**FIELD ROVER: Autonomous Food Delivery Vehicle for Open Spaces**

By

**Md. Rifat Murshed**

Roll: 1907018

&

**Md. Sunzidul Islam**

Roll: 1907079



**Supervisor:**

Md. Badiuzzaman Shuvo

Lecturer

Dept. of Computer Science and Engineering Signature

Khulna University of Engineering & Technology

**Department of Computer Science and Engineering**

**Khulna University of Engineering & Technology**

**Khulna 9203, Bangladesh**

**November 2023**

**Acknowledgements**

First and foremost, we must be grateful to and wish to acknowledge my insightful indebtedness to Md. Badiuzzaman Shuvo, Lecturer of the Department of Computer Science and Engineering and the supervisor of the project. His unfathomable knowledge in this field influenced me to carry out this project up to this point. His endless endurance, scholarly guidance, continual encouragement, constant and lively supervision, constructive criticism, and priceless suggestions made it possible to come up to this phase. Without his inspiration, enthusiasm, and encouragement, this work could not have been completed. Last, but by no means least, we thank Allah for the talents and abilities we were given that made it possible to undertake this project.

**Authors**

**Abstract**

This project focused on building a GPS controlled 4-wheel autonomous robot. It is a self-guided autonomous robot which can be maneuvered with the help of GPS module and compass together interfaced with the microcontroller Arduino Uno. The 4-wheels of the robot are interfaced with motor driver and then connected to the Arduino Uno. The speed of the robot is controlled using PWM signals sent from the Arduino board. When the robot starts, it locates its current position using the GPS module. Once the current location is fixed, it calculates the distance and heading between the two points. The compass module tells the current heading of the robot. The final heading is calculated by taking the difference between actual heading and current heading. With the help of final heading angle, the robot moves towards its desired location. As the robot moves close to the destination the distance reduces. The purpose of building this robot was to guide the robot to multiple locations autonomously with the destination locations predefined in the algorithm. To maneuver the robot to the multiple locations it is very important to calculate the accurate distance and heading. For this project, the main task was to design an algorithm that can calculate the exact distance between any two locations and guide the robot in the proper direction. The algorithm designed was able to fulfill these tasks and guided the robot to multiple locations and reach the final destination.

**Contents**

|  |  |  |
| --- | --- | --- |
|  |  | **Page** |
| Acknowledgement |  | ii |
| Abstract |  | iii |
| Contents |  | iv |
| List of Tables |  | vi |
| List of Figures |  | vii |

|  |  |  |
| --- | --- | --- |
| **1** | **Introduction** | 1 |
|  | 1.1 Background | 1 |
|  | 1.2 Objectives | 1 |
|  | 1.3 Scopes | 1 |
|  | 1.4 Project Planning | 2 |
|  | 1.5 Unfamiliarity of the problem | 3 |
| **2** | **Related Work** | 3 |
|  | 2.1 Related Works | 3 |
|  | 2.2 Discussion of research gap solution | 4 |
| **3** | **System Design** | 3 |
|  | * 1. Analysis of the system  |  |  | | --- | --- | | 3.1.1 | Circuit Diagram of the system | | 3  4 |
|  | * 1. System architecture   2. Tools used  |  |  | | --- | --- | | 3.3.1 | Arduino Uno R3 (ARD-00028) | | 3.3.2 | GPS Module U-Blox Neo M8N | | 3.3.3 | HMC5883L electronic compass | | 3.3.4 | Motor Driver L298N | | 3.3.5 | Wheels (130\*60 mm) | | 3.3.6 | Lipo Battery (12V) | | 3.3.7 | HC-05 Bluetooth Module | | 3.3.8 | Approximate Robot Model | | 3.3.9 | Arduino IDE | | 5  7  7  8  8  10  10  10  11  11  12 |
| **4** | **Project Implementation** | 12 |
|  | 4.1 System implementation  4.1.1 Interfacing Arduino Uno and GPS Module  4.1.2 Implementation of Complete Logic to Drive the Robot  4.1.3 Distance Measurement using Haversine Formula  4.1.4 Heading Angle | 12  12  14  15  16 |
|  | 4.2 Morality or Ethical issues | 17 |
|  | 4.3 Socio-economic impact and sustainability | 18 |
|  | 4.4 Financial analyses and budget | 18 |
| **5** | **Conclusions** | 19 |
|  | 5.1 Conclusion and challenges faced | 20 |
|  | 5.2 Future work | 20 |
|  |  |  |
|  |  |  |
|  | **References** | 21 |
|  |  |  |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| **Table No.** | **Description** | **Page** |
| 3.1 | Technical Specifications of Arduino Uno R3 | 7 |
| 3.2 | General specification U-Blox Neo M8N | 8 |
| 3.3 | Technical Specifications of Electronic Compass | 9 |
| 3.4 | Technical Specifications of Motor Driver | 10 |
| 4.1 | Comparison of Haversine and Normal Distance Formula | 16 |
| 4.2 | Financial analysis and budget of the project | 18 |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Description** | **Page** |
| 1.1 | Gantt Chart of Autonomous Robot System Project | 2 |
| 3.1 | Use Case Diagram | 4 |
| 3.2 | Circuit Diagram for Robot | 5 |
| 3.3 | Flowchart for running a robot | 6 |
| 3.4 | Arduino Uno | 7 |
| 3.5 | GPS U-Blox Neo M8N | 8 |
| 3.6 | HMC5883L electronic compass | 9 |
| 3.7 | Output from the Compass module | 10 |
| 3.8 | Motor Driver L298N | 10 |
| 3.9 | Wheel (130\*60mm) | 11 |
| 3.10 | Lipo Battery | 11 |
| 3.11 | HC-05 Bluetooth Module | 11 |
| 3.12 | Target Robot Model | 13 |
| 4.1 | Interfacing of GPS module with Arduino board | 14 |
| 4.2 | Output from the GPS module | 15 |
| 4.3 | Azimuth Angle | 17 |

1. Introduction
   1. Background

In recent years, the field of robotics has witnessed a surge in the development of autonomous systems, ranging from drones to ground-based vehicles. The integration of Global Positioning System (GPS) technology with robotics has opened up new possibilities for creating self-guided and location-aware machines. The project at hand is a testament to this technological fusion, focusing on the creation of a 4-wheel autonomous robot capable of navigating to predefined destinations using GPS coordinates.

