Oakland University IGVC 2026 - Team Smart Vehicles Club

Group Info

University/College Name:	**Oakland University, Rochester, MI 48309, USA**
Vehicle/Team Name:	**TurboMind2**
Date Submitted:	**15 May 2026**
Targeted Challenge:	**Self Drive**

Team Contacts

Role	Name	E-Mail	Phone
Team Captain (Main Contact)	Zhaodong Zhou	zhaodongzhou@oakland.edu	248-883-2814
Key Team Member (Secondary Contact)	Ali Irshayyid	aliirshayyid@oakland.edu	248-759-1853
Faculty Advisor (Tertiary Contact)	Dr. Jun Chen	junchen@oakland.edu	248-370-4797

Team Members		
Zhaodong Zhou	**Team Captain**	Keer Chen
Ali Irshayyid		Billur Haskara
Briana Popa		Neha Bhatt
Ronald Shoebottom		Nolan Mittison
Jonathan Nathan		Jordan Yang
Gjergj Kola		

Statement of Integrity:

As the faculty advisor, I certify that the design and engineering of the TurboMind2 self-drive vehicle by the currently listed student team has been significant and equivalent to what might be awarded credit in a senior design course. _____ 5/15/2026

1 System and Subsystem Requirements

The design requirements were identified from the IGVC Self Drive rules and the practical constraints of converting a ride-on vehicle into an autonomous platform.

The resulting requirements, what drove the requirement, measurement methods, and target values are summarized in Table 1.

2 Mechanical Design

The autonomous vehicle platform was developed from a commercially available children's ride-on vehicle chassis. The platform was selected due to its compact size and integrated drivetrain. Significant modifications were made, including upgrades to the rear drive system, steering system, wheel assemblies, sensor mounting structures, and electrical integration.

2.1 Drive System

The original plastic wheels were replaced with 13.5 in pneumatic rubber tires to improve traction. Because the pneumatic wheel bore exceeded the axle dimensions, custom wheel shims were fabricated, as seen in Fig. 1, to improve wheel centering and reduce vibration.

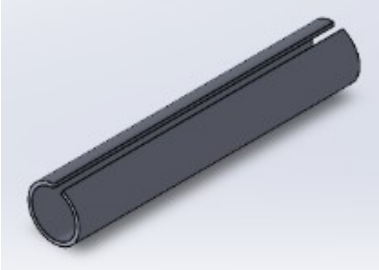


Figure 1: Wheel shims CAD model.

The propulsion of the vehicle is done by two rear motors. The original motors that came with the vehicle were replaced with motors that had encoders and could achieve higher speeds with the original gearbox.

To connect the wheels to the gearbox, a wheel adapter was designed, as seen in Fig. 2.

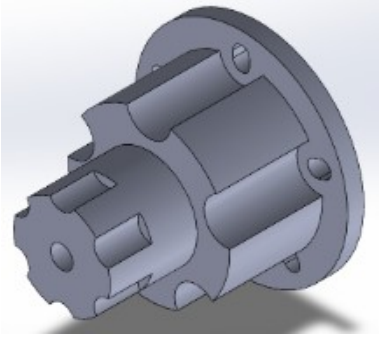


Figure 2: Rear wheel adapter CAD model.

Table 1: System and subsystem requirements for the Self Drive Challenge.

Category	**Requirement**	**Driven By**	**Measurement Method**	**Target Value**
Mechanical	Vehicle size compliance	IGVC parameters requirements	Tape-measure inspection	3–7 ft length, 2–4 ft width, <6 ft height
Mechanical	Steering and turning capability	IGVC course geometry and general maneuverability needs	Full-left/full-right turning-radius test	Turning radius $\leq$ 5 ft
Safety	Hardware E-stop functionality	IGVC safety rules	Activate mechanical and wireless E-stops during motion	Quick and complete stop
Safety	Vehicle state indication	IGVC safety light rules	Observe safety light when powered, and in manual or autonomous mode	Solid powered, flashing autonomous, solid after exit
Electrical	Protected onboard power distribution	Vehicle onboard power and subsystem protection needs	Wiring inspection and powered subsystem test	Onboard battery power with fused/protected distribution to major subsystems
Electrical	Vehicle speed control	IGVC speed limits	Timed run over known distance	>1 mph and $\leq$5 mph
Perception	Lane-marking segmentation	Self Drive lane keeping and intersection	Live camera-frame testing	Detect solid, dashed, and stop bar lane markings reliably
Perception	Obstacle detection	Self Drive obstacles, pedestrians, and stop signs	Camera/LiDAR detection tests	Detect relevant course objects in vehicle path
Driving Logic	Lane-following control	Self Drive lane navigation	Run vehicle through marked lane	Maintain centered lane motion
Driving Logic	Obstacle response behavior	Self Drive obstacle avoidance/road rules	Place obstacle in path and observe behavior	Stop or avoid obstacle safely
KPI	Lane-keeping performance	Self Drive qualification/function test	Measure lateral deviation during lane run	No boundary crossing; lateral deviation <2 ft
KPI	Runtime capability	Competition operating needs	Estimate runtime from power draw and battery capacity	Runtime $\geq$2 hours

Finite element analysis (FEA) was performed on the wheel adapter using the estimated maximum torque applied through the gearbox, as shown in Fig. 3. The wheel adapter was manufactured from ABS-M30 due to its strength and availability for 3D printing. The material yield strength is approximately 30 MPa, while the maximum stress observed in the FEA was 4.2 MPa, giving the rear wheel adapter an estimated safety factor of approximately 7.

