MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT & ENGINEERING

TEAM DARVIN

Design Report

Vehicle name: SOCRATES 2.0

Prof. Kashyap Joshi - Faculty Mentor - kashyap.joshi@nmims.edu

Sarthak Mishra - Team Captain - sarthak.mishra25@nmims.edu.in

Date Submitted: 13th May 2023

Team Members

Vaibhav Raheja - Co Captain - vaibhav.raheja34@nmims.edu.in

Parthak Mehta - Vice Captain - parthak.mehta23@nmims.edu.in

Rudra Makwana - Core Member - rudra.makwana19@nmims.edu.in

Harshil Shah - Core Member - harshil.shah22@nmims.edu.in

Pranav Lavande - Core Member - pranav.lavande029@nmims.edu.in

Saksham Gupta - Team Mentor - saksham.gupta82@nmims.edu.in

I, Professor Kashyap Joshi, hereby declare that the work done by Team D.A.R.V.I.N under my guidance for the IGVC competition 2022 has been significant and equivalent to what might be awarded credit in a senior design course.

Prof. Kashyap Joshi
EXTC Department
NMIMS MPSTME
(kashyap.joshi@nmims.edu)

Contents

Team Organization

After many hours of tedious hard work, Team Darvin achieved 3rd place in IGVC 2022 - Autonav. For IGVC 2023, Team Darvin embarked on an ambitious project to enhance the capabilities of the autonomous robot, SOCRATES 2.0. Through collaborative efforts and a multidisciplinary approach, our team worked tirelessly to push the boundaries of autonomous vehicle technology. In this report, we present our progress, design considerations, and solutions implemented to enhance SOCRATES 2.0's performance and autonomy. Our dedication, expertise, and teamwork have been instrumental in overcoming challenges and achieving remarkable results. As we continue our pursuit of innovation, we are excited for the future advancements that lie ahead. The work for this project was divided into mainly four sections: Mechanical, Electronics, Software, and Administrative.

Name	Position	Major work
Sarthak Mishra	Team Captain	Software, Electronics, Administrative
Vaibhav Raheja	Co Captain	Software, Mechanical, Administrative
Parthak Mehta	Vice Captain	Cyber Challenge, Administrative
Rudra Makwana	Core Member	Mechanical, Software
Harshil Shah	Core Member	Mechanical
Pranav Lavandhe	Core Member	Electronics

Design Process

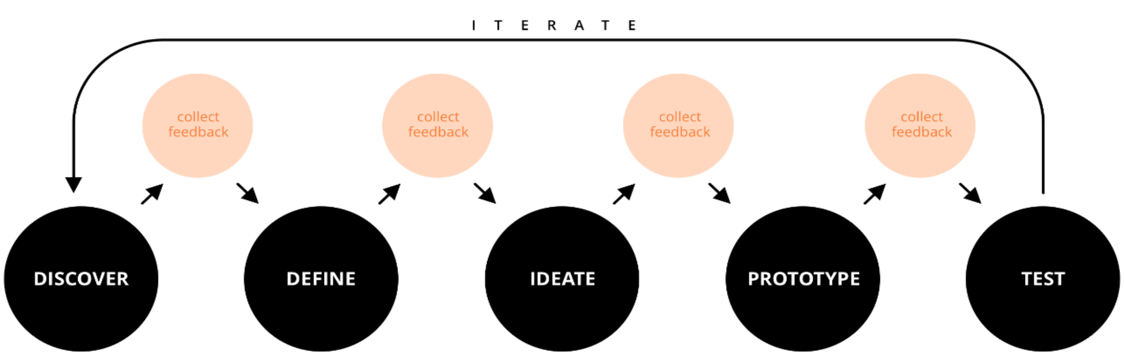

Fig 1 : Design Process

1. Discover:
   In the initial stage of the design process, we embarked on a thorough exploration and understanding of the requirements and challenges posed by the IGVC competition. This

involved researching and gathering information about the competition rules, objectives, and constraints. We also conducted a comprehensive analysis of previous successful designs to identify key factors for consideration.

