

DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**EOG SIGNAL INTERFACE BASED
AUTONOMOUS NAVIGATION OF
WHEELCHAIRS FOR DISABLED PEOPLE**

by
Lütfi MUTLU

February, 2017
İZMİR

EOG SIGNAL INTERFACE BASED AUTONOMOUS NAVIGATION OF WHEELCHAIRS FOR DISABLED PEOPLE

**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of
Philosophy in Mechatronics Engineering, Mechatronics Engineering Program**

**by
Lütfi MUTLU**

**February, 2017
İZMİR**

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled “EOG SIGNAL INTERFACE BASED AUTONOMOUS NAVIGATION OF WHEELCHAIRS FOR DISABLED PEOPLE” completed by LÜTFİ MUTLU under supervision of PROF. DR. ZEKİ KIRAL and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.



Prof. Dr. Zeki KIRAL

Supervisor



Prof. Dr. Mehmet KUNTALP

Thesis Committee Member



Assist. Prof. Dr. Aytaç GÖREN

Thesis Committee Member



Prof. Dr. Lale Canan DÜLGER

Examining Committee Member



Prof. Dr. Hakan TEMELTAŞ

Examining Committee Member



Prof. Dr. Emine İlknur CÖCEN

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

Firstly, I would like to thank to my supervisor Prof. Dr. Zeki Kır  l and my previous supervisor Prof. Dr. Erol Uyar for the supervision of this work and their flexibility, Prof. Dr. Mehmet Kuntalp and Assist. Prof. Dr. Ayta   G  ren for their support and advices during our thesis meetings.

I would like to thank Yusuf Sait Erdem for his valuable contributions on Linux and EOG signal processing software, Mehmet Alperen   ener and Mert Altınta   for motor drivers throughout this study.

This work has been carried out at Dokuz Eyl  l University in   zmir, Turkey and financially supported by T  B  TAK (The Scientific and Technological Research Council of Turkey) with 2130182 and 7150628 numbered TEYDEB (T  B  TAK's Technology and Innovation Grant Programmes Directorate) and 2013-40 numbered KOSGEB (Small and Medium Industry Development Organization of Turkey) projects.

Above all, I want to dedicate my thesis to my beloved family, my mother Ay  e Mutlu, my father H  seyin Mutlu and my brother Nuri Mutlu for their unlimited patience throughout this project and my entire life.

L  tfi MUTLU

EOG SIGNAL INTERFACE BASED AUTONOMOUS NAVIGATION OF WHEELCHAIRS FOR DISABLED PEOPLE

ABSTRACT

In this thesis, a smart wheelchair as an autonomous mobile robot for transportation and communication needs of disabled people who are unable to use their hands and feet has been developed. For this purpose; real time mapping, localisation, adaptive shortest path finding, autonomous navigation and control algorithms are developed and combined to transport the user to the desired destination point even if in previously unknown environments without any landmark and without commanding every step of the vehicle. Two prototype wheelchairs are developed: the first one works with Windows Operating System (OS) with encoder and Inertial Measurement Unit (IMU) based localisation algorithm. The second prototype works with Linux Ubuntu OS with encoder and Light Detection And Ranging (LIDAR) based localisation algorithm without IMU.

Destination points have been assigned with just one mouse click on the map of Graphical User Interface (GUI) by using electrooculography (EOG) and gyroscope signals of the user. Thus, the commands for transportation given by the user with EOG signals are reduced, became easier and practical.

Besides this, as an additional feature, a special screen keyboard is developed to make computer easier to use and faster to write without using hands. New and smart wheelchair for disabled people is achieved with the developed system.

Keywords: Localisation, mapping, EOG, optimal path finding, navigation control, optimisation, disabled people

ENGELLİ İNSANLARI İÇİN TEKERLEKLİ SANDALYELERİN EOG SINYAL ARAYÜZ TABANLI OTONOM YÖNELTİLMESİ

ÖZ

Bu tezde, ellerini ve ayaklarını kullanamayan kişilerin ulaşım ve iletişim ihtiyaçları için otonom bir mobil robot olarak akıllı bir engelli aracı geliştirilmiştir. Bu amaçla, gerçek zamanlı haritalama, konumlandırma, adaptif en kısa yol bulma, otonom navigasyon ve kontrol algoritmaları geliştirilmiş ve kullanıcının gitmek istediği noktaya daha önceden bilinmeyen bir ortam olsa bile her adamını kumanda etmeden ve herhangi bir yer işareti olmadan gitmesi için entegre edilmiştir. İki prototip engelli aracı geliştirilmiştir: ilki enkoder ve ataletsel ölçüm birimi (IMU) tabanlı konumlandırma algoritması ile Windows İşletim Sistemi'nde çalışmaktadır, ikinci prototip ise enkoder ve lazer mesafe ölçüm sensörü (LIDAR) tabanlı konumlandırma algoritması ile IMU olmadan çalışmaktadır.

Hedef noktalar kullanıcının elektrookülografi (EOG) ve cayro sinyalleri kullanılarak grafiksel kullanıcı arayüzündeki harita üzerinde sadece bir fare tıklaması ile belirlenebilmektedir. Böylece EOG sinyalleri ile kullanıcı tarafından ulaşım için verilen komutlar azaltılmış, daha kolay ve pratik olmuştur.

Bunun yanında, ek bir özellik olarak, elleri kullanmadan daha hızlı yazı yazabilmek ve bilgisayar kullanımını kolaylaştırmak için özel ekran klavyesi geliştirilmiştir. Geliştirilen sistem ile engelli kişilerin kullanımı için yeni ve akıllı bir engelli aracı geliştirilmiştir.

Anahtar kelimeler: Konumlandırma, haritalama, EOG, optimum yol bulma, navigasyon kontrolü, optimizasyon, engelli kişiler

CONTENTS

	Page
Ph.D. THESIS EXAMINATION RESULT FORM	ii
ACKNOWLEDGMENTS	iii
ABSTRACT	iv
ÖZ	v
LIST OF FIGURES	x
LIST OF TABLES	xiii
 CHAPTER ONE – INTRODUCTION	 1
1.1 Introduction	1
1.2 Aim of the Thesis	2
1.3 Autonomous Mobile Robots	3
1.4 Obstacle Avoidance	5
1.4.1 Classical Motion Planners.....	6
1.4.2 Heuristic Planning.....	6
1.4.3 "Complete and Correct " Sensor-Based Path Planning	7
1.5 Wheeled Mobile Robots (WMR) Research Projects	8
1.5.1 Automated Highway System (AHS).....	8
1.5.2 RHINO	9
1.5.3 SuperMARIO	9
1.5.4 Artificial Neural Networks.....	9
1.5.5 Roadmap Robots	10
1.5.6 Virtual Potential Field Robots – CARMEL	10
1.5.7 Other Virtual Potential Field Implementations	10
1.6 Wheelchairs Types for Disabled People Who are Unable to Use Their Hands and Feet.....	11
1.6.1 Wheelchair Controller Based On Brain Wave	11
1.6.2 Navigation System Using Brain Wave Signal and Navigation Control Method.....	12

1.6.3 Brainwave Responsive Wheelchair.....	12
1.7 Latest Developments (2013-2017).....	13
1.8 Significance of the Thesis	16
1.9 Outline of the Thesis	17
 CHAPTER TWO - GENERAL OPERATION ALGORITHM AND COMPONENTS OF THE WHEELCHAIR.....	19
2.1 General Operation Algorithm of the Wheelchair.....	19
2.2 Inertial Navigation System (INS).....	22
2.2.1 Accelerometer Sensors.....	22
2.2.2 Inertial Measurement Unit (IMU).....	23
2.2.3 IMU Data Collection and Validation	24
2.3 Laser Imaging Detection and Ranging (LIDAR).....	26
2.3.1 Operating Principle of the LMS.....	27
2.3.2 Distance Measurement	28
2.3.3 Direction Measurement	28
2.3.4 Influences of Object Surfaces on the Measurement.....	28
2.3.5 Scanning Range of the LMS	29
2.3.6 Data Interfaces	30
2.3.7 Data Communication Using Messages	30
2.3.8 Laser Radiation Safety	31
2.3.9 Mapping with LMS via Ethernet and Serial Port.....	31
2.4 Encoders	31
2.5 GPS Measurements with Kalman Filter.....	33
2.5.1 GPS Measurement Example 1:	34
2.5.2 GPS Measurement Example 2:	36
 CHAPTER THREE - MECHANICS OF THE WHEELCHAIR.....	39
3.1 Mechanical Parts of the Wheelchair	39
3.1.1 PC/Tablet/Monitor Holder	39

3.1.1.1 PC Holder for First Prototype	39
3.1.1.2 Tablet/Monitor Holder for Second Prototype	40
3.1.2 LIDAR Holder	42
3.1.2.1 LIDAR Holder for First Prototype.....	42
3.1.2.2 LIDAR Holder for Second Prototype	43
3.2 Kinematic Equations of the Wheelchair	44
 CHAPTER FOUR - ELECTRONICS OF THE WHEELCHAIR.....	51
 CHAPTER FIVE - THE GRAPHICAL USER INTERFACE (GUI) OF THE WHEELCHAIR.....	57
5.1 GUI for First Prototype on Windows OS.....	58
5.1.1 Path Finding Algorithms	59
5.1.1.1 Almost Shortest Path Finding with Static Obstacles	59
5.1.1.2 Adaptive Shortest Path Finding with Dynamic Obstacles	59
5.1.2 LIDAR Mapping Algorithms.....	61
5.1.3 MAP Processing.....	65
5.1.4 Video Recording and Playback Feature to Analyse Vehicle Behaviours	72
5.1.5 GUI Tests with Windows Tablet PC.....	73
5.2 Mapping with ROS (Robot Operating System) in Linux Ubuntu OS	74
5.3 Special Screen Keyboard	77
 CHAPTER SIX - PROCESSING OF EOG SIGNALS	80
6.1 Measurement of EOG Signals.....	80
6.2 Analysis of EOG Signals	84
6.3 Hands-Free Mouse Control Program to Assign Destination Points.....	88
 CHAPTER SEVEN - IMPLEMENTATIONS AND TESTING.....	90
7.1 Final Navigation Algorithms	90

7.2 Curved Shortest Path Estimation and Kinematic Equations	94
7.3 Real Environment Test.....	100
7.3.1 First Prototype.....	100
7.3.2 Second Prototype	101
CHAPTER EIGHT – CONCLUSIONS.....	105
8.1 Overview	105
8.2 Future Work	106
REFERENCES.....	108
APPENDICES	112

LIST OF FIGURES

	Page
Figure 2.1 Integrated approaches	19
Figure 2.2 General schematic and conceptual prototype	20
Figure 2.3 Electronic communication scheme	21
Figure 2.4 Euler angles, angular velocities and acceleration data	24
Figure 2.5 Free IMU (on the right) is used instead of ArduPilot board (on the left). 25	25
Figure 2.6 IMU software.....	26
Figure 2.7 SICK LMS100 laser measurement sensor.....	27
Figure 2.8 Measuring principle of the LMS.....	27
Figure 2.9 Principle of operation for pulse propagation time measurement.....	28
Figure 2.10 Reflection of the laser beam at the surface of an object.....	29
Figure 2.11 Scanning range of the LMS100 as a function of the target remission....	29
Figure 2.12 Laser warning label on the LMS	31
Figure 2.13 Atek ARS HS 50 256 HPL 3Y-C encoder	32
Figure 2.14 Encoder directly connected to motors for first prototype.....	32
Figure 2.15 SICK DBS36E-BBEK00360 incremental blind hollow shaft encoder ..	33
Figure 2.16 ArduPilot Mega 2.5 with GPS sensor.....	34
Figure 2.17 GPS measurements (blue) on map.....	34
Figure 2.18 Raw data (blue) and filtered results (red) while standing at a point.....	35
Figure 2.19 Combination of Google Map view with MATLAB figure.....	35
Figure 2.20 Raw (blue) and filtered (red) GPS position data	36
Figure 2.21 Superposition of raw (blue and black) and filtered (purple and red) GPS position data	37
Figure 3.1 Created assembly model with real wheelchair	39
Figure 3.2 Unbalanced old PC holder V.1	40
Figure 3.3 More balanced and ergonomic PC holder V.2.....	40
Figure 3.4 3D CAD model of tablet/monitor holder.....	41
Figure 3.5 Implementation of the functional tablet/monitor holder	41
Figure 3.6 Lilliput 9.7" resistive touch screen monitor.....	42
Figure 3.7 First designs for LIDAR connection part	42

Figure 3.8 Final design of the vehicle with LIDAR	43
Figure 3.9 Protection cage of LIDAR produced by 3D printer.....	43
Figure 3.10 Protection cage of LIDAR under legroom of the wheelchair for second prototype.....	44
Figure 3.11 Combined model of distance (D) and orientation (Q) controls in MATLAB Simulink	45
Figure 3.12 Velocities of left and right motors of the wheelchair	46
Figure 3.13 Coordinate system locations of the wheelchair	47
Figure 3.14 Left and right motor velocities of the wheelchair.....	48
Figure 4.1 Batteries, charging unit and power connections	47
Figure 4.2 First motor driver V1.1	52
Figure 4.3 Second motor driver V1.2 and Arduino Mega Board as main control board.....	53
Figure 4.4 Motor speed tests	53
Figure 4.5 Double sided dual motor driver V2.1	54
Figure 4.6 Main Control Board with double sided dual DC Motor Driver Board V2.2	54
Figure 4.7 Final layout of electronic system of first prototype	55
Figure 4.8 Control board software	55
Figure 4.9 Final layout of electronic system of second prototype	56
Figure 5.1 PC-User interface program written in VB.NET	58
Figure 5.2 Almost Shortest Path	59
Figure 5.3 Adaptive (smart) optimal path finding algorithm.....	60
Figure 5.4 Visualisation of LIDAR data	62
Figure 5.5 Laser beams visualisation.....	63
Figure 5.6 Laser beam scanning for removing obstacles on each cell.....	63
Figure 5.7 Basic LIDAR Mapping Program.....	64
Figure 5.8 New LIDAR, SICK TIM551	65
Figure 5.9 Matrix style map format	65
Figure 5.10 Map with picture format	66
Figure 5.11 File Open Dialog and converted map in GUI.....	67
Figure 5.12 LIDAR mapping test with adaptive shortest path calculation	68

Figure 5.13 Three colour obstacle display	68
Figure 5.14 Improved obstacle displaying with forbidden areas	69
Figure 5.15 General software structure of GUI	71
Figure 5.16 Video recording feature of the GUI.....	72
Figure 5.17 Video playing feature of the GUI	73
Figure 5.18 Interface program works on a Windows tablet	73
Figure 5.19 Mapping example with LIDAR in Linux Ubuntu OS	75
Figure 5.20 Super position of LIDAR map data (purple area with green points) with the sketch of the test environment (red lines)	76
Figure 5.21 LIDAR map data read by the interface program on Windows OS.....	77
Figure 5.22 First (simple) version of screen keyboard	77
Figure 5.23 Last version of special screen keyboard with word and sentence prediction feature.....	78
Figure 6.1 Emotive EPOC headset	80
Figure 6.2 EOG signals and connection quality of sensors with green dots.....	80
Figure 6.3 Change in sensors O1 and O2 while closing eyes	81
Figure 6.4 Change in sensors AF3 and AF4 while winking	82
Figure 6.5 Conditions and commands (rules) regarding to these conditions.....	82
Figure 6.6 A sample of magnetic density on brain while thinking about something	83
Figure 6.7 3D Brain activity map.....	83
Figure 6.8 Three eye blinks in Excel Graph	84
Figure 6.9 Simply filtered signals	84
Figure 6.10 Cross Correlation of raw EOG signal with search template in MATLAB86	
Figure 6.11 EOG signal analysis.....	88
Figure 6.12 Recording of sample signals.....	88
Figure 6.13 Left/right eye blinks and mouse move tests	89
Figure 7.1 General algorithm	91
Figure 7.2 Combined Matrix calculation	92
Figure 7.3 Distance Matrix calculation.....	93
Figure 7.4 Curved Shortest Path drawing	93
Figure 7.5 Turning radius calculation of the vehicle	95
Figure 7.6 The change in the vehicle heading angle.....	96

Figure 7.7 Change in the vehicle position.....	97
Figure 7.8 Estimated curved path	98
Figure 7.9 Calculated trips for starting angle 0°	98
Figure 7.10 Round trips for starting angles a: 0° , b: 90° , c: 180° and d: 270°	99
Figure 7.11 Calculated paths for all starting angle possibilities a (left): $x=0, y=0$ for destination position and b (right): $x=0, y=0$ for starting position.....	99
Figure 7.12 Real-time autonomous driving tests with static obstacles	100
Figure 7.13 Real-time autonomous driving tests with dynamic obstacles.....	101
Figure 7.14 Estimated and performed path comprehension on initial practices (black: estimated path, blue: performed path)	102
Figure 7.15 Estimated and performed path comprehension on last practices (black: estimated path, blue: performed path)	103
Figure 7.16 Real-time autonomous driving tests with static and dynamic obstacles	104

LIST OF TABLES

	Page
Table 2.1 Measurement errors of static GPS sensor at 1st GPS point	33
Table 2.2 Measurement errors of dynamic GPS sensor at 2nd GPS Point	34
Table 6.1 Correlation values of each capture between Search Template	82
Table 7.1 Abbreviations used in Figure 7.1	86
Table 7.2 Rear wheel velocities related to θ	88
Table 7.3 Abbreviations used in equations	89



CHAPTER ONE

INTRODUCTION

1.1 Introduction

People have searched for solutions to live better and to get rid of diseases and problems that come across throughout history. Physical disability that prevent us to live like a normal healthy person can be classified as situations of elderliness, handicapped disability or being ill.

The World Health Organization (WHO) estimates that worldwide, approximately 100 million people need a classical wheelchair (Cooper, 2010). These people have an existing solution but some patients of Alzheimer Disease, Abbreviated Injury Scale, Amyotrophic Lateral Sclerosis, Fourth Cervical Vertebra, Cerebral Palsy, Cerebrovascular Accident, Glasgow Coma Scale, Multiple Sclerosis, Multiple System Atrophy, Parkinson Disease, Progressive Supranuclear Palsy, Spinal Cord Injury and Traumatic Brain Injury are unable to use classical wheelchairs and they need a smart wheelchair for transportation (Simpson et. al., 2008).

It is estimated that approximately 1.4-2.1 million people will benefit from a smart wheelchair within U.S (as of July, 2006) (Simpson et. al., 2008). The number of wheelchair users has grown at an average annual rate of 5.9 percent a year (LaPlante, 2003). So if we consider annual rate of increase in wheelchair users and population ratio between U.S. and the World, it is estimated that approximately 44 million people can benefit from a smart wheelchair in 2016. Thanks to developing sensor technology, there is some academic works and prototypes for catering to transportation needs of disabled people. Some of these are navigated by ElectroOculoGraphy EOG / ElectroEncephaloGraphy (EEG) signals and some navigated by sounds, biting/blowing etc. 3D LIDAR/imaging system, GPS/DGPS or high-performance GPS-Aided Inertial Navigation System (GPS/INS) which provide robust perception in unstructured outdoor environments are also integrated to smart wheelchair systems (Bakker et. al., 2010; Schwesinger et. al., 2016)

1.2 Aim of the Thesis

In daily life we come across with some disabled people on wheelchairs but there is a lot more disabled people than we observe because they have difficulties on walking. More striking than this, we could not see any disabled people who are unable to use their hands and feet even if their number is 44 million in the World because they have much more difficulties than the disabled people who can use their hands but not feet. They don't have a sufficient solution yet.

There is a lot of study in the world and some of them are mentioned in Section 1.5 for the people who are unable to use their hands and feet. These wheelchairs can be controlled by sounds, bites, blows or EOG/EEG signals but does not have an autopilot (autonomous control system) with global path planning algorithm. The user has to give commands in every step of the movement. Therefore it becomes an unpractical solution because it is not very easy to give commands with EOG/EEG signals or bites etc.

For these reasons the user must give less commands with more autonomy. So the main aim of the thesis is to transport people who are unable to use their hands and feet with a smart wheelchair which have an autopilot system.

Thanks to this autopilot system a practical transportation opportunity is presented by using only one mouse click (just assigning destination point on the map sketch view) without commanding every step of the vehicle. Also the traveling stability is increased by reducing the number of commands given by the user.

Some of these wheelchairs could not become a commercial product because of inadaptability to daily life so the thesis also aims to present a commercial product which is adaptable to daily life.

Another aim of the thesis is to present a wheelchair for rehabilitation and sports facilities for some of the people who are unable to use their hands and feet like MS patients who lose sense of direction.

Besides these the thesis also aims to make computer easier to use and to make writing faster with a special keyboard which can be controlled by EOG signals or with just eye blinks without by using hands.

The special keyboard of the smart wheelchair system is aimed to use in the following fields for disabled people:

- ✓ Employment at home. (Some of computer based works)
- ✓ Education by using e-universities and some online education system
- ✓ Social network, computer games, e-government applications
- ✓ Telephone usage with some computer software like Skype etc.
- ✓ TV-radio usage with internet or TV cards connected to PC.
- ✓ Environmental devices (smart home systems, light/curtain control, air conditioner etc.) control with computer programs

1.3 Autonomous Mobile Robots

Autonomous robotic vehicle navigation relies on the vehicle being able to know where it is to an adequate degree of accuracy, and also to be able to sense the environment around it as required. The guidance and the navigation of unmanned vehicles are important issues for making autonomous robots. If a robot is going to make a decision about performing a task which is expected from it, first of all it has to know its whereabouts and state in the working environment (Roumeliotis et. al., 2000). For example, people use their eyes and brain for feedback from their surroundings to decide what to do next, depending on the task they will perform. Nerves that are responsible of seeing create the image on the retina part of the eye and then send the information to the brain for processing. Then, the brain makes a judgment about how to act in this situation, and send the commands via nerves to the necessary parts of the body.

Likewise, if a robot, whether mobile or not, is required to achieve a mission, it has to get feedback from its environment. That being so, the robot uses some sensing devices to be aware of its current situation; which may include force, radiation, temperature, direction, inclination, acceleration, velocity or position. If the system in question is assumed to carry out navigational demands, these feedback data should include the vehicle's position, velocity and acceleration. In order for a robot to determine its present position accurately, first it has to take measurements from the real world and then post-process these data to eliminate the uncertainties that are always inherent in the measurement information. Afterwards, the robot can be said to have a reliable estimate of its current position which may be used for navigation purposes (Negenborn, 2003).

An autonomous mobile robot is a robot which is moving and changing its workspace aiming to complete its tasks in limitations of the rules given or the rules it develops. In kinematic analysis of mobile robots, there are four main differences for kinematic analysis of robot manipulators (Muir, 1986).

- a. Stationary manipulators only form closed chains when they are contact with fixed objects whereas; wheeled mobile robots form many closed chains at the same time.
- b. The contact between a wheel and plane forms a higher-pair, but stationary manipulators contain only lower-pair joints.
- c. In wheeled mobile robots, only some degrees of freedom of a wheel are actuated. However, all DOFs of each joint of a stationary manipulator have at least one actuator.
- d. Each joint in stationary manipulator has position and velocity sensors. In wheeled mobile robots, only some degrees of freedom of a wheel have position or velocity sensors.

An autonomous navigation requires three main steps:

- a. The perception of the environment: It consists of detecting road, obstacles and other vehicles. A vision system composed of sensors like cameras, lasers, radars and GPS is usually used to achieve this goal. It provides a dynamic map of the near environment of the autonomous vehicle.
- b. The path planning: It consists of generating and choosing one trajectory (reference path) in the navigable space, according to several criteria.
- c. The vehicle control: It consists of handling the vehicle using actuators like brake, accelerator and steering wheel to follow the reference path. (Talj, 2013)

This thesis aims to focus all three main steps of autonomous navigation.

1.4 Obstacle Avoidance

Trajectory generators for wheeled mobile robots can be interpreted as evolutions of the motion planners of general robotic systems. Though guiding mobile robots is little different than industrial robots for example. Of course the separation into implicit and explicit methods still hold for the methods, but the evolution of these methods are more thought to be more important than the rough separation into these two groups. Therefore an historic overview is chosen to present some of the methods found in literature, more or less relevant to the wheelchair in the thesis (Keij, 2003).

Obstacles block the robot's sensors as well as its motion. The field of motion planning can be split by three guiding philosophies: classical path planning, heuristic planning and "complete and correct" sensor-based path planning. Each of these philosophies will now be discussed along with the advantages and disadvantages of each approach. The first group in this section is explicit, the second and third group are implicit. In the following overview, the path planner generates a trajectory for the robot in the free space which describes the allowable regions for robot traversal: the 2D environment minus the (interiors of) obstacles (Keij, 2003).

1.4.1 Classical Motion Planners

Classical motion planners assume that full knowledge of the geometry of the robot's environment is known a priori. This can be seen as a serious disadvantage if this class, but on the other hand the classical planners have the useful properties of correctness and completeness. A path is correct if it lies wholly within the free-space, and if the goal is reachable, connecting the initial position with the goal. This property is quite common for most known path-planners; otherwise they will be instinctively called incorrect. The second property of completeness is a highly desirable virtue: the planner generates a path if one is possible and halts otherwise in finite time. Latombe describes classical planners in some detail in his book, *Robot Motion Planning* (Latombe, 1991), in which he splits the classical planners into three major categories: roadmap algorithms; cell decomposition methods and potential field approaches. The first two categories seek to create maps or channels for robot navigation (Keij, 2003).

1.4.2 Heuristic Planning

The class of heuristic planners as well as the "Go To Waypoint" algorithm employed by the Sojourner and Rocky planetary Mars rovers share the useful property of being able to be made sensor-based much more easily than the classical planners and can be applied to unknown terrains. These planners dispense with the idea of creating global models of the environment in favour of "using the world as its own model" and using only local knowledge of the environments to inform the robot's reactions, usually chosen from a set of "behaviours". A very good example of heuristic planners is the bug algorithm. This algorithm is based on the behaviour of cockroaches.

The robot bases its trajectory on pre-programmed situations and criteria and executes pre-programmed commands. Although heuristic planners are designed to work well in most environment configurations, they lack completeness. There is no guarantee that the algorithm will halt, or that the robot will be able to find the goal even if a path exists. Some research projects report quite lengthy paths using this class of planners, which will be a big disadvantage in most cases (Keij, 2003).

1.4.3 "Complete and Correct " Sensor-Based Path Planning

The class of "Complete and correct" sensor-based path planning can be seen as an evolution of both previous described classes, the complete and correct classical planners and the sensor based heuristic planners. This class is mainly incremental in nature: the robot senses its environment, and then determines a local path segment based upon the resultant world model (Keij, 2003).

After moving along the local path, the robot begins the cycle again with its sensors. Using this model, three distinct approaches have been explored, two of which adapt classical methods to a local sensed region (Keij, 2003).

One set of methods incrementally builds "roadmaps" within the free space in the visible area. A successful example of this method is the Tangent Bug algorithm, developed by Kamon, Rivlin and Rimon. The second approach is based on approximate cell decomposition, filling in a grid-based world model incrementally, such as Stentz' D* algorithm. The third approach springs from the heuristic planners and includes the "Bug" algorithms of Lumelsky and Stepanov and Rao et al., which combine reactive behaviours with global parameters to reach the goal (Keij, 2003).

All of these methods maintain provable properties of completeness, yet are fully applicable to unknown terrains. Both heuristic planners and sensor-based planners of this kind share the disadvantage that their computational complexity is difficult to analyse, primarily due to the algorithms' reliance upon sensor input for decision-making. For this same reason, both types of planners are subject to sensor error, and it is uncertain how such errors affect the performance of many of the methods. In particular, several schemes rely upon "good" (or "perfect") dead-reckoning ability (Keij, 2003).

1.5 Wheeled Mobile Robots (WMR) Research Projects

Some examples of robots and research projects dealing with wheeled mobile robotic systems and obstacle avoidance in particular are presented in the following. As one can imagine this overview is far from complete, it is meant to illustrate different ways of dealing with the obstacle avoidance problem.

1.5.1 Automated Highway System (AHS)

The automated highway system, AHS belongs in the group of practical situations. In this group of projects several companies are trying to bring a high level of automation in automobiles highways. The current target is set on a train-like motion of several different cars, using the highway at a very short range but constantly aware of possible dangers, e.g. if one car stops for some reason or some other unpredicted incident happens: the individual car should react automatically avoiding any collisions, not compromising the safety of the driver/passenger (Keij, 2003).

Technically this is analog to a lane following problem including an obstacle avoidance function for a non-holonomous wheeled mobile robotic system. An extra difficulty in this context is the safety concerns that are inevitably in traffic situations. Highways present an unknown and dynamic environment with real-time constraints. In addition, the high speeds of travel force a system to detect objects at long ranges. Although there are a number of methods that can successfully detect moving vehicles, the more difficult problem of finding small, static road debris such as tires or crates remains mainly unsolved. It is very difficult to present more details on obstacle avoidance algorithms and sensor configurations for these projects, because a lot of different companies are tackling this problem, keeping their individual results as secret as possible (Keij, 2003).

