GPS Based Smart CAB Dispatch System

Final Year Design Project Report

Submitted by

Abdul Haleem Abbasi	2010002
Irtsam Ghazi	2010143
Muhammad Rashid Maqbool	2010231
Muhammad Sharjeel Akhtar	2010242

Advisors

Dr. Zia-ul-Haq Abbas	Assistant Professor FEE
Dr. Suleman Mazhar	Assistant Professor FCS

Faculty of Electronic Engineering

Ghulam Ishaq Khan Institute of Engineering Sciences and Technology

May 2014.

ACKNOWLEDGEMENTS

First, we would like to thank ALLAH Almighty for bestowing us with knowledge, courage and hope to complete this project. It required hard work and perseverance that we were unable to achieve without the shadows of ALLAH's help.

Second, we would like to thank our professors who being the guidelines and support for us helped us in accomplishing the task at our hands. Our faculty advisor Dr. Zia-ul-Haq Abbas for being the complete support and motivation for us and helped us through every stage of our project. He made this project a success. We owe our entire project to his help and appreciate him for giving us so much time out of his hectic schedule. We all are so grateful to him.

We are indebted to Dr. Suleman Mazhar, Dr. Nisar Ahmed, Dr. Faisal Khan, Dr. Adnan Noor, Dr. Husn Ul Maab, Mr. Mazhar Javed, Mr. Ramiz Hayat, Mr. Nayyer Abbas, Mr. Ahmed Kamal, Mr. Mehran Bashir and all other faculty members for their valuable guidance on certain issues pertaining to the project.

Last but not the least we wish to thank our families for their constant support and encouragement.

ABSTRACT

The project aims at development of an autonomous control system for managing cabs within a city, It employs the use of GPS and GSM technology for monitoring vacant cabs within a city and making them available to the passenger in the shortest possible time. Initially a request is initiated by the user/passenger using an Android based Application, this app transfers the latitude and longitude of the user to the main controller using the GSM of the handset, the main controller is composed of GSM module which receives the request (coordinates) and serially communicates it to a PC. The main controller is also receiving the location of vacant cabs within the city. It will calculate the shortest possible distance between the passenger and cab. The controller also develops the route required by the cab to reach its passenger Along with this it also sends a confirmatory SMS to the passenger.

Such a system has a variety of applications and can be used for other purposes such as guiding a completely autonomous robot to distressed areas. Developing the prototype of such an autonomous robot is also part of the project. Furthermore its implementation to cab systems will substantially reduce the cab fare as the cab drivers will no longer have to waste fuel while searching for passengers.

List of Symbols and Abbreviations

GSM	Global System for Mobile
AGV	Automated Guided Vehicle
SMS	Short Message Service
GPS	Global Positioning System
GUI	Graphical User Interface
DOD	Department of Defense
App	Application
OS	Operating System
Hz	Hertz (Unit of Frequency)
AT	ATtention
USB	Universal Serial Bus
ICSP	In Circuit Serial Programming
SMT	Surface Mount Technology
NEMA	National Marine Electronics Association
GPGLL	Geographic position, latitude / longitude
I^2C	Inter-Integrated Circuit Communication
SCL	Serial Clock
SDA	Serial Data
ACK	Acknowledge
NACK	Negative Acknowledge
LSB	Least Significant Bit
GPRS	General Packet Radio Service

UART	Universal asynchronous receiver/transmitter
PWM	Pulse Width Modulation

Contents

Chapter 1

1 Introduction

THE GLOBAL POSITIONING System (GPS) is a satellite positioning technology developed by the U.S. Department of Defense (DOD). Commercial products using GPS receivers are widely available on the market. In many areas of transport, GPS is being installed in vehicles with the primary objective of providing accurate monitoring of location. To provide better customer service in the competitive taxi business, fast and efficient dispatching is a critical factor. Taxi dispatching systems have relied traditionally on good staff and teamwork, but misunderstandings occur when bookings are transmitted by voice.

This project will initiate a new application using GPS, a smart satellite-based system for tracking and dispatching taxis to commuters. With the new system, the nearest taxi is located with the help of GPS and then bookings and routings are transmitted to the taxi driver's display unit in digital form. Through the use of this computerized satellite-based taxi dispatching and data network system, misunderstandings and costs are expected to be reduced.

1.1.1 Problems in existing DIAL-A-CAB Systems

Currently in Pakistan conventional radio based manual booking system is used. There have been constant complaints about difficulties in getting taxi service, especially in high-demand areas. Commuters often complain that it is not easy to get taxis by dial-a-cab service especially during peak periods, such as when people want to go to work

and on weekends. Taxi companies are frequently requested to improve their dial-a-cab service.

Communication is another problem, especially on rainy days or when taxis are far from radio transmitters. With conventional radio-based, manual booking systems, jobs are announced over the radio; the operator can only talk to about six taxis at a time. Sometimes, a cab driver who is nearer to the location cannot be heard, so another driver who is further may be dispatched. Radio-taxi drivers often complain that calls are not clear, and they are unsure where the passenger is to be picked up. If they are not sure of the pick-up place, they usually do not want to accept a call.

1.1.2 Why GPS based Dispatching System

With this system, computerized dispatching links both commuters and taxi drivers by computer and lets the operators know, by satellites, where their taxis are located. Every taxi has its location updated periodically. The system can find the nearest vacant cab, and the cab's display panel can tell the driver passenger's location. Once a job is acknowledged by the driver, the system can automatically contact the customer telling him or her taxi's license plate number through a synthesized text message.

As a future enhancement to this project frequent taxi commuters can open an account with the taxi company for free, either by Internet or by phone. Account holders can have their particulars and up to five expected pick-up points logged into the computer for speedy booking.

1.2 Review of Literature

1.2.1 A Collaborative Multiagent Taxi-Dispatch System

The paper "A Collaborative Multiagent Taxi-Dispatch System" presents by "Kiam Tian Seow" presents a novel multiagent approach to automating taxi dispatch that services current bookings in a distributed fashion. The existing system in use by a taxi operator in Singapore and elsewhere, attempts to increase customer satisfaction locally, by sequentially dispatching nearby taxis to service customers. The proposed dispatch system attempts to increase customer satisfaction more globally, by concurrently dispatching multiple taxis to the same number of customers in the same geographical region. To realize the system, a multiagent architecture is proposed, populated with software collaborative agents that can actively negotiate on behalf of taxi drivers in groups of size N for available customer bookings. Theoretically, an analysis of the boundary and optimal multiagent taxi-dispatch situations is presented along with a discussion of their implications. Experimentally, the operational efficiency of the existing and proposed dispatch systems was evaluated through computer simulations. The empirical results, obtained for a 1000-strong taxi fleet over a discrete range of N, show that the proposed system can dispatch taxis with reduction in customer waiting and empty taxi cruising times of up to 33.1% and 26.3%, respectively; and up to 41.8% and 41.2% reduction when a simple negotiation speedup heuristic was applied.

1.2.2 Decabot- A Study in "Go To" Navigation using GPS

The Decabot project was completed in 2005 by Santa Clara University. This project utilized Amigobot robots to develop a "go to" navigation system using GPS data, and to develop a multi robot test bed for use in future research. "Go to" navigation is defined by a desire to reach a single point. The vehicle uses it current state to determine how to drive to its destination point. Each vehicle was equipped with a GPS receiver and digital compass. The vehicles communicated their position and heading information to a central computer using wireless modems. The computer worked through a control algorithm for each robot, and then sent a new heading and drive data back to each of the vehicles, with the ultimate goal of getting each of the robots to the desired GPS coordinate.

1.2.3 Way Point Following by Robot

An assistant professor at San Francisco State University developed a vehicle to follow a set of GPS waypoints. The vehicle chosen was a simple remote controlled car. Using only a simple handheld GPS receiver and microprocessor, the vehicle was able to follow a series of GPS waypoints. The speed was controlled by having a set limit through each section of its course, and therefore the system did not need to compute an appropriate speed to travel through certain section of course. Another very interesting aspect of this paper is its description of position estimator it used. With the new GPS data available after every two seconds, it became necessary to estimate vehicle position between readouts using the vehicle's speed and heading

information. Using the state estimation technique allowed heading correction more often than basic GPS system will allow.

1.3 Project Objectives

This project aims at development of autonomous CAB dispatching system which accepts input from common users via a simple android application and as a result directs the nearest vacant CAB towards the passenger. For implementation purposes a simple robot has been used. Equipped with GPS and Compass module, this robot navigates between a series of way points to reach its destined coordinates.

Chapter 2

2 System Level Description

2.1 System Function Analysis

The project breaks down into several major components

1. Android Application
2. Main Controller
3. Navigation Control System
4. GUI for monitoring and Feedback.

- Android Application “Smart Cab” is relatively easy and simple to use. Its graphical layout consists of a button which when pressed performs two major functions. First it verifies whether the GPS of the device is on, if not it prompts the user to turn the GPS on. Secondly, it waits for the GPS to lock its location. Once the location is locked it fetches the latitude and longitude and then transfers them wirelessly to the main controller using the GSM of device. For Complete details on “Smart Cab” see Chapter # 3.

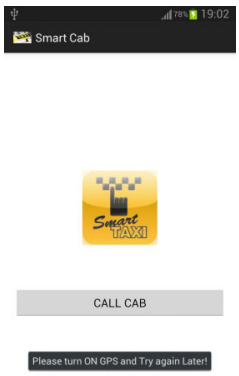

Figure 2.1: Snapshot of Android App

- The main controller consists of Arduino UNO board interfaced with Arduino GSM Shield and a PC. Main Controller is continuously receiving the location of vacant cabs and users. It evaluates shortest possible distance between user and vacant cabs. The relevant data is serially transferred to a PC where correct and nearest path is determined using the Google Maps API. The main controller also sends a confirmatory text message to user if a vacant cab is available. For Complete details on main controller, see Chapter # 4.
- The navigation Control System is responsible for performing series of important tasks which help the robot to navigate between two waypoints successfully. It takes the data from the GPS receiver and digital compass to compute the necessary heading. It compares the robot's current heading and desired heading. It then signals motor controller to make necessary course corrections. Detailed description of Navigation Control System can be found in chapter # 5.
- A graphical user interface has been developed for monitoring purpose. This GUI displays the location of all the cabs and users on Google maps. It also maintains a database of all the dispatching jobs on a certain day. Detailed description of this GUI can be found in chapter # 6.

2.2 System Tradeoff Analysis

For the four separate subsystems mentioned in the Function Analysis section, many alternative techniques arose to solve the intended goals.

- The first subsystem was the Navigation Control System. The most important and disputable was our decision to use the GPS receiver and digital compass for the position orientation of robot. Many possible solution arose but due to budgetary and availability constraints we selected U-BLOX U-kel GPS receiver with accuracy of 5 meter and update time of 5 seconds. We used Honeywell HMC 6352 Compass with an accuracy of 1 degree. We chose these sensors because of ease of design, reasonable price and ability to meet the desired objectives.
- The second important subsystem was steering mechanism. The question arose about the number of steering wheels. The possible options were either one steerable wheel or two connected with a common shaft/rod. Since two steerable drive wheels offer a great friction area and improve the reliability of turning, we decided to go for it.
- The third and perhaps the most important point of disagreement was choice of microprocessor for Navigation Control System (NCS). A couple of group members suggested using Panda board in NCS. But because of limited availability of Panda board in Pakistan and its high price, Arduino Mega board was used instead.