* 1. Objectives

The projects major purposes or objectives are to improve the accurate distance calculation, the accuracy of the GPS signals and add a new solution to the limitations.

* Development a robust interface between the GPS module and Arduino Uno microcontroller.
* Connecting and configuring a compass module to the Arduino Uno to obtain real-time orientation data.
* Design and implementation an autonomous navigation algorithm that utilizes GPS and compass data.
* Implementation of decision-making logic to guide the robot along optimal paths, considering obstacles as needed.
  1. Scopes

The successful implementation of a GPS-controlled 4-wheel autonomous robot opens up several promising scopes across various domains. Autonomous robots equipped with GPS navigation can be employed in a closed surface for efficient emergency food and medicine supply. It also reduces the need for manual labor. These robots can patrol autonomously and covering designated paths and providing real-time data. Autonomous mobile robots are self-operated vehicles that do not require any command from the operator. The movement is defined before the robot starts and it navigates according to the predefined coordinates. These robots have various applications.

* 1. Project Planning

Here’s a sample dialogue for the initial stages of planning a GPS-controlled autonomous robot project:

**Figure 1.1:** Gantt Chart of Autonomous Robot System Project

* 1. Unfamiliarity of the problem

The current project introduces aspects that were not covered in my previous course, specifically focusing on GPS navigation systems using a GPS module and compass. In this context, I've acquired knowledge about setting GPS localization and implementing the Haversine formula, providing a practical understanding of its functionality. Additionally, the project has enabled me to delve into calculating the heading angle, a topic that was not discussed in my previous coursework. This hands-on experience is proving valuable, allowing me to bridge gaps in my previous knowledge and gain a comprehensive understanding of GPS-based navigation systems.

1. Related Work
   1. Related works

Navigation systems are concerned with monitoring and controlling the movements of a vehicle or craft in physical space. To track the route, the navigation system utilizes waypoint information, as presented by Millington et al. [2]. In their study, waypoint information, consisting of coordinates, is employed for the navigation system in a vehicle. The coordinates are used to define the route, and both the waypoints information and the vehicle's position are displayed on the screen with a feedback system to control the vehicle's movement directions.

The use of the Global Positioning System (GPS) for creating waypoints, a set of coordinates in physical space for navigation systems, has proven to be very effective. In GPS-based navigation systems, the vehicle or craft's trajectory is determined through a series of waypoints, followed by navigating to the next waypoints until the destination is reached, as demonstrated in [3,4]. Hoi et al. [3] presented an autonomous mobile robot using GPS, where the robot follows its trajectory with feedback through the GPS receiver and evades obstacles with the help of photosensors. Additionally, one can track a mobile robot with a wireless RF communication module. Hamid et al. [4] presented another implementation of a navigation system in a mobile robot using GPS for navigation and sonar sensors for obstacle avoidance. The noteworthy aspect of this work is the utilization of the command loop daisy-chaining application method. Similar work is presented by Sukkarieh et al. [5] for land vehicle applications. Their work combines the inertial measurement unit with the global positioning system to enhance the integrity of navigation. The system considers both low-frequency faults and high-frequency faults in the inertial measurement unit and GPS, both during and before the fusion of the inertial measurement unit and GPS. Bruch et al. [6] present a land-based navigation system for vehicles implemented on the Man-Portable Robotic System Urban Robot. This navigation system utilizes a combination of a Kalman filter, waypoints, some inexpensive sensors, and GPS, and is implemented on an embedded processor.

* 1. Discussion of research gap solution

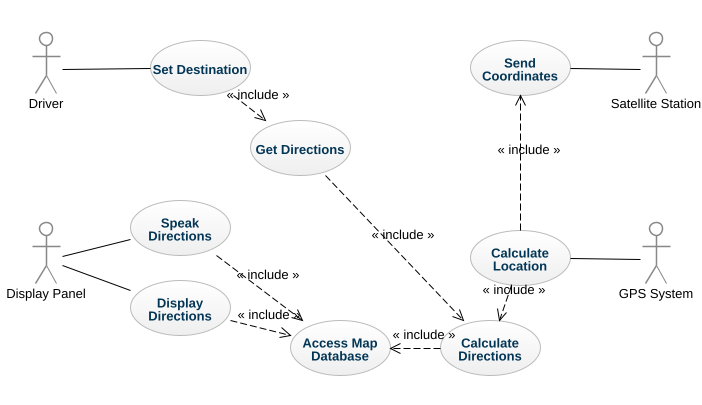
In the previous development of this robot, there were issues with distance calculation and the accuracy of GPS signals. Users faced challenges in obtaining the actual distance between two points and in receiving precise GPS signals. Consequently, there was a risk of misguiding the robot in unintended directions.

1. System Design

The system was designed initially with in mind the fact that it can be scalable and adaptable by the common people.

* 1. Analysis of the system

The overarching aims of constructing this autonomous robot was to empower it to navigate autonomously across multiple locations, each predetermined within the algorithm.



