Concept Design Report

The 12th Unmanned
Texas A&M University

INTRODUCTION

The SAE International and General Motors AutoDrive Challenge II is a four-year collegiate design competition with the goal of developing an SAE (J3016) Level 4 autonomous vehicle [1] over the course of four years. In year 2, we designed our system aimed at tackling the Vehicle Integration Challenge, Intersection Challenge, Highway Challenge, and Perception Challenge. In year 3, we build up on the progress of the team to tackle the dynamic challenges, which include the V2X Challenge, Vehicle Integration Challenge, Intersection Challenge, and Construction Challenge [2].

In the V2X challenge, the team is tasked with receiving MAP and SPaT (Signal Phase and Timing) data transmitted by a Roadside Unit through an Onboard Unit mounted on the car. Furthermore, using the received data, the team must identify the intersection of interest based on the vehicle's location and the signal group of interest based on the lane that the vehicle would utilize to pass through the intersection. The extracted information must then be transmitted on the CAN bus. In the Vehicle Integration challenge, the team must demonstrate the ability to safely control the vehicle by traversing a provided trajectory while maintaining desired acceleration and speed limits. Further, the team must demonstrate the ability to perceive traffic lights and objects that are in a field of view of $\pm 60^{\circ}$ and a range of 40 m, demonstrate the working of the blue light system, and toggling the blinkers of the vehicle through the CAN.

The vehicle integration challenge particularly aids the team in tackling the more involved intersection and construction challenge. In the intersection challenge, the team must be able to autonomously navigate their vehicle through intersections based on the situations and traffic controls. Furthermore, the vehicle must be able to avoid static and dynamic obstacles. Finally, in the construction challenge, the team must be able to navigate their vehicle through a series of waypoints autonomously, dynamically reroute the vehicle in case of blocked lanes or roads, and park their vehicle in a vacant parking spot.

The 12th Unmanned, Texas A&M's AutoDrive Challenge II team, has over 50 graduate and undergraduate students from a variety of majors and is supported by a large faculty and industry advisory group from a diverse range of engineering disciplines to tackle the above-stated dynamic challenges and various static challenges. The team is broadly divided into seven subgroups for the year three event: Hardware, Perception, Planning, CAN and controls, Simulation, Mobility Innovation, and Project Leadership. The team is further divided into many individual project teams all coordinating together to develop our system for the year three challenge.

In this report, we will detail the designs of our system for year three of AutoDrive Challenge II and the main changes made from Year 2. We will explore the current Sensor Suite, Vehicle Dynamics, Static and Dynamic Object Detection, and Positioning & Route Planning, and highlight the changes made from Year 2 for these sections. Further, we will explore building a minimum viable product from the perspective of a budding autonomous vehicle manufacturer and provide the design based on the areas of Sensor Suite, Vehicle Dynamics, Static and Dynamic Object Detection, and Positioning & Route Planning. For each of these sections, we will include a detailed analysis for picking the proposed design, which will be supported by data collected at our testing facility.

YEAR TO YEAR COMPARISON

SENSOR SUITE Our sensor suite consists of a GPS, a Lidar, and four cameras. We have chosen to use multiple cameras for our system as opposed to multiple Lidars primarily due to cost constraints imposed by the high price of Lidar units relative to cameras. In this regard, the same camera and LiDAR setup utilized last year is also being utilized this year. However, one major change compared to last year is the utilization of the camera setup, wherein we stitch the images from the different cameras to generate a larger image.

Figure 1: Current sensor suite

The specifications of the four cameras used are summarized in Table 1. Our system's Lidar is a Cepton Vista-X90 which features a 90° horizontal field of view, a 25° vertical field of view, 200 meter range, and a 16Hz frame rate. The depiction of a top view of the camera setup and the LiDAR, wherein the LiDAR is mounted on top of the center camera, is shown in Figure 2. It should be noted here that the horizontal field of view of the center USB camera is the same as the LiDAR, and hence, the LiDAR field of view (FOV) is not shown explicitly in this figure. The depiction of the field of view of the two center cameras is shown in Figure 3.

Current Placement	Resolution	Max Frame Rate (fps)	Field of View (FOV)
Primary Center	4000x3000	10	121°
Secondary Center	5472 x 3648	18	90.8°
Side Cameras	1920 x 1200	41	83.2°

Table 1: Overview of current camera specifications

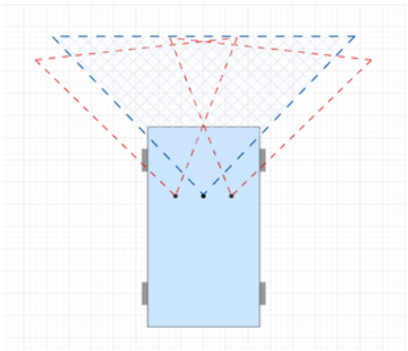

Figure 2: Current Vehicle Camera FOV

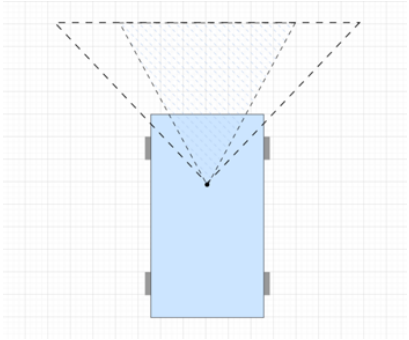

Figure 3: Center Camera Model

Same as last year, we use a NovAtel PwrPak7D GNSS module for GPS localization. However, compared to last year, we have procured a TerraStar C subscription from NovAtel to obtain a highly accurate location estimate from the unit. While our GPS accuracy was closer to 0.5m last year, we are consistently within 0.05 m with the procured subscription. Furthermore, by configuring the GPS driver, we are able to completely utilize the SPAN technology from NovAtel to obtain reliable heading angle data. This year, a change in the hardware we made was the installation of an OCP card on the Intel server, which provides us with an additional 4 ethernet port on the server.

VEHICLE DYNAMICS In the previous year, we utilized a PID controller for the longitudinal control to control the vehicle's speed and a pure pursuit model for lateral control for the vehicle [3]. The pure pursuit algorithm is a simple algorithm that utilizes a bicycle model, a kinematic model, for modeling the vehicle. In this algorithm, a lookahead point is selected on the desired trajectory to be tracked by the vehicle, and the corresponding steering angle is computed. However, one issue we encountered with this controller last year was higher-than-expected lateral errors. In many scenarios, a lateral error close to 0.7 m was obtained. Since we desired lateral errors to be close to 0.3 m for accurate tracking, we implemented the controller discussed in [4], which utilizes a feedforward component for tracking turns and a feedback term for correcting errors. Furthermore, a PI controller was used for longitudinal control since the acceleration data obtained from the IMU was observed to be noisy.

The need for the alternate lateral controller aims not only to meet the basic requirements of staying below peak longitudinal acceleration and deceleration, lateral acceleration, and maximum vehicle speed but also to surpass them by implementing additional error constraints such as heading, lateral, and heading rate errors. These enhancements allow for a more refined analysis and fine-tuning of the vehicle's control systems, ensuring superior performance and robustness.

The lateral controller system design was centered around the above-described three error metrics to compute the desired steering angle. The longitudinal controller was

centered around the speed error and the difference between the expected and desired position error to compute the desired motor torque or brake torque. The details for the same are discussed in the vehicle dynamics section in the minimum viable product, along with a comparison with another controller.

STATIC AND DYNAMIC OBJECT DETECTION We broadly classify static and dynamic object detection and tracking into traffic sign and object detection, object tracking, traffic light detection, and lane and limit line detection.

Traffic Sign and Object classification The Traffic Sign Detection and Classification module must be able to detect all traffic signs within 40 meters and $\pm 60^{\circ}$ of the front of the vehicle. Similar to last year, our method of classification of traffic signs utilizes YOLO [5], a robust machine learning algorithm that utilizes a single frame to classify data. We moved to the latest version of YOLO, which is YOLOv8, to obtain higher accuracy and faster computation time. In a similar manner, we utilize YOLOv8 for the detection of objects within 40 meters and $\pm 60^{\circ}$ of the front of the vehicle. Furthermore, one major change is the inclusion of diverse datasets for many signs and objects that the model was not trained for or failed to detect last year. The additional signs and objects considered, which are part of the competition requirements, include Left turn only, Right turn only, Railroad crossing, Yield signs (for a far distance), Barrel, and Type 3 barricade. One other major change is utilizing LiDAR data to obtain the relative location of the signs and objects as opposed to stereo vision.

The LiDAR point cloud data is used in conjunction with YOLOv8 for distance calculations after an object or sign has been identified. One of the critical tasks in this regard is to remove LiDAR points corresponding to the ground. This year, we use an alternative approach to remove ground points, which involves a two-stage approach: (i) Utilize patchwork++ [6], a fast algorithm based on adaptive ground likelihood estimation, to extract ground points faster than RANSAC, a previous algorithm we used, and (ii) Remove points that are below a certain height from the LiDAR (which was 1.9 m) to remove LiDAR points that were not removed by patchwork++.

Once the ground points are removed, we take the relative homogeneous transformation matrix that contains rotation and translation values between the LiDAR and camera and place the LiDAR points within the camera image. The 3D LiDAR points are then correlated with pixels within the camera frame through intrinsic calibration. This allows each LiDAR point to be represented within the image and an object's relative position to be determined [6]. The depth perception point selection for the 12th Unmanned vehicle uses the YOLOv8's bounding box as an initial boundary for point selection. Then, it filters out data points corresponding to the object's background and adjusts for outliers. An example of this can be seen in Figure 4. In this figure, Dz and Dx denote the relative longitudinal and lateral distance, respectively, from the LiDAR.

Figure 4: Lidar points for barrel classification

Object tracking Object tracking is performed primarily based on the lidar module, which utilizes a Dynamic Hungarian assignment method to obtain object identifications. The method comprises mainly of four steps: preprocessing, clustering, tracking, and classification. These steps are visualized in Figure 5. The output from clustering for tracking a pedestrian is shown in Figure 6. However, one change compared to last year is the classification step, wherein the classification is performed based on the image data instead of lidar data. The classification is performed by projecting the lidar bounding box on the camera image, and utilizing YOLOv8 to identify the object type. An example of this process is demonstrated in Figure 7, wherein the box shown in red is obtained from lidar clustering.

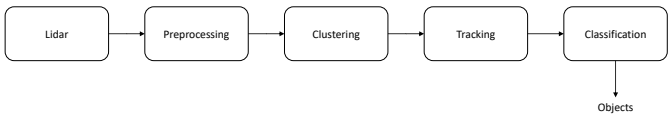

Figure 5: The object detection, tracking, and classification flow

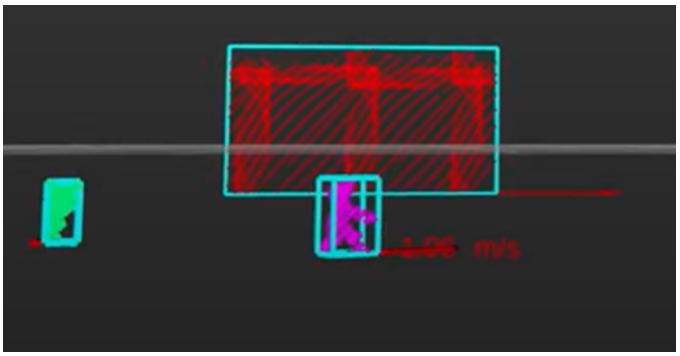

Figure 6: Clustering and object tracking

Traffic light Similar to last year, the traffic light detection is performed in two main steps as shown in Figure 8: identifying the traffic head using YOLO, and then identifying the type of traffic light on the cropped image of the traffic signal head, which includes obtaining the color and shape. A couple of changes were made, however, in the implementation: YOLOv8, the latest YOLO version, was used instead of YOLOv3 to improve performance and

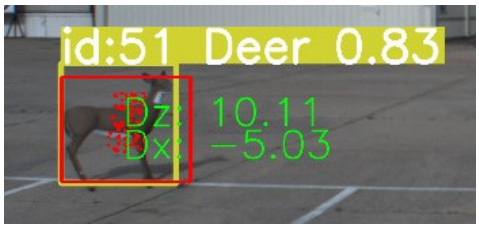

Figure 7: Camera and lidar bounding box

speed, and a custom smaller YOLO model was trained instead of a color-based thresholding method to obtain the color and shape. Furthermore, instead of utilizing a sequence analysis to track the state of the traffic light, we utilize the location of the traffic light head, previous camera frames, and confidence to identify the state of the light. Similar to traffic signs and object detection, lidar data is now used to obtain the relative location of the traffic light.

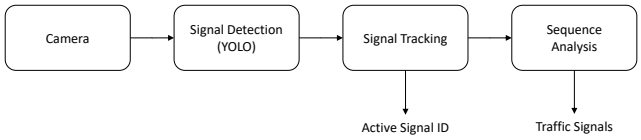

Figure 8: The traffic light detection and classification flow

Lane and limit line Lane detection was previously performed using image processing utilizing OpenCV [7] to identify the lane in which the vehicle is present. However, this year, we have transitioned to a Sequential convolutional neural network (SCNN) based method, which utilizes spatial data to identify the lanes. This transition is done to account for changes in the intensity of the surroundings, which we observed to impact the performance of the algorithm. Furthermore, for lane detection, we also utilize the lane information obtained from the high-definition map provided by Dynamic Map Platform (DMP). In this method, utilizing GPS information, we can obtain the lane information in the segment in which we are present. We use this additional stream of data to obtain lane information for objects in the surroundings, whereas SCNN is used to ensure that the vehicle maintains the desired lane.