2. Define:
   With a clear understanding of the competition requirements, we proceeded to define our specific design goals and criteria. This involved setting objectives for our bot in terms of performance, capabilities, and functionality. We also identified the constraints and limitations that needed to be taken into account during the design process.
3. Ideate:
   In this phase, we engaged in brainstorming sessions and collaborative discussions to generate a wide range of creative ideas and concepts. We explored different approaches and solutions to address the identified design goals and constraints. Various design alternatives were considered, and we evaluated their feasibility and potential impact on the overall performance of our bot.
4. Prototype:
   Based on the ideas generated during the ideation phase, we transitioned to the prototyping stage. Here, we translated the selected design concepts into physical or virtual prototypes. This involved creating 3D CAD models, conducting simulations, and fabricating physical components to bring our ideas to life. Prototyping allowed us to visualize and evaluate the feasibility and functionality of our design concepts.
5. Test:
   Once the prototypes were ready, we subjected them to comprehensive testing and evaluation. We conducted rigorous performance tests, simulated various scenarios, and assessed the bot's functionality and capabilities against the defined design goals. Testing helped us identify strengths, weaknesses, and areas for improvement in our design. We iteratively refined and modified the design based on the test results, incorporating necessary adjustments and optimizations.

By following this design process, we aimed to develop a bot that meets the specific requirements of the IGVC competition while showcasing innovation, reliability, and performance. Through continuous iteration, testing, and refinement, we strived to achieve an optimal design that maximizes our bot's capabilities and enhances its chances of success in the competition.

Innovations

A gist of the innovations in the various fields of SOCRATES 2.0 has been given below.

Mechanical

Based on the feedback, comparison with other bot's and problems we faced in IGVC 2022 with our bot, we have made several modifications to enhance the mechanical performance of our bot for 2023.

- Central Drivetrain Design instead of Front Drivetrain Design
- Brushless Hub Motors instead of Brushed DC Motors
- GPS Antenna Position and Mounting
- Compactness

Electrical

For our Electrical system, this year we wanted to focus on the following:

- Removing Non-Industrial components and replacing them with reliable ones. One such major component was Arduino.
- Adding safety features like fuses and circuit breakers for enhanced electronic protection.
- Heavy Duty Wiring and Connectors

Software

SOCRATES performed up to the mark for IGVC 2022 and we gained a lot of knowledge. For IGVC 2022 we wrote ROS based controllers and image processing pipelines. It was a naive approach in which the bot perceives the environment, decides and moves simultaneously without saving or using previous decisions. This can be considered as a flaw and needs to be fixed. We added the following to our software stack for IGVC 2023 to fix this issue:

- Accurate Odometry using Wheel Encoders, IMU, GPS
- Planning in Cartesian Coordinate points in real world based on lane, object detection and GPS Navigation
- Trajectory generation between current and target cartesian coordinate
- Replacement of PID/PD controllers with Adaptive Sliding Mode Controllers (ASMC)

Mechanical Design

Chassis

In 2023, the chassis design of SOCRATES 2.0 remains based on 3030 aluminum extrusion, providing modularity. However, the entire shape has been changed and it has undergone a reduction in size. The height of the ZED camera and GPS mount has been reduced to 3 feet. An enclosure box houses all the electronics and batteries and provides mounting points, weather protection and security to the electronic components. The GPS antenna has been mounted at the highest point possible to increase accuracy. The bot is supported by 2 castor wheels at the front and back and the drivetrain is positioned in the middle.

Fig 2, 3 & 4 : Bare Chassis, Complete Chassis, Wireless Electronic Kill Switch

Drivetrain Design

After observing our performance in IGVC 2022 with our bot SOCRATES, we decided to move the drivetrain to the center of the robot. This provides a number of advantages:

- Simplifies the Odometry: Standard odometry equations will hold true and no offset is required in the X,Y direction as the odometry point coincides with the center of the robot.
- Easier to avoid objects: The Lidar is positioned at the coordinate frame, so all objects will be perceived with respect to the center of the robot.
- On the Spot Rotation about Z axis: Since the coordinate frame of the bot lies in between the wheels, rotation about the Z axis while staying at a particular x,y point is possible

Our drivetrain consists of 10 inch brushless hub motors which increase the efficiency and create negligible noise while running. These have been mounted to 2 perpendicular aluminum extrusions with the help of 3d printed mounts.