1.5.2 *RHINO*

In the Deutsches National Museum in Bonn, visitors can take a guided tour accompanied by a mobile robot. This robot is not only capable of interacting with its guests and giving an audio-visual presentation it has also the preferable property of not colliding into artworks and people in the museum. This wheeled mobile robot called RHINO is designed at the University of Bonn in the group of Burgard making use of probabilistic roadmaps (PRM) and Artificial Intelligence techniques. Its sensor configuration is pretty extended but the results are at least promising (Keij, 2003).

1.5.3 *SuperMARIO*

All previous described projects are almost evolved out of the area of science into more or less commercial available applications. SuperMARIO is a research robot meant for testing and improving basic control tasks for wheeled mobile robots (WMR). The project is based at the University of Rome, Italy in the department of Informatics and Systems and has delivered several insights in the basics of mobile robot control, like feedback and feedforward control. (De Luca, 2002)

1.5.4 *Artificial Neural Networks*

Artificial neural network modelling has become quite common in the area of obstacle avoidance techniques. There are many research projects all over the world, like the ALVINN and ROBIN robots at the Carnegie Mellon University. Lagoudakis uses a Hopfield Neural Network for dynamic path planning and obstacle avoidance, or Neural maps for mobile robot navigation both applied on the NOMAD robot. The Boston University's Neurobot Lab uses a neural network for adaptive obstacle avoidance to apply on their Khepera robot. One should note that neural networks are mainly a way to model a problem in a different way. In most of the cases it is possible to describe, or better reformulate the neural network modelling into another existing technique (Keij, 2003).

1.5.5 Roadmap Robots

As stated in previous section Virtual Potential Field methods are widely applied. Probabilistic roadmap methods are widely developed but not that much applied on 'real' robots. This is mainly because of the relatively large complexity of these methods. There are a few projects worth mentioning though. J.-P. Laumond gives a very extensive overview in his book Robot Motion Planning and Control. Probabilistic Roadmap techniques are implemented mainly on simulation level by Kavraki, Svestka, Overmars and Latombe (Keij, 2003).

1.5.6 Virtual Potential Field Robots – CARMEL

The class of potential field techniques is being applied on a lot of different robots. Borenstein and Koren have developed and applied this method on their robot CARMEL. The robot was not only used to test VPF-like methods. Borenstein also applied a histogrammic in-motion mapping (HIMM), a vector field histogram (VFH) method and a model-reference adaptive motion controller (MRAC) (Keij, 2003).

1.5.7 Other Virtual Potential Field Implementations

After the encouraging results on both scientific simulations and implementations, the class of virtual potential field methods have been adapted widely, not only in wheeled mobile robotic systems. A few important implementations lay in the field of aids for the handicapped. The nursing robot is a robot which can help in simple tasks. This kind of robot is also available in office-like environments as a substitute for the internal mailman. The Blind guide robot is the robotic equivalent for a guiding dog for blind people. The NavChair is specific designed for visual handicapped people. The advanced wheelchair implementation is pretty related to the BellyBot project, where the user indicates a desired position where the path planner should warn or change the trajectory to avoid any obstacles on its way where necessary (Keij, 2003).

1.6 Wheelchairs Types for Disabled People Who are Unable to Use Their Hands and Feet

Voice-controlled electric wheelchairs have a control system that decodes the manoeuvring commands by speech-recognition techniques and transmits these commands to the wheelchair to effect the desired motion (Nolan et. al., 1998; Kurtzberg et. al., 2000). The commands are entered by a throat-engaging microphone, and backup commands are also recognized, including a command based on an excited utterance to stop the wheelchair. The control system is switchable by voice command between a first condition in which it executes other commands and a second condition in which it does not execute other commands (Nolan et. al., 1998).

Bite/blowing-controlled electric wheelchairs are based on Bio-signals as surface electromyogram signals. The Bioelectric signals are picked up from facial muscles then the Bio-signals are passed through an amplifier and a high pass filter. Motion control commands (Forward, Left, Right, Forward to the Right, Forward to the left and Stop) are classified by simple rule. These commands are used for controlling the electric wheelchair (Tamura et. al., 2010).

Typical EOG/EEG controlled electric wheelchairs based on a brain-machine interface and a method for processing the EOG/EEG signals are the same. The wheelchair comprises a preposed amplified signal pre-processing circuit which collects the EOG/EEG signal of a testee and essential electronic control circuits (Dong et. al., 2008). Some patented examples of brain wave controlled wheelchairs are presented in the following:

1.6.1 Wheelchair Controller Based On Brain Wave

The utility model discloses a wheelchair controller based on a brain wave, characterized in that the P1.5 of a P1 port, the P2.0 to P2.4 of a P2 port and the P5.0 to P5.7 of a P5 port of a one-chip microcomputer MSP430F169 are connected with a USB driving circuit separately, the P6.6 to P6.7 of a P6 port and a GND port are

connected with an X-Y end and a GND end of a wheelchair controller main chip respectively, and the two USB interfaces of a PC are connected with the USB interfaces of a brain wave driver and a microcontroller one-chip microcomputer MSP430F169 respectively. The wheelchair controller based on the brain wave of the utility model utilizes a head band to detect the brain wave movement and converts the brain wave movement into a concrete keyboard button or mouse action, various signals are processed into operation instructions via an intelligent central controller, and finally, a microcontroller controls the wheelchair motion. The wheelchair controller based on the brain wave is simple and convenient in operation and low in cost, is convenient to arrange on the conventional electric wheelchair, and is suitable for the limbs paraplegia patient to use. (Cheng et. al., 2012)

1.6.2 Navigation System Using Brain Wave Signal and Navigation Control Method

To obtain a navigation system using a brain wave signal in which an image of a guide object to be navigated by using the brain wave signal can be displayed freely and to obtain a navigation control method.

In a brain wave processing inside a data processing, a pattern of the brain wave signal corresponding to a control signal used to display the image of the guide object is stored in advance. Whether the pattern of the brain wave signal requesting the image of the guide object detected in a brain wave detection corresponds to the stored control signal is discriminated. When it corresponds, the image of the guide object corresponding to the detected pattern of the brain wave signal is displayed on a display part (Tatsumi, 2002).

1.6.3 Brainwave Responsive Wheelchair

A brainwave responsive wheelchair for providing control of an electric wheelchair by a physically handicapped person, includes a wheelchair having a seat portion, a back portion, a pair of back wheels and a pair of front wheels. A motor for selectively

rotating the back wheels independent of each other is mechanically coupled to the back wheels. An actuator for selectively rotating the back wheels is operationally coupled to the motor. The actuator is mounted to the back portion and includes a control that is electrically coupled to the motor. A brainwave responsive device is operationally coupled to the control. The brainwave responsive device includes an input device. The input device is electrically coupled to the control and adapted for reading brainwaves. The control actuates the motor with respect to distinct brainwave patterns (Bertha, 2003)

1.7 Latest Developments (2013-2017)

1.7.1 Design and Implementation of a Multi Sensor Based Brain Computer Interface for a Robotic Wheelchair (2017)

In this study, design and implementation of a multi sensor based brain computer interface for disabled and/or elderly people is proposed. Developed system consists of a wheelchair, a high-power motor controller card, a Kinect camera, electromyogram (EMG) and electroencephalogram (EEG) sensors and a computer. The Kinect sensor is installed on the system to provide safe navigation for the system. Depth frames, captured by the Kinect's infra-red (IR) camera, are processed with a custom image processing algorithm in order to detect obstacles around the wheelchair. A Consumer grade EMG device (Thalmic Labs) was used to obtain eight channels of EMG data. Four different hand movements: Fist, release, waving hand left and right are used for EMG based control of the robotic wheelchair. EMG data is first classified using artificial neural network (ANN), support vector machines and random forest schemes. The class is then decided by a rule-based scheme constructed on the individual outputs of the three classifiers. EEG based control is adopted as an alternative controller for the developed robotic wheelchair. A wireless 14-channels EEG sensor (Emotiv Epoch) is used to acquire real time EEG data. Three different cognitive tasks: Relaxing, math problem solving, text reading are defined for the EEG based control of the system. Subjects were asked to accomplish the relative cognitive task in order to control the wheelchair. During experiments, all subjects were able to control the robotic

wheelchair by hand movements and track a pre-determined route with a reasonable accuracy. The results for the EEG based control of the robotic wheelchair are promising though vary depending on user experience (Kucukyildiz et. al., 2017).

1.7.2 Modular and Adaptive Wheelchair Automation (2016)

The paper presents a novel framework for the design of a modular and adaptive partial-automation wheelchair. Our design in particular aims to address hurdles to the adoption of partial-automation wheelchairs within general society. In this experimental work, a single assistance module (assisted doorway traversal) is evaluated, with arbitration between multiple goals (from multiple detected doors) and multiple control signals (from an autonomous path planner, and the human user). The experimental work provides the foundation and proof-of-concept for the technical components of our proposed modular and adaptive wheelchair robot. The system is evaluated within multiple environmental scenarios and shows good performance (Argall, 2016).

1.7.3 EEG Based Brain Controlled Wheelchair for Physically Challenged People (2016)

Independent mobility is a necessity to live everyday life for human beings. A person with physical challenges has restricted mobility. For these people, Brain Computer Interface (BCI) provides a promising solution. Using Electroencephalogram (EEG) signals for movement of wheelchair the mobility of these persons can be improved. The proposed system is based on Artificial Neural Network (ANN) algorithm. This article presents Mu rhythm signals that provide commands to wheelchair. Using wireless link between head gear and computer, commands to control the wheelchair can be issued (Folane et. al., 2016).

1.7.4 Brainwave Controlled Robot (2015)

In the world number of people are handicapped. Currently they use different technologies which give physically impaired the ability to move around. But still there are numbers of people who are fully handicapped and paralyzed but their mind still work properly. So my task for those people who are physically handicapped and little disturb from mind site is to help them so that they can move around world using their mind power. For that I try to design one robot or wheelchair which is fully automated and control using Beta wave (human brain attention) using Mind wave sensor which detect brain signal and also use Arduino to control robot or wheelchair (Solanki et. al., 2015).

1.7.5 A Concept of Smart Wheelchair (2015)

The paper presents the design concept of a smart wheelchair for people with special needs. The first part of the paper explains reasons for increasing attention in the world for this topic not only in a context of technical soundness, but enhancement of human-life conditions, too. In the second part of the paper, features of nowadays smart wheelchairs and problems of localization and mapping, which are the main problems in robotics in general, are described. The third part presents the details of the novel design of the smart wheelchair, its CAD model and sensor equipment that would be implemented. The idea is to make an advanced wheelchair which will give comfort to a user within the acceptable price range. Further work includes the development of the control system and wiring, the implementation of the obstacle avoidance algorithms, as well as project methodology for efficient customer-oriented wheelchair design and production (Belic et. al., 2015).

1.7.6 Voice Controlled Autonomous Wheelchair (2013)

In this work, in order to support physically handicapped persons a low cost voice controlled wheelchair is proposed. The person can control the wheelchair by voice commands. This paper presents the proposal, design and implementation of a

microcontroller based voice controlled wheelchair. As speech is the preferred mode of operation for human being, this work intends to make the voice oriented command words for controlling wheelchair. Voice recognition module has to be added to the wireless network. The voice command is a person independent. The system comprises of transmitting and receiving module. Initially, the voice command is stored in the data base with the help of the function keys. Then the input voice commands are transmitted through wireless. The voice received is processed in the voice recognition system where the feature of the voice command is extracted and matched with the existing sample in the database. The module recognizes the voice and sends control messages to the microcontroller. The programmed microcontroller then processes the received data and switches the respective direction and motion of motors via connected through driver circuits. In this work advanced operation like how much angle user wants to rotate its wheelchair has to be implemented and this methodology is different from other prototype that have made been made or discussed earlier. This work also implements obstacle avoider circuit that will avoid obstacle accident and it will immediately move away from the obstacle so this will smart wheelchair for the user (Tiwari et. al., 2013).

1.8 Significance of the Thesis

There is a lot of study in the world some of mentioned in section 1.5 for the people who are unable to use their hands and feet. These wheelchairs can be controlled by sounds, bites, blows or EEG signals but does not have a smart autopilot (autonomous control) system with global path planning algorithm. The user have to give commands in every step of the movement, therefore it becomes an unpractical solution because it is not very easy to give commands with EEG signals or bites etc.

For these reasons it is important to lower commands given by the user with more autonomy. Accordingly, a smart wheelchair that have an autopilot system has been developed in the thesis.

Besides this the smart wheelchair also consists a special keyboard which can be controlled by EEG signals or with just eye blinks without using hands to make computer easier to use and to make writing faster.

Consequently, innovative and practical solution is developed and presented for the disabled people who do not have a sufficient solution for transportation. Also, it will help possible studies in the future as a basis.

1.9 Outline of the Thesis

In the following, an outline is given to guide the reader through the thesis chapters. Subject of the chapter and the main points covered are given briefly:

Chapter 1:

This chapter has focused on the current situation and the motivation behind this work has been briefly discussed. Moreover the main objectives have been defined with the boundary conditions involved. Theoretical background of the thesis is briefly described.

Chapter 2:

In this chapter, general algorithm of the smart wheelchair and components used in the system are given in detail. Inertial Navigation System (INS) and Laser Imaging Detection and Ranging (LIDAR) are described in detail and performed studies are explained with examples.

Chapter 3:

In this chapter, mechanical design, mechanical parts and kinematic analysis of the wheelchair are given in detail.

Chapter 4:

In this chapter, designed and produced motor drivers, electronic control system and other electronic components are given in detail.

Chapter 5:

In this chapter, graphical user interface of the computer program and maps used for navigation are given in detail.

Chapter 6:

In this chapter, EOG analysis to assign destination point are performed.

Chapter 7:

In this chapter, final state of the developed navigation algorithm is described in detail and performed last successful implementations are presented.

Chapter 8:

In this chapter, conclusions and significant results obtained from the thesis are given with planned future works.

CHAPTER TWO

GENERAL OPERATION ALGORITHM AND COMPONENTS OF THE WHEELCHAIR

2.1 General Operation Algorithm of the Wheelchair

Integrated approaches are needed for more practical and extensive usage of the wheelchairs which are developed for the people who are unable to use their hands and feet. Integrated approaches can be defined as combination of mapping, localization and motion control as seen in Figure 2.1.

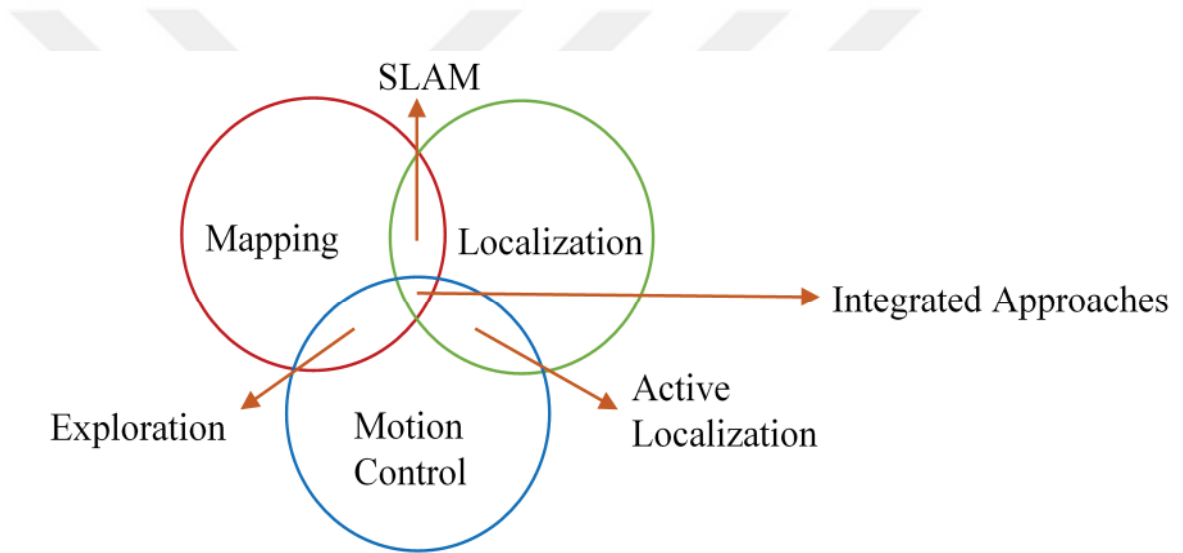


Figure 2.1 Integrated approaches (Stachniss et. al., 2005)

Therefore, a practical smart wheelchair solution which can be used by eye blinks and/or head movements for transportation needs have been developed, and presented in this thesis.

EOG signals which occur as a result of facial expressions and thoughts are measured by the sensors placed on head. Similarity of these signals are compared in real-time with pre-recorded search templates (sample signals) with the user defined threshold values for each search template. As a result of these analysis and gyro sensor data, mouse cursor is moved to desired destination point on GUI, and clicked to assign the destination point.

After assignation of destination points, the user doesn't have to give any more commands and smart navigation and control system is activated automatically. Real time mapping, localisation, adaptive shortest path finding, autonomous navigation and controlling operations are performed to achieve destination point safely and properly even if in previously unknown environments without any landmark. Shortest path and motion control parameters are updated real-time due to encountered static and dynamic obstacles and changing environmental conditions. General schematic and conceptual prototype is shown in Figure 2.2.

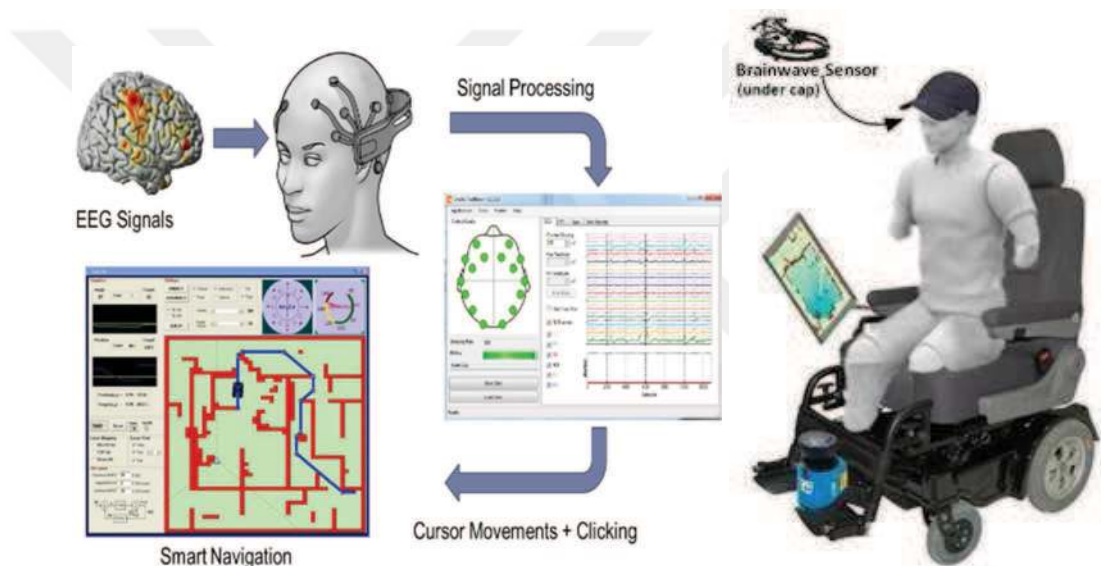


Figure 2.2 General schematic and conceptual prototype

With smart navigation software, the vehicle follows destinations autonomously by using optimal paths. While following the paths, the program updates map and optimal paths in the vehicle's memory as per static and dynamic obstacles for adaptive optimal path following. Map can be uploaded before or can be mapped after destination point is assigned.

Designed electronic communication scheme and its components are shown in Figure 2.3. Direction of the vehicle (yaw angle) is measured by IMU externally. Displacement of the vehicle is measured by encoders with assuming no slipping.

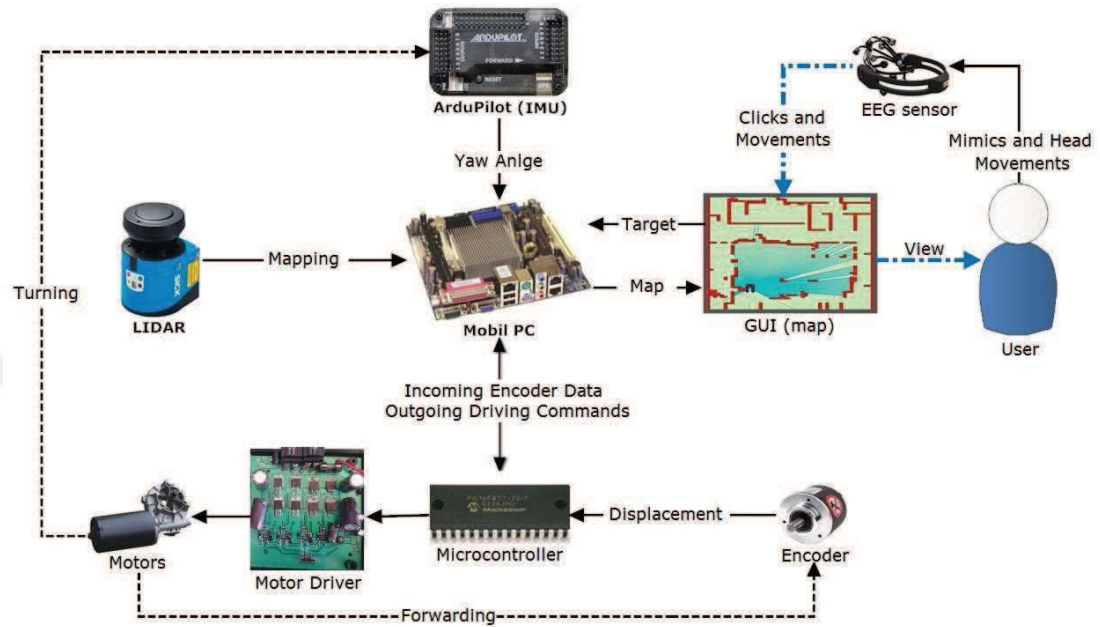


Figure 2.3 Electronic communication scheme

The thesis consists of five different software programs as given in the following:

1. IMU measurement microchip software which is mentioned in Chapter 2.
2. Main control board microchip software which is mentioned in Chapter 4.
3. Graphical User Interface (GUI) software which is mentioned in Chapter 5.
4. ROS software in Linux Ubuntu OS which is mentioned in Chapter 5.
5. Special screen keyboard software developed to help the user while using computer/internet by EOG signals is mentioned in Chapter 6.

The hardware components used in the development of the autonomous wheelchair are described in the following sections.

2.2 Inertial Navigation System (INS)

Inertial navigation systems are self-contained, non-radiating, non-jammable, dead-reckoning navigation systems which provide dynamic information through direct measurements. In most cases an INS must be integrated with other absolute location-sensing mechanisms to provide useful information about the vehicle position. Fundamentally, gyroscopes provide angular rate information, and accelerometers provide velocity rate information. Although the rate information is reliable over long periods of time, it must be integrated to provide absolute measurement of orientation, position and velocity (Barshan, 1995).

2.2.1 Accelerometer Sensors

Accelerometer sensors measure the acceleration experienced by the sensor and anything to which the sensor is directly attached. Accelerometer sensors have many applications. The most common commercial application is impact sensors for triggering airbag deployment in automobiles: when the acceleration exceeds 30 to 50 g's, an accident is assumed and the airbags deploy. Such sensors are designed to be rugged and reliable, and are made in high volume and at low cost by several chip manufacturers. Airbag sensors don't need to be very accurate: with a threshold of 50 g's, an accuracy of 1 to 2 g is acceptable. High precision accelerometer sensors have a variety of applications.

They are used with gyroscopes in inertial guidance mechanisms: the displacement is calculated by twice integrating the acceleration signal, and the gyroscopes indicate the direction of displacement. Such components are used to make small inertial guidance units in rockets and aircraft, which complement direct navigation using satellite global positioning.

When working with accelerometers in the earth's gravitational field, there is always the acceleration due to gravity. Thus the signal from an accelerometer sensor can be separated into two signals: the acceleration from gravity, and external acceleration.

The acceleration from gravity allows measurement of the tilt of the sensor by identifying which direction is “down”. By filtering out the external acceleration, the orientation of a three-axis sensor can be calculated from the accelerations on the three accelerometer axes. Orientation sensing can be very useful in navigation.

Accelerometer sensors can also be used to indirectly infer the status of a machine. One proposed application of accelerometer sensors is detecting when a washing machine goes out of balance. The range of acceleration is a few g’s, and the precision required is mg’s, with a bandwidth up to the frequency of rotation. By fixing a two-axis accelerometer (the axes perpendicular to the axis of rotation), an out-of-balance load is detected by excessive vibration. A more sophisticated analysis could determine in what way the rotor is off-balance and compensate appropriately. This application will be increasingly important as washing machine rotational speeds increase during the spin cycle to shed more water and reduce drying time, thereby decreasing the overall power consumption for washing clothes. One g is the acceleration due to gravity, 9.8 m/s^2 .

The goal of using an accelerometer is to measure the three-dimensional acceleration of the vehicle motion with adequate accuracy and precision, the necessary bandwidth for the vehicle motion, and the amplitude range required for the highest normal accelerations.

2.2.2 Inertial Measurement Unit (IMU)

An inertial measurement unit, or IMU, is an electronic device that measures and reports on a craft's velocity, orientation, and gravitational forces, using a combination of accelerometers and gyroscopes, sometimes also magnetometers. IMUs are typically used to maneuver aircraft, including unmanned aerial vehicles (UAVs), among many others, and spacecraft, including shuttles, satellites and landers. Recent developments allow for the production of IMU enabled GPS devices. An IMU allows a GPS to work when GPS-signals are unavailable, such as in tunnels, inside buildings, or when electronic interference is present. A wireless IMU is known as a WIMU.

The IMU is the main component of inertial navigation systems used in aircraft, spacecraft, watercraft, and guided missiles among others. In this capacity, the data collected from the IMU's sensors allows a computer to track a craft's position, using a method known as dead reckoning.

2.2.3 IMU Data Collection and Validation

In the first prototype, 9 DOF IMU is used to measure Euler angles (roll, pitch, yaw), angular velocity and acceleration data. The data format is as follows: “r: 4 p: 27 y:359 g=(0.0 0.0 -0.0) a=(9.0 -3.0 -16.9)“. Testing program interface written in Visual Basic can be seen in Figure 2.4. These values were planned to use for calculation of velocity and displacement at the beginning of the thesis. But, cumulative accelerometer errors caused big integral errors in displacement. Then, only the yaw angle is measured with IMU instead of using a compass sensor for the first prototype.

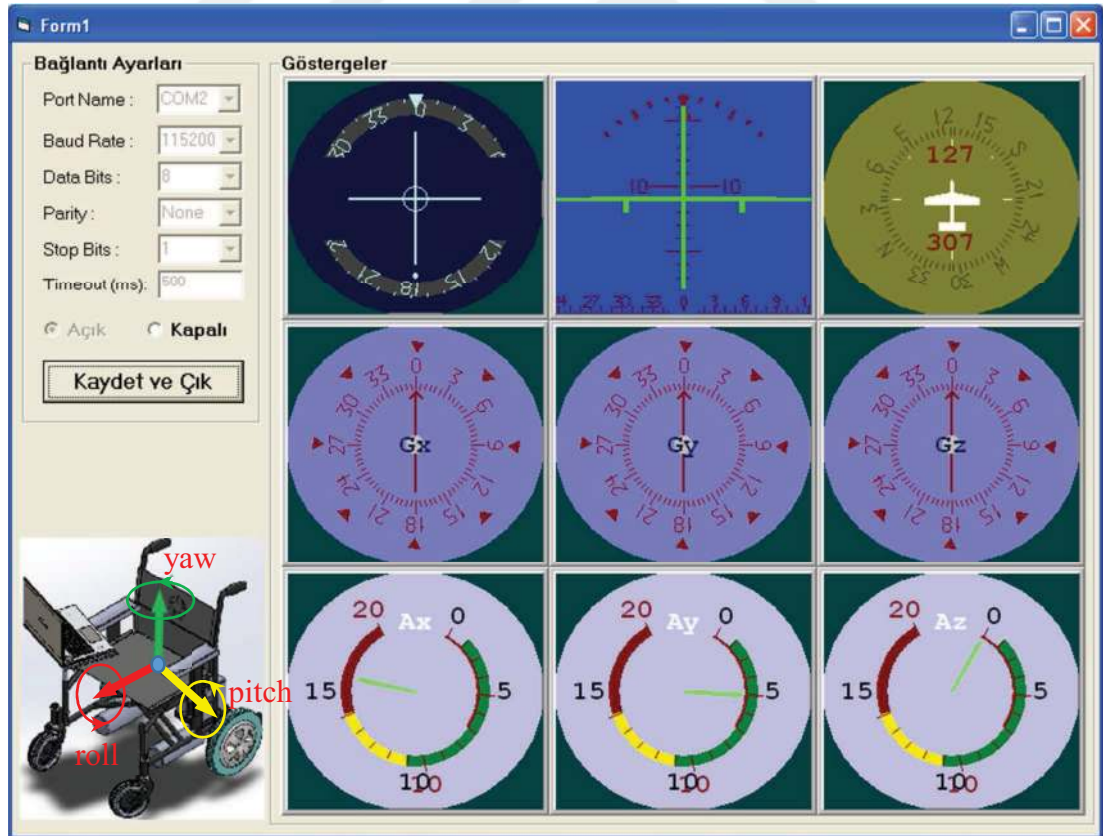


Figure 2.4 Euler angles, angular velocities and acceleration data

New IMU software with 10 DOF IMU (MPU 6050) instead of ArduPilot Board (Figure 2.5) has been developed and tested successfully being separated from the main control board. New IMU is used to accelerate the direction measurement operations.

