazebo, RViz, and Docker. The system follows a "simulate extensively, test physically" approach for rapid, reliable iteration. Structured software modules ensure smooth data flow, while version control enables scalable, collaborative development.

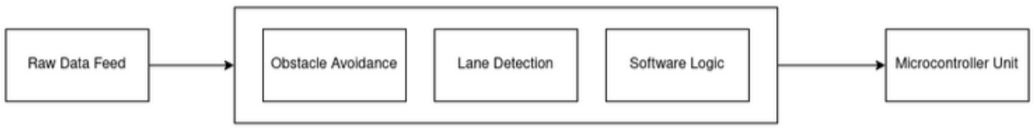


Figure 11: General Software Architecture

6.2 Obstacle Detection and Avoidance

Reliable obstacle detection and avoidance are critical for Ashwini's autonomous navigation. The system uses two main sensors:

RPLiDAR A3M1 360° 2D LiDAR that creates a dense point cloud to update a 2D costmap in ROS, marking obstacles as occupied cells.

Intel RealSense D455: Used mainly for lane detection and depth data. Although machine learning for object recognition was tested, real-time use was limited by computational load and lack of IGVC-specific training data, so LiDAR remains the primary obstacle input. Obstacles in the costmap are handled by the local path planner, which uses algorithms like Dynamic Window Approach (DWA) to generate collision-free paths around obstacles while guiding the robot toward its goal.

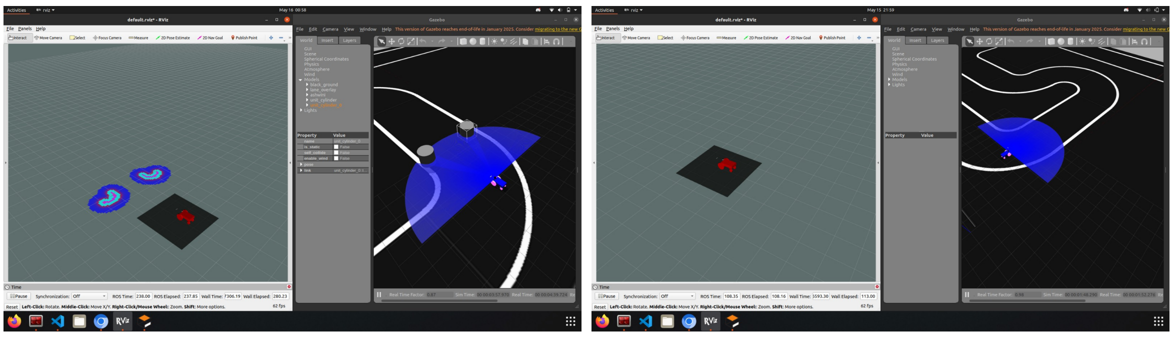


Figure 12: Obstacle detection demonstration

6.2.1 Software Strategy

Ashwini's software strategy leverages the ROS Noetic framework on its onboard laptop to integrate sensors (RPLiDAR, Intel RealSense Depth Camera D455, and our team-implemented RTK GPS) with perception, planning, and control algorithms. For robust localization, an Extended Kalman Filter (EKF) from the `robot_localization` package fuses IMU and odometry data with inputs from our RTK GPS system, providing an accurate pose estimate (`odom` frame). Custom Python-based ROS nodes handle specific tasks like sensor data publishing, motor control (interfacing with an Arduino and the motor controller), and specialized behaviors. Ashwini employs a hybrid decision-making approach, utilizing the `move_base` stack for goal-driven waypoint navigation and reactive control strategies for other tasks like lane following.

6.2.2 Path Planning

Ashwini's path planning is managed by the ROS `move_base` navigation stack, which features a two-tiered approach for charting optimal and safe courses:

- **Global Planning:** A global planner, typically an **A* (A-star) or Dijkstra-based algorithm** (e.g., `navfn/NavfnROS` or `global_planner/GlobalPlanner`), operates on the static map of the environment (generated via SLAM). It computes an optimal long-range path from Ashwini's current location to a specified goal, considering known obstacles represented in the global costmap.
- **Local Planning:** To execute the global plan and react to immediate, dynamic obstacles, a local planner is used. Our configuration employs the **Dynamic Window Approach (DWA)** or a similar trajectory rollout algorithm (e.g., via `base_local_planner`). This planner generates kinematically possible velocity commands by operating on a local costmap that is frequently updated with current LiDAR data, ensuring safe and adaptive navigation.