2.3 Team and Project Management

The team consists of four Electronic Engineers

1. Abdul Haleem Abbasi
2. Irtsam Ghazi
3. Muhammad Rashid Maqbool
4. Muhammad Sharjeel Akhtar

2.3.1 Division of Labor

To increase the efficiency of project and ensure every one stays on task to meet deadlines ahead of schedule; each member was assigned a different task depending on his strengths. Sharjeel wrote the algorithms for android application, main controller and was also in charge of procurement of equipments. Interfacing of module with micro controllers and their piece wise testing and debugging was handled by Abdul Haleem. Rashid was responsible for complete mechanical design and construction, along with this he also assisted Irtsam in writing codes for Navigation Control System and Graphical User Interface. However, every group member was involved in complete integration and testing.

2.3.2 Budget

The major problem that came up was budget of this project. We tried our best to avoid expensive electronic components. Many attempts were made to come up with the ideas that would reduce the need for high priced electronics. In order to save money, we made few compromises; we used relatively cheaper GPS and compass.

A copy of budget for this project can be found in Appendix A. It lists all components along with their prices.

Chapter 3

3 Android Application Development

3.1 Introduction

World is contracting with the growth of mobile phone technology. As the number of users is increasing day by day, facilities are also increasing. Starting with simple regular handsets which were used just for making phone calls, mobiles have changed our lives and have become part of it. Now they are not used just for making calls but they have innumerable uses and can be used as a Camera , Music player, Tablet PC, T.V. , Web browser etc . And with the new technologies, new software and operating systems are required.

Operating Systems have developed a lot in last 15 years. Starting from black and white phones to recent smart phones or mini computers, mobile OS has come far away. Especially for smart phones, Mobile OS has greatly evolved from Palm OS in 1996 to Windows pocket PC in 2000 then to Blackberry OS and Android.

One of the most widely used mobile OS these days is ANDROID. Android does a software bunch comprise not only operating system but also middleware and key applications. Android Inc was founded in Palo Alto of California, U.S. by Andy Rubin, Rich miner, Nick sears and Chris White in 2003. Later Android Inc. was acquired by Google in 2005. After original release there have been number of updates in the original version of Android.

Android is a powerful Operating System supporting a large number of applications in Smart Phones. These applications make life more comfortable and advanced for the users. Hardware that supports Android is mainly based on “ARM architecture” platform.

3.1.1 Android Applications

These are the basics of Android applications:

- Android applications are composed of one or more application components (activities, services, content providers, and broadcast receivers)
- Each component performs a different role in the overall application behavior, and each one can be activated individually (even by other applications)
- The manifest file must declare all components in the application and should also declare all application requirements, such as the minimum version of Android required and any hardware configurations required
- Non-code application resources (images, strings, layout files, etc.) should include alternatives for different device configurations (such as different strings for different languages)

3.1.2 Application Development Process

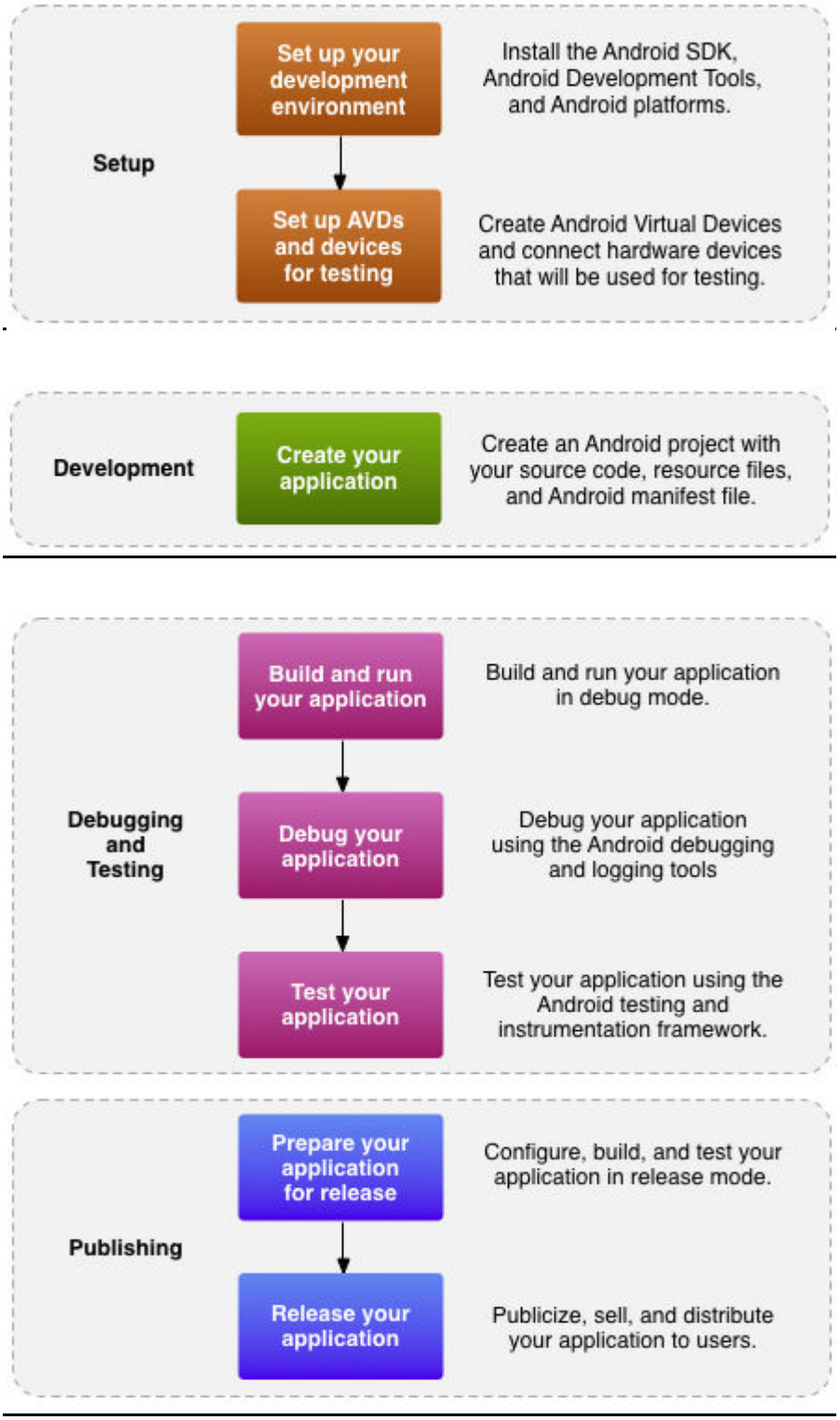

Figure 3.1: Android App Development Process

3.2 Smart CAB

The Android App we developed for this project is called “Smart CAB”. Android Application “Smart Cab” is relatively easy and simple to use. Its graphical layout consists of a button which when pressed performs two major functions. First it verifies whether the GPS of the device is on, if not it prompts the user to turn the GPS on. Secondly, it waits for the GPS to lock its location. Once the location is locked it fetches the latitude and longitude and then transfers them wirelessly to the main controller using the GSM of device.

The working of this operation can be divided into three different parts:

1. On Click Listener
2. Fetching of GPS coordinates
3. SMS/Telephony Manager

3.2.1 ON Click Listener

One of the most basic tasks one will run up against when starting programming for Android is how to implement on click handlers.

On click handlers are used to execute a part or code when a button or other part of the user interface is touched (clicked). These on click handlers are an instance of a class that one need to specify. In short it’s an object with a piece of executable code that get’s triggered by a user interaction.

First thing to know is that an on click handler implements the interface View. *OnClickListener*. You need to create a class that implements the interface because you need to add your code that is to be executed.

The Android platform also comes with a binding method of binding onclick handlers to layout files. This might be the cleanest solution, but does come with a drawback that it can only be used for layouts that are inflated for an activity. Now a day's it's recommended to use fragments for parts of your user interface in your Android apps, this method cannot be used with fragments.

While writing code for "Smart Cab", all of code was placed in the routine of *Onclick listener*. The system will wait for the user to press the button and once the button is pressed it will go on to execute the rest of the code such as fetching of GPS coordinates and sending these coordinates via SMS to a hard coded mobile number.

3.2.2 Fetching GPS Coordinates

Knowing where the user is allows your application to be smarter and deliver better information to the user. When developing a location-aware application for Android, one can utilize GPS and Android's Network Location Provider to acquire the user location. Although GPS is most accurate, it only works outdoors; it quickly consumes battery power, and doesn't return the location as quickly as users want. Android's Network Location Provider determines user location using cell tower and Wi-Fi signals, providing location information in a way that works indoors and outdoors, responds faster, and uses less battery power. In "Smart Cab" to obtain the user location both GPS and the Network Location Provider were used.

In Smart Cab a Location listener class was implemented for finding GPS coordinates. The user's location was found by means of callback. The function **requestLocationUpdates (void)** was called and passed to "LocationListener". The LocationListener must implement several callback methods that the Location Manager calls when the user location changes or when the status of the service changes.

In order to receive location updates from NETWORK_PROVIDER or GPS_PROVIDER, user permission were requested by declaring the ACCESS_COARSE_LOCATION and ACCESS_FINE_LOCATION permission, respectively, in the Android manifest file

3.2.3 SMS/Telephony Manager

In some situations, we want to send SMS messages from our own Android application. There are two different ways to do such a thing:

- By using *SmsManager* class
- By using an implicit *Intent*

In Smart Cab *SmsManager* class was used to send SMS. The use of *SmsManager* facilitates the SMS sending because it gives us the opportunity to customize this functionality with the manner we want, within our own Activity. In parallel, the use of Intents implies the use of built-in applications and SMS clients that are installed in each Android device.

Chapter 4

4 Main Controller

The main controller is the main part of our project, it is continuously receiving the coordinates and status of CABs and it is the function of main controller to receive the coordinates of user and then if there are two or more than two vacant CABs available it will calculate the shortest distance from the user to the nearest CAB and then main controller will dispatch the nearest CAB to the user.

The Hardware of the main controller consists of following two equipments.