Wireless E-Kill Switch

A new and compact enclosure for housing the e kill, PCB and battery was designed and 3d printed. This casing is made waterproof and houses the entire circuit needed for the e kill.

Electronic Design

Power and Kill Switch

SOCRATES 2.0 is powered by Lithium Polymer batteries, specifically, one 3S 10Ah battery and two 3S 8Ah batteries in series. We chose this configuration to separate the high current of the motors from the other low current devices. We carefully selected the battery capacity after considering the power consumption of all the onboard devices. Our 12v batteries have a capacity of 120Wh, which allows our bot to run for approximately 70 minutes on load. Meanwhile, our 24v battery has a capacity of 192Wh, providing ample power to our motors for extended periods of time. This will allow our bot to run for about 60 minutes on load.

Battery	Name	Power Consumption
12V 10Ah 120 Whr	Jetson Tx2 - Processor	60W
	Sparton AHRS-8P - IMU	5W
	ZED F9P - GPS	5W
	Velodyne VLP-16 - LIDAR	18W
	1. Lights 2. Relay 3. Zigbee	15W
	Total	103W
24V 8Ah 192 Whr	2 Hub Motors	200W
	Total	200W

We utilize a kill switch to effectively disconnect the power supply to our motor drivers and subsequently, the motors. Within the power lines of the motor drivers, we have incorporated a heavy-duty relay in series. The series connection is established using the normally closed (NC) and common contacts of the relay. This design minimizes power consumption since the relay coil is energized only when the kill switch is pressed.

For our wireless e-kill system, we leverage the signal provided by our XBee module to activate a secondary small relay. This small relay, in turn, triggers the main relay coil, enabling the disconnection of power.

Power Distribution and Regulation

In 2023, the power distribution system for SOCRATES 2.0 has undergone minor changes to improve efficiency and reliability. Similar to the 2022 design, two power distribution panels are utilized, one for 24V power and the other for 12V power. The 24V power distribution powers the two motor controllers and therefore the motors.

For the 12V power distribution, a buck-boost board stabilizes the voltage and provides consistent power to the various components. This buck-boost board regulates the voltage and prevents fluctuations, ensuring all components receive stable power. The regulated 12V power is then utilized to power the remaining components, including the Jetson, LIDAR, and a 5V step-down for the lights.

Motors and Motor Drivers

For IGVC 2022 with SOCRATES, we had chosen 200W brushed dc motors. We were not satisfied with their performance and hence switched to 10 inch brushless hub motors for IGVC 2023 SOCRATES 2.0. This had huge advantages to our system, it removed the need for wheels and couplers linking the wheels and motor shafts. The motor also has inbuilt hall effect encoders. The motor wires, both for power and encoder have been extended and terminated by using heavy duty 3 pin GX20 and 5 pin GX16 connectors.

For controlling the hub motors we chose the ODRIVE S1 motor controller. We have leveraged their inbuilt position, velocity and torque controllers by tuning their parameters. We have mainly used the velocity controller for driving. The ODRIVE's also feature regenerative braking hence during slopes or reducing the velocity, the reverse EMF is used to charge the batteries. Since we are using the velocity controller of the ODRIVE, it features position hold, so for example SOCRATES 2.0 stops on the ramp, the velocity controller will hold the position of the wheels at 0 RPM and the bot will remain at the position instead of sliding downwards.

Fig 5, 6 & 7 : Odrive S1, GX16 and GX20 Connector, 10 inch Hub Motor

Software Design

We have based our entire software on the ROS (Robot Operating System) and have divided every task into separate nodes. This makes fault detection easier and removes the interdependency of nodes. Our ROS Stack is depicted in the figure below.

It consists of 2 low level nodes namely: Motor_speed_controller and Odrive_controller. These nodes are of the most importance as they deal with the hardware directly.

There is 1 mid level node called Task queue and switcher.

There are 3 high level nodes namely: Lane detection and following, object detection and avoidance, GPS navigation and Joystick controller.