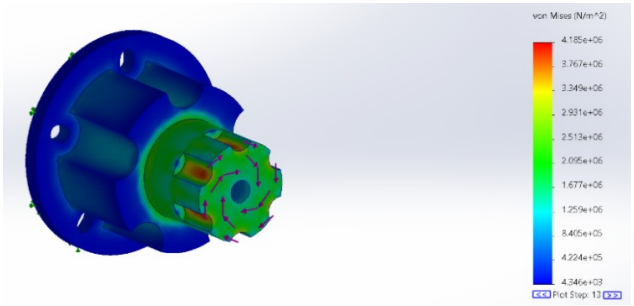


Figure 3: FEA analysis of the wheel adapter.

Stopping performance was evaluated to verify that the vehicle could reach a full stop within 3 ft. Using the maximum operating speed of 5 mph, the calculated stopping distance was approximately 0.64 ft, which is below the 3 ft target distance. This estimate is conservative because it does not take into account the friction loss from the tires or the gearbox.

2.2 Steering System

The steering system was redesigned to increase the steering angle and allow a 5 ft turning radius. To increase steering capability, the steering lever arm was extended using a fabricated steel plate. The added plate can be observed below in Fig. 4.

Figure 4: Steering system with extended lever arm.

Additionally, a custom steering coupler and motor mounting plates were designed to connect the servo to the steering shaft and align the actuator with the steering linkage. FEA was performed on the steering shaft coupler, shown in Fig. 5, giving an estimated safety factor of approximately 2.5.

A steering shaft connected to the servo passes through the slot in the extended steering arm, which is attached to the vehicle's original push-pull linkage. This allows servo motion to rotate the front steering assembly. The increased lever arm also increased the steering torque required

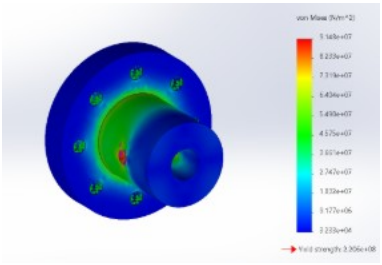


Figure 5: FEA for steering shaft coupler.

from the actuator. The actuator was therefore replaced with a higher-torque servo motor capable of meeting the estimated steering torque requirements.

2.3 LiDAR and Camera Mounts

The LiDAR was mounted using a ready-made adjustable laptop stand. This mount was selected because it was simple to attach to the vehicle, provided sufficient support for the LiDAR with an estimated safety factor of 3, and had a plate that matched the dimensions of the LiDAR mounting bracket. The use of an off-the-shelf mount reduced fabrication time while still providing a stable platform for the sensor.

The front camera was also mounted to the laptop stand using a custom 3D-printed bracket, as shown in Fig. 6. This location was selected as it provides optimal field of view for lane line perception while keeping the camera aligned with the LiDAR. A additional side camera was mounted along the side of the vehicle, shown in Fig. 7, to provide supplemental camera coverage. CAD models for the front and side camera mounts are shown in Fig. 8 and Fig. 9, respectively.

Figure 6: LiDAR and front camera mount.

Figure 7: Side camera mount.

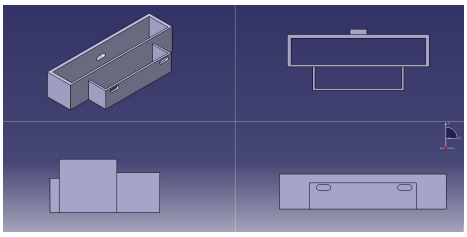


Figure 8: Front camera mount CAD model.

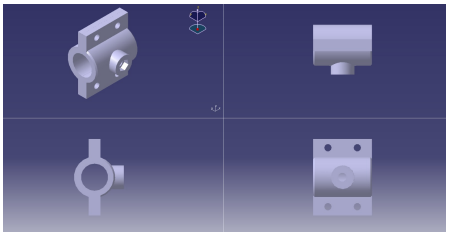


Figure 9: Side camera mount CAD model.

