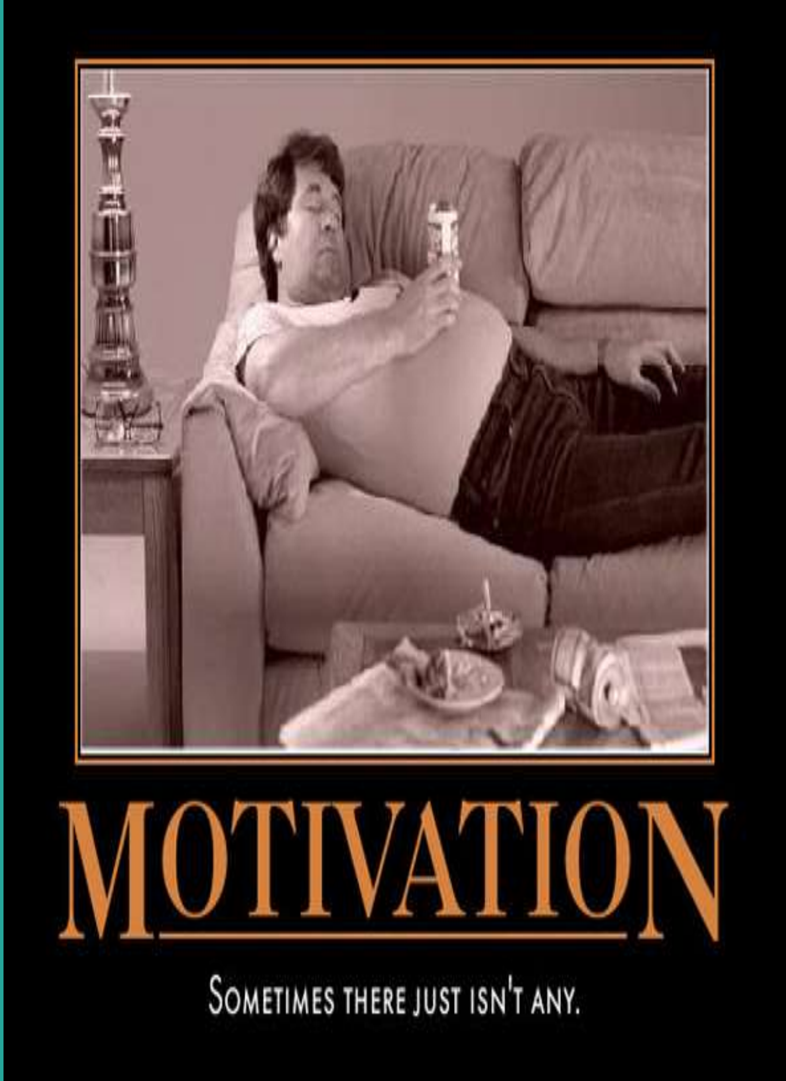
INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

IIIT Hyderabad

Obstacle Avoidance for Ground Robot

Team - Greedy_Video (Group 9)
Members - Aanvik , Divyansh , Hemang , Mohak

Motivation



JUST KIDDING!

This idea was unanimously decided by the team based on collective interests to work on the concept of obstacle avoidance and robotics. It was shortlisted from 5 selected topics of particular interests , being the first preference

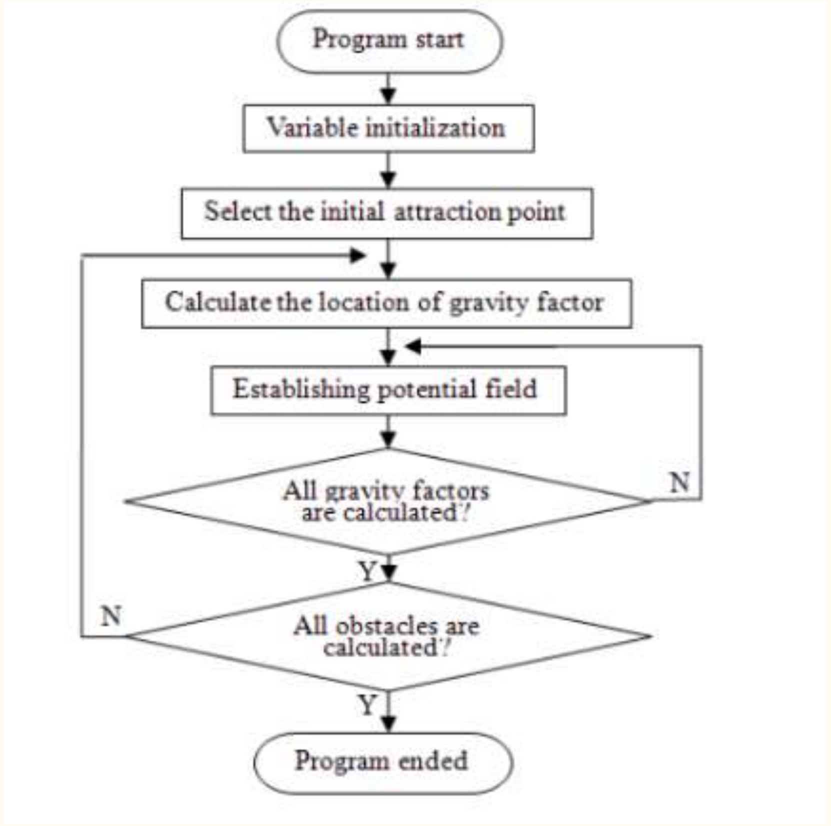
Implementation Approach

1. We would be constructing a ground robot (a car preferably) in which we would implement the functionality of obstacle avoidance and path planning.