1. Arduino Uno
2. GSM Shield

Figure 4.1: Arduino Uno with GSM Shield

4.1 Arduino Uno

The Arduino Uno is a microcontroller board based on the ATmega328. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

4.1.1 Specifications of Arduino Uno

- Microcontroller ATmega328
- Operating Voltage 5V
- Input Voltage (recommended) 7-12V
- Input Voltage (limits) 6-20V
- Digital I/O Pins 14 (of which 6 provide PWM output)
- Analog Input Pins 6
- DC Current per I/O Pin 40 mA
- DC Current for 3.3V Pin50 mA
- Flash Memory 32 KB
- SRAM 2 KB (ATmega328)
- EEPROM 1 KB (ATmega328)
- Clock Speed 16 MHz

4.1.2 Key Features

- High Performance, Low Power AVR 8-Bit Microcontroller
- Advanced RISC Architecture
- High Endurance Non-volatile Memory Segments
- Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
- One 16-bit Timer/Counter with Separate Prescaler, Compare and Capture Mode
- Real Time Counter with Separate Oscillator
- Six PWM Channels
- 8-channel 10-bit ADC in TQFP and QFN/MLF package
- 6-channel 10-bit ADC in PDIP Package
- Programmable Serial USART
- On-chip Analog Comparator
- External and Internal Interrupt Sources
- Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- Temperature Range: -40°C to 85°C
- Low Power Consumption at 1 MHz, 1.8V, 25°C
    - Active Mode: 0.2 mA
    - Power-down Mode: 0.1 µA
    - Power-save Mode: 0.75 µA

4.2 GSM Shield

The Arduino GSM Shield allows an Arduino board to connect to the internet, make/receive voice calls and send/receive SMS messages. The shield uses a radio modem M10 by Quectel. It is possible to communicate with the board using AT commands.

The shield contains a number of status LEDs:

- On: shows the Shield gets power.
- Status: turns on to when the modem is powered and data is being transferred to/from the GSM/GPRS network.
- Net: blinks when the modem is communicating with the radio network.

The shield supports AIN1 and AOUT1 as audio interfaces; an analog input channel and an analog output channel. The input, exposed on pins MIC1P/MIC1N, can be used for both microphone and line inputs. An electric microphone can be used for this interface. The output, exposed as lines SPK1P/SPK1N, can be used with either a receiver or speaker. Through the modem, it is possible to make voice calls. In order to speak to and hear the other party, you will need to add a speaker and microphone.

There are three small buttons on the shield. The button labeled "Arduino Reset" is tied to the Arduino reset pin, when pressed, it will restart the sketch. The button

labeled "SIM 900 Power" is connected to the modem and will power the modem on and off. The button labeled “SIM 900 RST” will reset the modem.

4.3 Working of GSM

GSM uses a combination of TDMA (time division multiplexing) and FDMA (frequency division multiplexing). This means that users A and B are not only sharing the channel in time but also frequency. This means that user A is on the channel 890Mhz for two seconds, then jumps to 900Mhz channel for the next two seconds, then jumps to 910Mhz for the next two seconds and so on. Thus, each user is uses a different frequency at different time slots. This is called Frequency Hopping. There are GSM 900, 1200, 1800, 2100etc. GSM 900 is the operational frequency of the GSM in mega hertz. There are 125 up link channel and 125 down link. Uplink channel means from mobile to base station and downlink means from base station to the mobile. 124 channels reserved for communication and 1 is used for safe guard (100 hertz in each) which provides a separation between the channels.

4.4 Introduction to AT Commands

AT commands are instructions used to control a modem. AT is the abbreviation of ATtention. Every command line starts with “AT” or “at” that’s why modem commands are called AT commands. AT commands set, GSM/GPRS modems and mobile phones support an AT command set that is specific to the GSM technology, which includes SMS related commands like

AT+CMGS (Send SMS message), AT+CMSS (Send SMS message from storage), AT+CMGL (List SMS messages) and AT+CMGR (Read SMS messages).

Note that the starting “AT” is the prefix that informs the modem about the start of a command line. It is not part of the AT command name.

Note that mobile phone manufacturers usually do not implement all AT commands, command parameters and parameter values in their mobile phones. Also, the behavior of the implemented AT commands may be different from that defined in the standard. In addition, some AT commands require the support of mobile network operators.

There are two types of AT commands: basic command and extended commands.

Basic commands are AT commands that do not start with “+”. For example, D (Dial), A (Answer), H (Hook control) and O (Return to online data stare) are basic commands. Extended commands are AT commands that start with “+”. All GSM AT commands are extended commands. For example +CMGS (Send SMS message), +CMSS (Send SMS message from storage), +CMGL (List SMS message) and +CMGR (Read SMS message), are extended commands.

4.4.1 Sending SMS by using AT commands

Steps	Command	Function	Response
Step No:1	"AT"	To initialize GSM modem	OK
Step No:2	AT+CMGS=\"+92xxxxxxxxxx\"	To send SMS to any particular number	>
Step No:3	byte(26)	To send Cltrl+Z to the GMS	+CMGS : 3,99

Table 4.1: Sending SMS

4.4.2 Receiving SMS by using AT commands

Steps	Command	Function	Response
Step No:1	"AT"	To initialize GSM modem	OK
Step No:2	AT+CNMI=1,2,0,0,0	To initialize GSM to the text mode	OK

Table 4.2: Receiving SMS

4.5 Working of GSM.h library

Arduino GSM Shield library enables an Arduino board to do most of the operations you can do with a GSM phone: place and receive voice calls, send and receive SMS, and connect to the internet over a GPRS network.

The GSM shield has a modem that transfers data from a serial port to the GSM network. The modem executes operations via a series of AT commands. The library abstracts low level communications between the modem and SIM card. It relies on the Software Serial library for communication between the modem and Arduino.

Typically, each individual command is part of a larger series necessary to execute a particular function. The library can also receive information and return it to you when necessary.

As the library enables multiple types of functionality, there are a number of different classes.

- The GSM class takes care of commands to the radio modem.
- Voice calls handling, managed by the GSM Voice Call class.
- Send/receive SMS messages, managed by the GSM_SMS class.
- GSM Client includes implementations for a client, similar to the Ethernet and Wi-Fi libraries.

4.6 Flow chart of Main Controller

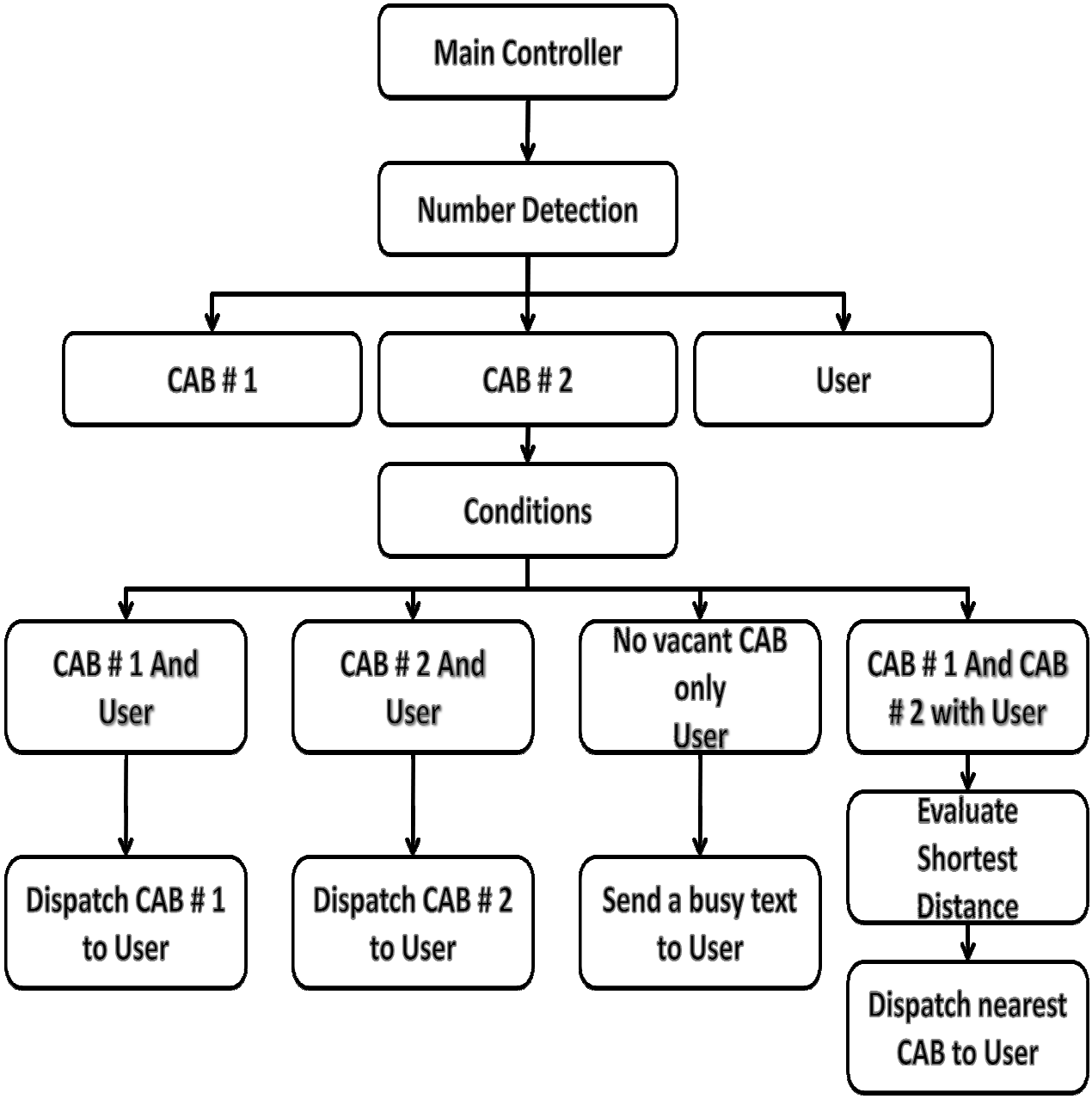

Figure 4.2: Main Controller (Flow Chart)

4.7 Working of Main Controller

The main controller consists of Arduino Uno and GSM shield, the programming of main controller has been done in such a way that it initially waits to receive the text message. Numbers of both CABs have already been stored in main controller and as the main controller receives the SMS it will perform a comparison to detect whether this SMS is form CAB1, CAB2 or user.

If the CAB is vacant then driver will press the vacant button, the main controller will receive the latest coordinates of the CAB and also the main controller will update the status of CAB as vacant, after that the CAB will automatically updates its coordinates to the main controller every two minutes.

The user will send his/her coordinates using his/her Smartphone application, the main controller will look for the number first and if this number is not either of the CAB then it will save these coordinates as the user's coordinates.

After number detection main controller will look for the following conditions.

4.7.1 CAB #1 and user

If the CAB 1 is available and the user has send the request for the CAB, the main controller will send the user's coordinates to the CAB 1 and also the main controller will send the confirmatory message (Your CAB has been directed to your present location) to the user, and after that the main

controller will up the status of CAB 1 as non vacant and the CAB 1will be guided by GPS towards the user using python language.

4.7.2 CAB #2 and user

If the CAB 2 is vacant and user is requesting for the CAB using his/her Smartphone application then main controller will send the user's coordinates to CAB 2 and send a confirmatory SMS to the user and then main controller will up the status of CAB 2 as non vacant and also it will guide the CAB 2 to the user using python language.

