Design And Implementation of Self-Driving Car Using Arduino And GPS System

A Research submitted in partial fulfilment for the requirements of the degree of B.Sc (Honor) in Electrical Engineering

Submitted by:

ADAM HASSAN MOHAMMED ALI
AMAAL HUSSEIN IBRAHIM MOHAMMED
MAHER SAAD SAAD ALSAREB
SAFA ALFADOUL MOHAMMED ABUHEEN

Supervised by:

Dr. MUGAHID OMER HAJ-ALTOM

Department of Electrical Engineering

Faculty of Engineering

Elshaikh Abdallah Elbadri University

Jan 2021

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيم

﴿اللَّهُ لَا إِلَهَ إِلَّا هُوَ الْحَيُّ الْقَيُّومُ لَا تَأْخُذُهُ سِنَةٌ وَلَا نَوْمٌ لَهُ مَا فِي السَّمَاوَاتِ وَمَا فِي الْأَرْضِ مَنْ ذَا الَّذِي يَشْفَعُ عِنْدَهُ إِلَّا بِإِذْنِهِ يَعْلَمُ مَا بَيْنَ أَيْدِيهِمْ وَمَا خَلْفَهُمْ وَلَا يُحِيطُونَ بِشَيْءٍ مِنْ عِلْمِهِ إِلَّا بِمَا شَاءَ وَسِعَ كُرْسِيُّهُ السَّمَاوَاتِ وَالْأَرْضَ وَلَا يَئُودُهُ حِفْظُهُمَا وَهُوَ الْعَلِيُّ الْعَظِيمُ٢٥٥﴾ [البقرة: 255]

صَدَقَ اللَّهِ الْعَظِيمِ

Acknowledgement

In the name of Allah, Most Gracious, Most Merciful.

We would like to express our gratitude to our primary supervisor, Dr. Mugahid Omer Haj-altom,who guided us throughout this project.

We would like to thank our family who supported us and offered deep insight into the study.

We would like to thank several individuals for their support, motivation, and inspiration to made this project done.

Thanks to our classmates and friends for every advice they gave for the sake of improving this project.

We wish to acknowledge the help provided by the technical and support staff in the Electrical Engineering department of the University of ElSheikh Abdallah ElBadri.

Abstract

Autonomous vehicles are virtually regarded as the panaceas for the future of road transport due to numerous promising benefits as safety, freedom of time, reduce traffic crowd and helping people with special needs for traveling. Thus, they have attracted wide attention from both academia and industry. Although associated technologies have been investigated and developed for decades, several obstacles still need to be overcome. One of the major obstacles is the total cost of the needed sensors. The main focus of this research is to design and implement a self-driving car prototype with inexpensive sensors that can do the main purpose. Arduino was used as a microcontroller to control the devices in this design. Tow DC motors were used to control moving which one control the direction and another control the movement. Different sensors were used to detect the position and direction and displayed these information on smart phone. Avoid collusion obstacles system was designed and implemented. The prototype was tested and it provided a very good results.

المستخلص

تعتبر سيارة التحكم الذاتي هي مستقبل مجال النقل والمواصلات وذلك بسبب ما تقدمه من فوائد كثيرة كالسلامة والراحة وتوفير الوقت والمجهود مما أدت الى جذب انتباه كبير من الجانب الأكاديمي والصناعي. وبالرغم من الاستثمار والتطوير في التكنولوجيا المتعلقة بالتحكم الذاتي لعقود كثيرة تظل هناك الكثير من المشاكل والعوائق التي تحتاج الى تقديم حل. أحد هذه العوائق الرئيسية هو التكلفة المطلوبة للحساسات والقطع المستخدمة. تم العمل في هذا البحث على تقديم نموذج لسيارة تحكم ذاتيه تقوم بالعمليات الأساسية وذلك باستخدام قطع ذات تكلفة منخفضة. حيث تم استخدام الأردوينو كوحدة معالجة تقوم بعمليات التحكم المختلفة لجميع العناصر المكونة للنموذج. وتم استخدام محركين تيار مستمر منفصلي التحكم حيث يقوم أحدهما بالتحكم في اتجاه السيارة والأخر بالتحكم في الحركة. كما تم استخدام حساسات مختلفة لتحديد اتجاه وموقع السيارة وكذلك تم تصميم نظام لتفادي الاصطدام بالحواجز وعرض بعض المعلومات المهمة على التلفون الذكي. تم اختبار هذا النموذج وقدم نتائج جيدة جداً في التنقل بين نقطتين معلومتين مع تفادي الاصطدام بالعوائق.

Table of contents

Chapter One- Introduction

Chapter Two - Background and Literature Review

Chapter Three-System Design

Chapter Four -Results and Discussion

Chapter Five –Conclusion and Recommendations

List of Figures

List of Tables

Abbreviation list

AoA	Angle of Arrival
APIs	Application Program Interfaces
APS	Ad Hoc Positioning System
DC	Direct Current
DV	Distance Vector
GNSS	Global Navigation Satellite System
GPRS	General Packet Radio Service
GPS	Global Position System
I/O	Input/Output.
IDE	Integrated Development Environment
INS	Inertial Navigation Systems
LIDAR	Light Detection and Ranging
MEMS	Microelectromechanical
MCS	Master Control Station.
OSRM	Open Source Routing Machine
PPS	Precise Positioning Service
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
SD	Secure Digital
SPS	Standard Positioning Service
TDoA	Time Difference of Arrival
TOA	Time of Arrival

Chapter One

INTRODUCTION

Chapter One
INTRODUCTION

1-1 Overview

Nowadays, many people are using the vehicles for traveling this causes many accidents happened .According to the world health organization, every year the lives of approximately 1.35 million people are cut short as a result of a road traffic crash. Between 20 and 50 million, more people suffer non-fatal injuries, with many incurring a disability as a result of their injury [1]. Road traffic injuries cause considerable economic losses to individuals, their families, and to nations as a whole. These losses arise from the cost of treatment as well as lost productivity for those killed or disabled by their injuries, and for family members who need to take time off work or school to care for the injured. Road traffic crashes cost most countries 3% of their gross domestic product [1]. Autonomous vehicles, also known as self-driving cars, belong to outdoor mobile robots. They employ many different types of sensors for localization and perceiving their environments. In addition, they depend on the embedded system to perform autonomous driving tasks. As a long-term goal, such vehicles will replace human drivers to drive passengers to their destinations. The widespread embrace of autonomous vehicles is widely believed to be a solution to reduce the rate of traffic accidents. This belief stems from the statistics data related to car accidents. The statistics results show that the leading cause of about 90% of reported car accidents results from human errors. Such errors are not limited to novice drivers but also including trained and experienced experts [2]. Such driving errors include drunk driving, fatigue driving, slow reaction time in emergencies and so on. Since most of such traffic accidents are induced by human errors, autonomous vehicles could eventually reduce the rate of vehicle related deaths. They will eventually eradicate vehicle related accidents. The mass adoption of autonomous vehicles has numerous benefits as: Safety, freedom of time, reduce traffic crowd and help people with special need for traveling [3].

1-2 Problem Formulation

Cars are the most common way to travel in world today. The risks are exist in traffic due to human mistakes, working for more comfortable ,decreasing the congestion and safe time are the motivations for this research .

1-3 Research Objectives

The main objective of this research work is to build and realize self-driving vehicle using elect-ronic elements. The specific objectives of this research are illustrated in the following points:

- To design and implement control circuit for navigation and detection direct and distance to waypoint .using Arduino, GPS chip and google earth application.
- To design and implement a movement system for moving, controlling and avoiding the collision with obstacles using DC motors.
- To design and implement a display system for important information using Bluetooth and smart phone.

1-4 Research Methodology

The main idea of this work is to design autonomous car according to navigation system. Firstly, Arduino has waited until GPS was read. Then, the distance between the car and obstacles was detected by the two sensors of ultrasonic. When the distance is short the car will avoid collision with obstacle. After that, SD card was read that stored points of wanted path from google earth. Then, DC motors were controlled according to comparison between the coordinates of the actual location and next waypoint of path. Finally, coordinates, distance to the next waypoint and distance from obstacles will be send to phone by Bluetooth.

1-5 Research Structure

Chapter One: Covers the research Introduction, problem formulation, objectives, and also the methodology followed herein.

Chapter Two: Covers the background material, literature and different surveys to understand the fundamentals of localization for self-driving car.

Chapter Three: Contains description of hardware used in the system, hardware design, circuit and hardware connections.

Chapter Four: This chapter provides the stages and steps by which the proposed system can be implemented and turned into being a practical model.

Chapter Five: The conclusions of the project and the future developments are suggested for better plan and execution of this project.

Chapter Two

BACKGROUND AND LITERATURE REVIEW

Chapter Two

BACKGROUND AND LITERATURE REVIEW

2.1 Overview

The usage and production of cars has become a leading industry in almost every area of the world. The world's car stock exceeded 80 million after the Second World War, then more than 90 million in 1960. Five years later this number was 130 million, 291 million in 1980, 419 million in 1990, and 731 million in 2011. According to the forecasts, it will reach two billion by 2020 [4]. Over the years and centuries, this industry has gone through enormous development, as the first vehicles were only powered by steam engine, then petrol and diesel came to public mind and currently it seems that the electric propulsion will be the future. Of course, with this development, faster and more useful vehicles can be produced, but in our accelerated world with more and more cars, unfortunately the numbers of accidents have increased. In most cases, these accidents are the fault of the driver. Therefore, it could be theoretically replaceable with the help of self-propelled cars. Human presence is the most important part in transport at present, although there are many areas where you can use a tool or feature that helps people achieve greater efficiency. Some examples for these features are the autopilot on aircraft, the cruise control in cars, and many other tools that help decision-making [5]. For 125 years, the automotive industry has been a force for innovation and economic growth. Now, in the early decades of the 21st century, the pace of innovation is speeding up and the industry is on the brink of a new technological revolution: "self-driving" vehicles. The new technology could provide solutions to some of our most intractable social problems the high cost of traffic crashes and transportation infrastructure, the millions of hours wasted in traffic jams, and the wasted urban space given over to parking lots, just to name a few. However, if self-driving vehicles become a reality, the implications would also be profoundly disruptive for almost every stakeholder in the automotive ecosystem [6]. Two decades ago, when it came to autonomous driving, people must think it only existed in the science fiction. Now it has become a reality and self-driving cars are gradually becoming consumer cars in the near feature. Especially after the driving has entered the public view and attracted interest from the public and companies alike. With the effort of the Google Self-Driving project, many research groups and other car manufacturers, autonomous driving has already achieved remarkable advances. However, many critical challenges still need to be overcome. From the technical perspective, the autonomous driving related technologies are not so reliable [7]. From the cost perspective, main sensors are still expensive in comparison to the cost of the vehicles. From the legal view, few states in the United States have passed legislation and laws to allow testing autonomous vehicles on public roads. To realize fully autonomous driving, therefore, it still has a long way to go. This dissertation focuses on addressing the fundamental problem of autonomous driving, localization. Most state of the art autonomous test vehicles are equipped with commercial high accuracy GPS-based Inertial Navigation Systems (INS), such as Applanix POS LV. Such systems are quite expensive in comparison with their platform vehicles. This hinders the mass adoption and the development of autonomous vehicles. Therefore, our purpose here is to employ fewer and cheaper sensors to get the same order of magnitude of accuracy as that of commercial systems. To realize this goal, a feature-map based scheme has been successfully implemented. Thousands of times of tests and real on-road tests verified its effectiveness and robustness. [8]

2.2 Evaluation of Self–Driving Cars

Autonomous cars are those vehicles which are driven by digital technologies without any human intervention. They are capable of driving and navigating themselves on the roads by sensing the environmental impacts. Their appearance is designed to occupy less space on the road in order to avoid traffic jams and reduce the likelihood of accidents. Although the progression is gigantic, in 2017, allowed automated cars on public roads are not fully autonomous: each one needs a human driver who notices when it is necessary to take back the control over the vehicle [9]. The dream of self-propelled cars goes back to the middle ages centuries before the invention of the car. An evidence for this statement comes from sketches of Leonardo De Vinci, in which he made a rough plan of them. Later, in literature and in several science fiction novels, the robots and the vehicles controlled by them, appeared. The first driverless cars were prototyped in the 1920s, but they looked different than they are today. Although the "driver" was nominally lacking, these vehicles relied heavily on specific external inputs. One of these solutions is when the car is controlled by another car behind it. Its prototype was introduced in New York and Milwaukee known as the American Wonder" or "Phantom Auto". CED in New York and Milwaukee known as the American Wonder" or "Phantom Auto" [10].

2.2.1 Autonomous Vehicles And Car Companies:

Most of the big names Mercedes Benz, Audi, BMW, Tesla, Hyundai etc. Have begun developing or forming partnerships around autonomous technology. They invested sizable resources into this, and by making this step, they wanted to be leaders at the market of self-driving cars.

Up to this point, numerous aids, software and sensors have been put into these cars, but we are still far from full autonomy. They use lasers that are testing the environment with the help of LIDAR ((Light Detection and Ranging). This optical technology senses the shape and movement of objects around the car; combined with the digital GPS map of the area, they detect white and yellow lines on the road, as well as all standing and moving objects on their perimeter. Autonomous vehicles can only drive themselves if the human driver can take over the control if needed. These are some features that driverless cars already use [11]:

•Collision avoidance

•Drifting warning

•Blind-spot detectors

•Enhanced cruise control

•Self-parking

Below we briefly present some companies that play the most important role in the innovation of this segment, to show how this industry has developed.

- **Tesla :**

Elon Musk, the Chief Executive Officer of Tesla, claims that every Tesla car will be completely autonomous within two years. The Tesla's "S" model is a semi-self-propelled car, where different cars are able to learn from each other while working together. The signals processed by the sensors are sent to other cars thus they can develop each other. This information teaches cars about changing lanes and detecting obstacles, and are continually improving from day to day. From October 2016, all Tesla vehicles have been being built by Autopilot Hardware 2, with a sensor and computing package that the company claims to allow complete self-driving without human interference [12].

- **Google :**

The Google team has been working on driverless cars for years, and last year a working prototype was presented (by them). Furthermore, Google also supports other car manufacturers with self-driving car technologies such as Toyota Prius, Audi TT, and Lexus RX450h. Their own autonomous vehicle uses Bosch sensors and other equipment manufactured by LG and Continental companies. In 2014, Google planned a driverless car that would be available without pedals and wheels to make it available to the general public by 2020, but according to the current trends, its fulfilment is still unlikely [12]**.**. Figure (2.1) shows some these products.

- **nuTonomy :**

A Small group of graduates of the Massachusetts Institute of Technology (MIT) created the nuTonomy software and algorithm especially to self-propelled cars. In Singapore, nuTonomy has already put sensors to the Mitsubishi i-MiEV electric car prototype, thus nuTonomy algorithms can control the car on these complex urban roads by using GPS and LiDAR sensors. Besides that, in November 2016, they announced that self-propelled cars would be tested in Boston as well [12].

(a)

(b)

Figure (2.1) The Google Self-driving Car project. (a)A row of Google Self-driving cars (b) The prototype of Google self-driving car**.**

2.3 Types of Autonomous Vehicles

The National Highway Traffic Safety Administration (NHTSA) adopted the levels of the Society of Automotive Engineers for automated driving systems, which provides a broad spectrum of total human participation to total autonomy [13].