3 Safety

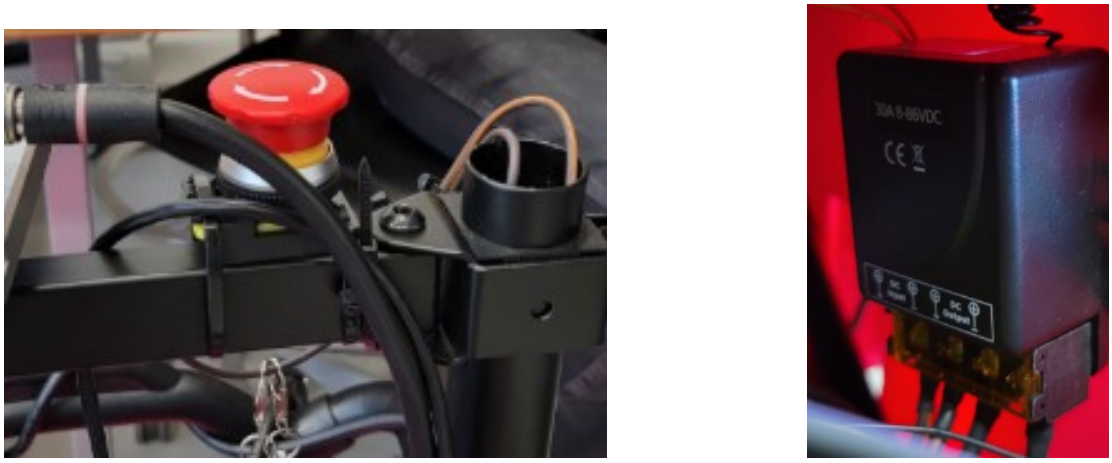
3.1 Transport, Parking, and Charging Safety

During transport, the battery is disconnected and stored separately from the vehicle to prevent the system from turning on and to minimize sliding. Additionally, the vehicle is secured using elastic cords wrapped around the rear wheels and axle.

When parked or stored, the battery remains disconnected from the vehicle power distribution system. If it is connected, all power is distributed with a fused power hub providing overcurrent protection. During charging, the battery is disconnected and removed from the vehicle and charged using the manufacturer-provided charger.

3.2 Operational Safety Systems

While operating on the course, the major safety mechanisms are the mechanical and wireless emergency-stops (E-stop), shown in Fig. 10. Both E-stops are wired to immediately disable propulsion power, allowing the vehicle to come to a quick and complete stop.



(a) (b)

Figure 10: (a) Mechanical emergency-stop mounted at the rear of the vehicle and (b) wireless emergency-stop receiver mounted within the vehicle.

The mechanical E-stop is red in color, push-to-stop, greater than 1 inch in diameter, and mounted at the rear of the vehicle between 2 ft and 4 ft above ground level satisfying the IGVC requirements. The wireless E-stop system includes two remotes and was verified to operate well beyond the required 100 ft range.

The vehicle also includes a visible LED safety indicator to communicate the current operating state. When the vehicle is powered on, the safety light remains solid. During autonomous operation, the light flashes to indicate autonomous mode. Once autonomous operation is disabled, the light returns to a solid state.

4 Electrical/Electronic Design

4.1 Electrical Architecture and Power Distribution

The vehicle power system is centered around a 24 V, 30 Ah Bioenno $LiFePO_4$ battery. The battery feeds a Powerwerx PD-5F fused distribution hub through an SB50-to-PP45 adapter, allowing the main battery connection to be made without cable splicing. The fused hub provides individually protected outputs for major electrical branches and improves wiring organization. The vehicle power distribution architecture is shown in Fig. 11.

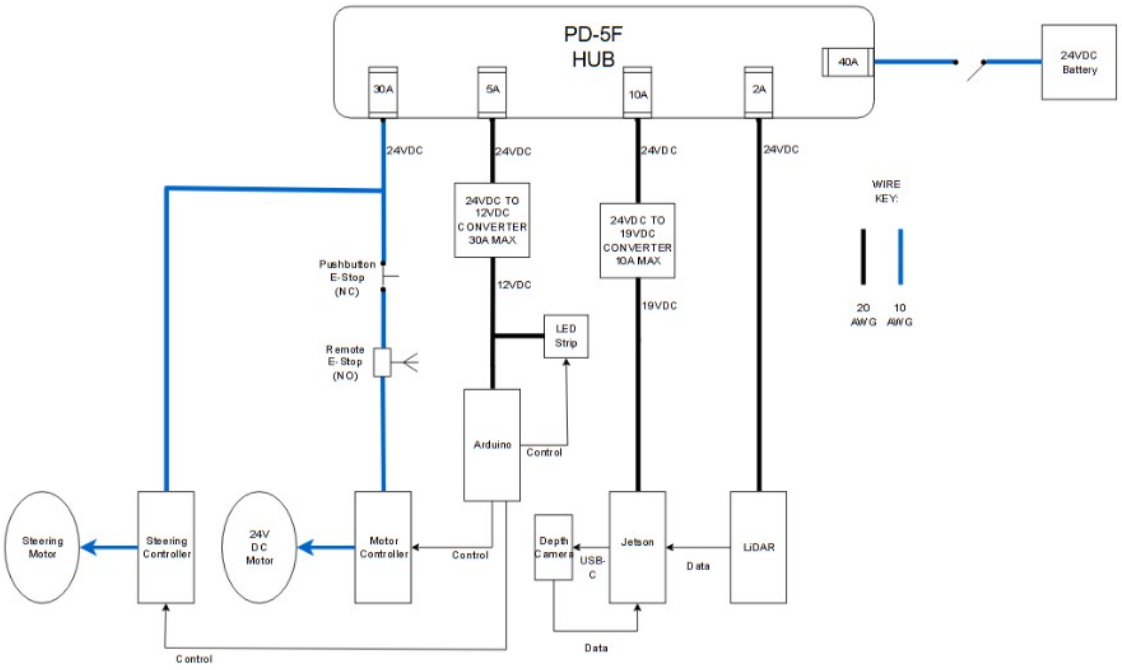


Figure 11: Vehicle power distribution architecture.

