Self-Driving Robot with GPS Navigation

Final Year Project Report

Presented by

Rehmat Khan
CIIT/FA19-EEE-006/ISB

Taha Aftab
CIIT/FA19-EEE-008/ISB

In Partial Fulfillment

of the Requirement for the Degree of

Bachelor of Science in Electrical (Electronics) Engineering

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

COMSATS UNIVERSITY ISLAMABAD

July 2023

Declaration

We hereby declare that this project neither as a whole nor as a part of has been copied from any source. It is further declared that we have developed this project and the accompanying report entirely based on our personal efforts made under the sincere guidance of our supervisor. No portion of the work presented in this report has been submitted in support of any other degree or qualification of this or any other University or Institute of learning, if found we should stand responsible.

Rehmat Khan

Signature:

_____

Taha Aftab

Signature:

_____

Self-Driving Robot with GPS Navigation

An undergraduate final year project report submitted to the Department of electrical and computer engineering

As a Partial Fulfillment for the award of Degree
Bachelor of Science in Electrical (Electronics) Engineering

by

Name	Registration Number
Rehmat Khan	CIIT/FA19-EEE-006/ISB
Taha Aftab	CIIT/FA19-EEE-008/ISB

Supervised by

Dr. Safwan Khalid
Assistant Professor

Co-Supervisor
Dr. Omer Ahmed
Assistant Professor

Department of Electrical and Computer Engineering

COMSATS UNIVERSITY ISLAMABAD

July 2023

Final Approval

Self-Driving Robot with GPS Navigation

Submitted for the Degree of

Bachelor of Science in Electrical (Electronics) Engineering

By

Name	**Registration Number**
Rehmat Khan	CIIT/FA19-EEE-006/ISB
Taha Aftab	CIIT/FA19-EEE-008/ISB

Has been approved for

COMSATS UNIVERSITY ISLAMABAD

Supervisor
Name: Dr. Safwan Khalid
Designation: Assistant Professor

_____ _____

Internal Examiner-1
Name,
Designation

Internal Examiner-2
Name,
Designation

_____

External Examiner
Name,
Designation

_____

Head
Department of Electrical and Computer Engineering

Dedication

Thanks to Allah Almighty, we proudly dedicate this project to our loved ones and friends; without their support and love, this would not have been possible. Most importantly, our parents and friends who supported us both morally and financially, if it were not for them our goals would be unattainable.

Acknowledgements

Firstly, we would like to express our gratitude to Allah Almighty, our creator, for granting us the ability and opportunity to complete our BS-EEE degree at COMSATS University. The list of individuals who have assisted and supported us during the project is extensive, and we would like to thank all of them for their contributions.

We would like to begin by acknowledging our honorable and esteemed supervisor, Dr. Safwan Khalid, and co-Supervisor, Dr. Omer Ahmed who has been with us since the project's inception, guiding us through every step and challenge. He identified our weaknesses, mentored us, and went out of his way to assist us in any way possible. Without his guidance, we would have been lost.

Our parents have been a constant source of support and guidance throughout the project, offering us their valuable advice and encouragement in challenging times. They have paved the way for our success, and we are grateful to them.

Additionally, we acknowledge the Electrical and Computer Engineering Department for providing us with the necessary resources to complete our project. Finally, we would like to express our appreciation to COMSATS University for granting us the opportunity to pursue our studies and receive the education required to complete this project successfully.

Rehmat Khan
Taha Aftab

Table of Contents

List of Acronyms

HSV……...… Hue Saturation Value

RGB ……..…. Red, Green Blue

ML ……..……...…..... Machine Learning

OpenCV ……..... Open-Source Computer Vision Library

GPS ……..... Global Positioning System

PID ……..….. Proportional Integral Derivative

CNN ……..…… Convolutional Neural Networks

YOLO……..……You Only Look Once

List of Figures

List of Tables

Abstract

Vehicles used for repetitive tasks like shuttle services at parks or road sweeper trucks with a fixed path to follow do not necessarily need human supervision. Such use-cases, potentially, can be fully automated. We aim to design an autonomous vehicle which is self-sufficient enough to cover a certain distance. The only input it would get would be the destination location through GPS. It will follow the given trajectory and use computer vision techniques to stay on its specified track as well as look out for accidental hazards. It will be an autonomous robot that conducts its tasks without human supervision. The primary purpose of this project is to make the robot cost-effective while allowing for maximum features. We aim to design a flexible system which can be employed in diverse fields and multiple use cases. Our motivation is to increase automation in Pakistan thus reducing the import bill while increasing efficiency and productivity of the industrial sector.

Chapter 1: Introduction

A self-driving robot is a cutting-edge technology which has experienced an exponential growth in the recent years. they are designed to operate autonomously without any human intervention. These robots are equipped with multiple sensors, cameras at different angles, and hardware processors that allow them to collect data about their environment, pass it through the processor, and make informed decisions on were to mauver. The goal of self-driving robots is to minimize human error and do repetitive and boring tasks so that humans don't have to do them.

Self-driving robots use advanced machine learning algorithms and computer vision techniques to extract data from their sensors and cameras. This enables them to understand the path, identify obstacles and potential risks, and detect other vehicles and objects on the road. A key feature of self-driving robots is their ability to make real-time decisions based on their environment, allowing them to alter their speed, switch among lanes, and avoid obstacles while following a specific destination.

While self-driving robots have a lot to promise, they still face various technical and regulatory challenges that engineers are still working for, before they can be a command commercial product. Nonetheless, there has been a significant progress made in recent years, and many researchers believe that AI-based self-driving robots will play a significant role in shaping the future of vehicles in all forms of transportation and delivery.

1.1 Purpose

- Develop an autonomous robot that combines computer vision algorithms and GPS navigation.
- To test and demonstrate the ability of the robot to navigate autonomously in different type of environments.
- Compare the robot's performance with already existing autonomous systems, highlighting its strengths and benefits.
- Analyse potential applications for the self-driving robot in various industries, such as logistics, common transportation, and shuttle services.
- Investigate potential future trends and opportunities in the field of autonomous vehicles and determine areas for further improvement and innovation.
- Create a plan for the commercialization of self-driving robots in Pakistan by manufacturing them locally and integrating them into various industries.

1.2 Document Convention

Throughout the document, you can expect to find definitions for key terms related to self-driving robots, their technology, and their applications. These definitions serve to provide clarity and facilitate effective communication.

1.2.1 Definitions

The following terms are used throughout this document and are defined as follows:

- **Computer Vision**: Computer vision is a field of study that focuses on enabling computers to extract meaningful information from visual data, such as images or videos. It involves developing algorithms and techniques that mimic human vision capabilities, allowing computers to understand and interpret visual content.
- **Open-Source Vision Library (OpenCV):** A programming function library that is open-source, free, and mostly targeted at real-time computer vision. It provides a broad range of capabilities primarily utilized in creating real-time computer vision applications.

- **YOLO:** YOLO is known for its efficiency and speed in detecting objects in an image or video stream. Unlike traditional object detection algorithms that require multiple passes over the image, YOLO divides the image into a grid and predicts bounding boxes and class probabilities directly using a single neural network. This allows YOLO to achieve real-time object detection with good accuracy.
- **Lane detection:** Lane detection is a computer vision technique used to identify and track the lane boundaries on a road or a similar environment. It is commonly employed in applications such as autonomous driving, advanced driver-assistance systems (ADAS), and lane departure warning systems.

1.3 Intended Audience and Reading Suggestions

- **The intended audience for this document includes:**

  Engineers and researchers working in the field of robotics and autonomous systems who are interested in self-driving robots and their applications.

  Professionals in the logistics and transportation industry who are exploring automation solutions for their operations.

  Decision-makers in the technology and manufacturing sectors who want to understand the potential of self-driving robots for commercialization.

  Students and academics studying robotics, artificial intelligence, and computer vision who want to gain insights into the advancements in autonomous systems.

- **Reading suggestions:**

  To gain a comprehensive understanding of the topics covered in this document, consider the following reading suggestions:

  Familiarize yourself with the basics of autonomous robots, computer vision, and machine learning.

  Explore research papers and articles on self-driving technology, algorithms, and sensors. Read about real-world applications of autonomous systems in industries such as logistics, transportation, and delivery services.

1.4 Project Scope

- **Design and development of an autonomous robot:**

  Integration of computer vision algorithms and GPS navigation.

  Selection and installation of sensors, cameras, and hardware processors.

  Development of software for data collection, processing, and decision-making.

- **Testing and demonstration:**

  Conducting tests to evaluate the robot's ability to navigate autonomously in different environments.

  Assessing its performance in terms of accuracy, efficiency, and safety. Demonstrating the robot's capabilities to stakeholders and potential users.

- **Comparison with existing autonomous systems:**

  Analysing and benchmarking the performance of the self-driving robot against other autonomous systems available in the market.

  Identifying the strengths and benefits of the developed robot in comparison to existing solutions.

- **Applications and potential use cases:**

  Exploring and analysing potential applications for the self-driving robot in industries such as logistics, common transportation, and shuttle services.

  Identifying specific use cases where the robot's capabilities can be utilized effectively.

- **Future trends and opportunities:**

  Investigating emerging trends and advancements in the field of autonomous vehicles and robotics.

  Identifying potential areas for further improvement and innovation in self-driving technology.

  Assessing the market potential and opportunities for commercialization of self-driving robots in Pakistan.

1.4.1 Product Perspective

- **Interaction with the environment:**

  The self-driving robot operates within a physical environment, such as roads, pathways, or designated areas.

  It interacts with various objects, obstacles, and potential risks present in the environment.

The robot uses its sensors, cameras, and algorithms to perceive and understand the environment, enabling it to navigate autonomously.

- **User interaction:**

  The self-driving robot may have interfaces or controls for user interaction in future, such as a user interface for monitoring and controlling its operations.

  Users may interact with the robot through a mobile application, a web interface, or physical controls for specific tasks or instructions.

- **Stakeholders:**

  The self-driving robot project involves stakeholders such as engineers, researchers, manufacturers, and potential customers.

  Stakeholders may include regulatory authorities responsible for defining and enforcing safety standards for autonomous systems.

  Depending on the application, stakeholders could also include end-users, such as logistics companies, transportation services, or other industries utilizing the self-driving robot.