6.2.3 Map Generation

Accurate mapping is essential for Ashwini's navigation. It uses ROS's gmapping package with a Rao-Blackwellized particle filter to create a 2D occupancy grid map from RPLiDAR scans. Grid cells are marked free, occupied, or unknown, with resolution balanced for detail and performance. Parameters like particle count are tuned via testing.

High-quality odometry is critical; Ashwini's EKF fuses IMU, odometry, and RTK GNSS data for drift-resistant pose estimates, improving map consistency. LiDAR data is pre-processed with laser filters to reduce noise and enhance accuracy.

6.3 Goal Selection and Path Generation

Ashwini's navigation to goals is managed through custom scripts interfacing with the ROS move_base stack. Goal Specification: Goals are waypoints (x, y, yaw) in the map frame, based on the IGVC course. They're loaded from mission files and sent sequentially to move_base via the waypoint.py script. Path Execution:Upon receiving a goal, move_base uses global and local planners to navigate the path. The waypoint.py script monitors progress and moves to the next waypoint when reached. Built-in recovery behaviors like costmap clearing and rotation ensure robust navigation.

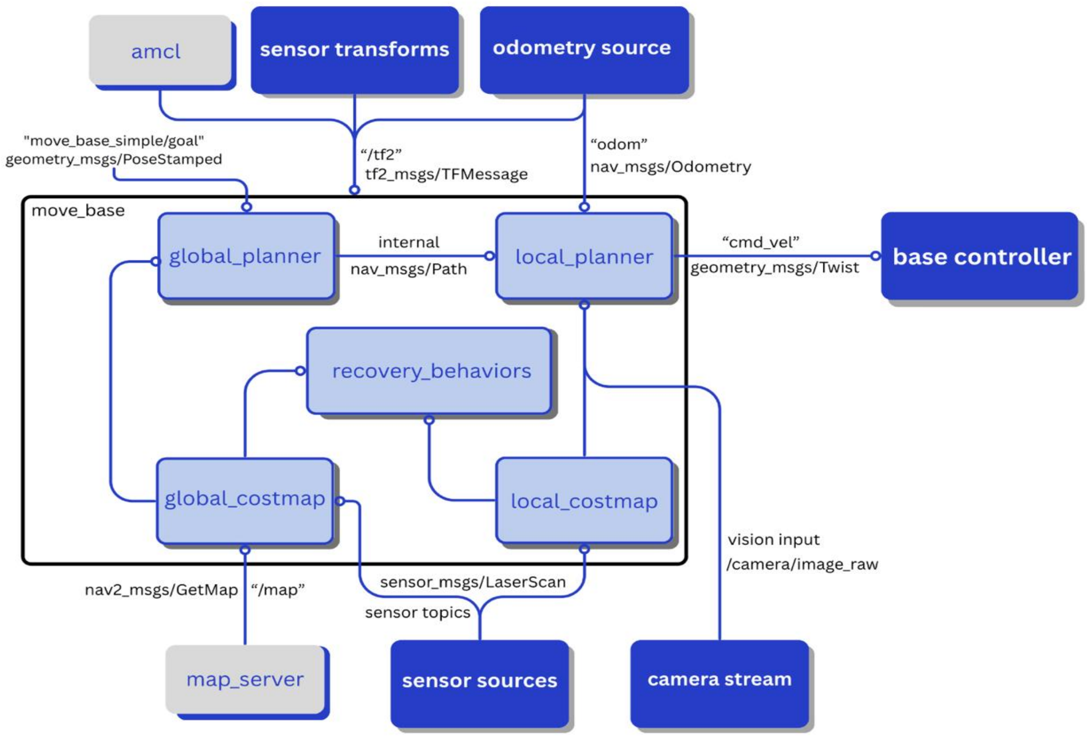


Figure 13: Navigation Stack

6.4 Additional Creative Concepts

Ashwini's design incorporates several creative concepts, emphasizing robust, cost-effective solutions on our onboard laptop to meet IGVC challenges:

- **Unified Lane Keeping via "Lane-as-Obstacle" Costmap:**Lane boundaries from the camera are projected as virtual obstacles in the local costmap. This "fake" laser scan lets the local planner (e.g., DWA) handle lane keeping and obstacle avoidance together, simplifying navigation and boosting reliability.
- **Team-Implemented RTK GPS for Centimeter-Level Accuracy:**Using two GNSS modules, we built a custom RTK GPS system providing centimeter-level positioning. This reduces odometry drift, improving map quality and enabling precise navigation.