**Figure 3.1**: Use Case Diagram

* + 1. Circuit Diagram of the system

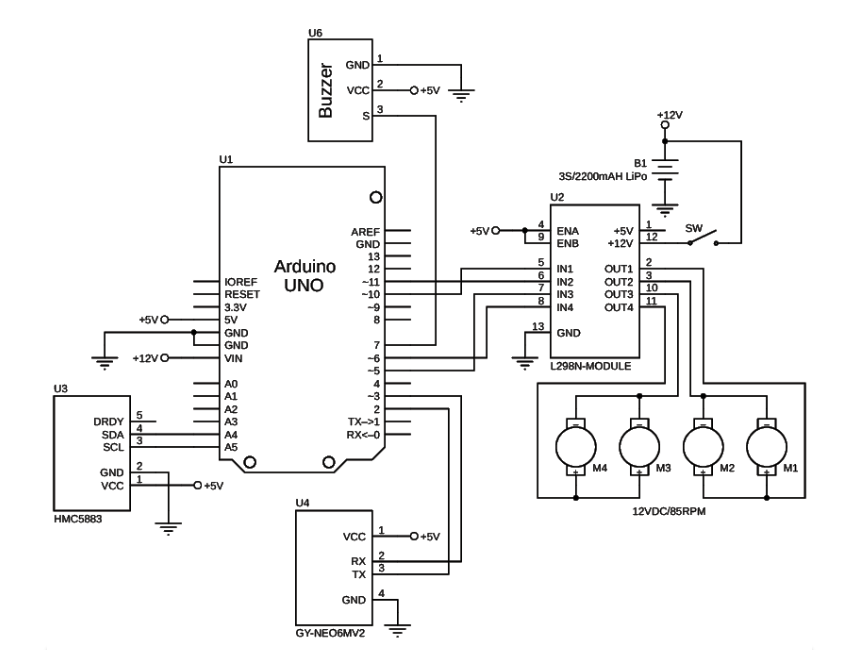


Figure 3.2: Circuit Diagram for Robot

* 1. System architecture

The below flow chart explains the operation of the algorithm designed for the robot. Starting from the top is the initialization section for motor pins, GPS module pins and the declaration of some variables that are used in the later part of the algorithm. Initially, when the Arduino starts the GPS module, compass and monster moto shield is powered ON. The GPS module locates the current position of the robot. The current location is stored in two variables as latitudes and longitudes. The destination coordinates are defined in the algorithm. Now the algorithm calculates the distance between the current location and destination using the distance formula. Further, the current heading of the robot is located using the compass module and also the heading from a current location to the destination is calculated using heading formula.

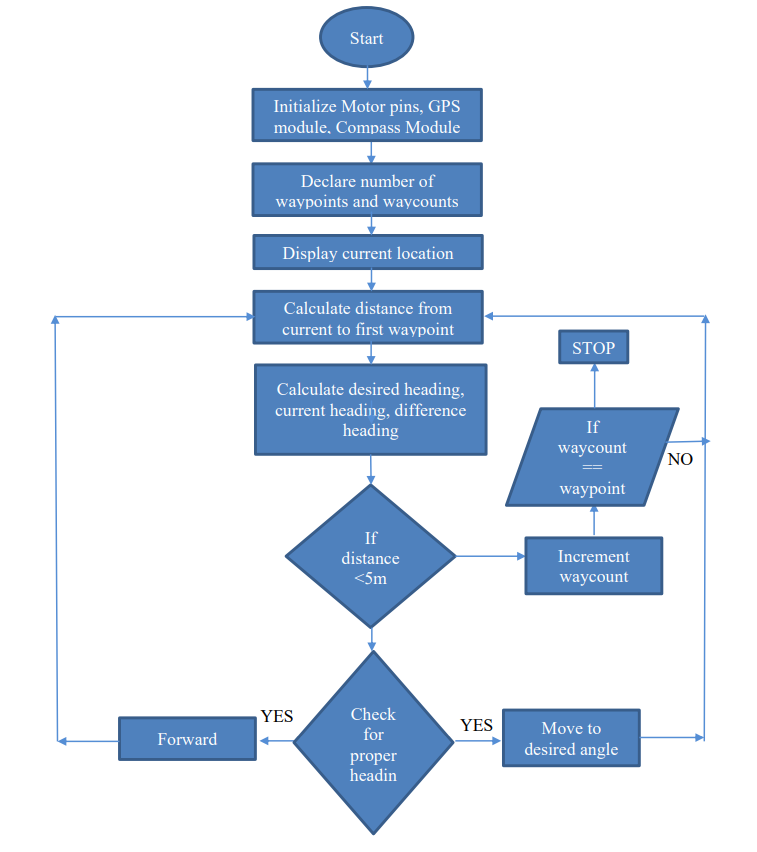


Figure 3.3: Flowchart for running a robot

In order to drive the robot in the desired direction, the difference of the current heading and the formula heading is calculated. Once these calculations are done the robot starts to move towards the first location given in the algorithm. While traveling towards the destination location the GPS module continuously updates the location and also the overall process is done till the robot reaches close to the destination. In the algorithm, the minimum distance condition of 5 meters is defined which is checked every time a new location is found. If this condition is satisfied it means that the robot has reached its first destination location. Now the counter is incremented and again the same process is repeated. Once the counter reaches the predefined value it means that the robot has reached to its final destination and it slowly stops. The advantage of this algorithm is that it calculates the accurate distance between ant two coordinates. As per the working of the robot, it is observed that the accuracy of the robot is 5 to 6 meters which is acceptable for this project.