1.4.2 Product Features

- **Autonomous Navigation:** The self-driving robot has the ability to navigate autonomously without human intervention. It uses advanced algorithms, sensors, and cameras to perceive its environment and make real-time decisions on path planning, obstacle avoidance, and lane switching.
- **Computer Vision:** The robot is equipped with computer vision techniques to extract and analyse data from its sensors and cameras. This enables it to detect objects, identify obstacles, and recognize road markings, allowing for accurate navigation and decision-making.
- **GPS Integration:** The robot incorporates GPS navigation to determine its location, calculate routes, and follow predefined paths. GPS integration helps the robot to efficiently navigate to specific destinations and perform tasks along the way.
- **Safety Systems:** The self-driving robot incorporates safety mechanisms to ensure the well-being of pedestrians, passengers, and other vehicles on the road. It may include features such as emergency braking, collision mitigation, and compliance with traffic rules and regulations.

- **Data Collection and Analysis:** The robot collects and processes data from its sensors and cameras for analysis and optimization. It can gather information about its performance, environment, and user interactions, enabling continuous improvement and adaptive behaviour.

1.5 Assumptions and Dependencies

1.5.1 Assumptions:

- **Availability of Reliable Sensor Data:** The self-driving robot relies on accurate and reliable data from its sensors, such as cameras, LiDAR, and GPS. It assumes that the sensor data will be available and sufficient for making informed decisions and navigating autonomously.
- **Adequate Infrastructure:** The robot assumes the presence of a suitable infrastructure, including well-maintained paths and clear road markings. It assumes that the infrastructure will support safe and efficient navigation.

1.5.2 Dependencies:

- **Sensor Performance and Calibration:** The performance and calibration of the robot's sensors, such as the camera and sonar sensors, are crucial for accurate perception of the environment. The robot's functionality and decision-making depend on the proper functioning and calibration of these sensors.
- **Availability of Power Supply:** The robot depends on a stable and sufficient power supply to operate continuously. It requires access to power sources or batteries with enough capacity to support its navigation and computation needs.
- **Maintenance and Support:** The robot may require regular maintenance, software updates, and technical support to ensure its optimal performance. It depends on the availability of maintenance services, spare parts, and expertise to address any issues or improve its functionality over time.

1.6 Broader Impact (UN SDGs)

The following SDGs are focused on our project:

- **Sustainable Cities and Communities (SDG 11):** Self-driving robots can enhance urban mobility and transportation systems, reducing traffic congestion, emissions, and the need for private vehicle ownership. They can contribute to creating inclusive, safe, and sustainable cities by improving accessibility and efficiency of transportation services.
- **Industry, Innovation, and Infrastructure (SDG 9):** Self-driving robots represent a technological innovation that can drive advancements in the robotics and automation industry. They can improve the efficiency, productivity, and safety of industries, such as logistics, transportation, and delivery, leading to sustainable economic growth and infrastructure development.
- **Responsible Consumption and Production (SDG 12):** By automating repetitive and mundane tasks, self-driving robots can optimize resource utilization, reduce waste, and improve production efficiency. They can contribute to responsible consumption and production practices by streamlining operations and minimizing environmental impact.
- **Climate Action (SDG 13):** Self-driving robots can help reduce greenhouse gas emissions by optimizing route planning, minimizing traffic congestion, and enabling efficient logistics and transportation. By promoting sustainable mobility solutions, they can contribute to mitigating climate change and promoting cleaner and greener transportation systems.

1.6.1 Targeted SDGs

The following SDGs are targeted in our project:

- Sustainable Cities and Communities (SDG 11)
- Industry, Innovation, and Infrastructure (SDG 9)
- Responsible Consumption and Production (SDG 12)
- Climate Action (SDG 13)

1.6.2 Potential Mapping

- **Innovation:** Self-driving robots are a cutting-edge technology that can revolutionize various sectors such as transportation, delivery, agriculture, and mining. They can offer new solutions for complex problems and create new opportunities for businesses and consumers.
- **Safety:** Self-driving robots can enhance safety by reducing human error, fatigue, and distraction. They can also perform tasks that are dangerous or hazardous for humans, such as mining, firefighting, and bomb disposal.
- **Social Impact:** Self-driving robots can have positive and negative impacts on society. On one hand, they can create new jobs, improve accessibility, and reduce emissions. On the other hand, they can displace workers, increase inequality, and raise ethical and moral dilemmas. By considering this checklist, stokehold
- **Scalability:** Self-driving robots need to be scalable and adaptable to different environments and situations. They need to have the ability to learn from their experiences, update their models, and communicate with other robots and humans.

Chapter: 02

Literature Review

2.1 Self-Driving Robot using Neural Network

Written by Akshay Mogaveera; Ritwik Giri; Mihir Mahadik; Anup Patil, this research talks about the growing importance of AI based self-driving vehicles and how they may be able to bring comfort and even more safety to drivers. They selected a path for the robot to travel and based on that path, they trained the robot. The problem that we feel in this project is that they did not only have to select a suitable path, but also train it on that specific area before implementation. In our project, we plan to make the project such that it will be trained at a general footpath which will potentially make it capable to drive on any other path without the need of training it on that specific area. [1]

2.2 Mobile Robot Platform with Arduino Uno and Raspberry Pi for Autonomous Navigation

Stelian-Emilian Oltean integrates features like navigation, mapping, and obstacle avoidance by using a microcontroller and an on-board computer on a fixed four-wheel chassis which is guided by the line following technique. This project, however, requires a line to be drawn on the required path which may become a tedious task and would require a different setup every time the desired path is changed. We propose to design the robot such that it is generalized enough to not require an explicit setup for every different location. It should potentially be able to adapt to any similar paths that it has been trained at. [2]

2.3 Deep learning-based image recognition for autonomous driving

Hironobu Fuji Yoshi's detailed research focuses on one of the most crucial parts of our project, as we extend our search in the field of image processing using AI, we find this paper to have extensive information about the theoretical aspects and implementation details. We hope to use this research as an aid for the image processing to be done in our project. [3]

2.4 A Lane Detection Approach for Driver Assistance

H. Wu, Y. Liu and J. Rong proposed a general technique for lane detection that combines filters developed by Freeman and Adelson with Canny edge detection algorithm. The steerable filters used in this paper are based on second derivatives of two dimensional Gaussians. These filters are helpful in many early vision and image processing tasks. The filter results are then processed to remove noise based on the road. There algorithm is able to provide accurate extraction of lane edge markings under different light and road situations. [4]

2.5 A new lane following method based on deep learning for automated vehicles using surround view images.

This literature review focuses on a research paper titled "A New Lane Following Method Based on Deep Learning for Automated Vehicles Using Surround View Images" by M. Lee, K. Y. Han, J. Yu, and Y.-S. Lee. The paper explores a novel approach to lane following in automated vehicles by leveraging deep learning techniques and surround view images. This literature review provides an overview of the key contributions and findings of the referenced paper. [5]

Chapter: 03

Image Processing in self-driving robots

3.1 Computer Vision

Computer vision is the study that enables computers to visualize and analyze images and videos based on pixels in a similar fashion as humans see things. This technology enables the computers to carry complicated tasks like object tracking and extracting useful data from images.

This is done by a combination of mathematical and physical concepts which involved taking input from cameras and sensors as frames and performing different computations on those images to extract useful data for multiple applications. These applications may involve color segmentation in forms like RGB and HSV and edge detection algorithms mostly used in self-driving cars.

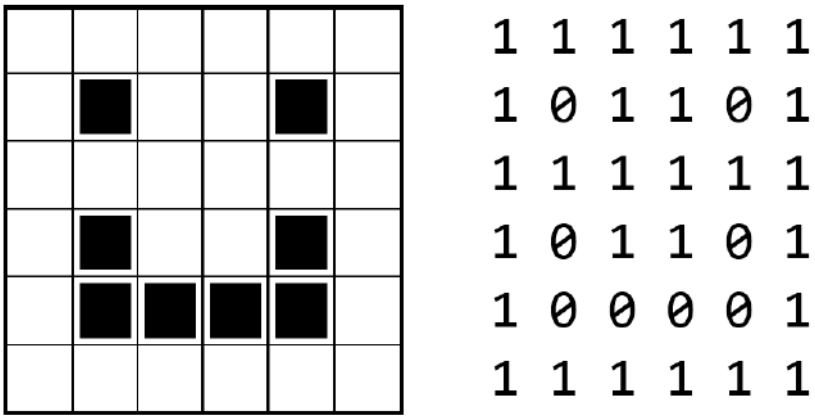

Figure 3-1 Computer Vision Data Representation

3.2 Common Libraries

3.2.1 OpenCV

OpenCV stands for open-source computer vision. This is a library which consists of numerous pre-defined programming functions and tools that enable computers to understand images, manipulate them according to the need of the application, and finally extract useful information to perform operations and interact with the physical world.

The library has a huge variety of functions used to manipulate and process digital images. To name a few, it consists of filtering, thresholding, and enhancement. The library also is capable to perform basic feature detection tasks and to provide coordinated of detected objects in images.

3.2.2 NumPy

Another important library commonly used in image processing is used for scientific computing in Python. It involves predefined mathematical functions that enable the programmer to compute complex equations and handle arrays and matrices easily. One of the most important tasks of NumPy in image processing is its capabilities to convert images into integer or float type matrices for enhancements and visualization of images.

3.3 Colour Models

Colour models are a mathematical way of defining colours in and image in the concept of image processing and computer vision. It allows users to interact with digital devices based on colour values and lets the machines do operations based on colour recognition. There are 2 main types of colour models, and both are necessary based on their special features which allow specific operations.

3.3.1 RGB colour Models

The RGB model stands for Red Green and Blue which is an additive model used to define colors. It's called an additive because it makes colors based on the addition of these 3 primary colors in different amounts and configurations. Each of the components is based on a value from 0 to 255 and combinations of all three of these channels and values produce different colors as humans perceive them. The combinations start from all the components being 0 which produces a black image, to all the possible combinations in between until all three of the channels become 255, that's when the output becomes totally white.

