



WAYPOINT NAVIGATION

Internship Project

1. SAHIL SHARIEF — VU22EECE0100037
2. SARVANI RAPETI — VU22EECE0100043
3. SAMITA POTNURU — VU22EECE0100371



OBJECTIVE

The Objective of the project is to send the ground vehicle to a prefixed location when a single command is given to it from the mobile application through which we allow the user to control the ground vehicle.



Abstract

The Instructables guide on building a GPS(global positioning system) -guided robot outlines the process of creating a robot that autonomously navigates using GPS coordinates. The project uses a microcontroller (Arduino), GPS module, and motor controller to direct the robot's movements. With step-by-step instructions, the tutorial covers wiring, coding, and testing, allowing the robot to travel to predetermined waypoints based on GPS signals. It's an ideal project for hobbyists looking to learn about robotics, GPS technology, and autonomous systems



PROBLEM STATEMENT

The project addresses the challenge of efficiently controlling ground vehicles remotely by solving the problem of complex manual navigation, allowing users to autonomously send vehicles to a predefined location with a single command, reducing the need for continuous human control and intervention.



Challenge:-

Complex manual navigation of ground vehicles.

Solution:-

Autonomous navigation to predefined locations.

Benefit:-

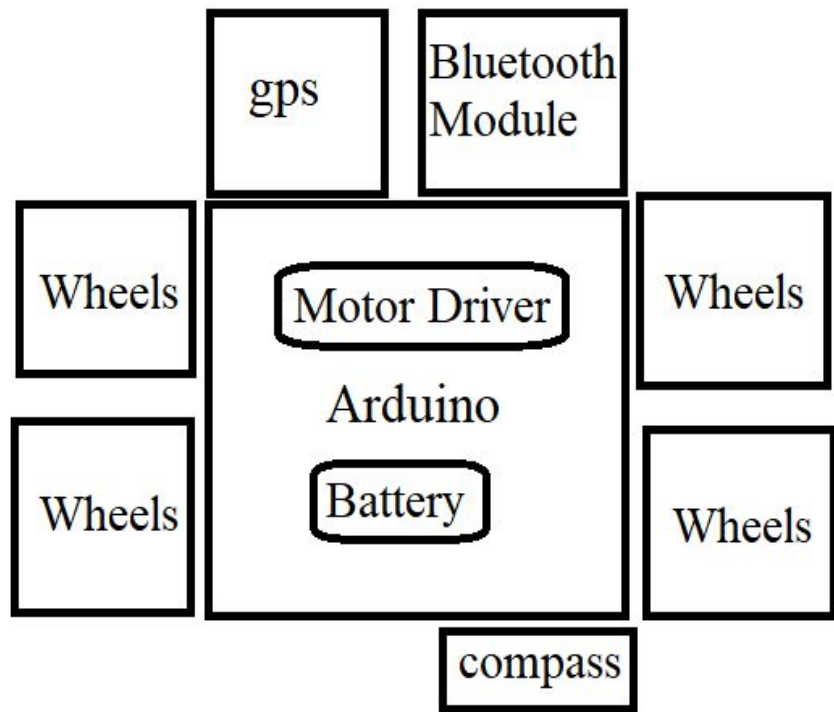
Reduced human control and intervention in vehicle operation.