7 Cyber Security Analysis Using RMF

As autonomous vehicles increasingly rely on interconnected systems and software-driven decision-making, cybersecurity becomes a critical design consideration. Our team followed the **NIST Risk Management Framework (RMF)** to evaluate and mitigate potential cyber threats to our autonomous ground vehicle, **ASHWINI**, particularly in shared environments such as the IGVC competition pit and course areas.

7.1 Understanding the RMF Process

The NIST RMF is a structured process for integrating cybersecurity and risk management activities into the system development life cycle. We adopted the following six steps of RMF:

1. Categorize the system
2. Select appropriate security controls
3. Implement the controls
4. Assess the effectiveness of controls

5. Authorize the system to operate
6. Monitor the controls continuously

7.2 Threat Modeling for ASHWINI

To validate the effectiveness of our controls, we performed the following actions:

- Disabled USB auto-mounting and monitored `dmesg` logs for unauthorized devices.
- Used UFW firewall on Ubuntu to block all inbound traffic except required ROS ports and SSH.
- Conducted penetration tests by attempting spoofed ROS topic publishing (e.g., false `/cmd_vel`) and verified that ASHWINI's motor control node rejects external/untrusted publishers.
- Ensured encryption and key-only access for all remote logins during debugging phases.
- Used `roswtf` and `rosnode info` to detect and terminate unexpected nodes.

Threat	**Source**	**Impact**
USB-based malware injection	Physical access to laptop port	Unauthorized software execution, data corruption
Wireless interference or spoofing	Nearby rogue devices	Navigation manipulation, data falsification
ROS topic spoofing or denial	External node injection	Misleading sensor data, command hijacking
Unauthorized SSH access	Open Wi-Fi connection	Full system takeover
Software crashes from malformed input	Faulty hardware or software attack	Robot malfunction or unsafe behavior

7.3 Ongoing Cybersecurity Monitoring

To ensure continued protection:

- Cron jobs run background monitoring scripts to detect new network connections or unauthorized ROS nodes.
- All system logs (`/var/log/auth.log`, ROS logs) are checked daily during development.
- A manual cybersecurity checklist is followed before every deployment or test run.

High-Risk Area	**Selected Controls (NIST IDs)**	**Implementation**
External USB Access	AC-19 (Access Control for External Devices)	USB ports disabled in BIOS; tape seals applied
Wi-Fi/SSH Access	AC-17 (Remote Access), SC-12 (Cryptographic Key Establishment)	SSH over RSA keys only; Wi-Fi disabled during competition
ROS Topic Security	SI-4 (Information System Monitoring), SC-7 (Boundary Protection)	Custom ROS node authentication; namespace isolation and topic access restrictions

Table 6: Security Controls Applied to High-Risk Areas

8 Analysis of Complete Vehicle

8.1 Simulations

A dynamic Python model integrated with Gazebo simulated ASHWINI's behavior, while Rviz was used for visualization and testing, guiding software architecture and control validation. Mechanical load analysis with a factor of safety of 2 confirmed the chassis could support up to 200 pounds, ensuring structural integrity.

System integration was modular, testing mechanical, electrical, and software subsystems independently before full assembly. Sensors (LiDAR, GPS, depth camera) were optimally mounted to reduce interference, and ROS nodes were validated via rosbag recordings. Safety features like emergency stop and manual override were tested thoroughly. Software development used GitHub with branch-based version control. Common issues such as LiDAR reflections, GPS drift, and node crashes were mitigated with retry logic and hardware fixes. Simulation-tested software releases preceded real-world trials. Performance closely matched predictions, with minor deviations in speed, battery life, GPS accuracy, and ramp climbing, which were addressed by planner tuning and speed adjustments to ensure robust autonomous navigation.