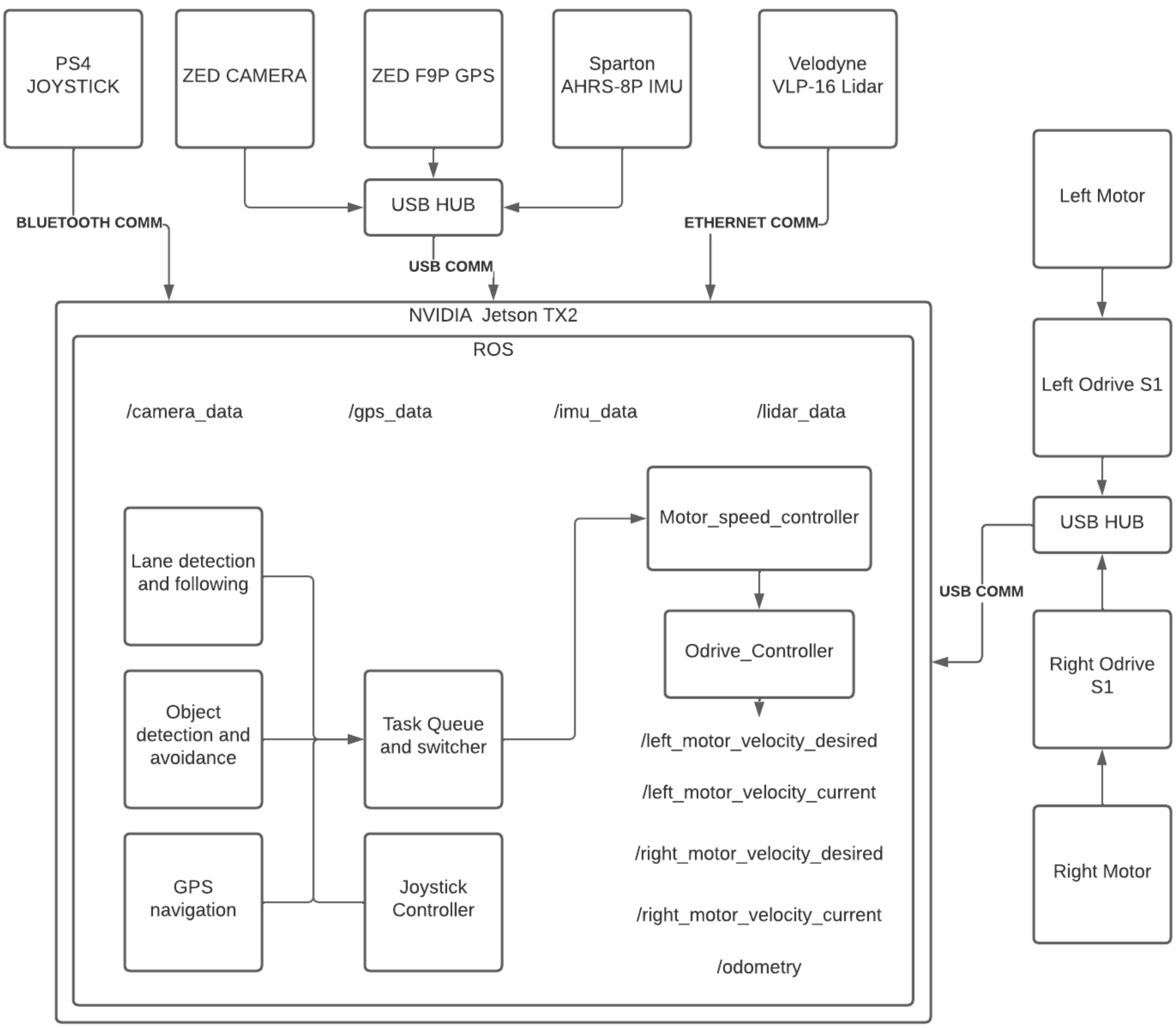

Fig 8 : ROS System Architecture

Odrive Control - Motor Velocity Control, Odometry and Fault Detection and Prevention

This node is at the lowest level in the system. The ODRIVE controllers have their communication tool based on python and hence to drive the motors, a value has to be assigned to the desired velocity variable in each ODRIVE. This node is majorly responsible for subscribing to 2 topics with velocity data for each motor and using that data to control the ODRIVE. While controlling the motors, the ODRIVE also provides the current position and velocity of the motor. This information is read and converted to odometry and published. ODRIVE also provides overcurrent, overvoltage, overtemperature, undervoltage and many other error detection signals. These are also handled by this node.