For the limit line detection, we utilize image processing, similar to last year, due to the good performance of image-based methods for identifying horizontal white lines. We detect the limit lines in the white binary image simply by identifying large horizontally aligned groups of pixels and fitting a nearly horizontal line through the pixels. We demonstrate some of the results for lane and limit line detection as a part of the same section under the Minimum Viable Product.

POSITIONING & ROUTE PLANNING Similar to the previous year, the positioning of the vehicle is based on the GPS data obtained from the NovAtel unit on the car, followed by utilizing the HD map provided by DMP to localize the car in the environment. The route planning module comprises the global planner and local planner, both of which utilize the HD map information. The global planner, similar to last year, takes the initial and final location of the vehicle as input and plans the path in the graph obtained from the HD map using Dijkstra's algorithm [8].

The local planner, which is a complex system, is responsible for generating a safe and feasible trajectory in real time. It works in conjunction with the Global Planner, which generates a high-level path to the destination. It uses data from the lidar and cameras to perceive the environment and plan a safe path Using the perceived objects, the different areas of the lanes that are blocked off through obstacles are identified, and a safe maneuver to go past the obstacles or stop in case of a dynamic obstacle is computed. We provide relevant data collected for the local planner in the minimum viable product section of this report.

MINIMUM VIABLE PRODUCT

SENSOR SUITE The 12th Unmanned team's current sensor configuration is shown in Figure 1. The vehicle's current sensor suite consists of a GNSS module with two receivers placed diagonally on the sensor mount, one LiDAR sensor, two center cameras, and two side cameras. The vehicle has also been tested using all of the sensors listed above. The following section will cover how the team plans to simplify the sensor suite.

Camera It is important to understand the requirement specification needed for the cameras to function properly for the competition. For the cameras to properly function with the CAN system, a minimum refresh rate of 10 Hz is required. In order for the cameras to coincidentally function with the LiDAR and the rest of the perception system, the vehicle must have a minimum field of view of 120° and a minimum view distance of 40m. In order to accurately capture, detect, and classify objects, the camera must be able to capture higher-definition images; for this reason, the team has decided to get at least HD cameras with 1080p resolution. The current cameras on the vehicle all meet the requirements entailed above.

FLIR offers a variety of camera products for different industries and applications. FLIR specializes in thermal cameras but has also developed infrared and machine vision cameras. The machine vision cameras offered by FLIR were the most applicable for the AutoDrive Challenge. Furthermore, lenses are separately equipped with different lenses from FujiFilm. These lenses provide the camera with different fields of view, magnification, and focal lengths. Table 1 shows the specifications of each camera and lens that is currently mounted on the vehicle. Tables 2 and 3 show the different camera and lens models that are currently on the vehicle and their retailed prices. For the camera setup, these two modules are what are

focused on for the challenge.

Current Placement	Model ID	Cost
Primary Center Camera	FLIR Blackfly GigE (BFS-PGE-04S2C-CS)	$491.00
Secondary Center Camera	FLIR Blackfly S USB3 (BFS-U3-200S6C-C)	$699.00
Left and Right Side Cameras	Blackfly GigE (BFLY-PGE-23S6C-C)	$595.00

Table 2: Current Vehicle Camera Cost

Current Placement	Model ID	Cost
Primary Center Camera	FUJINON (XF8mmF3.5 WR)	$799.00
Secondary Center Camera	FUJINON (XF14mmF2.8 R)	$899.00
Left and Right Side Cameras	FUJINON (XF16mmF1.4 R WR)	$999.00

Table 3: Current Vehicle Camera Lens Cost

Currently, the main front camera is the FLIR GigE Blackfly S make models equipped with a fish-eye lens from FujiFilm which provides the camera with a 122° field of view. The left and right cameras are the FLIR GigE Blackfly models with a 1900x1200 pixel resolution using the standard lens providing an 83.2° field of view. Currently, the side cameras are mounted 20° outwards, 70 centimeters to the side of the two main center cameras. Figure 2 shows a model of the three different cameras and their field of view from the mount.

Figures 9, 10, and 11 show the images taken from a test run of all three camera perspectives at an intersection. Here, the secondary center camera was considered since the primary center camera had a high distortion due to the fisheye lens. As expected, the main camera captures a wide field of view, but the side cameras provide a little bit of a wider perspective than the center camera. It is evident from both the testing footage and the vehicle model that there is a lot of overlap between the side cameras and the main center camera.

Figure 9: Left Camera View

As mentioned before, the primary center camera is equipped with a fisheye lens that allows for 121° FOV.

Figure 10: Secondary Center Camera View

Figure 11: Right Camera View

Considering the field of view, angle offset, and placement of the side cameras, the calculated field of view angle that the side cameras captured was 124°. Therefore, the vehicle is able to cover the necessary field of view with just the center camera or solely the left and right cameras.

The team considered two alternate designs for the minimal viable product design. The vehicle can still function with either the two center cameras or the two side cameras, as both designs meet the specifications needed for the vehicle to function.

Utilizing the two center cameras would cover the full 120° FOV requirement and would make use of an extra backup camera in the case that the main camera would malfunction. This design also minimizes the amount of data that would need to be processed through the perception system. A major drawback of using the center camera is that the fisheye lens distorts the images, especially on the ends of the image captured. This could potentially result in detection and classification malfunctions. This could possibly be corrected by transforming the image into a rectilinear image, but this process would be costly and possibly ineffective. The total cost of having these two center cameras would be $1,190. The model of this design can be seen below in Figure 3.

Using the two side cameras is another viable option. The two side cameras offer a wider field of view. If the two side cameras are used, it is important to take into account the blind spot point at which the two camera views intersect. The distances and angle offset of the cameras must be carefully calibrated so that the blindspot hits the hood of the vehicle. The camera placements mentioned before account for the blind spot, and using the current mount,

the blind spot will not be an issue. When it comes to cost, the total cost for the two side cameras also comes out to $1,190. The model of this design can be seen in Figure 12.

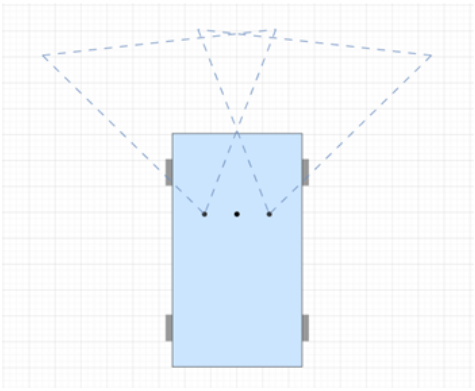

Figure 12: Side Camera Model

Ultimately, both designs were analyzed by comparing the cost, efficiency, and effectiveness of both configurations. After consideration, utilizing the two side cameras was determined to be the most effective and simplistic design.

With the understanding that two cameras would be necessary, other alternative cameras from FLIR were researched to ensure that the most cost-effective cameras from FLIR's machine vision line were being used. As mentioned previously, the current cameras being used are the Blackfly GigE models that retail for $595. Table 4 shows two alternative machine vision cameras offered by FLIR that meet the specification requirements.

Model	Resolution	Max FPS	Cost ($)
Chameleon3 USB3 (CM3-U3-13Y3C-CS)	1280x1024	149	415.00
FLIR Firefly DL (FFY-U3-16S2C-C-DL)	1440x1080	60	326.00

Table 4: Alternative Camera List

The Chameleon3 is a smaller-sized charged coupled device camera that is used as a simpler and cost-effective alternative to the Blackfly S model. Using the Chameleon3 would be a cheaper alternative to the current Blackfly S and would still meet all the camera specifications.

The FLIR Firefly DL allows the user to deploy a trained neural network to the camera itself with neuro-technology to reduce system cost and complexity by making on-camera decisions. Although the Blackfly S model has better camera specifications, the Firefly DL is more cost-effective and can offer an advantage in data processing.

With these things considered, it would be safe to assume that for the minimal viable product design, using the FLIR Firefly DL cameras for the side cameras with the FujiFilm XF16mmF1.4 R WR lens would be the minimal viable product design for the camera setup. The total cost would come out to $2,650.

Lidar The perception system functions with a Cepton Vista X90 LiDAR that has a 90° field of view. The Vista X90 also has an angular resolution of 0.09° x 0.09°, along with a range of 200m at 10%. Additionally, the sensor suite also contains a Cepton Vista X120 LiDAR with a 120° horizontal field of view. The Vista X120 has an angular resolution of 0.19° x 0.19° and a range of 200m at 25%.

A sensor suite with two LiDAR sensors results in more accurate data through redundancy and noise reduction. However, the competition requirement of a 120° horizontal field of view LiDAR can be met entirely by several different single sensors. Therefore, the Vista X90 LiDAR could be potentially removed from the sensor suite and the vehicle can operate solely on a 120° FOV sensor. There are many different combinations or options of LiDAR sensors that can be utilized on the vehicle, several of which can be seen in Table 5.

Model (Points/ Second (Million))	FOV (°H)	Res (°)	Range	Frame Rate (Hz)
Cepton Vista P-60 (.3)	60	.25 x .25	200m at 30%	10
Cepton Vista P-90 (.3)	90	.38 x .38	200m at 80%	10
Cepton Vista X-90 (1)	90	.20 x .20	200m at 10%	16
Cepton Vista X-120 (1)	120	.19 x .19	225m at 10%	16
Cepton Nova (.3)	120	.70 x .70	30m at 10%	20
Cepton Sora X-90 (NA)	90	.09 x .09	200m at 10%	246
Cepton Sora X-120 (NA)	120	.13 x .13	200m at 25%	250

Table 5: Lidar comparison

Table 5 details several potential LiDAR sensors available from Cepton, each with its unique advantages and qualities. Assuming that all sensors are potential possibilities for the sensor suite, several options can be eliminated based on the factors of cost, time, complexity, and the specifications of the sensor. Additionally, the exact prices of the sensors were unable to be determined via Cepton, so the sensors labeled as "cost-effective" are assumed to

be cheaper than those that are not.

Firstly, the Cepton Sora X-90 and X-120 can both be eliminated as a possibility for the sensor suite, as they are both assumed to be more expensive than the other sensors. This assumption is made because the sensors have an extremely high frame rate, so they are likely more expensive than the sensors with a more standard frame rate. Therefore, while the high frame rate would improve the capabilities of the sensor suite, a frame rate that high is unnecessary and is not worth the cost investment when compared to other sensors that are adequate and not as expensive.

The next two sensors that can be ruled out are the Cepton Vista P-60 and the Vista P-90. Both of these sensors have suitable specifications for the sensor suite, but they both have a horizontal FOV lower than the required 120° for the competition. This means that in order for these sensors to be utilized in the sensor suite, there would need to be more than one of these sensors. This is not practical as including another sensor would increase the cost and complexity of the sensor suite, and if multiple sensors need to be utilized anyway, there are other sensors that have better resolutions and frame rates.

Similarly, the Vista X-90 can be eliminated from the sensor suite potential choices. This sensor has better specifications than the P-60 and the P-90, but still encounters the issue of requiring two sensors to meet the 120° horizontal FOV requirement. As a result, this sensor option is not a viable option due to the required cost of purchasing two separate sensors.

The last two sensor options include the Cepton Vista X-120 and the Cepton Nova. The Vista X-120 meets the required horizontal FOV of 120° and also has a fairly good resolution, frame rate, and number of points/second. This makes it a viable choice for the sensor suite as it has good qualities all around and meets the requirements by itself, so only one sensor is needed, reducing the overall cost.

The other option is the Cepton Nova, which also meets the 120° horizontal FOV requirement, but it does have significantly worse resolution and range than the Vista X-120. However, the Nova is marketed as a ”mass-scale manufacturable” sensor that is cost-effective, suggesting that it is much cheaper than most of the other sensors created by Cepton. Additionally, the Nova has a higher frame rate than the Vista X-120 as well as a much greater vertical FOV of 90° compared to the 32° of the Vista X-120. This means that the Nova would have a much smaller blind spot vertically near the car, and if that is a parameter that needs to be prioritized, it could be a viable option. The Nova could also potentially cost less than half of the Vista X-120, meaning that depending on the exact cost from Cepton, two Nova sensors might be cheaper than a single Vista X-120. The depiction of the field of view is shown in Figure 13.

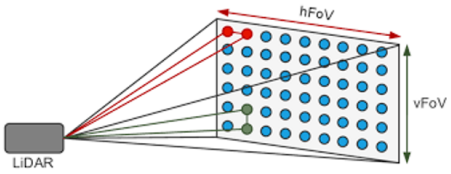

Figure 13: Field of view diagram

The final decision for the LiDAR selection for the sensor suite depends on the Cepton Vista X-120 cost and the Cepton Nova. If the Nova is less than half of the cost of the Vista X-120, then the Nova should be selected, and two Nova sensors could be utilized in the sensor suite, offering a great horizontal and vertical FOV, a higher frame rate than the X-120, and a reduced blind spot, all at a potentially cheap cost. Additionally, if the Cepton Nova is selected, the sensor could be mounted on top of the side cameras for ease of setup and design.

