MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Institution of MAHE

PROJECT MANAS
BRINGING BITS TO LIFE

OTTO

Team Captain
Aman Pandey | aman.pandey1@learner.manipal.edu

Sensing & Automation
Anjas Biswas | anjas.biswas@learner.manipal.edu
Deep Gupta | deep.gupta@learner.manipal.edu
Yashas Ranjan | yashas.ranjan@learner.manipal.edu

Artificial Intelligence
Divyam Chandalia | divyam.chandalia@learner.manipal.edu
Eshaan Shah | eshaan.shah@learner.manipal.edu
Nathan Saldanha | nathan.saldanha@learner.manipal.edu
Rudra Patel | rudra.patel@learner.manipal.edu
Samik Pujari | samik.pujari@learner.manipal.edu

Mechanical
Amogh Rao | amogh.rao@learner.manipal.edu
Divyam Shah | divyam.shah@learner.manipal.edu
Mahati Balaji | mahati.balaji@learner.manipal.edu
Prajwal Govind | prajwal.bhat2@learner.manipal.edu
Shreyas Gali | shreyas.gali@learner.manipal.edu
Yash Sethia | yash.sethia@learner.manipal.edu

I certify that the development of the vehicle, Otto, described in this report has been equivalent to the work in a senior design course. The students of Project MANAS have prepared this report under my guidance.

15.5.23

Professor & Head
Dept. of Computer Science & Engineering
Manipal Institute of Technology
Manipal - 576 104

Ashalatha Nayak
Professor
Department of Computer Science & Engineering
asha.nayak@manipal.edu

Submitted: 15th May 2023

Table of Contents

1. Design & Team Overview

1.1. Introduction

The members of Project MANAS, the official AI and Robotics team of Manipal Institute of Technology, India, have authored this Technical Design Report. The team has designed its 3rd generation autonomous bot, Otto, to compete in the 30th Intelligent Ground Vehicle Competition. It is equipped with a comprehensive software stack, sensor suite, and perception modules, housed in a robust mechanical framework to complete the Auto-Nav mission successfully.

1.2. Organization

Project MANAS comprises 34 students pursuing Undergraduate Degrees, across various engineering disciplines, including but not limited to Mechanical, Electronics and Electrical, and Computer Science. Over four months, the team has invested about **728 hours** and around **$6,000** towards developing Otto. The team is broadly classified into four subsystems.

1.2.1. Artificial Intelligence

Develops, trains, and deploys the perception modules and software stack for localization, path planning, and navigation.

1.2.2. Electronics

Ensures optimal performance of the overall system by managing power systems, integrating sensor suites, and interfacing with actuators.

1.2.3. Mechanical

Utilizes CAD software for detailed design, runs simulations to optimize materials, and fabricates the structure using various manufacturing methods.

1.2.4. Management

Oversees the financials, logistics, and coordination of the team and maintains the team's online presence, ensuring seamless communication with sponsors and the college.

1.3. Design Process

Our design process for Otto followed a **V-shaped model**, where we have first decomposed the problem statement into smaller sections, and then integrate our ideas and solutions towards fulfilling the same. This structured approach allowed us to systematically tackle complexities and ensure a high-quality end product, as elaborated in ***Figure 1***.

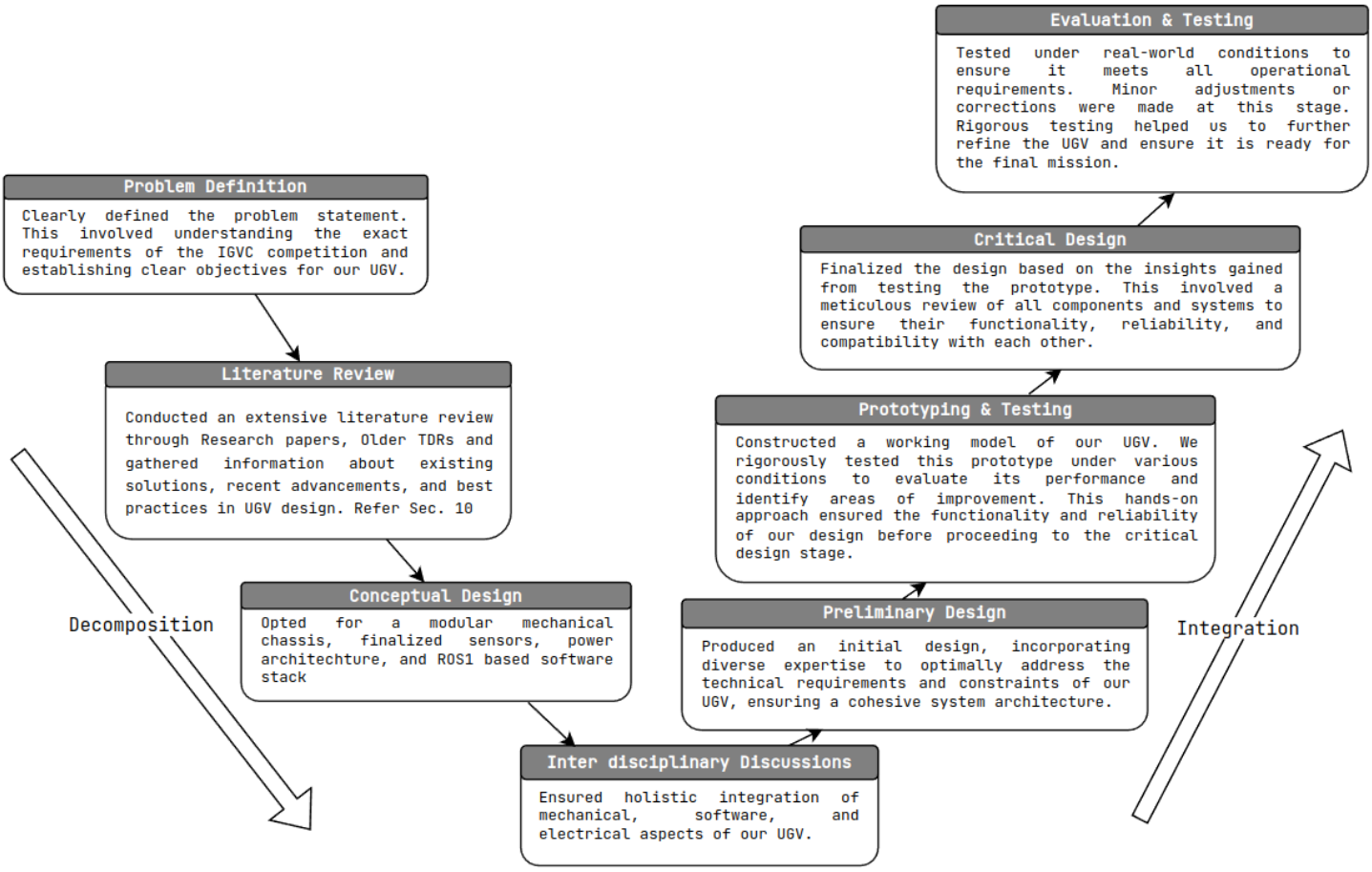

Figure 1: *Overview of the Design Process*

2. Effective Innovations

2.1. CLRNet

CLRNet [1] is a state-of-the-art lane detection algorithm that utilizes high-level and low-level semantics to detect lanes fully. This technique first detects lanes with high-level semantic features and then refines based on low-level features, significantly improving localization accuracy. The architecture is summarized in ***Figure 2*** [1].

One of the key advantages of CLRNet is that it treats lanes and obstacles as separate entities. This separation enables more robust lane detection, as even if an obstacle partially blocks the view of lanes, CLRNet can still project the lane lines beyond the obstructed view. We make use of a traditional color thresholding-based attention map to compensate for the requirement of a large amount of data to train CLRNet or deep learning-based approaches in general.