NHTSA expects automobile manufacturers to classify each vehicle in the coming years using SAE 0 to 5 levels. As shown in figure (2-2) these are the levels of SAE:

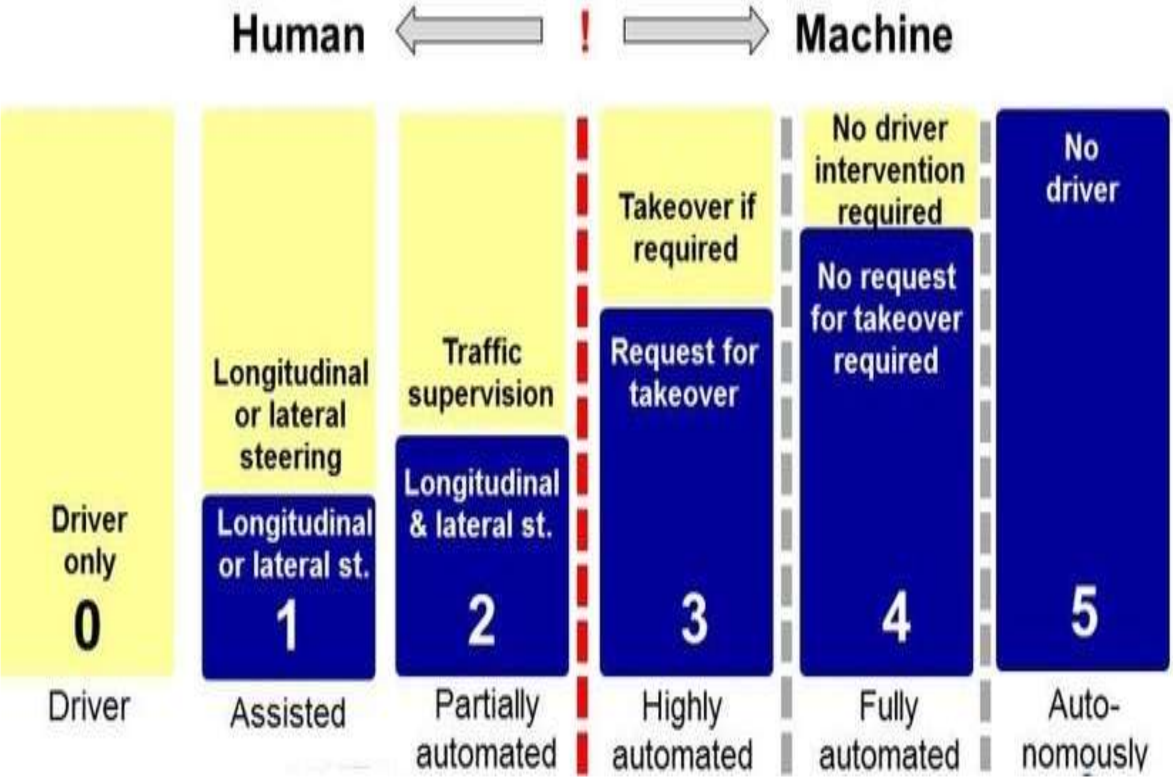

Figure (2-2) Types of autonomous vehicles

Level 0: No Automation

In this case, there is 100% of human presence. Acceleration, braking and steering are constantly controlled by a human driver, even if they support warning sounds or safety intervention systems. This level also includes automated emergency braking.

Level 1: Driver Assistance

The computer never controls steering and accelerating or braking simultaneously. In certain driving modes, the car can take control of the steering wheel or pedals. The best examples for the first level are adaptive cruise control and parking.

Level 2: Partial Automation

The driver can take his hands off the steering wheel. At this level, there are set-up options in which the car can control both pedals and the steering wheel at the same time, but only under certain circumstances. During this time, the driver has to pay attention and if it is necessary, intervene. This is what Tesla Autopilot was known for 2014.

Level 3: Conditional Automation

It approaches full autonomy, but this is dangerous in terms of liability, so therefore, paying attention to them is a very important element. Here the car has a certain mode that can take full responsibility for driving in certain circumstances, but the driver must take the control back when the system asks. At this level, the car can decide when to change lanes and how to respond to dynamic events on the road and it uses the human driver as a backup system.

Level 4: High Automation

It is similar to the previous level, but it is much safer. The vehicle can drive itself under suitable circumstances, and it does not need human intervention. If the car meets something that it cannot handle, it will ask for human help, but it will not endanger passengers if there is no human response. These cars are close to the fully self-driving car.

Level 5: Full Automation

At this level, as the car drives itself, human presence is not a necessity, only an opportunity. The front seats can turn backwards so passengers can talk more easily with each other, because the car does not need help in driving. All driving tasks are performed by the computer on any road under any circumstances, whether there's a human on board or not.

These levels are very useful as with these we can keep track of what happens when we move from human-driven cars to fully automated ones. This transition will have enormous consequences for our lives, our work and our future travels.

As autonomous driving options are widespread, the most advanced detection, vision, and control technologies allow cars to detect and monitor all objects around the car, relying on real-time object measurements. In addition, the information technology built into the vehicle is fully capable of delivering both external (field) and internal (machine) information to the technology built into the vehicle is fully capable of delivering both external (field) and internal (machine) information to the car [14]

2.4 Localization for Autonomous Driving

The exteroceptive sensors aim to measure information from the vehicle environment. Most of the exteroceptive sensors are used for localization relative to the immediate surroundings of the vehicle, although via feature extraction some of the relative localization sensors can also be used for absolute localization [15].

2.4.1 Global Navigation Satellites:

The development of global navigation satellite systems was initially motivated by defense applications. Currently, these systems are widely used in commercial applications for positioning, navigation, and tracking.

Absolute geo spatial positioning information obtained through the use of global navigation satellite systems, such as global positioning system (GPS) and globalnaya navigatsionnaya sputnikovaya Sistema (GLONASS) can augment other sensors to perform relative localization [16].

2.4.2 Range Sensors:

There is a variety of range sensors commercially available to enable different driver assistant functionalities for smart and autonomous vehicles.

This section presents laser and radar systems, which are the most commonly used range sensors. The use of a laser scanner robustly provides the vehicle with information about what is ahead [17]. Laser scanner-based sensors emit light impulses of electromagnetic waves and use time of arrival (TOA) to determine the range and bearing of an object. A laser-based range sensor consists of a light source, which emits collimated light beams, and a receiver, which receives the light reflected from surrounding objects. Often this type of sensor is referred to as LiDAR (light detection and ranging). There are two-dimensional (2-D) and three-dimensional (3-D) LiDAR sensors that all work on the same principle of TOA, but the ways to measure the distance vary among the manufactures. Most of the LiDAR sensors collect the light beam using an oscillating mirror for 2-D and a rotating mirror for 3-D. Radar sensors are LiDAR-like sensors that utilize radio waves instead of light. They are generally used for obstacle detection. The radio and light waves travel at the same speed, but the radio waves have much lower frequencies. To measure the distance to an object, radar sensors estimate signal TOA. Unlike LiDAR, the radar takes into account the Doppler shift of the echo; hence, it can observe the velocity of the moving object without further numerical processing [18].

Vision Sensors: The prototypes and commercialized computer vision technology have been developed as the key components of autonomous driving and intelligent vehicles. The applications of computer vision include lane detection and tracking road sign detection and understanding, traffic light detection, and many more. Originally, there were challenges in using vision for localization, since vision transforms the 3-D world into 2-D.Various techniques are proposed to recover 3-D features from images. However, this does not address the problem of high computational complexity with vision sensors [19].In this work, we used GPS navigation for localized the car because it has some advantages .it is low cost, easy and available

.in other hand, it has low precision, long range for error, disruption and outdoor usage.

2.5 Global Positioning System (GPS)

Concept of GPS started in the early 1970s as a way for the United States military to accurately identify locations throughout the globe.

Between 1978 and 1985, the DoD launched the first generation of GPS satellites exclusively for military use. However, in 1983, President Ronald Reagan decided to permit civilian use of GPS technology once it became operational. The first civilian uses of GPS were primarily in the realm of aviation and surveying (Rand Corporation).

In 1995, the second generation of GPS satellites became fully operational and commercial civilian use began to be more fully explored, with the first instances of GPS technology specifically designed for tracking humans surfacing a few years later [20].

2.5.1 GPS segments

GPS technology can be described in terms of three segments:

Space Segment. Consists of twenty-four satellites orbiting 11,000 nautical miles above the earth. Control Segment. Consists of five ground stations around the globe that manage the operational health of the satellites by transmitting orbital corrections and clock updates.

User Segment. Consists of various types of GPS receivers that can vary in complexity and sophistication. This segment is what most people are familiar with; examples include the navigation system in a car, or the GPS device in a cell phone.

GPS receivers are able to identify their location when three GPS satellites triangulate and measure the distance to the receiver and compare the measurements. A fourth satellite measures the time to the receiver. The information from all four satellites is compiled to determine the location. The sophistication of a GPS receiver affects the reliability and accuracy of the GPS data received), two digital codes, and navigation message. The codes and the navigation message are added the carriers as binary biphase modulation .the carriers and the codes are send mainly to determine the distance from the user's receiver to the GPS satellites. The navigation message contains. Along with other information, the coordinates of the satellites as a function of time. The transmitted signals are controlled by highly accurate atomic clocks onboard the satellites [21].

2.6 Methods of Localization

2.6.1 Ranging Techniques

The foundation of numerous localization techniques is the estimation of the physical distance between two sensor nodes. Estimates are obtained through measurements of certain characteristics of the signals exchanged between the sensors, including signal propagation times, signal strengths, or angle of arrival [23].

2.6.1.1 Time of Arrival

The concept behind the time of arrival (ToA) method (also called time of flight method) is that the distance between the sender and receiver of a signal can be determined using the measured signal propagation time and the known signal velocity. For example, sound waves travel 343 m/s (in 20 ◦C), that is, a sound signal takes approximately 30 MS to travel a distance of 10 m. In contrast, a radio signal travels at the speed of light (about 300 km/s), that is, the signal requires only about 30 ns to travel 10 m. The consequence is that radio- based distance measurements require clocks with high resolution, adding to the cost and complexity of a sensor network. The one-way time of arrival method measures the one-way propagation time, that is, the difference between the sending time and the signal arrival time (Figure 2-3(a)), and requires highly accurate synchronization of the clocks of the sender and receiver.

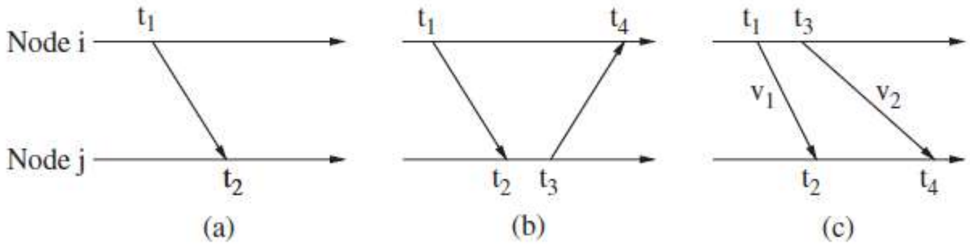

Figure (2-3) Comparison of different ranging schemes (one-way ToA, two-way ToA, and TDoA).

Therefore, the two-way time of arrival method is preferred, where the round-trip time of a signal is measured at the sender device (Figure 2-3(b)). In summary, for one-way measurements, the distance between two nodes i and j can be determined as:

$$\mathrm{dist}_{ij} = (t_2 - t_1) \times v \qquad (2\text{-}1)$$

Where t1 and t2 are the sending and receive times of the signal (measured at the sender and receiver, respectively) and v is the signal velocity. Similarly, for the two-way approach, the distance is calculated as:

$$\mathrm{dist}_{ij} = (t_4 - t_1) - (t_3 - t_2) \times v \qquad (2\text{-}2)$$

Where t3 and t4 are the sending and receive times of the response signal. Note that with one-way localization, the receiver node calculates its location, whereas in the two-way approach, the sender node calculates the receiver's location. Therefore, a third message will be necessary

in the two-way approach to inform the receiver of its location [23].

2.6.1.2 Time Difference of Arrival

The time difference of arrival (TDoA) approach uses two signals that travel with different velocities (Figure 2-3(c)). The receiver is then able to determine its location similar to the ToA approach. For example, the first signal could be a radio signal (issued at t1 and received at t2), followed by an acoustic signal (either immediately or after a fixed time interval $t_{wait} = t_3 - t_1$

Therefore, the receiver can determine the distance as:

$$dist_{ij} = (v_1 - v_2) \times (t_4 - t_2 - t_{wait}) \qquad (2\text{-}3)$$

TDoA-based approaches do not require the clocks of the sender and receiver to be synchronized and can obtain very accurate measurements. The disadvantage of the TDoA approach is the need for additional hardware, for example, a microphone and speaker for the above example. Another variant of this approach uses TDoA measurements of a single signal to estimate the location of the sender using multiple receivers with known locations. The propagation delay di for the signal to receiver i depends on the distance between sender and receiver i. Correlation analysis can then provide a time delay δ di dj which corresponds to the difference in path length to receivers i and j (Gustafsson and Gunnarsson 2003). The main disadvantage of this approach is that the clocks of the receivers must be tightly synchronized [24].

2.6.1.3 Angle of Arrival

Another technique used for localization is to determine the direction of signal propagation, typically using an array of antennas or microphones. The angle of arrival (AoA) is then the angle between the propagation direction and some reference direction known as orientation (Peng and Sichitiu 2006). For example, for acoustic measurements, several spatially separated microphones are used to receive a single signal and the differences in arrival time, amplitude, or phase are used to determine an estimate of the arrival angle, which in turn can be used to determine the position of a node. While the appropriate hardware can obtain accuracies within a few degrees, AoA measurement hardware can add significantly to the size and cost of sensor nodes[24].

2.6.1.4 Received Signal Strength

The concept behind the received signal strength (RSS) method is that a signal decays with the distance traveled. A commonly found feature in wireless devices is a received signal strength indicator (RSSI), which can be used to measure the amplitude of the incoming radio signal. Many wireless network card drivers readily export RSSI values, but their meaning may differ from vendor to vendor and there is no specified relationship between RSSI values and the signal's power levels. Typically, RSSI values are in the range of (0 ... RSSI_Max), where common values for RSSI_Max are 100, 128, and 256. In free space, the RSS degrades with the square of the distance from the sender. More specifically, the Friis transmission equation expresses the ratio of the received power Pr to the transmission power Pt as:

$$\frac{p_1}{p_2} = G_t G_r \frac{\lambda^2}{(4\pi)^2 R^2} \qquad (2\text{-}4)$$

where G_t is the antenna gain of the transmitting antenna and G_r is the antenna gain of the receiving antenna. In practice, the actual attenuation depends on multipath propagation effects, reflections, noise, etc., therefore a more realistic model replaces R^2 in Equation (2-4) with R_n with n typically in the range of three and five [25].

2.6.2 Range-Based Localization

2.6.2.1 Triangulation

Triangulation uses the geometric properties of triangles to estimate sensor locations. Specifically, triangulation relies on the gathering of angle (or bearing) measurements as described in the previous section. A minimum of two bearing lines (and the locations of the anchor nodes or the distance between them) are needed to determine the location of a sensor node in two-dimensional space. Figure (2-4(a)) illustrates the concept of triangulation using three anchor nodes with known locations (x_i,y_i) and measured angles α_i (expressed relative to a fixed baseline in the coordinate system, for example, the vertical line in the figure).

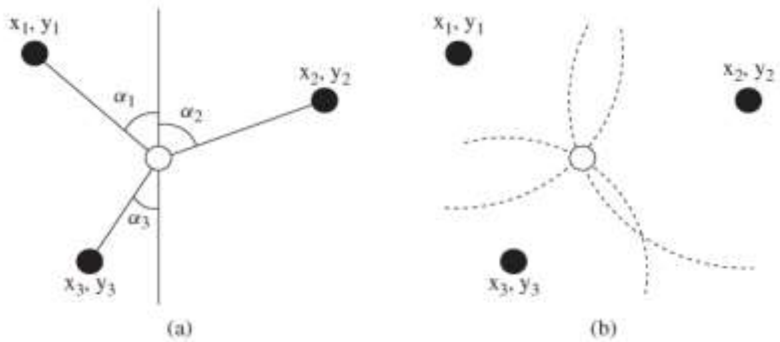

Figure (2-4): distance estimation methods (a) Triangulation (b) trilateration .

If more than two bearings are measured, the presence of noise in the measurements may prevent them from intersecting in a single point. Therefore, statistical algorithms or fixing methods have been developed to obtain a single location (Stansfield 1947).