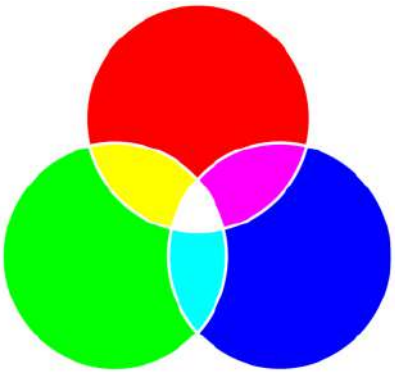

Figure 3-2 RGB color model

3.3.2 Grayscale colour model

Just like a single channel in the RGB color model, the gray scale model has a single channel for white and black color ranges typically from 0 to 255 with each of the level representing a single gray shade. While 0 represents completely dark, the shade gets lighter as we proceed from 0 to 255 after which the image becomes completely white.

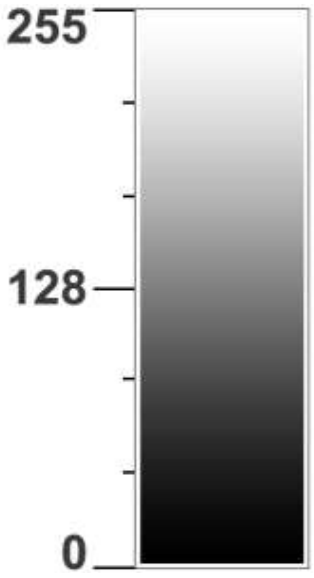

Figure 3-3 Gray Scale color model

Figure 3-4 Gray Scale vs RGB

3.3.3 HSV colour model

The HSV color model is used to improve the accuracy of color recognition by adjusting the brightness and saturation levels. In varying lighting conditions, the RGB color model may not be sufficient to accurately identify lane markings. The HSV color model separates the color information from the brightness information, allowing the algorithm to adjust the brightness and saturation levels to better detect the lane markings.

The color combinations provided by RGB may not be useful for all applications in computer vision. For that reason, we can shift to the HSV model which provides information in a more advanced manner and adds different detailed components in the color domain. Especially in self-driving cars when the subject is always in different lighting conditions. The HSV color models help a lot by superseding the color information with that of the brightness levels involved inside the image, it is this division that allows it to be more flexible in this regard. The color model mainly has 3 models as suggested by the name itself, Hue, Value and saturation.

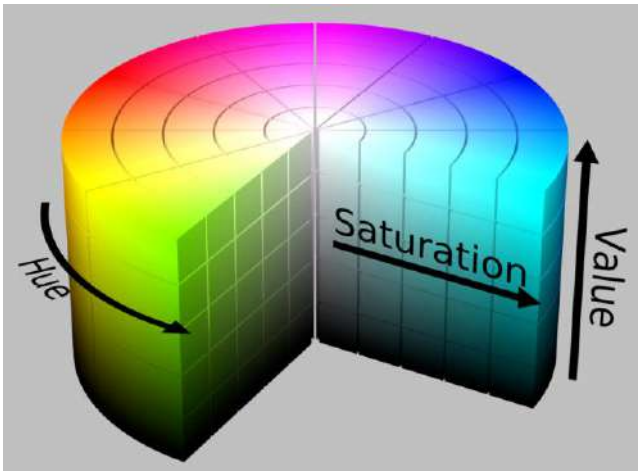

Figure 3-5 HSV colour model

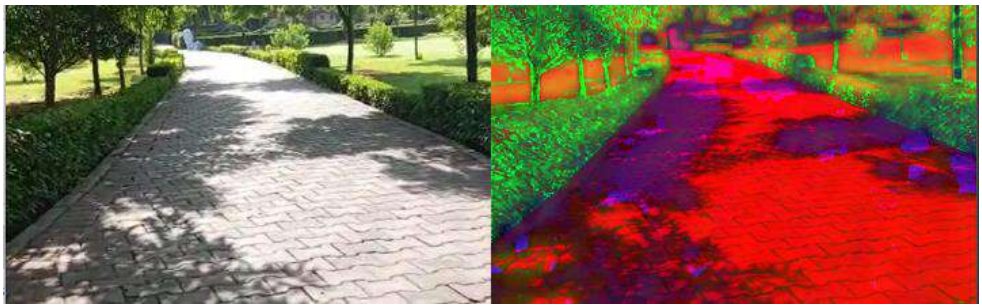

Figure 3-6 RGB to HSV conversion for lane detection

3.4 Lane Detection

3.4.1 Pre-Processing

The plan is to locate the road using colour recognition or edge detection and then determine the curve by summing up the pixels in the vertical direction, which is known as a histogram. The process will be divided into five stages: Thresholding, Warping, Histogram, Averaging, and Display.

This includes the identification of the desired path by colour segmentation and determining the curve value based on the deviation of pixel summation from the centre through a histogram. The process will be done in 4 stages: Thresholding, Warping the image, creating a Histogram, and averaging.

3.4.2 Thresholding

This is a process where a coloured image is converted into a black and white image called the binary image. This is done by setting a particular threshold value according to the desired colour on the bases of which the required part of the lane gets coloured black and all other part are turned white.

This is done by the help of the HSV colour format that allows the image to be divided into useful components and it helps in supressing the background and highlighting the required path.

After the thresholding part is done, a new image is formed which can be masked to the original image so that the area of interest can be highlighted.

Figure 3-7 Thresholding image

3.4.3 Image Warping

Image warping is one of the most important tools of image processing since it allows the transformation of an image through stretching and contracting based on the image coordinates given by the user. This is a technique which can be highly useful in terms of path detection and alignment of images.

Path detection in self-driving cars involves the identification of the curve just beyond the current position of the robot. However, the angle of the camera usually views straight from the front of the car which captures the image of what the robot would see form its perspective.

This is a problem because the curve at the current position is very different from the curve changing far away from the robot. However, when the camera captures the image, it looks at the curve available far away from the robot which makes it reject the curve present in the current position.

To solve this issue, if the camera could look at the path in a Bird Eye view, it would be able to focus on the current path much better than if it was looking straight in front of the robot. This is what warping allows the camera to do. It makes the current curve area wider and stretches the future curve area so that it is ignored, and the robot can make the curve decision based on the curve present in the current location rather than considering the curve available far away which may result in ambiguous movement of the robot.

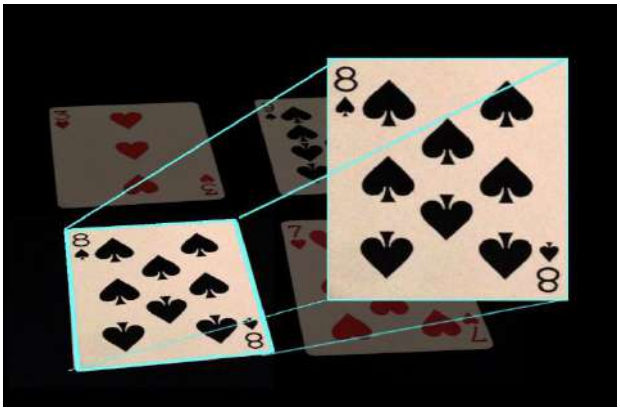

Figure 3-8 Warping image

3.4.3.1 Matrix Operation for warping

$$\begin{vmatrix} x' \\ y' \\ s \end{vmatrix} = \begin{vmatrix} a1 & a2 & b1 \\ a3 & a4 & b2 \\ c1 & c2 & 1 \end{vmatrix} \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

Scaling Factor

Transformation Matrix (M)

$a1 \quad a2$
$a3 \quad a4$ defines transformations such as rotation, scaling etc

$b1$
$b2$ defines translation vector

$c1 \quad c2$ projection vector

3.4.4 Creating a Histogram

After obtaining a bird eye view from the warping concept, the next part involves understanding the deviation caused in the path and the curves that comes ahead. To do this, we already know that the path can be distinguished from the background by obtaining a binary image. Now the curve can be extracted by creating a histogram with the values of pixels that are present in each column of the pixel frame. After receiving the histogram of each column, a threshold value can be set after which the column will be considered as a path. This threshold value is there to avoid noises and columns that have lesser white pixels due to unwanted regions so that the model can be made with better accuracy. However, this method has a few issues that need extra attention discussed in the next section.

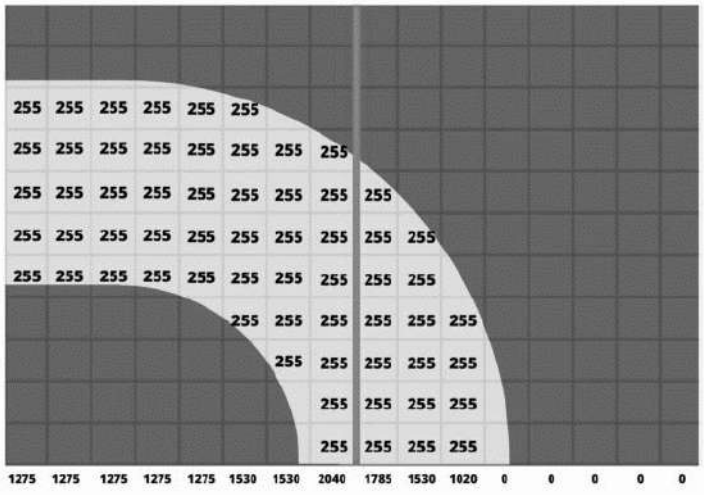

Figure 3-9 Histogram of pixel columns to analyze the curve direction.

3.5 The Histogram centre problem

3.5.1 Ideal Case

An ideal case of the centre line is when the robot tries to keep itself in the centre of the complete image it captures regardless of the path. This will work fine as long as the path is always at the centre of the complete frame the camera captures. Every time the paths curve would deviate from the centre, the robot will calculate the deviation with respect to the centre of the whole frame. However, this will result in a problem when the path is not in the centre.