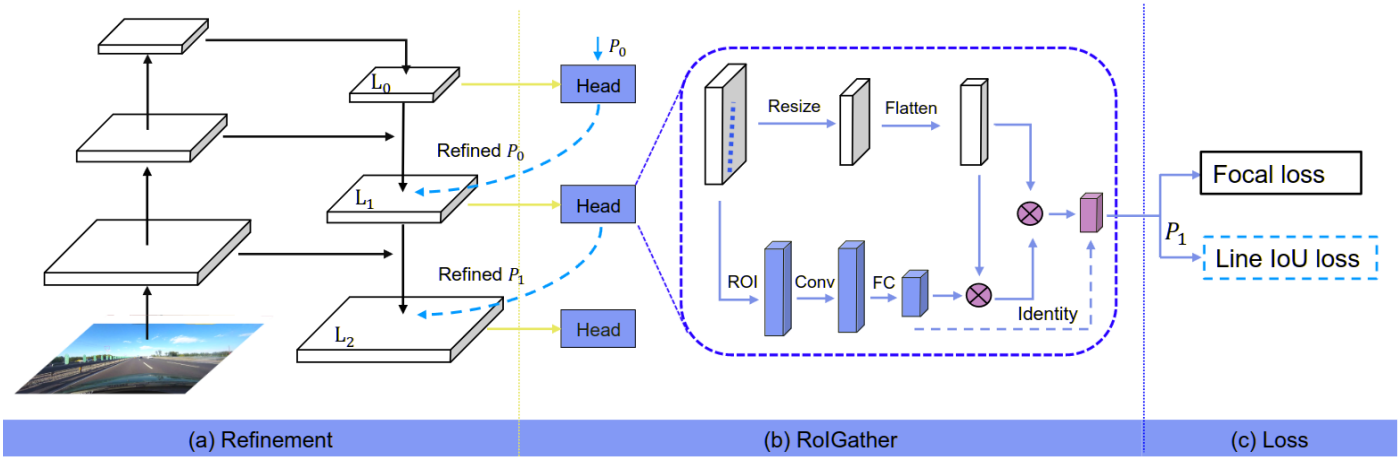

Figure 2: *CLRNet Architecture*

2.2. Vibration Mitigation

This year's competition regulations require an asphalt running field, which allows for higher vehicle speeds but poses a challenge for obtaining precise sensor readings due to vibrations. We have implemented several design improvements in Otto to mitigate this issue.

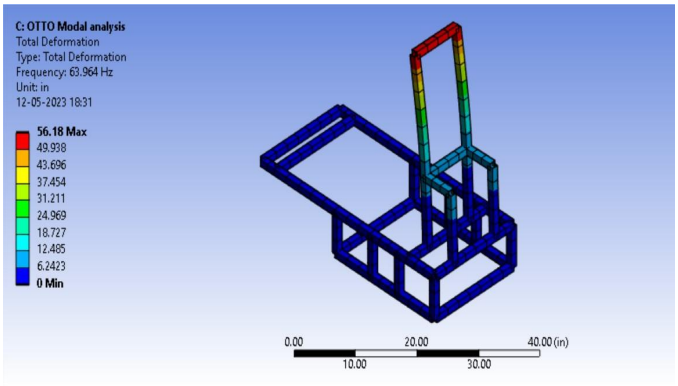

Figure 3: *Natural Frequency before CF Rods*

We incorporated 3K 200GSM 2×2 twill Carbon-fiber rods for their lightweight and high modulus of rigidity, enhancing the sensor pole's stability. Our choice of tubeless rubber tires with symmetrical block threading patterns ensured better traction and noise reduction. To further limit vibrations, we replaced hammer nuts with leaf spring hammer nuts, known for their resistance to free vibrations.

Additionally, including spring-loaded castor wheels at the rear end further diminished the impact of free vibrations on overall performance.

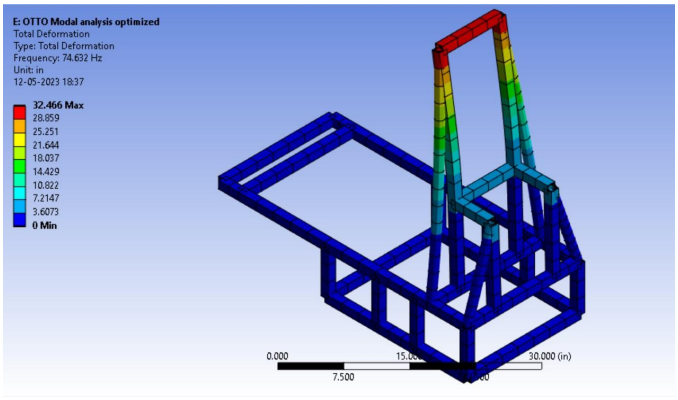

Figure 4: *Natural Frequencies after CF Rod*

Modal analysis was used to evaluate Otto's modifications. It identifies the system's natural frequencies, reflecting its vibration tendencies. The provided simulation focusing on the likely vibration direction illustrates the chassis's modal analysis with and without carbon fiber rods.

Figure 3 demonstrates that the chassis had a natural frequency of 64 Hz in the absence of carbon fiber rods. Upon reinforcing the chassis with carbon fiber rods, as shown in ***Figure 4***, the frequency increased to 75 Hz, indicating a significant 16.6% improvement.

This rise in natural frequency signifies the carbon fiber rods' effectiveness in enhancing the rigidity of the chassis and reducing vibrations. This frequency shift is particularly crucial as it moves the natural frequency away from the operational frequencies, thus mitigating the risk of resonance. The simulation results underscore the success of our modifications in bolstering Otto's structural integrity and vibrational resistance.

3. Mechanical Design

3.1. Chassis Overview

Otto's mechanical design as shown in ***Figure 5*** and ***Figure 6***, has been thoughtfully crafted to achieve optimal stability, strength, and performance and considered Design for Assembly while addressing challenges specific to the competition.

The modular framework utilizes T-slots and Leaf-spring nuts to absorb vibrations and provide flexibility for customization. Carbon fiber rods offer structural support to the sensor pole, while heavy-duty castor wheels provide maneuverability and reduce the impact of vibrations.

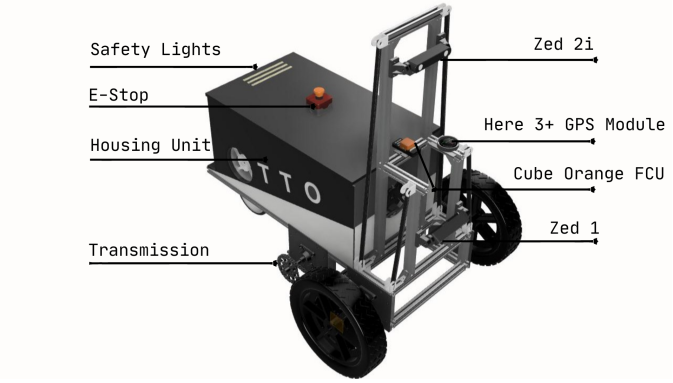

Figure 5: *Overview of the Assembly of Otto*

We designed the housing unit with groove fittings for fast and efficient assembly, and tubeless tires offer high durability and require low maintenance and repairability. The cooling system incorporates three fans to maintain optimal operating temperatures for critical components. The power transmission deploys sufficient torque at high RPM to the wheels from the BLDC motors, rated for high torque and low RPM applications.

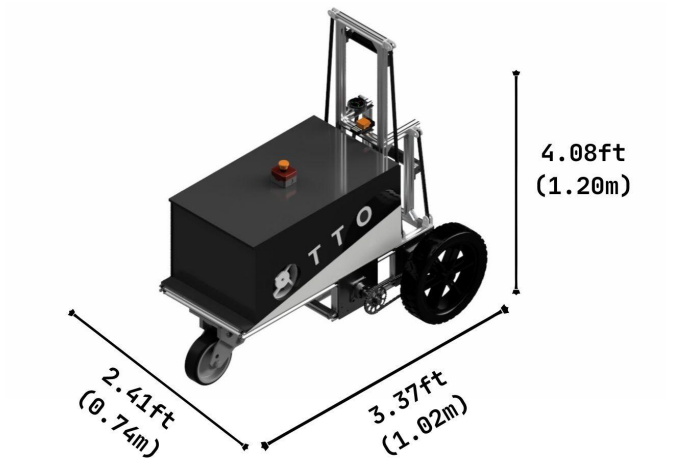

Figure 6: *Overview of the Assembly of Otto*

3.2. Power Transmission

To ensure swift mission completion while maintaining the necessary velocity, Otto's max operation speed has been set to 3 mph (1.3 m/s).