2. After a careful design of the robot and hardware , we would use the sensors on the robot to feed it real - time data of the obstacles around it.

3. We would be using the potential field method to implement obstacle avoidance

4. Potential field method - The artificial potential field (APF) method is widely used for autonomous mobile robot path planning due to its efficient mathematical analysis and simplicity .The application of this method, however, is often associated with the local minima problem which occur when the total force acting on a robot is summed up to zero although the robot has not reached its goal position yet


Program start
Variable initialization
Select the initial attraction point
Calculate the location of gravity factor
Establishing potential field
All gravity factors are calculated?
N
Y
All obstacles are calculated?
N
Y
Program ended

Components required

- Arduino UNO.(x1)
- ESP32(x1)
- L-Shaped 60 RPM BO Motor with 65X25 Wheel(x4)
- L2N3D/L298N Motor Driver(x2)
- HC - SR04 Ultrasonic Sensors(x8)
- IR sensors(x4)
- JHD162A LED Screen(x1).
- SG90 Servo Motor(x1).
- HC05 Bluetooth Module.(x1)
- LEDs and Jumper Wires!

Deliverables

Deliverables

The main functionality of the robot would cover:

- The robot will be able to avoid obstacles and will be capable of planning path to the desired point through potential field algorithm

We plan to implement the following additional functionalities in the robot

- Radar sensor which would show obstacles on the UI.
- Additional remote control of the robot from the UI by switching mode to remote control mode.
- Edge avoidance(to avoid falling down of an edge of the table).

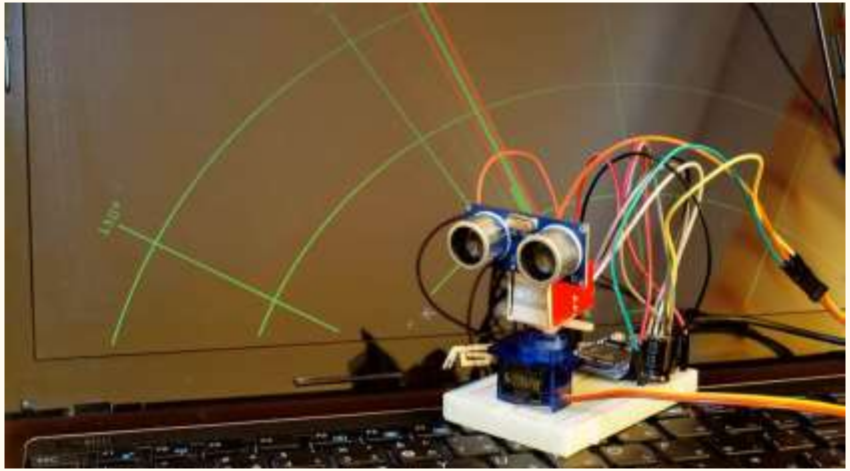
Functionalities

Radar unit:

We plan to implement a radar system in the vehicle by attaching an Ultrasonic sensor on top of a servo motor which sweeps an angle facing the front of the car and detects obstacle and plots them on a radar system implemented in our computers.

The ultrasonic will continuously keep sweeping an angle of around 150 degrees and detect obstacles in the current facing direction. Using the current angle and distance we will plot objects on a graph on the computers.

Code link - Code



Functionalities

Bluetooth module:

We plan to implement a Bluetooth controlled remote control system in the car to increase its utility . We will use a Bluetooth module to send signals to the car from a remote control UI in the mobile phone.

The car will switch modes between obstacle avoidance and remote control based on flags set up in the software. This part of the deliverables is highly ambitious and it's implementation is subjected to time constraints.

Wifi and ThingSpeak Integration:

Since we are using ESP32s , we plan to constantly send data to thingspeak. This data would include the speeds of motors and the sensor outputs of each sensor attached to the car.

This data can be used for data analytics and monitoring the condition of the car.

Functionalities

Edge Avoider :

Alongwith implementing an obstacle avoidance we are planning to implement an edge avoider which will consist of infrared sensors mounted on an extended part of the robot body at front.

It will detect the distance of the robot from the ground. After suitable calibration of the sensor module and testing multiple cases, the threshold will be set after which the robot will stop and rethink the route to be taken.

Obstacle Avoidance:

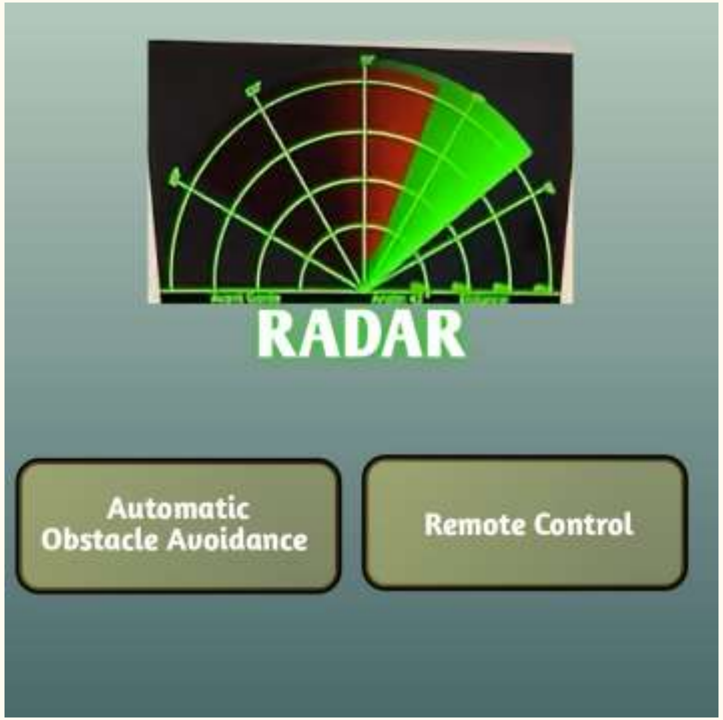
The fundamental and the principal part of our project is Obstacle Avoidance. We will be using ultrasonic sensors in front and in the back ,which when the distance of obstacle from the sensor is below a threshold, it overrides control over the motors and pulls the robot back and then looks for any other direction without an Obstacle and rotates and continues moving in that direction.