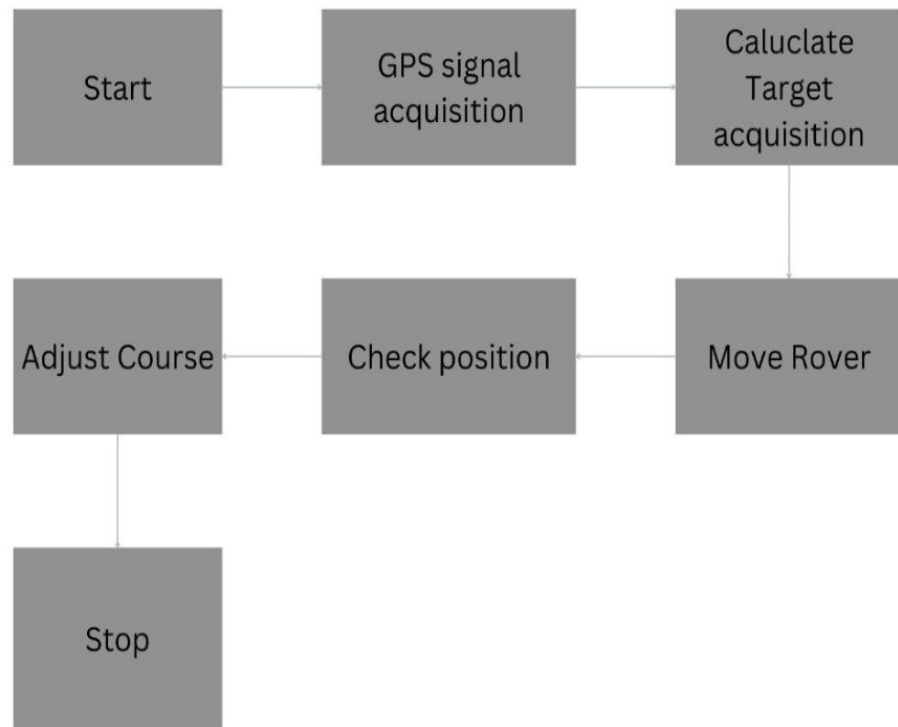
APPROACH USED

We used Arduino Mega 2560 to connect the GPS module , bluetooth module, motor drive and compass as the main components for the functionality of our project.

An application was developed using the MIT app inventor website and app to control the UGV(unmanned Ground Vehicle) from our mobile phone by connecting to the bluetooth module.



Schematic Diagram of Connections



Flow chart of Navigation

COMPONENTS USED



HARDWARE COMPONENTS:

Arduino mega 2560

Motor shield L293D

Bluetooth module

GPS Ublox Neo 6M

Compass HMC5883L

Dc Motors 4

Wheels 4

Battery and Jumper wires

SOFTWARE REQUIREMENTS:

Arduino IDE (required libraries based on the components)

MIT app inventor(website account and mobile app)

GPSRobot1_2

Screen1 ▾

Add Screen ...

Remove Screen

Project Properties

Publish to Gallery

Designer

Blocks

Palette

Search Components...

User Interface

Layout

Media

Drawing and Animation

Maps

Charts

Data Science

Sensors

Social

Storage

Connectivity

LEGO® MINDSTORMS®

Experimental

Extension

Viewer

☐ Display hidden components in Viewer

Phone size (320 x 505) ▾



All Components ▾

- divCompass
 - Label3
- HorizontalArrangemen
 - btnHeading
 - btnCalibrate
 - btnComps_Drive
- HorizontalArrangemen
 - btnPing
 - lblSpeed
 - btnSlider
- HorizontalArrangemen
 - Slider1
 - BluetoothClient1
 - Clock1
 - TextToSpeech1
 - NoBluetooth
 - Sound1

Rename

Delete

Media

Properties

txtbox_Window (TextBox)

▼ Appearance

BackgroundColor ⓘ

None

FontBold ⓘ

☐

FontItalic ⓘ

☐

FontSize ⓘ

15

FontTypeface ⓘ

default...

Height ⓘ

Automatic...

Width ⓘ

Fill parent...

Hint ⓘ

HintColor ⓘ

Default

TextAlignment ⓘ

center : 1 ▾

Blocks

- Built-in
- Control

Logic

Math

Text

Lists

Dictionaries

Colors

Variables

Procedures
- Screen1
- HorizontalArrangemen

btnConnectBT

btnDisconnectBT
- HorizontalArrangemen


Label4
- VerticalArrangement1

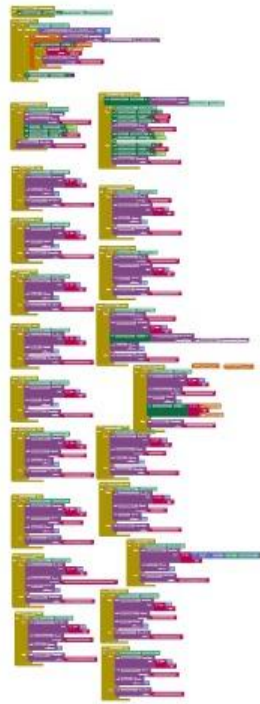
RenameDelete

Media

bluetooth_stom.png

Viewer









2

0

Show Warnings



Overall blocks in the MIT App



Stages of the project

Stage 1:- Bluetooth car

Initially a simple model where we control the car using the bluetooth module from the created app.