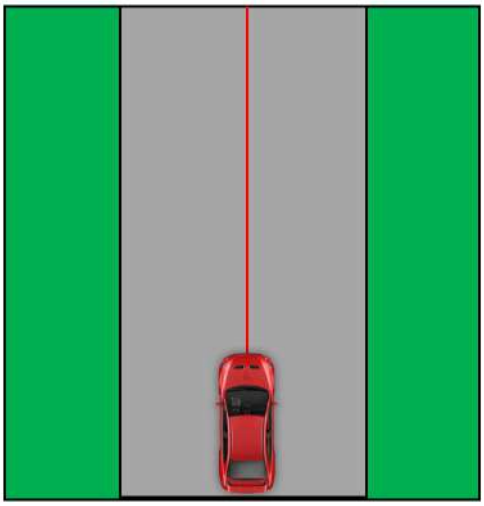

Figure 3-10 Ideal case of center line with path in the center

3.5.2 Non-Ideal Case

At times when the path is not in the centre of the frame, the pixels on either side of the centre of the frame will vary and the algorithm will try to turn the car even if the road is straight. This will result in ambiguous movement of the car since the centre line of the process is the middle of the frame and not the middle of the path itself. To solve this, the centre line should not be merely the centre of the complete frame received by the robot. But is should be the middle of the path regardless of how much it is deviated from the centre of the frame:

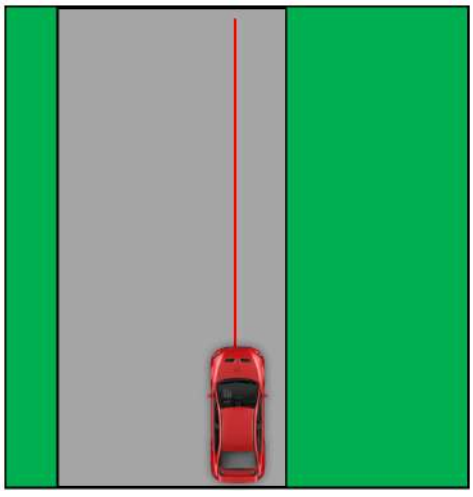

Figure 3-11 Non-ideal case with path not at the center of the frame

3.5.3 Solution

The problem lies in the calculation of the centre line and its bases on the centre of the frame. However, if we change the centre line so that it lies in the centre of the path instead of the frame regardless of where the path is with respect to the whole image. This will make the model much more flexible and at the same time more reliable because no matter where the path is with reference to the actual frame, the centre line will always be considered in the middle of the path itself.

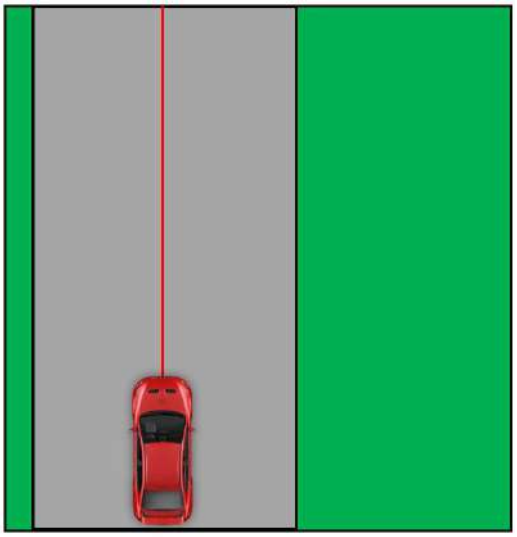

Figure 3-12 Adjusting the center line in the middle of the path rather than keeping it in the center of the frame

3.6 Machine Learning

Machine learning is a technology which allows computer systems to learn and improve over time by performing multiple iterations of a particular task repeatedly.

A traditional program uses a hard code method that sort of sets the exact rules and details on the bases which the computer performs its tasks. However, in the machine learning model, the computer is given the flexibility to make decisions even if completely new values are obtained as inputs and it made versatile to understand and analyze the data.

This is mainly categorized into 2 main types, unsupervised learning, and supervised learning. While supervised learning has properly labelled data that is used to train the algorithm. Unsupervised learning does not have labelled data and instead, it relies on the iterations of the task and the error that is received through each iteration that it tries to reduce through experience.

3.7 Convolutional Neural Network (CNN)

CNN is one of the most widely used concepts in image processing and object detection. It is a type of deep neural network that uses multiple convolution layers to identify any patterns in an input image that may be in close resemblance to the data on which the model is trained upon.
The convolution layers include kernels with different weights and sizes that pass through the complete image to gather useful information about the image for example edges.

After being passed through the convolution layers, the output is then passed through pooling layers that extracts the major information from the image and then sends it to the fully connected layers to finally classify the image based on its features.

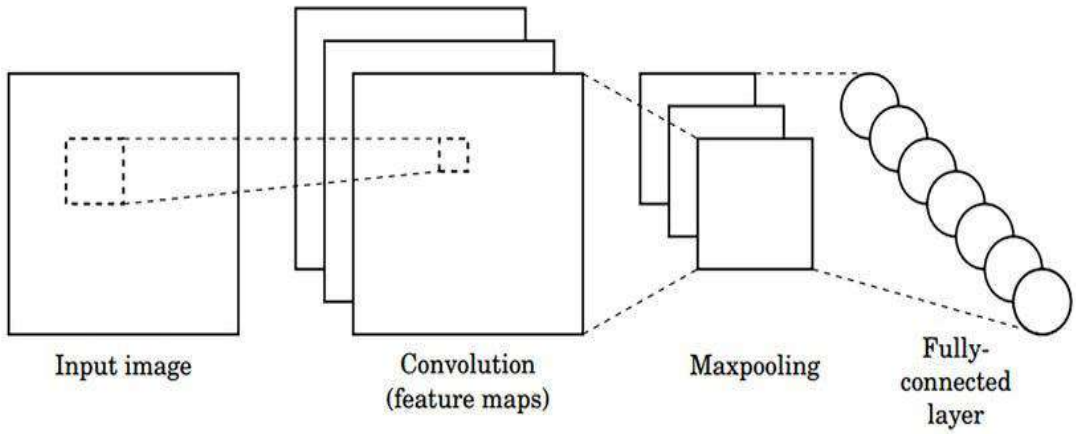

Figure 3-13 Convolution Neural Network

3.8.1 Convolutional Layer

This layer produces a feature matrix according to the most dominant information from the image. This is done by passing the input image through multiple types and sizes of kernels with which the image is convolved. Many different features can be detected from the image using these kernels or filters, the most common example is the edge detection kernel that is used in variety of applications especially in the lane detection of a self-driving car.

3.8.2 Pooling Layers

Another very important step in the CNN model is the pooling layer, this basically reduces the size of the original image but increases the significance of each pixel. This is done by taking the maximum of all the pixels inside the kernel and rejecting all other pixels which results in a smaller image with

only the information that matters the most. This also helps in the model being faster because of lesser data provided for processing due to a decreased size of the image.

3.8.3 Fully Connected Layers

This is the final step of the multi-layer network which is simply there to give out a final output based on the error and final predictions of the network. This is the layer which is responsible for finalizing the classification of the image based on the extracted features and provide the required output.

3.9 YOLO

3.9.1 Overview of YOLO

Yolo stands for You Only Look Once; it is one of the most popular machine learning algorithms that is used to detect objects in real time. It uses CNN Convolutional Neural Network to extract object data from images.

The architecture of YOLO is based on a DNN which is short for deep neural network. The images that it takes as an input goes through a variety of layers including convolution and activation functions. This results in the model giving the coordinated of the detected objects as an output on which bounding boxes can be drawn using OpenCV.

3.9.2 Object detection using YOLO

Object detection involves the identification of trained objects in an image or a live video. YOLO uses multiple CNN layers and each of them provide multiple predictions of all the objects involved in its dataset. The object which then has the highest probability wins and comes out to be the output. This probability is called the confidence scores. YOLO supports real time object detection with multiple detections in an image at a single time even if the objects are being overlapped.

3.9.3 COCO Dataset

COCO is a standard data set for common objects. it stands for Common Objects in Context. As the name suggests, the data set includes the images daily life common objects that usually surrounds us. This includes objects like chairs, scissors, cats, and dogs. The dataset is useful for common object detection which does not need a specified object to be detected and trained upon. For applications like these, this pre trained data set can be used either for training a new model or using already built models that are trained like YOLO.

3.9.4 Distance Estimation

As a solution to the object detection problem where the robot sees even the far away objects as a threat, a distance estimation algorithm is required so that it can estimate the distance of the objects closest to the robot. This will make sure that the robot only stops for the objects which are a threat to it. Without this feature, the robot would have stopped every time it looked at any object no matter how far away it is from the robot. It will also make it more efficient since a person who walks across the path far away from the robot will have no impact in the speed of the robot since he proves to be of no threat to the robot from such a far distance. However, without this feature, the robot would have stopped far away from the person and waited until the person crosses the complete path and gets out of the frame. Which in reality, has nothing to do with the robot's movement because by the time the robot reaches the location at which the person was crossing the road, the person would have crossed the road, way before it.

3.9.5 Methodology

The idea is to draw a bounding box about each possible object in the path and estimate the distance between the object and the robot based on the size of the bounding box. The bounding box coordinates will be given by the YOLO algorithm which was discussed in the previous topics and then it can be passed to the distance calculator to proceed with the next step. The greater the bounding box, the closer the object is estimated to be from the robot. This estimation can be done better by using the following calculation.

Distance = (real width of object * focal length) / pixel width of the object.

The **real width** of the object can be measured physically by measuring a few objects of the same class with slight variation and then taking an average of those sizes so that an estimated average size of the object can be used.

The **pixel width** can be easily extracted after the bounding box is made from OpenCV which will return the number of pixels present in the frame from one edge of the bounding box to another.

Finally, the **focal length** can be calculated using the same formula but at a given scenario when the distance is known. So, for a given case, if the distance is calculated of a person standing about 100 inches from the camera for a single time, the focal length can be calculated as a sort of a calibration method after which, the distance formula will be completed for future distance measurement. Since

the focal length will always remain the same, the formula can be manipulated so that the focal length of the camera is calculated only once before setup using the following formula:

Focal Length = (pixel width * known distance) / real width of object.

3.10 Challenges & Limitation in Distance Estimation using a single camera.

An accurate measure of distance usually requires a special hardware, doing this just by a single camera can only be an estimation. One of the main problems of calculating distance using a single camera is that it depends solely on the bounding box generated by the object detection module and nothing else. Having said that if the machine is calibrated by a thin person with a small hight. It will not work as good for a healthier person with a longer hight because the bounding box greater for the bigger person will be bigger for the same distance. And since the distance is calculated by that very size, the machine will think that the bigger person is closer. Another similar problem is that the gestures a person create will also be considered as the bounding box will be altered every time a person moves their hand. A graphical representation of this limit is provided below.

3.10.1 Limitation

Both objects (a) and (b) are standing at the same distance from the robot but object (b) appears closer due to a bigger bounding box.