Assume that the unknown receiver location is $\mathbf{X_r}=[x_r , y_r]^T$, the bearing measurements from N anchor points are expressed as $\beta=[\beta_1, ..., \beta_N]^T$, and the known anchor locations are $\mathbf{x_i}=[x_i , y_i]^T$. The measured bearings do not perfectly reflect the actual bearings $\theta(\mathbf{x})$ $[\theta_1(\mathbf{x}), ..., \theta_N(\mathbf{x})]^T$ due to some noise, that is, the relationship between measured and actual bearings is:

$$\beta = \theta(x_r) + \delta\theta \qquad (2\text{-}5)$$

Where $\delta\theta = [\delta\theta_1, ..., \delta\theta_N]^T$ is the Gaussian noise with zero-mean and $N \times N$ covariance

S = diag $(\sigma^2, ..., \sigma^2)$ (Gavish and Weiss 1992).

In two-dimensional space, the relationship between the bearings of N *anchors* and their locations can be expressed as (Mao *et al.* 2007; Tekdas and Isler 2007):

$$\tan\theta_i(x) = \frac{y_i - y_r}{x_i - x_r} \tag{2-6}$$

Various statistical methods have been applied to estimating a sensor's location. For example, the maximum likelihood (ML) estimator of the receiver location is:

$$x_r = \arg\min \frac{1}{2}(\theta(x_r - \beta)^T S^{-1}[\theta(x_r) - \beta] \tag{2-7}$$

$$= \arg\min \sum_{i=1}^{n}[\frac{1}{2}[\theta_i(x_r) - \beta_i]^2 \times \frac{1}{\sigma_i^2}] \tag{2-8}$$

His nonlinear least squares minimization can be performed using Newton–Gauss iterations:

$$x_{r,i+1} = x_{r,i} + (\partial_x(x_{r,i}))^T S^{-1}\,\partial_x(x_{r,i})^T S^{-1}[\beta - \partial_x(x_{r,i})] \tag{2-9}$$

Where $\partial_x(x_{r,i})$ is the partial derivative of ∂_x with respect to **x** evaluated at $x_{r,i}$. Equation (2-9) requires an initial estimate (e.g., obtained from prior information) that is close enough to the true minimum of the cost function [26].

2.6.2.2 Iterative and Collaborative Multilateration

While the literation technique relies on the presence of at least three anchor nodes to position a fourth unknown node, this technique can be extended to determine locations of nodes without three neighboring anchor nodes. Once a node has identified its position using the beacon messages from the anchor nodes, it becomes an anchor and broadcasts beacon messages containing its estimated position to other nearby nodes. This *iterative multilateration* process (Savvides *et al.* 2001) repeats until all nodes in a network have been localized. Figure (2-5(a)) visualizes this process: in the first iteration, the gray node estimates its location with the help of the three black anchor nodes and in the second iteration, the white nodes estimate their respective locations with the help of two original anchor nodes and the gray node. The drawback of iterative multilateration is that the localization error accumulates with each iteration. In ad hoc deployments of sensor and anchor nodes, it is possible that a node will not have three neighboring anchor nodes, therefore preventing it from determining its own location. In this case, a node can use a process called *collaborative multilateration* to estimate its position using location information obtained over multiple hops. Figure (2-5(b)) shows a simple example with six nodes: four anchor nodes A_i (black) and two nodes with unknown locations S_i (white). The goal of collaborative multilateration is to construct a graph of *participating* nodes, that is, nodes that are anchors or have at least three participating neighbors (e.g., all nodes in Figure (2-5)(b) are participants). A node can then try to estimate its position by solving the corresponding system of over constrained quadratic equations relating the distances among the node and its neighbors.

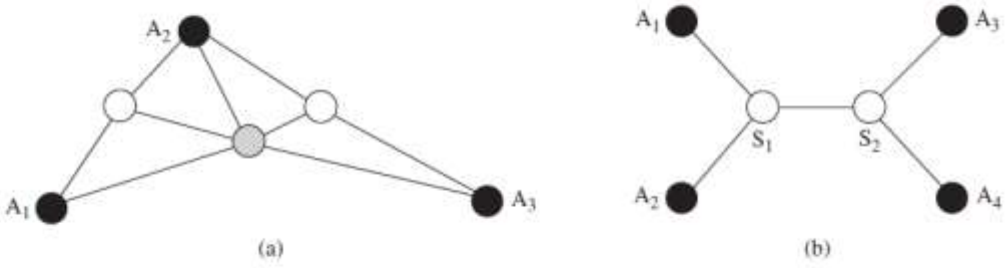

Figure (2-5) (a) Iterative multilateration and (b) collaborative multilateration.

2.7 GPS-Based Localization

The Global Positioning System (GPS) (Hofmann-Wellenhof *et al.* 2008) is the most widely publicized location-sensing system, providing an excellent literation framework for determining geographic positions (Hightower and Borriello 2001). GPS (formally known as NAVSTAR – Navigation Satellite Timing and Ranging) is the only fully operational global navigation satellite system (GNSS) and it consists of at least 24 satellites orbiting the earth at altitudes of approximately 11,000 miles. It began as a test program in 1973 and became fully operational in 1995. In the meantime, GPS has established itself as a widely used aid to civilian navigation, surveying, tracking and surveillance, and scientific applications. GPS provides two levels of service (Dana 1997):

- **The Standard Positioning Service (SPS)** is a positioning service available to all GPS users on a continuous worldwide basis without restrictions or direct charge. High-quality GPS receivers based on SPS are able to attain accuracies of 3 m and better horizontally.
- **The Precise Positioning Service (PPS)** is used by US and Allied military users and is a more robust GPS service that includes encryption and jam resistance. For example, it uses two signals to reduce radio transmission errors, while SPS only uses one signal [27].

GPS satellites are uniformly distributed in a total of six orbits (i.e., there are four satellites per orbit) and they circle the earth twice a day at approximately 7000 miles per hour. The number of satellites and their spatial distribution ensure that at least eight satellites can be seen simultaneously from almost anywhere on the planet. Each satellite constantly broadcasts coded radio waves (known as *pseudorandom code*) that contain information on the identity of the particular satellite, the location of the satellite, the satellite's status (i.e., whether it is working properly), and the date and time a signal has been sent. In addition to the satellites, GPS further relies on infrastructure on the ground to monitor satellite health, signal integrity, and orbital configuration. At least six *monitor stations* located around the world constantly receive the data sent by the satellites and forward the information to a *master control station* (MCS). The MCS (located near Colorado Springs, Colorado) uses the data from the monitor stations to compute corrections to the satellites' orbital and clock information, which are then sent back to the appropriate satellites via *ground antennas*. A GPS receiver (e.g., embedded into a mobile device) receives the information transmitted by the satellites that are currently in view by the receiver. The basic principle of GPS positioning is illustrated in Figure (2-6). Satellites and receivers use very accurate and synchronized clocks so that they generate the same code at exactly the same time. The GPS receiver compares its generated code with the code received from the satellite, thereby deter- mining the actual generation time (e.g., t_0 in Figure (2-6)) of the code at the satellite and the time difference *O* between the code generation time and the current time. Therefore, *O then* expresses the travel time of the code from the satellite to the receiver. Note that the received satellite data is attenuated due to the satellite–earth path even if no obstructions occur. Radio waves travel at the speed of light (about 186 000 miles per second), so if Δ is known, the distance from the satellite to the receiver (distance speed time) can be determined. Once the distance has been determined, the receiver knows that it is located somewhere on a sphere centered on the satellite with a radius equal to the computed distance. Repeating this process with two more satellites, the position of the receiver can be narrowed down to the two points where the three spheres intersect [28]. Typically, one of the two points be eliminated very easily, for example, because it would position the receiver far out in space or the receiver would travel at a virtually impossible velocity. While three satellites appear to be sufficient for localization, a fourth satellite is needed to obtain an accurate position. Positioning via GPS relies on correct timing to make accurate measurements, that is, the clocks of the satellites and the receivers must be synchronized precisely. Satellites are equipped with four atomic clocks (synchronized to each other within a few nanoseconds), providing highly accurate time readings.However, the clocks used for GPS receivers are not nearly as accurate as the atomic clocks onboard the satellites, introducing measurement errors that can have a significant impact on the quality of localization. Because radio waves travel at very high speeds (and therefore require very little time to travel), small errors in the timing can result in large deviations in position measurements. For example, a clock error of 1 MS would result in a position error of about 300 km. Therefore, a fourth measurement is required, where the fourth sphere should ideally

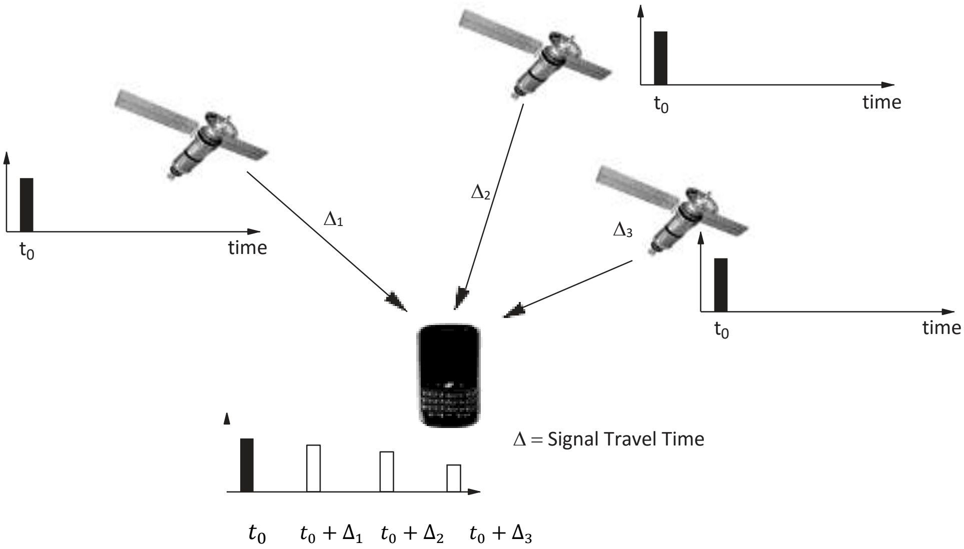

Figure (2-6) GPS positioning principle.

intersect the other three spheres at the exact location of the receiver. Because of timing errors, the fourth sphere may not intersect with all other spheres, even though we know that they are supposed to align. If the spheres are too large, we can reduce their sizes by adjusting the clock (by moving it forward) until the spheres are small enough to intersect in one point. Similarly, if the spheres are too small, we adjust the clock by moving it backwards. That is, because the timing error is the same for all measurements, a receiver can calculate the required clock adjustment to obtain a single intersection point among all four spheres. In addition to providing a means for clock synchronization, a fourth measurement also allows a receiver to obtain a three-dimensional position, that is, latitude, longitude, and elevation. While most GPS receivers available today are able to provide position measurements with accuracies of 10 m or less, advanced techniques to further increase the accuracy are available. For example, Differential GPS (DGPS) (Monteiro *et al.* 2005) relies on land-based receivers with exactly known locations to receive GPS signals, compute correction factors, and broadcast them to GPS receivers that are then able to correct their own GPS measurements. While it is possible to build wireless sensor networks where each sensor has its own GPS receiver, constraints such as high power consumption, cost, and the need for line of sight make a fully GPS-based solution impractical for most sensor networks. However, GPS receivers deployed on a few nodes in a WSN may be sufficient to provide location services based on reference points [29].

2.7.1 Range-Free Localization

The localization approaches discussed in the previous sections are based on distance estimations using ranging techniques (RSS, ToA, TDoA, and AoA) and belong therefore to the class of *range-based* localization algorithms. In contrast, *range-free* techniques estimate node locations based on connectivity information instead of distance or angle measurements. Range-free localization techniques do not require additional hardware and are therefore a cost-effective alternative to range-based techniques. This section describes various different approaches to localization without reliance on ranging techniques.

2.7.2 Ad Hoc Positioning System (APS)

APS (Niculescu and Nath 2001) is an example of a distributed connectivity-based localization algorithm that estimates node locations with the support of at least three anchor nodes, where localization errors can be reduced by increasing the number of anchors. Each anchor node propagates its location to all other nodes in the network using the concept of distance vector (DV) exchange (Lu *et al*. 2003), where nodes in a network periodically exchange their routing tables with their one-hop neighbors. In the most basic scheme of APS, called DV-hop, each node maintains a table X_i , Y_i , h_i , where X_i , Y_i is the location of node i and h_i is the distance in hops between this node and node i. When an anchor obtains distances to other anchors, it then determines an average size for one hop (called the correction factor), which too is then propagated throughout the network. The correction factor c_i of anchor i is determined as: $c_i = \frac{\sum \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}}{\sum h_i}$

for all landmarks j ($i \neq j$). Given the locations of the anchors and the correction factor, a node is then able to perform trilateration to estimate its own location. Figure (2-7) presents an example with three anchor nodes A_1, A_2, and A_3. Anchor A_1, knowing its Euclidean distances (50 m and 110 m) and hop distances (two hops and six hops) to the other two anchor nodes, computes a correction of (50 +110)/(2+6)=20, which represents the estimated distance of a hop in meters. In a similar fashion, A_2 computes a correction factor of 18.6 and A_3 computes a correction factor of 17.3. Corrections are propagated via controlled flooding

(i.e., once a node receives a correction, it ignores subsequent ones) to ensure that each node will only use one correction factor, typically from the closest anchor. For example, sensor node S in Figure 5 uses the correction factor obtained from A_2, that is, 18.6, to estimate its distances to the three anchors by multiplying the correction factor with the hop counts (leading to distances 3×18.6 to A_1, 2×18.6 to A_2, and 3×18.6 to A_3). Given these distances, triangulation (as described in 3.1) can be used to determine the position of S. In a variation of this approach, called the DV-distance method, distances between neighboring nodes are determined using radio signal strength measurements and distributed to other nodes in meters instead of hops. While this approach provides finer granularity (not all hops are estimated to be the same size), it is also more sensitive to measurement errors. Finally, in the Euclidean method, true Euclidean distances to anchors are used. A node must have at least two neighbors that have distance measurements to an anchor, where the distance between the two neighbors is known. Based on this information, simple trigonometric relationships can be used to determine the distance of a node to an anchor [30].

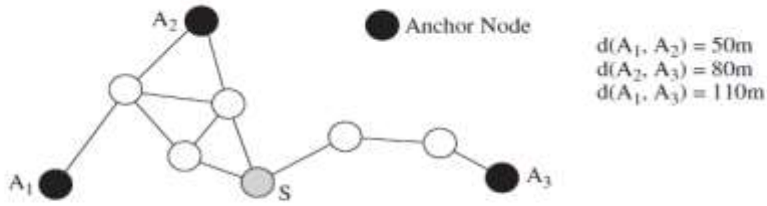

Figure (2-7) Example of DV-hop localization

2.8 Previous work:

In the past three decades, there has been great interest and progress in the field of intelligent vehicles for both researchers and industries. Various intelligent vehicle systems have been proposed to reduce road accidents and traffic congestions. The continuous development of sensing and computation technologies has led to the development of various driver assistant systems. This research builds upon several previous projects and researches and the descriptions below give a concise brief for some of those projects.

- Dhanasingaraja R, Kalaimagal S, Muralidharan G (2014) of Anna University, Chennai, India " System It Navigates The Vehicle Autonomously To Its Destination". In their paper, they described a system it navigates the vehicle autonomously to its destination. This system provides a communication between vehicle and internet using GPRS modem. This system interfaced with OSRM open source map through internet. Therefore, the robot decides the path from internet. In non-urban Domains such as deserts the problem of successful GPS-based navigation appears to be almost solved, navigation in urban domains particularly in the close vicinity of buildings is still a challenging problem. In such situations, GPS accuracy significantly drops down due to unavailability of GPS signal. This project also improves the efficiency in navigation. This system not only relay on GPS. To improve the efficiency it uses location information from inertial sensors also. This system uses rotatable laser range finder for obstacle sensing. This is also designed in such a way that it can be monitored from anywhere through internet [31].