* 1. Tools used

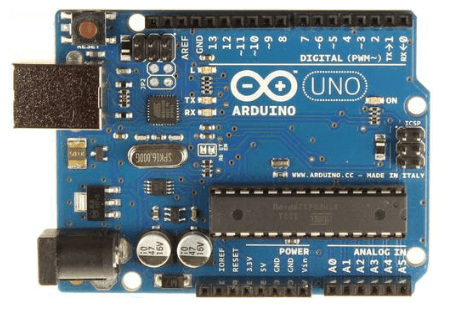
The project employs various essential tools to facilitate its development. The primary tools include:

* + 1. Arduino Uno R3 (ARD-00028)

Arduino UNO is based on an ATmega328P microcontroller. The Arduino UNO includes 6 analog pin inputs, 14 digital pins, a USB connector, a power jack and an ICSP (In-Circuit Serial Programming) header. It is programmed based on IDE which stands for Integrated Development Environment. The main technical specifications of Arduino Uno R3 are:

Table 3.1: Technical Specifications of Arduino Uno R3

|  |  |
| --- | --- |
| Microcontroller | ATmega328P |
| Operating Voltage | 5V |
| Input Voltage | 7-12V |
| Input Voltage (limit) | 6-20V |
| DC Current per I/O Pins | 20 mA |
| DC Current for 3.3V Pin | 50 mA |



**Figure 3.4:** Arduino Uno

* + 1. GPS Module U-Blox Neo M8N

The module in question is equipped with four essential pins, each serving a distinct purpose in its operation. The VIN pin functions as the power supply input, requiring a stable 5-volt source to ensure optimal module performance. Grounding is facilitated through the GND pin, providing a reference point for electrical potential. The RX pin is designated for receiving data via the serial protocol, enabling the module to accept incoming information. Conversely, the TX pin is dedicated to sending data via the serial protocol, facilitating the transmission of information from the module to external devices or systems. This pin configuration establishes a streamlined and efficient communication system, allowing the module to interface seamlessly with other components in the electronic setup. The general specifications of this modules are:

|  |  |
| --- | --- |
| Input Supply Voltage | 3-5V |
| Boot Time | 1S |
| Acceleration(g) | <4 |
| Maximum Attitude | 18000 |
| Capture Time | 0.1s Average |
| Receiver Type | 72-channel U-blox M8 engine |

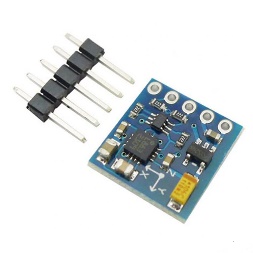
**Table 3.2:** General specification U-Blox Neo M8N

****

**Figure 3.5:** GPS U-Blox Neo M8N

* + 1. HMC5883L electronic compass

In order to guide the robot in the desired direction, it is essential to first know which direction the robot is currently heading. To know its current heading compass is one of the options. For this project HMC5883L Module Compass is used as shown in figure 3. This compass is designed for low-field magnetic sensing with a digital interface. This compass is precise in-axis sensitivity and linearity. It is capable of measuring the direction and the magnitude of Earth’s magnetic fields from milli-gauss to 8 gausses**.**

****

**Figure 3.6**: HMC5883L electronic compass

The output from the compass module is shown in figure 3.6:

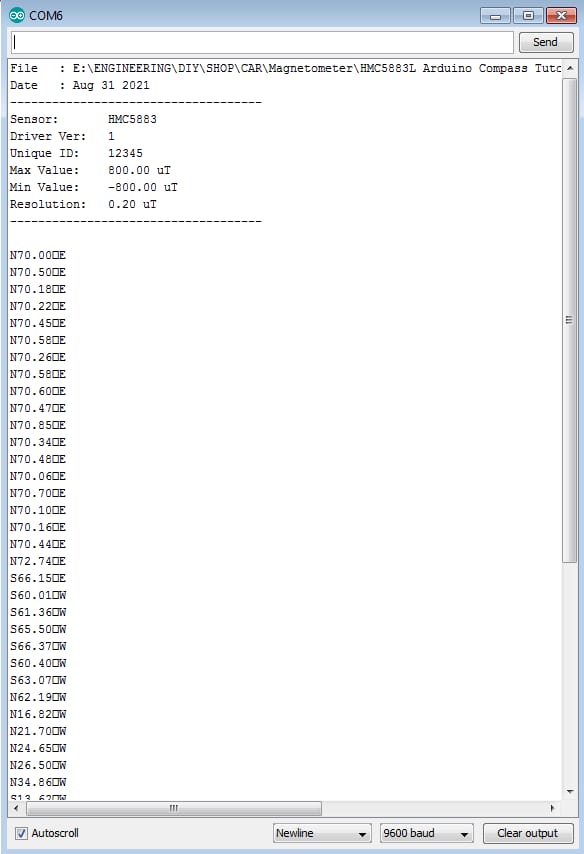


Figure 3.7: Output from the Compass module

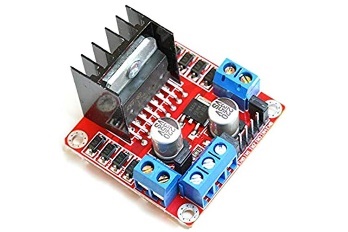
|  |  |
| --- | --- |
| Operating Voltage | 2.16 to 3.6V |
| Supply Current | 100 μA |
| Communication | I2C interface |
| Maximum Output Rate | 160 Hz |
| Resolution | 5 milli-gauss |

The technical specifications of the compass are listed in below:

**Table 3.3.** Technical Specifications of Electronic Compass

* + 1. Motor Driver L298N

Thisis a high powerful motor driver module for driving DC and Stepper Motors. This module consists of an L298 motor driver IC and a 78M05 5V regulator. **L298N Module** can control up to 4 DC motors, or 2 DC motors with directional and speed control.