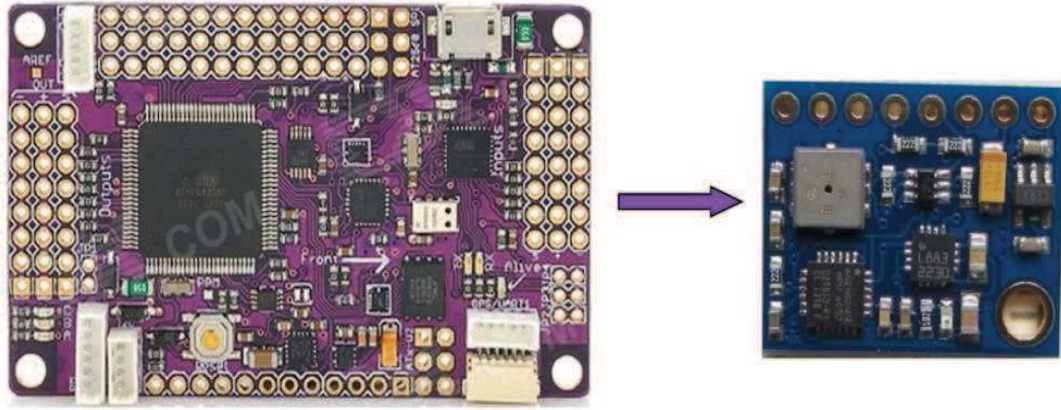


Figure 2.5 Free IMU (on the right) is used instead of ArduPilot board (on the left)

Microprocessor code of the IMU is updated and first setup errors have been eliminated and faster connection is established. Also the interface program has been updated and some zero values have been filtered. After that integration of the IMU to the main control microprocessor has been completed.

After some researches, it was noticed that IMU must consist of a magnetometer for precise yaw angle measurement. After some studies achieved to send yaw angles directly to the main control board by using ArduPilot board with IMU. Previously IMU was connected to the computer, after some software updates IMU connected directly to the main control board. Thus, the destination yaw angle control made more quickly with shortened feedback time.

ArduPilot codes are updated, and developed for continuous YAW angle measurement and is shown partially in Figure 2.6.



Figure 2.6 IMU software

2.3 Laser Imaging Detection and Ranging (LIDAR)

A laser measurement sensor (Figure 2.7) is a device which uses a laser beam to determine the distance to an object. The most common form of laser measurement sensor operates on the time of flight principle by sending a laser pulse in a narrow beam towards the object. Then it measures the time taken by the pulse to be reflected off the target and returned to the sender. Due to the high speed of light, this technique is not appropriate for high precision sub-millimetre measurements, where triangulation and other techniques are often used.



Figure 2.7 SICK LMS100 laser measurement sensor (SICK AG, 2008)

2.3.1 Operating Principle of the LMS

The LMS is an electro-optical laser measurement system that electro-sensitively scans the perimeter of its surroundings in a plane with the aid of laser beams (Figure 2.8). The LMS measures its surroundings in two-dimensional polar coordinates. If a laser beam is incident on an object, the position is determined in the form of distance and direction (SICK AG, 2008).

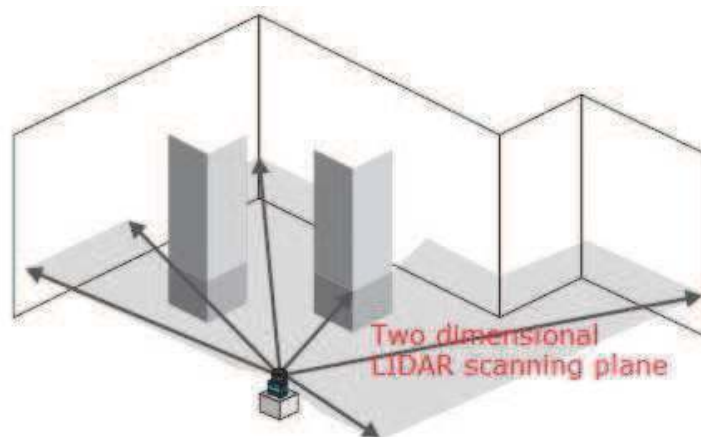


Figure 2.8 Measuring principle of the LMS (SICK AG, 2008)

Scanning takes place in a sector of 270° . The scanning range of the LMS is maximum 20 m on light, natural surfaces with an object remission $> 13\%$ (e.g. a white house wall).

2.3.2 Distance Measurement

The LMS emits pulsed laser beams using a laser diode. If such a laser pulse is incident on an object, it is reflected at its surface. The reflection is detected in the laser measurement system's receiver using a photodiode (Figure 2.9). The distance to the object is calculated from the propagation time that the light requires from emission to reception of the reflection at the sensor. This principle of "pulse propagation time measurement" is used by radar systems in a similar manner (SICK AG, 2008).

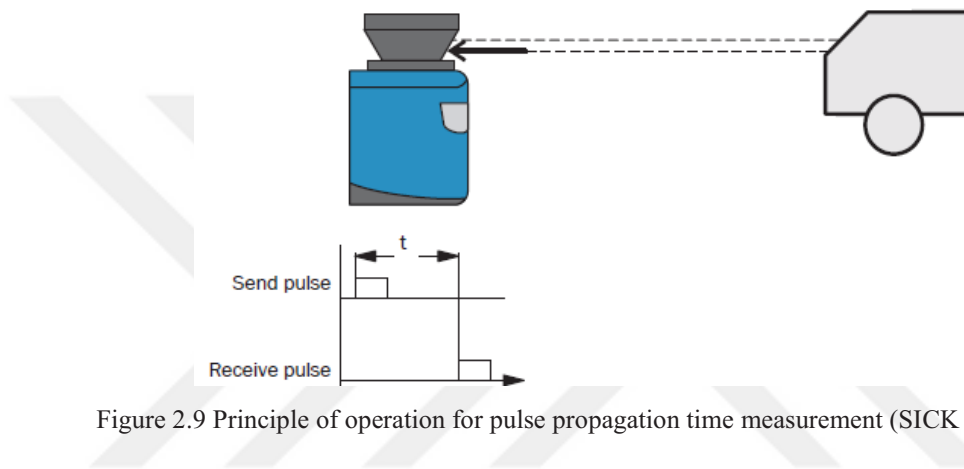


Figure 2.9 Principle of operation for pulse propagation time measurement (SICK AG, 2008)

2.3.3 Direction Measurement

The emitted laser beams are deflected using a mirror and scan the surroundings in a circular manner. The measurements are triggered at regular angular steps using an angular encoder. The LMS scans with a scanning frequency of 25 or 50 Hz. During this process, a laser pulse and therefore a measurement is triggered after an angular step of 0.25° or 0.50° for LMS100 model (SICK AG, 2008).

2.3.4 Influences of Object Surfaces on the Measurement

The signal received from a perfectly diffuse reflecting white surface corresponds to the definition of a remission of 100%. As a result of this definition, the remissions for surfaces that reflect the light bundled (mirrored surfaces, reflectors), are more than 100%. The reflection of the laser beam will vary as a function of the surface structure and colour (SICK AG, 2008).

Light surfaces reflect the laser beam better than dark surfaces and can be detected by the LMS over larger distances. Brilliant white plaster reflects approximately 100% of the incident light, black foam rubber approximately 2.4%. On very rough surfaces, part of the energy is lost due to shading (Figure 2.10). The scanning range of the LMS will be reduced as a result (SICK AG, 2008).

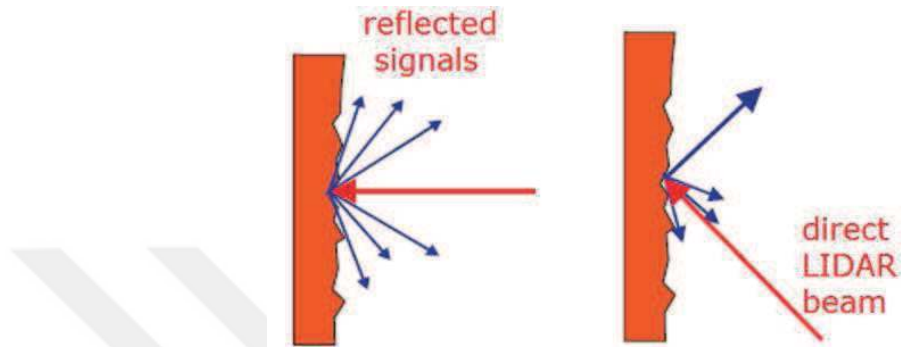


Figure 2.10 Reflection of the laser beam at the surface of an object (SICK AG, 2008)

2.3.5 Scanning Range of the LMS

The scanning range of the LMS is dependent on the remission of the objects to be detected. The better a surface reflects the incident radiation, the greater the scanning range of the LMS. The diagram given in Figure 2.11 indicates the relationship between remission and detectability (SICK AG, 2008). Obstacles detected over the distance 10 meter is neglected. Existing obstacles under the neglected laser beams are deleted for mapping.

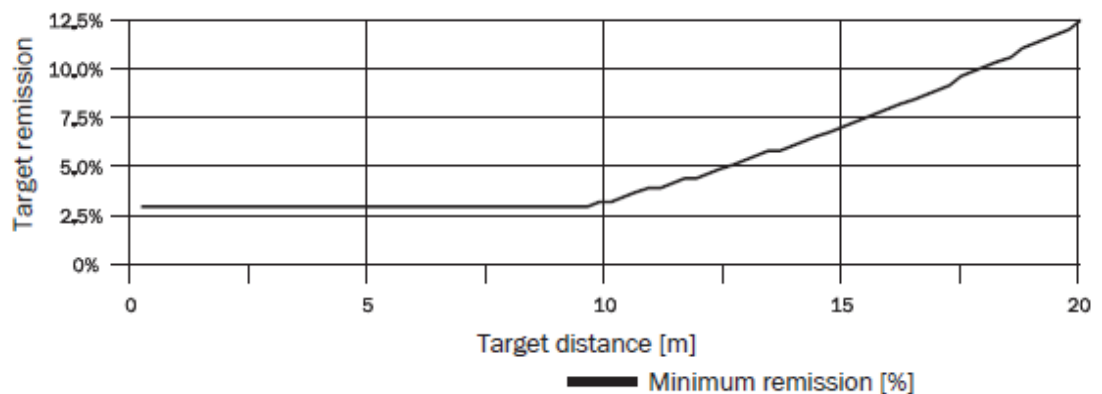


Figure 2.11 Scanning range of the LMS100 as a function of the target remission (SICK AG, 2008)

2.3.6 Data Interfaces

The LMS has different data interfaces for configuration and transmission of measured values.

- It is only possible to output all measured values of a scan in real-time using the Ethernet interface.
- The data transmission rate of the RS-232 interfaces is limited. Therefore these interfaces are not suitable for transmitting scan data in real time.

The Ethernet interface has a data transmission rate of 10/100 Mbit/s. The interface is a TCP/IP interface. Full duplex and half duplex are supported. The Ethernet interface allows the configuration of the LMS as well as the output of measured values. The factory setting for the Ethernet interface is as follows:

- IP address: 192.168.0.1
- Subnet mask: 255.255.255.0
- TCP port: 2111

2.3.7 Data Communication Using Messages

The LMS sends messages over the interfaces to communicate with a connected host. The following functions can be run using messages:

- Request for measured values by the host and subsequent output of the measured values by the LMS
- Parameter setting by the host for the configuration of the LMS
- Parameters and status log querying by the host

Measured value message can be requested using the “sRN LMDscandata” command for each scan (SICK AG, 2008)

2.3.8 Laser Radiation Safety

The LMS corresponds to laser class 1 (eye safe) as per EN 60825-1. The laser beam cannot be seen with the human eye. The laser operates at a wavelength $\lambda = 905$ nm (invisible infrared light). The radiation emitted in normal operation is not harmful to the eyes and human skin. The laser warning is on the LMS on the right side of the housing (Figure 2.12).

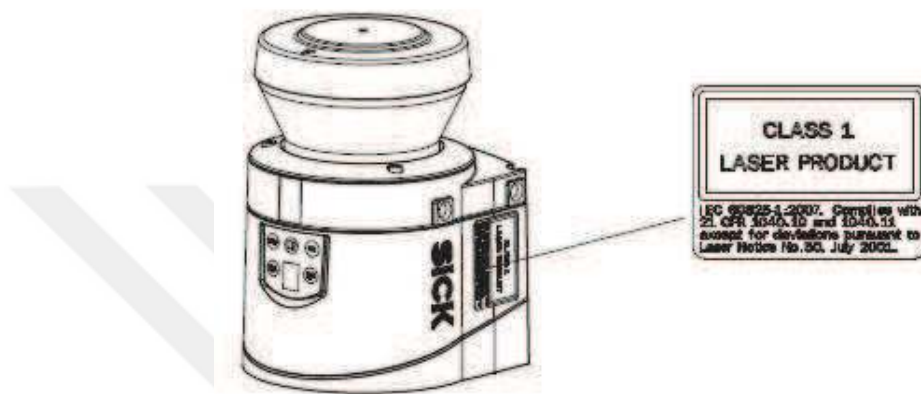


Figure 2.12 Laser warning label on the LMS (SICK AG, 2008)

2.3.9 Mapping with LMS via Ethernet and Serial Port

The LMS has different data interfaces for the configuration and the transmission of measured values.

- It is only possible to output all measured values of a scan in real-time using the Ethernet interface.
- The data transmission rate of the RS-232 interfaces is limited. Therefore these interfaces are not suitable for transmitting scan data in real time.

The Ethernet interface allows configuration of the LMS as well as the output of measured values (SICK AG, 2008)

2.4 Encoders

A rotary encoder, also called a shaft encoder, is an electro-mechanical device that converts the angular position or motion of a shaft or axle to an analog or digital code. There are two main types: absolute and incremental

(relative). The output of absolute encoders indicates the current position of the shaft, making them angle transducers. The output of incremental encoders provides information about the motion of the shaft, which is typically further processed elsewhere into information such as speed, distance and position (Rotary encoder, n.d.).

On the first prototype Atek ARS HS 50 256 HPL 3Y-C incremental hollow shaft optical encoders (Figure 2.13 and Figure 2.14) are used which give 256 pulses per revolution (PPR) and works with 5-24 VDC. Detailed displacement calculation for each pulse of encoder is given in chapter 4 with equation (3.17).



Figure 2.13 Atek ARS HS 50 256 HPL 3Y-C encoder



Figure 2.14 Encoder directly connected to motors for the first prototype (Personal archive, 2014)

On the second prototype SICK DBS36E-BBEK00360 incremental blind hollow shaft encoders are used which gives 360 pulses for each revolution and

works with 7-30 VDC. Encoders are directly connected to the motors as shown in Figure 2.15.



Figure 2.15 SICK DBS36E-BBEK00360 incremental blind hollow shaft encoder

2.5 GPS Measurements with Kalman Filter

Besides indoor navigation GPS data by using Kalman Filter in MATLAB are obtained, and analysed for outdoor navigation of the wheelchair. Some experiments are made with 3DR UBlox GPS + Compass Module with ArduPilot 2 board shown in Figure 2.16.

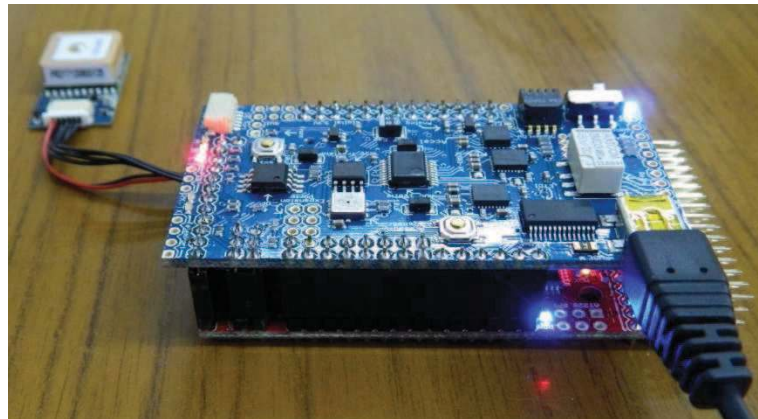


Figure 2.16 ArduPilot Mega 2.5 with GPS sensor (Personal archive, 2015)

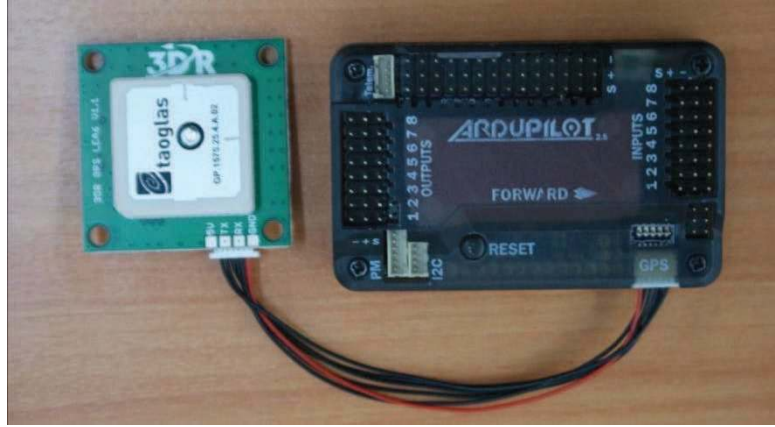


Figure 2.17 Continues (Personal archive, 2015)

2.5.1 GPS Measurement Example 1:

First GPS measurements are carried out at Ege University Campus by 180 meter walking which starts from right (east) to left (west) and then below (south) as shown in Figure 2.17.



Figure 2.18 GPS measurements (blue) on map

Before starting walking, some measurements are taken on fixed position as shown in Figure 2.17 and Figure 2.18. They are filtered with KALMAN Filter in MATLAB. Obtained results are shown with red dots in the figure. Measurements errors were nearly 5-6 meters, and reduced nearly to 1 meter with filtering.

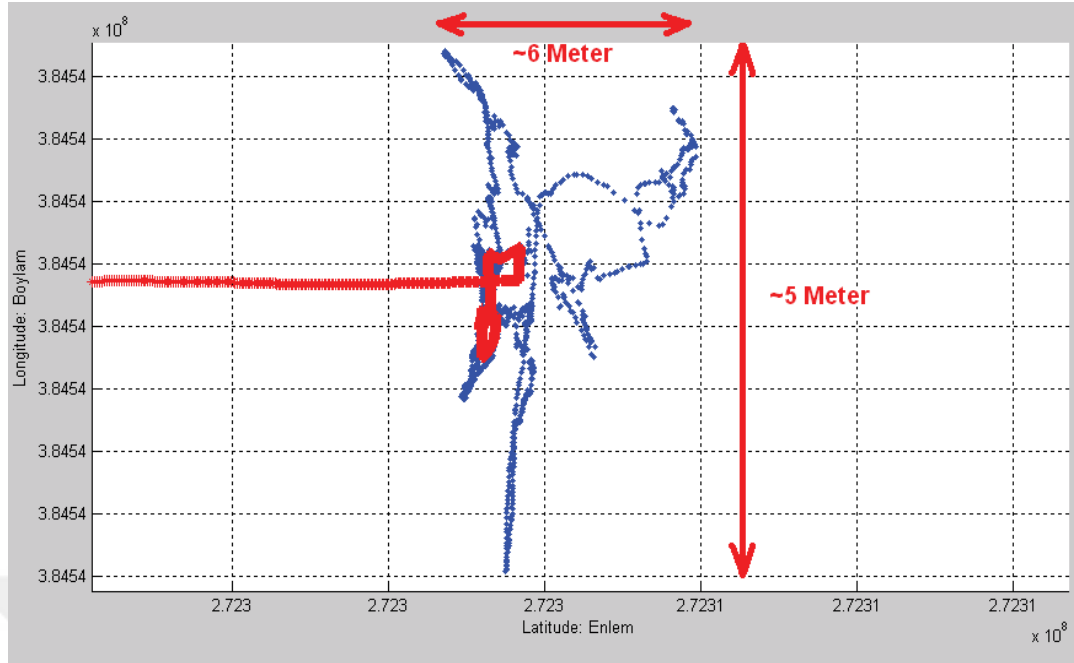


Figure 2.19 Raw data (blue) and filtered results (red) while standing at a point.

1 degree latitude equals to 111000 meters, and 1 degree of longitude equals to $111000 \cdot \cos(\text{latitude: you are at})$. For verification of the results, filtered GPS measurements visualised in MATLAB figure are super positioned with Google Map view as shown in Figure 2.19. The distance between point A and B is measured as 180 meter by Google Map, and it approves MATLAB figure.



Figure 2.20 Combination of Google Map view with MATLAB figure

2.5.2 GPS Measurement Example 2:

Second GPS measurements are carried out at Dokuz Eylül University Campus with fixed reference GPS positions. Static GPS sensor fixed at first reference point during the experiments for 30 minutes. Dynamic GPS sensor fixed at second reference point in first 10 minutes and then it has been moved for 20 minutes during 5 meter slow walking. Measured raw data are shown with blue lines, and filtered data with red lines for longitude and latitude values of static and dynamic points as shown in Figure 2.20.

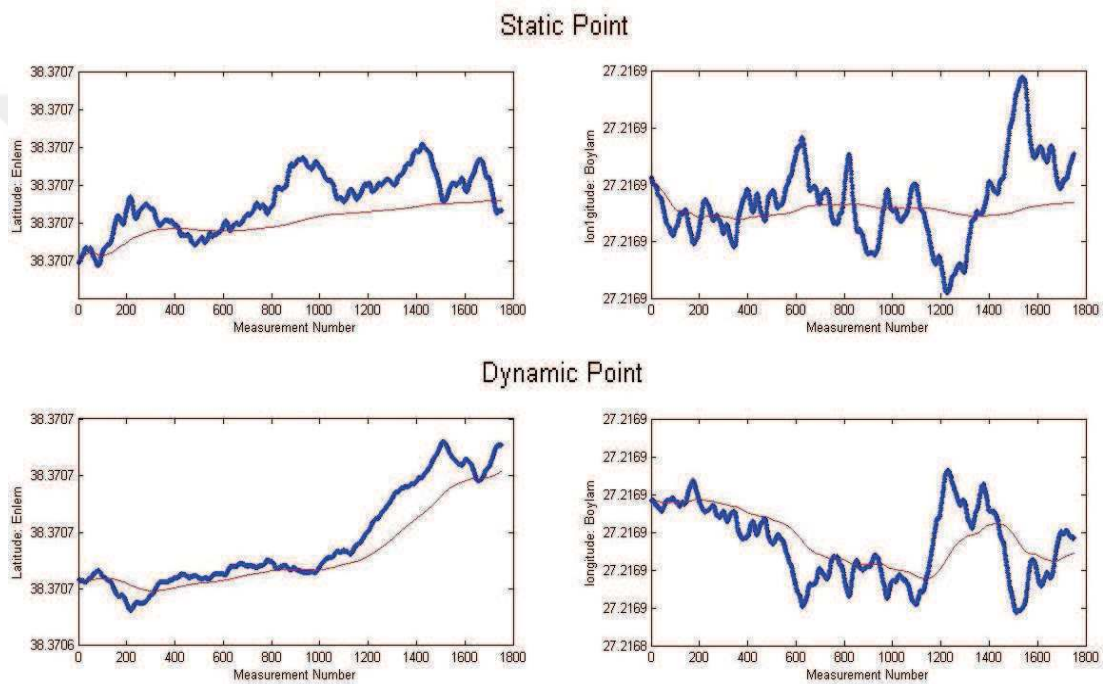


Figure 2.21 Raw (blue) and filtered (red) GPS position data

Measured and filtered GPS data are also super positioned to simulate trajectories and differences in MATLAB figure as shown in Figure 2.21.



Figure 2.22 Superposition of raw (blue and black) and filtered (purple and red) GPS position data

Measurements are obtained in WGS-84 Geographical Coordinate System and converted to WGS-84 Universal Transverse Mercator (UTM) Coordinate System to make calculations easier in Microsoft Excel program as given in Table 2.1 and Table 2.2 for two reference points. Resultant error of static GPS sensor at 1st point obtained as 0.39 meter with filtered mean GPS data in 30 minutes measurement time.

Table 2.1 Measurement errors of static GPS sensor at 1st GPS point

1st GPS Point		
	X/East	Y/North
Reference Data	518952.3527	4248669.2037
Measured Data	518952.6872	4248669.4112
Errors (m)	-0.33	-0.21
Resultant Error (m)	0.39	
Measurement time (min)	30	

Resultant error of dynamic GPS sensor at 2nd point obtained 0.27 meter with filtered mean GPS data in first 10 minutes of measurement time.

Table 2.2 Measurement errors of dynamic GPS sensor at 2nd GPS Point

2nd GPS Point		
	X/East	Y/North
Reference Data	518951.5569	4248662.9796
Measured Data	518951.6542	4248663.2327
Errors (m)	-0.10	-0.25
Resultant Error (m)	0.27	
Measurement time (min)	10	

Total Euclidean distance measured as 4.23 meter with tape measure and 4.6 meter with dynamic GPS sensor and 0.37 meter total error is obtained.

CHAPTER THREE

MECHANICS OF THE WHEELCHAIR

3.1 Mechanical Parts of the Wheelchair

The wheeled mobile robots model used in the thesis was originally a wheelchair which has four wheels and whose two wheels are actuated with two DC motors. Then tablet/monitor holder and LIDAR sensor holder parts are integrated to the vehicle. In the beginning, 3D model of the wheelchair is created with SolidWorks to analyse and present the output of the thesis. The assembly model is given in Figure 3.1 with view of the real wheelchair.



Figure 3.1 Created assembly model with real wheelchair

3.1.1 PC/Tablet/Monitor Holder

3.1.1.1 PC Holder for the First Prototype

Original wheelchair was come with a joystick holder shown in Figure 3.2 which was unbalanced and non-ergonomic for holding a laptop. In the first prototype, it is changed with a new PC holder (Figure 3.3) which is produced by cutting and welding techniques.



Figure 3.2 Unbalanced old PC holder V.1 (Personal archive, 2014)

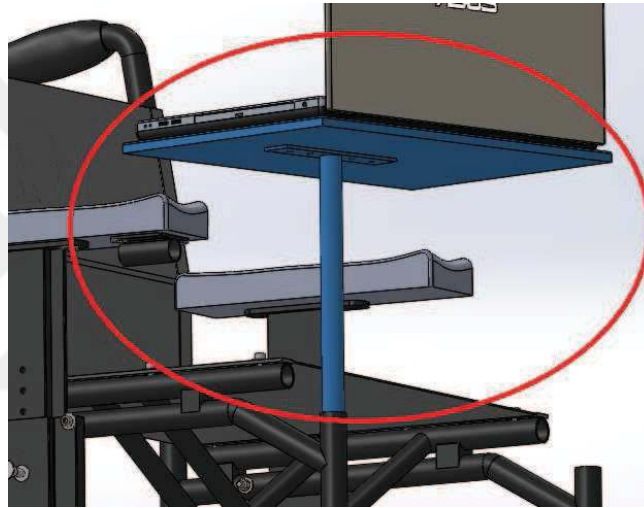


Figure 3.3 More balanced and ergonomic PC holder V.2

3.1.1.2 Tablet/Monitor Holder for the Second Prototype

More functional tablet/monitor holder for the second prototype is designed, and produced for more comfortable use of the wheelchair as shown in Figure 3.4 and Figure 3.5.