Motor Speed Control - Motor Velocity calculations, Simulation

This node is responsible for subscribing to a topic which contains the desired linear and angular speeds and calculating the speeds required for each wheel and then publishing that to the topic. This node also acts as a buffer and is used to control the motors in the simulation directly as the simulation bot subscribes to its topics directly.

Task Queue and Switcher

This is the central node of the entire system and binds the outputs of many nodes to the motor speed controller node. This node consists of a predefined list of tasks in a particular order such as lane detection, GPS navigation to point X etc. This list is created in accordance with the IGVC track. A counter keeps track of the current task to be executed. With the help of the counter the appropriate task is run and its output is used. The object detection and avoidance task is always running as it is required by both the lane detection and GPS navigation tasks. This node is also responsible for switching between autonomous mode and joystick/manual mode.

Lane detection and Following

Our lane detection algorithm has undergone small changes but still relies on OPENCV transformations and operations to detect the edge points of the lanes. A combination of HSV filtering, splitting of image, thresholding and histogram generation is used to detect the lanes and the steering angle is derived from that data. This angle can directly be used to control the bot but for IGVC 2023, we convert this angle to a real world cartesian point and plan a trajectory towards that point. The trajectory is basically a set of discrete real world cartesian points that connect the target and current points. The target point is generated with the help of a scaling factor for both the axes which is calculated by averaging manual measurements. All the points in the trajectory will always lie inside the lanes.

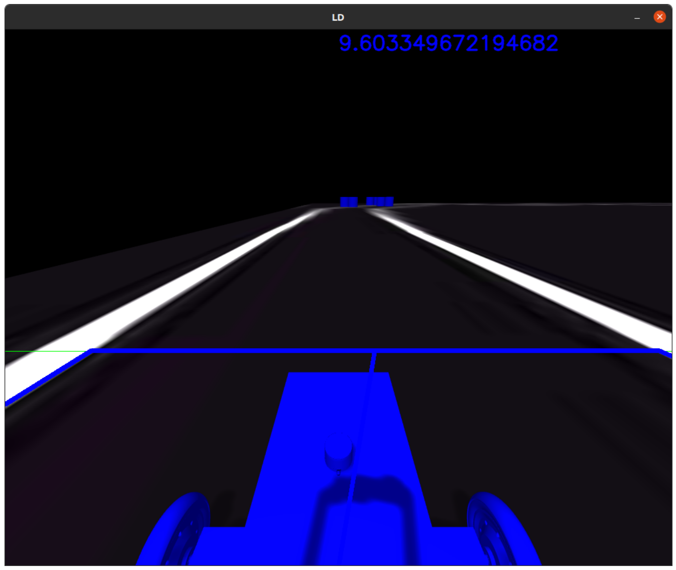

Fig 9 : Lane Detection and Trajectory Calculation

Object detection and avoidance

For detecting objects in front of SOCRATES 2.0 we are using the Velodyne LiDAR. The output of the lidar is the form of a list of distances at each angle from 0 to 359 degrees. We then classify this list into 2 lists that contain angles and points that are objects and that are not objects. After classification, any angle from the not object angle list can be chosen and it will cause the robot to avoid the object. An improvement in this algorithm is that when choosing the object avoidance angle, we can choose it in such a way that the angle chosen is closest to the command angle from the joystick/lane detection/GPS navigation pipelines. This allows the robot to avoid objects as well as follow the command of the other pipelines