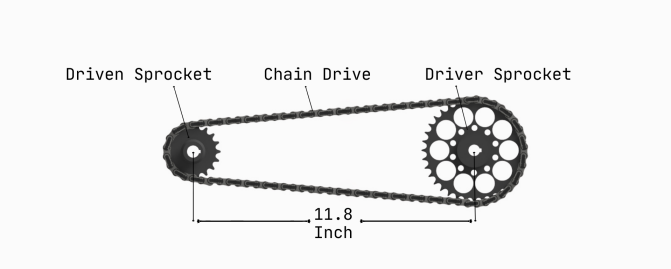

Figure 7: *Overview of the Power Transmission*

However, the locally sourced motors can only deliver a linear velocity of up to 1.6mph (0.7m/s). Hence, we implemented a 2:1 gear ratio using sprockets shown in ***Figure 7***, effectively doubling it to 3.2mph (1.4m/s). Additionally, the sprockets assist in a lower center of gravity by adding weight at the bOttom, enhancing stability and better turning maneuverability.

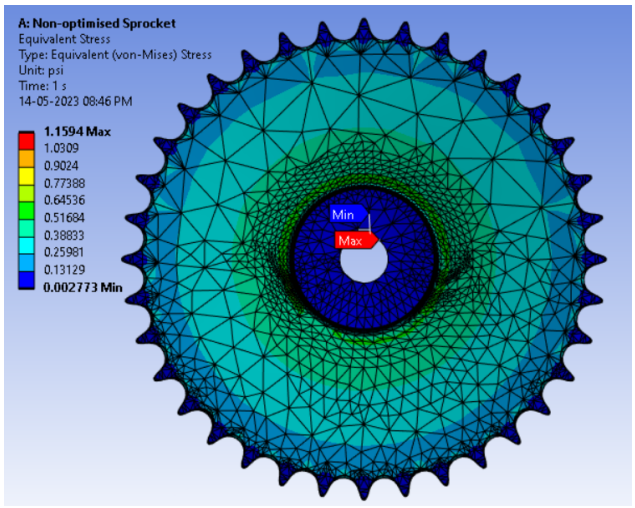

Figure 8: *Unoptimized Sprocket Design*

We manufactured the sprockets in-house using mild steel. Based on our calculations, we needed an 18-tooth driven sprocket and a 36-tooth driving sprocket, with a center-to-center distance of 11.8 inches (0.29m). We set this to a diametral pitch of 4 inches for precise specifications and machining accuracy, and procured a chain capable of handling 5.9 lb-ft (8N-m) of torque.

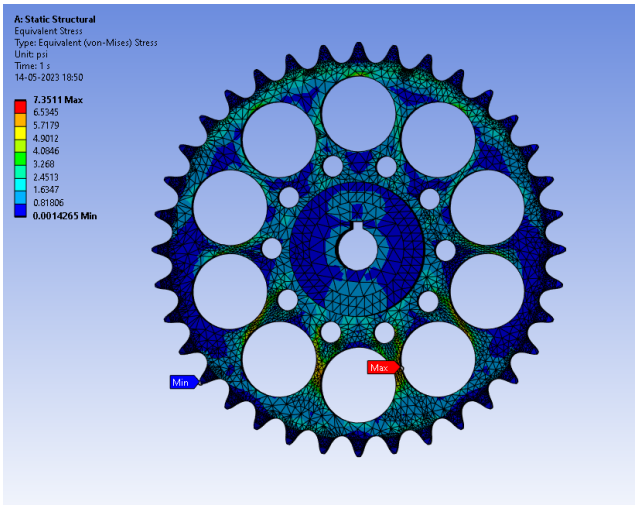

Figure 9: Optimized Sprocket Design

The sprocket design in ***Figure 8*** was considered over-engineered, leading to a new iteration in ***Figure 9*** with a 13% decrease in hub diameter and added holes to reduce weight and rotational inertia. These changes effectively lowered the starting torque requirement of the motors, reducing overshoots and related control issues. The results of structural simulations show a negligible deformation of 3.9µm and a max stress of 7.35psi (50.68kPa)—well below the yield stress according to von Mises failure criteria—attest to a reliable and robust ground vehicle.

3.3. Housing Unit

Otto's housing unit is built with a Tri-layer Oriented Strand Board Plywood to house all its components, as depicted in ***Figure 10***.

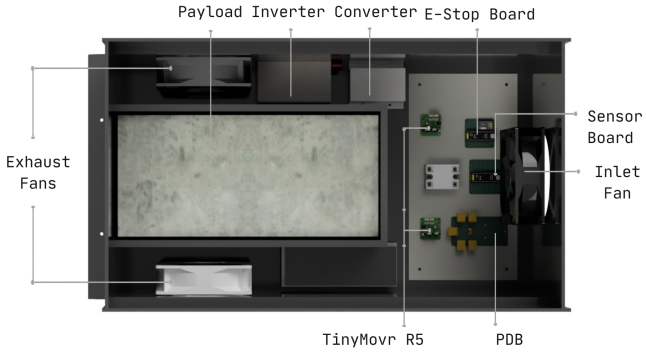

Figure 10: Overview of the Housing Unit

The modular design comprises grooves engraved on the faces, allowing them to easily interlock and slide, simplifying assembly and facilitating future changes. The compact design of the box, with a minimum one-eighth-inch clearance, accommodates all components, with the payload situated in a separate compartment at the rear of the box.

3.4. Cooling System

Our cooling system is strategically designed with three fans to ensure optimal temperatures for the electronics. The Arctic Bionix F140 fan, providing direct cooling, is supplemented by two Noctua NF-AF15 PWM fans serving as exhausts. This arrangement facilitates continuous air circulation within the electronics compartment, promoting peak performance of critical components like the Tinymovr, PDBs, Inverter, and laptop while preventing overheating and potential temperature-related damage.

3.5. Weather Resistance

3.5.1. Hydrophobic Coating

An equal mix of sealants and epoxy additives has been incorporated with water-resistant paint to create a hydrophobic coating. This layer provides Otto with a resilient barrier. This modification makes it resistant to rain and providing sufficient time for recovery in case of any emergencies.

3.5.2. Secured Fasteners

To prevent the potential degradation of the fasteners due to environmental factors, we utilize Loctite 243 on all nuts and bolts. This adhesive locks the bolts and nuts and acts as a protective barrier against environmental corrosion, thereby increasing the endurance of Otto's structural integrity.

3.5.3. Wet Surface Mobility

Otto's mobility in wet conditions is enhanced significantly by the use of complex tire tread patterns that are implemented to evacuate water efficiently. These patterns help to disperse the water and enhance grip on wet surfaces. The channels and grooves work by directing water away from the center of the tire and towards the edges, where it is effectively expelled.

4. Electronics Design

4.1.Electronics Architecture

Otto uses a Lenovo Legion 5 Pro laptop for computation, two RMCS-2071 100W BLDC motors for mobility, and sensors including ZED cameras, Cube Orange flight controller, Here3+ GNSS, and Hall encoders for precise control and navigation. ***Figure 11*** summarizes the Electronics Architecture.

4.1.1. Computation Hardware

The primary processing of Otto's entire navigation stack takes place on a laptop housed inside the mechanical chassis. We chose to use a laptop for running the entire navigation stack due to its portability, cost efficiency, versatility, and ability to facilitate efficient testing and

debugging. The 6GB VRAM on the laptop of choice allowed us to run advanced and computationally intensive lane detection algorithms like CLRNet, natively.