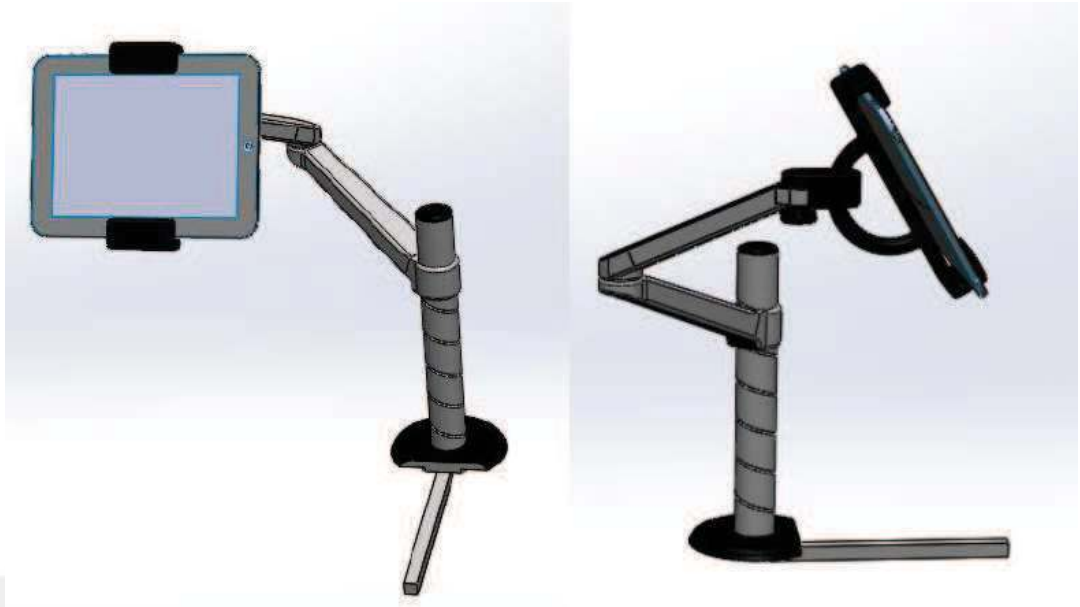


Figure 3.4 3D CAD model of tablet/monitor holder



Figure 3.5 Implementation of the functional tablet/monitor holder (Personal archive, 2016)

Finally Lilliput Fa1000, 9.7" 5-wire resistive touch screen monitor (Figure 3.6) is used to show GUI with Lenovo M700 Intel Core i5 6400T 2.8GHz Mini PC on the second prototype.



Figure 3.6 Lilliput 9.7" resistive touch screen monitor

3.1.2 LIDAR Holder

3.1.2.1 LIDAR Holder for the First Prototype

In the beginning of the thesis, two different parts are produced to hold LIDAR sensor on the first prototype. First design (Figure 3.7 left) was easy to mount and unmount but it was not parallel to the ground. Then the second design (Figure 3.7 right) was attached to vehicle body directly with two bolts. Final design of the first wheelchair prototype drawn with SolidWorks is shown in Figure 3.8.

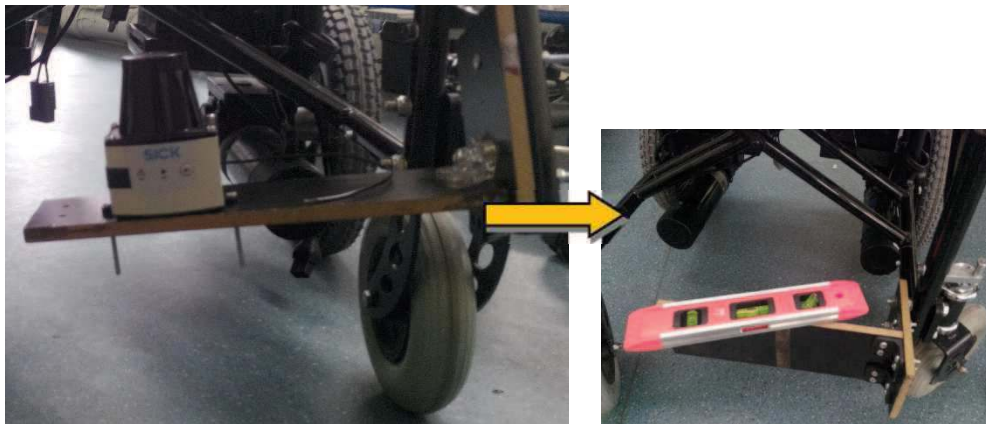


Figure 3.7 First designs for LIDAR connection part (Personal archive, 2014)



Figure 3.8 Final design of the vehicle with LIDAR

3.1.2.2 LIDAR Holder for the Second Prototype

New LIDAR protection cage for the second wheelchair prototype is designed, and produced by 3D printer as shown in Figure 3.9.



Figure 3.9 Protection cage of LIDAR produced by 3D printer (Personal archive, 2015)

Then it is produced by using laser cutting and bending techniques for more stable holding. Legroom of the wheelchair is covered to protect LIDAR for possible liquid drop as shown in Figure 3.10.



Figure 3.10 Protection cage of LIDAR under legroom of the wheelchair for the second prototype

3.2 Kinematic Equations of the Wheelchair

A representative control block diagram for controlling the first prototype vehicle's movements for combined distance and orientation control is given in Figure 3.11. The only input of the system is EOG signals received from the user. The signals are processed and destination points assigned on the PC. After high level algorithms and calculations made by the PC, low level distance and orientation calculations are made by microcontroller (distance/orientation controller). Going forward and turning

operations are separated in the first prototype because of encoder based position control algorithm. This problem is solved by LIDAR based real time position control and mapping algorithm in the second prototype.

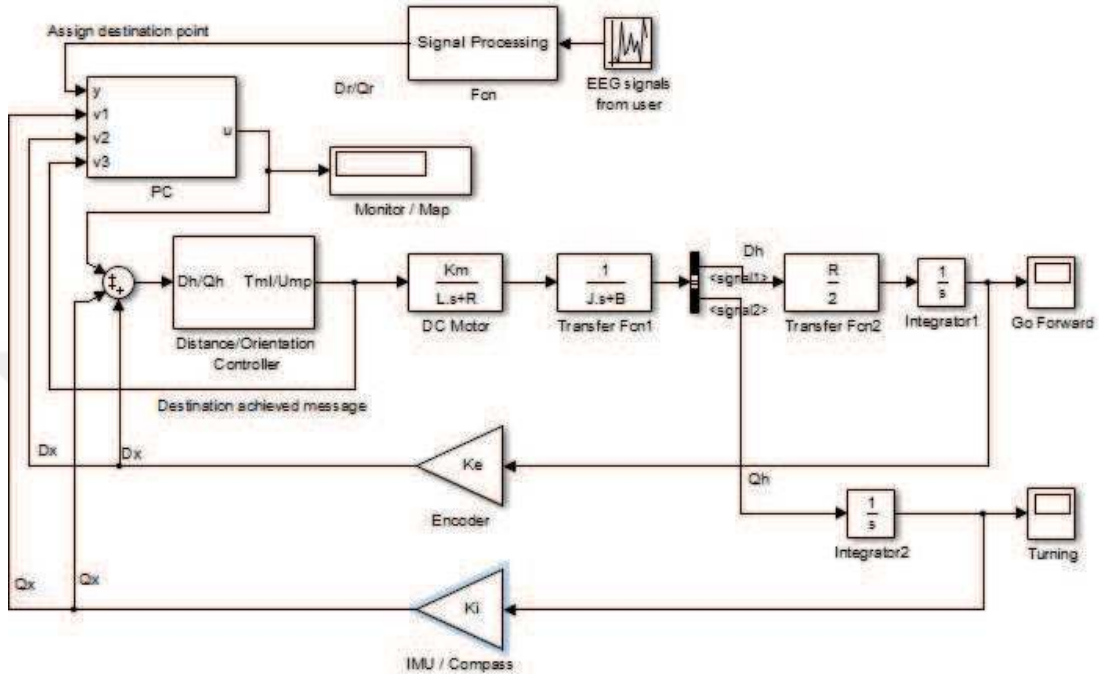


Figure 3.11 Combined model of distance (D) and orientation (Q) controls in MATLAB Simulink

For effective and balanced motion control, four wheeled mobile robot is planned to use. Left and right motor velocity equations of a four wheeled mobile robot whose two wheels are actuated can be seen in Figure 3.12.

The variables used in the following equations are described below:

r_c : Radius of curvature [cm],

L : Distance between the middles of two front wheels [cm],

v : Linear velocity of the mobile robot [cm/s],

$v_L(t)$: Linear velocity of the left front wheel [cm/s],

$v_R(t)$: Linear velocity of the right front wheel [cm/s],

θ : Heading angle [rad],

$w(t)$: Angular velocity in z coordinate of the mobile robot [rad/s],

$\{X_s, Y_s\}$: Stationary coordinate axes,

$\{X_m, Y_m\}$: Moving coordinate axes,

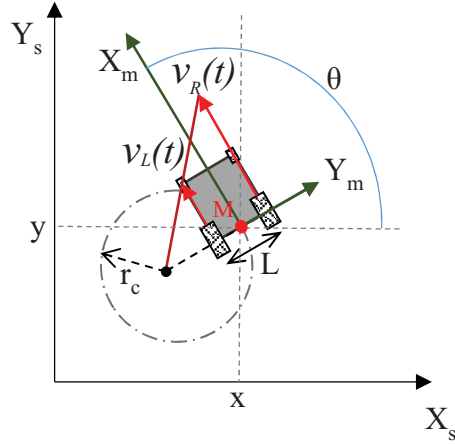


Figure 3.12 Velocities of left and right wheels of the wheelchair

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos \theta(t) & 0 \\ \sin \theta(t) & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} \quad (3.1)$$

Equation (3.2) shows the non-holonomic constraint of the model.

$$\dot{y} \cos \theta - \dot{x} \sin \theta = 0 \quad (3.2)$$

$\kappa(s)$ is the curvature at s and $r(s)$ is the radius of curvature at s in equation (3.3) (Gören, 2007).

$$\kappa(s) = r(s)^{-1} \quad (3.3)$$

Velocities of left and right motors can be calculated with equation (3.4) and equation (3.5) (Gören, 2007).

$$v_L = v \left(1 - \frac{L \cdot \kappa(s)}{2} \right) \quad (3.4)$$

$$v_R = v \left(1 + \frac{L \cdot \kappa(s)}{2} \right) \quad (3.5)$$

$$v_R = v \frac{r(s) + \frac{L}{2}}{r(s)} \quad (3.6)$$

In Figure 3.13, reference coordinate system and moving coordinate system; in Figure 3.14, velocity variations in respect to instantaneous center of rotation of the wheelchair can be seen.

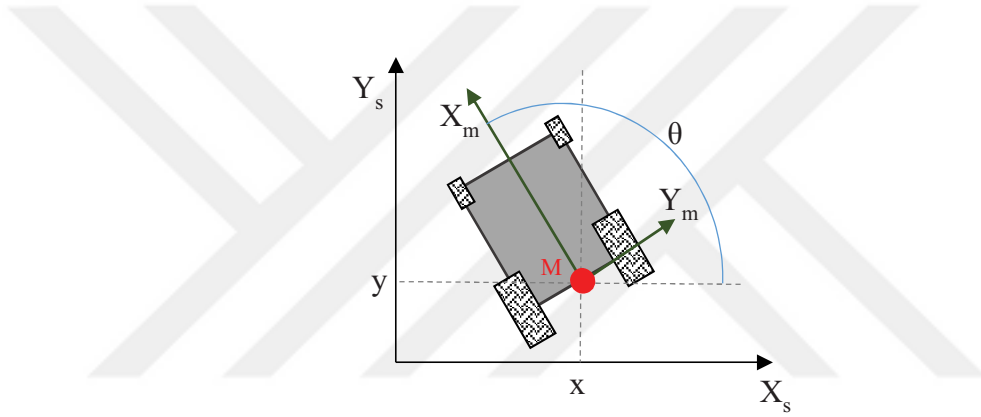


Figure 3.13 Coordinate system locations of the wheelchair

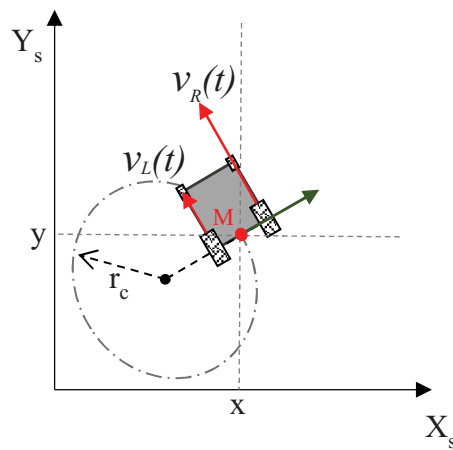


Figure 3.14 Left and right motor velocities of the wheelchair

Experimental vehicle position with respect to reference coordinates is:

$$q = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad (3.7)$$

To describe robot motion in terms of component motions, it will be necessary to map motion along the axes of the global reference frame to motion along the axes of the robot's local reference frame. The components of motion along this robot's local axes are computed (Siegwart et. al., 2004). Rotation matrix of the reference coordinates with respect to moving coordinate system is:

$$R(\theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.8)$$

We can compute the vehicle's motion in the global reference frame from the motion in its local reference frame by using rotation matrix. Velocities of the right and left side back wheels in the vehicle's local reference frame is shown in equation (3.9). θ is the angle between the stationary coordinate axis and the tangent of the curve at point M.

$$\begin{bmatrix} Vx(t) \\ Vy(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \left(\frac{r}{2}\right) & \left(\frac{r}{2}\right) \\ 0 & 0 \\ -\left(\frac{r}{L}\right) & \left(\frac{r}{L}\right) \end{bmatrix} \begin{bmatrix} w_L \\ w_R \end{bmatrix} \quad (3.9)$$

or

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos \theta(t) & 0 \\ \sin \theta(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} \quad (3.10)$$

In condition of going straight forward (if the velocity of the right and left wheel is

equal);

$$w(t)=0 \rightarrow \theta: \text{constant} \quad (3.13)$$

$$\text{and } v(t)=v_L(t)=v_R(t) \quad (3.14)$$

If the wheelchair turns around on a point,

$$v(t)=0 \text{ and} \quad (3.15)$$

$$w(t) = \frac{2}{L} v_R(t) \quad (3.16)$$

This motion is one of the most known advantages of the dual drive vehicles. In this case, instantaneous center of rotation of the wheelchair model is on the middle of the front axis. And the model can turn around without going forward or backward (Gören, 2007).

Travelled distance (x) of each wheel is calculated by using equation (3.17) for the first prototype which has 256 PPR encoders (while the second prototype has 360 PPR encoders). In the equation: n is pulses measured by encoders for each revolution of the wheels; d_{wheel} is the diameter of rear wheels of the vehicle and r_{gear} is the reduction ratio of the motors.

$$x = n[pulse] \times \frac{1 [rev]}{256 [pulse]} \times \frac{\pi d_{wheel} [cm]}{1 [rev]} \times r_{gear} \quad (3.17)$$

$$\text{for } d_{wheel} = 31.2 [cm] \text{ and } r_{gear} = \frac{1}{29} \rightarrow x = n \times 0.01337 [cm]$$

Every pulse of encoders are converted to distance in unit of centimetre (each pulse equals 0.01337 cm) and used to calculate velocity of the rear wheels (v) with equation (3.18). (Δx : Travelled distance between last two measurements; Δt : Elapsed time between last two measurements)

$$V = \frac{\Delta x [\text{cm}]}{\Delta t [\text{s}]} \quad (3.18)$$

PWM values which will be applied are obtained by sum of last applied PWM values with difference of reference velocity (v_{ref}) and calculated real velocity (v_{real}) with equation (3.19)

$$\text{PWM} = \text{PWM} + (v_{\text{ref}} - v_{\text{real}}) \quad (3.19)$$



CHAPTER FOUR

ELECTRONICS OF THE WHEELCHAIR

Driving system consists of a main control board with microprocessor, DC motor driver board, encoders, motors, LIDAR and a PC. Main control board gets reference velocity commands from PC, applies Pulse Width Modulation (PWM) values to the DC motor driver board, and calculates wheel velocities by incoming pulse data from encoders which are connected to motors. PWM values which will be applied to the motors are consistently calculated by using the reference velocities. Real-time velocities which are calculated by feed backed pulses to reach reference velocities for better position control at the right time.

The wheelchair is supplied with two 12 Volt gel batteries, charger converts 220 VAC to 24 VDC. Controller and motor driver are supplied by 24 Volt. The power connections is shown in Figure 4.1.

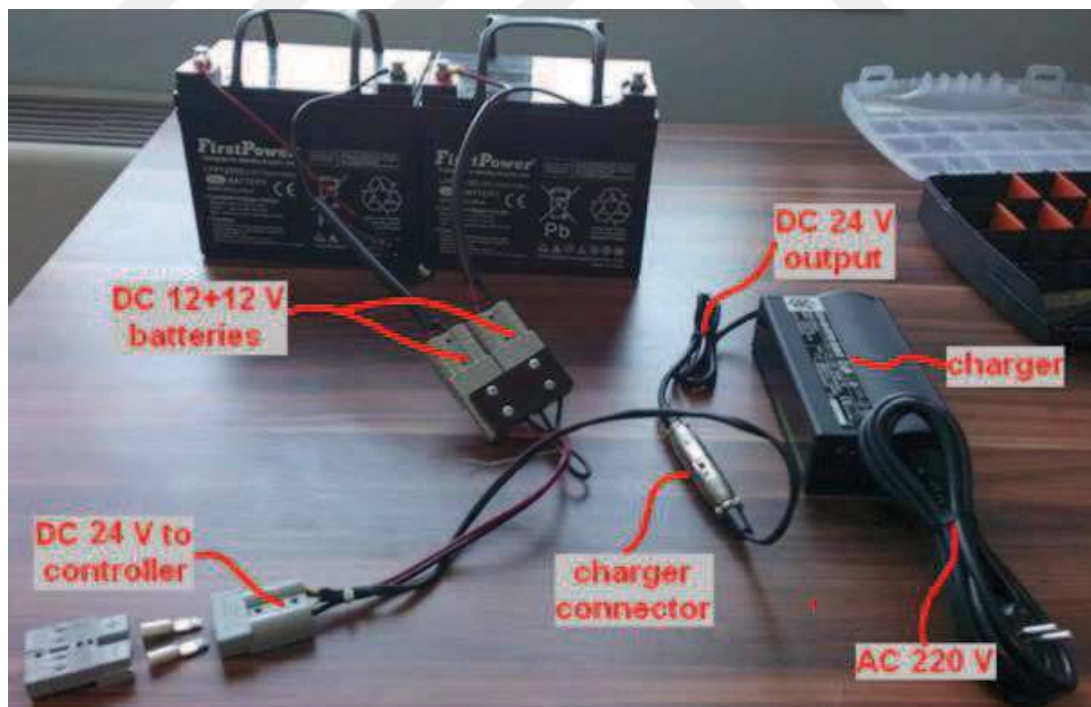


Figure 4.1 Batteries, charging unit and power connections (Personal archive, 2014)

Motor drivers have been designed with IRFZ44N MOSFET transistors for faster and more powerful driving and working ranges up to 55 V and 49A. Motor drivers are designed, and produced for 24 VDC motors as shown in Figure 4.2.

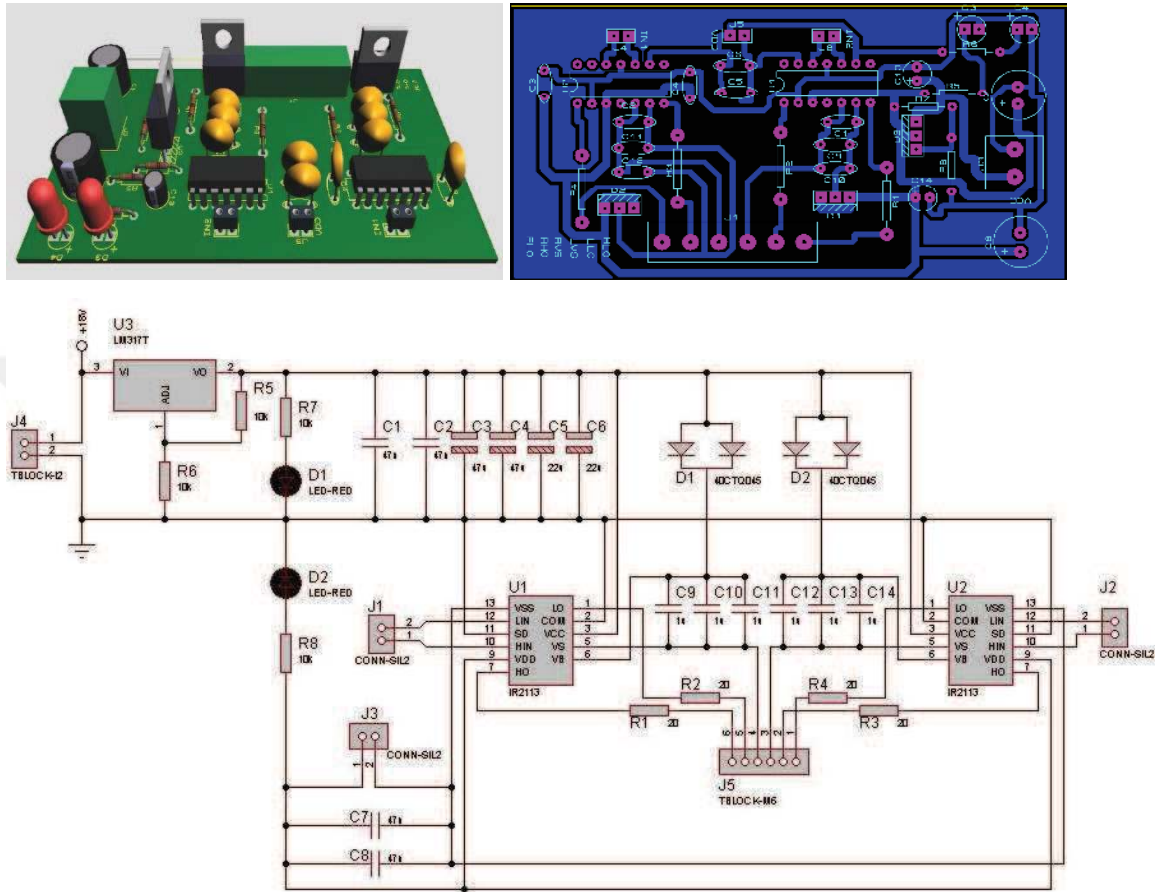


Figure 4.2 First motor driver V1.1

After that, noise on motor driver V1.1 is reduced with opto-isolator, wider solder traces, higher capacitors and higher MOSFET gate resistors. Second motor driver V1.2 is designed as shown in Figure 4.3. It can work between 17-33V with 20A maximum current.

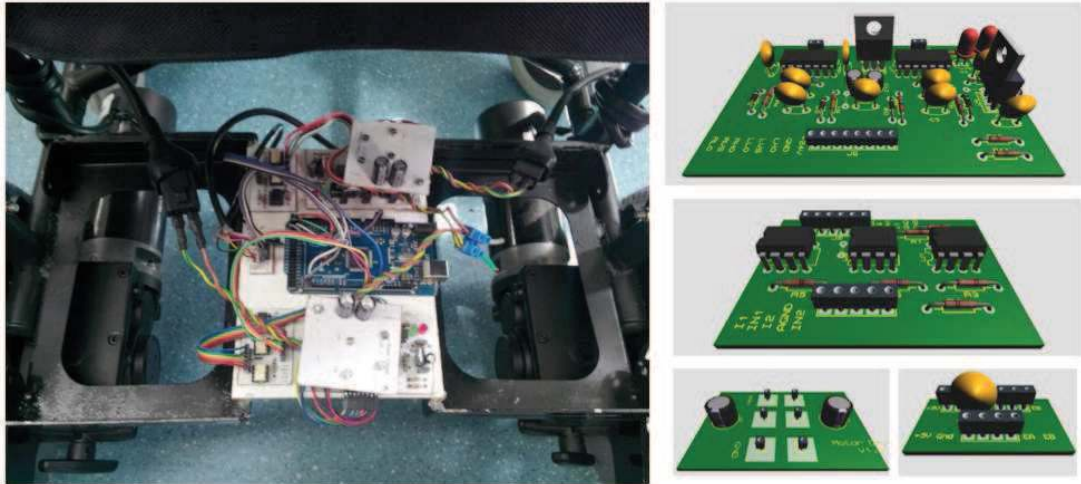


Figure 4.3 Second motor driver V1.2 and Arduino Mega Board as main control board

Motor driver is tested with tachometer, and the landmarks on the floor and wheel as shown in Figure 4.4. Left and right motor speeds are checked and calibrated for same speeds by using Arduino Mega control board as speed controller.



Figure 4.4 Motor speed tests (Personal archive, 2015)

After several burns on left side of the motor driver V1.2, motor driver V2.1 is designed with MOSFET transistors in a single piece and produced as shown in Figure 4.5. It does not need any opto-isolator and no velocity difference between right and left motors. It connected to interface program in computer for autonomous motor driving.

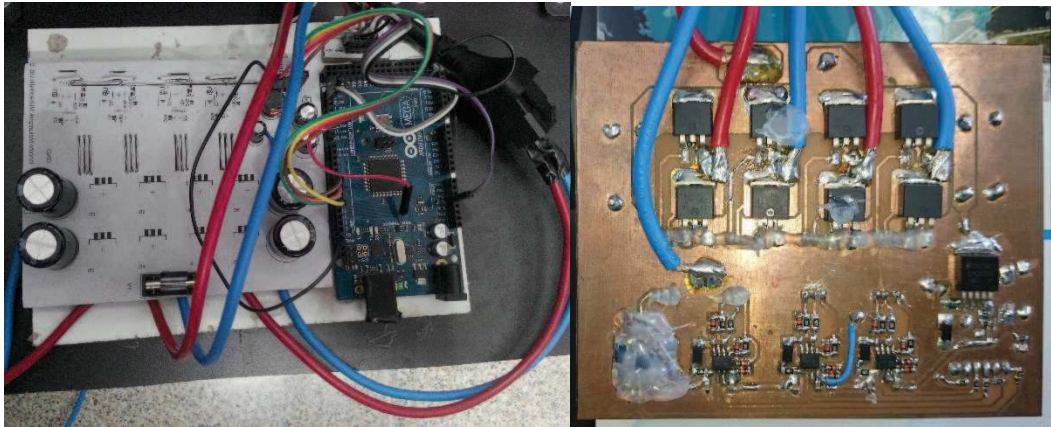


Figure 4.5 Double sided dual motor driver V2.1 (Personal archive, 2015)

After obtaining successful results with motor driver V2.1, motor driver V2.2 is designed, and produced with new connections, sockets and box as shown in Figure 4.6.

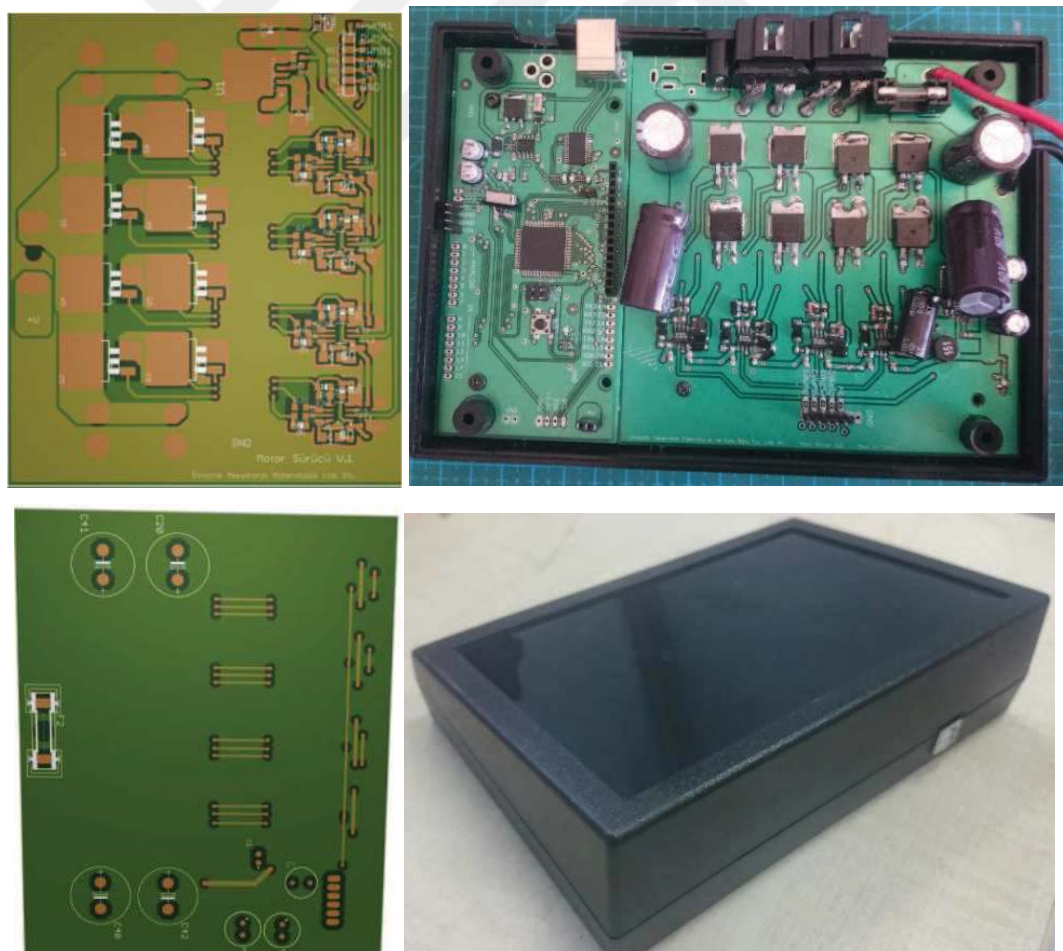


Figure 4.6 Main Control Board with double sided dual DC Motor Driver Board V2.2

Main control board and motor driver are placed between batteries (Figure 4.7) to reduce vibration and shaking which are encountered the mechanical problems on straight headway on the first prototype. Mechanical connections of the vehicle have been checked, and used plastic clips to provide more stability. Different speeds (PWM values) are applied to right and left motors with main control board.