Stage 2:- app development

Making an interface for controlling the bluetooth car using the MIT app inventor.

Stage 3:- Attached the compass

Then compass is connected and calibrated and entered the offset values in the main code.

Stage 4:-GPS module

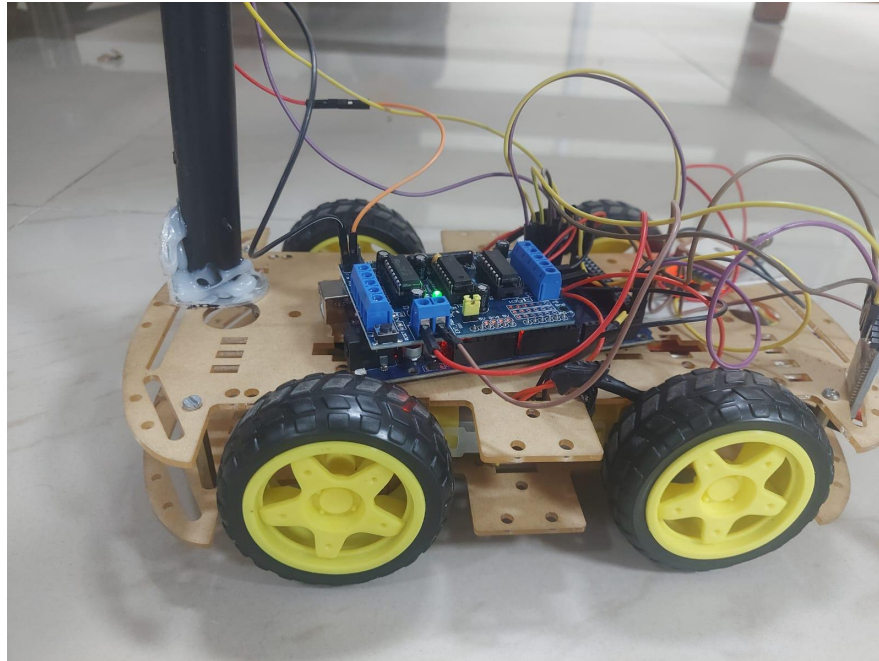
The GPS module is receiving and storing the latitude and longitude values for navigation.