A 24 V terminal block was used to distribute power to multiple subsystems, simplifying wiring connections and improving organization of the power distribution system. This includes the Ouster OS1 LiDAR, rear propulsion motors, and steering servo.

Commercial DC-DC converters were used to generate 19 V and 12 V. The 19 V converter supplies the Jetson, and the 12 V converter supplies the Arduino and safety lighting. Bench testing was performed on the 24 V-to-19 V and 24 V-to-12 V converters under varied input and load conditions. Both converters maintained stable output voltages during individual and simultaneous operation. After full system integration, the vehicle electronics operated without overheating or power instability during testing.

Based on the estimated subsystem power consumption, the total nominal load is approximately 242 W. Using an 85% derating factor on the 720 Wh battery capacity gives an estimated usable capacity of 612 Wh and an estimated runtime of approximately 2.5 hours. This exceeds the two-hour runtime target for competition operation.

4.2 Vehicle Control and Actuation Electronics

The Arduino Mega 2560 REV3 serves as the low-level controller. It was selected due to its large number of digital and analog I/O pins, allowing simultaneous interfacing with the steering motor driver, propulsion motor driver, wheel encoders, limit switches, LED indicators, and Bluetooth communication module. The Bluetooth module allows for manual vehicle control through a mobile application and uses the Arduino's UART interface and 5 V supply rail.

The mechanical and wireless emergency-stop systems discussed in the prior section both have an operating voltage of 24 V. The mechanical E-stop is wired in series with the wireless E-stop and the 24 V supply feeding the propulsion motor driver. As a result, actuating either E-stop immediately removes power from the propulsion system while allowing sensitive electronics, such as the onboard GPU and sensors, to remain powered. The wireless E-stop operates in latched mode, meaning propulsion power toggles between enabled and disabled states through the remote transmitter.

The propulsion subsystem consists of two rear-mounted 24 V brushed DC motors driven in parallel by a single high-current motor driver. The selected motors include integrated encoder feedback and provide sufficient torque and speed capability for vehicle operation within IGVC limits. The propulsion system is capable of successfully achieved and maintained both the minimum competition speed requirement of 1 mph and the maximum allowable competition speed of 5 mph. Each motor has a rated current of approximately 6.5 A with significantly higher transient stall currents, requiring a motor driver with sufficient current margin.

A single-channel motor driver was selected to drive both propulsion motors simultaneously. The use of a single driver simplifies the propulsion system and ensures that both rear wheels receive identical PWM and direction commands. This is optimal and safe as the rear motors should always remain synchronized since they are used only for propulsion. The selected motor driver supports 24 V operation and provides sufficient continuous and peak current capability for the combined propulsion load.

Although the propulsion motors are mechanically identical, they are mounted on opposite sides of the rear axle with opposing shaft orientations. As a result, one motor was wired with reversed electrical polarity so that both wheels rotate in the same direction under a shared driver command.

The steering subsystem is actuated using a high-torque servo motor connected directly to the steering shaft. The steering motor driver includes adjustable tuning potentiometers which were used to center the servo and limit the steering range to approximately $\pm 30^\circ$ from center.

To protect the steering system and plastic vehicle frame from over-rotation, mechanical limit switches were installed at the left and right steering extremes, preventing the steering system from exceeding the allowable steering range.

5 Software System

The software system is organized as a VLM-supervised hybrid architecture. High-level behavior planning is performed on the Jetson Thor using ROS 2, with a vision-language model (VLM) acting as the trajectory planner at approximately 4 Hz. Classical perception (YOLOP-based detection on the ZED-2i and DBSCAN clustering on the Ouster LiDAR) runs at (20-30) Hz and supplies both geometric and semantic information to the planner. Low-level motor and steering control is handled

by the Arduino-Mega, which receives steering and speed commands from a Pure-Pursuit tracker that follows the trajectory selected by the VLM.

The ZED 2i Stereo Camera uses a fine-tuned YOLOP model to detect and track people, extracting a list of detected objects with their respective spatial (x, y, z) coordinates. This model is trained to precisely detect stop signs, traffic barrels, and other necessary traffic classes.

The Ouster LiDAR publishes a dense 3D point cloud that provides precise 360-degree spatial scanning for obstacle localization and distancing. Prior to clustering, the full cloud is vertically cropped to keep data in relevant regions. DBSCAN density-based clustering segments clusters (obstacles) from noise, producing object candidates.