Functionalities

User Interface (UI)

The UI will consist of several features . Firstly, It will display the Radar readings as discussed in the Radar functionality earlier. Secondly , It will have buttons to Switch between Modes , such as Automatic Obstacle Avoidance or Remote control.

The Automatic Obstacle Avoidance, as it suggests by the name , it keeps moving the robot forward and changes direction when an Obstacle comes in its path or it reaches at an edge. The latter gives the movement control to the user , which can be used to move in any direction, but it gets overridden by Obstacle and Edge Avoidance Mechanisms.



Code base

The entire codebase for the project will be managed on a private github repository.

Owner of repository - divyanash911

Contributors - hemang-n00b , MohakSomani , AanvikBh

Task Timeline

We split our project into 5 phases:

- **Planning** - We plan the functionalities and objectives of the robot , the materials required and deciding on relevant software to be used.
- **System Design** - We design the robot's hardware and software(using simulations).
- **Implementation** - We start constructing the robot based on our system design (first the hardware then the software).
- **Integration and testing** - After successfully implementing the idea , we test our robot in an actual obstacle based path and gather failures to modify our system accordingly (failure analysis)
- **Data analytics and UI** - We will attempt to map the obstacles and reconstruct the path in our computers based on the path taken by the robot (as per the artificial potential field method) and create an interactive UI to demonstrate the same.

Timeline

- **Planning** : 1 Week.
- **System Design** : 2 Weeks.
- **Implementation** : 4 - 6 Weeks.
- **Integration and Testing** : 1-2 Weeks
- **Data analysis and UI** : 1 Week

Team Memberwise Proposed Plan

- **Aanvik Bhatnagar(2022101103)** - Software(Integrating microcontrollers) and testing.
- **Divyansh Pandey(2022101111)** - Hardware design , implementation of the robot.
- **Hemang Jain(2022101086)** - Sensor implementation in the robot , data analytics.
- **Mohak Somani(2022101088)** - Motors implementation, design in the robot and UI.

References

- https://www.researchgate.net/publication/224173055_A_novel_potential_field_method_for_obstacle_avoidance_and_path_planning_of_mobile_robot
- https://www.researchgate.net/publication/313389747_Potential_field_methods_and_their_inherent_approaches_for_path_planning
- https://journals.sagepub.com/doi/pdf/10.1177/1729881418799562
- https://www.cs.cmu.edu/~motionplanning/papers/sbp_papers/integrated1/borenstein_potential_field_limitations.pdf

Phase 1: Planning (12th Sept - 19th Sept)

- Initial part of the planning phase was looking into the variations we can add to the basic agenda of the project- obstacle avoidance, and we got to know that there is a huge scope underlying in the project, out of which we have picked up the points mentioned in the previous slides.
- Taking into consideration the work distribution, and understanding the fact that many portions depended on completing the basic hardware layout, we decided to focus the work distribution on the hardware implementation initially.
- Alongside, we worked around the basic codes for all the sensors, to get to know about the functioning that sensor's library is able to provide.
- One of the essential parts of the planning phase included hardware collection, which allowed us to view the scope of the project, and what all things we need to change in our initially proposed model.

Phase 1: Planning (12th Sept-19th Sept)

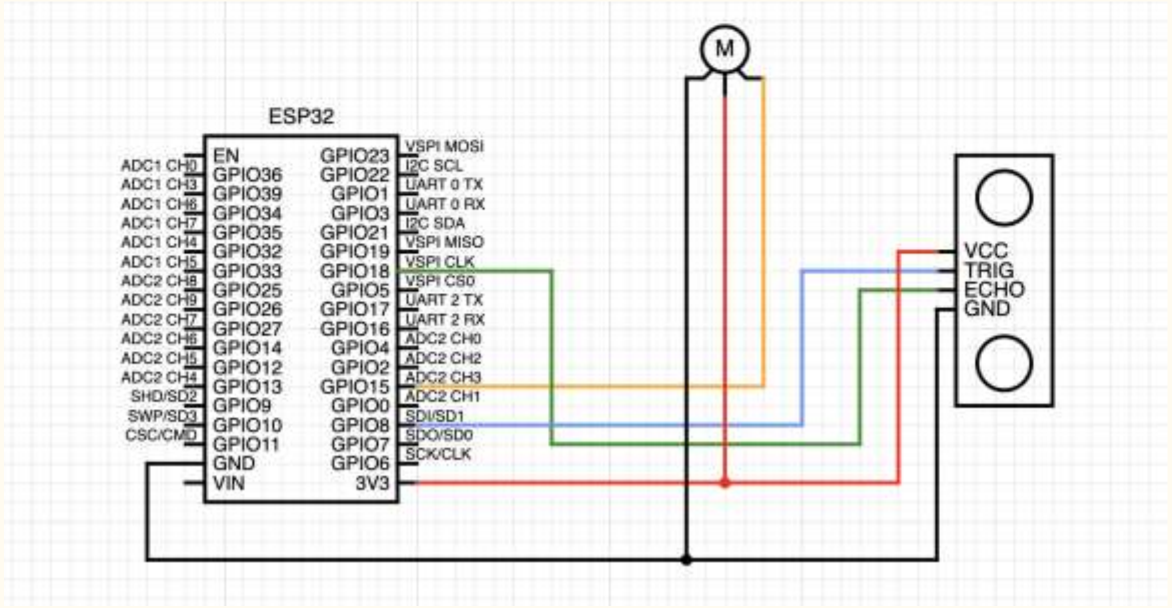
- We also had a presentation under the professor on 14th Sept, explaining whatever we plan to do and what resources are available to us.
- As soon as we got our hardware, we shifted to system design phase.