If the Vista X-120 is cheaper than purchasing two Nova sensors, the Vista X-120 should be selected for the sensor suite. This sensor offers great resolution, range, and number of points per second. In addition, the use of one LiDAR sensor decreases the overall complexity of the sensor suite. A reduction in the number of sensors in the sensor suite would decrease the time spent calibrating the sensors in that each sensor must be properly calibrated before use, and a reduction in this time spent means that the time could be spent elsewhere.

Network Switch and Network Card The current network switch is a GS716Tv3. The potential benefits a new system provides are a higher switching capacity and that this is a managed network switch.

By having a higher switching capacity, the network switch is able to handle a higher level of data traffic that the GS716Tv3 couldn’t. This makes the network switch more reliable in the case of high network traffic, and it will perform better overall. Another benefit is that the individual speed per port is increased to 10 Gbps and that multiple port configurations are accessible, as opposed to the GS716Tv3, which only has access to a single default configuration.

An unmanaged network switch is typically for simple systems that do not require much configuration and are generally seen as ‘plug and play’ type switches. Managed switches, on the other hand, give developers the ability to customize different aspects of the network, such as how data is transmitted, individually configure the switch’s ports, choose which traffic to prioritize, and manage the network remotely. These new features and flexibility that the managed network switch offers make it the ideal choice for a network switch over an unmanaged network switch.

	Brand	Model	Switching Capacity	Ports	SFP Ports	Min Speed/Port	Max Speed/Port	Type	Cost
Current	Netgear	GS716Tv3	32 Gbps	16	2	-	1 Gbps	Unmanaged	**$230**
Potential	Netgear	M4300-8X8F	480 Gbps	0	8	100 Mbps	10 Gbps	Managed	**$2,160**
Potential	Netgear	MS105	25 Gbps	5	-	100 Mbps	2.5 Gbps	Unmanaged	$150
Potential	Netgear	GS308V3	10 Gbps	8	-	10 Mbps	1 Gbps	Unmanaged	$28
Potential	Netgear	XS505M	100 Gbps	4	1	100 Mbps	10 Gbps	Unmanaged	$430
Potential	Netgear	GS305Pv3	10 Gbps	4	1	100 Mbps	1 Gbps	Unmanaged	$70
Potential	FS	S3900-24T4S	128 Gbps	24	4	1 Gbps	10 Gbps	Managed	$329
Potential	FS	S3150-8T2FP	20 Gbps	8	2	-	1 Gbps	Managed	$229
Potential	FS	S3900-48T6S-R	216 Gbps	48	4	1 Gbps	10 Gbps	Managed	$459
Potential	FS	S3400-24T4FP	56 Gbps	24	4	-	1 Gbps	Managed	$439
Potential	FS	S2805S-8TF	20 Gbps	8	2	-	1 Gbps	Managed	$69
Potential	FS	S3150-8T2F	20 Gbps	8	2	-	1 Gbps	Managed	$99
Potential	FS	S2805S-8TF-P	20 Gbps	8	-	-	1 Gbps	Managed	$119
Potential	FS	S1900-8TP	15 Gbps	8	-	-	1 Gbps	Unmanaged	$89
Potential	FS	IES2100-5FE	1 Gbps	5	-	-	100 Mbps	Unmanaged	$99
Potential	FS	S1900-8T	16 Gbps	8	-	1 Gbps	2 Gbps	Unmanaged	$32
Potential	Linksys	LGS310C	20 Gbps	8	2	1 Gbps	2 Gbps	Managed	$100
Potential	Linksys	LGS108	20 Gbps	8	-	10 Mbps	1 Gbps	Managed	$50
Potential	Linksys	LGS310MPC	16 Gbps	8	2	1 Gbps	2 Gbps	Managed	$180
Potential	Linksys	LGS108P	16 Gbps	8	-	10 Mbps	1 Gbps	Unmanaged	$117

Figure 14: Cost-benefit analysis of network switch

The minimum requirements for the network switch on the autonomous vehicle are that it has to have a speed of 10 Gbps and it has to be a managed system. Many commercial managed network switches typically are above the 10 Gbps limit, as cheaper unmanaged switches are the network switches that are usually below that limit. After considering multiple models, which are summarized in Table 14, the Netgear XS505M would be the minimum viable network switch that can be used for this system at $430. Since only a maximum of 5 ports were necessary for the system, and a minimum of 10 Gbps speed per port was needed.

GPS AVs rely on advanced positioning technologies for safe and efficient navigation. NovAtel, a leading navigation technology provider, offers a range of products. The Chevy Bolt our team uses has the NovAtel PwrPak7D-E2, a high-precision GNSS (Global Navigation Satellite System) receiver featuring an OEM7 series board within an enclosure designed for diverse applications needing accurate navigation. The PwrPak7D-E2 offers dual-antenna inputs for precise heading and moving base station functionality, which is suitable for AVs. It supports multiple satellite signals, including GPS. Key features include NovAtel CORRECT with Real-Time Kinematics (RTK) for centimeter-level accuracy and interference mitigation to counteract RF interference in dense signal environments [9]. Additionally, its integrated IMU maintains accurate positioning when GNSS signals are unavailable, which is beneficial in locations like urban canyons, areas where tall buildings reflect or block signals, or dense forests. The PwrPak7D-E2 provides robust and precise positioning for various scenarios.

NovAtel provides key positioning components for AVs, including antennas, IMUs, and correction services. Their integrated system solutions are designed for easy installation by vehicle vendors. The following comparison highlights the differences between combined systems offered by NovAtel based on antenna type, IMU, and correction services. Please note that we were unable to gather pricing details during research.

Combined System Name	Antenna Type	Main Features	Position Accuracy (RMS)	ALIGN Heading Accuracy (RMS)	Velocity Limit	Data Rate GNSS/INS	Correction Services	Additional Features
PwrPak7-E1	Single Antenna	SPAN GNSS+INS	1.2 m	N/A	600 m/s	200 Hz/20 Hz	TerraStar-X5 TerraStar-L5 TerraStar-C PRO5	IMU, Wi-Fi
PwrPak7-E2								Enhanced IMU, Wi-Fi
CPT7700								IMU, Water Immersion
PwrPak7D-E1	Dual Antenna	SPAN GNSS+INS ALIGN Heading		2 m/ 0.08° 4 m/ 0.05°			TerraStar-L5 TerraStar-C PRO5	IMU, Wi-Fi
PwrPak7D-E2								Enhanced IMU, Wi-Fi
CPT7								IMU, Water Immersion

Figure 15: Comparing NovAtel's combined systems based on antenna type [10]

Figure 15 presents a comparison of the types of antennas included in each combined system. All systems utilize the OEM 7 engine with SPAN GNSS+INS, which combines GNSS positioning with inertial navigation for comprehensive 3D positioning, velocity, and attitude solutions [10]. The choice between single and dual antennas hinges on the need for accurate heading information. Dual antenna configurations are superior for urban environments where structures can cause GNSS signal issues, which can cause potential inaccuracies with single antenna systems.

NovAtel's ALIGN® firmware further refines positioning by offering precise heading and relative position data, enhancing maneuvering and navigation capabilities. For the safety and reliability required in urban settings, dual-antenna systems like the PwrPak7D-E1, PwrPak7D-E2, and CPT7 in Figure 15 are recommended, providing resilience against the path interferences common in such environments.

Figure 16 shows the IMUs included in NovAtel's combined systems. The specifications of these IMUs align with the performance standards required for AVs. Considering the cost and assuming higher-specification IMUs would be pricier, we recommend the Epson G320N IMU as a cost-effective yet capable solution. This makes the PwrPak7D-E1 an optimal choice for a comprehensive system setup. Lastly, it is important to consider that alongside the selected antennas, AVs require precise navigation. This is achieved with correction services that enhance GNSS data accuracy. A fundamental approach to achieving this involves combining Raw GPS data with a Kalman filter, using IMU inputs for more accurate positioning. While raw GPS may only provide 5–10-meter accuracy, integrating it with a Kalman filter can refine this to about a meter [10]. IMUs, which are prone to drift, benefit from GPS corrections for improved navigation. Thus, advanced navigation systems can combine GPS, Kalman filters, and IMU data to achieve better accuracy.

Figure 17 demonstrates that applying a Kalman filter to Novatel's low-cost GPS data markedly improves accuracy across Easting, Northing, and Altitude, a closer match to the actual path. However, this GPS-Kalman filter-IMU in-

Combined System Name	IMU	Horizontal /Vertical Position Accuracy (RMS)	Horizontal/ Vertical Velocity Accuracy (RMS)	Altitude Roll &Pitch/ Heading Accuracy (RMS)	Data Rates	Accelerometer/ Gyroscope Performance
PwrPak7-E1	Epson G320N MEMS	1.00 m/ 0.60 m	0.020 m/s / 0.010 m/s	0.020°/ 0.090°	125 Hz	Accelerometer Dynamic range: 5 g Bias instability: 0.1 mg Velocity random walk: 0.05 m/s/√hr Gyroscope Dynamic range: 150 °/s Bias instability: 3.5 °/hr Angular random walk: 0.1 °/√hr
PwrPak7D-E1						
PwrPak7-E2	Epson G370N MEMS		0.015 m/s / 0.010 m/s	0.013°/ 0.070°	200 Hz	Accelerometer Dynamic range: 10 g Bias instability: 0.012 mg Velocity random walk: 0.025 m/s/√hr Gyroscope Dynamic range: 450 °/s Bias instability: 0.8 °/hr Angular random walk: 0.06 °/√hr
PwrPak7D-E2						
CPT7700	Honeywell HG4930			0.010°/ 0.030°	100 Hz	Accelerometer Dynamic range: 20 g Bias instability: 0.075 mg Velocity random walk: 0.06 m/s/√hr Gyroscope Dynamic range: 400 °/s Bias instability: 0.45 °/hr
CPT7						

Figure 16: Comparing NovAtel's combined systems based on IMU [10]

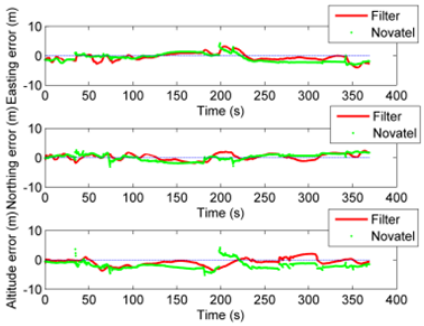

Figure 17: Error between ground-truth vs low-cost GPS and ground-truth vs filter solution [12]

tegration adds complexity and requires significant computation, potentially restricting its use in resource-limited systems.

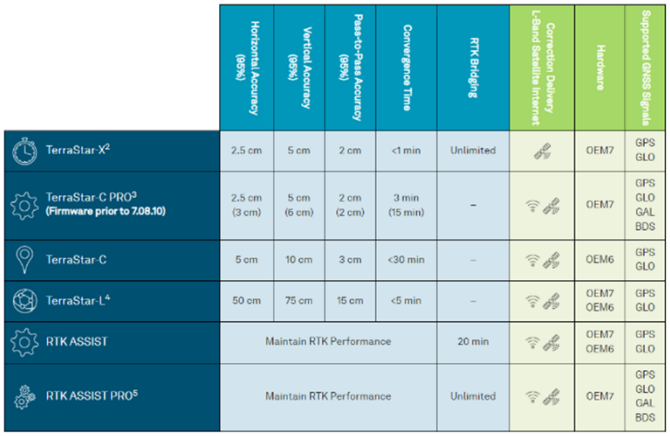

Figure 18: Correctional services used in NovAtel products

Figure 18 outlines the correction services compatible with various NovAtel products. While RTK offers exceptional accuracy, its dependency on close proximity base stations and signal obstructions can limit its use. TerraStar services provide the precision necessary for an AV without these constraints. From Figure 15, PwrPak7D-E1 supports TerraStar-L5 and TerraStar-C PRO5. The recommended accuracies for AVs are ± 0.45 m horizontally and ± 0.51 m vertically [14]. Both TerraStar options exceed these, with TerraStar-L5 chosen with the assumption of a lower cost for less precision. Thus, for a functional AV positioning system, we suggest the PwrPak7D-E1 with TerraStar-L5.

Ethernet card The ethernet switch is used to physically connect many devices on the car together when there are not enough ports on the CPU itself. The ethernet card is attached to the CPU and determines the number of ports available. Because all available ethernet cards have only 2 or 4 ports, the network switch is necessary. It is estimated that the ethernet card will need 10 gigabytes/second, and 25 gigabytes/second is considered overkill. The adapter must be compatible with "710" such as XXV710-DA2. Additionally, the card must be compatible with PCIE. The cheapest ethernet card that meets these requirements is the Intel X710-BM2 for $199.