In order to map detected ZED objects to LiDAR clusters, they must be in the same frame. This relationship is expressed as the matrix T_{LC}, a 4×4 homogeneous transform defined as:

$$T_{LC} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}, \quad t = \begin{bmatrix} 0.05 \\ 0.0 \\ -0.1 \end{bmatrix} \tag{1}$$

where R is the rotation matrix between the Camera and LiDAR frames (an identity matrix, since both sensors share the same orientation), and t is the translational offset (+0.05m in X / forward of LiDAR, 0.0m in Y, −0.1m in Z / below LiDAR).

5.1 Perception Module

5.1.1 Dataset

Table 2 illustrates the distribution of a total of 4,870 labeled samples across training, validation, and test sets, categories in four main traffic objects: Stop sign, Traffic barrel, Car tire, and Pedestrian. All samples are collected in the COCO[1] dataset format, with each object category sourced individually as separate datasets via Roboflow[2]. However, since the YOLOP model expects data in the BDD100K[3] format, all samples were converted accordingly to maintain consistency with our annotation pipeline. The Stop sign category is presented in three subcategories during the detection, including correct, fake, and vandalized. The Traffic barrel class includes both standard barrels and traffic cones, which are visually similar but differ in structure and use. The Car tire category includes labels for tires attached to vehicles, specifically those visible in driving scenes or parked vehicles. Finally, the Pedestrian category is retrieved from the MOT17[4] dataset video frames and uniquely combines human and mannequin samples to increase robustness in pedestrian recognition. Each category is assigned a unique category ID ranging from 1 to 7, ensuring structured labeling for downstream tasks.

Additionally, we utilize the BDD100K [1] dataset for drivable area and lane line segmentation. BDD100K is a large-scale, multi-task dataset consisting of 100K driving images annotated for 10 diverse tasks. We follow the official split, using 70K images for training, 10K for validation, and

[1]COCO Dataset. Retrieved May 14, 2026, from `https://cocodataset.org/`
[2]Roboflow. Retrieved May 14, 2026, from `https://app.roboflow.com/`
[3]BDD100K Dataset. Retrieved May 14, 2026, from `https://bair.berkeley.edu/blog/2018/05/30/bdd/`
[4]MOT17 Dataset. Retrieved May 14, 2026, from `https://motchallenge.net/data/MOT17/`

20K for testing.

Table 2: Distribution of training, validation, and test samples in our generated traffic object detection dataset across various traffic object categories.

Type	Category	Train	Validation	Test
Stop sign	Stop-sign	456	131	65
	Stop-sign-fake			
	Stop-sign-vandalized			
Traffic barrel	Traffic-barrel	778	223	101
	Traffic-cone	558	165	86
Car tire	Car-tire	488	191	104
Pedestrian	Pedestrian	607	102	6

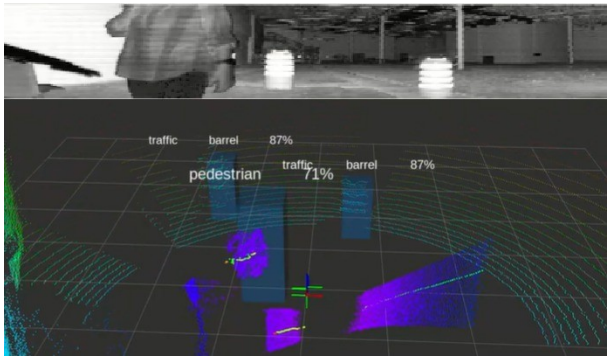
5.1.2 Model

In this project, we employed YOLOP [2] for object detection and U-Net [3] for lane line segmentation.

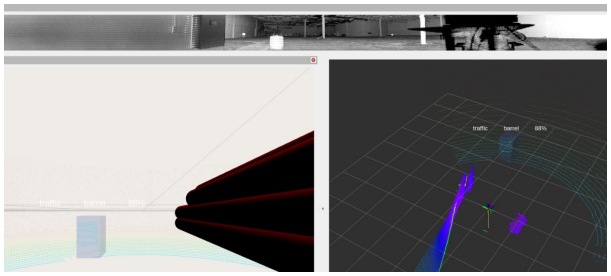
For object detection, the input image passes through a shared backbone for feature extraction; here, CSP-Darknet [4] is used as the backbone, which addresses the issue of gradient duplication during optimization. Then, these features are enhanced through a shared neck module. The neck consists primarily of a Spatial Pyramid Pooling (SPP) [5] module and a Feature Pyramid Network (FPN) [6] module. SPP generates and integrates multi-scale features, while FPN merges features from different semantic levels. As a result, the enhanced features contain information from multiple scales and semantic levels. Finally, features from different layers are directed to separate heads to perform their respective tasks. For lane line segmentation, we employ a U-Net architecture, which uses a symmetric encoder–decoder structure with skip connections between corresponding encoder and decoder stages. The encoder progressively downsamples the input image through successive convolution and pooling operations to capture contextual features, while the decoder upsamples the feature maps back to the original resolution to produce a pixel-wise segmentation mask. The skip connections preserve fine spatial details that would otherwise be lost during downsampling, which is particularly important for recovering thin, elongated structures such as lane markings. The output is a binary segmentation mask identifying lane line pixels, which is then post-processed to extract the left and right lane boundaries used by the trajectory planner.