Phase 2: System Design (19th Sept-26th Sept)

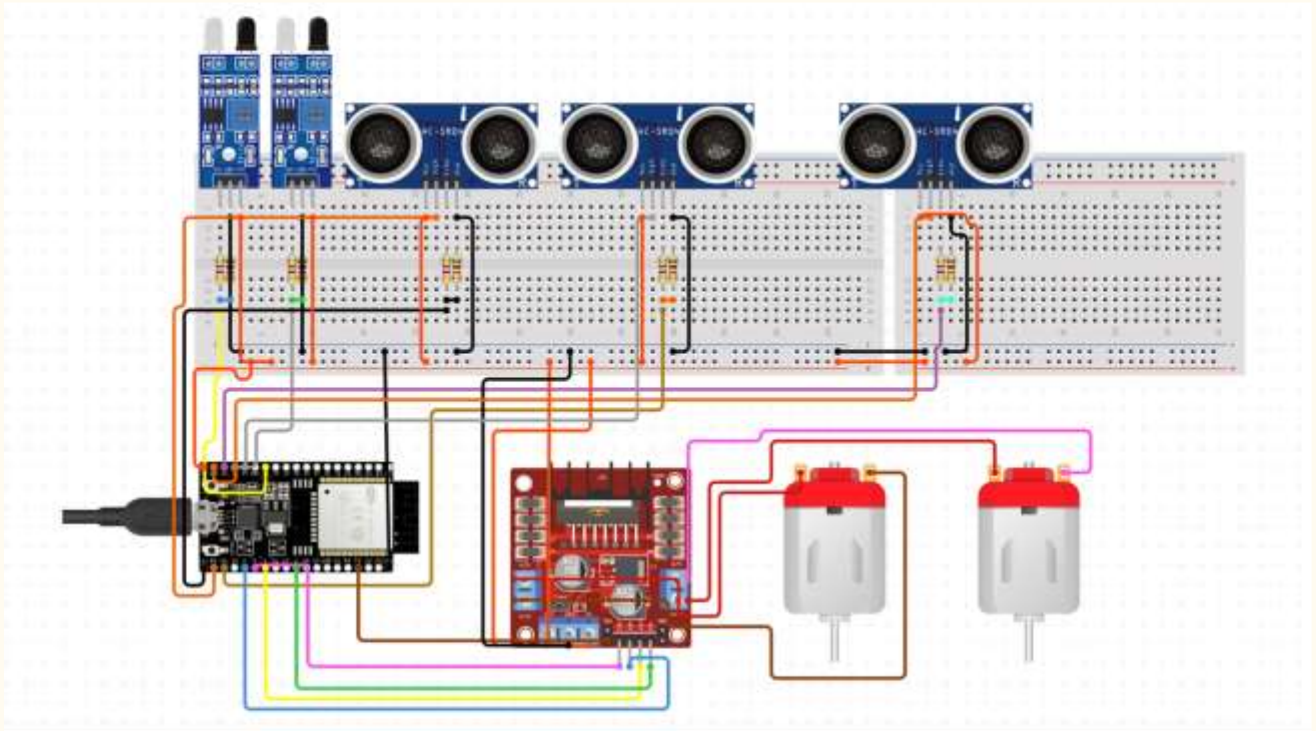
- Before the specific implementation of the sensors, we needed to ensure that the sensors were in a working condition, i.e. they were able to provide the bare minimum results as expected from them. So we moved into hardware testing phase.
- Divyansh and Mohak worked on the working of motors and motor driver. One of the motors is working, but the speed manipulation needs to be done, and the soldering needs to be redone.
- Hemang and Aanvik worked on the working of display screen and the servo motor on which the screen will be mounted. We need to mount the display screen on the servo motor.
- All of the testing codes are in the git repository mentioned above.