Intel Controller (# ports)	Cost ($)	Supported speeds (GB/s)	Host interface (PCIe)
X710-BM2 (2)	199	1, or 10	3.0 x 8
82599ES (2)	219	1, or 10	2.0 x 8
X550-AT2 (2)	269	1, 2.5, 5, or 10	3.0 x 4
E810-XXVAM2 (2)	339	1, 10, or 25	4.0 x 8
XXV710-AM2 (2)	349	1, 10, or 25	3.0 x 8
XL710-BM1 (4)	439	1, 10	3.0 x 8
XL710-BM2 (2)	520	1, 10, or 40	3.0 x 8
E810-CAM2 (2)	689	100	4.0 x 16

Table 6: Ethernet cards

CONCLUSION AND RECOMMENDATIONS Through the analysis provided, it would be safe to assume that for the minimal viable product design, using the FLIR Firefly DL cameras for the side cameras equipped with the FujiFilm XF16mmF1.4 R WR lens would be the minimal viable product design for the camera setup. For the LiDAR, depending on the cost of the Vista X120 and two Nova units, the cheaper LiDAR setup can be utilized for the MVP. If the former setup is selected, the Vista X120 can be mounted on the center of the car, whereas if the latter setup is selected, it can be mounted on top of the side cameras. After considering multiple models for the network switch, the Netgear XS505M would be the minimum viable network switch that can be used for this system at $430. Since only a maximum of 5 ports were necessary for the system, a minimum of Gbps speed per port was needed. With regards to the ethernet cards consid-

ered, all cards are able to provide 10 GigaBytes per second. Because a network switch is being used, the number of ports is not very important. Assuming no compatibility issues arise, the Intel X710-BM2 is the cheapest option that satisfies requirements. Finally, through the analysis of the GPS units considered, the NovAtel PwrPak7D-E1 with TerraStar-L5 subscription can be utilized for the minimum viable product to satisfy the requirements for reliable location information.

VEHICLE DYNAMICS

To enhance lateral control, we introduced a Compound Feed controller combining a trajectory-based Feedforward and a tuned Feedback controller. This year's advancements include resolving implementation issues, modeling vehicle parameters for the Feedforward controller, and fine-tuning Feedback gains.

Simultaneously, a Stanley controller [11] was implemented to compare with the Compound Feed controller across various test scenarios. The Stanley controller, simpler and involving only one gain value, aims to enhance the efficiency of lateral control systems.

The goals for the controller that we desired to achieve are summarized in Table 7. It should be noted here that the competition requirements for the controller include only acceleration requirements. We have included the additional design requirement of lateral error bound, since we desire the vehicle to track the desired trajectory well.

Req. #	Metric	Importance	Value
1	Lateral Position Stray (m)	5	<0.3
2	Lateral acceleration (m/sˆ2)	4	<3
2	Longitudinal acceleration (m/sˆ2)	4	<3

Table 7: Design Metrics

CURRENT LONGITUDINAL CONTROLLER The longitudinal controller utilizes a PI (Proportional-Integral) approach. This method adjusts the vehicle's throttle and braking based on the errors from the desired speed. By integrating error values over time, it minimizes overshoot and stabilizes the response to speed changes, enhancing control over the vehicle's acceleration and deceleration.

In terms of gain settings within the controller, there are distinct sets for various driving scenarios: high speed, cruising speed for minimal oscillation, usual speed for normal operation, and braking gains for deceleration. The control decisions are based on calculated control outputs (u), derived from current and desired speeds, as well as positional errors, as

$$u = -K_p \Delta v - K_i \Delta \|x\|, \tag{1}$$

where K_p is the proportional gain, K_i is the integral gain, Δv is the difference between the current and desired speeds, and $\Delta\|x\|$ is the positional error over time (accumulated error). Based on whether the computed acceleration is positive or negative, a motor torque or brake torque is commanded from the vehicle.

CURRENT LATERAL CONTROLLER This controller includes two parts, combining feedforward and feedback strategies to manage the vehicle's lateral positioning. The feedforward controller computes the necessary steering input δ_{ff} based on the planned vehicle trajectory, while the feedback controller corrects steering with input δ_{fb} in real time based on three errors relative to the trajectory. The three errors for lateral control are

- Lateral error (e_{lat}): the distance between the vehicle and the currently planned path point.
- Heading error ($\Delta\psi$): the angle between the current direction the vehicle is facing and the current direction the planned path is facing.
- Heading rate error ($\Delta\dot{\psi}$): the difference between the rate of change of the heading of the vehicle and the rate of change of the heading of the planned path.

Therefore, the lateral control law to compute the desired steering angle is given by

$$\delta = \delta_{ff} + \delta_{fb}, \tag{2}$$

where δ_{ff} is given by

$$\delta_{ff} = \frac{L}{R} + K_{sg}(v, R)\frac{v^2}{R}. \tag{3}$$

Here, L is the wheelbase of the car, R is the desired turning radius, K_{sg} is the understeer gradient of the vehicle, which is a parameter to be tuned, and v is the speed of the vehicle. The understeer gradient describes how much lateral force will be generated at a given speed, turn radius, and steering angle. Using circular paths of various lateral speeds and radii, the team calculated the understeer gradient from data collected. Using these newly found steering gradient values, a basic linear interpolation system was employed to compute understeer gradient values based on the vehicle's current longitudinal speed.

The feedback controller is given by

$$\delta_{fb} = -K_e e_{lat} - k_\theta \Delta\psi - k_\omega \Delta\dot{\psi}, \tag{4}$$

where the gains are tuned based on the circular and straight-line trajectories.

SIMPLER LATERAL CONTROLLER: STANLEY While the feedforward and feedback controllers introduced previously can operate independently, both controllers have

their shortcomings. The feedforward portion has no method to compensate for errors generated by non-ideal conditions or errors within vehicle dynamic calculations and assumptions. The feedback controller has no predictive capabilities, causing large errors to take place at the beginning of curved paths and the vehicle to take an oscillatory path as it tries to converge with the planned trajectory. Due to these issues, the vehicle can be marginally jerky when entering into a turn and exiting from a turn; furthermore, straight-line tracking when the vehicle is started marginally away from the straight line can be aggressive. In this regard, we have studied and implemented an alternate simpler controller, which has been shown to perform well, which is the Stanley controller.

Known for its precision in lateral control and adaptability to diverse driving conditions, the Stanley Controller aligns with the team's objectives of superior handling, safety, and performance in autonomous vehicle development. It effectively calculates the vehicle's yaw error and lateral error using cubic spline fit, and commands a desired steering angle of

$$\delta = -\Delta\psi + \tan^{-1}\left(\frac{-K_e e_{lat}}{v}\right). \quad (5)$$

Here, K_e is the only gain to be tuned for the lateral controller to ensure responsiveness and stability of the vehicle's path tracking.

RESULTS

Improvement in feedforward controller: parameter tuning
Beginning with the Feedforward component of the Compound Feed controller, the K_{sg} value was optimized in order to have better predictive tracking performance. A total of 18 tests with varying speed, turning radii, and direction of turn were collected for several turns, as shown in Figure 19, to optimize the K_{sg} value. The improvement in the trajectory of the vehicle after optimizing K_{sg} over a nominal assumed K_{sg} is shown in Figure 20.

The radii of the corrected K_{sg} paths better adhere to the initial K_{sg} paths. This improvement was quantified into averaged lateral error in Table 8. The quantified data showed a 71% reduction in radial percent error. This improvement will facilitate easier gain tuning when working with the Feedback controller component.

Testing	Lateral Error (% of 9 m)
Pre Ksg	0.0989
Corrected Ksg	0.0290

Table 8: Ksg Radial Error

Comparison of Stanley and Compound Feed Controller
The Compound Feed Controller with implemented optimized feedback gains and the Stanley Controller across driving scenarios, including curved paths and straight

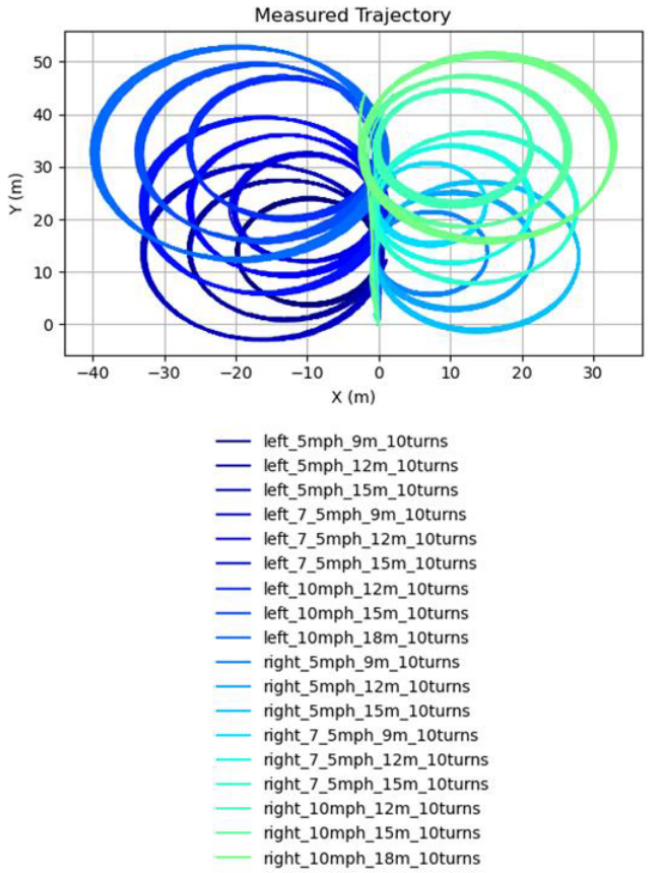

Figure 19: Experiments conducted for computing K_{sg}

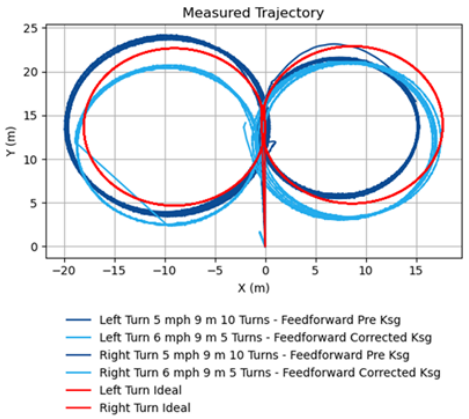

Figure 20: Feedforward Controller Optimization

lines conducted at 5 mph, to compare their trajectory adhering capabilities.

As illustrated in Figure 21, in curved path assessments, the Compound Feed Controller showed tighter adherence to the intended trajectory, particularly in turns, suggesting better curvature handling. In contrast, the Stanley controller exhibited slight outward deviations during turns, which might indicate a delayed response to curvature changes.

Heading error and lateral error, crucial metrics for assess-

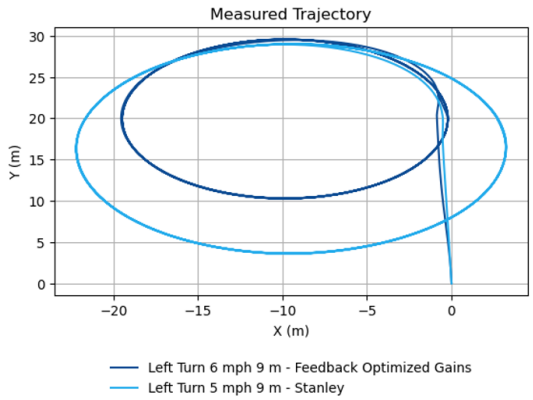

Figure 21: Curved Path Trajectories

ing autonomous navigation accuracy, were also analyzed, as shown in Figure 22. Notably, the Stanley controller occasionally allowed lateral errors up to 3 meters, exceeding safe limits for lane integrity. This underlines the importance of precise controller calibration to meet stringent navigational standards.

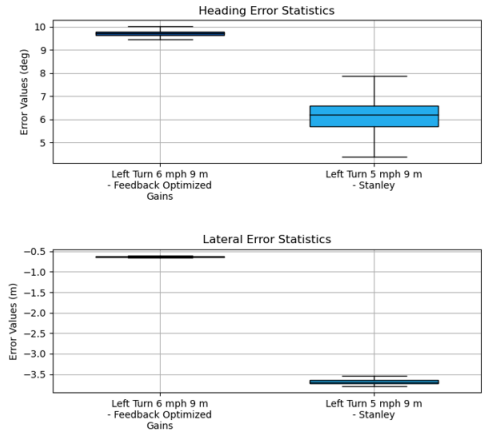

Figure 22: Curved Path Errors

When examining the lateral and longitudinal acceleration of both controllers for the curved paths, which are given in Figure 23 and Figure 24, respectively, we can observe that both meet the design requirements.