- Tanveer Hossen Sakkhor, Samin Saksiat Zaman, et all (2014) of BRAC University developed a system on their final year project "Autonomous Car Using Full Mapping GPS System" using full mapping GPS. Which is based on a Laptop computer to generate the path coordinates and an Android phone to obtain the GPS data and used the mobile camera as the obstacle detection image-processing unit. An Arduino is used as the brain of the system. The car that they made has electric motor for driving of each of the two front wheels via independent controllers and has full drive-by-wire control of the throttle, steering and braking system [32].

- Emilio Hernández Chiva (2016) of Barcelona Tec university developed a system in his project " Design and construction of an electric autonomous driving vehicle" using a light sensor (LDR) with camera and other sensor to detect the path and compare the information given by the sensor with the orders followed by the CPU as well as a light sensor (LDR) to help deciding if the camera and the computer vision algorithm should be used. Since the GPS can have an error as big as some meters, the information of the position must be refined by fusing it with the information of the inertial sensor and the camera. This way, the knowledge of the positioning will be much more exact. To remove uncertainty from the GPS signal, a map is constantly updated internally to have already a previous map from the same location, with the current and predicted location of the obstacles near it. This map is updated with the vehicle's movement [33].

- Pierre-Jean Rigole (2014) Master of Science Thesis,Stockeolm in his research " Study of a Shared Autonomous Vehicles Based Mobility Solution in Stockholm" provides an analysis of potential benefits of a fleet of Shared Autonomous Vehicles (SAV) providing a taxi service to replace private car commuter trips in a metropolitan area. he developed a framework for dynamic allocation of SAVs to passenger trips, empty-vehicle routing and multi-criteria evaluation with regard to passenger waiting time, trip time and fleet size. Using a representation of current private trip and a detailed road network representation, different scenarios are compared with respect to passenger travel time, number of vehicles needed and vehicle mileage. The results show that an SAV-based personal transport system has the potential to provide an on-demand door-to-door transport with a high level of service, using less than 10 % of today's private cars and parking places.

Chapter Three

SYSTEM DESIGN

Chapter Three

SYSTEM DESIGN

3.1 Overview

This chapter focus on the project design, implementation method and hardware components will be introduced as well .This control system consist of two main parts hardware part and software part. The hardware part consist of a GPS chip for localization , two dc motors one for movement and another for direction, Bluetooth module for connecting with phone, SD card module for store the points of way-point and ultrasonic for obstacle detection. The software part used in this work is ARDUINO IDE ,Google Earth program to detect the path and Bluetooth Terminal for displaying the information.

3.2 System block diagram

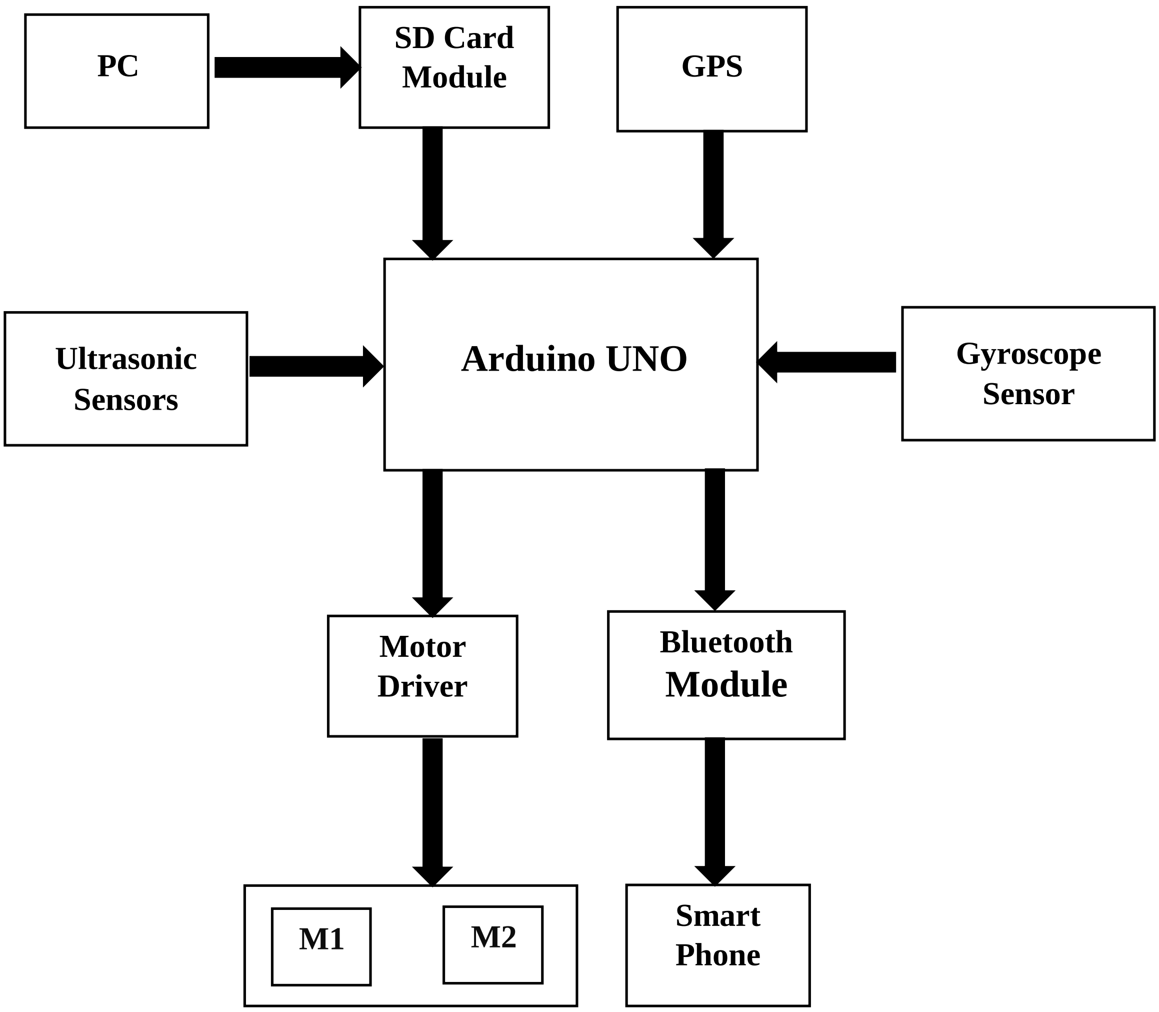

Figure (3-1) self-driving system block diagram

This proposed system is semi-autonomous-vehicle that uses to move a car on wanted path and avoid collision obstacles without human help. The block diagram for this system consists of eight blocks Arduino UNO, GPS chip, ultrasonic sensor, mini SD card module, Bluetooth module, motor system, gyroscope sensor and smart phone as shown in figure (3-1).

3.3 System hardware components

3.3.1 Arduino

Arduino is an open –source platform used for building electoincs projects Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and software ArduinoIDE (integrated development environment) used the write and upload computer code to microcontroller. Arduino offers some advantages for teachers, students, and interested amateurs over other systems such as:

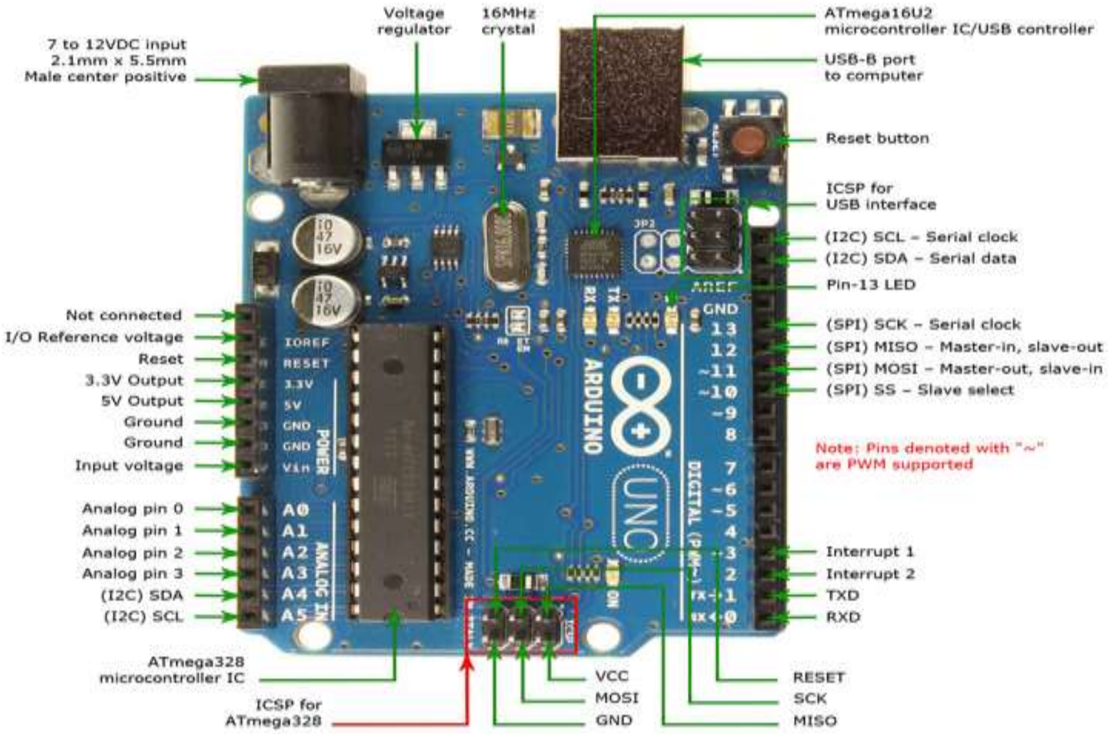

Figure (3-2) Arduino Uno

- Cross-platform- The Arduino software runs on Windows, MacintoshOSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
- Simple, clear programming environment. The Arduino programming environment is easy to use for beginners, yet flexible enough for advanced users. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with the look and feel of Arduino.
- Open source and extensible software. The Arduino software is published as open source tools, available for extension by experienced programmers.

- Open source and extensible hardware - The Arduino shown in figure(3-2) is based on ATmega328 microcontrollers .

Table (3-1) Arduino UNO technical characteristics

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	20V
Digital I/O Pins	14 (of which 14 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

3.3.2 Sensors

Sensor is a device which converts one form of energy into another form such as a microphone convert sound into an electrical form or a light sensor give output according to the intensity of light. Sensors are the key components for perceiving the environment. A sensor is a device that detects and responds to some type of input from the physical environment. The specific input could be light, heat, motion, moisture, pressure, or any one of a great number of other environmental phenomena. The output is generally a signal that is converted to human readable display at the sensor location or transmitted electronically over a network for reading or further processing. In short, a sensor is a device that detects and responds to some type of input from the physical environment such as light, heat, motion, moisture, pressure and used to switch voltages or currents we used various sensors to overcome the problem of obstacle detection, direction and localization. These sensors enabled us to roam our vehicle safely and more accurately. The sensors we used are:

- GPS chip
- Ultrasonic sensor
- Bluetooth module

- SD card module
- Gyroscope sensor

3.3.2.1 Ultrasonic sensor

Figure (3-3) ultrasonic sensor

An Ultrasonic sensor is a device that can measure the distance to an object by using ultrasound waves. It measures distance by sending out an ultrasound wave at a specific frequency and listening for that sound wave to bounce back. By recording the elapsed time between the sound wave being generated and the sound wave bouncing back, it is possible to calculate the distance between the sonar sensor and the object. Since it is known that sound travels through air at about 344 m/s (1129 ft/s), you can take the time for the sound wave to return and multiply it by 344 meters (or 1129 feet) to find the total round-trip distance of the sound wave. Round-trip means that the sound wave traveled 2 times the distance to the object before it was detected by the sensor; it includes the 'trip' from the sonar sensor to the object AND the 'trip' from the object to the Ultrasonic sensor (after the sound wave bounced off the object). To find the distance to the object, simply divide the round-trip distance in half.

$$d = \frac{v_{sound} \times t}{2}$$

Where v is a velocity of sound (344 m/s), t is time of travel sound wave (seconds)

An ultrasonic sensor that uses electrical and mechanical energy transformation to measure the distance from the sensor to the target object. It consists of a transmitter and receiver that are available as separate units or embedded together as a single unit.

This sensor consists of four pins as show in figure (3-3):

(1) VCC-connect to the 5v DC.

(2)Trigger- Pulse input that triggers the sensor.

(3)Echo-Indicates the reception from the target.

(4)Gnd- Ground.

Working principle :

A sonar sensor work through six steps:

Step 1: Make a "Trig" pin of the sensor high for "10 micro second". This indicates a sensor cycle.

Step 2: 8*40KHz pulse will be sent from the transduces of the sensor, after which time the “Echo” pin of the sensor will go from low to high.

Step 3: The 40 KHz sound will bounce off the nearest object and return to the sensor.

Step 4: When the sensor detect the reflected sound wave, the echo pin will go low.

Step 5: The distance between the sensor and the detected object can be calculated based on. the length Of the Echo pin is high.

Step 6: If no object is detected, the Echo pin will stay high for 38ms and then go low.

The figure (3-4) shows the working principle of an ultrasonic.

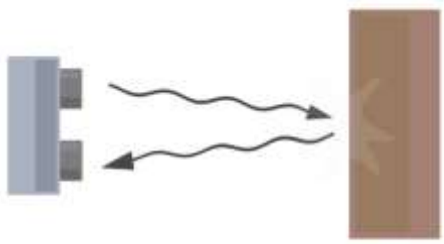

Figure (3-4) working principle of an ultrasonic

3.3.2.2 GPS

The Global Positioning System (GPS) is a network of about 30 satellites orbiting the earth at an altitude of 20,000 km. Any device as a SatNav, mobile phone or handheld GPS unit, can receive the radio signals that the satellites broadcast. Wherever we are on the planet, at least four GPS satellites are ‘visible’ at any time. Each one transmits information about its position and the current time at regular intervals. These signals, travelling at the speed of light, are intercepted by your GPS receiver, which calculates how far away each satellite is based on how long it took for the messages to arrive as show in figure (3-5). Once it has information on how far away at least three satellites are, your GPS receiver can pinpoint your location using a process called trilateration. GPS satellites circle the earth twice a day in a very precise orbit and transmit signal information to earth. GPS receivers take this information and use trilateration to calculate the user's exact location. Essentially, the GPS receiver compares the time a signal was transmitted by a satellite with the time it was received. The time difference tells the GPS receiver how far away the satellite is. Now, with distance measurements from a few more satellites, the receiver can determine the user's position and display it on the unit's electronic map. A GPS receiver must be locked on to the signal of at least 3 satellites to calculate a 2-D position (latitude and longitude) and track movement. With four or more satellites in view, the receiver can determine the user's 3-D position (latitude, longitude and altitude).Once the user's position has been determined, the GPS unit can calculate other information, such as speed, bearing, track, trip distance, distance to destination, sunrise and sunset time and more.

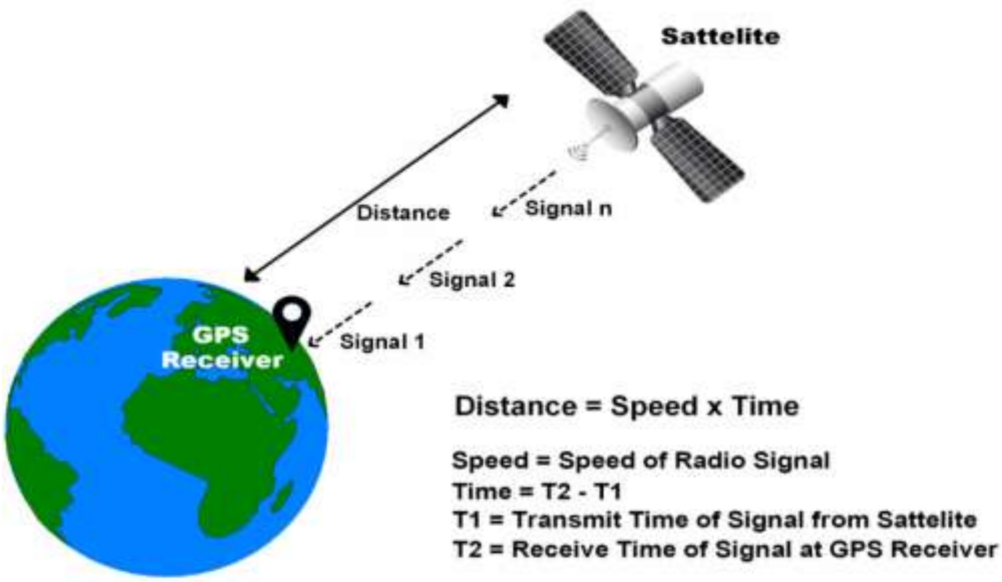

Figure (3-5) working principle of GPS

In this work, RoyalTek REB-3571LP module was used (figure (3-6)) to determine the coordinates. RoyalTek REB-3571LP low power and small form factor board is the newest generation of RoyalTek GPS module.