4.7.3 No vacant CAB only user

If there is no vacant CAB available and user send request of CAB to the main controller using Smartphone app, then main controller will send a busy text (Sorry, No vacant CABS are available.) to the user.

4.7.4 CAB #1, CAB #2 and user

If both of the CABs are available and user is requesting for the CAB, then main controller will evaluate the shortest distance path to see which of the CAB is nearest to the user by using haversine formula, then the main controller will send the user's coordinates to the nearest CAB and up the status of the nearest CAB as non vacant and then send the confirmatory message to the user, after that the main controller will guide the CAB to the user using python language.

Chapter 5

5 Navigation Control System

5.1 Introduction to AGVs

Automated guided vehicles (AGVs) increase efficiency and reduce costs by helping to automate a manufacturing facility or warehouse. The first AGV was invented by Barrett Electronics in 1953. The AGV *can tow objects* behind them in trailers to which they can autonomously attach. The first AGV was brought to market in the 1950s, by Barrett Electronics of Northbrook, Illinois, and at the time it was simply a tow truck that followed a wire in the floor instead of a rail.

5.2 Navigation

5.2.1 Wired

A slot is cut in to the floor and a wire is placed approximately 1 inch below the surface. This slot is cut along the path the AGV is to follow. This wire is used to transmit a radio signal. The sensor detects the relative position of the radio signal being transmitted from the wire.

5.2.2 Guided tape

AGV use tape for the guide path. The tapes can be one of two styles: magnetic or colored. The AGC is fitted with the appropriate guide sensor to follow the path of the tape.

5.2.3 Laser target navigation

The navigation is done by mounting reflective tape on walls, poles or fixed machines. The AGV carries a laser transmitter and receiver on a rotating turret. The laser is transmitted and received by the same sensor. The angle and (sometimes) distance to any reflectors that in line of sight and in range are automatically calculated.

5.2.4 GPS guided Navigation

Another and one of the most famous forms now days is the use of the Global Positioning System in order to navigate the vehicle. In this case, vehicle is continuously is in communication with the satellites and taking its current coordinates and make comparison with its destination coordinates and reach the destination.

5.2.5 Vision guidance

Vision-Guided AGVs can be installed with no modifications to the environment or infrastructure. They operate by using cameras to record features along the route, allowing the AGV to replay the route by using the recorded features to navigate.

5.3 Hardware

5.3.1 Arduino AT Mega 2560

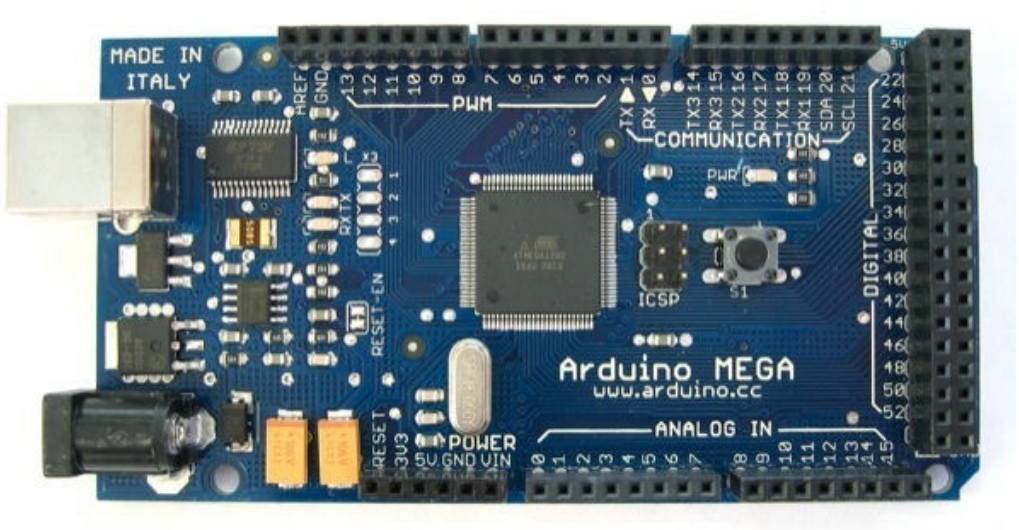

Figure 5.1: Arduino Mega 2560

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. Its key features are given in the table:

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (Recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Table 5.1: Specifications of Arduino Mega 2560

5.3.2 u-blox VK16U6 TTL GPS Module

Figure 5.2: VK16U6

The key feature, specifications, performance and operational temperature of ublox VK16U6 TTL GPS module are discussed below.

25*25*4mm GPS antenna
TTL level output
Adopts KDS 0.5PPM hgih precision TCXO
Built-in RTC cystal hot start
Built-in EEPROM
5Hz location refresh speed

Table 5.2: Features of VK16U6

2D dimension	5m
Drift	<0.02m/s
Time precision	1us
Coordinate system	WGS-84
Max.altitude	18000m
Max.acceleration	500m/s
Accelerator	<4g

Table 5.3: Position Performance of VK16U6

Main chip	Ublox
C/A code	1.023MHz
Receive frequency	L1[1575.42MHz]
Follow channel	50
Operating Temperature	-30 to 80 degrees
Output	NMEA 0183 V3.0

Table 5.4: Specifications of VK16U6

5.3.3 GSM Module SIM900D

Figure 5.3: SIM 900D

SIMCom presents an ultra compact and reliable wireless module-SIM900D. This is a complete Quad-band GSM/GPRS module in a SMT type and designed with a very powerful single-chip processor integrating AMR926EJ-S core, allowing you to benefit from small dimensions and cost-effective solutions.

Quad-Band 850/ 900/ 1800/ 1900 MHz
Weight: 6.2g
Dimensions: 33*33*3mm
Control via AT commands (GSM 07.07 ,07.05 and SIMCOM enhanced AT Commands)
SIM application toolkit
Supply voltage range : 3.2 ... 4.8V
Operation temperature: -40 °C to +85°C
Low power consumption: 1.0mA(sleep mode)

Table 5.5: Features of GSM 900D

5.3.4 Compass Module Honeywell HMC 6352

Figure 5.4: HMC6352

Figure 5.4: HM6352

The Honeywell HMC6352 is a fully integrated compass module that combines 2-axis magneto-resistive sensors with the required analog and digital support circuits, and algorithms for heading computation.

Compass with Heading Output
Full Integration of 2-Axis Magnetic Sensors and Electronics
Small Surface Mount Package
Low Voltage Operation (2.7 to 5.2V)
I^2C 2-Wire Serial Interface
Wide Magnetic Field Range

Table 5.6: Features of HMC6352

5.3.5 Robot

The robot used in our project is a toy jeep. We have removed the upper part of the jeep and mounted a plastic plate on it. All the components like microcontroller, GSM module, GPS module batteries are attached to the plate and fixed with the help of cable tie. Compass is attached a little higher than the other components. The reason is that compass is affected by the magnetic field. To avoid magnetic field it is placed a little higher than the other components. To make its upper part to look like a cab, we took a yellow plastic sheet and mould it to give it a shape of CAB.

5.3.6 Battery

After determining the power consumption characteristics for each component we had choose a 12V battery that specification are given in the table.

Cell type	Ni-Cd
Size	12cm x 6cm
Nominal Voltage	12V
Current	2.2Ah
Weight	100g

Table 5.7: Specifications of Battery

And the consumption of all the necessary components are given in the table below

Voltage Source	**Component**	**Voltage**	**Current**	**Power**
12V	Microcontroller	12	70-100mA	0.68-1.2 W
5V	GPS	5V	110mA	0.55W
5V	GSM	5V	100mA	0.50W
12V	Motors	12V	2A	24W

Table 5.8: Battery Consumption

5.3.7 Motor

The specification of the motors used in our project is:

Power rating	24W
Nominal Voltage	12V
No load current	0.5A
Terminal resistance	1.71Ω
Starting current	2.5A

Table 5.9: Specifications of Motor

5.4 Global Positioning System (GPS)

5.4.1 Introduction

The Global Positioning System (GPS) is a space-based satellite navigation system that provides location and time information in all weather conditions, anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. The system provides critical capabilities to military, civil and commercial users around the world. It is maintained by the United States government and is freely accessible to anyone with a GPS receiver.

The GPS project was developed in 1973 to overcome the limitations of previous navigation systems, integrating ideas from several predecessors, including a number of classified engineering design studies from the 1960s. GPS was created and was originally run with 24 satellites. It became fully operational in 1995.

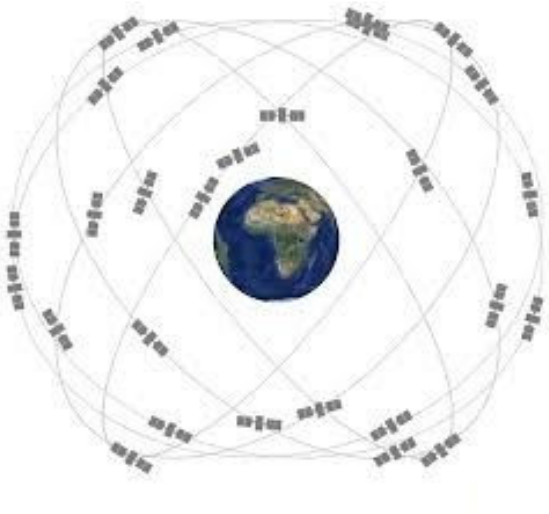

Figure 5.5: GPS Satellites

5.4.2 Description

GPS is used for general purpose positioning and navigation. It is a constellation of 24 satellites that orbit the earth at a very high altitude of approximately every 12 hours, emitting signals to earth at precisely the same time. The position and the time information transmitted by these satellites is used by a GPS receiver to triangulate a location coordinate on earth using four or more satellites.

The satellites broadcast on two carrier frequencies of L-band of the electromagnetic spectrum. One is the “L1” or 1575.42MHz and the other is “L2” or 1227.6MHz. On these carrier frequencies are broadcast codes, much like a radio or television station broadcast information on their channels (frequencies). The satellites broadcast two codes, a military-only encrypted Precise Position Service (PPS) code and a civil-access or Standard Position Service (SPS) code.

As stated earlier, the u-blox VK16U6 TTL GPS selected for use in this project sends information formatted in the National Marine Electronics Association (NMEA) standard. This data must be interpreted into a format that can be used for navigation. This task was carried out by receiving the GPS sentences in the primary microcontroller and then transmitting them over to the CAB.

5.4.3 NEMA Sentence

There are 19 NEMA interpreted sentences which are given below.