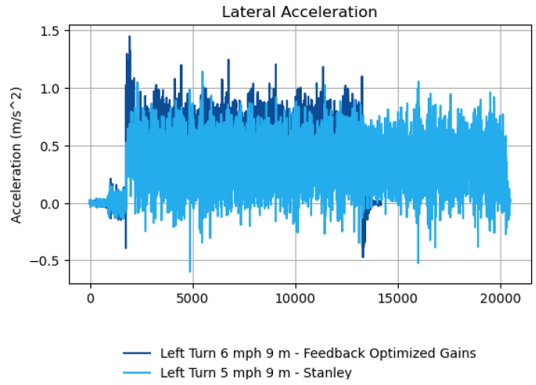

Figure 23: Curved Path Lateral Acceleration

On the other hand, in straight-line tests, both controllers

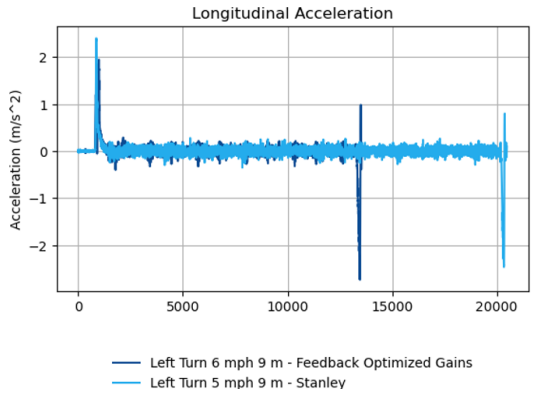

Figure 24: Curved Path Longitudinal Acceleration

effectively maintained linear paths, critical for highway driving, as shown in Figure 25. In the error comparison as shown in Figure 26, the Stanley controller displayed a lower median and less variability in heading and lateral errors, suggesting more consistent vehicle alignment with the intended trajectory. In the straight line test, the analysis of lateral and longitudinal accelerations showed that both controllers adhered to the stringent requirements set by the Vehicle Integration Challenge, which are shown in Figures 27 and 28, respectively.

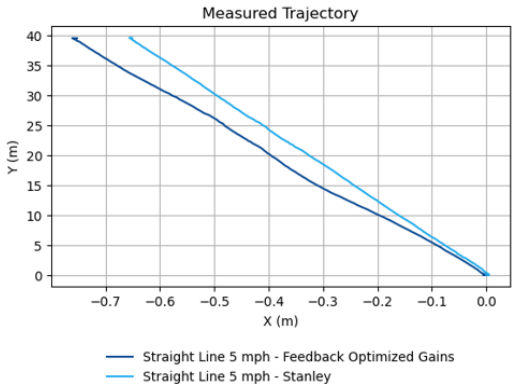

Figure 25: Straight Line Trajectories

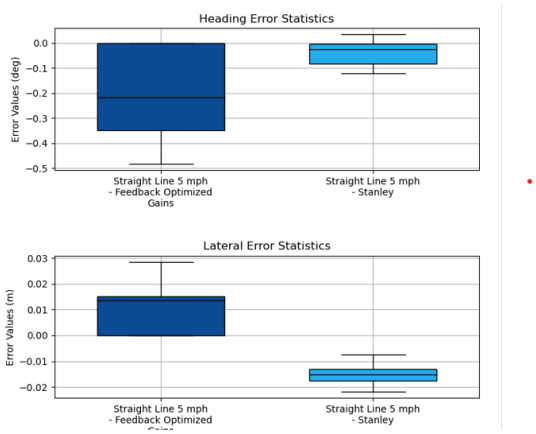

Figure 26: Straight Line Errors

The lateral and longitudinal accelerations and decelerations, which were consistently achieved by both controllers, affirmed their robustness in managing vehicle dy-

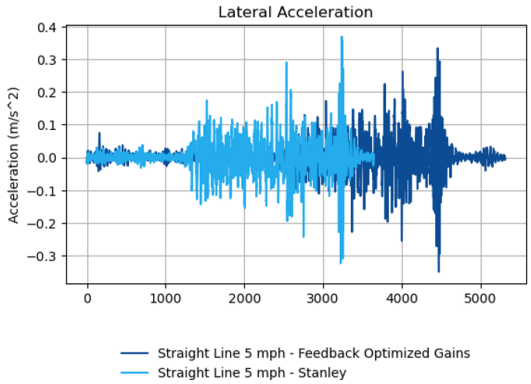

Figure 27: Straight Line Lateral Acceleration

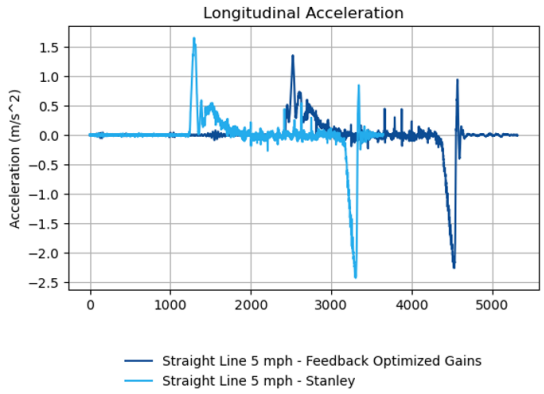

Figure 28: Straight Line Longitudinal Acceleration

namics under straight-line conditions. Overall, evaluations highlighted the strengths and limitations of each controller, with the Compound Feed Controller excelling in curvature handling and the Stanley Controller maintaining consistent heading accuracy in straight-line trajectories.

CONCLUSION AND RECOMMENDATION The experiments that were performed highlight the criticality of the controller design for curved line tracking, which took us nearly two months to perform well with the feedforward controller. However, the benefit obtained due to the tuning of the parameter is evident from the performance of the compound controller for curved trajectory tracking. From our comparison with a simpler controller, it is evident that the simpler controller can be used in conditions wherein the vehicle is not expected to perform complicated maneuvers involving circular trajectories. However, from the perspective of an AV manufacturer, we recommend the use of controllers with more error terms than just one, such as the compound feed controller, that can be tuned to optimize the performance of the controller.

STATIC AND DYNAMIC OBJECT DETECTION

Perception algorithms are deep-learning computer algorithms that allow a self-driving car to understand and respond to the world around it. These algorithms rely on sensory inputs to classify actors around the vehicle properly. The algorithms fuse information from sensors such as cameras and LiDAR, whose orientation and selection depend on the vehicle design. This fused data is then used to produce a clear picture of the world with the identification, location, and movement of the actors determined. [15] The algorithms used for static and dynamic object detection and tracking were discussed previously. Furthermore, the algorithms for traffic light, lane detection, and limit line detection were also discussed. In this section, the discussed modifications over stock algorithms and naive methods for these modules are presented since, from the perspective of a minimum viable product, it becomes imperative to understand the minimally functional algorithm to perform these tasks.

Remark: For the system designed for the car currently, we track only those signs, objects, traffic lights, lanes, and limit lines as specified in the rule book [2]. In this regard, our perception system performs the tasks required for the AutoDrive challenge and is, therefore, efficient.

TRAFFIC SIGN AND OBJECT TESTING The stock YOLOv8 method can detect a limited number of objects that are pertinent to driving within an urban environment. These objects include cars, stop signs, and people. This limited selection of objects is not adequate for a minimum viable product that must be capable of maneuvering safely within an urban environment while adhering to the rules of the road. That is why the 12th Unmanned team worked to train the YOLOv8 software on more objects. These additional objects include barrel cones, type 3 barricades, deer, yield signs, speed signs, left-turn-only signs, right-turn-only signs, pedestrians, and railroad crossing signs. These objects were included within the detection algorithm because they are common, directly affect vehicle behavior, and can be encountered during competition within the M-City environment. If the vehicle is not capable of recognizing and reacting to these objects, then driving conditions can be considered unsafe. Figure 29 above shows an example of the stock YOLOv8 failing to identify an untrained barrel cone when compared to the successful identification from the modified algorithm.

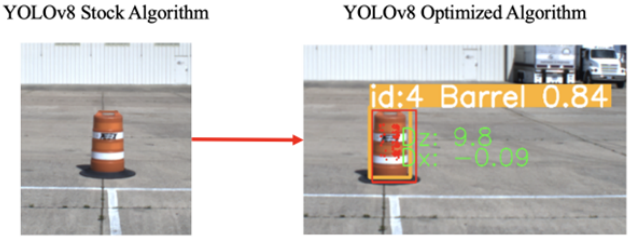

Figure 29: Stock vs Optimized Algorithm Recognition of Barrel Cone from Same Image

The static and dynamic testing below serves to evaluate the autonomous vehicle within the hypothetical operating environment of the competition. The testing field of view is restricted to a 40-meter radius from the center of the

sensor suite as restricted within the competition rulebook [2]. The center camera and LiDAR sensor were used as a proof of concept for the perception and tracking algorithms. Testing was occupied within a 60-degree field of view in the direct center of the vehicle, as seen in Figure 30.

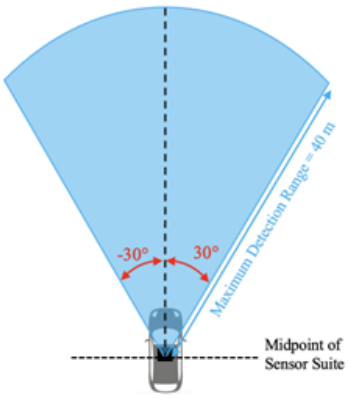

Figure 30: Testing Perception Boundaries

Throughout testing, ten different objects were used to test the software. These objects can be split into two distinct categories: road signs and obstacles. The road signs include stop signs, right-turn-only signs, left-turn-only signs, yield signs, speed limit signs, and railroad signs. The four obstacles used in testing include a barrel cone, a pedestrian, a deer, and a vehicle. All objects tested were either real-life objects or life-size models for accurate testing conditions.

Static testing overview Static testing was composed of taking instances of all ten objects in 12 distinct, known positions within the restricted 60-degree field of view of the center camera and LiDAR. The experimental distances can be seen in Figure 31. These instances were then analyzed using the standard and modified perception algorithms and compared for robustness and accuracy. Three signs/objects were present in each captured instance, and all objects were rotated between the various positions and distances. This allowed each object to be tested at every distinct position. All distinct x and z positions were recorded as a ground truth condition to compare with the algorithms' outputs within the results section.

Dynamic testing overview Dynamic testing was conducted to test the sensor and software robustness of the autonomous vehicle's system under constantly varying conditions. The dynamic testing consisted of two independent moving variables: the ego vehicle and one actor. The experiments iterated through conditions where either one or both of the independent variables were moving, creating three possible dynamic testing scenarios: one where the ego vehicle is in motion, one where an actor is in motion, and one where both are in motion. For this experiment, the actors in motion were limited to objects

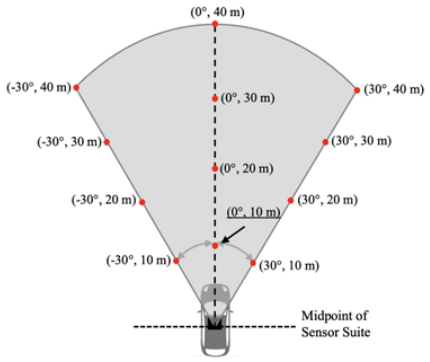

Figure 31: Static Testing Points of Interest

commonplace on local roadways. The objects selected were a pedestrian, a deer, and another vehicle. The static objects used included the stop sign and the yield sign due to their direct impact on the vehicle's behavior. The dynamic procedure was conducted within the parameters given, and all speeds were kept at under 10 mph.

The first set of dynamic testing performed included the movement of an actor in front of the stationary autonomous vehicle as depicted in Figure 32A. A pedestrian, a deer, and a vehicle drove a straight path perpendicular to the autonomous vehicle at distances varying from 15 to 25 meters from the sensor suite. The second round of dynamic testing built on these experiments by moving a pedestrian, a deer, and a vehicle through a straight path 15 to 25 meters from the autonomous vehicle's starting point and allowing the autonomous vehicle to drive in a straight path forward. This can be seen in Figure 32B.

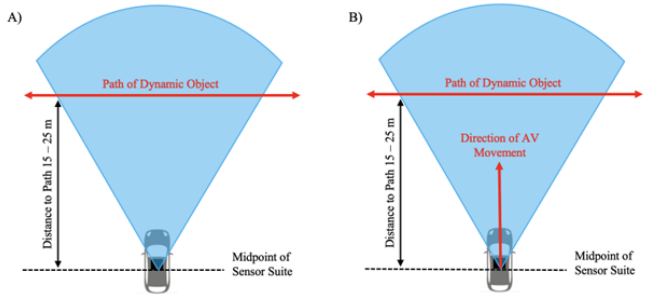

Figure 32: A) Dynamic Testing Set 1, B) Dynamic Testing Set 2

RESULTS

Static Testing Results During static testing, a total of 40 tests with 120 data points were performed, and both classification and depth perception data were collected. The static testing results can be split into two main categories: the performance comparison of both the stock and modified YOLOv8 classification models and the depth perception performance.

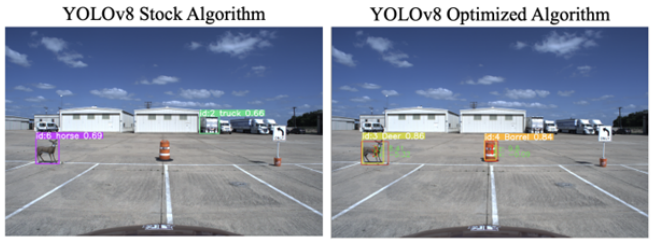

Figure 33: Comparison of Algorithm Performance on Same Image

The stock and modified YOLOv8 algorithms were compared by running both algorithms against the same instance or camera image. An example of this can be seen in Figure 33. Data was collected on whether the object was perceived and the accuracy of the perceived objects for each algorithm in every instance.