Figure(3-5) REB-3571LP Interface Board Front View

Table (3-2) RoyalTek REB-3571LP module pins

Pin	Function
GND	Connected to ground
RX	The RXA pin is the serial input data (Default NMEA) .
TX	The TXA pin is the serial output data (Default NMEA)
232RX	The RXB pin is the serial input data (Default Null).
232TX	The TXB pin is the serial output data (Default Null)
PPS	The maximum power consumption of active antenna is about 85mW. The input gain ranges are 19~ 22dB.
5V	Connected to 5 Volt.

3.3.2.3 Micro SD Card Module for Arduino:

The module (Micro-SD Card Adapter) is a Micro SD card reader module, and the SPI interface via the file system driver, microcontroller system to complete the Micro-SD card read and write files. Arduino users can directly use the Arduino IDE comes with an SD card to complete the library card initialization and read-write.

Micro-SD Card was used to store the coordinates of desired path .We get those coordinates form the google earth application .it detects the shortest path and divide the path to points . We store the coordinates of point to SD card to compare them to the car position by Arduino to control the movement.

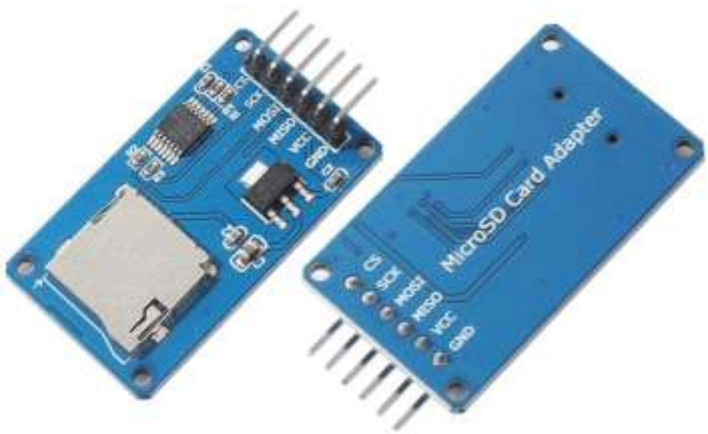

Figure (3-6) MicroSD card module

Control Interface: A total of six pins (GND, VCC, MISO, MOSI, SCK, CS), GND to ground, VCC is the power supply, MISO, MOSI, SCK is the SPI bus, CS is the chip select signal pin.

3.3.2.4 Bluetooth module (HC-05):

HC-05 Bluetooth Module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. Its communication is via serial communication which makes an easy way to interface with controller, PC or phone. HC-05 Bluetooth module provides switching mode between master and slave mode which means it able to use neither receiving nor transmitting data. It was used to display some information to Android phone via Serial Bluetooth Terminal application.

Figure (3-7) HC-05 Bluetooth Module

Pin	Description	Function
VCC	+5V	Connect to +5V
GND	Ground	Connect to Ground
TXD	UART_TXD, Bluetooth serial signal sending PIN	Connect with the MCU’s (Microcontroller and etc) RXD PIN.
RXD	UART_RXD, Bluetooth serial signal receiving PIN	Connect with the MCU’s (Microcontroller and etc) TXD PIN.
KEY	Mode switch input	If it is input low level or connect to the air, the module is at paired or communication mode. If it’s input high level, the module will enter to AT mode.

Table (3-3) HC-05 Bluetooth module pins

3.3.2.5 L293D Motor Driver Module:

L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motors with a single L293D IC. Dual H-bridge Motor Driver integrated circuit (IC). It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. The figure (3-8) show L293D motor driver module:

.

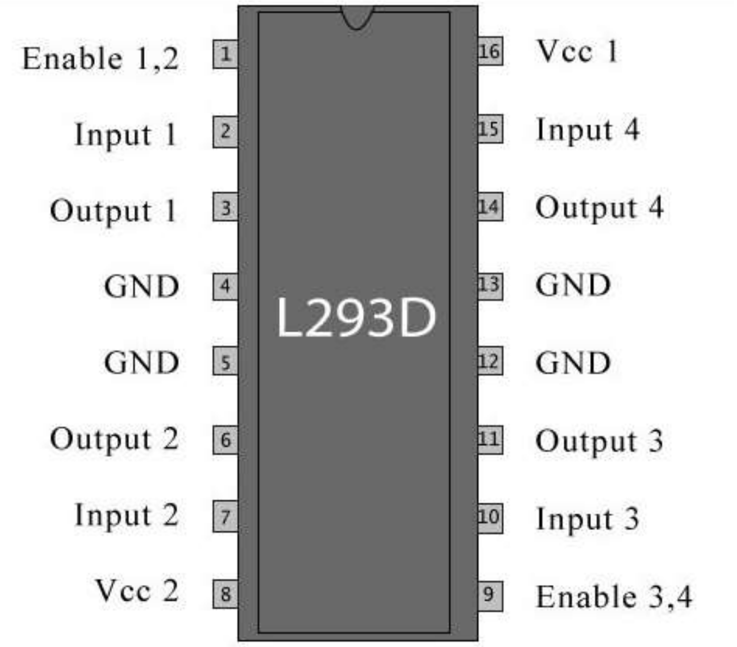

Figure (3-8) L293D motor driver module

3.3.2.6 Gyroscope Sensor (MCU6050)

Microelectromechanical (MEMS) systems are microsystems that have a very small size. It is an advanced form of technology that has given rise to different electronic components like accelerometer and gyroscope sensor. An accelerometer is a device that can be used to detect non-gravitational motion. In other hand, a gyroscope sensor can be used to detect gravitational motion. Thus, a gyroscope sensor and an accelerometer complement each other. A gyroscope sensor is a type of sensor which be found inside IMU (Inertial Measurement Unit). It can be used to measure the rotation on a particular axis. It consists of a rotor which is nothing but a freely rotating disc. The rotor is mounted on a spinning axis, which is present in the center of another larger wheel. The module MPU6050 IMU (shown in figure (3-9)) has both 3-Axis accelerometer and 3-Axis gyroscope integrated on a single chip. The outputs of the gyroscope are in degrees per second.

Gyroscope

The MPU-6050 consist of a 3 axis gyroscope which can detect rotational velocity along the x,y,z axis with micro electro mechanical system technology (MEMS). When the sensor is rotated along any axis a vibration is produced due to Coriolis effect which is detected by the MEMS.16-bit ADC is used to digitize voltage to sample each axis. The full-scale range of output is +/- 250, +/- 500, +/- 1000, +/- 2000. Angular velocity is measured along each axis in degree per second unit.

Figure (3-9) MPU6050 sensor

Accelerometer

The earth's gravity is a constant acceleration where the force is always pointing down to the center of the Earth. When the accelerometer is parallel with the gravity, the measured acceleration will be 1G, when the accelerometer is perpendicular with the gravity, it will measure 0G.Gyro sensor detects angular velocity created by the sensor's own movement. Angles are detected via integration operations by a CPU.

Gyroscope Sensor Working Principle

Depending on the direction, there are three types of angular rate measurements (shown in figure (3-10)): Yaw the horizontal rotation on a flat surface when seen the object from above, Pitch-Vertical rotation as seen the object from front and Roll the horizontal rotation when seen the object from front. The concept of Coriolis force is used in Gyroscope sensors. In this sensor to measure the angular rate, the rotation rate of the sensor is converted into an electrical signal. Working principle of Gyroscope sensor can be understood by observing the working of Vibration Gyroscope sensor. This sensor consists of an internal vibrating element made up of crystal material in the shape of a double – T- structure. This structure comprises a stationary part in the center with 'Sensing Arm' attached to it and 'Drive Arm' on both sides. This double-T-structure is symmetrical. When an alternating vibration electrical field is applied to the drive arms, continuous lateral vibrations are produced. As drive arms are symmetrical, when one arm moves to left the other moves to the right, thus canceling out the leaking vibrations. This keeps the stationary part at the center and sensing arm remains static. When the external rotational force is applied to the sensor vertical vibrations are caused on Drive arms. This leads to the vibration of the Drive arms in the upward and downward directions due to which a rotational force acts on the stationary part in the center. Rotation of the stationary part leads to the vertical vibrations in sensing arms. These vibrations caused in the sensing arm are measured as a change in electrical charge. This change is used to measure the external rotational force applied to the sensor as Angular rotation.

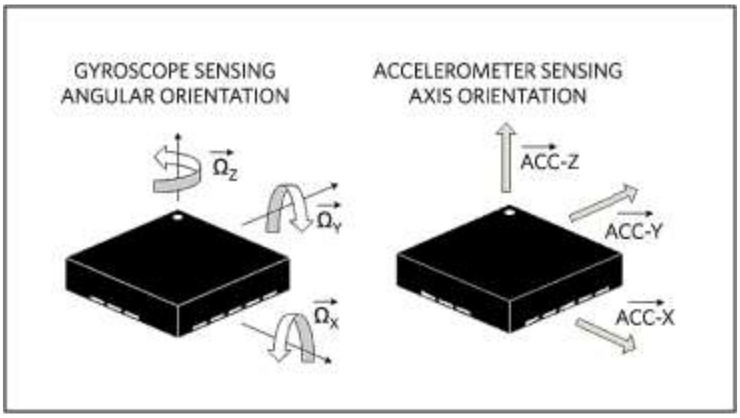

Figure (3-10) Angular versus linear motion

CHAPTER FOUR

RESULTS AND DISCUSSION

Chapter Four

RESULTS AND DISCUSSION

This chapter provides the stages and steps by which the proposed system can be implemented and turned into being a practical model.

4.1. System Flow Chart

The system operation stages can be explained in five processes:

4.1.1. Detection position coordinates

GPS chip was connected as a serial device to the pins 3, 4 of Arduino .GPS chip gets the information by connecting to the satellites. At least four GPS satellites are 'visible' at any time. Each one transmits information about its position and the current time at regular intervals. These signals, travelling at the speed of light, are intercepted by GPS receiver, which calculates how far away each satellite is based on how long it took the messages to arrive. Once it has information on how far away, at least three satellites are. The GPS receiver compares the time a signal was transmitted by a satellite with the time it was received. The time difference tells the GPS receiver how far away the satellite is. Now, with distance measurements from a few more satellites, the receiver can determine the position and display it on the Arduino. GPS receiver must be locked on to the signal of at least three satellites to calculate a 2-D position (latitude and longitude). With four or more satellites in view, the receiver can determine the user's 3-D position (latitude, longitude and altitude).After that, latitude and longitude were stored to EEPROM of Arduino as a global variable. Finally, to display the coordinates, they were sent to android phone by Bluetooth module (HC-05) as shown in figure (4-4) .Serial Bluetooth Terminal Application was used to connect between Bluetooth module and android phone .figure (4.1) show the reading GPS and displaying process flow chart.

4.1.2. Avoiding obstacles

This algorithm is about an obstacle avoidance. As illustrated in the previous section, Ultrasonic sensor module (HC-SR04) is used and it provides ranging of 2-400 cm non-contact measurement with accuracy 3mm. This module includes an ultrasonic transmitters, receiver, and control circuit. The basic principle of work as pointed figure (4-2):

1. Using IO trigger for at least 10us high level signal
2. If the sensor automatically sends eight 40 kHz and detect whether there is a pulse signal back.

3. If the signal back, through high level, time of high output IO duration is the time from sending.

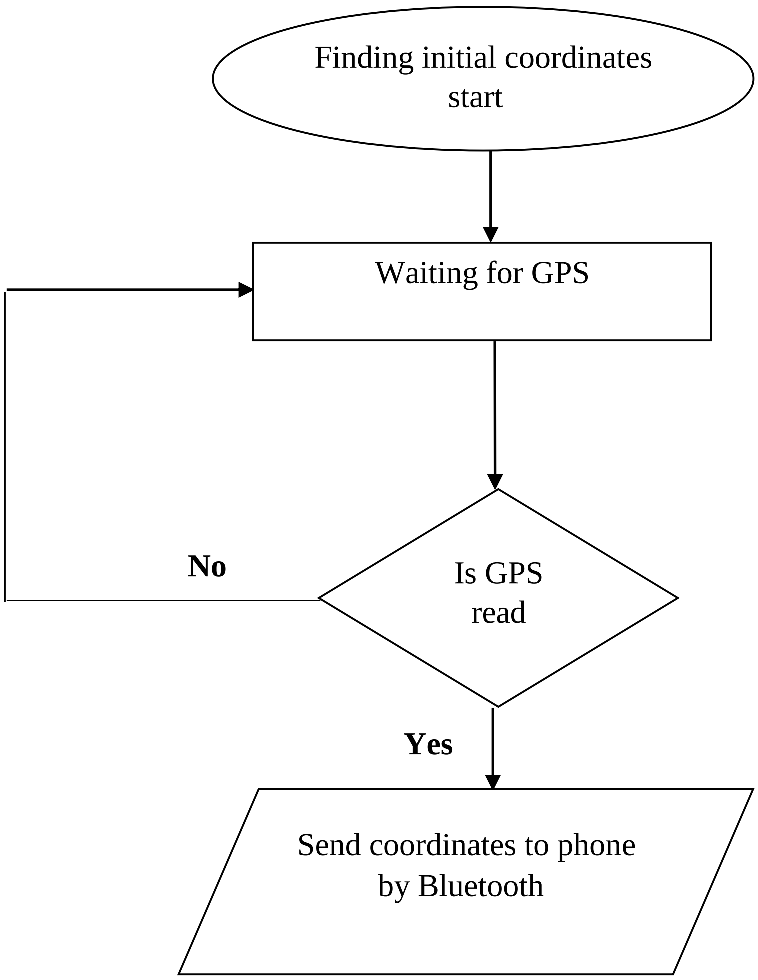

Figure (4.1) finding initial coordinates process

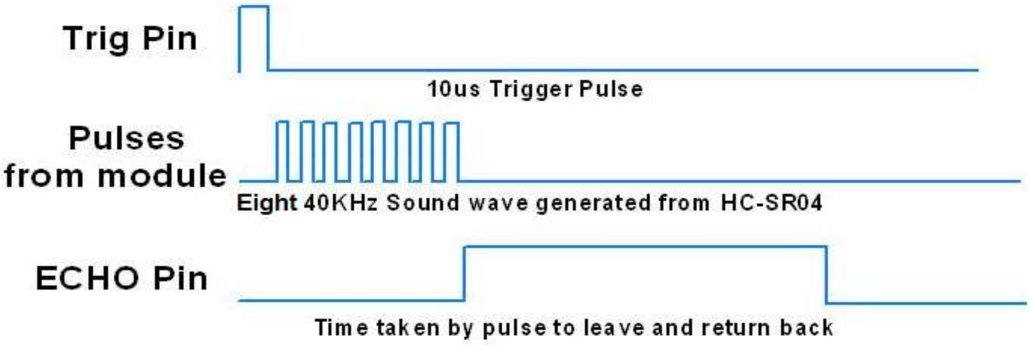

Figure (4-2) Time diagram for the ultrasonic sensor

$$\text{distance} = \frac{\text{high level } \times v_{\text{sound}}}{2}$$

Figure (4.3) shows the reading ultrasonic sensors, avoiding an obstacle and displaying process flow chart. Tow ultrasonic sensors were used, one in front the car and the other behind.