$GPBOD - Bearing, origin to destination

$GPBWC - Bearing and distance to waypoint, great circle

$GPGGA - Global Positioning System Fix Data

$GPGLL - Geographic position, latitude / longitude

$GPGSA - GPS DOP and active satellites

$GPGSV - GPS Satellites in view

$GPHDT - Heading, True

$GPR00 - List of waypoints in currently active route

$GPRMA - Recommended minimum specific Loran-C data

$GPRMB - Recommended minimum navigation info

$GPRMC - Recommended minimum specific GPS/Transit data

$GPRTE - Routes

$GPTRF - Transit Fix Data

$GPSTN - Multiple Data ID

$GPVBW - Dual Ground / Water Speed

$GPVTG - Track made good and ground speed

$GPWPL - Waypoint location

$GPXTE - Cross-track error, Measured

$GPZDA - Date & Time

5.4.4 Data Format from GPS Receiver

The GPS receiver transmits data through standard RS-232 serial connection. The unit can be configured to work at baud rates ranging up to38400 baud. For microcontrollers 9600 baud rate is considered optimum. The next task was to determine what information (or NEMA string) do we need and how to best format the data. The format of NEMA strings is shown in figure 5.7.

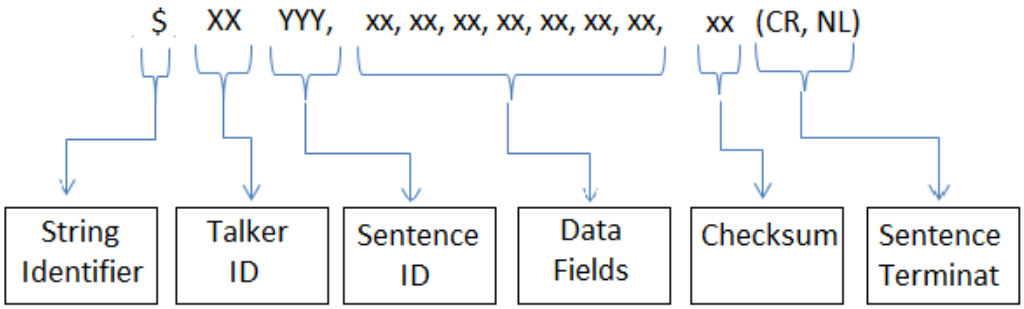

Figure 5.6: NEMA strings format

5.4.5 GPS Data Parsing

Before the data could be sent to the main controller, it needed to be formatted so that it would be usable. The targeted NMEA sentence was GPGLL because it provided us with all the necessary information in minimum length.

Since the data received by the microcontroller is in the sentence form, it had to parsed to extract information to a variable as a floating point (where appropriate) number instead of as a string. This is particularly important for the Longitude and Latitude. Data for Latitude and Longitude is is received as ddmm.mmmm (where "d" represent degree, "m" represent minutes, and 60 minutes is equal to 1 degree), meaning that the units are inconsistent. For example, Latitude at GIKI received as 34mm.mmmm, meaning that the receiver is located at 34 degrees mm.mmmm minutes. In order to ease navigation, this must be changed into only degrees. That same point in GIKI, when written simply in degrees, would be 34.xxxxxxx degree. The data is further adjusted for north or south (Latitude) and west or east (Longitude) readings, with south and west locations being assigned a negative value. Once in this format, locations can be easily compared.

5.4.6 Interfacing with Micro Controller

After the data was received and formatted, it needed to be put to use. Rather than sending the data every time it was received, it was undesirable for the controller to check for the data as a part of its routine. To accomplish this, the microcontroller was programmed to continuously wait for the string identifier and then start receiving desired string.

5.5 Distance Calculation

The navigation of the robot is handled using a formula known as Half-Sine formula or simply Haversine Formula. The formula requires original position and the destination coordinated (all in radians) to find the distance and the relative angle at which the destination lies compared to the original location.

5.5.1 Haversine Formula

The Haversine formula is an equation important in navigation, giving great-circle distances between two points on a sphere from their longitudes and latitudes. It is a special case of more general formula in special trigonometry, the law of haversines, relating the sides and angles of spherical "triangles".

$$Haversine(\theta) = Sin^2(\theta/2)$$

For two points on the sphere (of radius R) with latitude lat_1 and lat_2, ($\Delta lat = lat_2 - lat_1$), and longitudes $long_1$ and $long_2$, ($\Delta long = long_2 - long_1$), where angles are in radians, the distance d between two points (along a great circle of the sphere; see spherical distance) is related to their locations by the formula:

$$a = Sin^2(\Delta lat/2) + Cos(\Delta lat) * Sin^2(\Delta long/2)$$

$$c = 2 * tan^{-1}(\sqrt{a(1-a)})$$

$$Distance = a * c$$

5.5.2 Heading Control Theory

The heading of the robot or simply the direction in which robot must travel is calculated with the help of a Compass. The compass used in our project is of Honeywell HMC 6352. The compass is a fully integrated compass module that combines 2-axis magneto-resistive sensors with the required analog and digital support circuits, microprocessor and algorithms for heading computation.

5.5.2.1 I^2C Communication Protocol

The HMC6352 communicates via a two-wire I^2C bus system as a slave device. The HMC6352 uses a layered protocol with the interface protocol defined by the I_2C bus specification, and the lower command protocol. The data rate is the standard-mode 100kbps rate. The bus bit format is an 8-bit Data/Address send and a 1-bit acknowledge bit. The format of the data bytes (payload) shall be case sensitive ASCII characters and binary data returned. Negative binary values will be in two's complement form. The HMC6352 Serial Clock (SCL) and Serial Data (SDA) lines do not have internal pull-up resistors, and require resistive pull-ups (Rp) between the master device (usually a host microprocessor) and the HMC6352. Pull-up resistance values of about 10k ohms with a nominal 3.0-volt supply voltage. The SCL and SDA lines in this bus specification can be connected to a host of devices. The bus can be a single master to multiple slaves, or it can be a multiple master configuration. All data transfers are initiated by the master device which is responsible for generating the clock signal, and the data transfers are 8 bit long. All devices are addressed by I_2C's unique 7 bit address. After each 8-bit transfer, the master device generates a 9^{th} clock

pulse, and releases the SDA line. The receiving device (addressed slave) will pull the SDA line low to acknowledge (ACK) the successful transfer or leave the SDA high to negative acknowledge (NACK). All bus transactions begin with the master device issuing the start sequence followed by the slave address byte. The address byte contains the slave address; the upper 7 bits (bits7-1), and the Least Significant bit (LSb).

5.5.3 Working

The controller performs the calculations by getting a new reading from the compass and measures the difference in the last calculated angle and current angle. In this way, the controller comes to know either to move straight or left or right by the angle difference of previous two readings.

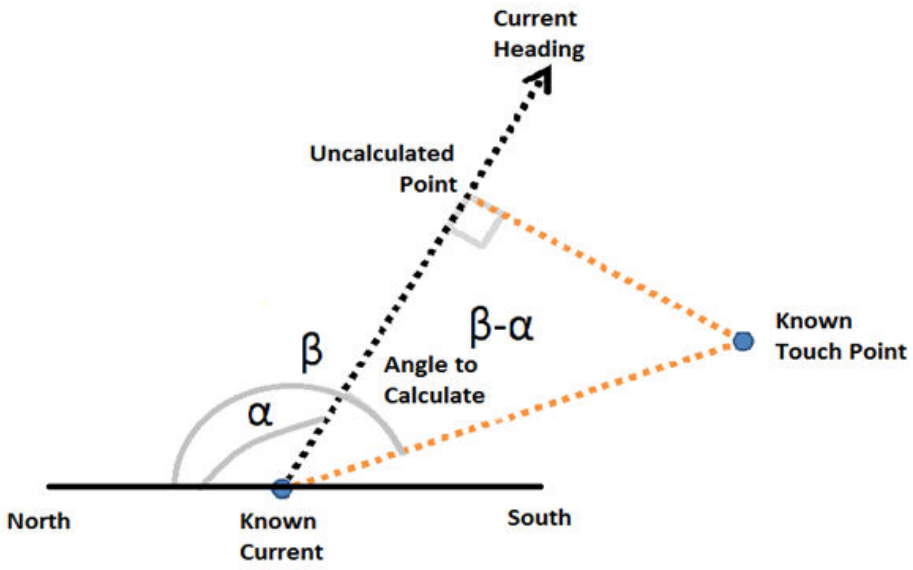

Figure 5.7: Heading Calculation

5.6 Motors Control

One of the important tasks of our project was the DC motor control. The motor controller was needed because we want to control the speed and the direction of the motor. For this purpose, we designed and constructed two H-Bridges using TIP142 Darlington Pair Transistor. General description of H-Bridge circuit is given below.

5.6.1 H- Bridge Design

In general an H-Bridge is a rather simple circuit, containing four switching elements, with the load at its center, in an H-Bridge configuration.

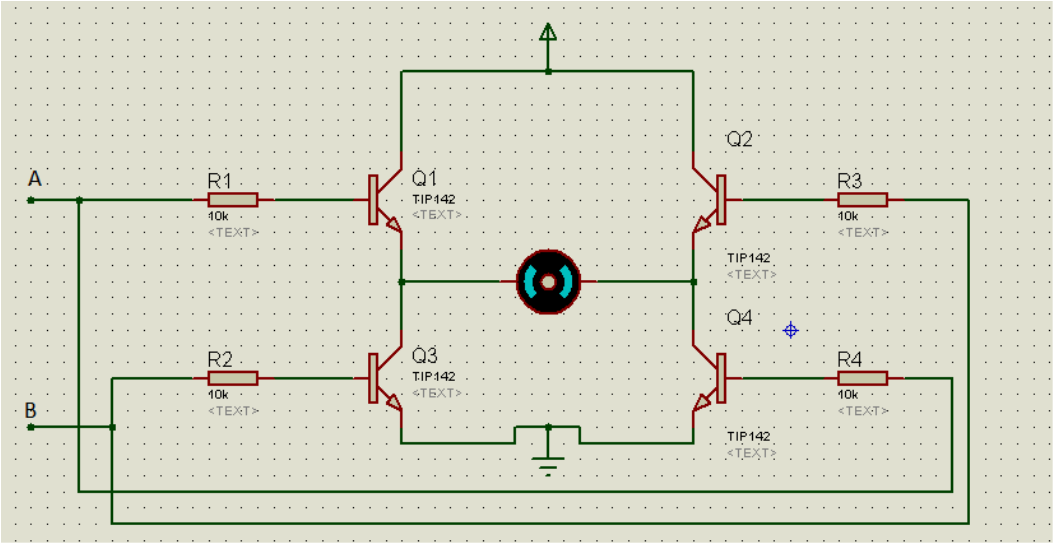

Figure 5.8: H-Bridge Design

The switching elements are usually bi-polar transistors. In high current cases, diodes called freewheeling diodes are also used. In general, all the switching elements can be turned on or off independently, though there are some obvious restrictions.

5.6.2 Basic Operation

The basic operation of an H-Bridge is fairly simple: if Q1 and Q3 are turned on, the right lead of the motor will be connected to the ground. While the left lead will be connected to the power supply. Current starts flowing through the motor which energizes the motor in the forward direction and the motor shaft starts spinning. If Q2 and Q4 are turned on, the converse will happen, the motor gets energized in the reversed direction, and the shaft will spinning in that way. If less than full speed operation is intended one of the switches are controlled in the PWM fashion.