An STM32F4 MCU-based sensor board collects data from X8R Receiver, Temperature sensor, and INA226 Power monitor IC and sends it to the laptop after processing.

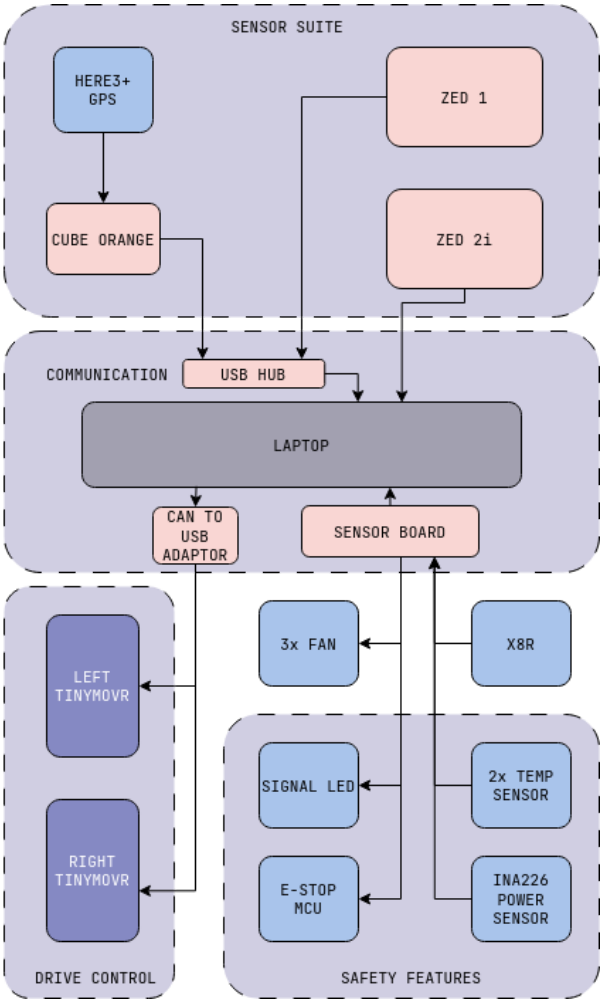

Figure 11: *Electronics Architecture*

4.1.2. Drive Control

The core of Otto's control strategy revolves around achieving precise velocity control through two independently controlled motor controllers, with each wheel having its dedicated controller. Two RMCS-2071 100W BLDC motors actuate the differential drive robot. Initially meant for testing, these motors have become an integral part of our final design due to logistical complications with the manufacturing and shipping of our first choice, Maxon BLDC motors with incremental encoders. The RMCS-2071 motors were initially chosen for their affordability and local availability and later adapted into the final design. They feature a built-in planetary gearbox with a 1:96 gear ratio, which, in conjunction with a 2:1 chain sprocket mechanism, effectively boosts the RPM while satisfying torque requirements.

We control these motors by utilizing TinyMovr R5 BLDC motor controllers, which actively provide velocity control through the implementation of Field Oriented Control (FOC). FOC is a superior control strategy that optimizes BLDC motor control for precise torque and speed management which leads to smoother and more efficient operation, minimized vibrations, and faster response times.

For setting velocity setpoints on each motor for following a path decided by the planner, we leverage the Robot Operating System (ROS) and Python, utilizing the 'cmd_vel' topic to relay velocity commands. The TinyMovr R5 motor controllers also provide precise encoder feedback, serving as a low-noise data source for odometry, enabling us to close the control loop.

Our control strategy includes a feedforward plus PI controller approach for each wheel. The feedforward term, adaptive based on the previous state of the vehicle, compensates for mechanical complexities like backlash and chain slack. This dramatically enhances the responsiveness of Otto, particularly when transitioning from a stationary state to motion.

The tuning process involved adjusting the PI gains, integrator deadband, and feedforward gains to ensure reliable tracking of linear and angular velocities. We initially faced challenges with low-speed maneuvers, but through fine-tuning, we optimized Otto's operation to track the desired velocities under a range of conditions efficiently.

4.1.3. Sensor Suite

ZED 2i

The ZED 2i is an advanced stereo camera renowned for its long-range 3D depth perception capabilities, covering distances from 5 ft to 115 ft. It offers a horizontal FOV of 72 degrees and a vertical FOV of 44 degrees. With exceptional depth accuracy, it achieves 98% depth accuracy up to 33 ft and maintains 93% up to 99 ft.

ZED 1

The ZED 1 stereo camera provides a depth range of up to 65 feet and offers a wide 110-degree field of view. To eliminate blind spots, it is strategically positioned at a lower height on the sensor pole.

Here 3+

The Here3+ is a cost-efficient GNSS system that provides a Positioning accuracy of 8 ft with a 3D fix and works over the DroneCAN protocol. This positional accuracy is further improved by the fusion of two such GPS modules with the sensor array on the cube through an EKF for positional state estimation.

Cube Orange

The Cube Orange is a flight controller that handles the EKF Sensor Fusion of the IMUs, Magnetometers, and the GPS data to get a vehicle state estimate which provides filtered IMU and GPS data that is published over MavROS to the Laptop.

The Cube Orange features three IMUs which are temperature-controlled by onboard heating resistors for optimal operation.

Two sets of IMU are vibration-isolated mechanically, reducing the effect of frame vibration on state estimation. There are three magnetometers, one on the cube orange and one each on the GPS modules, which upon fusion through an Extended Kalman Filter, allows for precise yaw estimation.

The advantages of using an FCU such as the Orange cube provided to us are:

- Filtered GPS and IMU data provide us with accurate and relatively inexpensive, and reliable GPS and IMU data for higher level sensor fusion for odometry on the ROS nav stack.
- The use of a flight controller allowed us to use the native DroneCAN protocol on the Here3+ without having to integrate the GPS modules with regular CAN for use with our CAN bus for the motor controller.

Motor Encoders

Each BLDC motor in Otto is equipped with Hall encoders, which are connected to the TinyMovrs. These encoders provide feedback to the motor controllers, enabling the FOC algorithm to effectively control the velocities of the BLDC motors.

Moreover, the encoder position estimates are not only utilized for velocity control but also transmitted over the CAN bus through the ROS node. Another node utilizes this data to publish wheel odometry information.

4.1.4. Communications

Otto employs a robust communication and interfacing system to integrate the data stream from the sensors, the perception and planning algorithms, and the control of the actuators.

The Tinymovr Motor Controllers are daisy-chained together with a CAN to USB adaptor, ensuring a robust data stream between the laptop and the motor controller sensors interface with the laptop through USB to serial connections (UART).

A USB hub consolidates these devices, providing ease of use and expandability. The flight controller communicates via MAVLink, which is converted to ROS topics and services through the MAVROS package. High-accuracy Here3+ GPS modules, repurposed from our unmanned autonomous drone project, connect to the flight controller using the droneCAN protocol.

The wireless E-Stop remote communicates to Otto using NRF24F Radio pair, which operates at 2.4 GHz with a data rate of 1mbps. The transmitter can provide a maximum range of 0.5 miles and sends 20 messages per second to ensure fast response times. The sensor board is equipped with an X8R receiver to enable manual control functionality, which is paired with a Taranis controller for precision handling. Utilizing the high-speed SBUS protocol, the X8R receiver transmits command packets to the MCU at 100Hz, providing a responsive and reliable connection for seamless user control.

This communication and interfacing strategy offers reliable data transmission, streamlined connectivity, and ease of use, making it a suitable choice for Otto projects.

4.2. Power Architecture

Otto's power system architecture ensures efficient energy distribution and optimal performance. A 6S (22.2V nominal) 22000mAh LiPo battery is the primary power source, powering 24V BLDC motors through motor controllers and stepping down the voltage to 12V and 5V for low-voltage electronics via buck converters.