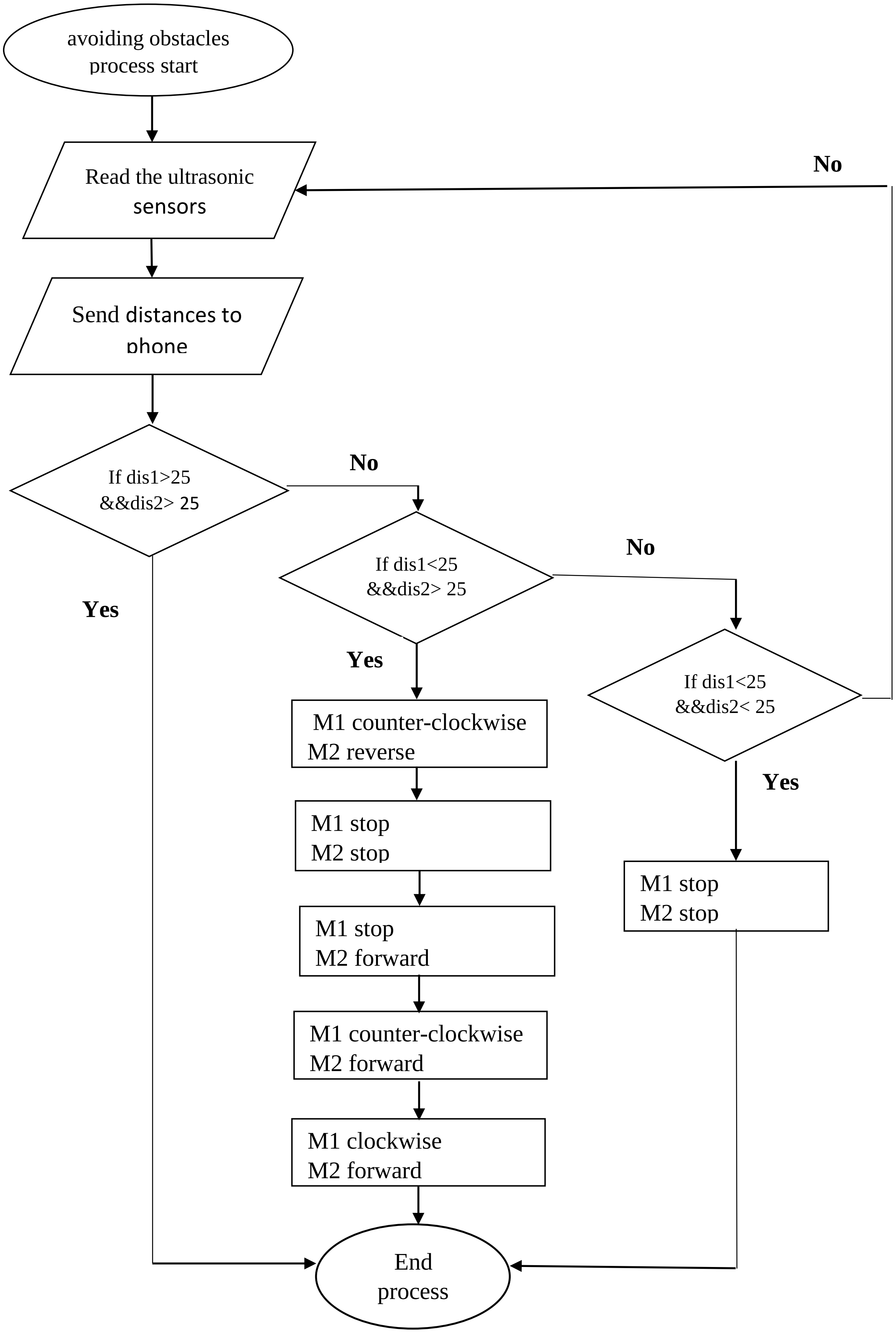

Figure (4-3) avoiding obstacles process

Firstly, ultrasonic sensors detect the distances between the car and obstacles in front and behind. Then. They send the distances to Arduino. Here we have three cases:

Case one: if the distances are less than the safe distance for the tow sensors the car will be stopped.

Case two: if the distance that in front is less than the safe distance and the distance that behind is bigger than the safe distance; Arduino will send signals to control the motors as shown in figure (4-3).

Case three: if the distances are bigger than the safe distance the program will go on.

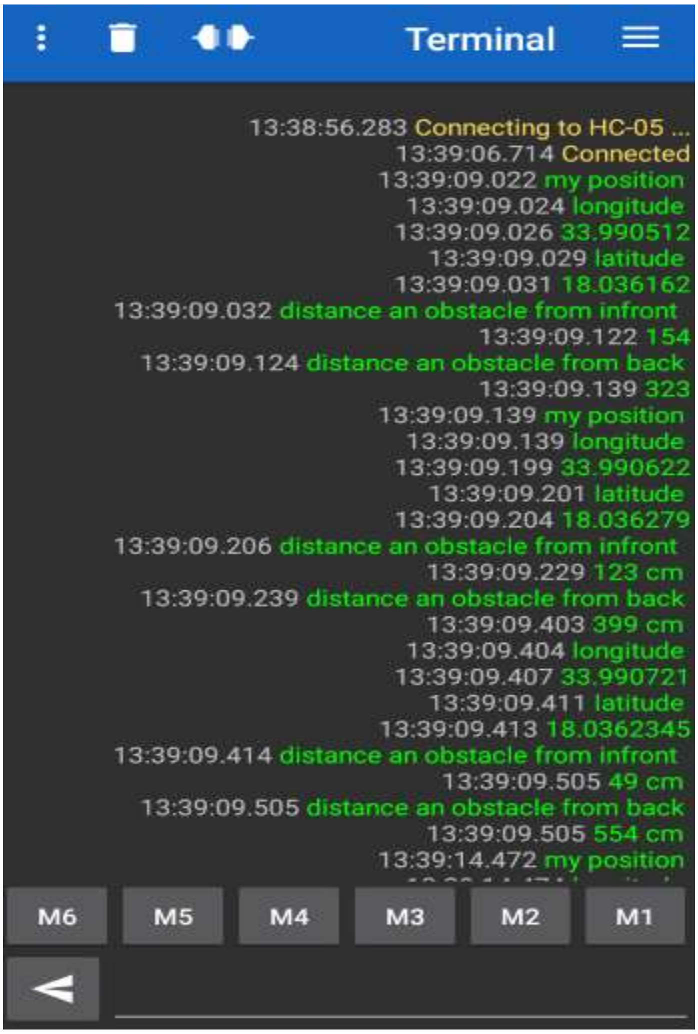

Figure (4-4) output of sending information to phone by Bluetooth

4.1.3. Getting the coordinates of path :

This process divide to four stages:

Stage one: Google Earth Pro program was used to detect three information of desired path: Detect the paths that are possible, detect the shortest path, divide the desire path to small points and detect their coordinates. After the path was detected, it will be saved as KML file. Figure (4.4) shows this stage.

Stage two: GPS Utility application was used to output TXT file from KML file and organize it. Figure (4.5) shows this output.

Stage three: storing the coordinates of desire path to MICRO SD CARD. information that be gotten form google earth application stored into SD card by PC as a text file.

Stage four: connect SD card to micro SD card module to write the information to Arduino. SD card module uses SPI protocol that enable to send or receive serial information as a slave or master device. Arduino reads the coordinates from sd card and store them as a global variables one by one. Figure (4.7) shows how to get coordinates of path processes.

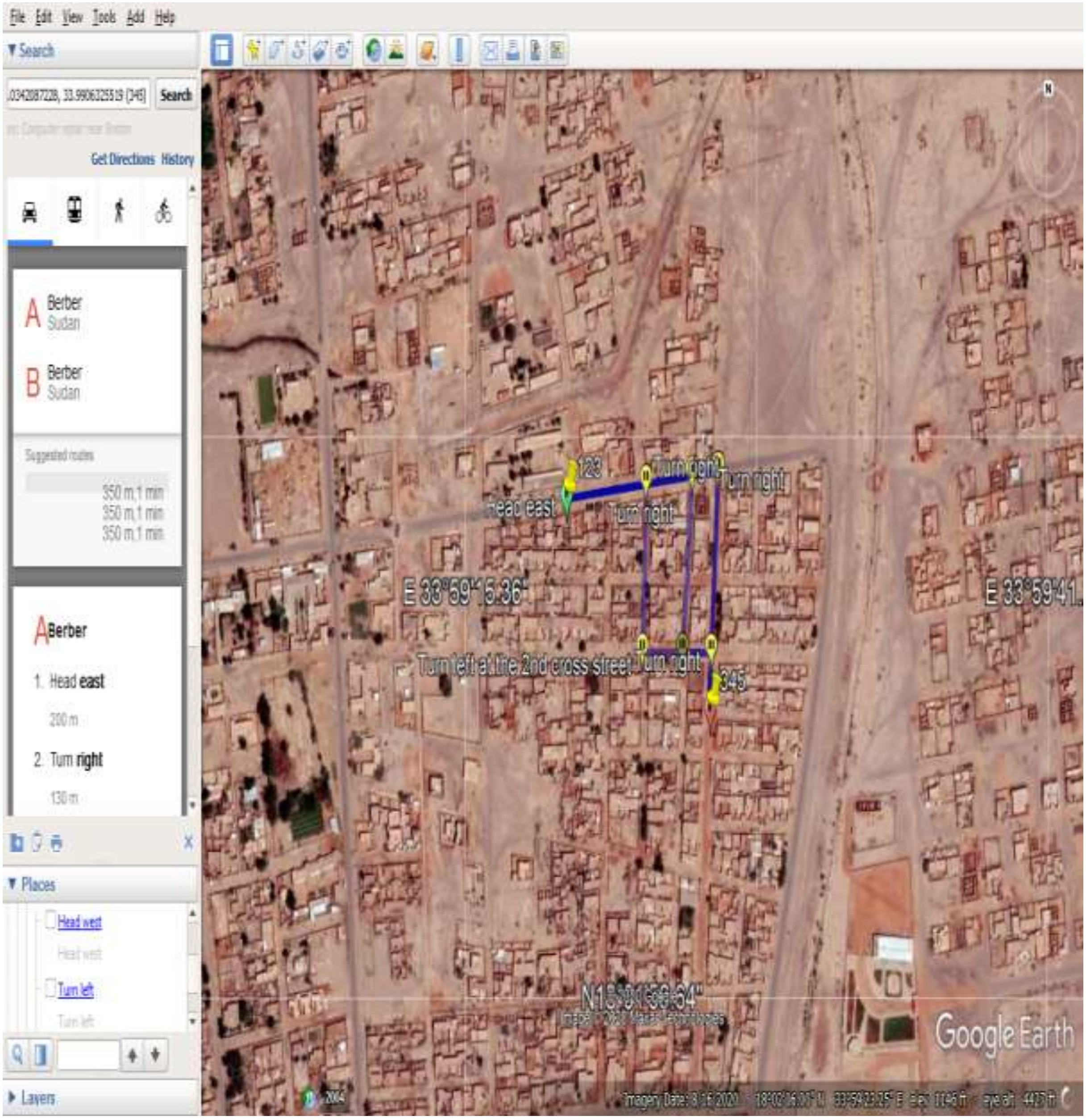

Figure (4.5) detecting path by google earth

```
[{{FileHeader}}]
Desc = Data exported from GPS Utility

[W001++00000000-0000-0000-00DE-29A74D93E540++]
Type = Mark
CreateTime = 2020-12-21 12:13:55Z
LatLon = 18.035602 N 033.988853 E
ExistsOutsideCollection = TRUE
[W002++00000001-0000-0000-00DE-29A74D93E540++]
Type = Mark
CreateTime = 2020-12-21 12:13:55Z
LatLon = 18.035626 N 033.989097 E
ExistsOutsideCollection = TRUE
[W003++00000002-0000-0000-00DE-29A74D93E540++]
Type = Mark
CreateTime = 2020-12-21 12:13:55Z
LatLon = 18.035652 N 033.989372 E
ExistsOutsideCollection = TRUE
[W004++00000003-0000-0000-00DE-29A74D93E540++]
Type = Mark
CreateTime = 2020-12-21 12:13:55Z
LatLon = 18.035695 N 033.989838 E
```

Figure (4.6) converting desire path to text file by GPS utility

4.1.3. Calculate bearing and distance

- **Calculate distance :**'haversine' formula to calculate the great-circle distance between two points that is, the shortest distance over the earth's surface – giving a distance between the points 'as-the-crow-flies' (ignoring any hills they fly over)

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2)$$

$$c = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{(1-a)})$$

$$d = R \cdot c$$

Where φ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km).

note that angles need to be in radians to pass to trig functions. the simple spherical law of cosines formula($\cos c = \cos a \cos b + \sin a \sin b \cos C$) gives well conditioned results down to distances as small as a few meters on the earth's surface.

- **Calculate bearing** : The following formula is for the initial bearing (sometimes referred to as forward azimuth) which if followed in a straight line along a great-circle arc will take you from the start point to the end point:

$$\theta = \operatorname{atan2}(\sin\Delta\lambda \cdot \cos\varphi_2 , \cos\varphi_1 \cdot \sin\varphi_2 - \sin\varphi_1 \cdot \cos\varphi_2 \cdot \cos\Delta\lambda)$$

Where φ_1,λ_1 is the start point, φ_2,λ_2 the end point ($\Delta\lambda$ is the difference in longitude)

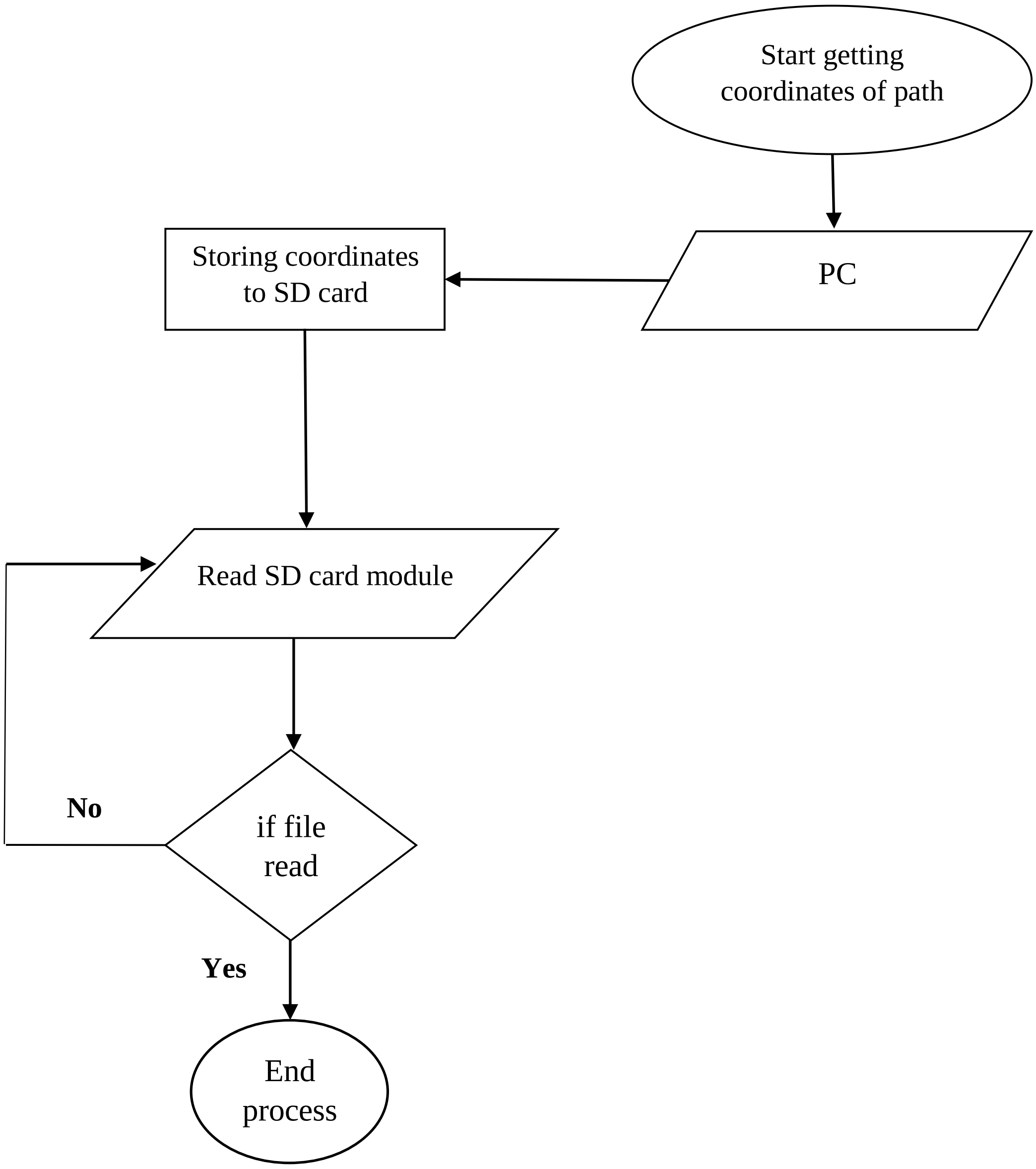

Figure (4.7) getting the coordinates of path process

4.1.4. Detecting the heading

Gyroscope sensor is a device return an output proportional from the rotational velocity. Gyro sensor can sense rotational motion and change in orientation .Gyro sensor was connected to the car to detect the initial heading. There is two cases:

Case one: when the bearing is a negative number, the car will turn left until the gyro sensor equal the bearing.