5.2 Sensor Fusion

The fusion node combines the classified objects from the ZED camera with LiDAR clusters using the homogeneous transform T_{LC} (see Fig. 12). For each detected object, the fusion node publishes (i) the class label from YOLOP, (ii) the LiDAR-derived relative-distance to the vehicle, and (iii) a confidence score. By combining semantic labels with relative distance, the object list provides the scene information in addition to spacial clues needed by the VLM planner to select a trajectory. The camera provides recognition, while the LiDAR provides the distance values that the VLM cannot reliably infer from a monocular image alone.



(a) Ouster LiDAR point cloud.



(b) Fused ZED and LiDAR point cloud.

Figure 12: Visual comparison of LiDAR-only and sensor-fused point cloud representations.

6 Driving Logic

The team is targeting the Self Drive Challenge only. The driving logic is structured as a three-layer pipeline: (1) the fusion node supplies a list of detected objects with their relative distances, (2) the VLM planner selects a behavior-level trajectory at a ∼4 Hz; (3) a Pure-Pursuit tracker converts the selected trajectory into steering and speed commands at (20-30) Hz, which are sent to the Arduino over serial.

The planner maintains a fixed list of five candidate maneuvers, *straight*, *soft left*, *soft right*, *sharp left*, and *sharp right*, each defined as a sequence of waypoints in the vehicle frame. At each planning cycle, all five trajectories are projected into the current camera image and overlaid with labels (1) through (5) (see Fig. 13).

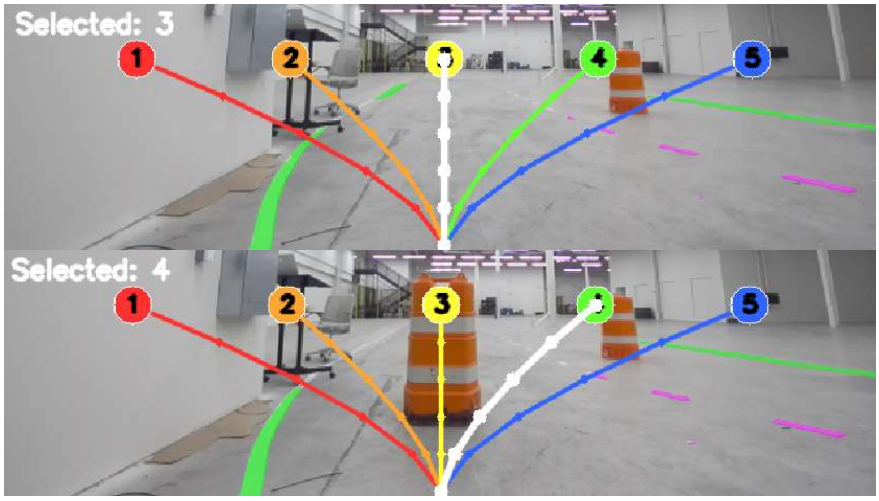


Figure 13: VLM selected trajectory given an input camera frame.

The VLM receives two inputs per query:

1. The annotated camera frame showing the five labeled candidate trajectories overlaid on the live scene in addition to the green and purple lane-lines markers.

2. A text prompt describing the fused-perception output, including each detected object's class (e.g., stop sign, pedestrian, barrel, tire, pothole), its distance to the vehicle in meters, and its lateral position (left lane, current lane, right lane). The prompt also includes the driving task (e.g., "stay in lane unless an obstacle blocks it").

The VLM returns only the index of the selected trajectory. Restricting the output to a discrete choice (rather than free-form pixel or metric coordinates) keeps the VLM's task within its strengths, high-level visual reasoning, and yields a planning latency of approximately 0.25 s per query.

The reason VLM is adopted is that the classical object or lane-lines detectors are not always reliable. From prior testing, YOLOP's detection confidence degrades noticeably under harsh outdoor lighting; direct sun, glare, and strong shadows can suppress true detections and occasionally produce false positives. The VLM, with its much broader visual training and ability to reason about scene context (sign content, sign placement plausibility, lighting conditions, and the spatial relationship between objects and lane markings) serves as a second-stage check that catches these failure modes.

Since the SelfDrive course is mainly based on local road perception, the current driving logic does not rely on GPS waypoint navigation. Instead, the vehicle uses camera-based lane detection for path generation and LiDAR-camera fusion for obstacle detection.

Table 3: Driving Logic Requirements and Current Implementation

Requirement	**Target Value**	**Current Implementation**
Lane following	Maintain centered motion in a 10 ft lane	Camera-based lane detection generates a reference centerline.
Lateral tracking	Lateral deviation less than 2 ft over 30–50 ft	Pure Pursuit tracks the generated reference path in the vehicle frame.
Obstacle detection	Detect obstacles in the vehicle path	LiDAR-camera fusion publishes an obstacle Boolean signal.
Obstacle avoidance	Avoid obstacle and return to lane following	Lane node switches to a lane-change state when an obstacle is detected.
Steering control	Smooth and stable steering response	Pure Pursuit generates steering commands and sends them to the Arduino actuator.
System integration	Real-time sensing-to-actuation pipeline	ROS2 connects the fusion node, lane node, and Arduino control system.