The low-voltage components include motor encoders, LEDs, current sensors, STM32F4CCU6 microcontrollers for onboard systems monitoring, and a comprehensive e-stop system. A 12V to 230VAC 300W inverter is integrated to provide AC power compatible with the laptop charger, enabling maximum processing output from the CPU and GPU.

The power distribution board developed in-house by the team effectively distributes power to each component. It has an onboard INA266 Current and Voltage Sensor for power monitoring, which enhances the overall reliability of the Otto’s electrical system. This entire setup has been highlighted in ***Figure 12***.

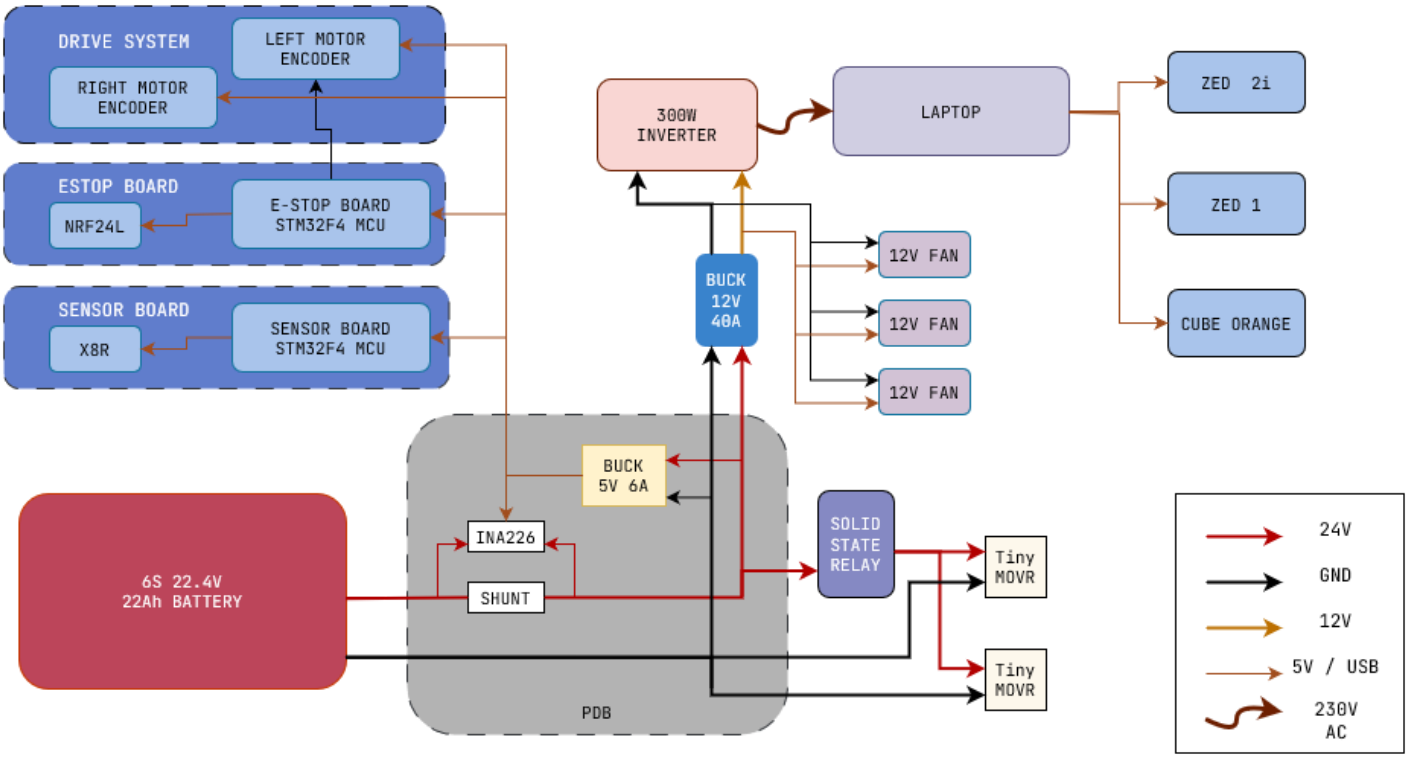

Figure 12: Power Architecture

4.3. Safety Features

The primary function of the emergency systems is to protect Otto and the surrounding environment in case of a system failure or the occurrence of any unexpected or unplanned event. We've implemented multiple measures to ensure the safety of everyone around the bot.

4.3.1. E-Stop

Otto has an Emergency-Stop (E-Stop) system that immediately halts the drive system by cutting power to the motor controller through a solid-state relay. This relay is controlled by the STM32F4 controller present on the E-Stop board.

The E-Stop can be triggered in three ways: mechanically, wirelessly, or by software. The mechanical E-Stop is a button placed in an accessible location on Otto for instant deactivation in case of unexpected behavior.

The wireless E-Stop allows remote operation and additionally manages a latch lock on the chassis for extra safety. It communicates with the E-Stop board using an NRF24F radio pair. ***Figure 13*** shows the E-Stop systems architecture.

We have also implemented a software-based E-Stop system that activates automatically under specific scenarios, such as if Otto tips over or receives unreasonably high velocity commands (determined by thresholds). These safeguards enhance Otto's safety and reliability.

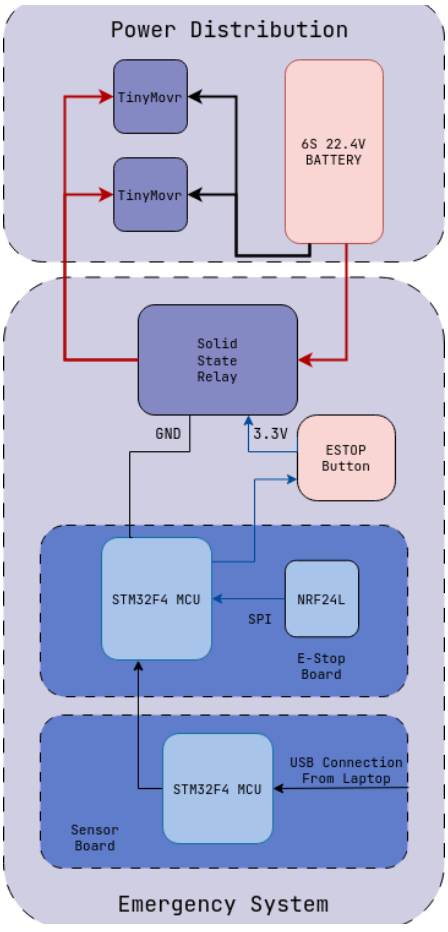

Figure 13: Emergency System Overview

4.3.2. Power and Temperature Monitoring

An Onboard Systems Monitoring feature managed by an STM32F4 MCU on a sensor board communicates with the main laptop via USB for seamless integration with the ROS framework. To maintain safe internal temperatures, a thermal monitoring system with three thermistors provides real-time data on critical components like motors and batteries. Air cooling is achieved using three 12V fans, with two for intake and two for exhaust. The PDB is equipped with a high current power monitoring INA226 ic, which measures critical stats like current drawn, bus voltage, and charge left in the battery. The IC is connected to the sensor board using an i2c bus, which relays this information through ROS.

4.3.3. Safety Features of Motor Controller

Our motor controllers and software integration packages have multiple safety features to ensure reliable and safe performance. Incorporating a software watchdog timer promptly halts the driver system in case of a disconnection between the motor controllers and the Laptop, preventing potential damage. We have also integrated overcurrent and overspeed protection that prevent the motors from exceeding their rated current or maximum permissible rpm, maintaining system stability and integrity in case of failures. Our controllers also feature regenerative braking capabilities, which direct the back current generated during braking back into the battery pack, thus recharging it and protecting the motors from high back currents. Otto is fitted with a set of signal LEDs to indicate the running status of the bot.

5. Software & Mapping Strategy

5.1. Overview

Our software system is built upon the Robot Operating System (ROS). We use a stereo camera, a ZED2i, as our primary source of visual and depth input. The point cloud generated by the same allows us to detect and track obstacles accurately.