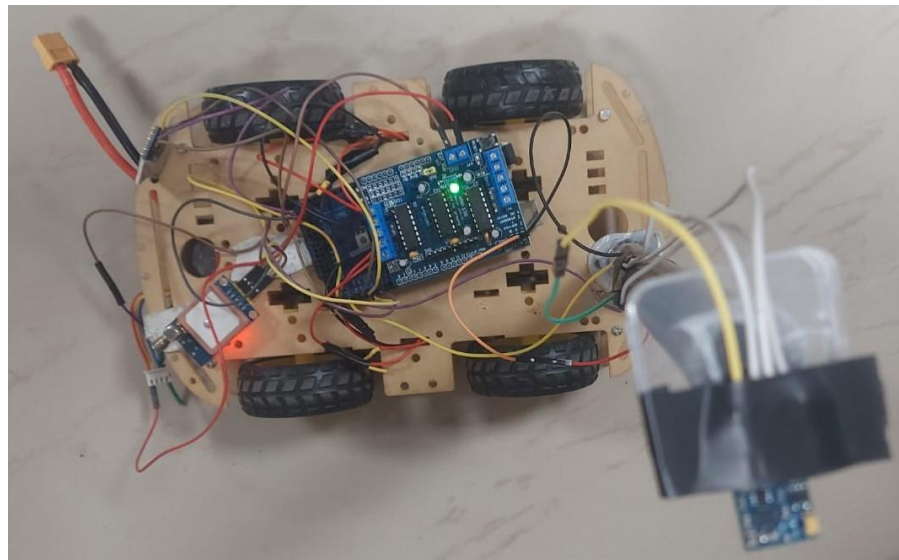
Methodology to go to the waypoint

1. **Initialization:** The robot's GPS module acquires satellite signals to get real-time coordinates.
2. **Path Calculation:** The microcontroller compares the current GPS coordinates with the predefined target location and computes the distance and direction.
3. **Movement:** Based on direction calculations, the robot moves forward, adjusts motor speeds, and steers accordingly using DC motors.
4. **Course Correction:** Periodically recalibrates its direction using updated GPS signals to ensure it stays on track.
5. **Final Destination:** The robot stops when it reaches the predefined GPS coordinates within a specified threshold range.

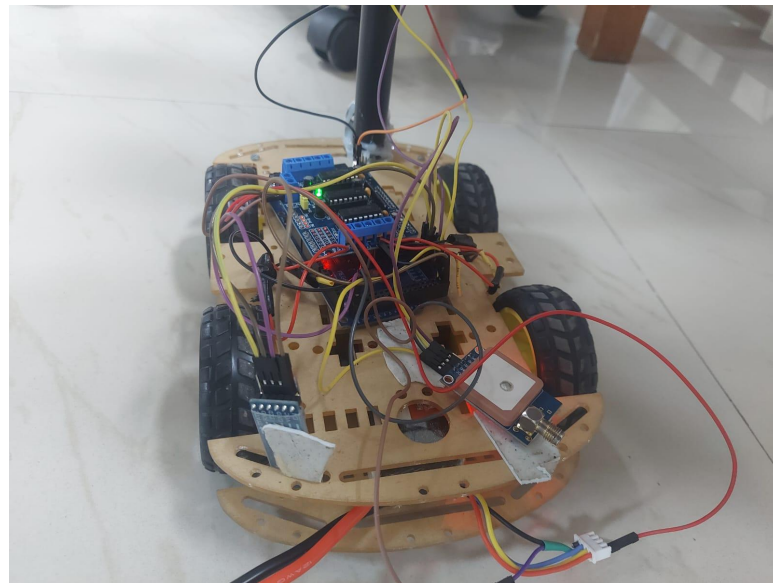
Project



Side view



Top View



Front View

App interface Look





Flow of project in the app

1. Connects to the Bluetooth
2. Uses the 'Set Way ' button in the app to fix a waypoint. We can provide upto 5 waypoints in the app. Then click on 'Done'.
3. Now when the UGV is away from the desired point. Click on the 'go to waypoint' to send it to that location.
4. If we give more than 1 waypoint , once the first one is reached click on the ' go to waypoint' to go to the next destination.



Weekly Report

S no.	Week	Work done
1.	1st Week	Learned about the hardware requirements , researched papers
2.	2nd Week	Integration of components
3.	3rd Week	Created an application in mobile phone using mit app inventor
4.	4th Week	Documentation and working of gps guided robot



Future challenges:

- Obstacle Detection and Avoidance
- Indoor Navigation
- Weather and Environmental Conditions
- GPS Accuracy and Reliability



CONCLUSION

We have successfully established the connections between the hardware components using the Arduino mega 2560. Also integrated the bluetooth module with the created application.

The unmanned Ground Vehicle (UGV) is reaching the prefixed waypoint while controlling through the app.

The background is a solid blue color with various white abstract elements. There are several thin, wavy white lines scattered across the frame. In the top left corner, there is a cluster of small white dots. In the bottom right corner, there is another cluster of small white dots. There are also several white geometric shapes, including triangles and circles, some of which are partially cut off by the edges of the frame. The text "THANK YOU" is centered in the middle of the image in a white, hand-drawn, slightly irregular font.

THANK YOU