7 Key Performance Indicators

The key performance indicators (KPIs) were selected to evaluate whether the vehicle can reliably complete the Self Drive Challenge. These KPIs focus on lane-following performance, obstacle response behavior, and runtime capability during autonomous operation, as summarized in Table 4.

Table 4: Key performance indicators for the Self Drive Challenge.

KPI	Target Value	Measurement Method	Current Result
Lane-keeping performance	No lane boundary crossing; lateral deviation < 2 ft	Observe lane-following runs and estimate lateral deviation	Vehicle maintained lane following during testing with occasional correction after lane-change maneuvers
Obstacle response behavior	Detect and safely avoid or stop for obstacles	Place obstacles in vehicle path during testing	Obstacle detection and lane-change behavior demonstrated during testing
Runtime capability	Runtime ≥ 2 hours	Estimate runtime from power draw and battery capacity	Estimated runtime approximately 2.5 hours

The KPI results indicate that the vehicle is capable of basic autonomous functions. The estimated runtime exceeds the team's target for competition operation. Although lane following was successfully demonstrated, physical testing also showed that additional steering correction was sometimes needed after lane-change maneuvers, indicating that further controller tuning and perception refinement may improve stability and robustness.

8 Analysis of Complete Vehicle

Several lessons were learned during both vehicle construction and system integration. During construction, the team learned how to work around physical constraints, while also meeting mechanical and electrical requirements. During system integration, the team learned that the mechanical, electrical, and software subsystems had to be tested together in order to observe how each subsystem affected the others. For example, testing showed how wheel-to-floor friction influenced vehicle steering performance and how camera mounting location affected the vehicle's field of view and perception performance.

8.1 Component Failures and Prevention of Repeat Failures

One major mechanical failure was with the steering system. The steering system required careful mechanical alignment because the modified steering linkage increased the available steering range, and the high-torque steering servo could overpower the plastic frame if the motion was not properly limited. During early testing, the steering assembly experienced mechanical overtravel near its limits, which damaged part of the plastic frame. A steering motor mounting plate was added to repair the damage and improve motor alignment and securement. To prevent repeat failures, the steering assembly was corrected and left and right limit switches were added. The Arduino firmware continuously monitors these switches and prevents additional steering motion in the direction of an active limit switch, while still allowing the servo to remain powered for steering commands in the opposite direction. After this change, steering testing was more controlled, and the same type of steering overtravel failure did not reoccur.

One major electrical failure was related to the rear motor wiring. The rear motors came with small connector housings, so early wiring attempts used thinner wires, which overheated and failed during testing. To prevent repeat failures, the wiring approach was changed so that larger current-rated wires could be used at the motor connections. An approach was then tested to carefully secure the thicker wires in place. After this change, the same rear motor wire failure did not reoccur.

8.2 Software Testing, Bug Tracking, and Version Control

Software testing was performed incrementally. First, vehicle control was tested through manual operation to ensure that the control commands worked as expected. For autonomous operation, individual functions such as lane detection, steering command generation, and Arduino communication were tested separately before being integrated into autonomous functions.

When it comes to bug tracking, issues were typically found through observation, then suspected causes were considered, changes were made to the code or approach, and finally retested. For example, early testing showed issues with serial command handling and lane-detection stability, so these issues were corrected and retested to improve performance and reliability.

Software versions were managed by preserving stable working versions before major modifications, saving shared backups in Google Drive, and testing changes incrementally to avoid introducing multiple unknowns at the same time.

8.3 Simulation-Based and Physical Testing

Simulation-based testing was used primarily for steering-control development. A MATLAB/Simulink model was developed to evaluate the Pure Pursuit controller before full vehicle testing. The simulation used a reference path generated from waypoints and compared the simulated vehicle trajectory against the desired path. The reference path was meant to reflect a lane-change scenario.

Through the simulation, the team observed how the Pure Pursuit controller performed with different lookahead distances. For example, if the lookahead distance was larger, the vehicle response became smoother, but the overshoot could become too large for ideal lane keeping. If the lookahead distance was smaller, the controller became more responsive, but the steering response could become less smooth.

The simulation did not fully capture real-world performance effects, such as steering linkage imperfections, wheel-to-floor friction, and the front-facing camera field of view. In physical testing, after completing a lane change, the vehicle sometimes required more aggressive steering correction than the Pure Pursuit controller initially provided in order to realign with the lane. Without this correction, the vehicle could lose the reference path from the camera's field of view and go off course.