The images from the cameras are used for lane and pothole detection. The lanes are mapped as obstacles, ensuring Otto stays within bounds at all times. Optimal path and trajectory algorithms then calculate the route of the bot and the velocity commands to be passed onto the motors. The same is summarized in ***Figure 14***.

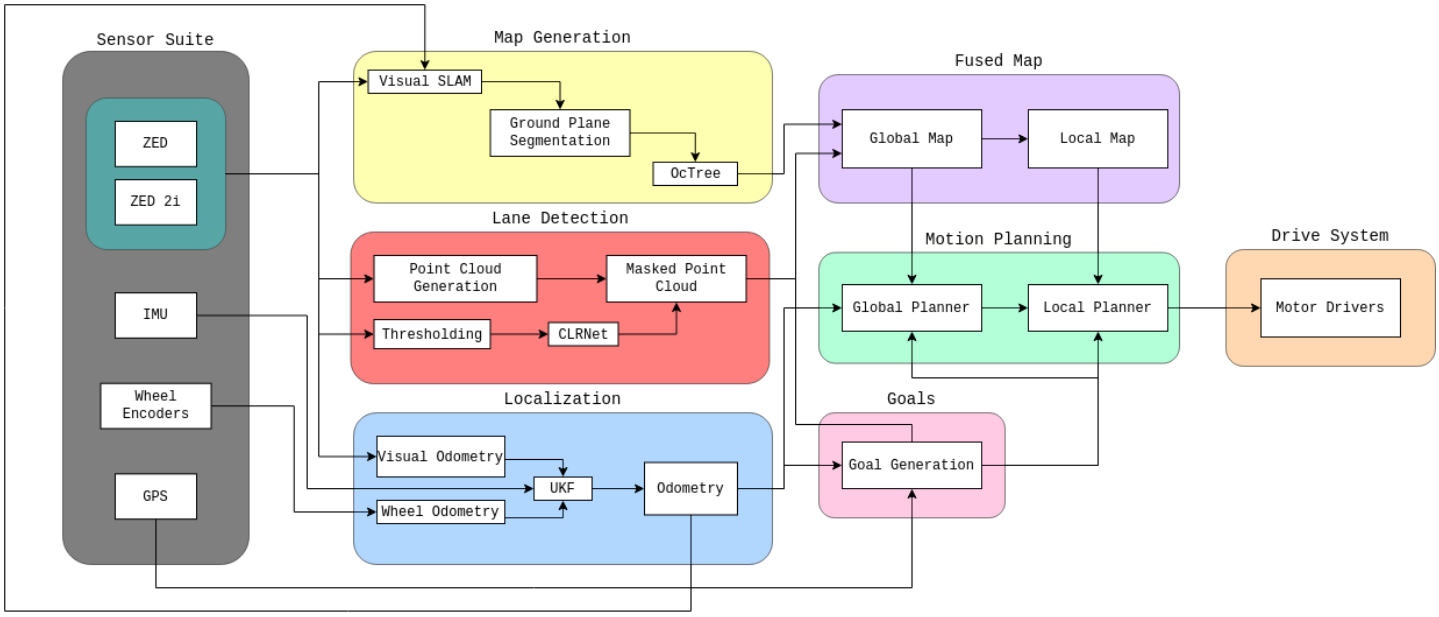

Figure 14: *Software Overview*

5.2. Perception

5.2.1. Lane & Pothole Detection

Deep Learning based solutions for Lane Detection are excellent at detecting lanes and are extremely robust if trained well. We used CLRNet, a novel approach to Lane Detection. A key advantage of CLRNet is its ability to distinguish between lanes and obstacles, allowing for the extrapolation of lane lines even when obstacles are obscuring vision.

Using custom datasets for lanes per the IGVC rulebook for training allowed us to tailor the training data to our specific use case and improve the model's performance for Otto. However, a drawback of CLRNet and Deep Learning-based solutions, in general, is that they require

large amounts of training data specific to the lanes to be used in the IGVC course.

To resolve this, we employed a traditional thresholding technique to get an attention map that our deep learning model can use to make predictions. ***Figures 15, 16, and 17*** show sample detections.

This fusion of conventional and deep learning approaches allowed us to create a robust, generalizable pipeline that also allows for advanced features like lane extrapolation without the required training data.

Pothole Detection is treated as a problem of detecting ellipses, as circles appear to be ellipses from Otto's perspective. Ellipses are fit on contours in the image by using the least squares method to find the ellipse parameters and the orientation of the ellipse. To get the final detections these ellipses are filtered out based on how well they fit the contours.

5.2.2. Obstacle Detection

The ZED provides accurate visual and depth information for a range of up to 130ft (40m), which generates a point cloud of the surrounding environment. The ground is segmented out from this point cloud using iterative random sampling, leaving only the obstacles.

5.2.3. Post Perception

The data obtained from the sensors need to be processed before mapping can begin. Since lanes and potholes are detected within an image, they are overlaid with the point cloud generated from the ZED to obtain their positions in 3D space.

Figure 15: *Original Image*

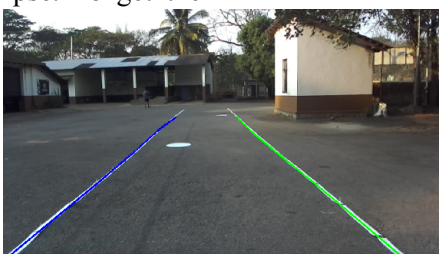

Figure 16: *CLRNet Predictions*

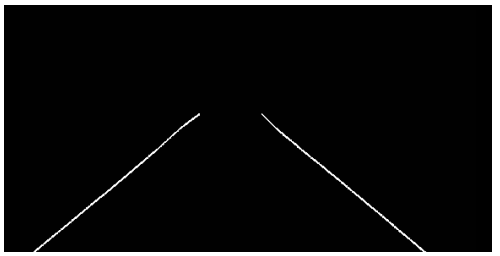

Figure 17: *Final Binary Mask*

5.3. Localization

Localization is the process of determining a robot's position and orientation within its environment. This process is vital for completing the IGVC mission successfully.

Otto utilizes sensor fusion to achieve accurate localization. A combination of wheel odometry, Inertial Measurement Unit (IMU) data, and visual odometry are used to estimate pose.

Wheel odometry is obtained from encoders on the Brushless DC (BLDC) motors. The kinematics of a two-wheel differential drive system is used to calculate this local odometry. In parallel, IMU data from the Flight Controller is acquired, providing additional information about Otto's movement in 3D space.

Visual odometry is a crucial component of our localization strategy. We estimate motion using the ZED 2i stereo camera based on the visual input from consecutive camera frames. This process allows us to determine changes in Otto's position and orientation over time.

These diverse data sources are fused using an Unscented Kalman Filter (UKF) [3]. UKF's superior performance in handling non-linearities allows for a more precise and stable estimate of Otto's state. Additionally, the UKF effectively manages noise in the sensor readings, ensuring the reliability and robustness of our odometry data.

5.4. Mapping

5.4.1. Adding Obstacles to Map

The obstacle data is first transformed from a point cloud into an Octree [4]. This representation involves a tree of nodes, each with eight child nodes, representing the eight smaller cubes derived from cutting the large parent cube in half along the three axes. By limiting the depth of the Octree, this representation allows for dynamic resolution at different parts of the map, vastly improving operational efficiency.

The tree's last (leaf) nodes are given a probability of occupancy based on the obstacle information obtained from ZED. This probabilistic approach completely solves the problem of sensor noise appearing in the final map.

Additionally, when all children of a node reach a certain threshold occupancy, they can be destroyed or pruned from the tree. This optimization massively saves memory. If new data necessitates updating these nodes, they can be regenerated with the same threshold occupancy as their destruction.

5.4.2. Adding Lanes to Map

To enforce Otto's adherence to the lane boundaries, we project the detected lanes onto the cost map as well. To achieve this, a new Octree, similar to the one described in ***Sec 5.4.1***, is initialized. As lanes are continually detected, the Octree updates the probabilities of the relevant nodes accordingly.