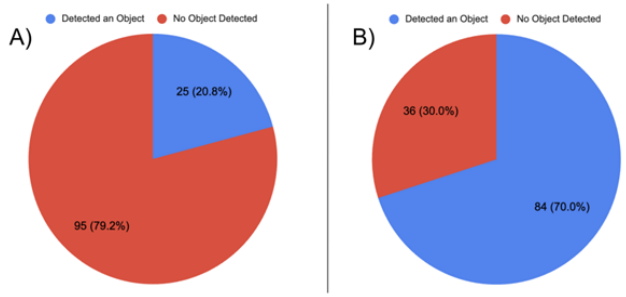

Figure 34: Ability for Algorithm to Detect Objects, (A) Stock YOLOv8 Model (B) Modified Model

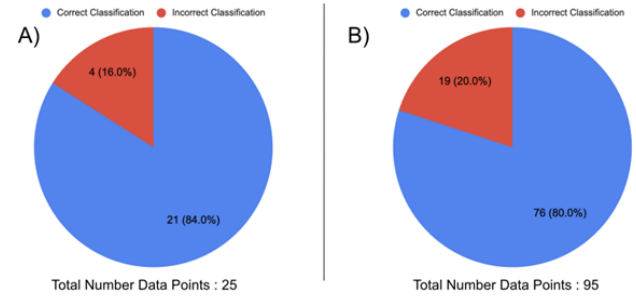

Figure 35: Accuracy on Objects Detected, (A) Stock YOLOv8 Model (B) Modified Model

The modified YOLOv8 algorithm was found to be significantly more effective in object detection when compared to the stock YOLOv8 algorithm, as seen in Figure 34. This is due to the modified YOLOv8 algorithm's additional object training. When comparing the accuracy of the objects detected, the overall accuracy was relatively equivalent with the stock YOLOv8 algorithm having 84% accuracy and the modified algorithm YOLOv8 algorithm having 80% accuracy as seen in Figure 35. This indicates that the YOLOv8 algorithm continues to perform at the same accuracy even when more training inputs have been introduced to the algorithm.

The performance between the two algorithms was also

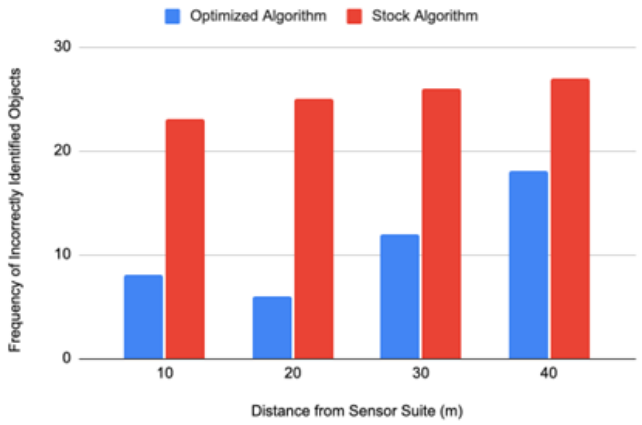

Figure 36: Frequency of Incorrectly Identified Objects by Distance and Algorithm Type

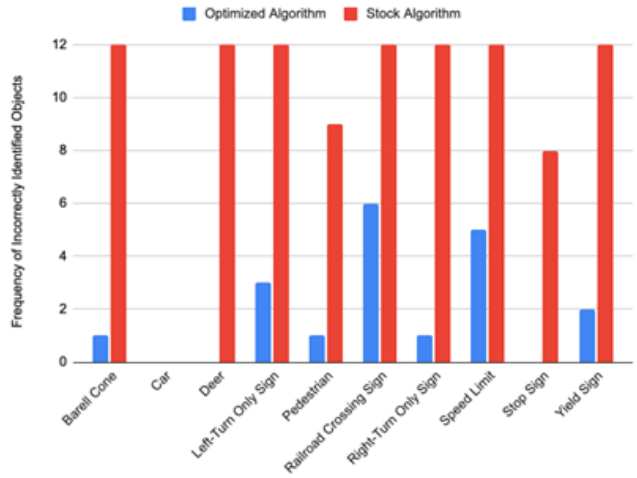

Figure 37: Frequency of Incorrectly Identified Objects by Distance and Algorithm Type

analyzed against the distance of the object from the sensor suite and how well the algorithms performed on each individual object tested. These results can be seen in Figures 36 and 37 respectively. The ability to detect objects for both algorithms was found to be inversely proportional to the increased distance from the sensor suite. This is due to objects at further distances containing a small amount of pixels, which are more difficult to identify than objects closer to the camera, which take up more pixel space. The modified algorithm performed better overall with a smaller frequency of identified objects across all distances. The accuracy per object was found to have this same trend, with the stock algorithm having a higher number of incorrectly identified objects for all objects other than the car. The car was found to have no incorrectly identified objects because it was trained on both algorithms and was large enough to be identified at larger distances. The railroad crossing and speed limit signs were found to have the most incorrectly identified instances for the modified algorithm, which could indicate that the modified model needs additional training for these objects.

Overall, the performance of the modified YOLOv8 classification algorithm was significantly better than the stock YOLOv8 algorithm and would be a better algorithm decision for the minimum viable product to ensure safe vehicle operation in urban environments.

The depth perception values were used only on the modified YOLOv8 algorithm due to its larger value of unique data points with depth data. The accuracy of the depth perception algorithm in the x and z was compared against the distance from the sensor suite and the individual object type.

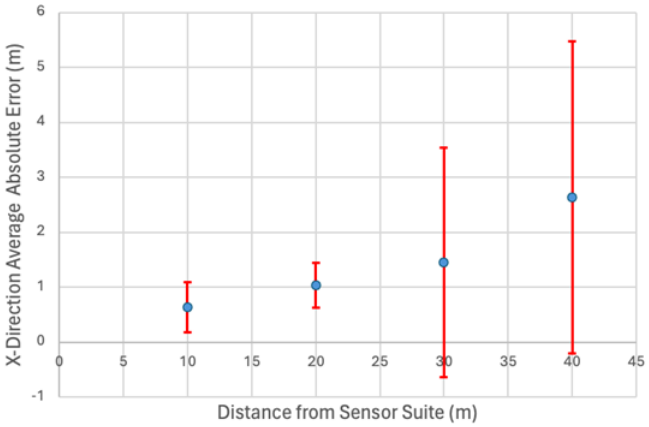

Figure 38: Average Absolute Error and Standard Deviation in x Position by Distance

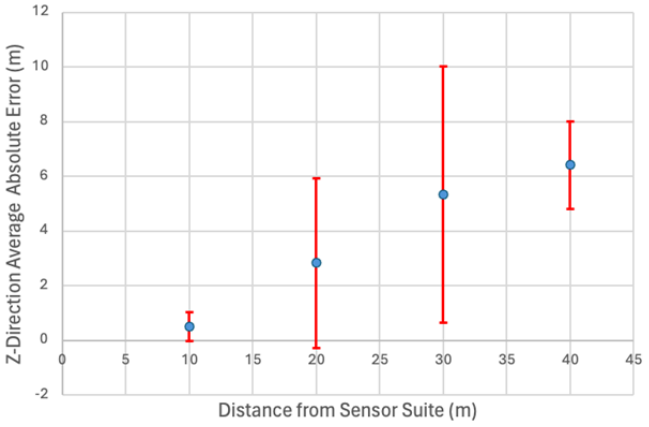

Figure 39: Average Absolute Error and Standard Deviation in z Position by Distance

Figure 38 and 39 have the average absolute error with standard deviations as a function of the distance of the object from the sensor suite in the x and z directions, respectively. It can be seen that the average absolute error increases as the distance from the sensor suite increases. This is due to the smaller image space taken up by objects at further distances. The number of selected LiDAR points is smaller, and the likelihood of picking up data from the background of the object is larger.

Figure 40 and 41 have the average absolute error with standard deviations for each individual object in both x

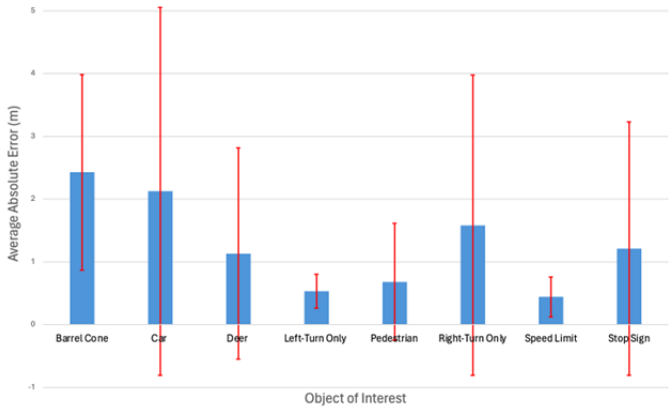

Figure 40: Average Absolute Error in X Position by Object of Interest

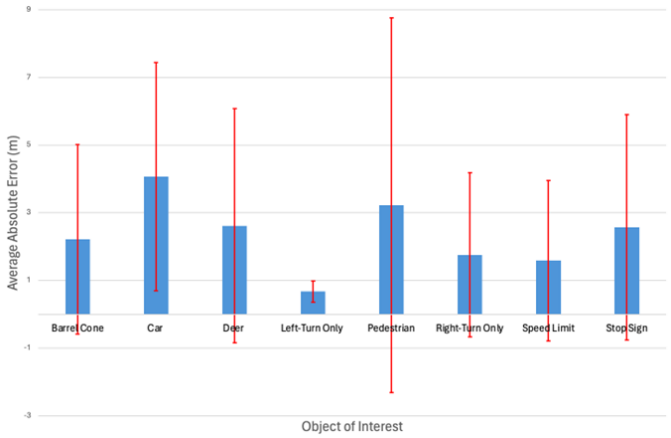

Figure 41: Average Absolute Error in Z Position by Object of Interest

and z directions, respectively. The error and standard deviation of the car were the largest in the x direction due to the object's large size. This makes the exact location of the object's center more difficult to identify, creating more errors. The pedestrian objects had the largest variance due to their irregular shapes and the difficulty of selecting LiDAR points that accurately represent the pedestrian's location. This is even more prevalent at larger distances.

Overall, the absolute error of the depth perception is relatively low, with the larger errors occurring at larger distances where the vehicle has large enough reaction times to maneuver safely. The depth perception is adequate for a minimum viable product but can be improved upon with additional point filtering methodologies in the future.

Dynamic Testing Results For the dynamic testing, a moving car, pedestrian, and deer were tested, and three data sets were collected for each object in motion. The relative velocity of the dynamic object was measured by the LiDAR sensors on the vehicle. The measured velocities were compared to the true velocities. The autonomous system's ability to correctly detect and identify

objects followed the results found in the static portion of the testing. The LiDAR correctly identified both the car and the pedestrian greater than 98% of the time, as shown in Figure 42. However, Figure 43 shows that the LiDAR sensor and dynamic object detection system had a high absolute error when determining dynamic object velocity. The LiDAR measured lower speeds than the known, true values for all dynamic objects. The car had the highest error due to its relatively large bounding box. While the deer and the pedestrian were similar in size, the deer had a larger absolute velocity error that was attributed to more algorithm training for pedestrians than deer.

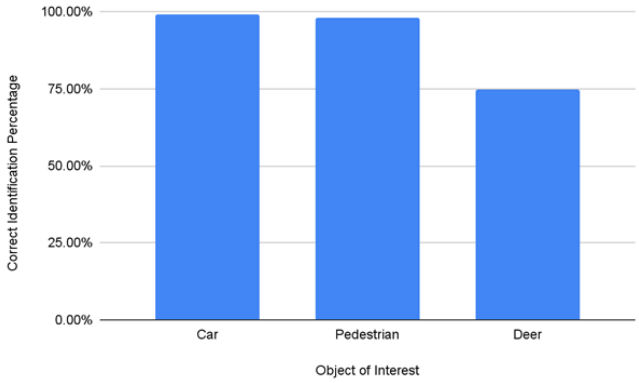

Figure 42: Average Percentage of Correctly Identified Dynamic Data Points

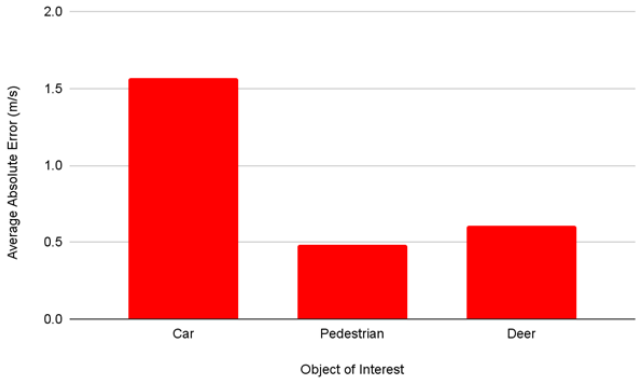

Figure 43: Average Absolute Error of Dynamic Object Velocities

LANE AND LIMIT LINE DETECTION Our current design incorporates two methods of lane detection: using the HD map and GPS data, as well as SCNN. The HD map provides the lanes available based on the GPS location of the vehicle and is the simplest method of obtaining reliable lane information if the manufacturer has access to the map from the Dynamic Map Platform. The second approach to detecting lanes is through image processing. Many standard algorithms exist for lane detection through neural network-based methods. One such method is SCNN, which is a Sequential convolutional neural network. A sample result obtained from such an implementation is shown in Figure 44, wherein the three-lane lines corresponding to the direction of the vehicle were identified by the algorithm. Due to the low cost associated with implementing the algorithm, due to the widely available resources online, and due to the extensive studies that have explored the implementation of such algorithms for different scenarios, utilizing standard algorithms for lane detection suffices for lane detection for a minimally viable product.