5.6.3 Component Selection

For most part, the key decision to make for an H-Bridge is the selection of the switching elements. There are many factors to be considered, the most important ones are the operating current, the operating voltage and the switching (PWM) frequency. But in the case of heavy loads the biggest factor becomes the current requirements of the motors. For the heavy loads of around 6 kg, as was the case of our robot, the current requirements can go as high as 2A. Therefore, we had used TIP142 as it can easily accommodate current up to 4A.

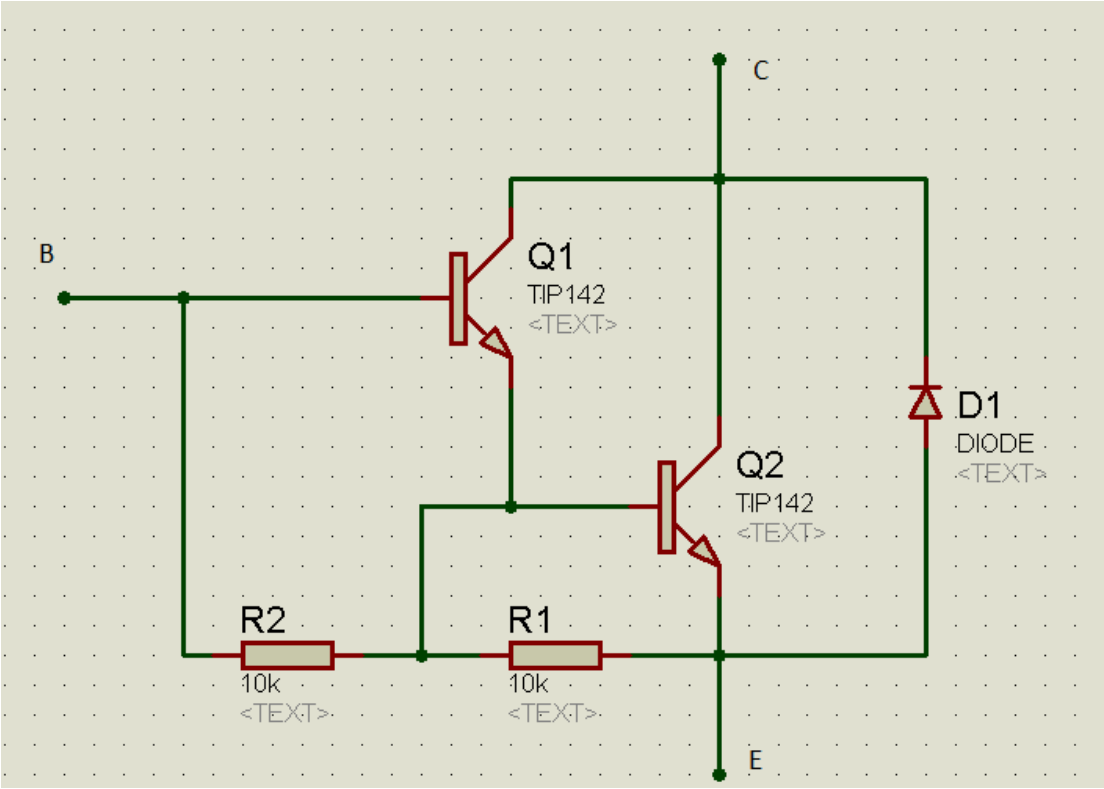

Figure 5.9: TIP 142

5.6.4 Freewheeling Diode

Freewheeling diodes are often overlooked or just briefly mentioned in most H-Bridges descriptions, but are very important components.

5.6.4.1 Basic Principle

The basic principle is very simple: while the bridge is on, two of the four switching elements carry the current, the diodes have no role. However, one the bridge is turned off, the switches will not conduct current anymore. As discussed earlier, by far the most common load for an H-bridge is an electric DC motor, which is an inductive load. What this means is that during the on time the motors will build an

electromagnetic field inside it. When the switch is turned off, that field has to collapse, and until that happen, current must still flow through the windings. That current cannot flow through the switches since they are off, but it will find a way. The catch diodes are in the design to provide a low- resistance path that collapse current and thus keep the voltage on the motor terminals within a reasonable range. For bipolar transistors TIP 142, these catch diodes are provided in the intrinsic way so no need to attach the external diode.

5.6.5 Drive Modes

A bridge can be driven in many different ways. In general, the on-time behavior is rather simple. Turn on one high-side and the opposite low-side switch to allow current to flow through the motor. It is the off-time drive that makes the difference. Since Q1 and Q2 or Q3 and Q4 should never turn on at the same time, there are only three different combinations for those two switches. Either Q1 conducts or Q2 conducts or none of them conducts but never both at the same time.

The direction of dc motor is controlled by giving one or zero to the four gate drivers from the micro controller. The control signals are shown in the table given below

State of motion	Q1	Q2	Q3	Q4
Stop	0	0	1	1
Clockwise	1	0	1	0
Anticlockwise	0	1	0	1

Table 5.10: Motor Control Signals

5.7 Robot Navigation

5.7.1 Flow Diagram

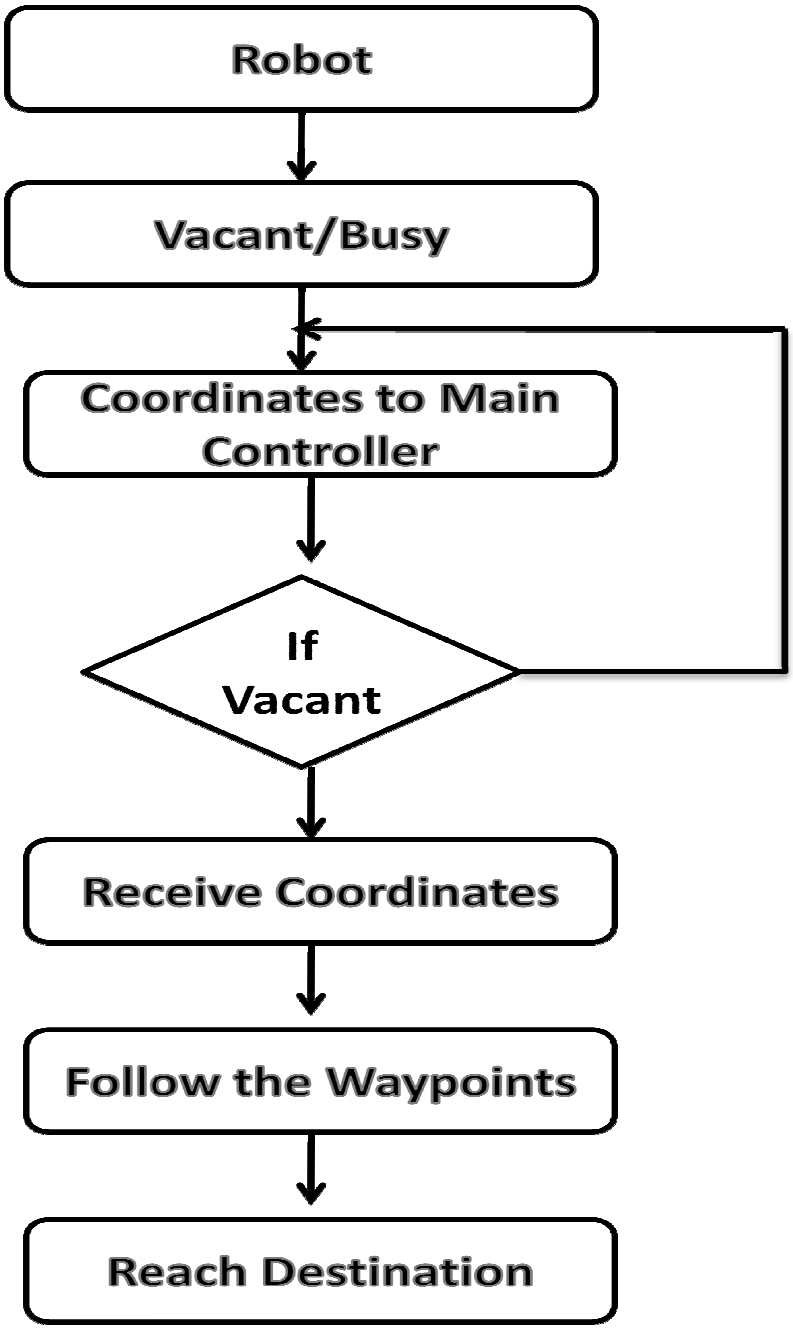

Figure 5.10: Robot Navigation Flow Diagram

5.7.2 Working

Initially the user initiates the request. The main controller receives the request, look for all the cabs in the city and dispatch the nearest vacant cab towards the user. As a prototype, we have used the robots in order to demonstrate our project.

We have also included a button in the robot as busy/vacant button. When the driver pushes the button it sends a message to the main controller that CAB is vacant. As stated earlier, main controller is attached to the python via a computer, so after getting the cab coordinates and user coordinates it will create a path between these two points as shown below.

After dividing the whole path into numerous waypoints, the coordinates of all the waypoints are sent to the robot via GSM and are stored in it.

Now the robot is ready to follow the path and to reach its destination. First of all it moved straight for about 500ms in order to get its current heading. Robot updates its coordinates after every 0.25s. After getting the current heading, it performs comparison with the heading of waypoint 1. If the difference is of 5 degrees or less, it will keep moving straight. And if the difference is greater than -180 or less than 0 then it will move right and vice versa.

After reaching waypoint 1, it will update it destined coordinate once and will start comparing the heading in the same fashion as stated above until it reaches waypoint 2. Following the same principle it will cross each waypoint and reach its destination.