(a) Small Bounding box implying greater distance b) Larger bounding implying decreased distance

Figure 3-14 Bounding Boxes

This is a problem that occurs when calculating distance through a single camera. As a prototype, this is enough estimation for the robot to when to stop if the object is closer than a threshold distance. However, in cases where a more accurate distance tracking model is required, special hardware should be attached like 2 cameras used in distance measurement or dedicated hardware like Lidar.

Chapter: 04

Motion Control

4.1 Driving Mechanism

4.1.1 Differential drive

To keep the mechanism simple, all 4 motors can be fixed at their position and the direction of the robot can still be altered according to the need of the path. A normal car can be imitated by using servo motors that turn the wheel according to the need of the robot. But differential dive mechanisms are just as popular, and they are mostly used in robotics.

The direction is controlled by controlling the speed or direction of one side of the robot compared to its opposite side. If the robot require a complete turn, both the sides will be working in the opposite direction to make it rotate. Under normal conditions, the robot takes slight turns which is achieved by a all 4 motors working in the same direction, just one side of the motors have their speed reduced. The diagram below illustrates this concept.

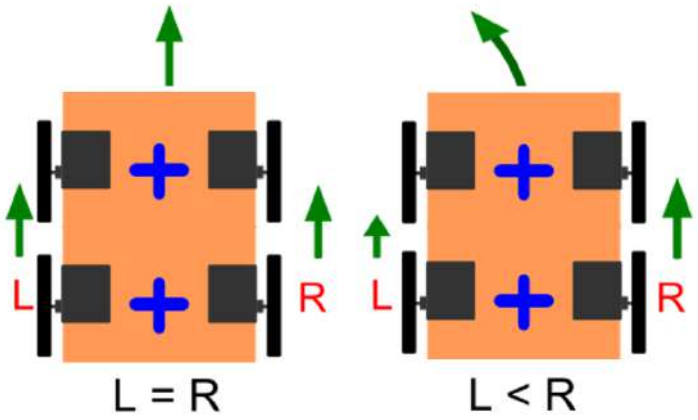

Figure 4-1 Motor Working Mechanism

4.1.2 Instability with differential drive

Due to the sudden changes in the curve values provided to the robot, it becomes unstable and produces oscillations in the robot's movement. Specially in the differential drive mode, since the speed of the

wheels are constantly changing, it produces jerk. This movement can get more stable by using a PID controller.

4.2 PID Controller

A PID controller short for Proportional Integral Derivative controller is one of the most common controllers used in robotics and in various control system-based machines that allow a particular movement to become more accurate and smoother. The proportional element of the controller provides and output which is directly proportional to the error. This error is calculated as the difference between the required value and the current value of the system. So, the proportional component increases linearly when the error increases and decreases according to the error as well.

A Integral part of the controller gives an output which is directly proportional to the error but the difference is that is relates to the accumulated error and not the instantaneous error. This allows the system to handle the steady state error better.

The Derivate art of the controller is responsible for the damping of the movement since it corresponds to the rate of change of error. This helps the system to reduce the effects caused by momentum and stop in a decelerating fashion.

CL RESPONSE	RISE TIME	OVERSHOOT	SETTLING TIME	S-S ERROR
Kp	Decrease	Increase	Small Change	Decrease
Ki	Decrease	Increase	Increase	Decrease
Kd	Small Change	Decrease	Decrease	No Change

4.2.1 P Controller

In a **P controller**, since the output varies directly with the error, the resulting output has an overshot.

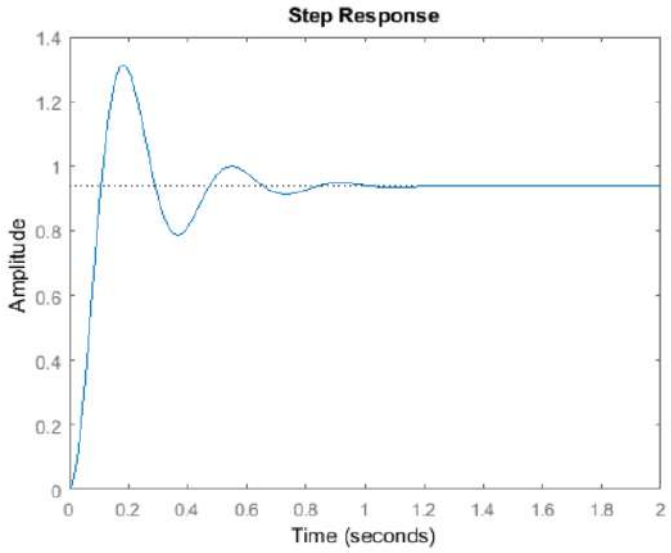

Figure 4-2 P-Controller

4.2.2 PI Controller

The integration component of the controller decreases the rise time and the steady state error of the machine.

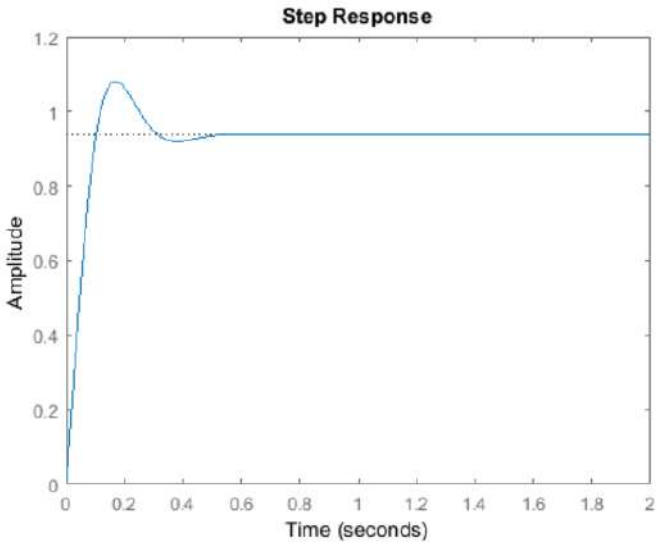

Figure 4-3 Simulation of Integral component

4.2.3 PID Controller

The derivative element in the PID controller is added to improve the response time of the system and decreases the overshoot that causes oscillations.

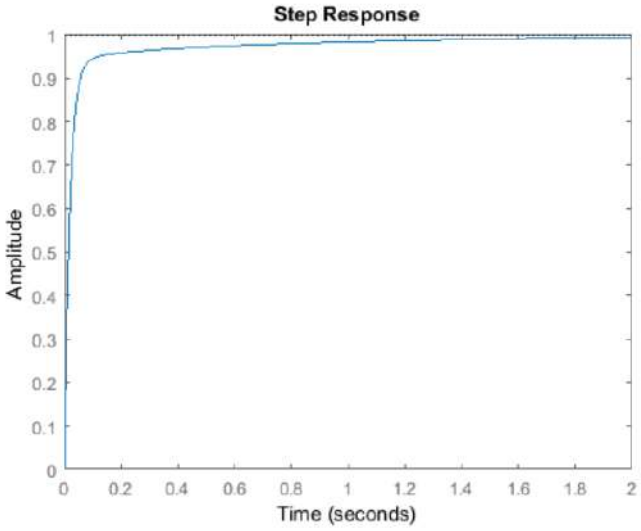

Figure 4-4 Simulation of derivative component

4.3 PID Arduino Library

The PID library in Arduino consists of predefined functions that include all the mathematics involved in the PID controller. It has in built functions that take input values and pass them through a transfer function that is defined by the Kp, Ki and Kd values defined in the input parameter of the function.

4.4 P Implementation of PID in Self-deriving robot

The lane detection module consistently provides the deviation of the robot's instantaneous position from the centre. This can be thought of as the error which needs to be fed as the input to the PID algorithm. The microcontroller then passes that error value to the PID controller which tries to minimize the error based on the values provided to it by the user. The user has to initially tweak these values in order to obtain a stable movement.

Chapter: 05

Hardware of the robot

5.1 Chassis

The robot is primarily made by an acrylic base with wooden sidewalls. The base has a support frame below it which is further supported by aluminum brackets. These brackets also act as the rigid mounts for the motors attached to the robot.

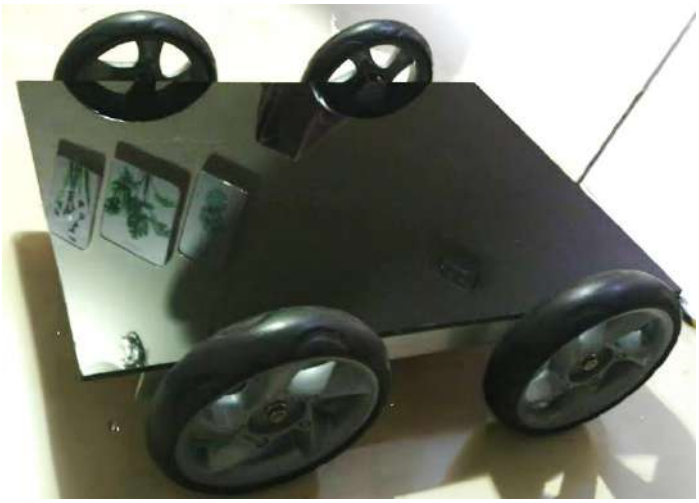

Figure: 5-1 Base of the Robot

5.2 Motion Control

5.2.1 Wheels

The choice of wheel was made carefully since it was a key element which played a huge role in the overall stabilization of the robot.
Firstly, the surface of the wheel was chosen to be of a rubber material. This absorbs a great amount of shock and plays an important part in the overall stability of the robot.

Secondly, the size of the wheel was important because the smaller the wheel, the higher chances it had to bypass the edges and small puddles throughout the path. Especially the edges and the dip it

experiences in the middle of any 2 bricks. Having a wheel of a size as big as this easily bypassed those disturbances and resulted in the stability of the robot.

5.2.2 Motors

Since the speed of the robot did not matter more than the load that it could bear, the motors used are fully geared which allows them to convert that speed in the favour of a higher torque. Since the applications of the robot mostly require load attached on board for example delivery items, the speed was reduced by adding the geared motor as a trade-off for higher torque. One more fact worth noting about the geared motors is that these motors usually require more current. In our case, the peak current of our motors which is the current at when the motor experienced its maximum load is about 3 amperes. A normal high-speed motor without any gears attached may not experience such great current, however when the load is attached, the current is also increased. This was catered by using a suitable motor driver shield that is discussed below.