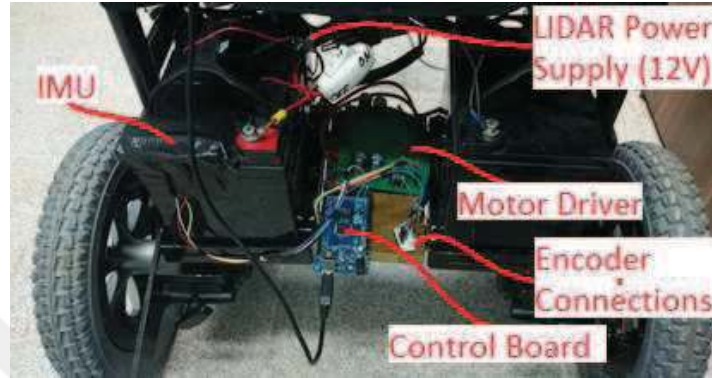


Figure 4.7 Final layout of electronic system of the first prototype (Personal archive, 2015)

Main control board software (Figure 4.8) has been further improved: In case of stopping or turning, yaw angle data are sent continuously. In case of headway, only the headway data measured by encoders is being sent to computer. All the headway and turning data are send via the same port. If the shortest path calculated on the interface program changes, then Destination Angle and Destination Headway value in the main control board is updated. The vehicle's direction and position in the interface program is updated according to incoming data from the main control board.

```
Engelli_Araci Encoder Motors_Direction Serial_Port Zero_Point
#define encoder0PinA 21
#define encoder1PinA 20

float refhiz=0.0,yol=0.0;
int pwm1=200,pwm2=200;
float reflpwm=40,ref2pwm=40; //MAX 250
int bufer[6];
int motor1apwm=6,motor1bpwm=7;
int motor2apwm=8,motor2bpwm=9;
int buffer_sayici=0, degree=0, refyol=0, refdegree=0, referans=0,dump;

volatile long int encoder0Pos=0;
long unsigned int newposition1=0;

void setup() {
  pinMode(13, OUTPUT);
  pinMode(motor1apwm, OUTPUT);
  pinMode(motor1bpwm, OUTPUT);
  pinMode(motor2apwm, OUTPUT);
```

Figure 4.8 Control board software

Main Control Board instead of Arduino Mega Board is designed and electronic components are soldered to PCB board on the second prototype as shown in Figure 4.9. Motor drivers are worked successfully after replacing some burned transistors and fuse. New encoders for new motors are mounted by removing original magnetic brake system on motors.



Figure 4.9 Final layout of electronic system of the second prototype (Personal archive, 2016)

Reference velocity values are used as main control board inputs instead of reference PWM values for controlling motors with LIDAR based real time localisation and mapping algorithm which are mentioned in Chapter 7.

CHAPTER FIVE

THE GRAPHICAL USER INTERFACE (GUI) OF THE WHEELCHAIR

Graphical user interface programs are developed to assign destination points, to show map and location of the vehicle and to calculate high level mathematical operations and algorithms. Visual Basic .NET (VB.NET) which is a multi-paradigm, object-oriented programming language, implemented on the .NET Framework is used for GUI of the first prototype on Windows OS. C# which is a multi-paradigm programming language encompassing strong typing, imperative, declarative, functional, generic, object-oriented, and component-oriented programming language is used for the second prototype on Linux Ubuntu OS.

Evolution of the GUI and Navigation Algorithm is as follows:

- In the beginning, the almost shortest path finding algorithm is developed, and implemented to achieve destination point in static environment basically.
- After that LIDAR sensor is implemented to consider dynamic obstacles, and simultaneous mapping feature is added with encoder and IMU based localisation. Shortest path finding algorithm have been made adaptive due to dynamic environment conditions and some improvements are made for optimal transportation. Some shift errors of map are reduced but the desired performance have not been achieved.
- For more precise real time localisation and mapping; LIDAR based localisation algorithms of Robot Operating System (ROS) libraries are implemented with some adjustment. Kinematic equations of the wheelchair is obtained and used to calculate curved shortest paths. Software has been finalised by implementation of originally developed collision detection and door passing algorithms.
- For an extra feature, a special keyboard is developed for people who are unable to use their hands to make computer use easier and to make writing faster.

5.1 GUI for the First Prototype on Windows OS

In the first prototype PC-User interface program is developed to assign destination points, to show map and location of the vehicle with VB.NET as shown in Figure 5.1. Menu and status bars are included to make development easier and faster.

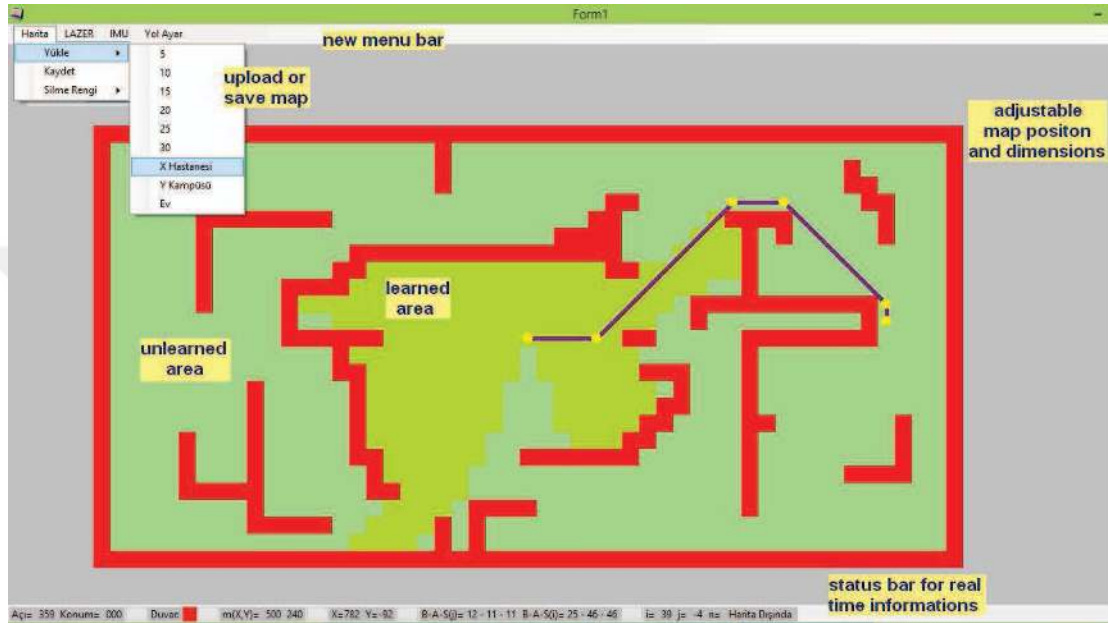


Figure 5.1 PC-User interface program written in VB.NET

Mouse functions are arranged and simplified as follows:

- Single Left Click: Assign destination point to left clicked cell.
- Left Click Drag: Assign starting point to start of left clicked cell and destination point to end of left clicked cell.
- Single Right Click: Make wall (obstacle) to right clicked cell.
- Right Click Drag: Make walls from starting point of the right clicked cell to end of right clicked cell as a straight line.
- Single Middle Click: Clear a single cell if it has wall (obstacle).
- Middle Click Drag: Clear cells from start of middle clicked cell to end of middle clicked cell as a rectangle.

5.1.1 Path Finding Algorithms

5.1.1.1 Almost Shortest Path Finding with Static Obstacles

First of all, almost shortest path finding algorithm is developed, and implemented to achieve destination point in static environment basically. The map is divided to cells and numbered by starting from assigned destination point to starting point as shown in Figure 5.2. After the starting point has been numbered the almost shortest path drawn by assigning sub destination points. Then the algorithm is optimised by deleting unnecessary sub destination points.

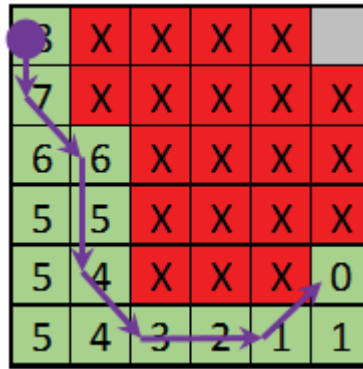


Figure 5.2 Almost Shortest Path

5.1.1.2 Adaptive Shortest Path Finding with Dynamic Obstacles

Afterwards LIDAR implemented for detection of dynamic obstacles and mapping feature. LIDAR mapping and yaw angle measurement with IMU functions are combined in one timer (function) to prevent coincidence and reading delays. Command processing times are measured with stopwatch function. Some operations which are not compulsory made elective, and processing performance of the program has been increased. Map loading and saving operations are also added to interface program.

This Adaptive Shortest Path Finding is based on updating of optimal path for changing dynamic environment and/or learning static environment. Main destination point is assigned by clicking on the map in computer program. Then program calculates, and draws the optimal path and defines sub-destination points. Program (Figure 5.3) recalculates and redraws the optimal path to the main destination point when the vehicle reaches each of these sub-destination points in order. If the vehicle comes across with a new obstacle or new closed door in updated map while following the optimal path estimation, program calculates new optimal path for the main destination point. It redraws new optimal path and redefines sub-destination points. In other words, the vehicle looks for open doors, avoids from dynamic and static obstacles, and finds the optimal path. Recalculation is important for avoidance of new obstacles on the way. In this algorithm, the vehicle searches and finds the optimal path to the main destination and this makes the vehicle smart. Also this method can be used for mapping and self-learning of the environment.

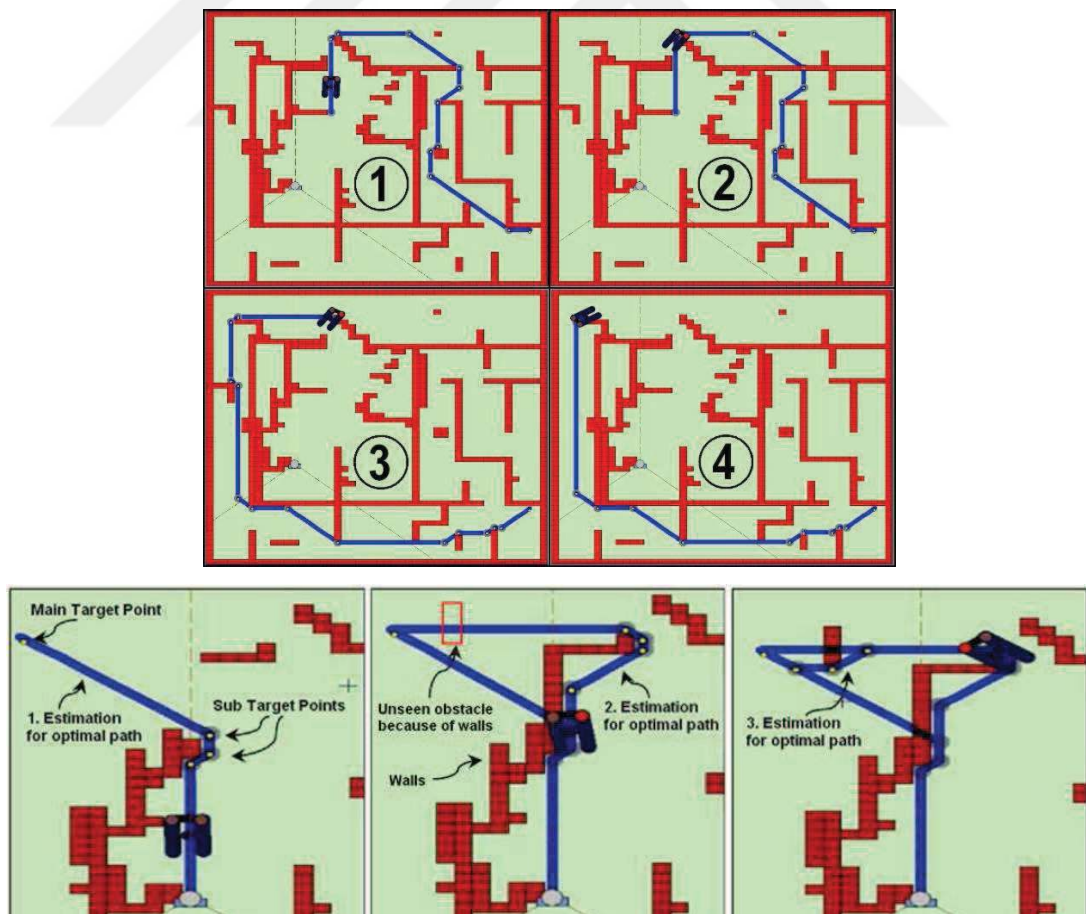


Figure 5.3 Adaptive (smart) optimal path finding algorithm

5.1.2 LIDAR Mapping Algorithms

Obstacle detection and mapping feature is developed for faster data transmission via Ethernet port and integrated to the existing computer program. The program for the first prototype can communicate with LIDAR by TCP/IP or RS-232 protocol. TCP/IP is used for better communication performance in 0.25° angular step scanning mode of SICK LMS100 LIDAR sensor (for first version of the first prototype). The program can get data in every 10 ms via Ethernet. Afterwards SICK TIM551 LIDAR sensor with 1° angular step scanning resolution, and SICK TIM561 LIDAR sensor with 0.33° angular step scanning resolution are used for the second prototype.

By using LIDAR environment is mapped and various objects in the surrounding is dedicated with its location and position. Scanned data of laser measurement sensor are processed and visualised in the computer program is explained in Figure 5.4.

Abbreviations used in Figure 5.4 are explained below:

n: number of distance measurements in each scan

i: cell number in a row

j: cell number in a column

Cw: width of a cell

Ch: height of a cell

as: the angular step of laser beam (0.25, 0.33, 0.5, or 1.0)

D: Measured distance by LIDAR

Wall: Defines the cell as obstacle (wall)

{Cx, Cy}: Center of the vehicle

{Mx, My}: Detected obstacle coordinates

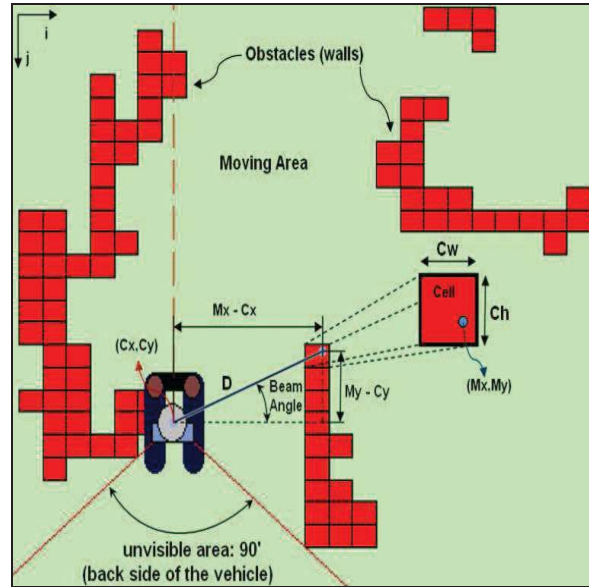


Figure 5.4 Visualisation of LIDAR data

The following software part is executed for each obstacle point measured by LIDAR.

```

Mx(n) = Cx + D(n) * Cos(315 + n * as)
My(n) = Cy + D(n) * Sin(315 + n * as)
i = Round(Mx(n) / Cw)
j = Round(My(n) / Ch)
Cell(i,j) = Wall

```

Detected obstacles can be viewed as points, map (connected points with lines) and/or visible area (laser beams in different shapes) as shown in Figure 5.5 on demand.

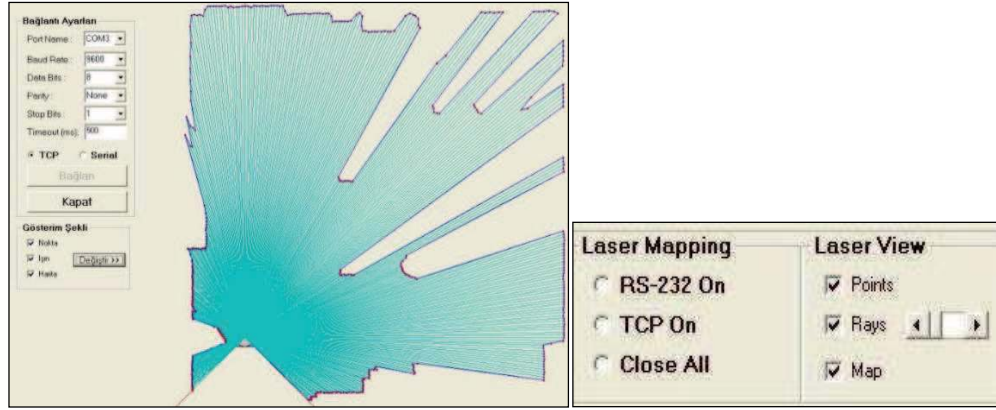


Figure 5.5 Laser beams visualisation

Marking dynamic obstacles on the map is a very easy issue beside to removing them in case of their movement to another position. For this purpose every obstacle marks under each laser beam to the nearest obstacle are deleted. In every LIDAR measurement data set occupied cells by each laser beam have to scan in slices as shown in Figure 5.6. In the figure, cell width and height is 200 pixels, scanning rate of blue laser beam is 400 pixels. The scanning rate of green laser beam is 1 pixel to prevent any possible obstacle jump.

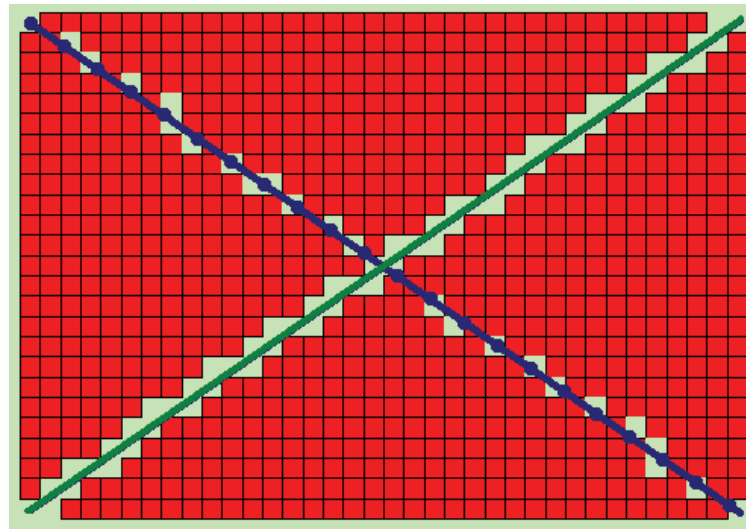


Figure 5.6 Laser beam scanning for removing obstacles on each cell

This situation can cause up to 1 million data control on maximum resolution and on maximum scanning rate with high screen resolutions.

Map in the simulation program is divided into two parts as shown in Figure 5.7: One part is visible area that learned, and the other part is invisible (dark) area that hasn't learned yet. In invisible area there can be some more obstacles, but when calculating the optimal path computer program doesn't consider these obstacles until LIDAR detects.

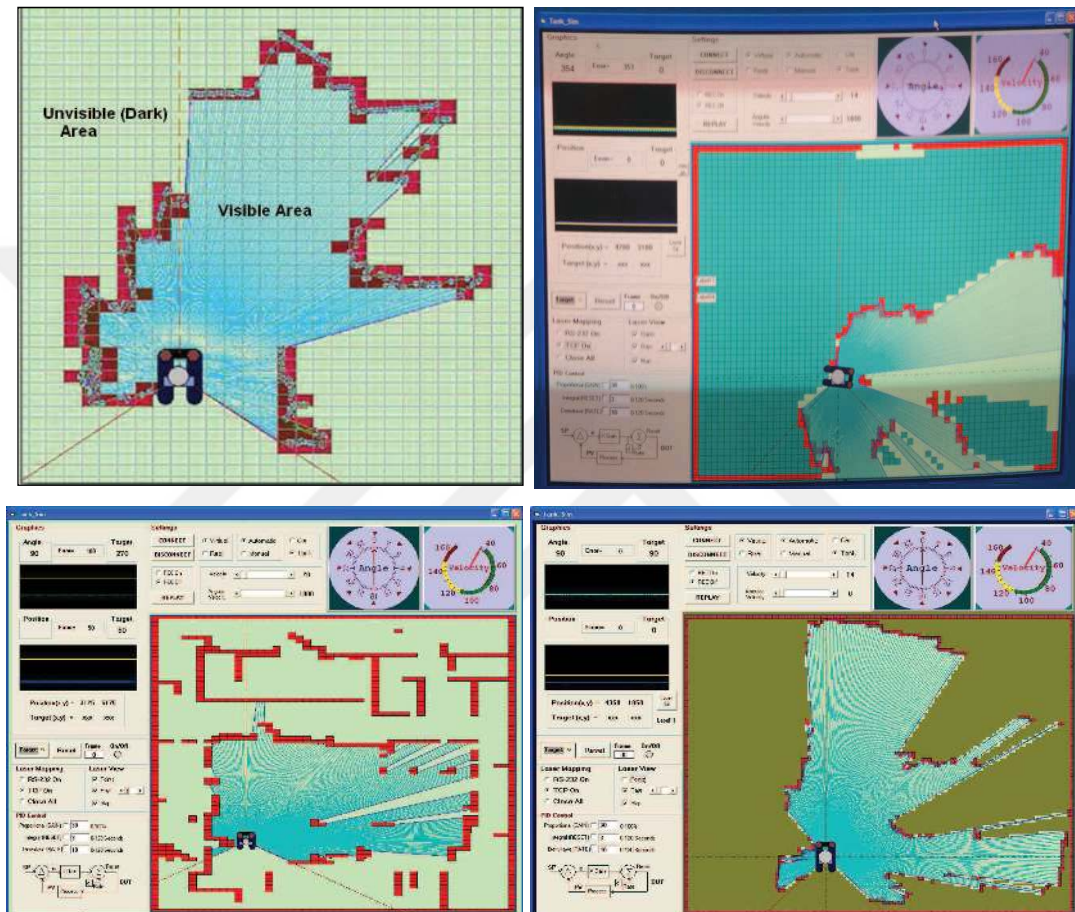


Figure 5.7 Basic LIDAR Mapping Program

New LIDAR sensor SICK TIM551 (Figure 5.8) is smaller and lighter than old LIDAR LMS100 with sensing range of up to max. 10 m and monitoring area of up to 235 m² is used for the second prototype. It has high ambient light tolerance due to HDDM technology and low power consumption of just 3 W for longer battery life. It has two communication options: One is from micro USB and the other is from Ethernet port.

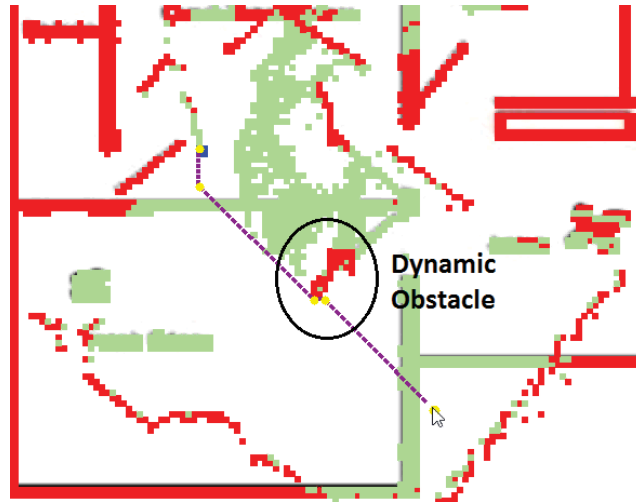


Figure 5.12 LIDAR mapping test with adaptive shortest path calculation

Detected obstacle points' widths are increased. Forbidden (impossible) areas are marked to avoid collisions and displayed with different colours (orange and green) as seen in Figure 5.13.

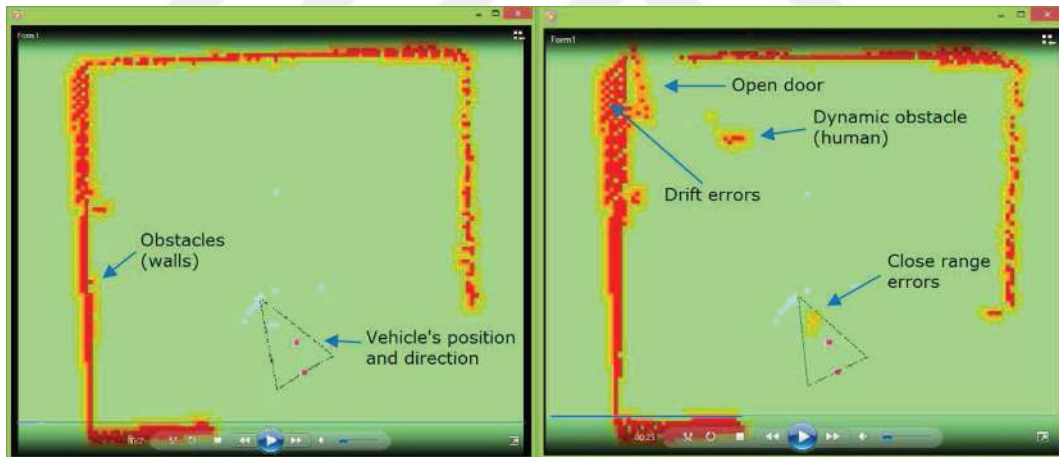


Figure 5.13 Three colour obstacle display

Obstacle and forbidden areas display has been improved according to the vehicle dimensions. Map is rotated according to the direction of the vehicle. The vehicle icon on the screen is kept constant (like navigation devices) instead of moving the icon while in motion. Horizontal and vertical scroll bars are added for viewing large scale maps as shown in Figure 5.14.

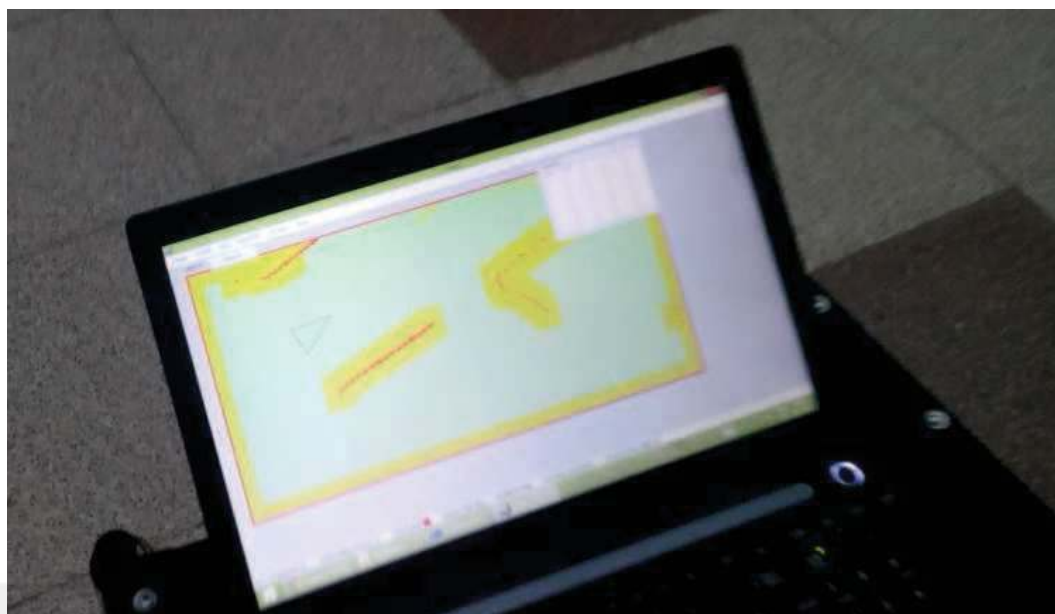


Figure 5.14 Improved obstacle displaying with forbidden areas (Personal archive, 2015)

New main control board software and interface program software are changed at the same time. So that makes it difficult to synchronize. After synchronization, showed that moving map application is effected significantly to the vehicle control period. After that, the map is decided to hold fixed, and the old interface software has been synchronized with the new main control board software.