This method ensures that any noise encountered due to vibrations or jerky movements does not make it to the final map, as such points will not cross the minimum occupancy threshold. ***Figure 18*** summarizes the pipeline

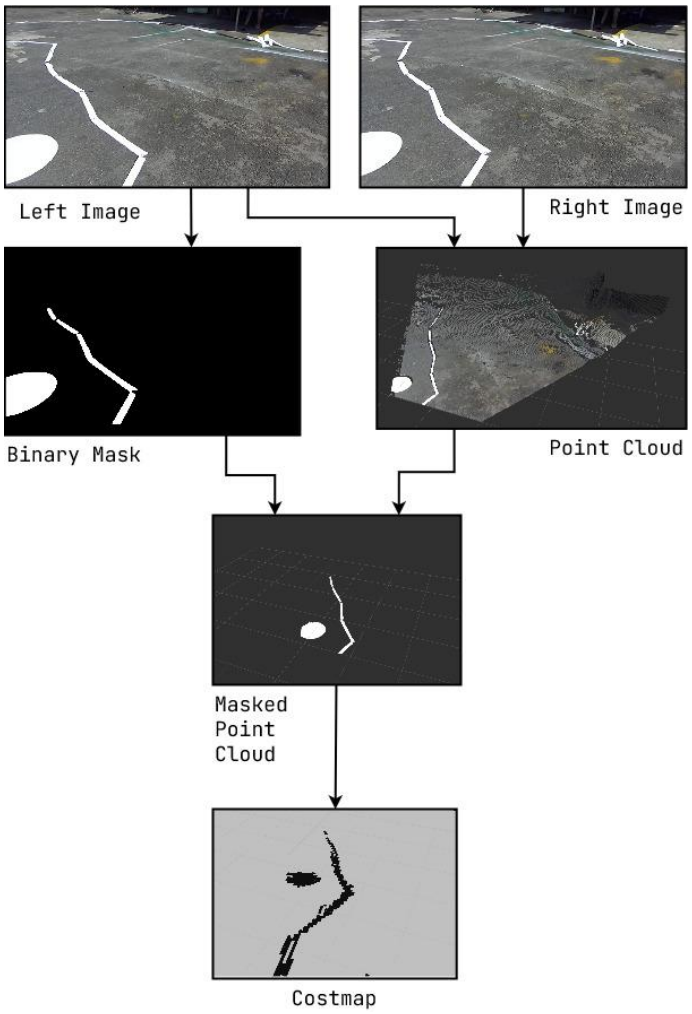

Figure 18: *Lane Detection to Costmap Pipeline*

5.4.3. Map Generation

The Octrees consisting of lane and obstacle data are combined into one. Using this combined Octree, two maps are generated. These maps represent a birdseye view of the course, detailing the positions of all lanes and obstacles detected up to the present.

The first of these two maps is the global map, which is a large, low-resolution map covering the entire course area, showing the positions of large, obvious obstacles. The second map is the local map, a small, constantly updating map representing only the current environment at a very high resolution. This is used for immediate trajectory planning.

For robustness, we modified the algorithm to constantly update the path within the Field of View of the ZED2i, even on the global plan, to ensure the route always avoids all obstacles close to the bot.

This ensures that any noise on the cost map is filtered away as soon as the sensors notice a discrepancy.

5.5. Goal Calculation

5.5.1. Lane Following

A weighted average of the detected lanes provides a means to calculate goals within the bounds of the lanes. This method assigns a higher weight to points that are farther away, thereby generating suitable goals aligned with the center of the lane.

In the event of only one lane being detected, this approach encourages Otto to follow a trajectory close to the detected lane, which reduces the probability of it deviating from the course.

If neither lane is detected, the next goal is computed using GPS waypoints, ensuring Otto continues navigating toward its intended destination.

5.5.2. Waypoint Navigation

Once Otto is within a threshold distance of its current GPS goal, the position of the next GPS point is computed with respect to its current position. This goal is recalculated at regular intervals, and the results are passed through an Unscented Kalman Filter to correct for drift, if any, in the GPS readings.

5.6. Motion Planning

5.6.1. Path Planning

The NavFN planner generates a global plan for the current goal. This planner uses a modified version of Dijkstra's algorithm, which finds the shortest path between any two vertices on a graph while remaining cost-efficient.

5.6.2. Trajectory Optimization

The global plan generated by NavFN only roughly outlines the general path to be followed. It does not take into account smaller or moving obstacles. We use the Timed-Elastic-Band (TEB) [5] controller to optimize this plan. It uses the local plan to generate a time-optimal trajectory to achieve a local goal that lies along the global plan. The velocity commands calculated by TEB are then passed onto the motor controller to make Otto move.

5.7. Experimental Concepts Tested

5.7.1. Using Point Cloud for Mapping

Inverse Perspective Mapping (IPM) has been widely employed to estimate depth in computer vision applications. While IPM certainly has its merits, we decided to overlay our 2D image over a 3D point cloud. Since both of these are linearly ordered in the same manner, a simple calculation converts the 2D coordinates of a pixel in the image to the index in the point cloud. This approach provides several benefits over traditional IPM.

First and foremost, IPM assumes that the plane being transformed is flat and parallel to the ground, which is not always true. Secondly, it is computationally expensive. Third, any calibration for IPM only yields one view. Overlaying the image allows for extracting the desired view.

5.7.2. State-based Lattice Planner

We evaluated a State-based Lattice-based planner [2], which generates a discrete lattice of paths based on the vehicle's kinematics and utilizes algorithms like A* or Dijkstra's to find the optimal path. However, it was determined that the computational demands of this planner were excessively high for the differential drive robot in use. Furthermore, considering the Otto's specific requirements, a higher update rate would be more advantageous.

6. Failure Identification & Mitigation

Failure Mode/Point	Description	Mitigation
High Curvature Paths (Software & Mapping)	CLRNet had difficulty detecting lanes with high curvature.	Complemented CLRNet with color thresholding-based lane detection to improve performance.
Camera Blind Spot	Single ZED 2i camera had a limited field of view, creating a blind spot.	Added a second camera to cover the blind spot, enhancing Otto's perception capabilities.
Structural Failures	Multiple structural failures, including bearing load issues, heavy sprockets, and loosening bolts.	Changed the loading condition of the shaft, optimized the shape of the sprockets, and used Loctite 243 for securing the bolts.
Control Issues from Chain Drive Mechanism	Challenges in control due to chain slack and related issues introduced by the chain drive mechanism.	Implemented an adaptive feedforward control strategy and meticulous tuning of the PI controller parameters and integrator deadband for more precise and responsive control.

Table 1: *Failure Identification & Mitigation*

7. Simulations

7.1. Overview

Our team employs Gazebo, a simulation platform distinguished by its realistic physics and strong ROS support, for the robust testing and refinement of algorithms pre-deployment.

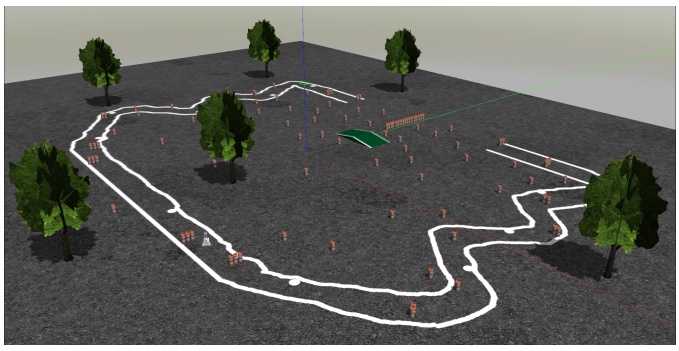

Figure 19: *Simulation World*

To ensure the simulated Otto accurately reflects the actual vehicle, we developed an exact model incorporating differential drive mechanics from ANSYS data. This model includes precise dimensions, mass, and inertia, facilitating optimal sensor positioning. We further integrated Gaussian noise into the sensor data to emulate real-world readings.