Figure: 5-2 3A peak current geared motor similar to the ones used in the robot

5.2.3 Motor Drivers Used: ZK-5AD

We used the ZK-5AD 5-amp driver as our motor driver. This driver can handle up to 5 amps of current, which is sufficient for the motors we used. The driver also has various protection mechanisms to prevent damage to the motor and the driver itself.

The motor driver used in our case handles a peak current of 5 amps which is sufficient for our case since the max current of any motor is 3A and that is just the worst case because ideally the motors never operate on their max load. The drive used also has a protection mechanism that

burns itself in case of any failure to save the motor or the processor that controls it. This driver can handle 2 motors at once. Providing us a great deal because we could easily control all 4 motors using just 2 drivers.

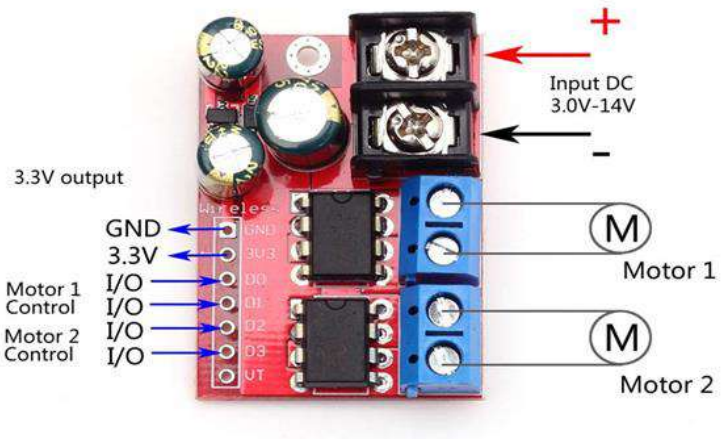

Figure 5-3 Motor Driver ZK-5AD

5.2.3.1 Motor Drivers Specifications

Specification	Description
Input voltage	3V to 15V DC
Output current	Up to 5A continuous
PWM frequency	15kHz
Maximum power dissipation	25W
Control signal voltage	3.3V to 24V DC
Operating temperature range	-20°C to 85°C
Dimensions	50mm x 44mm x 27mm
Weight	40g
Protection features	Over-current protection, under-voltage protection, over-temperature protection

Table 5-1 Motor Driver Specification

5.3 Processors

5.3.1 On Board Computer

The image processing part which involves lane and object detection both are done on the onboard processor. For this we initially planned to use a raspberry pi but later we shifted to a laptop PC itself for multiple reasons

We already had an extra laptop PC with a decent RAM and processor which was way faster and powerful compared to the raspberry pi. While the pi had 1 GB ram, the laptop had 8GBs.

Moreover, the cost of the Pi had been increased up to 3 times its normal cost which was just a useless expense specially after the availability of the laptop. Yes, the Pi was way smaller and more portable, but in our case, we had more than enough space in our laptop which could easily accommodate a single laptop without any problems.

5.3.2 Arduino Mega

The Arduino Mega is a microcontroller board based on the ATmega2560 microcontroller. It is one of the largest and most powerful boards in the Arduino family, with 54 digital input/output pins, 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz quartz crystal, and a USB connection for programming and serial communication.

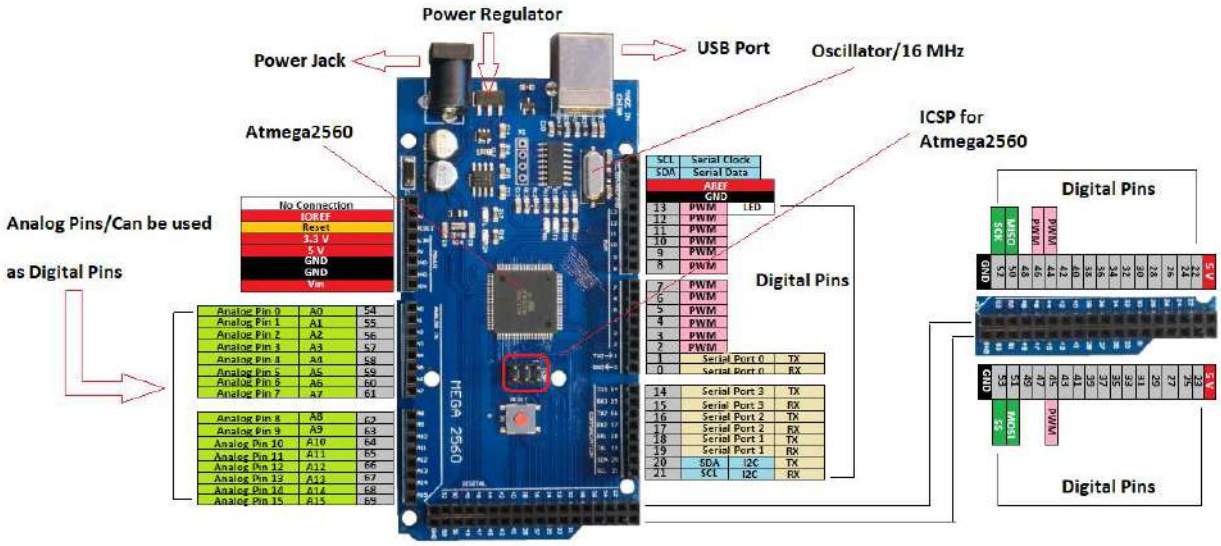

Figure 5-4 Arduino Mega Pin Configuration

5.3.2.1 Arduino Mega Specifications

Specification	Description
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB is used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3 mm
Weight	37 g

Table 5-2 Arduino Mega Specifications

5.4 Safety features

5.4.1 Ultra Sonic Sensor

Including ultrasonic sensors in the robot can add an additional layer of safety to prevent any accidents or damage. Ultrasonic sensors work by emitting sound waves and detecting the time it takes for the waves to bounce back. This allows the robot to detect obstacles or drop-offs in its path.

In case of any program failure or glitch, the ultrasonic sensors can help prevent the robot from falling off the path or colliding with any objects. This is particularly important in situations where the robot is operating in an uncontrolled environment or in close proximity to humans.

By integrating ultrasonic sensors, the robot can detect obstacles in its path and adjust its movement accordingly. For example, if the robot detects an obstacle in front of it, it can either stop or change its direction to avoid the obstacle. This can help prevent any damage to the robot or any other objects in its path.

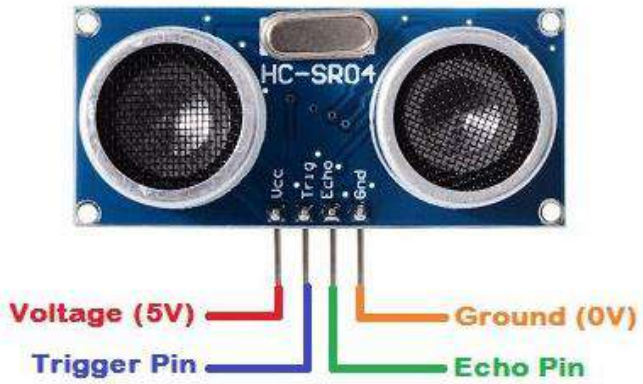

Figure 5-5 Ultra Sonic Sensor

5.4.1.1 Ultra Sonic Sensor Specifications

Specification	Description
Operating Voltage	5V DC
Detection Distance	2cm to 400cm
Accuracy	3mm
Operating Frequency	40kHz
Output Signal	Digital
Operating Current	<15mA
Dimensions	45mm x 20mm x 15mm
Weight	5g

Table 5-3 Ultra Sonic Sensor Specifications

5.4.2 Camera

A camera was needed for the main image capturing of the path. This was done using the mobile phone since it was readily available, had high quality, and saved the overall cost of the project. The camera used had to be of a premium quality since the better the input image is provided, the better the processing is going to be for a reliable result. Mobile phones also have a stability feature attached to them along with the high quality which ensures a better performance of the camera.

5.5 Location

5.5.1 GPS Module

The GPS module in the self-driving robot is an essential component that helps the robot determine its location and navigate autonomously. The GPS module receives signals from the Global Navigation Satellite System (GNSS) and calculates the robot's latitude, longitude, and altitude. These values are used to provide the robot's location and enable it to navigate towards a specific destination. The GPS module plays a crucial role in achieving the overall project goals of building a self-driving robot that can operate without human intervention.

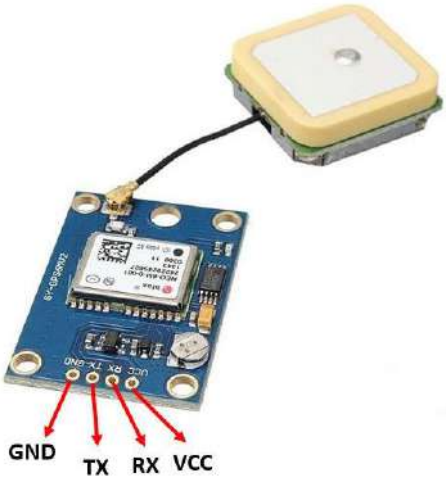

Figure 5-6 GPS module

5.5.1.1 GPS Module Specifications

Specification	Description
GPS Chipset	Unbox NEO-6M
Position Accuracy	< 2.5 m (CEP)
Time Accuracy	30 ns
Maximum Altitude	50,000 m
Data Rate	9600 bps
Operating Voltage	3.3 V
Power Consumption	20 mA (tracking), 45 mA (acquisition)
Dimensions	25 x 25 x 4 mm
Weight	5 g
Operating Temperature	-40 to 85 °C

Table 5-4 GPS Module Specifications

5.6 Power Source

Figure 5-7 Battery Source

5.6.1 Battery Specifications

Voltage: 12 volts

Current: 6 amps

Capacity: 5 Ah

The voltage rating of the battery, 12 volts, indicates the electrical potential difference across its terminals. In this case, the battery provides a stable 12 volts of electricity. The current rating, 6 amps, denotes the rate at which the battery can deliver electrical charge. It represents the amount of current flowing through the battery when it is discharged.

The capacity of the battery, 5 Ah, refers to its ability to store and deliver charge over a specific period. An ampere-hour (Ah) is a unit of electric charge, representing the capacity of a battery. In this case, the battery can supply a current of 5 amps for one hour before it is fully discharged.