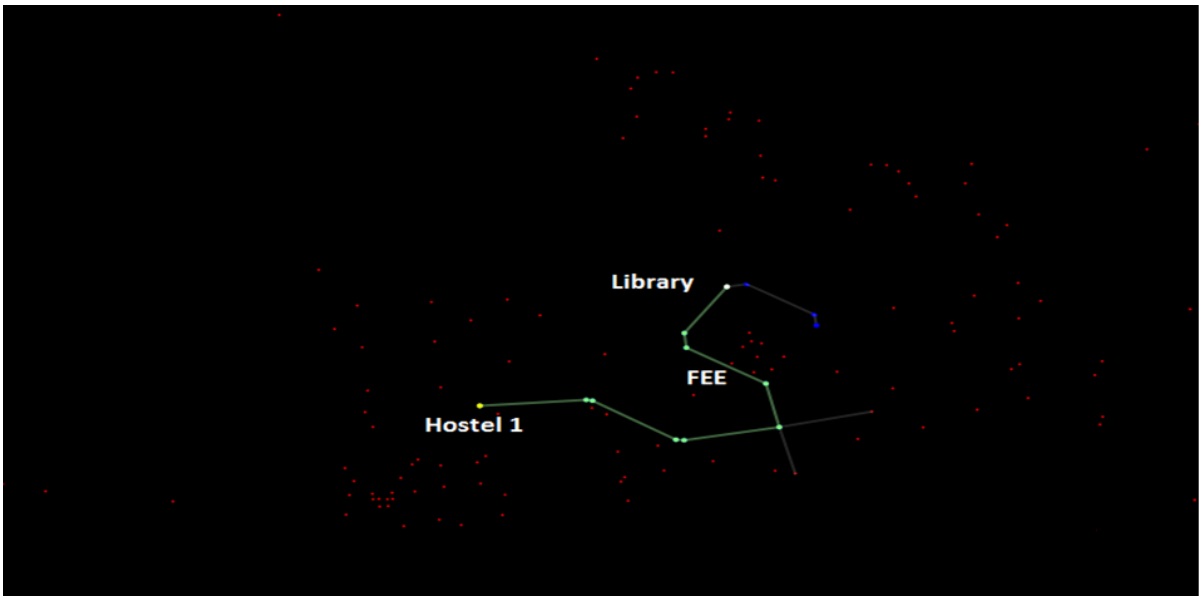

Figure 5.11: Python

Chapter 6

6 Graphical User Interface

The graphical user interface has been developed by interfacing various platforms to develop a user interface that displays:

- The user location by an android phone icon
- The cab location by a cab icon
- The path between user and cab
- The map and thence the surrounding vicinity of our users and cab at an adjustable zoom level

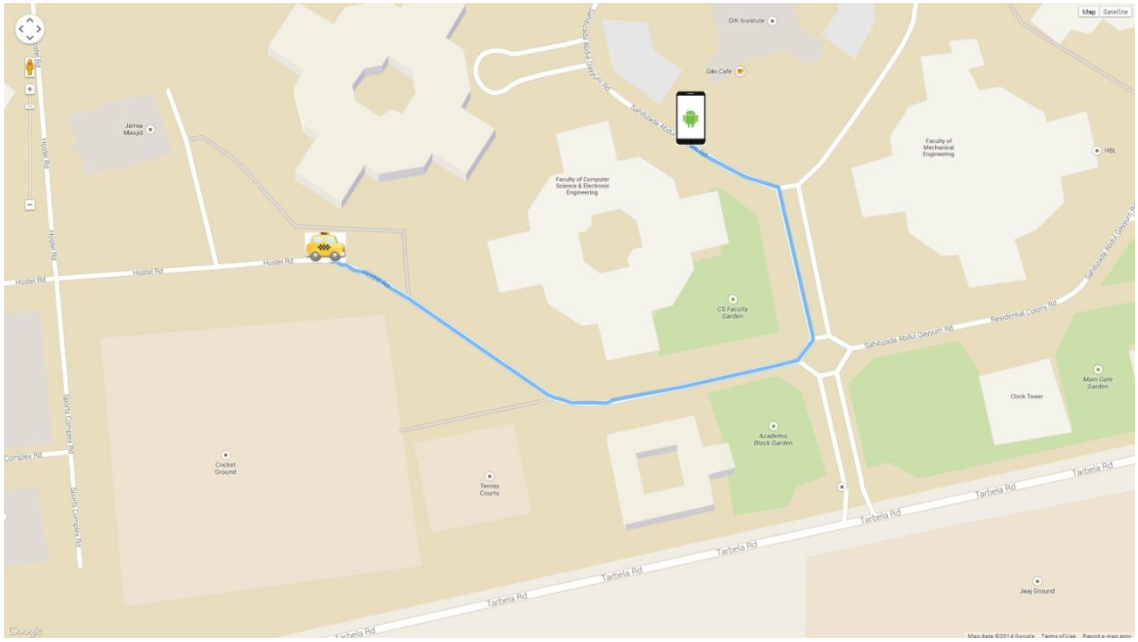

Figure 6.1: Graphical User Interface (GUI)

Thence if anyone wishes to see how many cabs are busy and how many are being utilized, or find out the location of a certain cab it is possible by viewing this GUI. The GUI is displayed on an LCD which has been interfaced with a CPU which communicates with our Arduino serially.

The software/platforms used to develop this GUI are:

1. Python
2. MySQL database
3. JavaScript
4. php
5. Google Maps

These various platforms and the way they have been used is explained in the following paragraphs.

6.1 Python

Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic

type system and automatic memory management and has a large and comprehensive standard library.

Like other dynamic languages, Python is often used as a scripting language, but is also used in a wide range of non-scripting contexts. Using third-party tools, such as Py2exe, or Pyinstaller, Python code can be packaged into standalone executable programs. Python interpreters are available for many operating systems.

CPython, the reference implementation of Python, is free and open source software and has a community-based development model, as do nearly all of its alternative implementations. CPython is managed by the non-profit Python Software Foundation.

In our project we have used various libraries of python and used it for two different purposes. Firstly it is used as an interface to obtain data serially and store it in our computer. The arduino constantly transmits data serially and a portion of our code fetches this data from the serial port of our PC.

The libraries of Python that have been used in our project and their respective functions are described below:

6.1.1 Pyserial

This is not a library but rather a script that we wrote ourselves. The basic functionality of the script was to keep fetching data from the serial port and then sequentially call the libraries involved. This code after receiving coordinates of the user and cab from our Arduino first calls the pyroute library which computes all the

intermediate coordinates through which our robot must traverse to enable it to navigate between road turns to reach its destination. The pyroute library has been altered so that before returning control to the calling function i.e. pyserial it transmits all the computed coordinates back to our arduino. The pyserial library then calls pymysql library which stores our user and cabs coordinates to our database from which we will fetch it later.

6.1.2 Pyroute

Pyroute is a routing program written in Python by Ojw. In pyroute we can add any map of our choice and then it uses its routing capabilities to compute coordinates between any two input locations. Each intersection of roads is called a node in python and has its own associated coordinates range. On inputting a map an OSM file is generated in which all nodes and their corresponding coordinates are saved. The map was downloaded from *OpenStreetMap*. One of the drawbacks of this library was that it took input in form of node number thus for the locations between which the user wants navigation coordinates the user has to first read the nodes which correspond to the these locations and then input them to the library. Thus we altered the library so that we can directly provide coordinates instead of nodes. For this purpose we wrote a code which converted these input coordinates to nodes by reading from our map's OSM file. Furthermore we also wrote inserted another script which serially transmits all the coordinated that have been computed.

6.1.3 PymySQL

PyMySQL is a database connector for Python. It is a librarythat enables Python programs to talk to a MySQL server. We have utilized this library to save our user and cabs location into a database from which these coordinates can be read and displayed on Google maps.

6.2 MySQL Database

MySQL is the world's second most used open source relational database management system (RDBMS), and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line toolsor use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use. This was our database choice for our project and where we stored our coordinates. Each new coordinate was stored in a new location according to its type i.e. if the coordinate was of user it was declared type zero and one if it was of cab.

6.3 PHP

PHP code is interpreted by a web server with a PHP processor module, which generates the resulting web page. PHP commands can be embedded directly into an HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone graphical applications. For our purpose we have used PHP script to read data from a

database and transmit it to Google maps. The coordinates read from our database are input to the functions which are displayed on google maps.

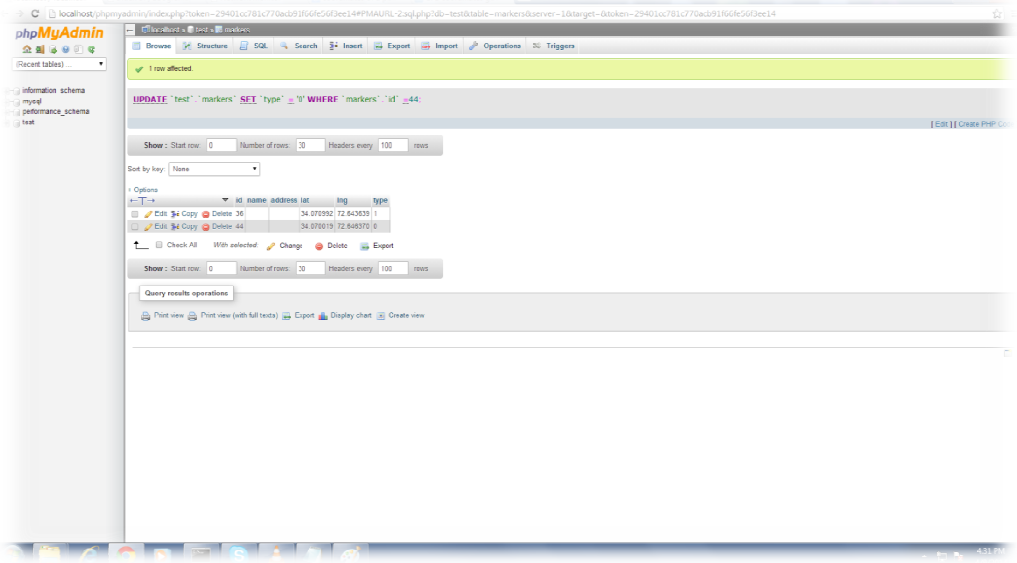

Figure 6.2: PHP

6.4 Google Maps

Google Maps is a desktop and mobile web mapping service application and technology provided by Google, offering satellite imagery, street maps, and Street View perspectives, as well as functions such as a route planner for traveling by foot, car, bicycle (beta test), or with public transportation. Also supported are maps embedded on third-party websites via the Google Maps API,[1] and a locator for urban businesses and other organizations in numerous countries around the world.. For our purpose we input the coordinates by reading from our database. The location is regularly updated by reading from the database repeatedly.

Chapter 7

7 Applications

The project has multiple applications and its applications can be divided according to its modules. Firstly it can be used as an autonomous cab dispatch system within a city by using its central controller and GUI modules but the autonomous robot can also be used in a large number of other applications. The projects applications are summarized below.

7.1 Autonomous cab dispatch system:

One of our projects major applications is its use as an autonomous cab dispatch system. The taxi dispatch problem involves assigning taxis to callers waiting at different locations. In such an application our system can be deployed to provide enhanced fuel and time efficiency. Furthermore being autonomous and intelligent utilizing our system in such a system will also require very few operators whose interference would be negligible. The system is also expandable as any map can be downloaded and used as well as constantly updated for any city of the world.

7.2 Autonomous Robot

The robot too has multiple applications and can as such be used for surveillance, maintenance and other purposes. The robot has the capability to navigate itself according to a predefined map which we can design according to our specifications which gives it flexibility to be utilized in a large number of environments. Its body

size is changeable along with its speed by utilizing different motors thence its body can be tailored according to our application requirements. Its major applications are:

7.2.1 Surveillance

The robot can be used for surveillance purposes as well. A camera can be installed on it and it can send back real time feedback of its environment and thence it can in a way be utilized as a drone or can be used for the Google car.. As its route is easily definable it can be used in multiple locations for different routes, furthermore due to its compact design it can also be used in homes and offices for monitoring purposes.

7.2.2 Office Bot

This type of vehicle can be used in a relatively structured environment to perform redundant tasks such as handling office documentation, delivering TCS and other similar tasks to reduce the cost of hiring human capital for such tasks. Furthermore due to its route calculation abilities map of any floor can be used for the robot.