****

**Figure 3.8:** Motor Driver L298N

|  |  |
| --- | --- |
| Maximum Voltage | 30V |
| Maximum Current | 2Amp |
| Continuous Current Supply | 2Amp |
| Maximum PWM Frequency | 40 kHz |
| MOSFET on-resistance | 19 mΩ (per leg) |

**Table 3.4:** Technical Specifications of Motor Driver

* + 1. 4 Wheels (130\*60 mm)

These wheels are made of soft black rubber with deep tread and spikes for traction. They can be mounted to any motor with a 4mm output shaft. The dimensions indicate a wheel diameter of 130 mm and a width of 60 mm, providing a balanced combination of stability and surface contact.



Figure 3.9: Wheel (130\*60mm)

* + 1. Lipo Battery (12V)

A 12V 5000mAh LiPo (Lithium Polymer) battery is a compact and high-capacity power source commonly used in various electronic devices and projects.

  
 Figure 3.10: Lipo Battery

* + 1. HC-05 Bluetooth Module

The HC-05 Bluetooth module is a versatile and widely used component for wireless communication in electronic projects. Operating on Bluetooth 2.0, it supports both Master and Slave modes, allowing for flexible device connections.

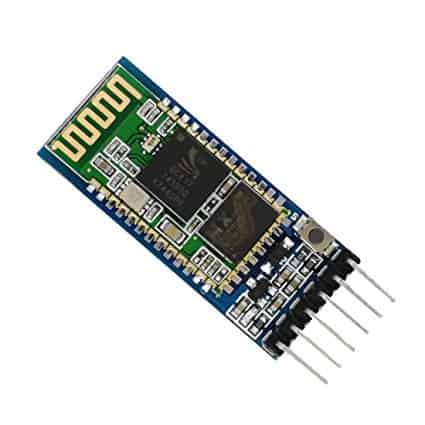


Figure 3.11: HC-05 Bluetooth Module

* + 1. Approximate Robot Model

This is a demonstration model of our robot. Since the hardware integration is still in progress, we aren't able to provide a real image at this time.



Figure 3.12: Target Robot Model

* + 1. Arduino IDE

The Arduino Integrated Development Environment (IDE) is a user-friendly and open-source software platform designed for programming Arduino microcontrollers. Its simple interface includes a code editor with features like syntax highlighting, auto-indentation, and integrated examples for various functionalities. The IDE supports the C and C++ programming languages and provides tools such as a Library Manager.

1. Project Implementation

This part is most important for the robot to function and demonstrate desired output.

* 1. System Implementation

For the implementation of the autonomous robot system, the Arduino IDE served as the foundation for programming the microcontroller. The control logic and algorithms were coded using the Arduino programming language. The motor drivers and sensor interfaces were configured within the Arduino environment. Additionally, a GPS module and compass were integrated into the system, with the GPS providing location data and the compass aiding in determining the robot's heading. The entire system's firmware was developed within the Arduino IDE to ensure seamless communication and coordination between the various hardware components.

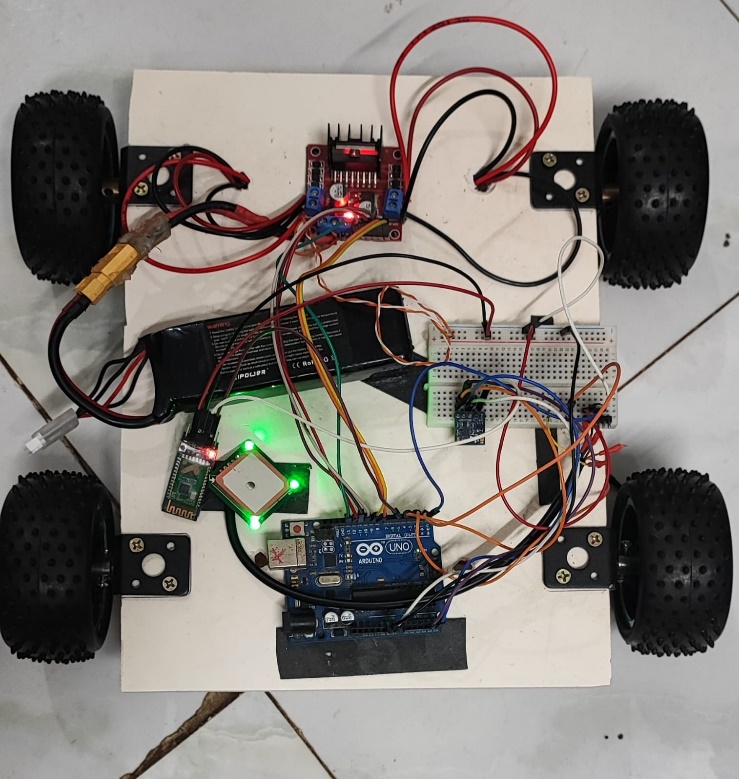
* + 1. Interfacing Arduino Uno and GPS Module

In this part, the GPS module is first interfaced with Arduino Uno is shown in the figure 9 [13]. It is essential to know the connection pins used for the serial communication. TinyGPS library is used to test the module and check whether the coordinates are being transmitted to the Arduino board. The sample program is written in Arduino IDE software using TinyGPS library. The pin connections to connect GPS module with Arduino Mega are given below.