Individual Occupancy Map Matrix like Occupancy Grid Mapping Algorithm has been created. With this method, obstacle occupancy value on a cell changes between 0-200. If a cell occupancy value is bigger than 100 than it is assumed that the cell has an obstacle on it.

Last two shortest paths of the vehicle and total distances of these have been calculated and recorded to memory. Although it is possible to compare these two paths, it was not used. Because, these comparisons would increase processing and control time lengths. (Because, it requires to separate the penultimate path to small slices, then it has to be checked if the cells that belong to junction points, have an obstacle)

Direction of the vehicle has been considered on the shortest path calculation instead of path distance comparison. In this way, the shortest path to the vehicle's front part

(LIDAR center) is calculated. After that auxiliary waypoints have been deleted, and moved according to rotation center (the center of the rear wheels) during headway.

Algorithms used in interface program have been simplified and standardised by changing the variables into matrix format. Additional software codes written to prevent errors, usually increase the response time. For example: Forbidden Area codes (that shows the walls wider) to prevent collisions with the walls increase response time.

At this time, additional software codes must written to decrease response time. For example: Instead of painting every pixels of a cell (5x5 pixels) to show obstacles on the map, decided to paint hatched. So, this is increased the cell painting speed by 4 times.

Also, array variables are used instead of ListBox/TextBox variables to increase code processing speeds. After all these updates and developments, desired performance has been obtained on obstacle avoidance, adaptive shortest path finding and mapping with the first prototype.

Final version of the software structure with regions are shown in Figure 5.15 below:

```

Kütüphane
Public Class Form1
    DEĞİŞKENLER
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
    #Region "FARE olayları"
    Private Sub Form1_MouseDown1(ByVal sender As Object, ByVal e As System.Windows.Forms.MouseEventArgs)
    Private Sub Form1_MouseUp1(ByVal sender As Object, ByVal e As System.Windows.Forms.MouseEventArgs)
    Private Sub Form1_MouseMove(ByVal sender As Object, ByVal e As System.Windows.Forms.MouseEventArgs)
    #End Region
    #Region "HARİTA olayları"
    Sub HaritaYukle(ByVal boyut As Single) ...
    Private Sub btnHaritaYukle_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Private Sub btnHaritaKaydet_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    #End Region
    #Region "BOYAMALAR"
    Private Sub DuvarBoya() ...
    Private Sub HucreBoya() ...
    Private Sub HucreCerceveCiz() ...
    Private Sub HaritaCerceveCiz() ...
    #End Region
    #Region "EN KISA YOL Bul/Çiz"
    Private Sub EnKisaYolBulCiz(ByVal X0 As Single, ByVal Y0 As Single, ByVal X1 As Single, ByVal Y1 As Single) ...
    Private Sub HucreDegerSil() ...
    Private Sub HucreDegerHesapla() ...
    Private Sub YolCiz() ...
    #End Region
    #Region "IMU - Seri Port"
    ...
    Private Sub btnYukleIMU_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Private Sub btnBaglanIMU_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Private Sub btnKapatIMU_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Private Sub btnGonderIMU_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Private Sub SerialPortIMU_DataReceived(ByVal sender As Object, ByVal e As System.DataReceivedEventArgs)
    Private Sub ReceivedText(ByVal [text] As String) ...
    Private Sub cmbPort_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Private Sub cmbBaud_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
    ...
    
```

Figure 5.15 General software structure of GUI

```

...
#End Region
#Region "LAZER - Ethernet"
Private Client As socketClient
Private Sub clientLogMessage(ByVal Message As String) ...
Private Sub Form1_FormClosing(ByVal sender As Object, ByVal e As System.Windows.Forms.Fc
Private Sub btnClientConnect_Click(ByVal sender As System.Object, ByVal e As System.Even
****Zorunlu
Private Sub handleClientConnected() ...
Private Sub handleClientDisconnected() ...
Private Sub handleClientIncomingData(ByRef Data As String) ...
****Tercihen
Private Sub handleClientConnectionError(ByVal Message As String) ...
Private Sub handleClientDisconnectError(ByVal Message As String) ...
Private Sub handleClientIncomingDataError(ByVal Message As String) ...
Private Sub handleClientSendDataError(ByVal Message As String) ...
Private Sub btnClientDisconnect_Click(ByVal sender As System.Object, ByVal e As System.E
Private Sub txtClientSend_KeyPress(ByVal sender As Object, ByVal e As System.Windows.For
Private Sub btnGonderLAZER_Click(ByVal sender As System.Object, ByVal e As System.EventA
Private Sub btnFiltreleLAZER_Click(ByVal sender As System.Object, ByVal e As System.Even
Private Sub btnCizLAZER_Click(ByVal sender As System.Object, ByVal e As System.EventArgs
Sub LazerSil(ByVal x0 As Single, ByVal y0 As Single, ByVal x1 As Single, ByVal y1 As Sin
Private Sub TimerLAZER_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
#End Region
#Region "MENÜ"
Menü HARİTA
Menü LAZER
Menü IMU
Menü YolAYAR
#End Region
YEDEK Bilgi
DENEMLER
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Ha
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Ha
End Class

```

Figure 5.16 General software structure of GUI

5.1.4 Video Recording and Playback Feature to Analyse the Vehicle Behaviours

For an additional work, video recording and playback feature is developed to analyse the vehicle behaviours. With this function program starts to record interface program screen by clicking “REC On” (Figure 5.16 left) in the beginning or in the middle of movement and stops recording by clicking “REC Off”. Recorded frame quantity can be followed in real time on main program interface as shown in Figure 5.16 right.

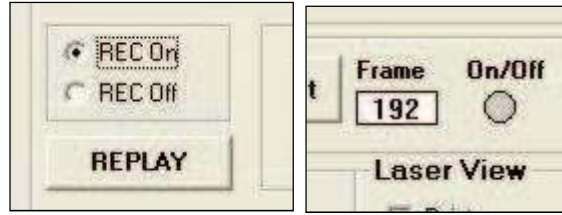


Figure 5.16 Video recording feature of the GUI

By clicking “REPLAY” button a new window (Figure 5.17) opens for playback and we can view every movement frame by frame or in different video rate. Video also can be saved as picture or video format on demand.

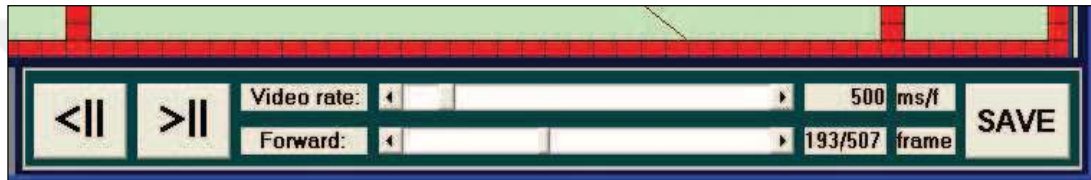


Figure 5.17 Video playing feature of the GUI

5.1.5 GUI Tests with Windows Tablet PC

The developed GUI for the first prototype is also installed and tested with a Windows tablet as shown in Figure 5.18.



Figure 5.18 Interface program works on a Windows tablet (Personal archive, 2015)

5.2 Mapping with ROS (Robot Operating System) in Linux Ubuntu OS

ROS provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license (Documentation n.d.).

LIDAR based real time localisation and mapping libraries (SICK TIM, GTK+, boost, pthread, **iocctl**, **termios**) of ROS are used to obtain more sensitive position control and mapping for the second prototype instead of encoder and IMU based localisation and mapping on the first prototype. After some errors and tests first LIDAR scan data are obtained and viewed as shown in Figure 5.19 and visualised as shown in Figure 5.20.

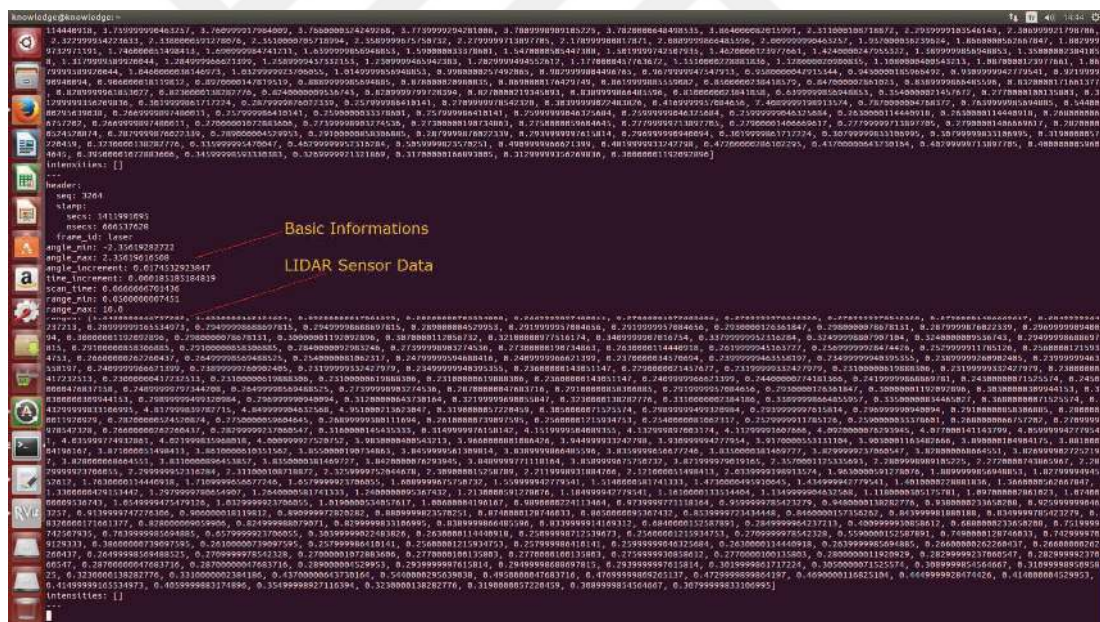


Figure 5.19 Received data with terminal

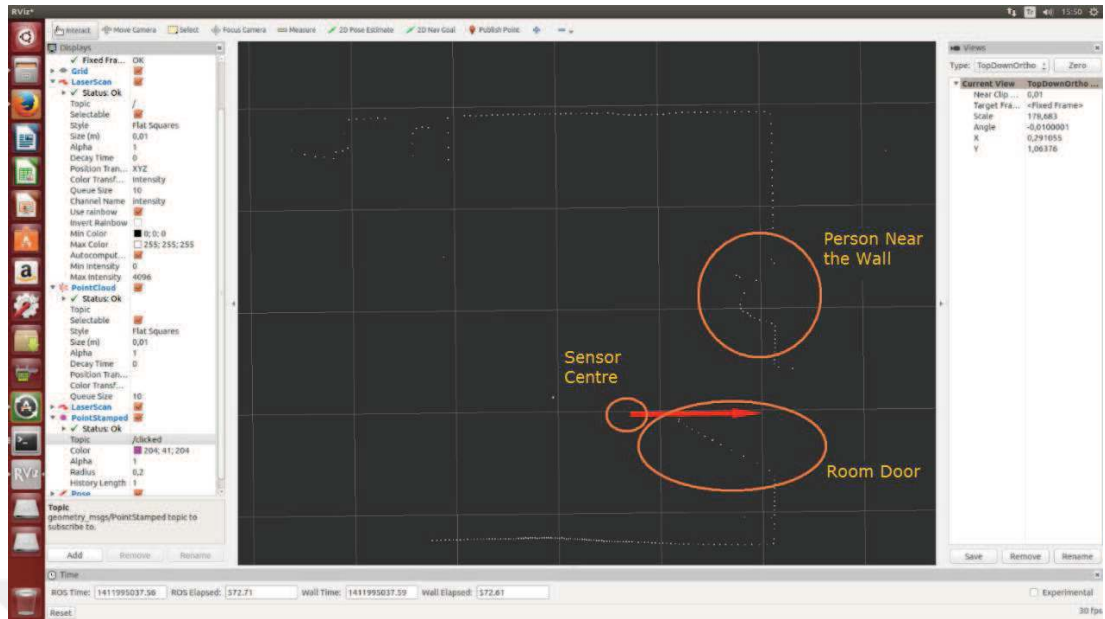


Figure 5.20 First LIDAR scan in ROS

After some unsuccessful studies indoor maps are generated successfully as shown in Figure 5.21.

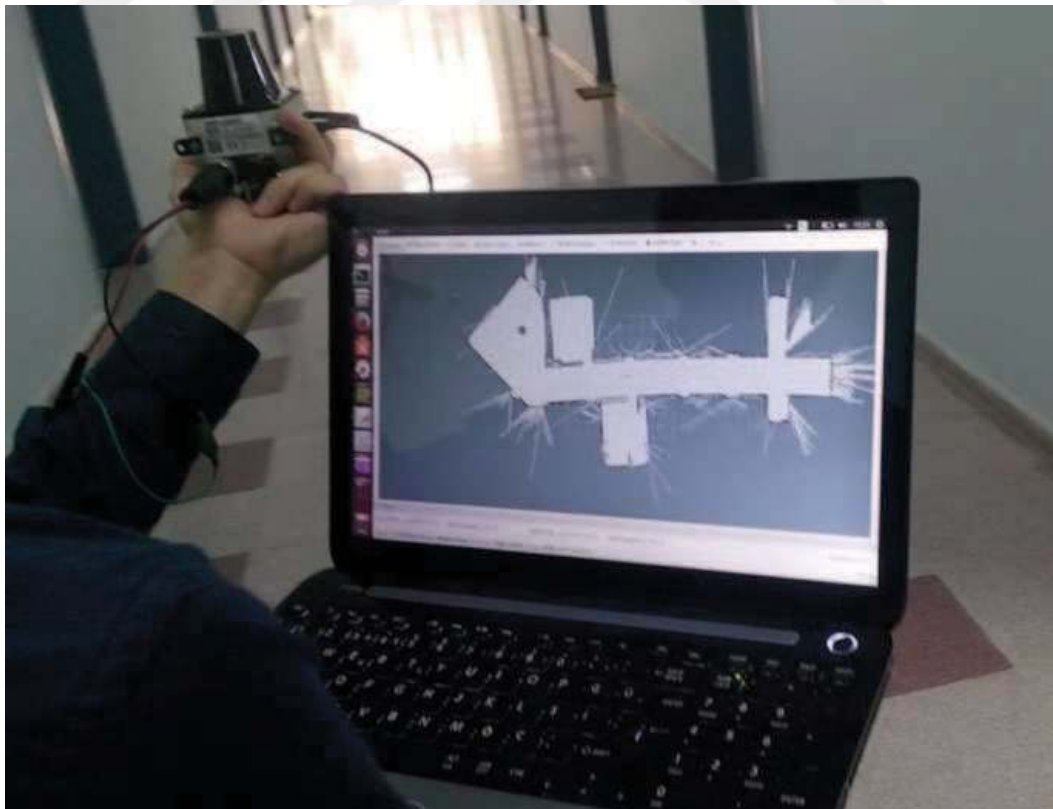


Figure 5.19 Mapping example with LIDAR in Linux Ubuntu OS (Personal archive, 2016)

Real time connection could not be established between ROS and the first GUI in Windows OS but the recorded maps could be uploaded. Different virtual machines and different Linux operating system versions are tested to solve this problem but could not get a satisfactory result. Port devices over virtual machine doesn't always work because the virtualization messes with the port latencies. Lastly, ROS worked without a virtual machine in Windows OS environment but maps could not be obtained then a new GUI is designed in Linux Ubuntu OS for the second prototype. All the path finding algorithms, USB and Ethernet connections are implemented to the new GUI. The user-defined areas are defined to mark obstacles that are outside the field of LIDAR sensor vision.

Prior to implementations and real environment test, mapping test with LIDAR is carried out. In the beginning, each wall, gate and door of the test environment is measured manually by a laser meter and a tape measure and sketched on a computer by using SolidWorks program. After that the vehicle with LIDAR sensor is walked around for mapping and map image is recorded. Finally sketch and map image are superposed and accuracy of the map which was obtained by LIDAR is checked as shown in Figure 5.22.

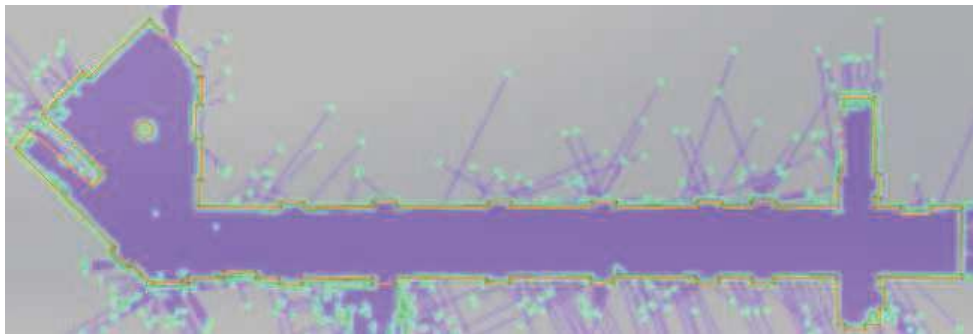


Figure 5.20 Super position of LIDAR map data (purple area with green points) with the sketch of the test environment (red lines)

Some LIDAR measurement errors as points in Figure 5.22 can be cleaned by using the program GUI and a clarified map is obtained as shown in Figure 5.23. As an additional feature, building sketches or pre-recorded maps can be uploaded as initial map to accelerate the process of learning and mapping environment.

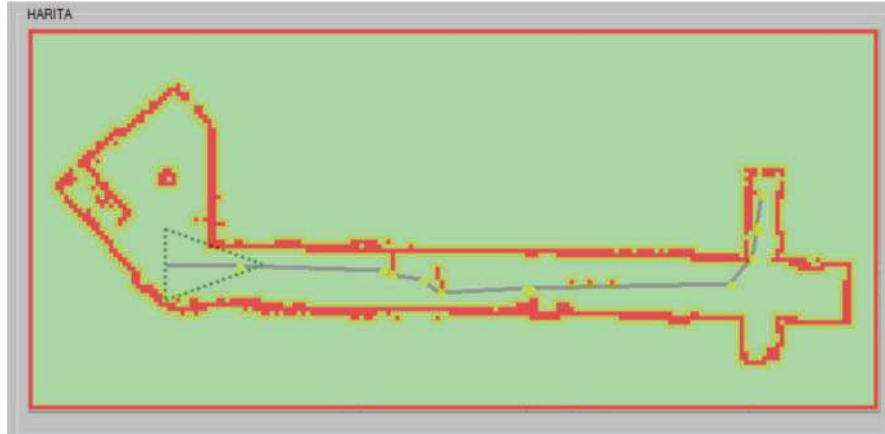


Figure 5.21 LIDAR map data read by the interface program on Windows OS

5.3 Special Screen Keyboard

For an extra feature a special keyboard is developed for people who are unable to use their hands. The keyboard can be used with just left eye blinks and the user can write something or can surf on internet. First simple version of the special screen keyboard is shown in Figure 5.24.

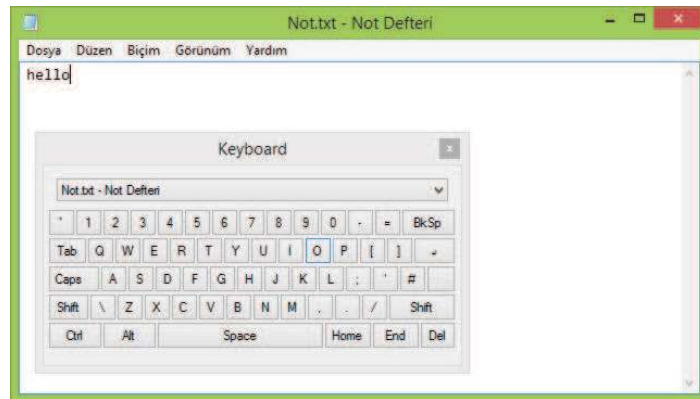


Figure 5.22 First (simple) version of screen keyboard

For faster writing purposes word and sentence prediction algorithm is developed and integrated to keyboard as shown in Figure 5.25. Special function like cut, copy, past etc. are integrated to keyboard. Most used applications are added to keyboard to make easier to open most used applications like internet explorer, my computer, my

documents or notepad to write something for communication with other people at home, rehabilitation center or hospital.

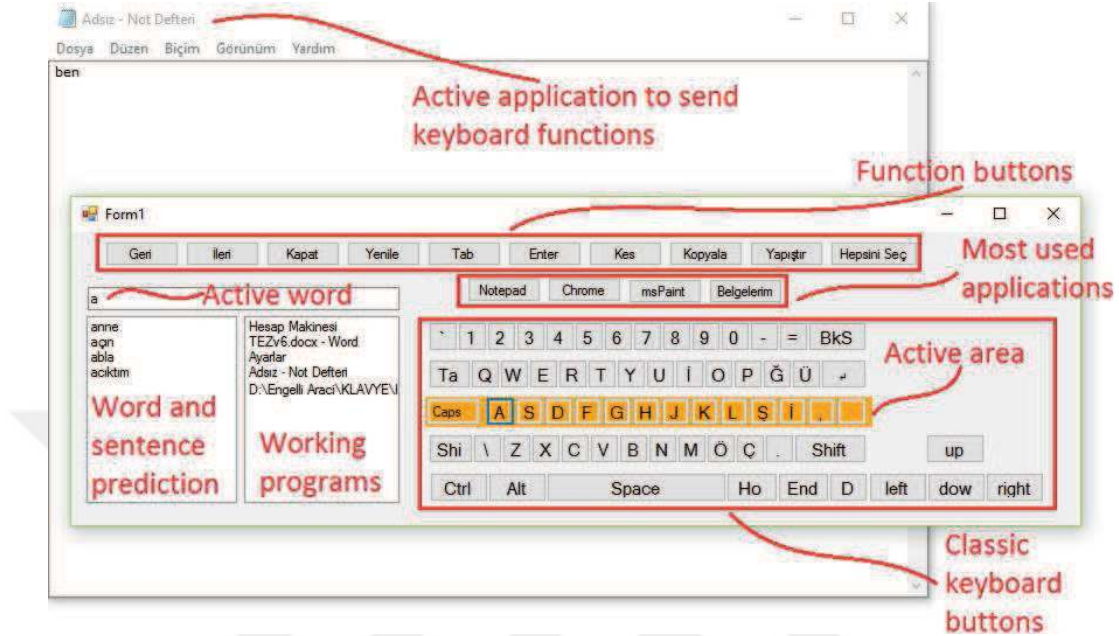


Figure 5.23 Last version of special screen keyboard with word and sentence prediction feature

Active area changes in every 3 seconds and the keyboard always waits an input from the user to select active area. There are four main active area:

1. Function buttons
2. Most used applications
3. Word sentence and prediction area that also consists working programs
4. Classic keyboard buttons area

After selection of main active area by incoming input signal like eye blinking, lower (smaller) areas are activated under that main active area. For example classic keyboard buttons area has 5 lower area consist of each 5 rows. Likewise each lower area or each row has lowest areas like separate buttons of each rows. The selection of active areas by incoming signal continues to selecting of lowest active area. After selection of lowest active area for example a button, the selected character of that button is written to active word area as shown in Figure 5.25. Then, word and sentence prediction is activated. Predicted words for active word is suggested and if there is a word written

before an active word, the possible words are also suggested to make writing faster. So it makes possible to communicate with clicking classic keyboard buttons, and sometimes without using classic keyboard buttons if the user used that sentence before.

If the disabled people or patients who unable to use their hands and unable to speak are thought, they mostly use similar sentences in daily life like “I am hungry/thirsty”, “open/close air conditioner/lights/TV etc.”. Therefore the special screen keyboard is very important to tell the user’s daily troubles to his/her caretaker or to tell what he/she wants to eat at dinner.



CHAPTER SIX

PROCESSING OF EOG SIGNALS

6.1 Measurement of EOG Signals

Emotive EPOC headset is used in this study is shown in Figure 6.1. The interface programs developed by Emotive Company are used. The data measurement rate is 128 samples per second with 14 bit resolution. Signals received from 16 different EOG/EEG sensors on headset with a wireless communication are transferred to the computer and are visualised on Emotiv TestBench program (Figure 6.2).



Figure 6.1 Emotive EPOC headset (EMOTIV, n.d.)

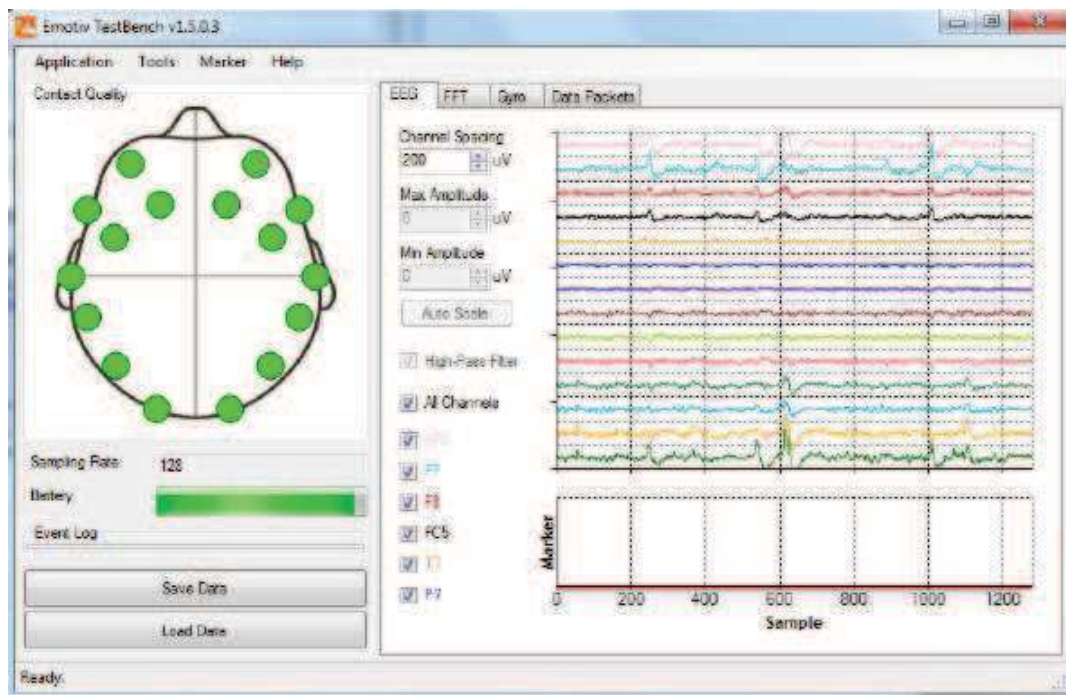


Figure 6.2 EOG/EEG signals and connection quality of sensors with green dots

16 green dots in Figure 6.2 show positions of EOG/EEG sensors and colour of these dots (green, yellow, red and black) indicate connection quality of each sensor. Felt pads in sensors must be wetted to get more quality data with a special solution before headset is used. If all the sensors are nearly green then we can start signal processing. Signals received from different parts of the head change with thoughts, speaking and facial expression.

Regarding to signals received in different conditions, if similar signals which defined before are received in a time interval, then the program recognizes it, and outputs the detected condition.

Sometimes two sensors data are enough to analysis for detecting these conditions (closing eyes in Figure 6.3 and winking Figure 6.4) instead of analysing all sensors.