Figure 44: Lane detection using SCNN

Similar to lane detection, we use two methods for limit line detection: using the HD map and GPS data and color-based thresholding methods. Contrary to lane detection based on color-based thresholding, we observed that this method performs better for stop-line detection. For this method, given an image from the camera, a region of interest is selected as shown in Figure 45, and through perspective projection, a top-down view of the image is obtained, as shown in Figure 46. Through the application of gaussian blur, the pixels corresponding to the limit line are amplified. Further, using the gaussian blur, the pixels in the horizontal direction are highlighted, after which the horizontal line can be obtained, as shown in Figure 47. The final limit line obtained is shown in Figure 48. Similar to the conclusion for the lane detection module, the algorithm outline here utilizes standard image processing techniques available through OpenCV [7] and can be observed to perform in dim light conditions as well. Hence, such a standard available algorithm can be used for limit line detection for a minimally viable product.

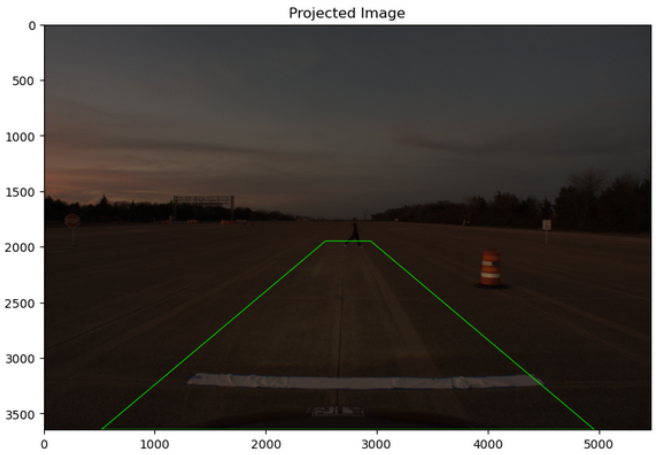

Figure 45: Trapezoidal region of interest

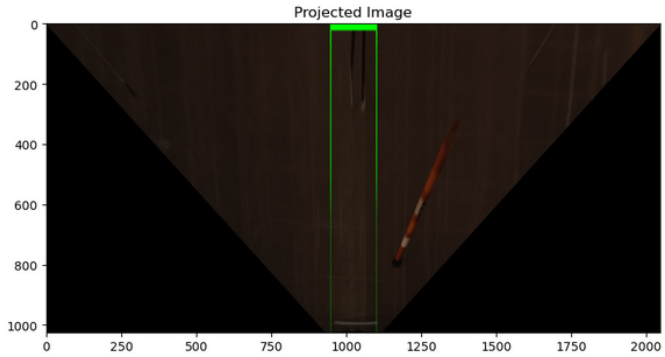

Figure 46: Top-down view for limit line

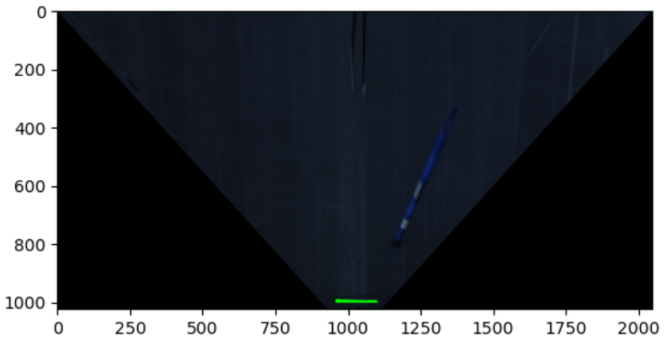

Figure 47: Horizontal line output after blur

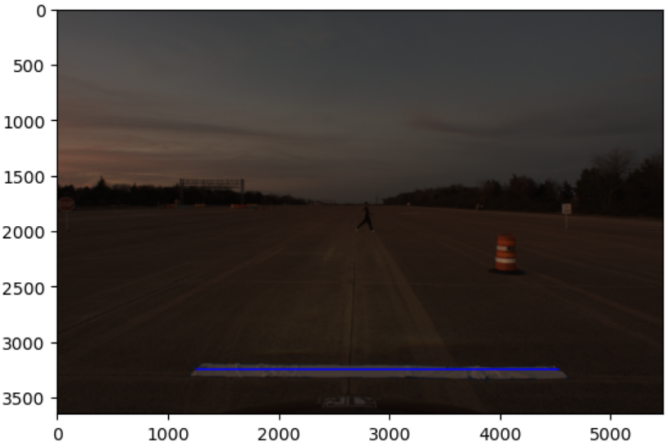

Figure 48: Limit line output

TRAFFIC LIGHT DETECTION The traffic light detection module for the current vehicle design uses a two-staged approach as discussed previously: utilizing YOLO for identifying the traffic light head and then utilizing a custom-trained neural network for identifying the color and shape. The traffic light module is critical for correct vehicle behavior at the intersections; in this regard, it is imperative that the minimally viable product does not compromise the accuracy of the detection. In this regard, two naive approaches are considered: utilizing stock YOLOv8 for obtaining the traffic light head and state or using color thresholding-based state identification and shape identification using OpenCV techniques [7].

The OpenCV-based method is computationally inexpensive and is easy to debug and tune. However, small changes in the brightness and/or contrast require parametric returning. Furthermore, the algorithm is less robust as environment objects can be perceived as objects of interest based on color similarities. Figures 49 demonstrate the results of this approach. The environment objects such as leaves of trees are falsely perceived as traffic lights.

Figure 49: OpenCV based identification of traffic light

Another naive approach is to utilize YOLOv8 to identify the traffic light and state directly. This approach is better than the OpenCV-based approach since it is more robust. Furthermore, light tracking becomes easier, as YOLO accounts for it. However, the stock model can fail for scenarios wherein the model is not trained, such as in Figure 50. Furthermore, we identified issues in predicting the state of the light for traffic light heads with more than two lights on at the same time, such as in Figure 51.

Figure 50: New scenario encountered by stock YOLO

Instead, utilizing the two-staged approach, wherein the stock YOLO model identifies the traffic light head followed by a custom-trained YOLO model for state identification,

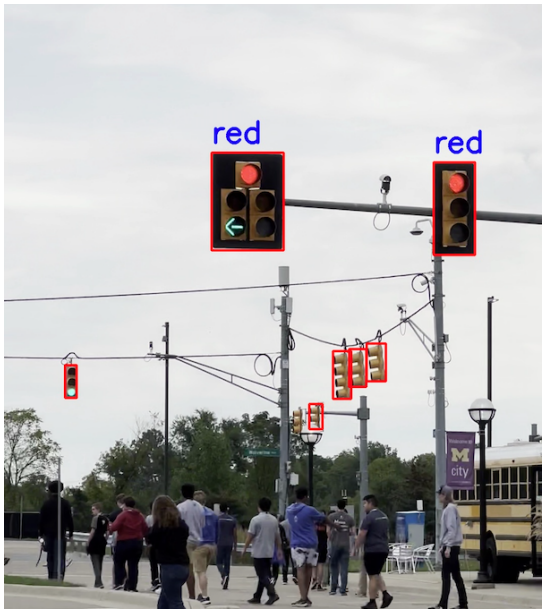

Figure 51: Scenario with multiple traffic lights encountered by stock YOLO

Figure 53: Two-staged custom YOLO-based identification of traffic light

can help rectify the above-stated issues. Figure 52 shows a flowchart for the proposed method, which is the current method used on our vehicle, wherein YOLOv8x, which is a larger YOLO model, is used for the head detection, and a smaller custom model is used for the state. Figure 53 shows the effectiveness of the developed approach in detecting traffic lights.

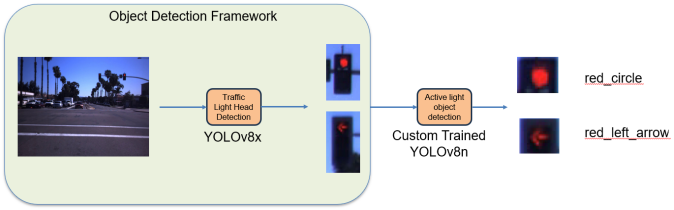

Figure 52: Framework utilized for traffic light detection on car

A dataset of a total of 25 different frames was considered from our visit to MCity last fall. The OpenCV framework demonstrated a 60% accuracy in detecting traffic lights, however over 50% false positives were detected. The direct YOLO approach assigns only one label to each detected traffic light, and it fails when multiple traffic light heads or colors are simultaneously active. Consequently, this approach was not considered for further development. Finally, the two-staged approach successfully detected tracking traffic lights over 90% of the time with less than 10% false positives on the same dataset.

CONCLUSION AND RECOMMENDATION The Minimum Viable Product for autonomous vehicles relies on static and dynamic perception of objects by the car's sensors and perception algorithms. Last year's concept design was developed into a minimally viable product by training the perception algorithms to correctly identify the minimum signage and actors to navigate in commercial applications safely. The results from the static and dynamic experiments on the autonomous vehicle demonstrated that the vehicle's overall performance in static situations has dramatically improved with the addition of the modified YOLO. A 237% increase in static object identification was achieved in the trained YOLOv8 algorithm. The increase in the number of identifiable objects and the identification accuracy demonstrate a significant improvement over the previous year's concept. Similar observations were obtained for the traffic light detection module, wherein a custom-developed module substantially outperformed stock models available for light detection, with significantly fewer false positives. However, from the analysis of lane detection and limit line algorithms, we can observe that there are applications for stock algorithms that are widely available which can perform well. In this regard, we observe that custom models may need to be developed depending on the module considered.

Remark: Through the detailed analysis that was performed in this section, wherein we explored simpler algorithms that could help reduce the number of resources used, we observe that the perception stack developed for our current design performs the tasks required for the SAE AutoDrive challenge. Furthermore, the simpler approaches explored were observed to fail to perform the required tasks, such as traffic light detection, lane detection, and static object detection. Therefore, we have developed a minimally viable perception stack for the competition, keeping in mind the robust performance required from the perception stack for reliable performance from the autonomous vehicle.

POSITIONING & ROUTE PLANNING

Autonomous vehicles (AV) need accurate positioning and advanced route planning for safety and efficiency. This section explores navigation technologies and systems required for precise vehicle location and optimal pathfinding under various conditions.

POSITIONING AND LOCALIZATION AVs achieve precise positioning through a combination of technologies that ensure safety and accuracy. The foundation relies on GPS and its enhanced form, Differential GPS (DGPS), which uses ground-based stations to correct GPS signals, improving accuracy. However, GPS is not always reliable, such as in urban environments where signals can be obstructed. To counteract this, Inertial Measurement Units (IMU), consisting of accelerometers and gyroscopes, track the vehicle's motion and orientation independently. They are prone to drift, however.

Complementing these are LiDAR and radar technologies. LiDAR creates detailed 3D maps using laser pulses, while radar uses radio waves to detect objects' distance and speed, both essential in poor visibility. Cameras equipped with computer vision identify road features and traffic signs, integrating this data into the vehicle's navigation system. High-definition maps and Simultaneous Localization and Mapping (SLAM) algorithms provide and continuously update a detailed road map, enhancing navigation [20].

For the minimally viable product, the simplest method of localization is to utilize the highly reliable GPS data obtained from selected NovAtel PwrPak 7D-E1 in the sensor suite with the TerraStar L5 subscription selected. Similar to the current vehicle design, wherein we utilize the NovAtel PwrPak 7D-E2 with a TerraStar C subscription to obtain highly precise location and heading angle information, the minimally viable product can utilize the obtained location using the HD map provided by Dynamic Map Platform. The HD map provides a highly accurate and detailed data representation of the AutoDrive Competition site at MCity in Ann Arbor, Michigan. The HD Map module that we have developed receives GPS data in a continuous stream over the GPS data topic, and outputs the upcoming limit lane, traffic light, lane lines, etc. The HD map itself is in the form a postgreSQL database. Upon receiving a request for HD map data from another module, the HD Map module generates a series of SQL commands necessary to query the requested information from the database. When the database query has been completed, the module will process the returned data and pack it into a list of the requested information, which is then transmitted back to the module that made the request. An alternative approach would be to utilize openly available maps for localization, such as OpenStreetMap. However, OpenStreetMap lacks a detailed description of the lanes, limit lines, traffic lights, among others, which is described in depth in the upcoming section.

ROUTE PLANNING Route planning for AVs involves sophisticated processes and algorithms to navigate from start to end efficiently and safely. This includes calculating the optimal route and adapting in real-time to environmental factors. Algorithms like A* and Dijkstra's are used to find the shortest path through a known data graph like Google Maps, while incremental algorithms such as D* or D* Lite allow for dynamic recalibration of the route as conditions change. These algorithms handle both global path planning, which plans a complete path, and local path planning for immediate navigation challenges.

The use of AI, machine learning, and various sensors enables continuous data collection and processing, enhancing the vehicle's understanding of its environment, detecting obstacles, and anticipating other road users' actions. This technology allows the vehicle to adjust its route continually, ensuring safety and efficiency.