1. The GND pin of GPS module is connected to any GND pin in Arduino board.

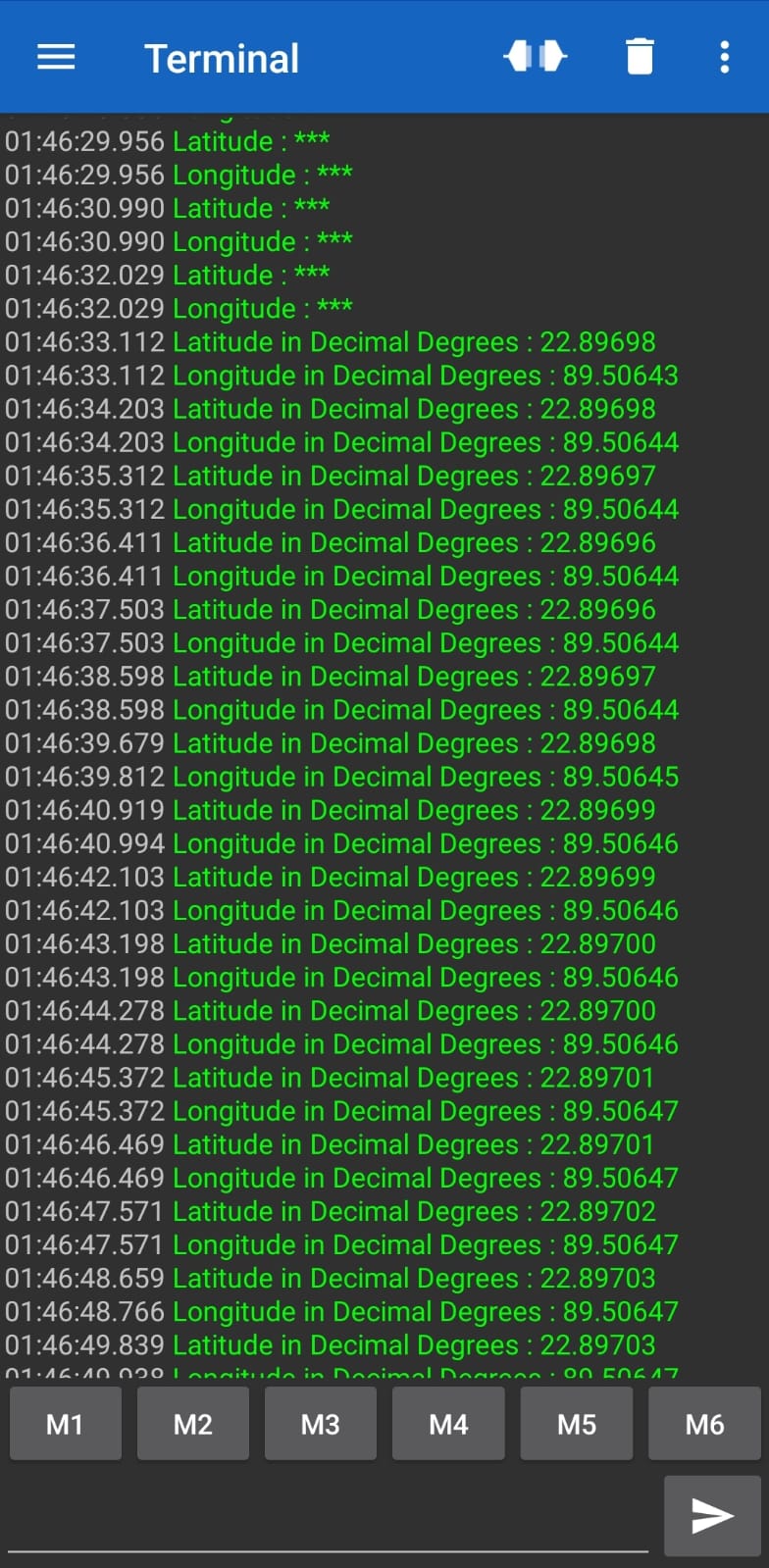
2. The GPS module needs 3.3V which is connected to 3.3V pin on Arduino.

3. The TX pin of GPD module is connected to pin 48 on Arduino board.



**Figure 4.1**: Interfacing of GPS module with Arduino board (Primary Level)

Figure 4.1 shows the interfacing of GPS module with Arduino Uno in primary level. The pins of the GPS module are connected to the respective pins of Arduino Mega. There is an antenna connected to GPS module which is used for the good reception of signals. Once the Arduino board is connected to a computer it powers ON. The GPS module also turned ON which is indicated by an LED on the module. After the LED is turned ON it remains ON for a long time. This indicated that the GPS module is acquiring signals from the satellite. In the beginning, when the GPS module is turned ON for the first time it takes a lot of time to acquire signals from the satellite. After some time, the LED starts to blink which means it is receiving signals from the satellite. The output is shown in the form of latitudes and longitudes which can be seen on the serial output of the Arduino IDE software. Figure 4.2 shows the output of the GPS module. It shows the location of the robot in terms of longitudes and latitudes. The output from the GPS module is imported to an online GPS visualizer to track the coordinates.



**Figure 4.2:** Output from the GPS module

* + 1. Implementation of Complete Logic to Drive the Robot System implementation

After working on the GPS module and developing a sample code it was clear that the GPS module is acquiring signals from the satellite. The code is modified to display the latitudes and longitudes of the location. Now to drive the robot to the target it was necessary to calculate certain mathematical parameters. The target location is given in the algorithm as latitudes and longitudes. After the current location is fixed the algorithm will calculate the distance between the current and target location. The algorithm will also calculate the heading needed to reach the target. The compass module will tell the current heading of the robot. A difference in the heading will be calculated by comparing the heading from the compass module and from the calculations. The resultant heading will be the final heading on which the robot should proceed. After the heading is decided, the motor drives the robot to the destination location. Since the GPS module has to update the current information of the robot it was necessary to keep the speed of the robot slow. As the robot moves closer the GPS module will update the location and the distance will decrease. The formula for calculating the distance is very accurate for shorter distances.