Phase 2: System Design (19th Sept-26th Sept)

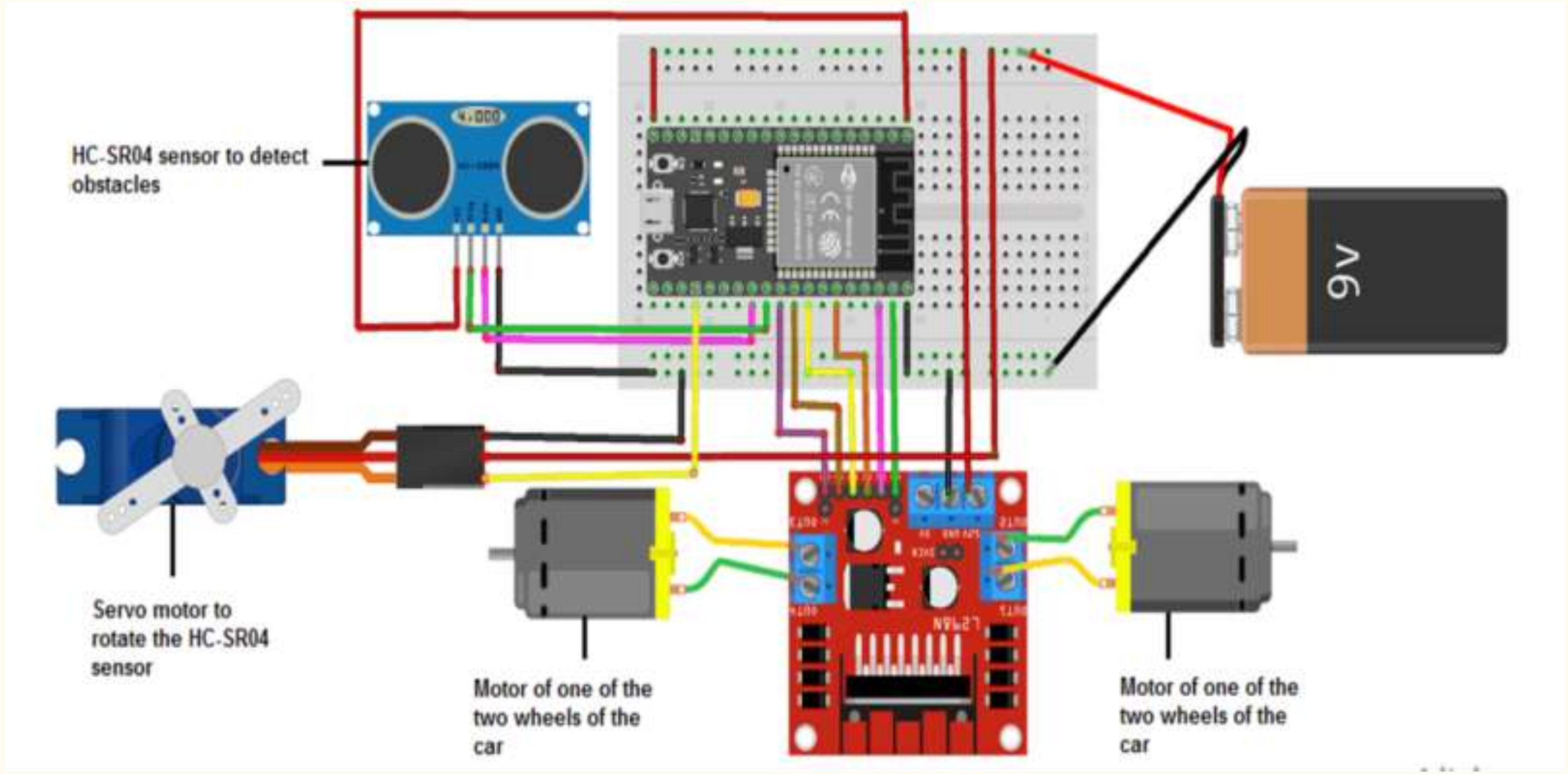
- The circuit diagrams are displayed below.
  a) For Radar Unit:



b) For Motor Driver and Sensor Unit:



The combined view would look somewhat like this(some implementation points are different):


HC-SR04 sensor to detect obstacles
Servo motor to rotate the HC-SR04 sensor
Motor of one of the two wheels of the car
Motor of one of the two wheels of the car
9V

(source: https://www.robotique.tech/robotics/obstacle-avoiding-robot-using-esp32/)