7.2.3 Maintenance Robot

As mentioned above some of the ideas incorporated for the building of this robot were taken from the sewerage system maintenance robot K.U.R.T as such this robot can also be used for the maintenance of sewerage system and air ducts. In such a system the routing capability will once again serve as an advantage however the body and motors power of the robot can be altered to give maximum battery time or make the robot more robust and suitable for its environment.

7.2.4 Geological

In hazardous places such as volcanoes, where its nearly impossible for humans to go inside in order to check the state of the volcano to check whether its dormant, active or extinct our robot can be used. The loss of such a robot is obviously nothing compared to the possible loss of a human life. Furthermore the robot can also be used in mining to take samples from creeks crevices in caves before bombarding to ensure that resources exist.

7.2.5 Spying Purposes

Areas where it would be harmful for humans to go the robot can be utilized thence it can also be used for spying purposes. In such cases as all data can be wirelessly communicated back to its central controller and as the robot has no data within it is safe to spy at a reasonably low price. It can be especially used to spy within military camps as its small size will make spotting it difficult.

7.2.6 Entertainment

In the gaming arena, the robot can act as a maze solver or can be used as a racecar or for such other purposes. In such instances its power capability will come to great advantage. Furthermore it can be used in various robot matches as well for entertainment purposes.

Chapter 8

8 Troubleshooting and problems

A lot of debugging and troubleshooting was required in this project due to interlinking of various platforms and the hardware of the robot involved. The more common problems are discussed below:

1. The transistor kept heating up at an escalated rate as the robot was tested on road and the temperature there was especially. We employed heat sinks but still our purpose was not served. So we made a larger heat sink by combining two heat sinks to prevent heating up of the robot.
2. Initially the GPS we used was of holux technologies, was highly inaccurate and did not provide the efficiency required for our purposes. So we used another GPS of ubloxx which provided much better results.
3. Even with ubloxx sometimes the coordinate values did not change soon enough so we employed a coding check and in such scenarios refreshed the GPS power. This was a solution we accidentally came across and it greatly helped improve our robots maneuvering capabilities.
4. We also found out that even after employing such measures the heading being received from GPS was not very accurate. So we interfaced a magnetic compass to prevent this problem causing detrackof our robot.

5. Another problem we faced was the improper weight distribution of the robot which caused it to move towards one side or rather turn than go straight as it was supposed to. To address this issue we balanced our motors by coding our arduino so that our robot was able to go straight.
6. The python libraries available at python forum were not properly debugged and some bugs had to be removed from the library itself.
7. As we used arduino and python to serially communicate, another one of the issues was that of character types being sent by arduino not being the one that python read. So in such cases we employed character and string conversions.

Appendices

A1 Budget

Equipment	Quantity	Estimated Cost(in PKR)
Arduino Mega 2560 R3	2	2500x3 =5000
Arduino GSM Shield	1	8000x1 =8000
GPS Module (M8929)	2	2500x2 =5000
GSM Module(SIM 900)	1	4500
12V, 2A battery	1	1800x1 =1800
Robots Hardware	1	1x3000 =3000
Miscellaneous	-	5000
Total Cost	**-**	**=32300/- (approx.)**

Table A1: Budget

A2 Arduino Uno

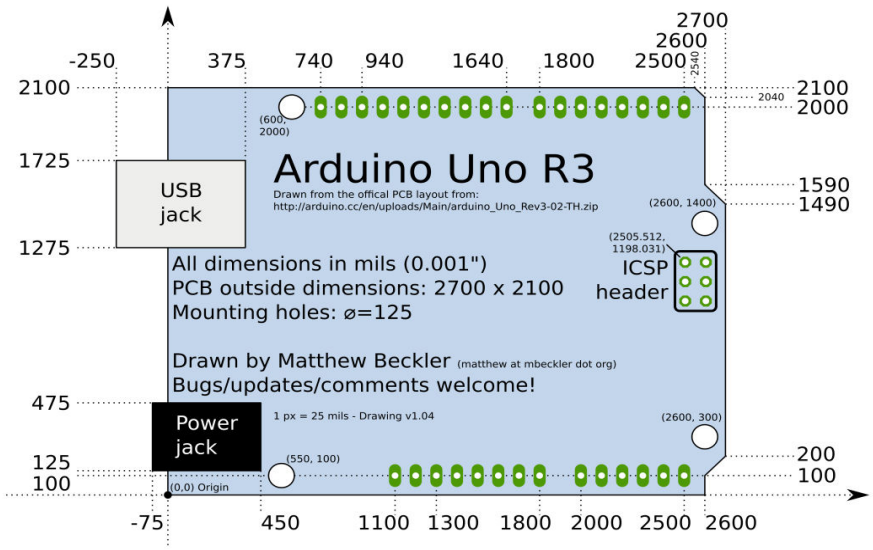

Figure A2 (a): Arduino Uno Block Diagram

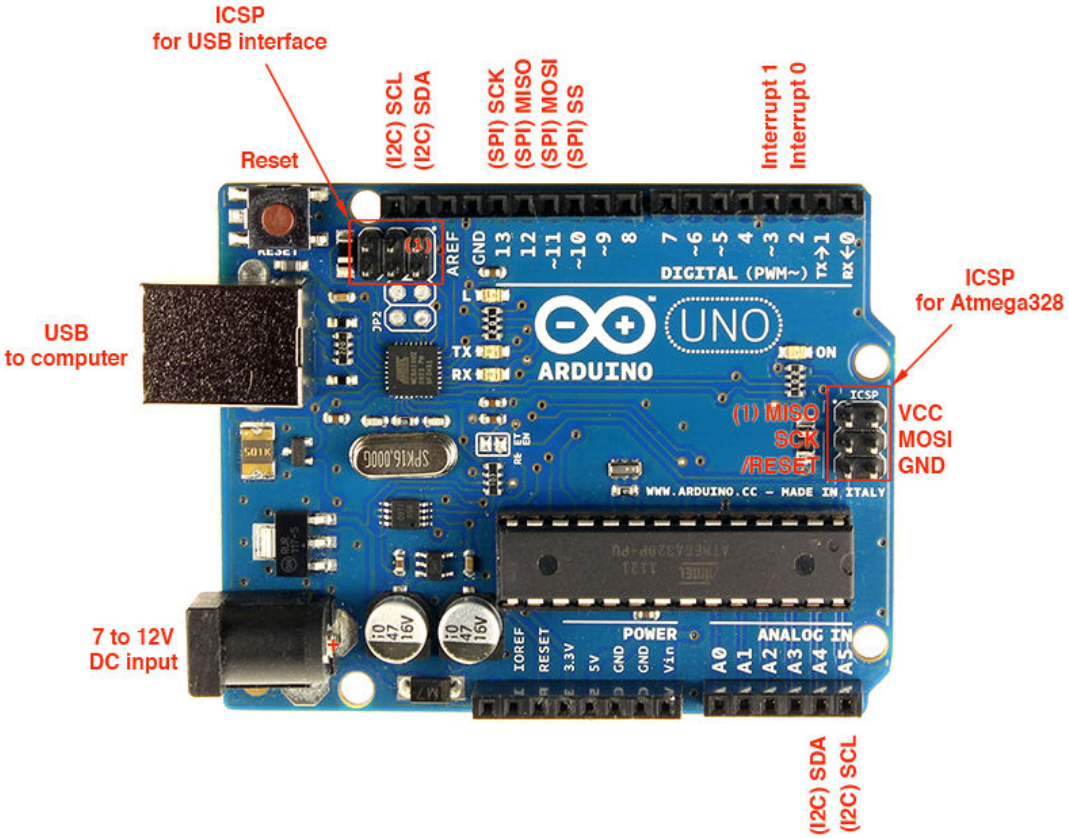

Figure A2 (b): Arduino Uno Pin Configuration

A3 Arduino Mega 2560

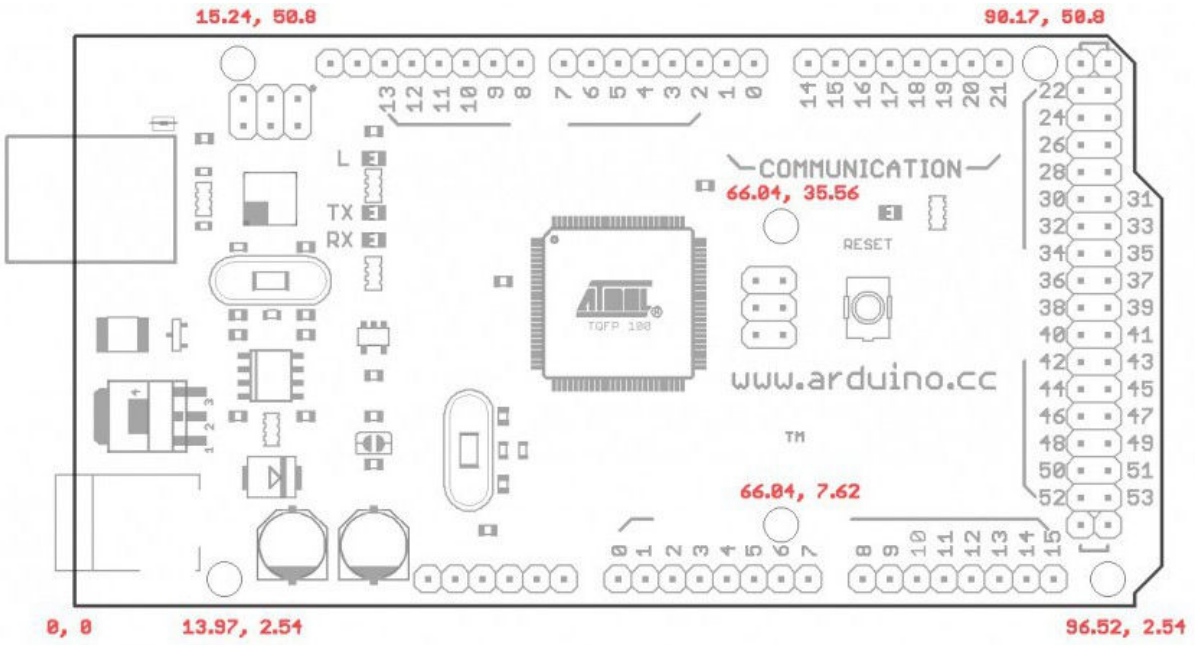

Figure A3: Arduino Mega 2560 Circuit

References

[1] Douglas W. Gage 1995, *Unmanned Systems Magazine*, 13(3); 1-8, March 11.

[2] J. Borenstein 1, H.R. Everett2, L. Feng3, and D. Wehe4 1999. *Journal of Robotic Systems*, 14(4); 231-249.

[3] Ibrahim Kamal 2007. Project; *Build a 5A H-Bridge Motor Controller* [Online]. Available; http;//www.ikalogic.com/projects.php[4 February 2007]

[4] Martin Lundgren, "Path Tracking and Obstacle Avoidance for a Miniature Robot", 2003.

[5] "The Basics of Automated Guided Vehicles". AGV Systems. Savant. 5 March 2006

[6] AGV Drive and Steering Options Transbotics Corp., 2009.