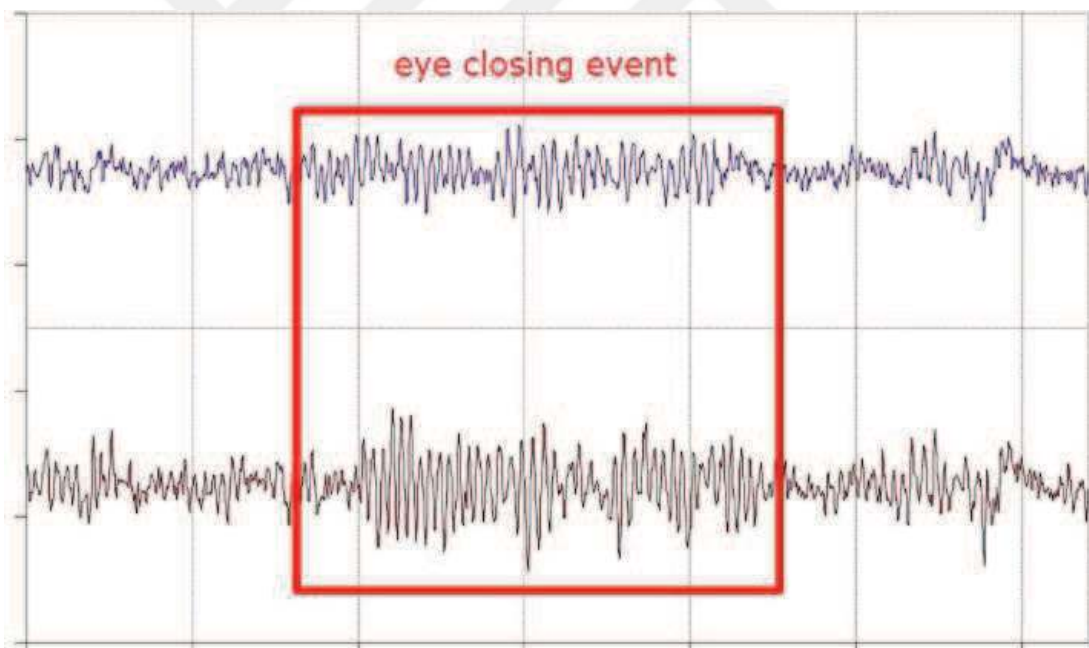


Figure 6.3 Change in sensors O1 and O2 while closing eyes (Adelson, 2011)



Figure 6.4 Change in sensors AF3 and AF4 while winking (Adelson, 2011)

Expected condition and action regarding to this condition is defined in EmoKey program shown in Figure 6.5. Desired commands can be given if a thought or facial expression occurs. For example, left clicking of mouse is linked with left winking with EmoKey program. Mouse cursor movements are controlled with a gyroscope on headset.



Figure 6.5 Conditions and commands (rules) regarding to these conditions

We can also map our brain activities and visualize 16 EEG signals on Emotiv EPOC program as shown in Figure 6.6 and Figure 6.7.

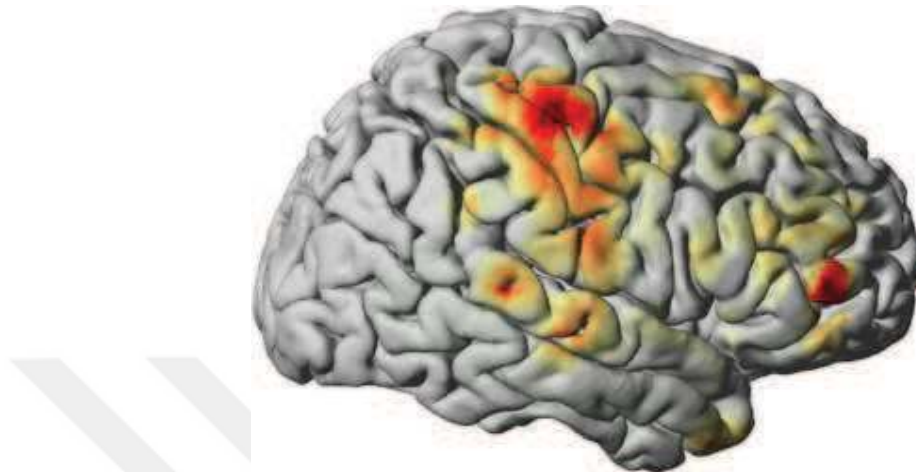


Figure 6.6 A sample of magnetic density on brain while thinking about something (Oschler, 2010)

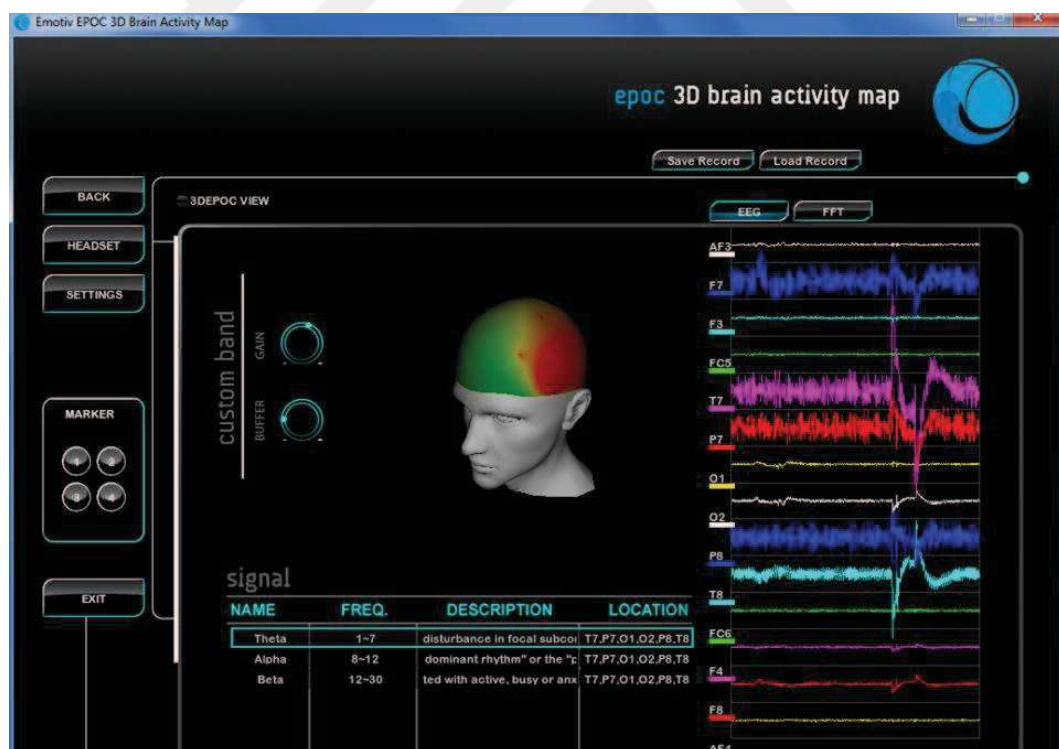


Figure 6.7 3D Brain activity map

6.2 Analysis of EOG Signals

In the beginning, received EOG signals are recorded to a text file and put in a graph (Figure 6.8) in Excel to understand the signals and eye blinks.

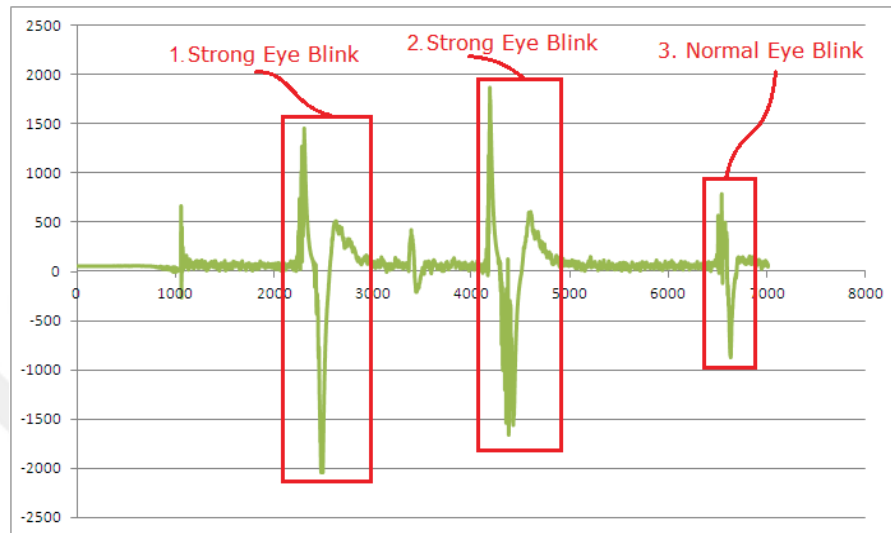


Figure 6.8 Three eye blinks in Excel Graph

After raw EOG signals are filtered and converted simply, eye blinks can be characterized with $[0, 1, 0, -1]$ as shown in Figure 6.9. But this was not enough to detect signals properly.

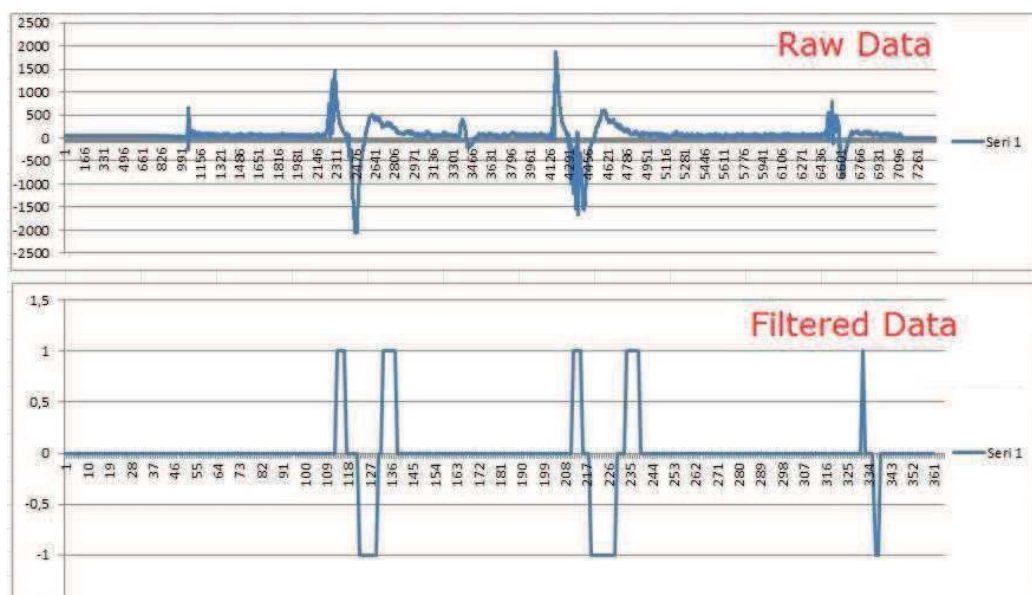


Figure 6.9 Simply filtered signals

After that, for preliminary study of mouse control program to assign destination points, raw EOG signals are analysed. Eye blinks are detected with normalized cross correlation method in MATLAB by using Emotiv EPOC 14 Channel Wireless EEG Headset.

For signal-processing applications in which the amplitude of the signal and search template can vary due to mimics and thoughts, the signal can be first normalized. Normalized cross correlation is used for comparing template (t) and last actively buffered signal (f) in every step. Mean is subtracted and divided by the standard deviation (σ_f : standard deviation of actively buffered signal and σ_t : standard deviation of the template) as given in equation (6.1). The cross-correlation of a template is given as $t(x,y)$ and the actively buffered signal is given as $f(x,y)$. The number of pixels in $t(x,y)$ and $f(x,y)$ is given as n , the average of f is given as $\bar{f}$ and the average of t is given as $\bar{t}$ in the equation.

$$\frac{1}{n} \sum_{x,y} \frac{(f(x,y) - \bar{f})(t(x,y) - \bar{t})}{\sigma_f \sigma_t} = \frac{\frac{1}{n} \sum_{x,y} f(x,y)t(x,y) - \bar{f}\bar{t}}{\sigma_f \sigma_t} \quad (6.1)$$

Normalized cross correlation matches the series. Its result is in the range of +1 (exactly same) and -1 (exactly opposite). Pre-recorded Search Template of an eye blink's EOG signal graph is shown in Figure 6.10a. Raw EOG signal which consists of three eye blinks is shown in Figure 6.10b and detected eye blinks and their correlation values are shown and marked on the graph as illustrated in Figure 6.10c.

In this example peak threshold value is taken as 0.3, and the similarity (correlation) threshold value is taken as 0.4. Three similarities are found as shown in Table 6.1 and Figure 6.10c. Desired similarities can be filtered with proper threshold values and proper search window size for exclusive usage or different signal types.

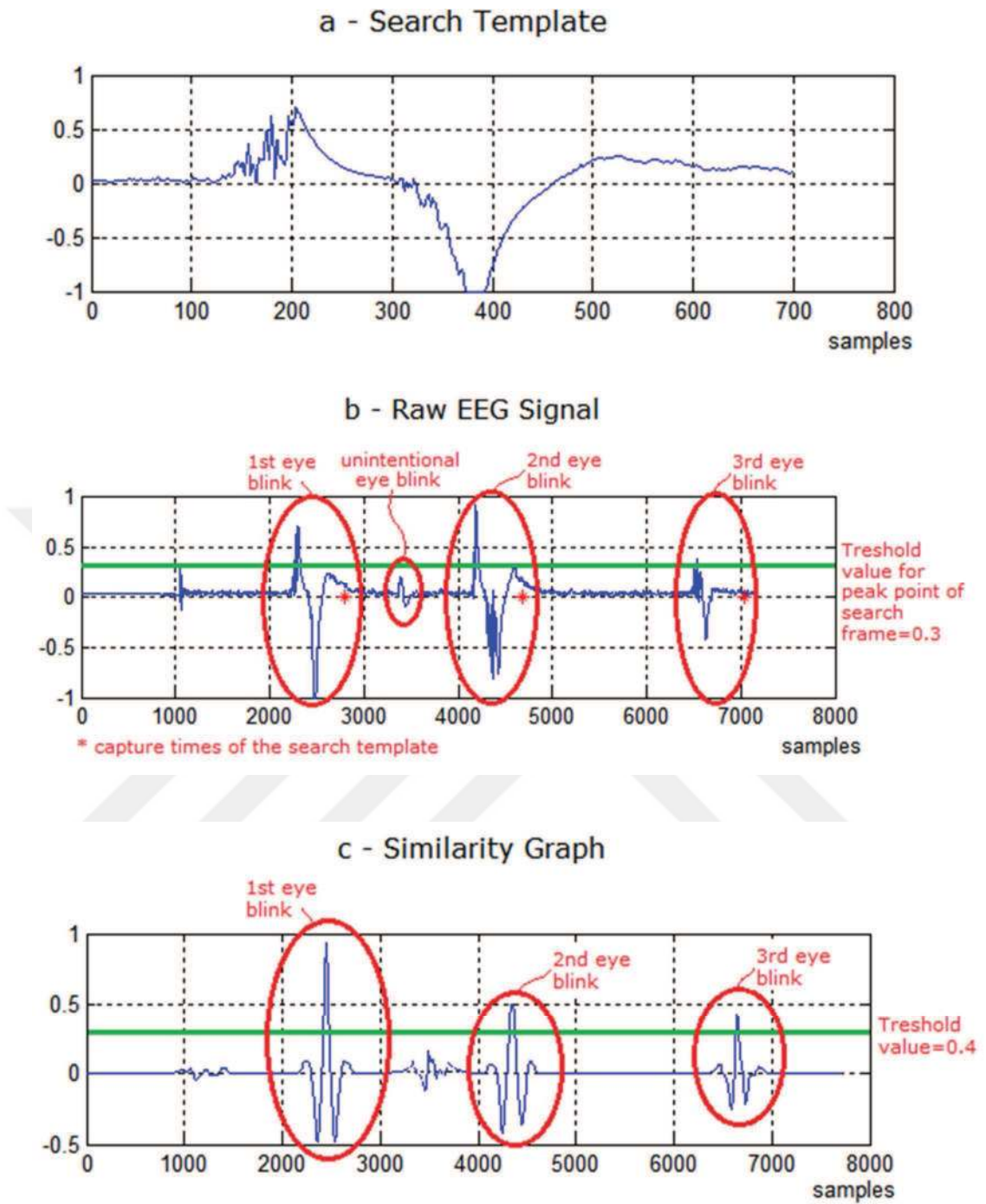


Figure 6.10 Cross Correlation of raw EOG signal with search template in MATLAB

Table 6.1 Correlation values of each capture between Search Template

Peak Point Detections	1st	2nd	3rd	4th	5th
Peak Value of Active Search Window	0.3237	0.7085	0.2056	0.9126	0.3843
Is over peak threshold value (0.3)?	✓	✓	X	✓	✓
Similarity Value of Active Search Window	0.0375	0.9325	0.2674	0.4945	0.4306
Is over similarity threshold value (0.4)?	X	✓	X	✓	✓
Time (sample)	1551	2801	3887	4685	7038
Is an eye blink?	X	✓	X	✓	✓

It is understood that unintentional eye blink (3rd peak point detection) can be captured by using 0.2 as peak threshold value, and 0.2 as similarity threshold value without capturing incorrect signal (1st peak point detection) if it is desired.

After these preliminary studies this method is used to detect real time EOG signals outside MATLAB, and adopted to a computer program (Figure 6.11). In this program search templates can be recorded to detect in real time, and similarities can be shown on the graph.

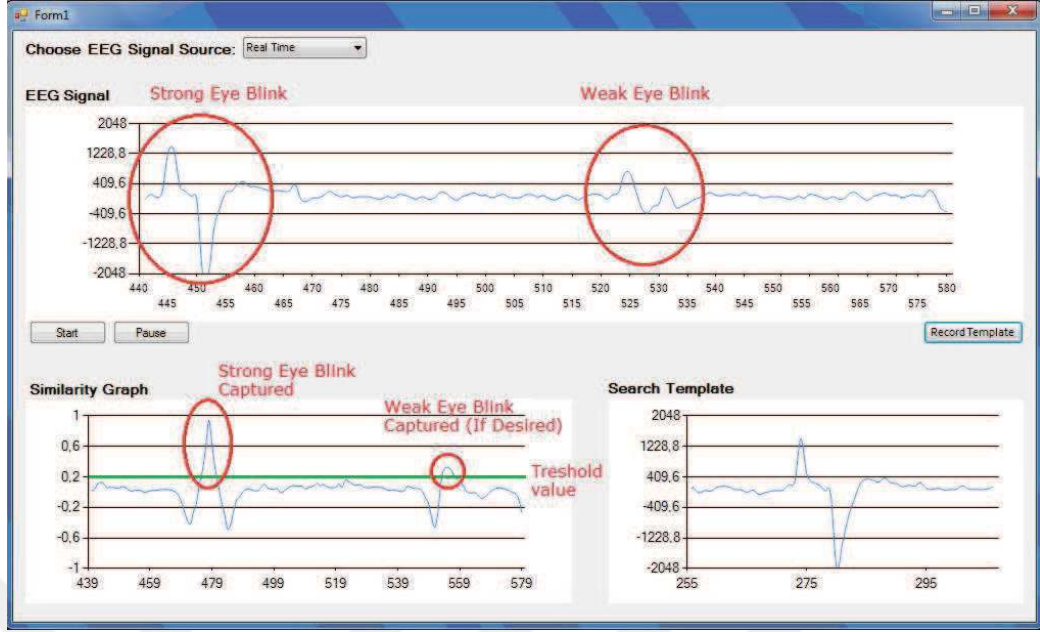


Figure 6.11 EOG signal analysis

6.3 Hands-Free Mouse Control Program to Assign Destination Points

Mouse control program has been developed to control mouse clicks and mouse movements by using EOG and gyro data that received from the headset. Search templates are recorded at program start up as shown in Figure 6.12.

```

D:\RAPORLAR - EA\2015-02-22 EEG\emotiv gyro - 8.5_8\EmotivEeg...
*Sol göz kırpma = Sol tıklama

Sağ göz kırpma kaydı için: 1
Sol göz kırpma için: 2
Kaş kaldırma için: 3
1Sağ göz kırpma kayıt başlatıldı.
1Sağ göz kırpma kayıt tamamlandı.
2Sol göz kırpma kayıt başlatıldı.
2Sol göz kırpma kayıt tamamlandı.
3Kaş kaldırma kayıt başlatıldı.
3Kaş kaldırma kayıt tamamlandı.

```

Figure 6.12 Recording of sample signals

Left eye blinks are used for mouse left clicks, right eye blinks are used for right mouse clicks and raise eyebrows are used for middle mouse clicks as shown in Figure 6.13. Left mouse clicks (left eye blinks) are used to assign destination points, right

mouse clicks (right eye blinks) are used to mark fixed obstacles. Middle mouse clicks (raise eyebrows) are used to delete obstacles on the map.

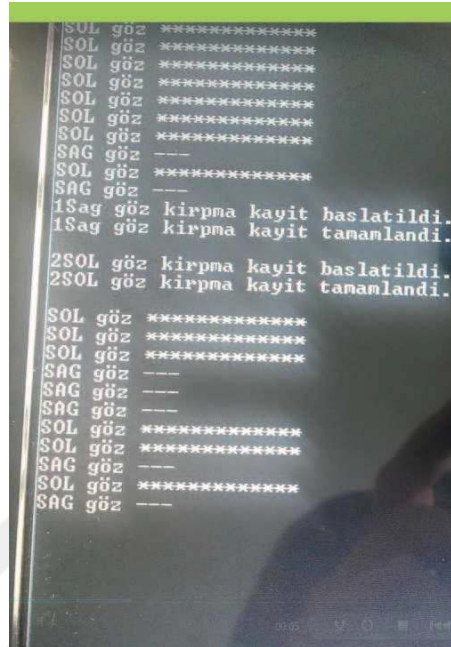


Figure 6.13 Left/right eye blinks and mouse move tests

CHAPTER SEVEN

IMPLEMENTATIONS AND TESTING

7.1 Final Navigation Algorithms

Finally Curved and Adaptive Shortest Path Finding Algorithm is developed for more comfortable driving and for more practical transportation. The flow chart of the developed algorithm is given in Figure 7.1. The abbreviations used in the flow chart are explained in Table 7.1.



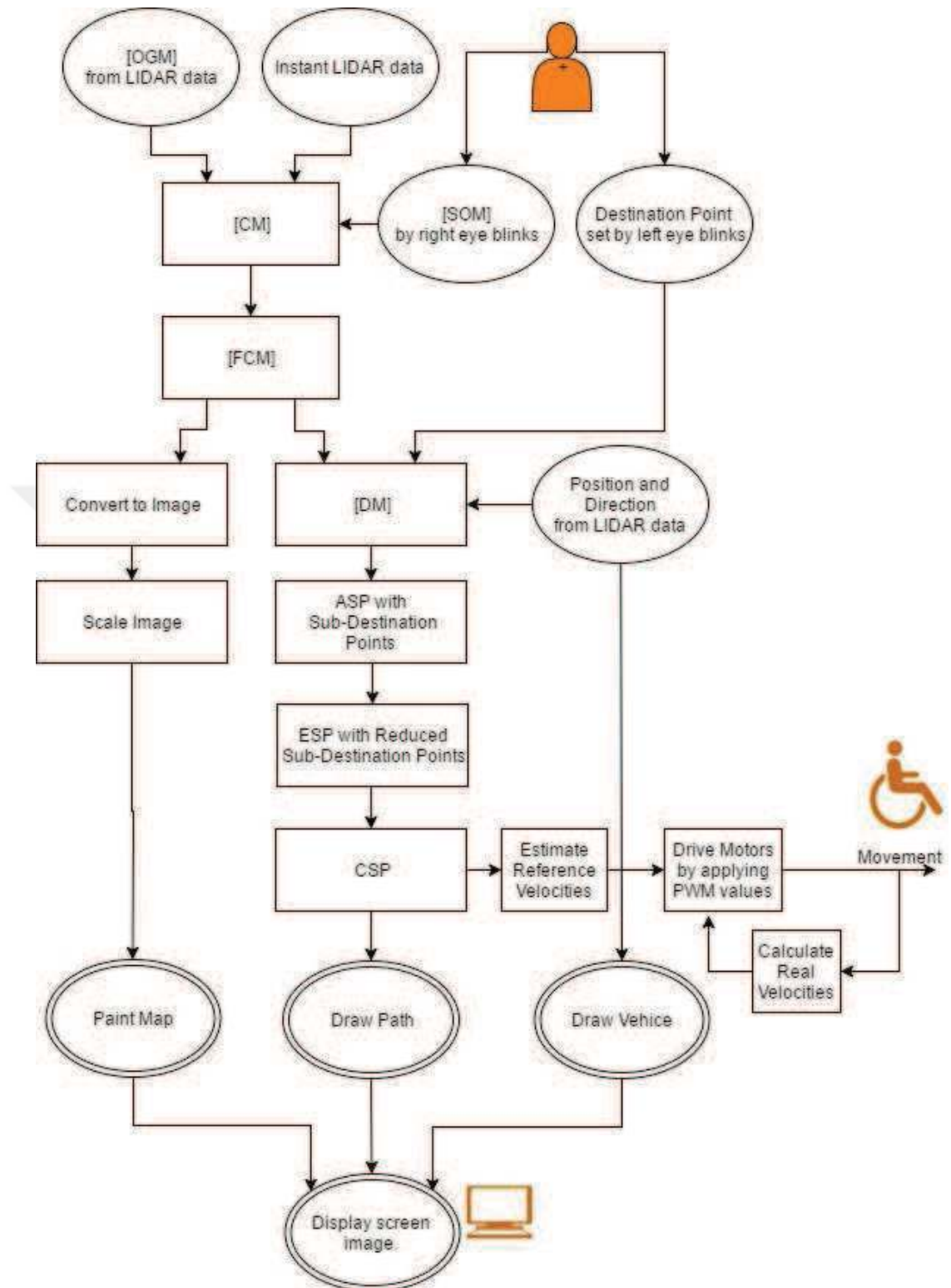


Figure 7.1 General algorithm

Table 7.1 Abbreviations used in Figure 7.1

[OGM]:	Occupancy Grid Matrix	[DM]:	Distance Matrix
[SOM]:	Static Obstacle Matrix	ASP:	Almost Shortest Path
[CM]:	Combined Matrix	ESP:	Exact Shortest Path
[FCM]:	Final Combined Matrix with Forbidden Areas	CSP:	Curved Shortest Path

Firstly, the user marks once static obstacles which are unable to detect by LIDAR to avoid collisions with right eye blinks and [SOM] is obtained. After that [CM] is obtained as a combination result of [OGM] from LIDAR data, instantaneous LIDAR scanning data (for faster obstacle avoidance) and [SOM] in every step of driving as mentioned in Figure 7.2.

[OGM] which has cell numbers in the range of -1 to 100 (-1: unknown areas; 0-49: areas without obstacles; 50-100: areas with obstacles) is obtained as the cumulative result of every instant LIDAR scan data to show the probability of each cells occupancy.

$$\begin{array}{|c|c|c|c|c|c|} \hline -1 & -1 & 0 & 0 & 0 & -1 \\ \hline -1 & -1 & 30 & 80 & 49 & -1 \\ \hline -1 & 20 & 40 & 90 & 70 & -1 \\ \hline -1 & 0 & 30 & 50 & -1 & -1 \\ \hline 0 & 0 & 0 & 20 & 0 & 0 \\ \hline -1 & -1 & 0 & 0 & 0 & 0 \\ \hline \end{array}
 +
 \begin{array}{|c|c|c|c|c|c|} \hline 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}
 =
 \begin{array}{|c|c|c|c|c|c|} \hline 1 & 1 & 5 & 5 & 2 & 1 \\ \hline 1 & 1 & 2 & 5 & 2 & 1 \\ \hline 1 & 2 & 2 & 4 & 4 & 1 \\ \hline 1 & 2 & 2 & 4 & 1 & 1 \\ \hline 2 & 2 & 2 & 2 & 2 & 2 \\ \hline 1 & 1 & 2 & 2 & 2 & 2 \\ \hline \end{array}$$

[OGM] + [SOM] = [CM]

Figure 7.2 Combined Matrix calculation

After these combination, forbidden areas (with number 3) are marked around obstacles (4: Obtained by LIDAR, 5: User Defined); unknown areas are shown with number 1, learned safe areas without obstacles shown with number 2 (Figure 7.3). Afterwards these cells are transformed to map image with pixels for each areas in different colours. These numbers are also useful for detection of changing cells of [FCM] to update only changed pixels of map image instead of updating all pixels for better map refresh rate.

[DM] is obtained from [FCM] to find the shortest path by increasing distance numbers to neighbour cells which don't have forbidden areas (with number 3) and obstacles (with number 4 and 5) from the vehicle position (starting position) to the destination point.

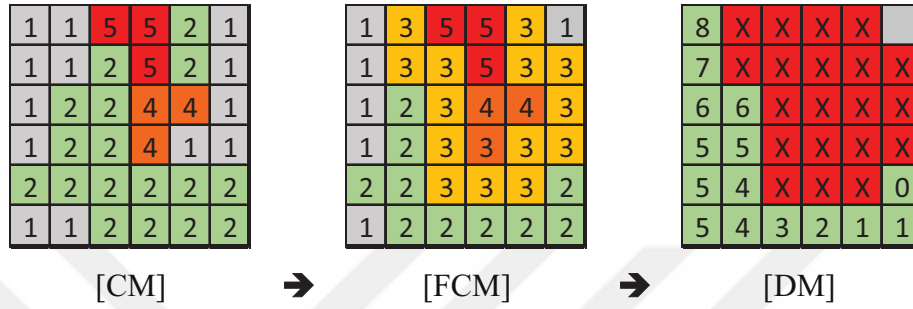


Figure 7.3 Distance Matrix calculation

ASP is drawn from the destination point to the starting position by following decreasing neighbour cell values on DM. Break points of the shortest path where directions change are recorded as sub destination points. After that the path is simplified to get ESP by deleting sub destination points which redundantly extends the path. Finally CSP is obtained, and drawn by using sub destination points and kinematic equations (Figure 7.4). In every step of driving, the most comfortable curved path to next sub destination point is calculated iteratively. The velocity values for each curved path's first step, are applied to motors in every step of driving.