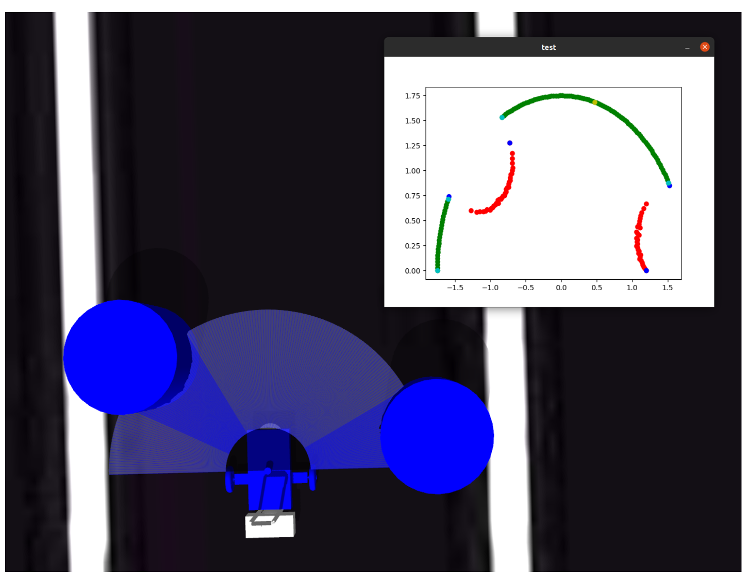

Fig 10 : Object Detection and Avoidance Angle and Trajectory Calculation

GPS navigation

Our GPS navigation algorithm remains unchanged from last time. The destination GPS point is virtually plotted and the angle between it and the current point is calculated with the help of simple trigonometry and triangulation and with respect to the current position. This angle is then simply sent to the motor speed controller node via a topic which drives in the direction of the point. In some time the current point will be within the threshold range of the destination point.

Failure Modes

We have identified some failure modes in SOCRATES 2.0 and have listed them below:

3D Printed Hub Motor Mounts

During high-radius turns, the 3D printed hub motor mount was prone to breaking due to frictional forces. To address this issue, we implemented a solution by reinforcing the printing process using fiberglass. This reinforcement provided increased strength and durability to withstand the forces experienced during turns, reducing the likelihood of mount failure.

IMU

The IMU experienced magnetic interference from the relay, causing inaccurate readings and misalignment of the compass with the true north. To mitigate this issue, we employed a magnetic shielding solution. By implementing appropriate shielding measures, we effectively reduced the magnetic interference, ensuring accurate compass readings and reliable orientation data from the IMU.

Wiring

To prevent wiring-related failures, we implemented a solution by terminating the wires with heavy duty connectors for important and sensitive components. This approach improved the overall reliability of the electrical connections, making it easier to troubleshoot and replace components when necessary. The use of connectors facilitated efficient and secure connections, reducing the likelihood of loose or faulty wiring that could lead to failures.

Simulations

Since we are using ROS, the Gazebo simulation and RVIZ tool were the best software that we could use for simulations and testing. We modeled the entire robot in URDF and also the entire IGVC track in Gazebo. No changes in the code were necessary when we switched between simulation and real life testing as our package was designed in a modular way. Various iterations of the lane curves, object placement, starting poses and GPS coordinates were tested.