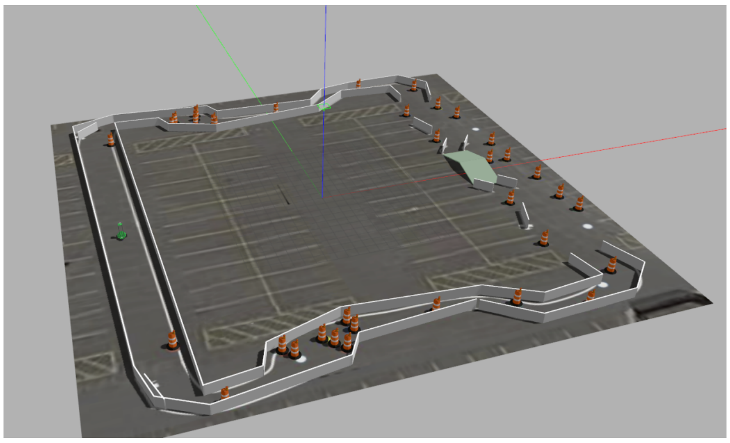


Figure 14: Simulated Environment

8.2 Performance Testing to Date

Ashwini was tested in an outdoor environment similar to the IGVC competition, featuring a track layout resembling the competition setup. To thoroughly evaluate the vehicle's performance, white lines were marked on the grass, and barrels were placed intermittently, creating an ideal pseudo-competition environment. The outdoor testing phase took place once all system modules were independently operational and seamlessly integrated.

To validate the mechanical aspects, rigorous tasks were performed, including ramp climbing, maneuvering on wet surfaces, and navigating rough terrain using RC control. The system's mechanical robustness was assessed by executing sudden maneuvers to ensure its stability and durability.

Electrical testing and validation were conducted to verify the proper functioning of each electronic component and sensor. This involved analyzing the signals of these components in Rviz to ensure they met the desired specifications. The integrity of the power distribution system was monitored using a digital serial oscilloscope, ensuring its reliability and performance.

9 Unique Software, Sensor, and Control Strategies Tailored for AutoNav

ASHWINI's autonomous system is specifically designed to tackle the unique demands of the IGVC AutoNav challenge. Its sensor fusion framework integrates RTK-enabled u-blox F9P GPS for centimeter-level waypoint accuracy, RPLiDAR A3M1 for 360° obstacle detection, and Intel RealSense D455 for lane perception. A key innovation is the use of a custom "lane-as-obstacle" strategy, where lane boundaries detected by the vision system are injected into the ROS costmap as virtual obstacles. This allows the standard local planner (DWA) to naturally guide the robot within lanes without needing a separate lane-following algorithm, significantly simplifying control logic and improving adaptability in varying environments. On the software side, ASHWINI features a modular ROS-based architecture with optimized planners and navigation logic. The global planner uses A*/Dijkstra algorithms to generate paths toward GPS waypoints, while the local planner dynamically adjusts trajectories based on real-time obstacle data. A custom waypoint navigation script manages GPS transitions, with goal-switching logic, fail-safe handling, and pause/resume support. All modules were rigorously validated in a Gazebo-based virtual IGVC environment before field deployment. This simulation-first approach, combined with tailored perception and control strategies, ensures robust, real-time autonomous navigation in IGVC's complex AutoNav course.

10 Initial Performance Assessments

Ashwini follows all the requirements specified in the rules of IGVC 25'. In the initial testing phase, each and every component and sensor was individually tested and passed all performance criteria. We have tested Ashwini in both simulated and actual courses.

1. On-board computer (Jetson) was tested instead of controlling the bot by laptop. The performance of the bot was up to the mark.
2. A maximum speed of 5mph was achieved during testing.
3. The battery life during the testing phase was found to be 45 min. At full working capacity.
4. Ramping ability has been rigorously tested over multiple inclines of up to 16-degree in inclination.
5. Lidar was individually tested for obstacle detection and avoidance.
6. Intel RealSense camera was tested for visual odometry and white lane detection.
7. Software algorithm (A* algorithm) for lane detection was tested successfully.