Case tow: when the bearing is a positive number, the car will turn right until the gyro sensor equal the bearing.

Finally, when the distance between the positions of car to the waypoint is bigger than error distance (nearly five meters) the program go on until the distance becomes zero. Figure (4-8) shows these processes.

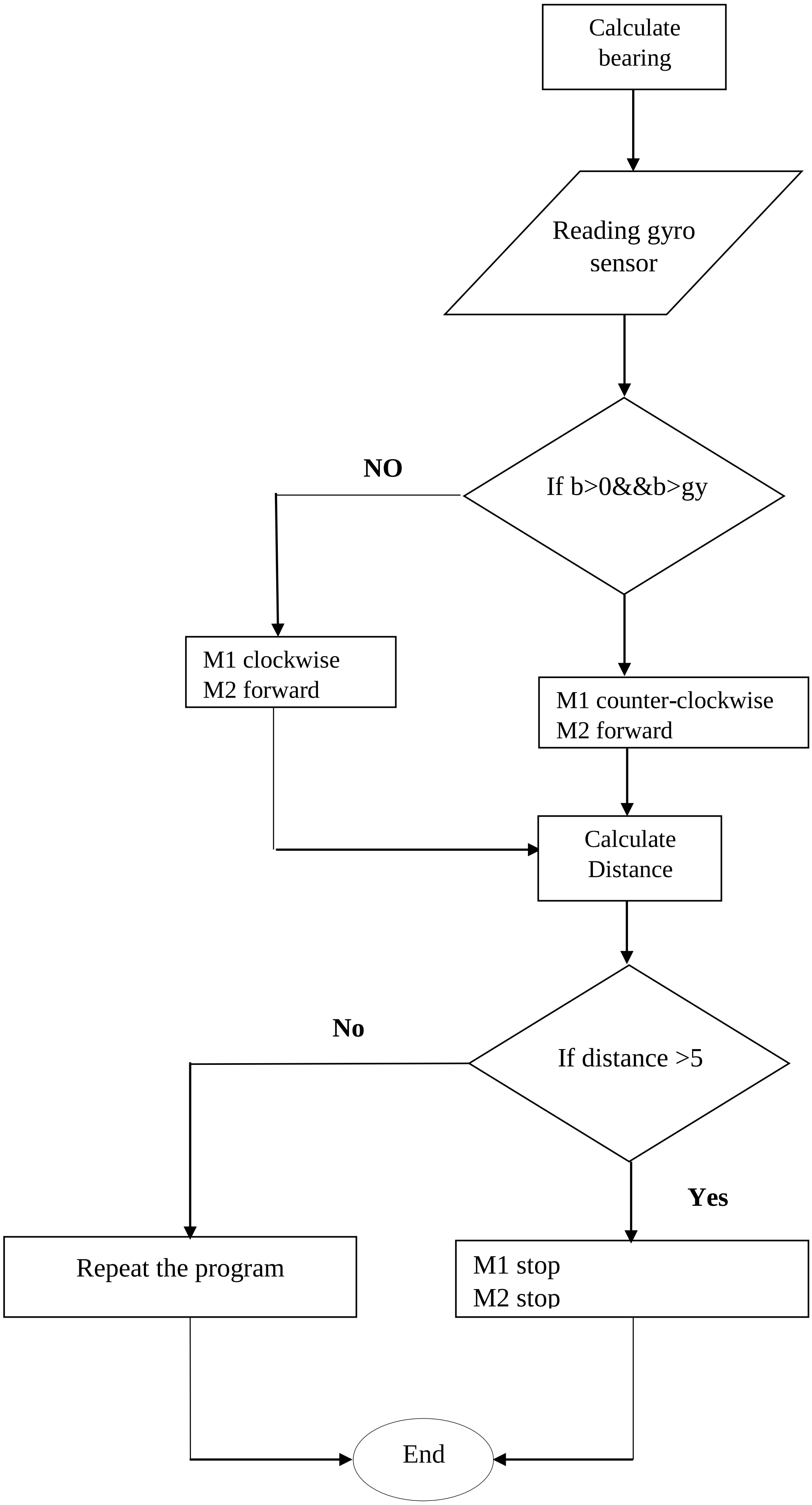

Figure (4-8) detecting heading process and moving to waypoint

4.2. System design :

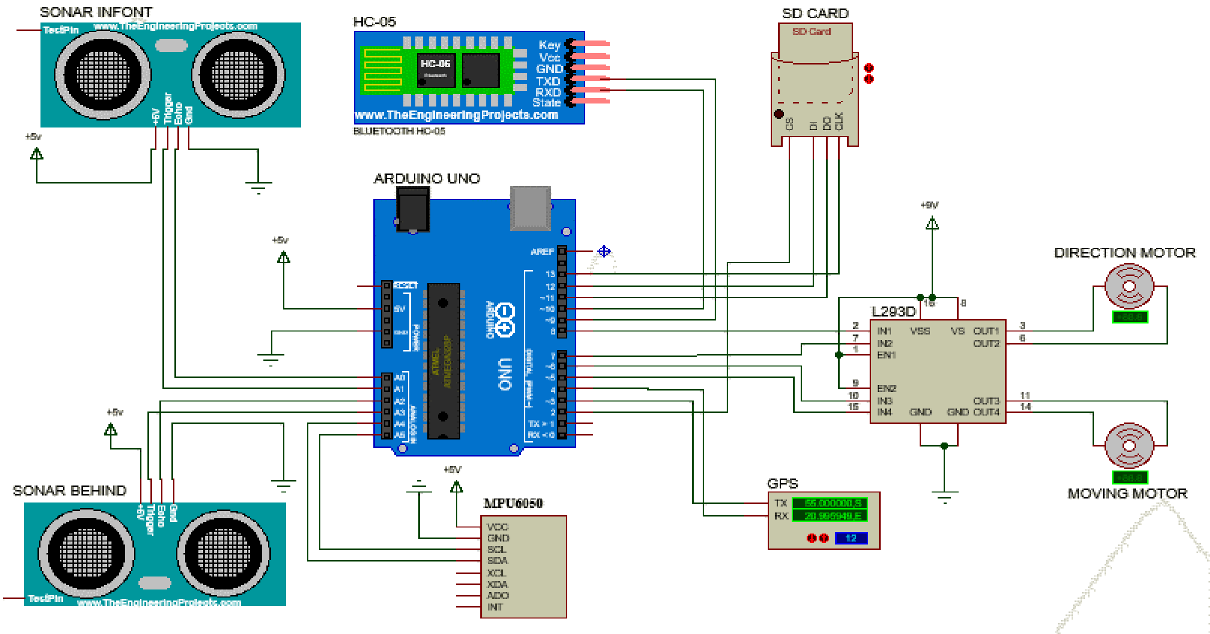

Figure (4-9) connection of practical circuit

As shown in figure (4-9) GPS chip was connected in 3,4 pins of Arduino Uno , motor driver L293D was connected to 5,6,7,8 pins ,Bluetooth module was connected to 9,10 pins ,SD card was connected to I2C pins (11,12,13),2 as CS port ,gyroscope sensor was connected to SPI pins (A4,A5) and finally ultrasonic sensors was connected to A0,A1,A2,A3 pins as a digital ports. Figure (4-10) shows the prototype that was used.

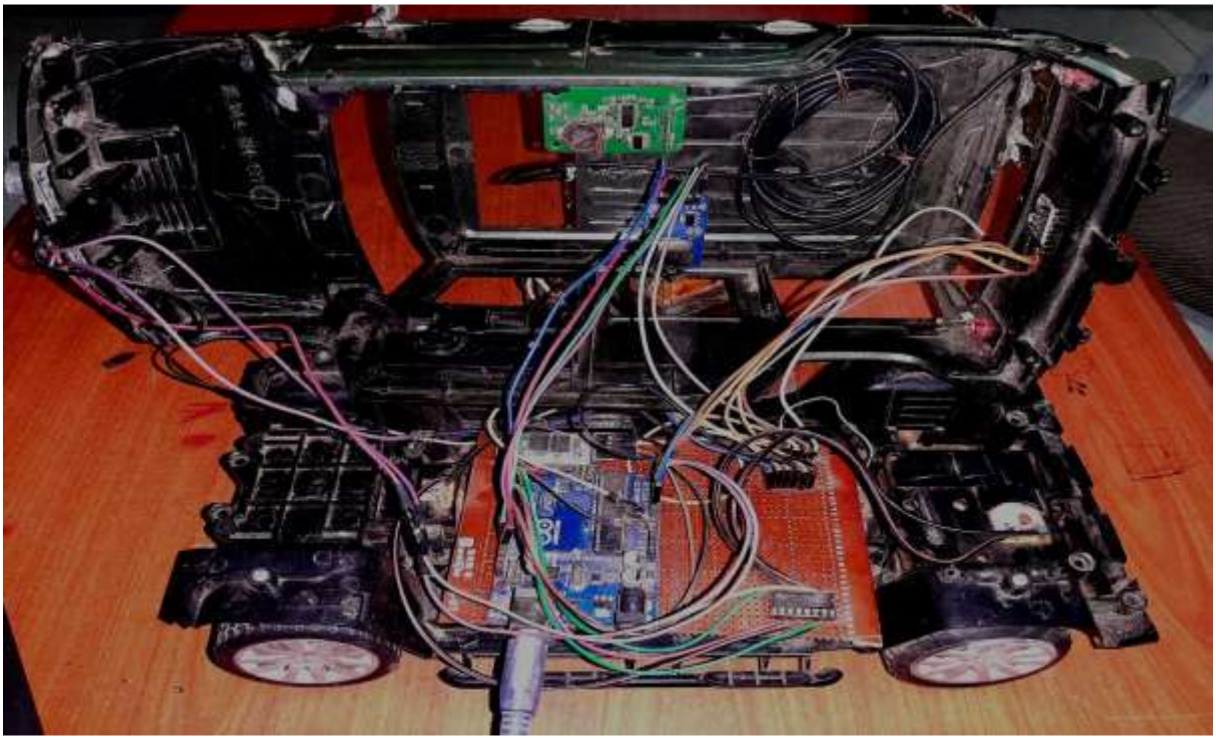

Figure (4-10) connection of circuit in prototype

4.3. Results

The results showed that the proposed system could navigate the car to move from point to another in a selected path without collision obstacles. In addition, it could detect the location and direction for the car any moment and displayed them in application on smart phone.

4.3.1. Results of navigation and movement

Start point and end point was selected on google earth as shown in figure (4.5) then the coordinates of path was stored to SD card. After that, Arduino was waiting for GPS .When GPS sent latitude and longitude to Arduino, the program will start. Firstly, Bearing was calculated then the car turned to correct direction as shown in figure (4-11). Then the car kept moving in straight way to nearest waypoint as shown in figure (4-12). Finally, the car stopped when arrived to the waypoint.

Figure (4-11) correct the direction

Figure (4-12) moving to waypoint

4.3.2. Results of avoiding collision with obstacles

When the car was moving, ultrasonic sensors was reading the distances between the car and any obstacles. When the car faced an obstacle, it started algorithm to avoid collision with the obstacle as shown in figure (4-13).

Figure (4-13) avoiding the obstacle

4.3.3. Link of video for implementation the project

https://youtu.be/V80WX5BoCwQ [34].

4.4. Discussion and problem faced

The proposed framework of this research is how to build an integrated system that is based on simple types of techniques to achieve a complex work called self-driving car. Self-driving provide many advantages as : reduce the rate of traffic accidents that happened due to human mistake, freedom of time, reduce traffic crowd , help people with special need for traveling and availability to choose a preferable path. Despite advanced researchers is using a modern techniques as internet of thing (IOT), artificial intelligent (AI) and image processing, this proposed system achieve an acceptable results in self-driving car field. The results evinced that this system is able to control the car for traveling between tow points independently, detecting the direction, calculate remaining distance, avoiding collision obstacles correctly. There is two main problems was faced:

- The direct calculation between the first and the end points was giving a wrong result .It might give a blocked path or a path through buildings. Therefore, google earth program was used to locate possible tracks and highlight the turning points. The coordinates of path and turning points were stored to a memory. This step helped to resolve another problem. When the car moved under a tree or any cover the GPS signal became weak. So, if the GPS stop for a small while (5-10 seconds), the car will move according to the last comparison between GPS and stored points.

- GPS amateur chips do not work well on indoor places and be affected by weather. In addition, they take a long time to connect to the satellites in first time running (5-10 minutes).depending on a car place-work, a small adjusting on the proposed system solved this problem. GPS sensor in a smart phone has good advantages that could be solved this problem.it has an acceptable accuracy (5-10 meters), fast reading and availability to read on closed places. Therefore, the system was supplied the car position by the GPS phone sensor. To achieve that, Bluetooth module was used as a master-slave instead of slave only.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATIONS

Chapter Five

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Prototype module to drive a car automatically has been designed and tested and it was successful at its major purpose of controlling a car correctly and safely. The results showed that the car could depend on GPS coordinates to move in the desired path and avoid collision obstacles. The path was detected by google earth to choose a preferable path. The information was stored on SD card .Then; GPS chip detected the initial coordinates in real time. The Arduino calculated the bearing and distance between the position and next waypoint .the car would turn to correct direction according to gyroscope sensor. The car would avoid collision with obstacles by tow ultrasonic sensors. The manual process in this prototype is detecting the path and storing them to SD card.

5.2 Recommendations

- Use GPS API to send the coordinates of path automatically instead of SD card
- Use the image processing to achieve traffic signs and to detect the size and kind of obstacles.
- Design a system to control the car manually depends on user need and in an emergency situation.
- Use a suitable sensor to avoid the excavation and digging.

REFERENCES

REFERENCES

[1] Road Traffic Injuries-World Health Organization- https://www.who.int/health-topics/road-safety#tab=tab_1

[2] Anderson M. James, Kalra Nidhi, Stanley D. Kalyn, "Autonomous Vehicle Technology"-Rand corporation,1st ed,2016.

[3] J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric Beacons", IEEE Transactions on Robotics and Automation, 1991.

[4] Ronald K Jurgen, "Autonomous vehicles for safer driving", Training Corporation , 2007.

[5] Arnay, R., Morales, N., Morell, A., Hernandez-Aceituno, J., Perea, D., Toledo,J. T., Hamilton, A., Sanchez-Medina, J. J., & Acosta, L. "Safe andreliable path planning for the autonomous vehicle verdino". IEEE Intelligent Transportation Systems Magazine, (2016)..

[6] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, "Autonomous driving in urban environments: Boss and the urban challenge."Journal of Field Robotics, 2008.