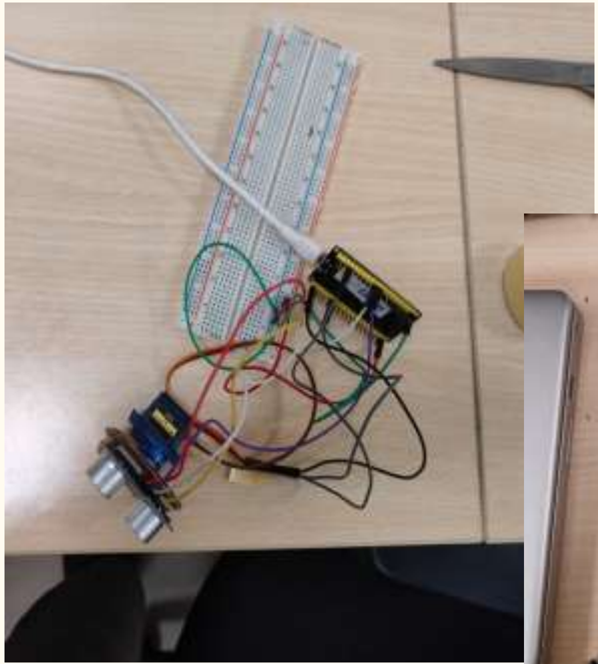
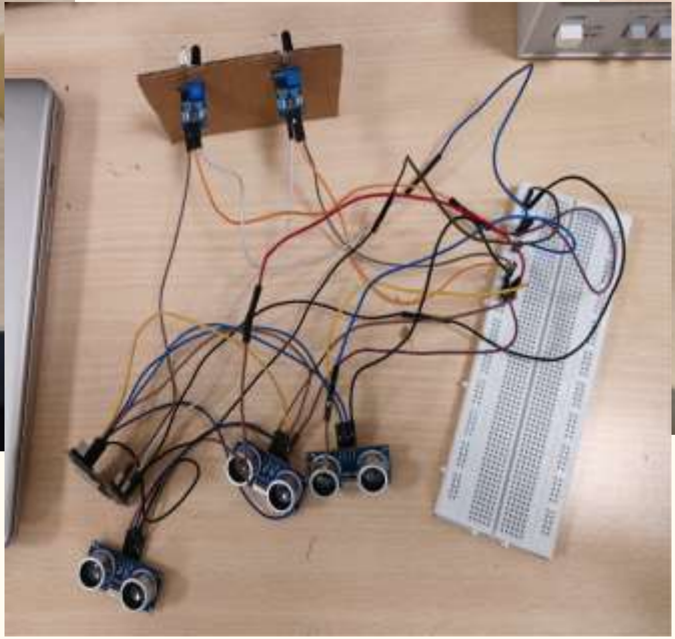
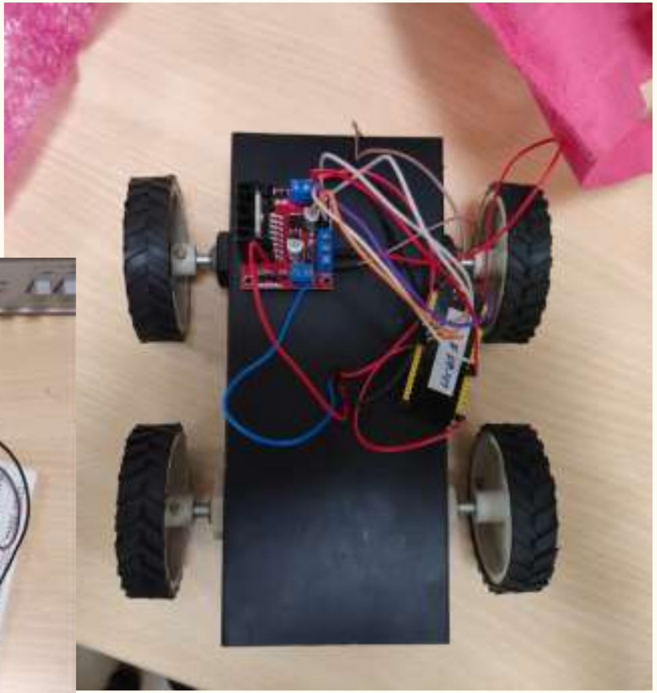
Phase 2: System Design (26th Sept-3rd Oct)

- Now that we had tested all of our components for their basic implementation, we shifted to the required implementation of sensors.
- Divyansh worked majorly on the calibration of motor driver and the motor , along with developing the logic and writing the code for running the ground robot. Also he figured out, how to implement the direction change where both the motors were made to run in opposite directions.
- Mohak designed the circuit for IR sensor for edge avoidance and Ultrasonic sensor for obstacle detection. He also developed the logic and wrote the code after integrating the two which returned a flag whether there is an obstacle or not.Also helped Divyansh with the motor driver part.

Phase 2: System Design (26th Sept-3rd Oct)

- Aanvik started the Thingspeak implementation, and a rough remote control UI, which sends the data on Thingspeak channel. This data sent will act as a flag for switching modes and allowing restrained movements as per the user. Currently the data sent to the channel includes some buttons from the website which when clicked, should result in a trigger action on hardware (requires integrating all the hardware, hence to be implemented towards the implementation phase).
- Hemang worked on the major implementation of the Radar functionality . He firstly started by doing the hardware part , mounting the ultrasonic sensor on the servo motor . Then he wrote the code to rotate the servo degree by degree and take data from ultrasonic sensor , printing it to COM3 Serial Port . Then he wrote code for Processing , which takes data from COM3 Serial Port and plots the Live Radar .

Current Progress:

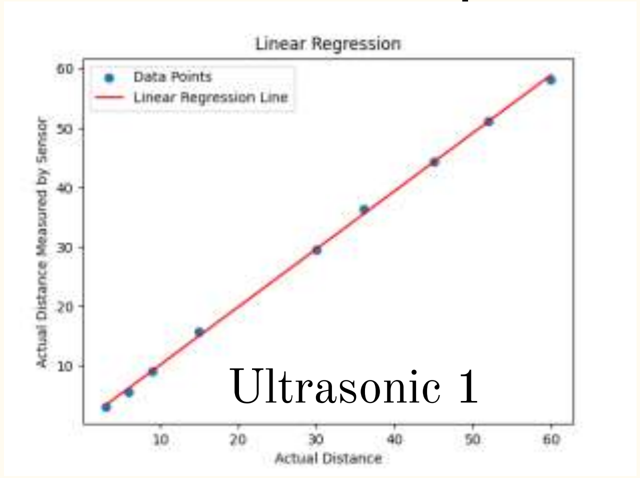
Plan for the upcoming weeks

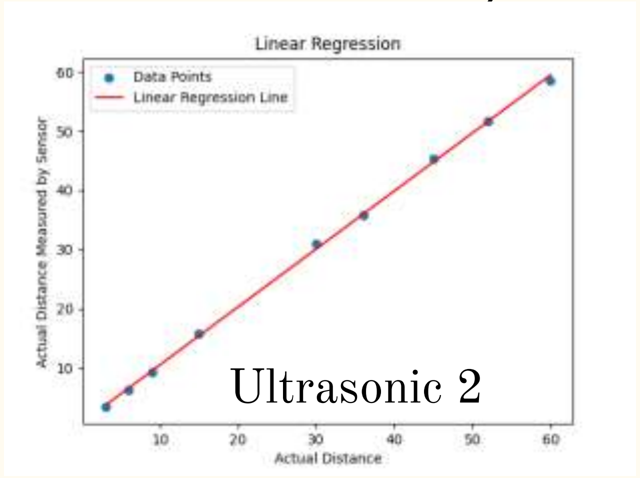
- Integration of all the individually working sensor bodies onto the car frame, and after procuring the battery for power supply of motor driver, testing the speed maneuvering .
- Shifting our operations to OM2M, development of UI, and processing data obtained from all the sensors. Thingspeak is only used for hardware testing with the UI, but we will shift our operations to OM2M to enhance interoperability and reduce latency.
- Working on mode-switching. Code out the conditions for triggering the switch, and ensuring that whenever the necessary conditions are fulfilled, minimum latency in the context switch is ensured.

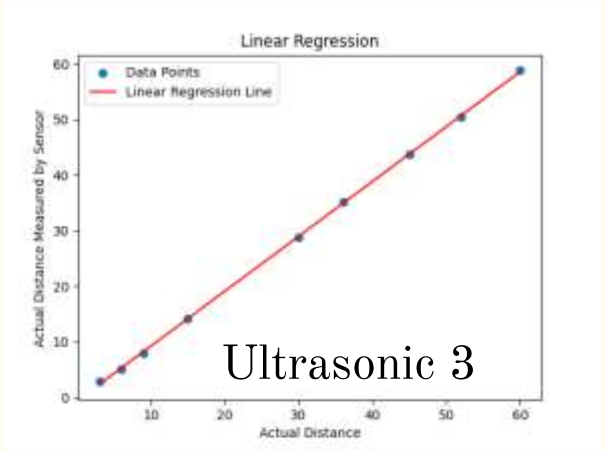
Phase 3: Implementation (3rd Oct - 10th Oct)

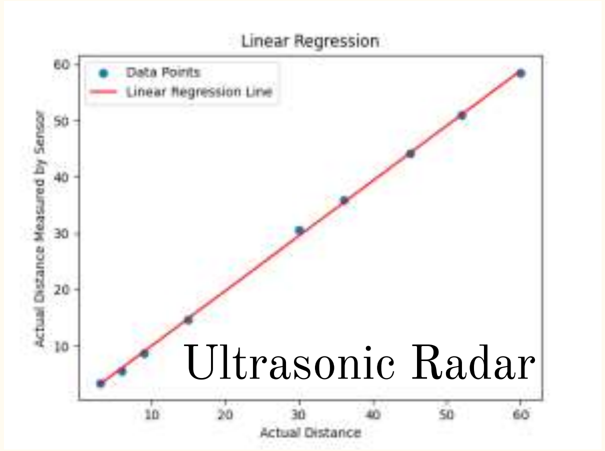
We were told in the mid evals to have a plot ready for checking whether the values of ultrasonic sensors are accurate or not, and how much error do we get, so we first wrote a python code which uses data points gathered from 4 different ultrasonic sensors used by our car, and uses linear regression to check their accuracy. Below are images from 4 different ultrasonic sensors.

Phase 3: Implementation (3rd Oct - 10th Oct)









Phase 3: Implementation (3rd Oct - 10th Oct)

- Now that all of our individual components are working properly, and we are ready to shift our collected data to OM2M, we move into the implementation phase, where we have first started with integrating the motor drivers and ultrasonic sensors.
- Hemang and Divyansh worked on arranging the circuit on the car frame, which included placing the ultrasonic sensors in the positions desired for obstacle detection(front, right and left at an angle).
- Aanvik and Mohak worked on attaching the IR sensors for edge avoidance and also tested radar for mapping again, and calibrated the thresholds a bit.
- Since we needed to focus more on hardware integration as a team, we decided to keep OM2M and data collection after the integration and first testing is done.

Phase 3: Implementation (10th Oct - 17th Oct)

- This part of implementation was mainly focussed on debugging the obstacle avoidance code, and edge avoidance code. We primarily wanted to set thresholds for ultrasonic sensors to avoid obstacle, while also altering the speed of the wheels, and checking for edge below.
- Divyansh and Aanvik worked on adjusting the distance values which ultrasonic sensor can detect properly, and decided to keep threshold as 10 cm, at least for the first testing once implementation will be completed.
- Hemang and Mohak worked on circuit testing, and making sure that no wires connected to the circuit were damaged/had gotten damaged over the course of testing. They were working on hardware checkup.

Phase 3: Implementation (17th Oct - 24th Oct)

- By now we were able to attach the motor driver, breadboard, ESP32, and IR sensor over the car frame, and our initial prototype was ready. All we had to do now was to ensure that code and hardware were in coherence, and no part of hardware got damaged while testing.
- Hemang, Divyansh and Mohak were handling the circuits, ensuring that whenever we did not get a compilation and upload error on Arduino, and car wasn’t moving, then hardware connections were intact, and motor driver was getting enough voltage.
- Aanvik started to look into the code for remote control change, and OM2M Resource Tree set up in the meantime, so that once after this week, we know that hardware was working, we will shift to UI integration and Switch.