8.4 Evaluation of VLM-Based Planning Approaches

A vision-language model (VLM) [7] was integrated as a 4 Hz decision supervisor alongside the (20-30) Hz classical perception nodes. The team evaluated three formulations before adopting the production design described in Section 6. The VLM was first asked to output reference trajectory points directly in pixel coordinates; although this is close to the bounding-box grounding task VLMs

are trained on, smaller models showed limited pixel-level localization accuracy, and predicted points frequently fell outside the drivable area identified by the lane detector. The VLM was next asked to output trajectory waypoints in the vehicle frame in meters, shifting all metric reasoning onto the language model; since VLMs are not trained to produce calibrated metric outputs from monocular images, the generated trajectories were often physically unrealistic, passing through barrels, crossing lane boundaries, or specifying infeasible lateral offsets. The adopted approach instead maintains a fixed library of five maneuvers (straight, soft left, soft right, sharp left, sharp right), each with a predefined vehicle-frame trajectory; at each query the five trajectories are projected into the image plane and labeled (1)-(5), and the VLM returns only the index of the best candidate given the fused-perception text prompt and the annotated image, as shown in Fig. 13.

8.5 Speed Control and Emergency Stop Evaluation

Physical testing on the assembled vehicle is used to verify subsystem requirements under real operating conditions. The longitudinal speed is regulated by a PID controller with a target velocity of 1.75 mph, which satisfies the IGVC speed requirement of 1-5 mph. Fig. 14 shows a representative velocity profile from a test run: the vehicle accelerates to the target speed and maintains it with minor fluctuations during operation. At approximately $t = 11.3$ s, the wireless emergency stop was triggered, and the vehicle decelerates sharply to a complete stop, confirming that the safety requirement of a quick and complete stop is met in practice.

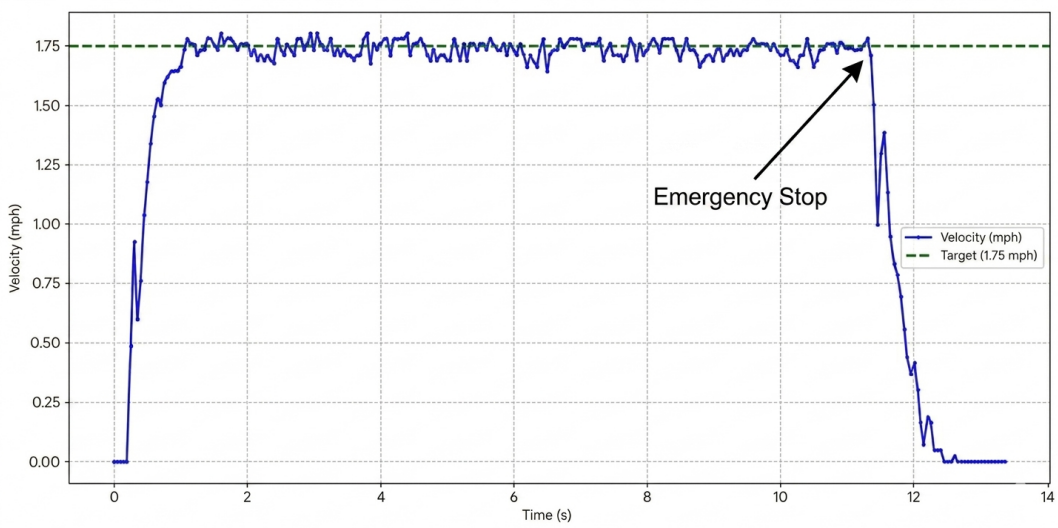


Figure 14: Vehicle velocity during a test run, showing acceleration to the 1.75 mph target, steady-state tracking, and the emergency-stop response near $t = 11.3$ s.

9 Cyber Security Analysis

The platform contains many of the same cyber security concerns found in modern robotic and human driven vehicles.

The vehicle contains several ports and connections that, while very convenient and useful during development, could also become cyber attack vectors if an unauthorized person gains physical access

to the vehicle. One way to increase security is to physically protect the ports with locked covers so that someone cannot casually access them. Additionally, even if someone does plug into a port, they should still be required to authenticate before being able to access the system or make any changes.

The vehicle also uses a Bluetooth module to communicate with a mobile application for manual control and for switching the vehicle into autonomous mode. Bluetooth is another cyber attack vector because an unauthorized user could potentially gain access to the vehicle and send control commands that could create unsafe vehicle behavior. This risk can be reduced by limiting Bluetooth access to trusted paired devices and using a secure authentication process before accepting commands.

A final vulnerability to consider is software supply-chain risk from third-party libraries, drivers, and software packages used on the vehicle. The vehicle relies on several software components from different sources, if one of these become outdated or vulnerable, it could introduce a cyber security risk where an attacker could exploit known software weaknesses. To reduce the risk, the software should come from trusted sources and be regularly updated with only approved and verified versions that do not have any known vulnerabilities.

References

[1] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2636–2645.

[2] D. Wu, M.-W. Liao, W.-T. Zhang, X.-G. Wang, X. Bai, W.-Q. Cheng, and W.-Y. Liu, "Yolop: You only look once for panoptic driving perception," *Machine Intelligence Research*, pp. 1–13, 2022.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[4] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[6] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[7] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge *et al.*, "Qwen3-vl technical report," *arXiv preprint arXiv:2511.21631*, 2025.