[7] Tomislav Kos, Ivan Markezic, and Josip Pokrajcic," Effects of multipath reception on GPS positioning performance", Robotics and Automation (ICRA), 2010.

[8] Jesse Levinson and Sebastian Thrun," Map-based precision vehicle localization in urban environments. Robotics", Science and Systems, 2007.

[9] Rintanen K et al “Development of an autonomous navigation system for an outdoor vehicle”. Control Eng.Practice,Vol. 4, No. 4, (1996)

[10] Bennet Fischer, TinoschGanjineh, DanielG◌hring, Sebastian Hempel, et al. "Technical report autonomous ",AutoNOMOS Labs, 2011.

[11] ZJ Chong, Baoxing Qin, Tirthankar Bandyopadhyay, MarceloHAng, Emilio Frazzoli, and Daniela Rus."Synthetic 2d LIDAR for precise vehicle localization in 3d urban environment" In Robotics and Automation (ICRA), 2013.

[12] M. Bojarski, D.D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal,et al., “End to End learning for self-driving cars”, 2016.

[13] A. Elfes. "Using occupancy grids for mobile robot perception and navigation Computer", Transportation Science, June 1989.

[14] Chen, L., Fan, L., Xie, G., Huang, K., & Nuchter, A. "Moving-object detection from consecutive stereo pairs using slanted plane smoothing". IEEE Transactions on Intelligent Transportation Systems, 2017.

[15] Juan Nieto, Jose Guivant, and EduardoNebot."Anovel hybrid map representation for denseslam in unstructured large environments. Technical report, Australian Center for Field Robotics, Department of Mechanical and Mechatronic Engineering, The University of Sydney, 2003

[16] P. Misra and P. Enge. "Global positioning system: signals, measurements, and performance". Ganga-Jamuna Press, 2011.

[17] S Hofmann, C Kuntzsch, MJ Schulze, D Eggert, andMSester. "Accuracy analysis of a low-cost platform for positioning and navigation". ISPRS Annals of Photogrammetry, 2012.

[18] J.S. Levinson, S. Thrun, D. Koller, M. Levoy." Automatic Laser Calibration, Mapping, and Localization for Autonomous Vehicles". Stanford University, 2011

[19] Chu, K., Kim, J., Jo, K., & Sunwoo, M. , "Real-time path planning of autonomous vehicles for unstructured road navigation", International Journal of Automotive Technology, (2015).

[20] Shih-I Chen et al. “The Design of Embedded GPS Navigation System Based on Internet Structure” 3rd ed, navigation and communication, 2009

[21] Amundson I & Koutsoukos XD,"A survey on localization for mobile wireless sensor networks". In R. Fuller & X.D. Koutsoukos, Mobile Entity Localization and Tracking in GPS-Less Environments. Berlin: Springer, 2009.

[22]Gregory T. French,"Understanting the GPS an introduction to the Global Positioning System What it is and How it works",GeoResearch, Inc,1996.

[23] Xiuyan Guo," Feature-Based Localization Methods for Autonomous Vehicles",Freien University Berlin, Feb-2017.

[24] Hofmann-Wellenhof, B., Lichtenegger, H., and Collins, J., " Global Positioning System: Theory and Practice "(5th ed). Springer. 2008.

[25] Davide Ronzoni, Roberto Olmi, Cristian Secchi, and Cesare Fantuzzi. Agv global localization using indistinguishable artificial landmarks. IEEE International Conference ,2011.

[26] Gavish, M., andWeiss, A.J. "Performance analysis of bearing-only target location algorithms". IEEE Transactions on Aerospace and Electronic Systems,1992

[27] Hightower, J., and Borriello, G. (2001) "Location systems for ubiquitous computing. Computer",2001.

[28] Shang, Y., Ruml, W., Zhang, Y., and Fromherz, M. (2004) Localization from connectivity in sensor networks. IEEE Transactions on Parallel and Distributed Systems

[29] Tekdas, O., and Isler, V. (2007) Sensor placement algorithms for triangulation based localization. Proc. of the IEEE International Conference on Robotics and Automation

[30] R¨omer, K. (2003) The lighthouse location system for smart dust. Proc. of the 1st International Conference on Mobile Systems, Applications and Services

[31] Dhanasingaraja R, Kalaimagal S, Muralidharan G "Autonomous Vehicle Navigation and Mapping System" Madras Institute of Technology, Anna University, India,2014

[32] Tanveer Hossen Sakkhor, Samin Saksiat Zaman, et all," "Autonomous Car Using Full Mapping GPS System"BRAC University, 2015.

[33] Emilio Hernández Chiva , " Design and construction of an electric autonomous driving vehicle", Barcelona Tec university 2016 .

[34] https://youtu.be/V80WX5BoCwQ

APPENDIXES

Appendix A : Arduino code

```
// sd: cs 2 , MOSI - pin 11,  MISO - pin 12, CLK - pin 13
//gps: rx 3 ,tx 4
// motors 5,6,7,8
//blutooth rx 9,tx 10
// ultarasonic1  trigPin = A5; echoPin = A4
//ultrasonic2 trigPin2 = A2 echoPin2 = A3
//  * A4            |  SDA  * A5            |  SCL
#include <SoftwareSerial.h>
#include <SPI.h>
#include <SD.h>
#include <math.h>
#include <Wire.h>
SoftwareSerial blu(9,10);
File myFile;
int trigPin = A0;      // trig pin of HC-SR04
int echoPin = A1;     // Echo pin of HC-SR04
int trigPin2 = A2;      // trig pin of HC-SR04
int echoPin2 = A3;     // Echo pin of HC-SR04
long duration, distance,duration2, distance2;
float x1,x2,y1,y2;
int t;
int clk=9;
int dt=10;
float test;
double p0,p1,p2,p3;
float DES;
String inString = "";
String firstVal, secondVal, thirdVal, forthVal;
float elapsedTime, time, timePrev;        //Variables for time control
int gyro_error=0;                       //We use this variable to only calculate once the gyro data error
```

```
float Gyr_rawX, Gyr_rawY, Gyr_rawZ;     //Here we store the raw data read
float Gyro_angle_x, Gyro_angle_y;          //Here we store the angle value obtained with Gyro data
float Gyro_raw_error_x, Gyro_raw_error_y; //Here we store the initial gyro data error
//Acc Variables
int acc_error=0;                        //We use this variable to only calculate once the Acc data error
float rad_to_deg = 180/3.141592654;        //This value is for pasing from radians to degrees values
float Acc_rawX, Acc_rawY, Acc_rawZ;    //Here we store the raw data read
float  Acc_angle_y;          //Here we store the angle value obtained with Acc data
float  Acc_angle_error_y; //Here we store the initial Acc data error
float  Total_angle_y;
void setup(){
Serial.begin(9600);
blu.begin(9600);
 pinMode(trigPin, OUTPUT);         // set trig pin as output
  pinMode(echoPin, INPUT);
   pinMode(trigPin2, OUTPUT);         // set trig pin as output
  pinMode(echoPin2, INPUT);  //set echo pin as input to capture reflected waves
pinMode(5,OUTPUT);
pinMode(6,OUTPUT);
pinMode(7,OUTPUT);
pinMode(8,OUTPUT);
   digitalWrite(5,LOW);
   digitalWrite(6,LOW);
   digitalWrite(7,LOW);
   digitalWrite(8,LOW);

 Wire.begin();
 Wire.beginTransmission(0x68);
 Wire.write(0x6B);
 Wire.write(0x00);
  Wire.endTransmission(true);
```

```
  //Gyro config
  Wire.beginTransmission(0x68);
  Wire.write(0x1B);
  Wire.write(0x10);
  Wire.endTransmission(true);
  //Acc config
  Wire.beginTransmission(0x68);
  Wire.write(0x1C);
  Wire.write(0x10);
  Wire.endTransmission(true);
  time = millis();

}
void loop(){
  while (blu.available() > 0)
  {
    inString = blu.readString()
    for (int i = 0; i < inString.length(); i++)
    {
      if (inString.substring(i, i+1) == "*")
      {
        firstVal = inString.substring(1, i);
        secondVal = inString.substring(i+1);
        //Serial.println("Latitude: " + (firstVal) + ", second: " + (secondVal));
      }
    }
    for (int i = 0; i < secondVal.length(); i++)
    {
      if (secondVal.substring(i, i+1) == "%")
      {
        thirdVal = secondVal.substring(0, i);
    }
  }
```

```
//ultrafront
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);     // send waves for 10 us
  delayMicroseconds(10);
  duration = pulseIn(echoPin, HIGH); // receive reflected waves
  distance= duration*0.034/2;  // convert to distance
  Serial.println("distance");
  Serial.println(distance);
  //  blu.println("distance1");
//blu.println(distance);

//ultraback
    digitalWrite(trigPin2, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin2, HIGH);     // send waves for 10 us
  delayMicroseconds(10);
  duration2 = pulseIn(echoPin2, HIGH); // receive reflected waves
  distance2= duration2*0.034/2;  // convert to distance
  Serial.println("distance2");
  Serial.println(distance2);

delay(500);
    Serial.println("my position");
    x1=firstVal.toFloat();
 Serial.println("Latitude  x1 = ");
    Serial.println(x1,6);
      y1=secondVal.toFloat();
            Serial.println("Longitude y1 = ");
                  Serial.println(y1,6);
      delay(1000);
   if (distance < 25 && distance2 >25)
  {
```

```
digitalWrite(5, LOW);
digitalWrite(6, LOW);
 digitalWrite(7, LOW);
  digitalWrite(8, LOW);
  delay (200);
  digitalWrite(5,HIGH); //BACK
   digitalWrite(6, LOW);
    digitalWrite(7, HIGH); //left
     digitalWrite(8, LOW);
   delay(500);
    digitalWrite(5, LOW);
     digitalWrite(6, LOW);
    digitalWrite(7, LOW);
     digitalWrite(8, LOW);
     delay(200);
      digitalWrite(5, LOW);
       digitalWrite(6, HIGH);//forward
        digitalWrite(7, HIGH);//RIGHT
         digitalWrite(8, LOW);
         delay(500);
           digitalWrite(5, LOW);
       digitalWrite(6, HIGH);//forward
        digitalWrite(7, HIGH);//right
         digitalWrite(8, LOW);
         delay(1000);}

         else if (distance < 25 && distance2 <25){
          digitalWrite(5, LOW);
           digitalWrite(6, LOW);
            digitalWrite(7, LOW);
             digitalWrite(8, LOW);}
read_sd();}}
  void read_sd(){ ;
```

```
   while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
  Serial.print("Initializing SD card...");
  if (!SD.begin(2)) {
    Serial.println("initialization failed!");
    while (1);
  }
  Serial.println("initialization done.");
  // re-open the file for reading:
  myFile = SD.open("345.txt");
  if (myFile) {
    Serial.println("123.txt:");
    // read from the file until there's nothing else in it:
    while (myFile.available()) {
int  y=0;
  int i=0;
  double b;
String xx=myFile.readStringUntil('\0');
int   x =xx.indexOf("LatLon",x);

   delay(500);
  char c,a;
  String c0,c1;

if (xx.startsWith("LatLon",x)){

   for (i=0;i<9;i++){
    c=(xx[x+9]);
      Serial.println(xx[x+9]);
    a=(xx[x+22]);
   c0 +=(char)c;
   c1 +=(char)a;
```

```
      x++;
    }}
    y = 0;
    i=0;
   delay(500);
  x=x+1;
 b=c0.toFloat();
 Serial.println("x2=");
 Serial.println(b,6);
  Serial.println ("y2=");
   Serial.println(c1.toFloat(),6);
    }
    myFile.close();
    delay(500);
  }
 comp();
  }
void  comp(){
  const double pi=(3.141592654);
   const double rad=(pi/180);
   p0=x1*rad;
   p1=x2*rad;
   p2=y1*rad;
   p3=y2*rad;
    double q0=atan2(sin(p3-p2)*cos(p1),cos(p0)*sin(p1)-(sin(p0)*cos(p1)*cos((p3-p2))));
  float q =q0*(180/pi);
  DES=acos(sin(p0)*sin(p1)+cos(p0)*cos(p1)*cos(p3-p2))*637000;
   Serial.println("DES");
      Serial.println(DES);
  while  (DES >= 5 && distance > 25 && distance2 >25){
      digitalWrite(5, LOW);
              digitalWrite(6, HIGH);
               digitalWrite(7, LOW);
```

```
              digitalWrite(8, LOW);
              delay(500);
             Serial.println(DES);
              }
 if (q>=20 &&q<=-20){
do{
 gye();
  if (q>Total_angle_y){
           digitalWrite(5, LOW);
            digitalWrite(6, HIGH);
             digitalWrite(7, HIGH);
              digitalWrite(8, LOW);}
}
   while(Total_angle_y>q);
        }}

   void gye(){
     if(acc_error==0)
 {
   for(int a=0; a<200; a++)
   {
     Wire.beginTransmission(0x68);
     Wire.write(0x3B);                       //Ask for the 0x3B register- correspond to AcX
     Wire.endTransmission(false);
     Wire.requestFrom(0x68,6,true);

     Acc_rawX=(Wire.read()<<8|Wire.read())/4096.0 ; //each value needs two registres
     Acc_rawY=(Wire.read()<<8|Wire.read())/4096.0 ;
     Acc_rawZ=(Wire.read()<<8|Wire.read())/4096.0 ;
     Acc_angle_error_y = Acc_angle_error_y + ((atan(-1*(Acc_rawX)/sqrt(pow((Acc_rawY),2) + pow((Acc_rawZ),2)))*rad_to_deg));

     if(a==199)
```

```
      Acc_angle_error_y = Acc_angle_error_y/200;
      acc_error=1;
    }
  }
//end of acc error calculation

if(gyro_error==0)
{
  for(int i=0; i<200; i++)
  {
    Wire.beginTransmission(0x68);            //begin, Send the slave adress (in this case 68)
    Wire.write(0x43);                        //First adress of the Gyro data
    Wire.endTransmission(false);
    Wire.requestFrom(0x68,4,true);           //We ask for just 4 registers
    Gyr_rawY=Wire.read()<<8|Wire.read();

    Gyro_raw_error_y = Gyro_raw_error_y + (Gyr_rawY/32.8);
    if(i==199)
    {
      Gyro_raw_error_y = Gyro_raw_error_y/200;
      gyro_error=1;
    }
  }
}//end of gyro error calculation
timePrev = time;                        // the previous time is stored before the actual time read
time = millis();                        // actual time read
elapsedTime = (time - timePrev) / 1000; //divide by 1000 in order to obtain seconds


  Wire.beginTransmission(0x68);            //begin, Send the slave adress (in this case 68)
  Wire.write(0x43);                        //First adress of the Gyro data
  Wire.endTransmission(false);
  Wire.requestFrom(0x68,4,true);           //We ask for just 4 registers
```

```
  Gyr_rawY=Wire.read()<<8|Wire.read();

  Gyr_rawY = (Gyr_rawY/32.8) - Gyro_raw_error_y;

  Gyro_angle_y = Gyr_rawY*elapsedTime;


 Wire.beginTransmission(0x68);     //begin, Send the slave adress (in this case 68)
 Wire.write(0x3B);                 //Ask for the 0x3B register- correspond to AcX
 Wire.endTransmission(false);      //keep the transmission and next
 Wire.requestFrom(0x68,6,true);    //We ask for next 6 registers starting withj the 3B

 Acc_rawX=(Wire.read()<<8|Wire.read())/4096.0 ; //each value needs two registres
 Acc_rawY=(Wire.read()<<8|Wire.read())/4096.0 ;
 Acc_rawZ=(Wire.read()<<8|Wire.read())/4096.0 ;

 Acc_angle_y = (atan(-1*(Acc_rawX)/sqrt(pow((Acc_rawY),2) + pow((Acc_rawZ),2)))*rad_to_deg) - Acc_angle_error_y;


 Total_angle_y = 0.98 *(Total_angle_y + Gyro_angle_y) + 0.02*Acc_angle_y;
   Serial.print("Yº: ");
 Serial.print(Total_angle_y);
 Serial.println(" ");}
```

Appendix B : Some final Project Images