Phase 4: Integration and Testing (24th Oct - 31st Oct)

- We were running a bit ahead from the schedule we originally had in mind for Implementation, so we shifted to Integration and Testing. Since the motor driver needs a high power DC current supply, we had shifted our operations to Nilgiri 117 during implementation phase.
- Hemang and Mohak worked on testing the car with obstacles now, and ensuring the connections made to the car remained intact.
- Divyansh and Aanvik worked on developing the OM2M Resource Tree, and data from Ultrasonic sensor mounted on Radar to be sent to the endpoint. They also worked upon trying to send HTTP requests from website to OM2M, but it was showing some complications in that regards.

Phase 4: Integration and Testing (31st Oct- 6th Nov)

- We were mostly done with integrated hardware testing, and also had readjusted some thresholds and components now, so we decreased the frequency of testing, since it would put stress on the wheels, and then after some point of time, they would not rotate properly, and needed to be adjusted. Hemang and Mohak were looking into it.
- They also worked on mounting the radar circuit with the car frame, and fixing it so that with the sudden speed, it does not fall.
- Meanwhile Divyansh and Aanvik focussed on implementing our website with OM2M, but due to latency issues and data formatting issues on OM2M, we decided to shift our remote control model via bluetooth in ESP32.

Phase 5: Data Analysis and UI (6th Nov-13th Nov)

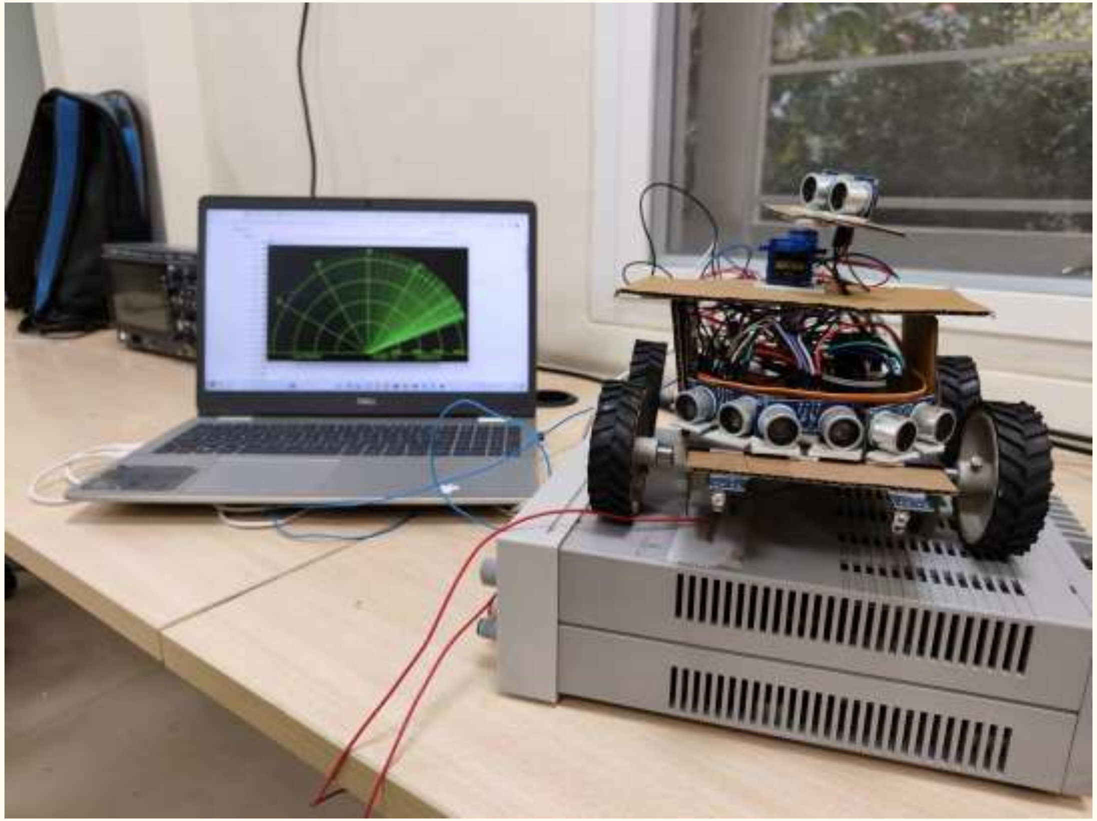
- Now that we had to work with Bluetooth, Aanvik and Mohak worked on sending commands via Bluetooth using a mobile app “Arduino Bluetooth Controller”. Here there is a in-built button option, which when sent to ESP32 via bluetooth, will perform the desired action. They also worked on the code implementation of remote control car.
- Divyansh and Hemang worked on collecting data from the radar, which was now mounted on top of the car, and sending the data to OM2M for analytics. But due to latency issues faced while implementing OM2M with radar, we decided to keep it as a secondary approach, and not include it in our final project.

Phase 5: Data Analysis and UI (13th Nov-20th Nov)

Now we are in the final phase of our prototype, and with the deliverables fulfilled, we have started the final testing and data collection with bluetooth and radar.
The whole team is involved in the final testing, each member handling some deliverable aspect: Hemang made sure that hardware was not getting damaged while testing, and it remained intact, Divyansh made sure that obstacle was getting properly avoided with the logic of the code motor driver was working on, Mohak made sure that radar was detecting obstacles at the same time, as the car would, and Aanvik ensured that remote control switch was working with the entire code.

A working video was also sent to the TA for their reviews.

We present to you… The final model



Thank You !!