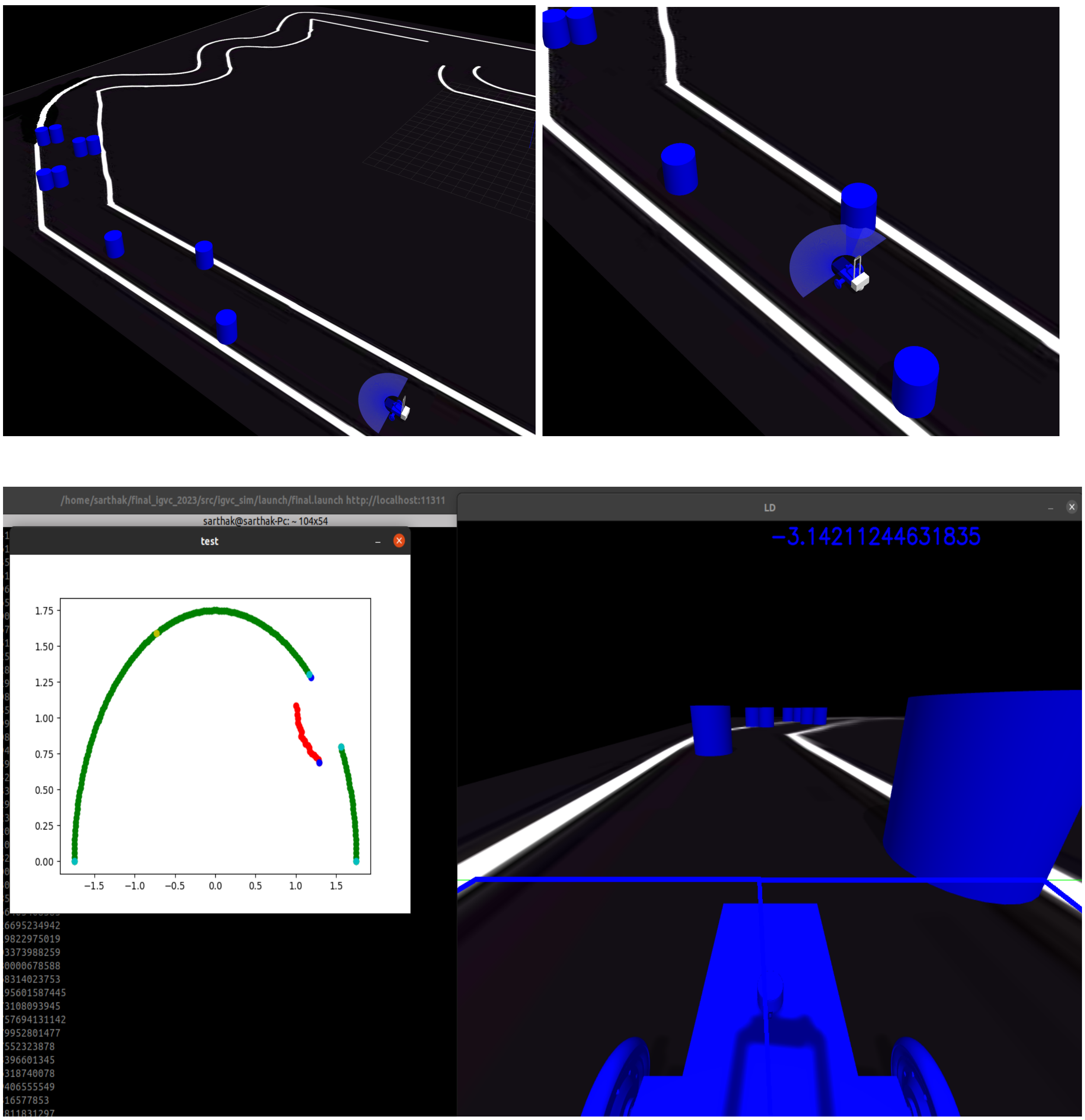

Fig 11 : Simulation in Gazebo

Performance Testing

To evaluate the performance of SOCRATES 2.0, our team conducted comprehensive performance testing across various aspects of the robot's functionality. These tests aimed to assess the capabilities, efficiency, and reliability of the system, enabling us to identify strengths and areas for improvement. The performance testing covered the following key areas:

1. Navigation and Autonomous Movement: We tested SOCRATES 2.0's ability to navigate autonomously through tight lanes and obstacles. The robot's navigation algorithms were evaluated for accuracy, speed, and responsiveness in different scenarios, including straight paths, curves, and complex maneuvers.
2. Object Detection and Avoidance: Performance tests were conducted to assess the effectiveness of the object detection and avoidance system. We evaluated the robot's ability to detect and appropriately respond to obstacles in its path, ensuring safe and reliable operation.
3. Communication and Connectivity: Tests were conducted to evaluate the communication and connectivity between different system components, including sensors, processors, and control modules. We assessed the reliability and latency of data transmission, ensuring seamless integration and coordination within the system.

Initial Performance Testing

Based on the initial performance assessment, SOCRATES 2.0 demonstrated promising capabilities and performance across several key areas. The robot exhibited reliable autonomous navigation, successfully maneuvering through predefined routes and effectively avoiding obstacles in its path. The object detection and avoidance system showed satisfactory performance, detecting and responding to obstacles with accuracy and efficiency. The robot successfully traversed the test track while maintaining an average speed greater than 1.5 mph (2.4 km/h). However, during the initial assessment, certain areas for improvement were identified. These included fine-tuning the navigation algorithms for more optimized path planning and refining the object detection system to enhance its robustness in complex environments. Overall, the initial performance assessment provided valuable insights into SOCRATES 2.0's capabilities and identified areas where further development and optimization are needed. The team remains committed to addressing these areas and continuing to enhance the performance and capabilities of the robot.