Formula for Calculating Battery Duration:

Battery Duration = Battery Capacity / average Motor Current

Calculation:

Battery Capacity = 5 Ah

Motor Current = 6 amps

Battery Duration = 5 Ah / 6 A

Battery Duration ≈ 0.8333 hours (or approximately 50 minutes)

:

5.7 Circuit Diagram

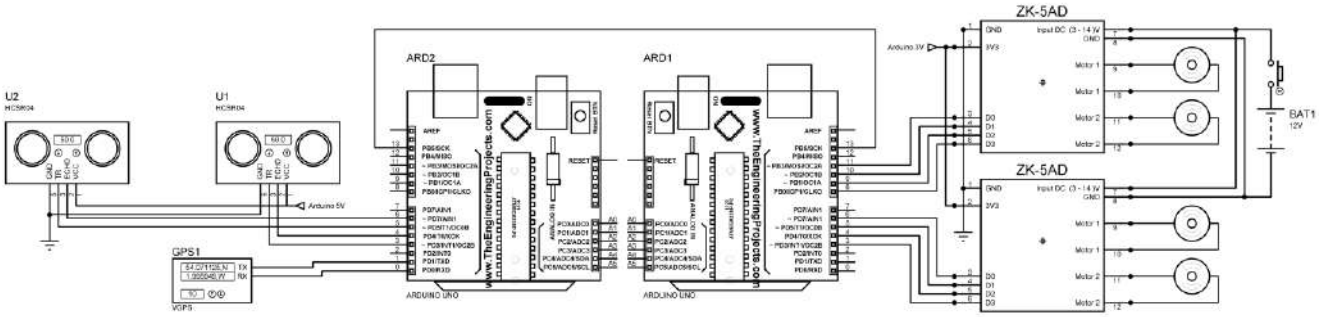

Figure 5-8 Circuit Diagram

Chapter: 06

Working of the Robot

6.1 Final Look

Figure 6-1 Robot Image

6.2 Block Diagram

The process starts off with the camera which is feeding live video to the computer on board for the surroundings to be analyzed. The microcontroller relates to ultrasonic sensors connected to for extra safety and guidance.

Apart from image sensing, the controller is also be connected to the GPS module so that it can process the navigation data and stay on its desired path. This data combined with the signals achieved from the sensors will be processed to finally provoke the motors for a successful maneuverer.

The microcontroller itself doesn't have enough current at its output to control the motors. Apart from this, the motors also produce back emf or overload current which can burn the microcontroller withing seconds. To solve these problems, a heavy duty high current motor driver is used so that it can handle all the protection tasks with high current.

A graphical representation of the above explanation is shown in the following block diagram:

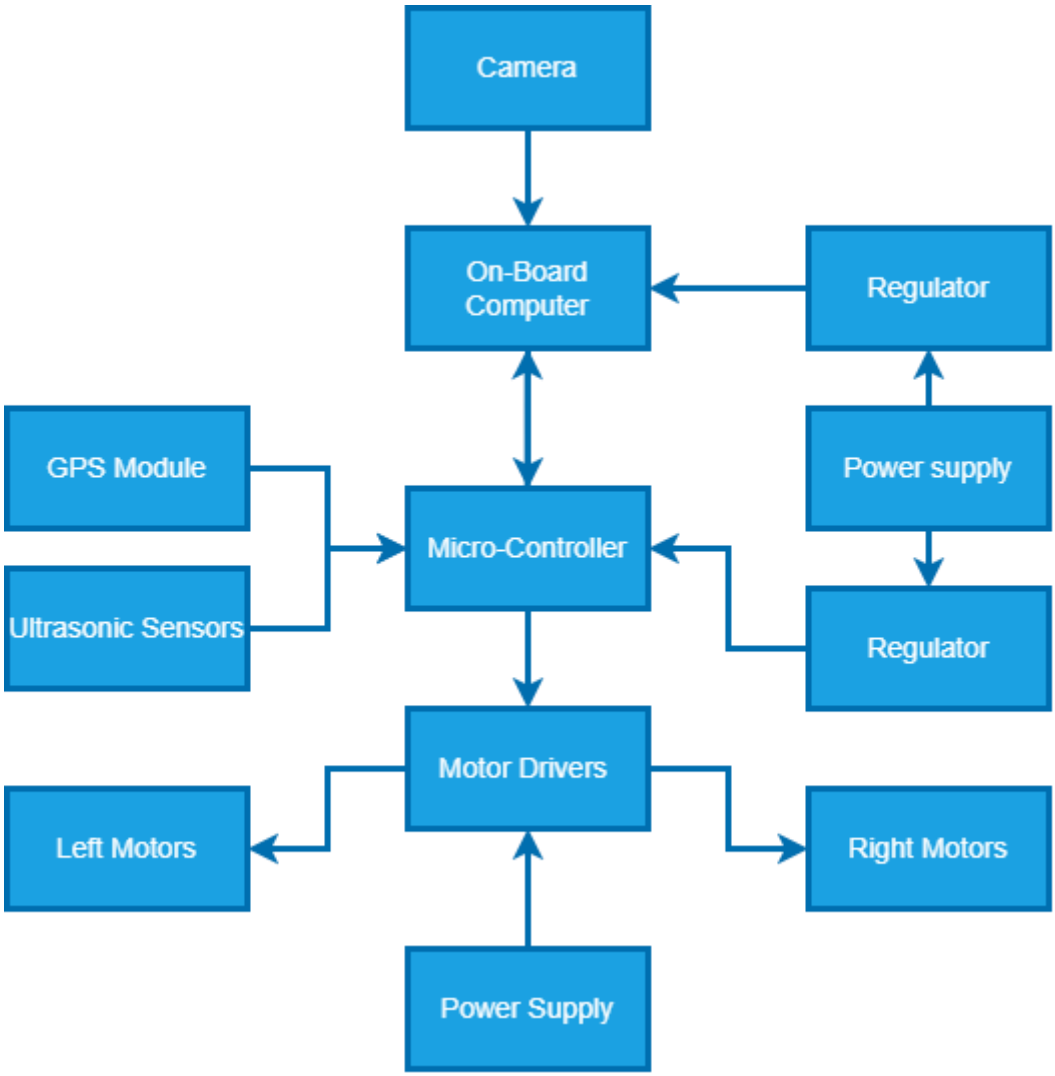

Figure 6-2 Block Diagram

6.3 Lane detection results

6.3.1 Warping

As discussed previously, the curve which is far away from the robot seems to interfere with the decision-making process of the direction of the robot. For this reason, warping was done to solve this issue and focus on the curve in the current location rather than the curve far away from the robot to prevent any sort of misguidance.

Warping of the image requires the 4 coordinates of the original image where the warped image would be mapped. This could be done manually by entering the values inside the warping functions one by one. However, a more reliable and easier method was to draw dot on the coordinates and alter them using track bars in real time so the user can choose a suitable warping position easily according to the path.

The trackbars work as an aid for the user to easily adjust the coordinates of the image according to the camera angle and the width of the path compared to the width and height of the robot so that an optimum warp value can be achieved for best results of the robot considering only the instantaneous curve rather than the curve far away.

The following images show the results for both warped and un-warped images to understand exactly how curve values from the far away path can be neglected. The blue dots shown are the warped coordinated which the user can select using the trackbars in real time.

(a)Results with 0 warping *(b) results with warping according to the set points*

Figure 6-3 Result of warping

6.3.2 Overcoming Noise

The histogram method explained earlier can now be used for the reduction of noise. The example below shows the scenario when there is another path that is being generated on the right side of the frame circled in the image. This has 2 problems, firstly, the computer could select one of the detected paths and try to follow either one of them at a given time. Secondly, since the center which is given by the black dot is always the center of the path and not the frame, the falsely detected path will interfere with the average point of the original path resulting in a false center of the main path.

To overcome this problem, the histogram method is used which ignores all the columns lesser than a certain threshold point and calculates the center of the original path only.

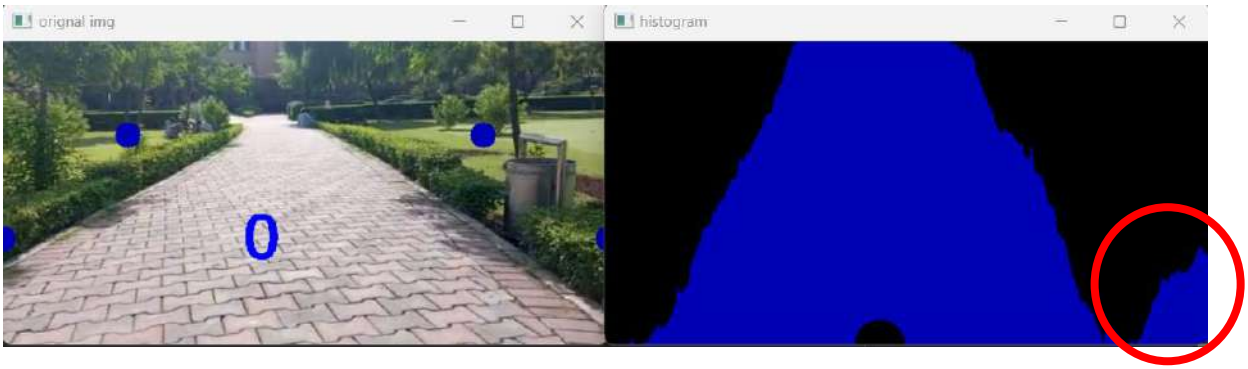

Figure 6-4 histogram method to overcome noise

6.4 Distance Estimation

The distance of a person was measured according to the theoretical background of the 2nd chapter. Firstly, the real width of a person was finalized by calculating the width in inches of different people and finding an average which came out to be 16 inches.

Next, a person was made to stand at a known distance from the camera and the pixel width was found by OpenCV. This information was enough for the focal length to be found according to the formula stated in chapter 2. The focal length of the specific camera that was used came out to be 800.

Finally, this distance was displayed along with the object detection module which returns the bounding box of the person found as shown in the figure below.