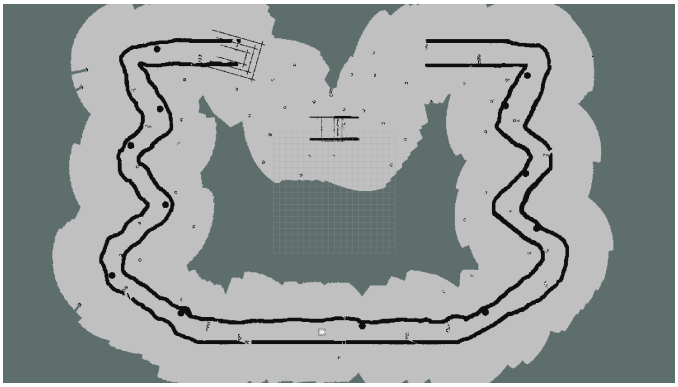

Figure 20: *Map Generated from Simulation*

Adhering to IGVC 2023 rules, we crafted a life-sized, realistic simulation environment (***Figure 19, 20***) featuring asphalt ground, obstacles, a ramp, and a layout mimicking the IGVC course. This enabled the software team to simultaneously develop and test the logic of their codebase while the mechanical subsystem was engaged in Otto's design, construction, and modifications.

7.2. Concepts Implemented

7.2.1. Local Planner Selection

The simulation environment was instrumental in evaluating the Timed-Elastic-Band (TEB) and Dynamic Window Approach (DWA) [6] local planners under various edge

cases simulating the IGVC conditions. We assessed these against metrics like path quality, computational load, trajectory smoothness, and responsiveness. TEB outperformed in all metrics bar computational complexity. Hence it was selected, as computational complexity was not a limiting factor given our adequate computing resources.

7.2.2. Simulation-Aided Detection Optimization

The simulation also proved invaluable for refining our lane detection approach. By testing various strategies, we found that augmenting CLRNet with thresholding best handled extreme curves potentially present in the IGVC course, resulting in robust performance even with locally-collected, unpredictable data.

7.2.3. Goal Publishing Methodology Selection

Finally, the simulation facilitated the exploration of diverse methods of goal publishing, allowing rapid experimentation and iteration. This contributed to identifying and implementing the most effective approach for Otto's precise navigation.

8. Performance Testing

8.1. Mechanical Testing

Otto underwent rigorous mechanical testing to ensure its structural integrity and durability. Factors like the weight-bearing capacity, traction of wheels, and resilience of the sprocket-chain power transmission mechanism were scrutinized. Furthermore, the weatherproofing measures, such as the hydrophobic paint and the Loctite on fasteners, were tested against various humidity conditions.

We physically tested our design against both Free and Forced vibration conditions with an ADXL345 IMU, the physical tests showed the sensor pole's natural frequency to be 55Hz without carbon fiber rods and 110Hz with carbon fiber rods. This implied that the precautionary measures taken sufficiently drifted the natural frequency of the sensor pole from the operational frequency.

8.2. Electronic Testing

The electronic systems of Otto, including the Cube Orange IMU/MCU, Here3+ GPS, ZED 2i and ZED 1 stereo camera, Tmovr R5 motor controller, and the custom-designed PCBs, underwent thorough testing to ensure reliable operation. The power system, which includes the 22.2V, 22,000 mAh LiPo battery, and the inverter for the laptop, was tested for longevity and consistency. The CAN bus interface was tested for efficient and reliable communication.

8.3. Software Testing

We employed Gazebo simulations extensively for preliminary software testing, ensuring the navigation algorithms, sensor fusion logic, and control schemes performed as expected. In addition to this, real-world testing of the software was also undertaken, where Otto was placed in various real-life scenarios, and its software's performance was closely monitored and evaluated.

8.4. Mission Testing

During the final testing stage, mission testing was conducted to evaluate Otto's performance in both simulated and real-life environments as shown in ***Figure 21***. The testing combined all systems, including the ability to navigate a simulated IGVC course, avoid obstacles, stay within lanes, and accurately reach GPS waypoints. Real-world field tests were also conducted to assess performance in unpredictable environments and the ability to tackle real-world challenges. The goal was to ensure that Otto is fully equipped to handle any situation it may encounter in its operation.

Figure 21: *Real-Life Mission Testing Course Setup*

9. Initial Performance Assessment

As we prepared Otto for the IGVC competition, we initiated a comprehensive performance assessment to gauge its readiness and efficiency in real-world conditions.

This involved creating a test environment that closely mimics the IGVC course setup outside our workshop. The constructed environment included a green-painted ramp, lanes demarcated with tape, and cones serving as obstacles.

In the initial segment of the test course, Otto displayed adept lane-following and obstacle-avoidance capabilities. Upon reaching a predetermined GPS waypoint, signaling the end of this segment, Otto smoothly transitions to the 'no man's land'. Here, it relied on waypoint navigation to travel to the ramp's start, ascend and descend the ramp, and then make its way to the end of 'no man's land', all the while avoiding obstacles. Guided by another GPS waypoint, Otto

seamlessly re-enters the lane-following segment and navigates to the end of the course. Our consistent testing efforts have resulted in four successful full-course runs before the submission of our TDR. Each run offers invaluable insights, allowing us to fine-tune our systems and resolve any arising issues. As we continue this iterative refinement in the run-up to the competition, we are confident of presenting a robust Unmanned Ground Vehicle, fully capable of flawlessly executing the Auto-Nav Challenge at IGVC 2023.

10. Scope for Further Development

Area	Improvement	Description
Software	Reinforcement Learning-Based Control	Reinforcement Learning (RL) could enable our UGV to improve its decision-making capabilities over time. The UGV could autonomously develop optimal control policies, enhancing its ability to navigate complex and uncertain environments.
Software	Deep Learning-Based Pothole Detection	Implementing a deep learning-based approach specifically for detecting potholes would allow us to detect potholes in more dynamic scenarios, making it more robust.
Software	Transition to ROS2	ROS2 provides better support for real-time control, and enhanced security.
Software	Dynamic Obstacle Avoidance	Dynamic obstacle avoidance can enhance Otto's real-time adaptability and safety. This will allow Otto to intelligently navigate unpredictable environments by adjusting its path based on moving obstacles.
Electronics	CAN Bus Integration	Integrating all electronics onto a single CAN bus could streamline communication between components, simplify the system's overall architecture, and potentially improve reliability.
Mechanical	Enhanced Dust Resistance and Fan Sealing	Implementing dust resistance features and sealing the fans could prevent particulate matter from entering the UGV, potentially improving the longevity of components and the system's overall reliability.
Mechanical	Laptop Placement	Reconsidering the placement of the laptop could enhance accessibility, making it easier for operators to interface with the UGV, making debugging easier
Mechanical	Payload Accessibility	Improving payload accessibility could ease the installation and removal of equipment, speeding up setup and teardown times.

Table 2: *Overview of Scope for Further Development*

11. References

[1] Zheng, T., Huang, Y., Liu, Y., Tang, W., Yang, Z., Cai, D., & He, X. (2022). CLRNet: Cross-Layer Refinement Network for Lane Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 898-907).

[2] S. Pancanti, L. Pallottino, D. Salvadorini, and A. Bicchi.Motion planning through symbols and lattices. In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA), 2004.

[3] E. A. Wan and R. Van Der Merwe, "The unscented Kalman Filter for nonlinear estimation," in Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373). IEEE, 2000, pp. 153–158

[4] Tatarchenko, M., Dosovitskiy, A., & Brox, T. (2017). Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs. In IEEE International Conference on Computer Vision (ICCV).

[5] C. Rösmann, F. Hoffmann and T. Bertram: Kinodynamic Trajectory Optimization and Control for Car-Like Robots, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, Sept. 2017.

[6] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," in IEEE Robotics & Automation Magazine, vol. 4, no. 1, pp. 23-33, March 1997. doi: 10.1109/100.580977