* + 1. Distance Measurement using Haversine Formula

For a robot to how much to travel it is necessary to calculate the distance between the current position and the target. For this project, haversine distance formula is being implemented [16]. The significance of this formula is it calculates spherical distance on earth using trigonometric functions. On a spherical surface, the shortest path between two points is along an arc of a great circle. It is a circle drawn on earth with the same radius as that of the earth. Any two points that lie on a unique great circle divide the arc into two arcs. The shortest path between the points is along the shortest arc among the two arcs. The haversine function is defined as below.

ℎ𝑎𝑣𝑒𝑟𝑠𝑖𝑛(𝜃) = 𝑠𝑖𝑛2 (𝜃 /2)

The haversine distance formula is given as follows.

ℎ𝑎𝑣𝑒𝑟𝑠𝑖𝑛 (𝑑/2𝑅) = ℎ𝑎𝑣𝑒𝑟𝑠𝑖𝑛(∅2 − ∅2 ) + cos(∅1 ) cos(∅2 ) ℎ𝑎𝑣𝑒𝑟𝑠𝑖𝑛(𝜆2 – 𝜆1)

Solving for the value of d we get the distance formula-

𝑑 = 2𝑅 sin-1 (√𝑠𝑖𝑛2 (∅2−∅1/ 2) + cos(∅1) cos(∅2 ) 𝑠𝑖𝑛2 ( 𝜆2− 𝜆1/ 2 ) )

Here,

d is the distance between two co-ordinates,

R is the radius of earth i.e. 6371 km or 3961 miles

∅1, ∅2 are latitudes of point 1 and latitude of point 2

𝜆1, 𝜆2 are longitude of point 1 and longitude of point 2

Apart from the haversine formula there is also one formula to calculate the distance between any two points on the earth. The normal distance formula is given as follows.

𝑑 = 𝑅 ∗ √2

The haversine formula calculates the distance between any two points on earth, taking into consideration that the earth has a spherical surface. But the normal distance formula in 2 calculates the distance considering the earth as a flat surface without considering the spherical curvature of the earth. For small distances, the difference between these two formulas is very small as compared with the online results. But as the distance increases the difference between these two formulas increases and the normal distance formula gives inaccurate results. For a GPS guided robot getting accurate results in the range of 5-6 meters is very essential. Considering the accuracy haversine formula is preferred over normal distance formula for all ranges of distances. The comparison of the distance calculated using both haversine and normal distance formula is shown in table 4.1.

|  |  |  |  |
| --- | --- | --- | --- |
| Co-ordinates as Latitudes and Longitudes | Normal Distance Formula | Haversin Distance Formula | Actual Distance from Google |
| A 39.28357, 120.51933,  B 39.61731, 120.53251 | 37.1 km | 37.1 km | 37.1 km |
| A 42.69078, 119.08012  B 44.33957, 120.28325 | 207.4 km | 207.4 km | 207.4 km |
| A 42.59784, -91.21056  B 46.90922, -118.81658 | 2208.4 km | 2219.6 km | 2220.3 km |

**Table 4.1:** Comparison of Haversine and Normal Distance Formula

From the above table, it is clear that haversin formula gives more accurate results as compared to normal distance formula. So, haversin formula is being used for this project to calculate the distance between any two coordinates on the surface of the earth.

* + 1. Heading Angle

Heading angle is a term used to navigate the robot in the desired direction. It is the angle between two points and the robot as to proceed in that direction in order to reach the destination. It is a common term used for navigation in the field of aircraft or marine, vehicle navigation. Heading angle is defined as the north-south line on earth or the meridian and the line connecting the two points on earth. A compass is a component that gives heading information in order to reach the destination. For this project a forward azimuth formula is being implemented which trace a path along a great circle arc [17]. Below is the heading formula which will calculate the desired heading of the robot to reach the target point.

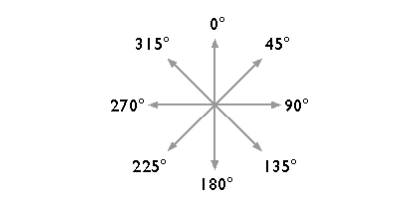
ℎ = 𝑎𝑡𝑎𝑛2(sin(𝜆2 – 𝜆1 ) cos(∅2 ) , cos(∅1) sin(∅2) − sin(∅1) cos(∅2) cos(𝜆2− 𝜆1))

Here, h is the heading,

∅1, ∅2 are latitudes of point 1 and latitude of point 2,

𝜆1, 𝜆2 are longitude of point 1 and longitude of point 2.

The above formula calculates the heading between the two points. The heading is calculated in radians, so it is converted into degrees by multiplying it with π/180. True heading is based on true north. Angles are measured clockwise from north. The range of angles is from 0 to 3600. After the heading is known it is compared with the current heading of the robot given by compass. The difference between the two angles will be the final heading in which the robot should proceed to reach the destination.



**Figure 4.3:** Azimuth angle

* 1. Morality or ethical issues

Developing a GPS-guided 4-wheel autonomous robot raises ethical concerns, particularly regarding privacy and data security. As the robot moves along predetermined coordinates, safeguarding location data and respecting privacy rights becomes essential. To address this, data security measures are crucial to prevent unauthorized access to GPS information. It's reassuring to note that the details used to build the robot are provided in the reference section, ensuring transparency in its development. Importantly, the project is committed to not compromising anyone's personal information, assuring future data security.