Additionally, digital mapping services like Google Maps are utilized for route planning by providing real-time traffic updates and route suggestions. These services help in plotting efficient travel paths and are helpful in aiding AVs to make informed routing decisions based on current traffic conditions and road statuses.

Global planning We currently use Dijkstra's algorithm [8] for route planning with a map from Dynamic Map Platform (DMP), known for its dense and detailed data. While effective for optimal pathfinding, the map's level of detail is not practical for broader automotive industry applications due to scalability challenges. Dijkstra's algorithm excels by calculating the least costly path, factoring in distance, traffic conditions, and road types, but requires efficient data management to handle the map's complexity.

Alternatively, detailed maps required for autonomous navigation can be obtained through OpenStreetMap (OSM), an open-source geographical database with information provided by collaborating users. OSM has nodes for use by an autonomous driving system. However, it is much less dense and precise, as can be observed from Figure 54. While this means faster computation time for pathfinding, more specific road conditions would need to be determined by GPS or other sensors. In Figure 54, the graph on the left is obtained from DMP, whereas the graph on the right is obtained from OSM.

To inform our decision to use either DMP or OSM, we compared the computational time and total path length of running our algorithm on the same ten different routes shown in Table 9. A comparison of the routes is shown for one instance in Figure 55.

Table 9 shows that DMP requires more computational time to plan a path. However, all times were less than a second, so computational time becomes less of a factor. Comparing path lengths, they were very similar for a run. The differences come from the definition between the

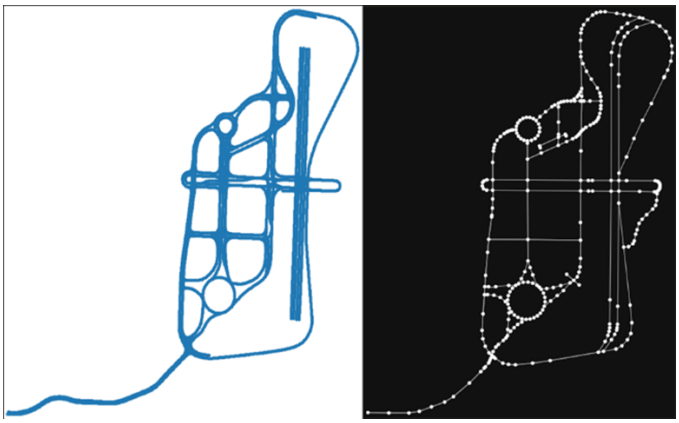

Figure 54: DMP vs. OSM nodal map for MCity

Run	DMP Length (m)	DMP Comp. Time (s)	OSM Length (m)	OSM Comp. Time (s)
1	444.53	0.641	454.31	0.016
2	102.15	0.344	107.04	0.016
3	126.28	0.375	129.79	0.047
4	72.75	0.351	81.92	0.031
5	145.76	0.312	146.76	0.031
6	155.53	0.344	161.334	0.031
7	324.19	0.437	331.23	0.016
8	452.86	0.594	457.63	0.047
9	399.59	0.516	406.78	0.031
10	1051.16	0.984	1052.15	0.047

Table 9: Input Map Comparisons

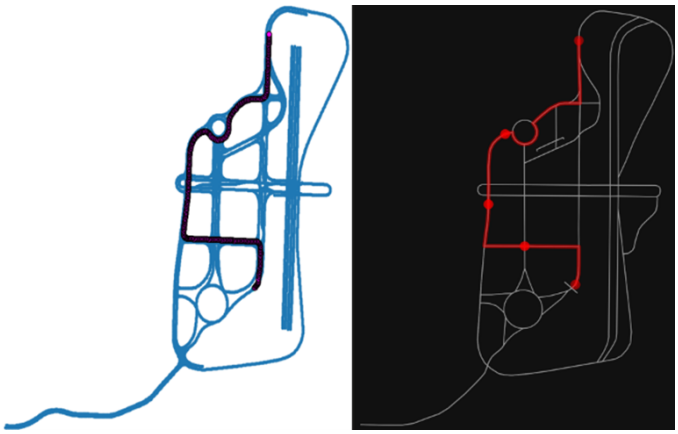

Figure 55: Similar routes generated using both methods

two maps. It is hard to align nodes selected on OSM to DMP due to DMP's density of nodes. Additionally, curves, turns, and lane changes are more defined in DMP than OSM (Figure 56), which also possibly led to lesser path length in DMP.

Local planning Having selected the map provided by DMP, the method utilized by the current vehicle's local planner, which utilizes the HD map and LiDAR-based clustering data and image recognition, is a viable method

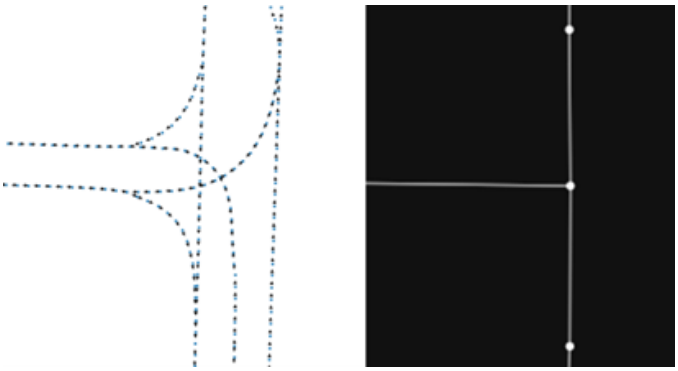

Figure 56: DMP vs. OSM T-intersection

for a minimally viable product due to the robustness and ease of implementation. Once the vehicle is localized, the HD map provides the lane information, based on which lane points for a lookahead distance can be obtained. Furthermore, the global path obtained from the global planner without accounting for obstacles, the local planner, which reroutes the vehicle in the presence of obstacles, and the lane markings are available to the user through Rviz, a visualization tool for the Robotic Operating System (ROS). A sample output of the same is shown in Figure 58, wherein the path in green denotes the global path, the path in orange depicts the local path, the red objects depict obstacles, and the blue lines depict the lane markings.

Figure 57: Sample barrel testing

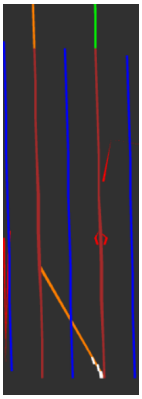

Figure 58: Sample RViz output for rerouting vehicle

Through the use of the available resources, a simplified algorithm for the local planner has been developed and

can be used for the minimum viable product due to the reduced number of resources required for the same.

Conclusion and recommendation DMP also comes with a price, unlike OSM, which is free. Both can be easily implemented with Python packages for route planning. However, DMP offers consistent and accurate updates, including crucial data like traffic signals and road closures vital for AVs. The extra information contained in DMP's map, such as multiple lanes, lane changes, and detailed turns that are not in OSM is crucial. If OSM is used, more time and resources would be spent to be able to do the more complex maneuvers already included in DMP's map. This could end up costing more than just selecting DMP in the first place. Thus, DMP's precision, accuracy, and reliability make it preferable over OSM.

When it comes to selecting the minimum viable products and options for an AV to use for positioning and route planning, a fine balance between cost and performance is seen. The minimum product should be precise and accurate enough to ensure that an AV is reliable to use. Neglecting this can lead to fatal mistakes, such as road accidents and traffic laws being broken. A theoretical example can be shown below in Figure 6. An intersection is shown in Google Maps and Rviz, a visualization tool for the Robotic Operating System (ROS), which is used to visualize the HD map provided by DMP. The black ovals in both represent an AV. The figure on the left from Google Maps is an outdated dataset of Rellis that is available for free, whereas the data from the HD map is current. It can be observed that if one were to rely on information that is not accurate, such as Google Maps, in this case, and if the given data is off by a few meters, the AV could be in an entirely wrong position and complicate any other vehicles in the intersection. Therefore, utilizing highly accurate maps to localize the vehicle and perform global and local route planning becomes imperative for a minimally viable product to ensure vehicle safety.

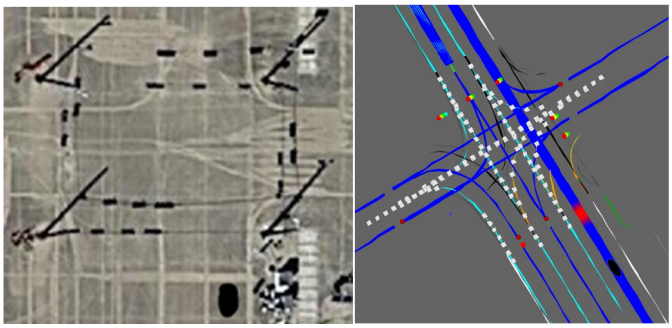

Figure 59: Intersection in Google Maps vs. RVIZ

CONCLUSION

In this document, we have described the updates to the software and hardware from year 2 of the competition to tackle the new challenges introduced in year 3 of the AutoDrive competition. Furthermore, we propose a minimally viable design from various aspects such as sensor suite, vehicle dynamics, static and dynamic object tracking, and positioning and route planning. We finally highlight the design for the minimally viable product for each of these aspects based on our prior experience in the competition and our team's design.

PATENTS, PAPERS, AND CONFERENCES

- Aaron Kingery, Dezhen Song, *Improving Ego-Velocity Estimation of Low-cost Doppler Radars for Vehicles*, IEEE Robotics and Automation Letters, vol. 7, no. 4, pp. 9445-9452, Oct. 2022.

REFERENCES

[1] "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles," SAE Standard J3016B, June 2018.

[2] SAE, "SAE AutoDrive Challenge II Year 3," 2023.

[3] D. S. Lal, A. Vivek and G. Selvaraj, "Lateral control of an autonomous vehicle based on Pure Pursuit algorithm," 2017 International Conference on Technological Advancements in Power and Energy (TAP Energy), Kollam, India, 2017, pp. 1-8, doi: 10.1109/TAPENERGY.2017.8397361.

[4] M. Liu, K. Chour, S. Rathinam and S. Darbha, "Lateral Control of an Autonomous and Connected Following Vehicle With Limited Preview Information," in IEEE Transactions on Intelligent Vehicles, vol. 6, no. 3, pp. 406-418, Sept. 2021, doi: 10.1109/TIV.2020.3033773.

[5] M. Bjelonic, "YOLO ROS: Real-Time Object Detection for ROS", 2018.

[6] S. Lee, H. Lim and H. Myung, "Patchwork++: Fast and Robust Ground Segmentation Solving Partial Under-Segmentation Using 3D Point Cloud", arXiv, 2022.

[7] G. Bradski and A. Kaehler, "The OpenCV Library", Dr. Dobb's journal of software tools, vol. 3, 2000.

[8] E.W. Dijkstra, "A note on two problems in connexion with graphs," Numerische mathematik, 1(1), pp.269–271, 1959.

[9] "Products." NovAtel, NovAtel, novatel.com/products. Accessed 01 Apr. 2024.

[10] "Combined Systems." NovAtel, NovAtel, novatel.com/products/gnss-inertial-navigation-systems/combined-systems. Accessed 15 Apr. 2024.

[11] S. Kundu, "Understanding geometric path tracking algorithms - Stanley Controller," Medium, https://medium.com/roboquest/understanding-geometric-path-tracking-algorithms-stanley-controller-25da17bcc219 (accessed Apr. 8, 2024).

[12] J. M. Dolan, “Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments,” Carnegie Mellon University, 2016, ppms.cit.cmu.edu/media/project_files/2016_Dolan.pdf. Accessed 10 Apr. 2024.

[13] NovAtel. (n.d.). TerraStar Correction Services. NovAtel. https://novatel.com/products/gps-gnss-correction-services/terrastar-correction-services.

[14] L. Chen, F. Zheng, X. Gong, X. Jiang, “GNSS High-Precision Augmentation for Autonomous Vehicles: Requirements,” Solution, and Technical Challenges. Remote Sensing. 2023; 15(6):1623. https://doi.org/10.3390/rs15061623 Accessed 10 Apr. 2024.

[15] C. Xiang, ”Multi-Sensor Fusion and Cooperative Perception for Autonomous Driving: A Review,” in IEEE Intelligent Transportation Systems Magazine, vol. 15, no. 5, pp. 36-58, Sept.-Oct. 2023, doi: 10.1109/MITS.2023.3283864.

[16] Mean average precision (MAP): A complete guide. Kili Technology. (n.d.). https://kili-technology.com/data-labeling/machine-learning/mean-average-precision-map-a-complete-guide.

[17] M. Banoula (2023, August 29). Naive Bayes classifier: Simplilearn. Simplilearn. https://www.simplilearn.com/tutorials/machine-learning-tutorial/naive-bayes-classifier.

[18] Boesch, Gaudenz. “A Guide to YOLOv8 in 2024.” viso.ai, https://viso.ai/deep-learning/yolov8-guide/. Accessed 17 April 2024.

[19] S. Joshi. “Top Object Detection Models in 2024.” hitech, https://www.hitechbpo.com/blog/top-object-detection-models.php

[20] ”Wevolver.” LIDAR SLAM - Wevolver, Wevolver, www.wevolver.com/article/lidar-slam Accessed 01 Apr. 2024