Figure 6-5 Object Detection Testing

6.5 Final Output

Both the lane and object detection modules were combined to give 4 main outputs to the user. The first output defines the value of the curve itself, and it also displays the exact position of the 4 coordinates which show the location of warping.
Secondly, the threshold result is shown after warping so that not only can the user select the most optimum position, but also check if the path is being detected properly. Otherwise, the user could go back and check the color model to fix the noise.

The third part is the histogram itself. This shows the performance of the machine for trying to keep the center black dot in the center of the path, deviation in this dot can suggest that the path is not being recognized properly. In order to fix this, the nose threshold values can be altered for optimum results.

The final part of the image processing is distance estimation which shows all the objects that it detects and calculates the distance of people it detects in the frame. This also returns a stop signal if the distance of the person goes beyond the threshold point.
The image below shows all the four frames displayed by the robot during its journey:

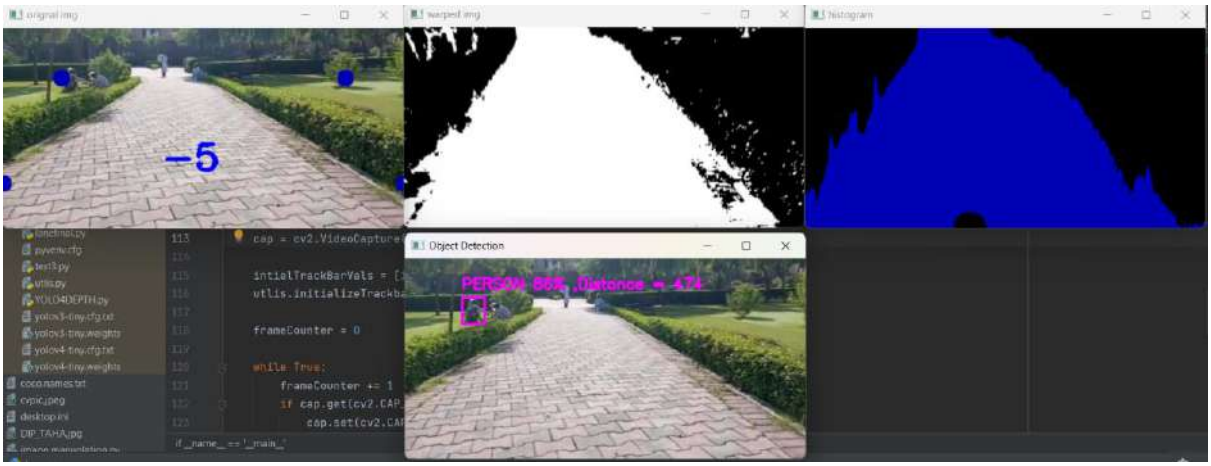

Figure 6-6 Final Result

6.6 GPS Testing

Another essential element of the self-driving robot is the GPS module that helps the robot to continuously determine its real time location and help it reach its destination fixed by the user.

The latitude and the longitude of the robot is calculated by the GNSS which stands for global navigation satellite system. One of the very few inputs that the robot would require is the destination of the journey to which the robot has to travel. So this play a crucial role for the overall objective of the project

6.6.1 User Interface

The current model requires the coordinates to be mannually fed into the code. The coordinates are kept as a refrence point and the code waits for difference between the current position and the refrence point to become lesser than a specific threshold after which it gives a stop command.
This value can be uploaded to be drawn on a map in real time, but for now, the code just sends the stop command whenever it reaches it specific positon.

To test the working of the GPS, we stored the coordinates in a file and then later uploaded them on an online map to see if it was returning coordinates properly. The figure below shows the testing of the GPS module on a path outside our department.

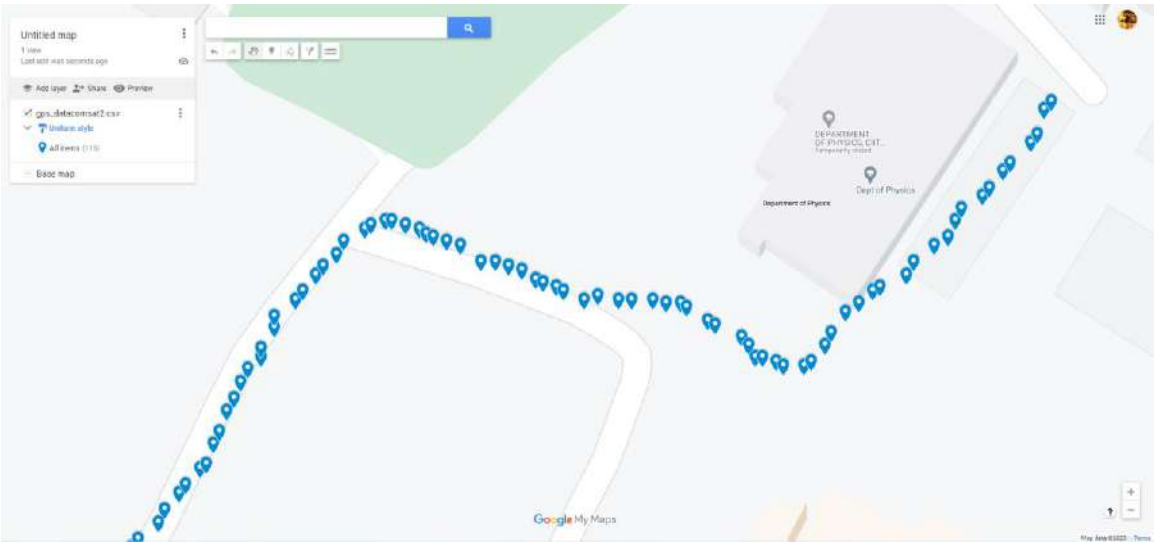

Figure 6-7 GPS Testing

Chapter: 07

Conclusion and Future Work

7.1 Conclusion

In conclusion, our prototype for a self-driving robot includes lane and object detection along with GPS functionality and a few ultra sonic sensors for safety. The project is a motivation for introduction of autonomous vehicles in many sectors like food delivery and shuttle services. These vehicles will not only reduce the human costs but also have the potential to be safer in terms of reduced human error.

A variety of hardware and software components were utilised in building this project. A micro controller which controls the motors and peripherals like sensors and GPS module and an on-board computer which processes the real time video stream and calculates the deviation of the robot from the centre of the path. The robot is fully equipped with the right components to follow a desired path with safety precautions and stop wherever it feels a threat. The destination of the journey can be fixed by a user input of GPS coordinates.

The robot has been made to ensure as much safety and reliability as possible under its limited ability considering the cost constraints. The safety of the robot can greatly be increased with further efforts and integration of more sophisticated hardware like depth sensing cameras. The current model uses a single camera which estimates the distance of the object. This is not a very reliable approach, but it is the best thing that can be done using a single camera. So, the project has a lot of areas of improvement needed in this project to be able to fulfil this aim by making the robot more accurate and reliable.

7.2 Future Work

7.2.1 Mobile Application to send GPS Coordinates and plot real time data on map.

A mobile application must be built which allows users to pass GPS coordinates in GUI environment. For now, the GPS coordinates are defined manually inside the python script which is not a very feasible approach. A user-friendly application will enable the user to easily pin a location on the map

which will enter the coordinates in the program. The application will also be able to receive real time data from the GPS about the current location and plot it on a map so that the user always knows about the journey and position of the robot.

7.2.2 Integration of AI in Path Detection

Another potential future work could be the integration of AI in path detection to avoid noise being recognized as a path. Currently, the robot's path detection is based on color segmentation and edge detection, which can be susceptible to noise in the image. By integrating AI techniques such as deep learning, the robot could learn to distinguish between noise and actual paths, improving the accuracy and reliability of its navigation. This would require collecting and labelling a large dataset of images with both path and noise instances and training a deep neural network to classify them accurately.

7.2.3 Adding a depth sensing camera for better safety and accuracy

Since the method currently used to calculate the distance includes a single camera and estimated results. A special hardware used in this regard can greatly enhance the reliability of the system. This can be done with special distance measuring cameras or Lidar sensors that calculate the distance in a much accurate manner without the limitations we faced when using a single camera.

References

[1] A. Mogaveera, R. Giri, M. Mahadik and A. Patil, "Self-Driving Robot using Neural Network," 2018 International Conference on Information , Communication, Engineering and Technology (ICICET), 2018, pp. 1-6, doi: 10.1109/ICICET.2018.8533870.

[2] Stelian-Emilian Oltean, Mobile Robot Platform with Arduino Uno and Raspberry Pi for Autonomous Navigation, Procedia Manufacturing, Volume 32, 2019, ISSN 2351-9789, https://doi.org/10.1016/j.promfg.2019.02.254.
(https://www.sciencedirect.com/science/article/pii/S2351978919302896)

[3]Hironobu Fujiyoshi, Tsubasa Hirakawa, Takayoshi Yamashita, Deep learning-based image recognition for autonomous driving, IATSS Research, Volume 43, Issue 4, 2019, ISSN 0386- 1112, https://doi.org/10.1016/j.iatssr.2019.11.008.
(https://www.sciencedirect.com/science/article/pii/S0386111219301566)I.

[4] H. Wu, Y. Liu and J. Rong, "A Lane Detection Approach for Driver Assistance," 2009 International Conference on Information Engineering and Computer Science, Wuhan, China, 2009, pp. 1-4, doi: 10.1109/ICIECS.2009.5365404.

[5] [4] M. Lee, K. Y. Han, J. Yu, and Y.-S. Lee, “A new lane following method based on deep learning for automated vehicles using surround view images,” Journal of Ambient Intelligence and Humanized Computing, 2019. [176]

Bibliography

https://theailearner.com/tag/cv2-warpperspective/

https://www.vojtech.net/posts/intro-convolutional-neural-networks

https://ctms.engin.umich.edu/CTMS/index.php?example=Introduction&section=ControlPID

https://images.app.goo.gl/M5aCizQD1TxgABZy5

https://images.app.goo.gl/U2YeVtFzEmBNoYEV7

https://images.app.goo.gl/Z4yZVS5VHaQk88Rb8

https://images.app.goo.gl/PWHXKjzGkhpuVRyG8

https://images.app.goo.gl/hU62WbakKsuGXEDz6

https://images.app.goo.gl/Hn1TrHaD3EiA1WiFA

https://images.app.goo.gl/UbJQiAeUTZLWezw77

https://images.app.goo.gl/4pMr2wZV3Ab519pf9

https://arnab.org/blog