* 1. Socio-economic impact and sustainability

The deployment of a GPS-guided autonomous robot not only raises ethical considerations but also has potential socio-economic impacts and sustainability implications. The project aims to uphold data security and privacy, addressing concerns related to unauthorized access to GPS information. In terms of socio-economic impact, the robot's development may create opportunities for skills development and new jobs in fields such as maintenance and programming. Additionally, the commitment to transparency aligns with sustainable practices, fostering responsible technology development. Overall, the project seeks to balance technological advancements with ethical considerations, socio-economic well-being, and environmental sustainability.

* 1. Financial analyses and budget

Financial analysis and budget for this project involves the cost of development, equipment, implementation, maintenance and other relevant expenses. A general measurement of the budget has been estimated in following table 4.2

|  |  |
| --- | --- |
| **Types of cost** | **Budget(tk)** |
| Personnel Salaries | 1,00,000 – 1,50,000 |
| Hardware | 50,000 – 80,000 |
| Software | 10,000 – 40,000 |
| Development tools | 10,000 – 15,000 |
| Robot training | 10,000 – 20,000 |
| Robot management tools | 5,000 – 12,000 |
| Miscellaneous | 5,000 – 8,000 |
| Total: | 1,90,000 – 3,20,000 |

**Table 4.2:** Financial analysis and budget of the project

The overall budget is characterized by minimal external expenditures, with majority of costs being self-funded, utilization of available software and technologies.

1. Conclusion

The primary goal of this project was to build an autonomous robot that can move from one location to another with the help of GPS coordinates. The important task of this project was to calculate the accurate distance between any two points on the surface of the earth. Further implementing an algorithm that can store multiple coordinates which will guide the robot to reach the final location. These major tasks were not accomplished in the previous development of this project hence, it became very important to design a new algorithm to make the robot more robust and innovative from the previous one. Before implementing a new algorithm, it was very important to understand the working of all the components. At first, the GPS module was studied and interfaced with Arduino Mega. A sample code was written to check whether the GPS module is acquiring signals from the satellite. From the reading of the GPS module, it was observed that the accuracy of the GPS module is about 5 to 6 meters which is acceptable for this project. Next, the GPS module is interfaced with compass and the 6-wheel robot chassis and DC motors. The compass was used to know the current heading of the robot. The DC motors are high power motors so it was necessary to keep the speed of the motors very low for the proper working of the robot. After studying all the components individually, they are assembled together to make a complete robot. The output shows that the robot is able to gather all the data needed to reach destination location. The final output of this project is shown in a video.

* 1. Conclusion and challenges faced

In the development of the autonomous robot project, several challenges were encountered, analogous to the obstacles faced in the described case. Firstly, achieving accurate navigation and distance calculation posed a significant challenge, especially in environments with varying GPS signal precision. The refinement of algorithms to address these challenges, similar to the accurate extraction of facial features in the mentioned case, required extensive testing and adaptation to diverse scenarios. Despite these hurdles, the challenges in the autonomous robot project were addressed directly, mirroring the proactive approach taken in the described case, ultimately ensuring the success of the overall project.

* 1. Future Study

# In our future plans, we envision enhancing our facilities by incorporating Raspberry Pi and a webcam into our autonomous robot system. This strategic integration aims to equip the robot with advanced capabilities, enabling it to navigate roads safely and make informed decisions. The enhanced system will possess the ability to discern lanes, interpret traffic signals and ensure accident-free movement. Moreover, during various natural calamities, the robot will play a crucial role in delivering emergency medical supplies, extending its utility beyond routine operations. Our vision is to develop an autonomous robot that can not only accompany humans safely along roadways but also serve as a reliable resource for delivering essential items such as food, to specific destinations. This expansion aligns with our commitment to leveraging technology for the greater good, enhancing the robot's versatility and impact in various scenarios. At the end of the project, we will develop an app that can operate the system automatically.

# References

[1] T. Davies and A. Jnifene, “Path planning and trajectory control of collaborative mobile robots using hybrid control architecture,” Journal of Systemics, Cybernetics and Informatics, vol. 6, no. 4, pp. 42–48, 2008.

[2] M. Ibrahim, S. Ragavan, and S. Ponnambalam, “Way point based deliberative path planner for navigation,” in Advanced Intelligent Mechatronics, 2009. AIM 2009. IEEE/ASME International Conference on. IEEE, pp. 881–886.

[3] D. Mulvaney, Y. Wang, and I. Sillitoe, “Waypoint-based mobile robot navigation,” in Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on, vol. 2. IEEE, 2006, pp. 9063–9067.

[4] J. Latombe, “Robot motion planning. 1991,” Kluwer Academic Publishers, vol. 25, pp. 100–107.

[5] Y. Wang, D. Lane, and G. Falconer, “Two novel approaches for unmanned underwater vehicle path planning: constrained optimisation and semi-infinite constrained optimisation,” Robotica, vol. 18, no. 2, pp. 123–142, 2000.

[6] G. Raidl and B. Julstrom, “A weighted coding in a genetic algorithm for the degree-constrained minimum spanning tree problem,” in Proceedings of the 2000 ACM symposium on Applied computing-Volume 1. ACM, 2000, pp. 440–445.

[7] Z. Fu, A. Kurnia, A. Lim, and B. Rodrigues, “Shortest path problem with cache dependent path lengths,” in Evolutionary Computation, 2003. CEC’03. The 2003 Congress on, vol. 4. IEEE, pp. 2756–2761.

[8] J. Kuri, N. Puech, M. Gagnaire, and E. Dotaro, “Routing foreseeable lightpath demands using a tabu search meta-heuristic,” in Global Telecommunications Conference.