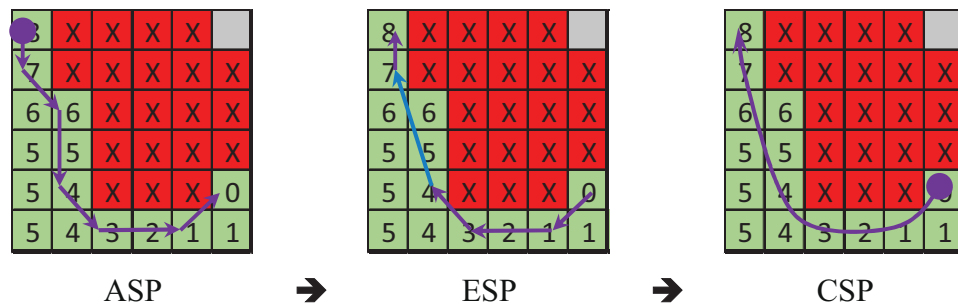


Figure 7.4 Curved Shortest Path drawing

On the other hand, [FCM] values are converted in to image variable with RGB colour values, scaled and painted pixels on the map.

7.2 Curved Shortest Path Estimation and Kinematic Equations

As a result of some wheelchair u-turning experiments, linear velocity (v) of inner (slower) wheel chosen as 5 cm/s and outer (faster) wheel chosen as 30 cm/s for sharpest turning with comfortable driving. After sharpest turn (velocity ratio of 5-30) the vehicle has to be driven with a different velocity ratio to reach its destination after a threshold angle difference (Θ) between the vehicle's direction and destination point. Regarding to this, $\pm 45^\circ$ has been chosen as the threshold angle difference for comfortable driving after some experiments. Velocity of slower wheel is calculated by using equation (7.1) for every difference of angle under the threshold value. Left and right wheel velocities for each case are summarized in Table 7.2.

$$v = 30 - \frac{25\theta}{45} \quad (7.1)$$

Table 7.2 Rear wheel velocities related to θ

θ ($^\circ$)	Velocity of left wheel: v_1 [cm/s]	Velocity of right wheel: v_2 [cm/s]
-180 to -45	5	30
-45 to 0	$30 - \frac{25\theta}{45}$	30
0 to 45	30	$30 - \frac{25\theta}{45}$
45 to 180	30	5
θ : Angle difference between the destination point and the wheelchair centre ($^\circ$)		

Calculation of instant position and orientation changes of the vehicle centre:

Abbreviations used in the following equations are explained in Table 7.3.

Table 7.3 Abbreviations used in equations

r_c	Turning radius of the wheelchair centre	[cm]
r_1	Turning radius of inner wheel	[cm]
L	Distance between rear wheels	[cm]
P	Position of the of the wheelchair centre	[cm, cm]
D	Travelled distance between last two positions	[cm]
H	Linear distance between last two positions	[cm]
v	Linear velocity of the wheelchair centre	[cm/s]
v_1	Linear velocity of left wheel	[cm/s]
v_2	Linear velocity of right wheel	[cm/s]
α_{vehicle}	Heading angle	($^\circ$)
2α	Difference between last two heading angle	($^\circ$)
θ	Angle difference between the destination point and the wheelchair centre	($^\circ$)

The linear velocities of the wheels are known from equation (7.1) and Table 7.2. But instant position and orientation changes of the vehicle centre are not known. Thus, these velocities are used to calculate instant position and orientation changes of the vehicle centre as below. The turning radius (r_c) of the vehicle (Figure 7.5) must be calculated by using equation (7.2)

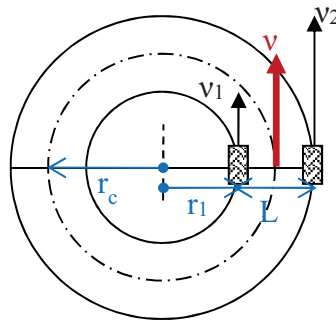


Figure 7.5 Turning radius calculation of the vehicle

$$\frac{v_2}{v_1} = \frac{2\pi(r_1+L)}{2\pi(r_1)} \rightarrow \frac{v_2}{v_1} = 1 + \frac{L}{r_1} \rightarrow r_1 = \frac{Lv_1}{v_2-v_1}$$

$$r_c(v_1, v_2) = r_1 + \frac{L}{2} = \frac{Lv_1}{v_2 - v_1} + \frac{L}{2} \quad (7.2)$$

The change in the vehicle heading angle is illustrated in Figure 7.6.

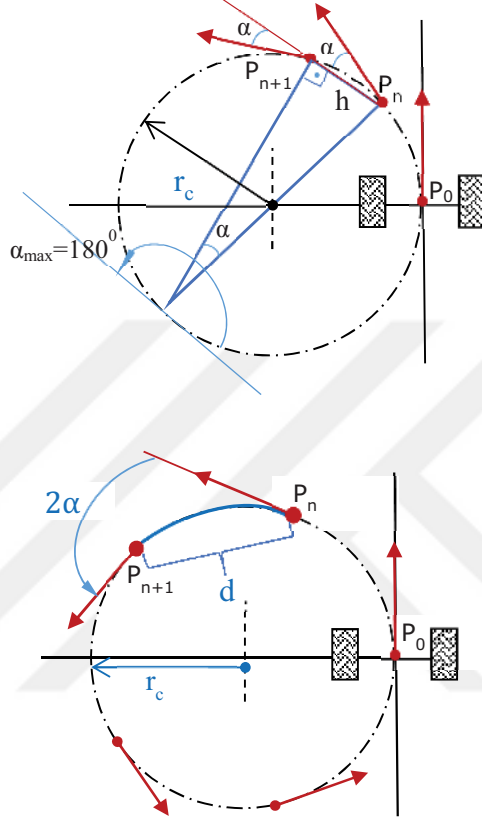


Figure 7.6 The change in the vehicle heading angle

From the geometry,

$$d = \widehat{P_n P_{n+1}} = 5\text{cm (user-defined value)} \quad (7.3)$$

$$\text{and } \frac{\alpha_{\max}}{d_{\max}} = \frac{180}{2\pi r_c} \rightarrow \alpha = \frac{90d}{\pi r_c} \quad (7.4)$$

$$\text{Euclidean distance: } h = \sqrt{\sum_{i=1}^N (P_n - P_{n+1})^2} = |P_n P_{n+1}| = 2r_c \sin \alpha \quad (7.5)$$

By using α and h , the displacement of P_x (Δx) and displacement of P_y (Δy) are found as below:

$$\Delta x = -h \cos(180 - 2\alpha - \alpha_{\text{vehicle}}) \quad (7.5)$$

and

$$\Delta y = h \sin(180 - 2\alpha - \alpha_{\text{vehicle}}) \quad (7.6)$$

New position ($P_{x_{n+1}}, P_{y_{n+1}}$) and new orientation ($\alpha_{\text{vehicle } n+1}$) of the vehicle is illustrated in Figure 7.7.

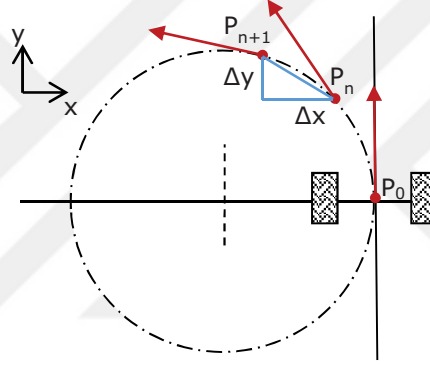


Figure 7.7 Change in the vehicle position

$$\begin{aligned} P_{x_{n+1}} &= P_{x_n} + \Delta x \\ P_{y_{n+1}} &= P_{y_n} + \Delta y \\ \alpha_{\text{vehicle } n+1} &= \alpha_{\text{vehicle } n} + 2\alpha \end{aligned} \quad (7.7)$$

By using these equations, calculated and estimated curved path from the starting point P_0 to destination point P_1 with the starting angle of zero can be obtained as illustrated in Figure 7.8.



Figure 7.8 Estimated curved path

Some examples of calculated paths according to combinations of starting position, starting heading angle and destination position are shown below:

Example 1: Paths of four different starting positions with starting angle 0° to the destination point ($x=0, y=0$) (Figure 7.9)

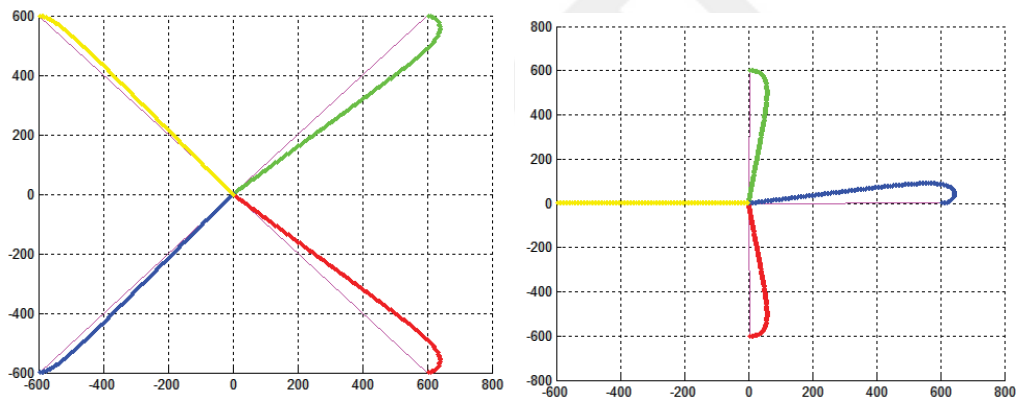


Figure 7.9 Calculated trips for starting angle 0°

Example 2: Paths (round trips) of four different starting positions with starting angle $0^\circ, 90^\circ, 180^\circ$ and 270° to the destination point ($x=0, y=0$) as shown in Figure 7.10.

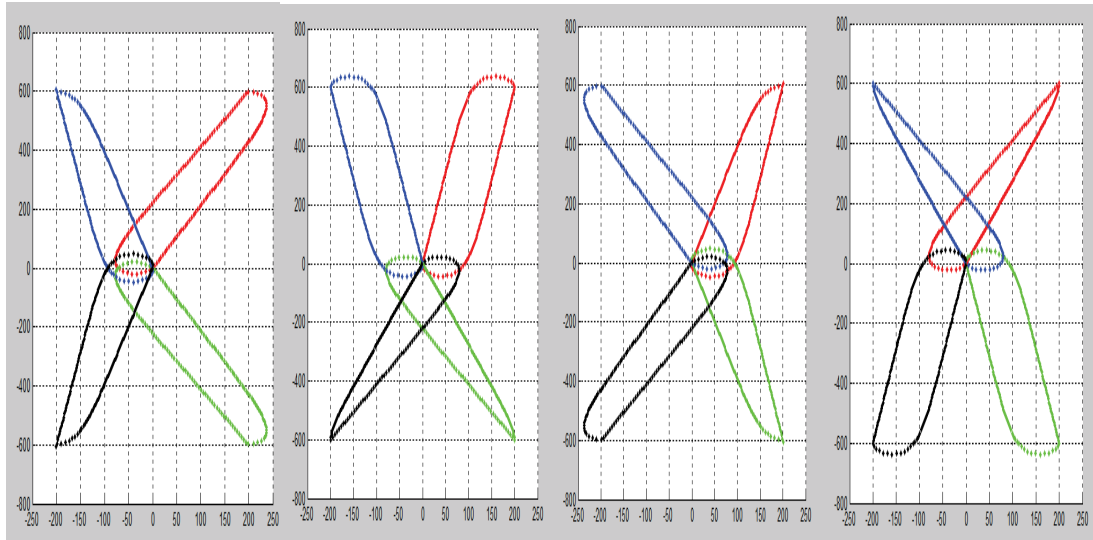


Figure 7.10 Round trips for starting angles a: 0° , b: 90° , c: 180° and d: 270°

Example 3: Calculated paths for all starting angle possibilities ($\theta = 0^\circ$ to 359°) (Figure 7.11)

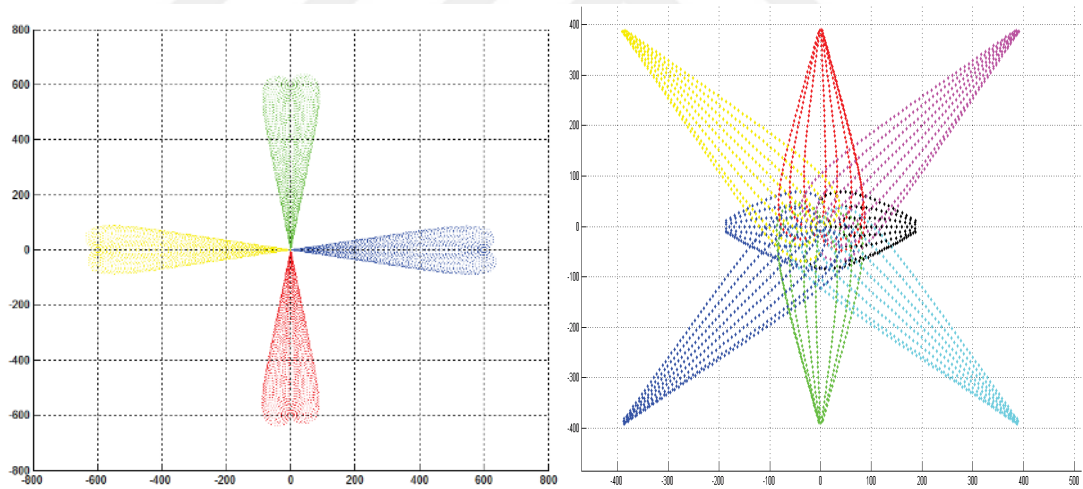


Figure 7.11 Calculated paths for all starting angle possibilities a (left): $x=0, y=0$ for destination position and b (right): $x=0, y=0$ for starting position

After estimation of curved path it has been checked for existing obstacles on it. If there is an obstacle on newly estimated path, then one more step sharper curved path is estimated. It is checked again till to whirl around or a new curved path without obstacle has been found. Thus optimum and most comfortable curved path is followed in every step.

7.3 Real Environment Test

7.3.1 *The First Prototype*

In first autonomous driving tests, the vehicle is gone in a zigzag manner because of the adaptive algorithm which updated in every 250 milliseconds. After some improvements, these errors are pretty much eliminated. Real-time autonomous driving tests with low speeds have been completed with static obstacles (Figure 7.12) and dynamic obstacles (Figure 7.13)

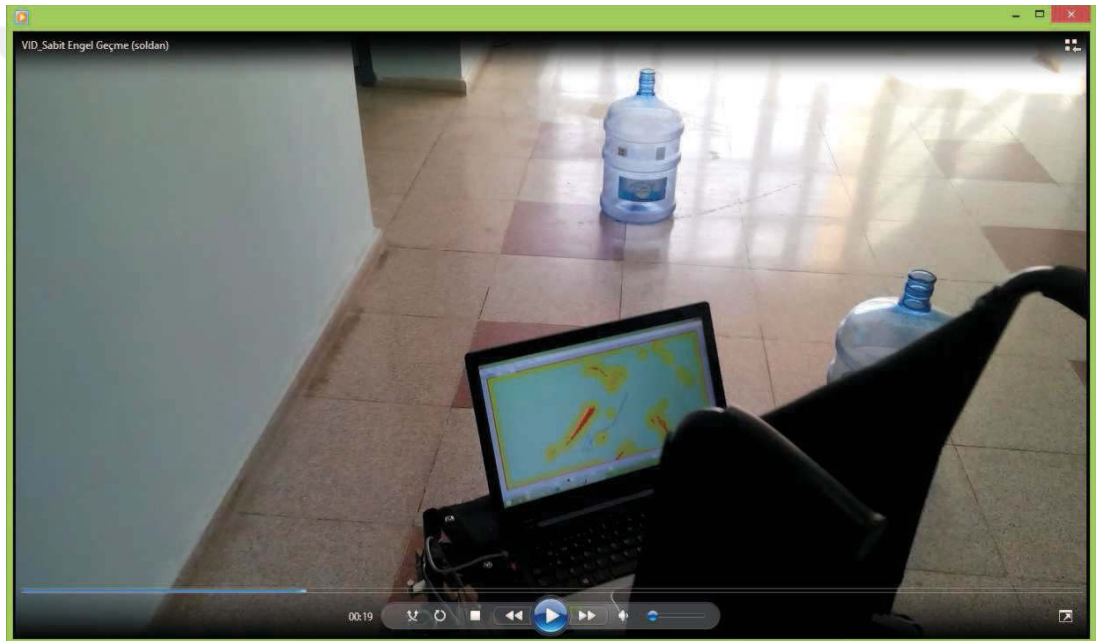


Figure 7.12 Real-time autonomous driving tests with static obstacles (Personal archive, 2016)



Figure 7.13 Real-time autonomous driving tests with dynamic obstacles (Personal archive, 2016)

By using initial map or without initial map, the vehicle can find the destination point by accepting unlearned areas without obstacles. It estimates the shortest paths.

7.3.2 *The Second Prototype*

Eventually, hands-free mouse control program to assign destination points, electronic control and driving system, kinematic equations and navigation algorithm are implemented on a classical battery power wheelchair mechanism. In a given task, estimated and performed paths are drawn on the map. The position errors are compared. Maximum position error of the vehicle centre is decreased 25 cm to 10 cm after optimisation studies as shown in Figure 7.14 and Figure 7.15.

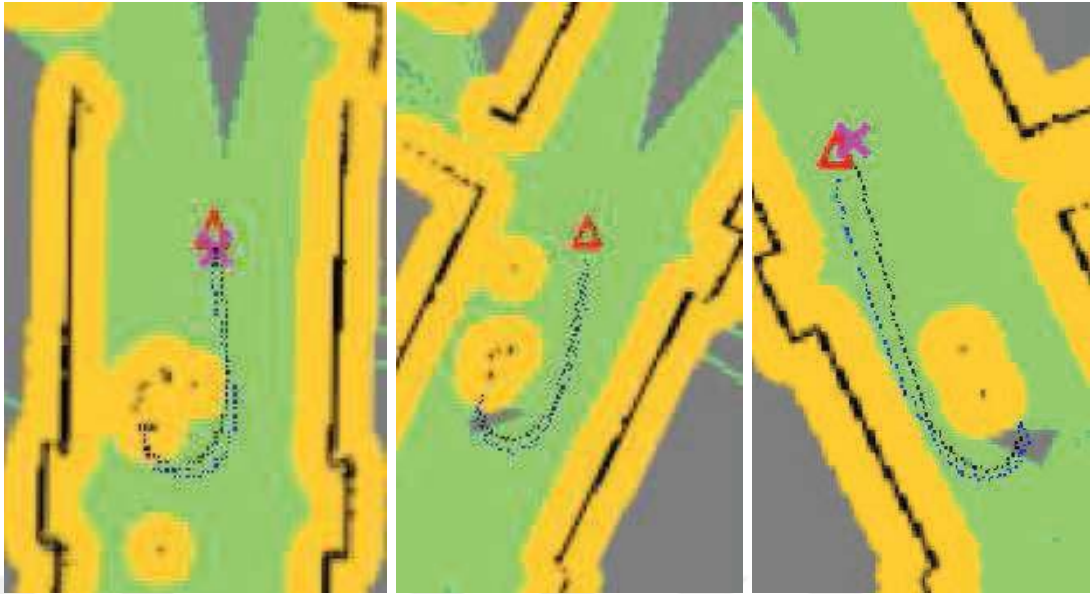
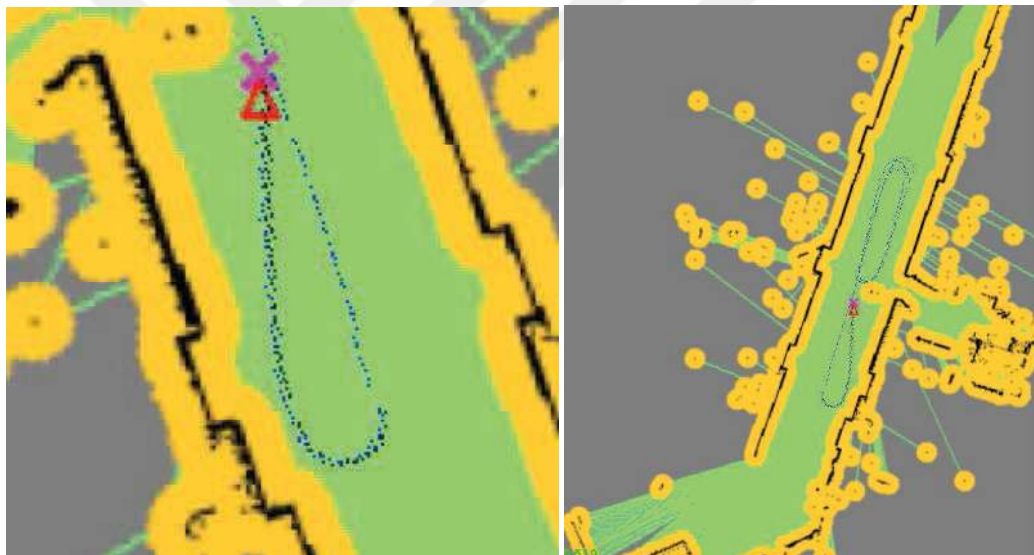


Figure 7.14 Estimated and performed path comprehension on initial practices (black: estimated path, blue: performed path)



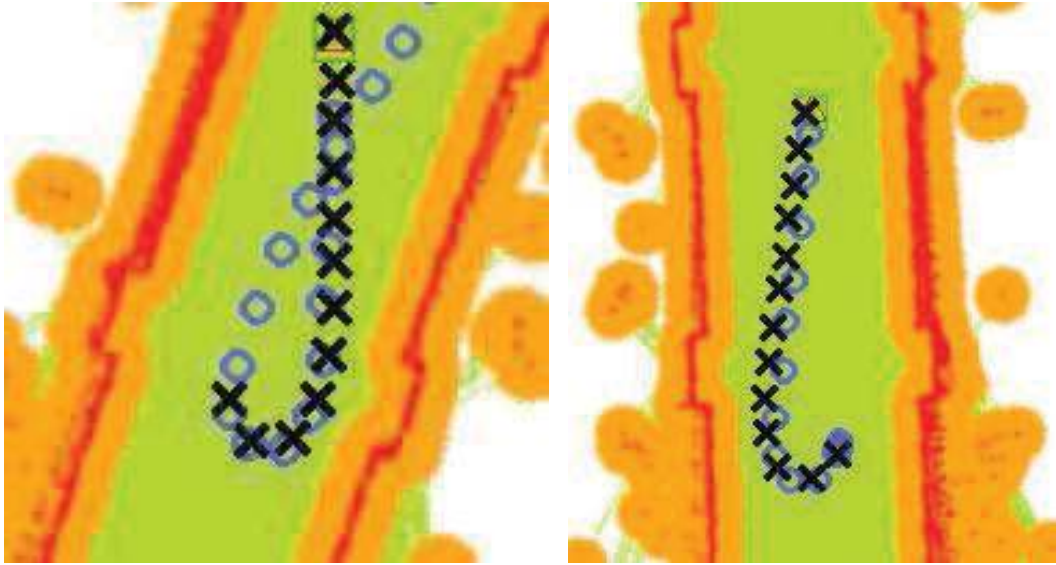


Figure 7.15 Estimated and performed path comprehension on last practices (black: estimated path, blue: performed path)

Additionally, square shaped forbidden areas around the cells which have obstacles are converted to circle shaped forbidden areas. Thus, forbidden areas are marked more effectively. Ability of passing between the narrow obstacles are improved.

Real-time autonomous driving tests have been completed with static and dynamic obstacles to test curved shortest path estimation and following ability of the vehicle as shown in video frames shown in Figure 7.16. Some door passing, collision detection and preventing algorithms are also developed, tested and optimized for safer and better manoeuvring capability.



Figure 7.16 Real-time autonomous driving tests with static and dynamic obstacles (Personal archive, 2016)

CHAPTER EIGHT

CONCLUSIONS

8.1 Overview

The wheeled mobile robots model used in the thesis was originally a wheelchair which has four wheels and whose two wheels are actuated with two DC motors. Then tablet/monitor holder and LIDAR sensor holder parts are integrated. The vehicle is equipped with a driving system with encoders and 24 VDC motors, main control board with ATmega2560 microcontroller, monitor and a system unit with i5 6th generation Intel processor to run high level mathematical operations and algorithms of self-developed autopilot system and to simulate the vehicle movements.

With the studies performed in this thesis, a smart wheelchair with practical and adaptive autonomous navigation system which can be used by disabled people unable to use their hands and feet has been developed and tested successfully in real environment. Destination points have been assigned with just one click on the map of Graphical User Interface (GUI) by using EOG and gyroscope signals of the user. Thus, the commands which have to be given for transportation by the user with EOG signals are reduced and became easier. So a practical solution has been developed.

Smooth and more stable path following accomplished by using the dynamic equations. The desired performance has been obtained on real time obstacle avoidance, adaptive shortest path finding, simultaneous localisation (approximately 10 cm localisation error) and mapping. Working space of the wheelchair can be restricted by drawing on the map GUI to prevent collisions with obstacles which can't be detected with LIDAR sensor like sharp or higher level objects.

The smart wheelchair with autopilot system and special screen keyboard will have positive influence on disabled people who are unable to use their hands and feet. It will make them happier and their life more enjoyable with more self-confidence.

Finally, the obtained prototype vehicle has the following features:

- Clicking the destination point on the screen by EOG device
- Achieving the assigned destination with smartest way
- Can work without an initial map, can save or reload a map.
- Localisation and mapping with LIDAR and encoder.
- Pictures (JPG, BMP, PNG format) can be uploaded as map
- 210 degree sight view for obstacles
- 10485.76 square meter working area (width=102.4 m and height=102.4 m)
- Special screen keyboard for easier PC use and faster writing.

Thanks to these features, EOG controlled smart and autonomous wheelchair is obtained for transportation, sport and rehabilitation needs of disabled people without commanding every step of the vehicle like others.

8.2 Future Works

In addition and for future work, obstacle detection shield around the vehicle by using bump or distance sensors can be integrated to detect dynamic or static obstacles outside the LIDAR sensor view plane. Thus, dynamic obstacles on the back side of the vehicle or static obstacles like tables can be detected without the need to draw as static obstacle on the GUI or the vehicle can be drawn backward.

The developed autopilot system as a part of the thesis can be a basis or can be implemented to many other vehicles apart from wheelchairs like described as following:

- ✓ Autopilot system can be implemented to unmanned security vehicles for factories, housing estates etc. by using camera, temperature, gas, humidity, colour or sound sensor.
- ✓ Autopilot system can be implemented to unmanned material/apparatus transportation vehicle in storerooms, harbours or factories.

- ✓ Autopilot system can be used for unmanned tractors, mineral or mine exploration vehicles or military security vehicles.

EEG signals instead of EOG signals can be also used to assign destination points with future signal processing algorithms.



REFERENCES

- Adelson, M. (2011). *An experimentation and mind-reading application for the Emotiv EPOC*. Two-semester independent work, Princeton University, New Jersey.
- Argall, B. D. (2016). *Modular and adaptive wheelchair automation*. In *Experimental Robotics* (835-848). Springer International Publishing.
- Bakker, T., Asselt, K. V., Bontsema, J., Müller, J., & Straten, G. V. (2010). A path following algorithm for mobile robots. *Autonomous Robots*, 29(1), 85–97.
- Barshan, B., & Durrant-Whyte, H.F. (1995). Inertial navigation systems for mobile robots, *IEEE Transactions on Robotics and Automation*, 11(3).
- Belic, D., & Kunica, Z. (2015). A concept of smart wheelchair. *Annals of the Faculty of Engineering Hunedoara*, 13(1), 37.
- Bertha, F. (2001). *Brainwave responsive wheelchair*. Patent number: US6587713 (B1).
- Cheng, C., Yang, X., Xiaomin, Y., Yongli, D., & Jikui, T. (2012). *Wheelchair controller based on brain wave*. Patent number: CN202794939 (U).
- Cooper, R. A. (2010). Wheeled mobility: current and future developments. *Rehabilitation: Mobility, Exercise & Sports*.
- De Luca, A., Oriolo G., & Venditteli, M. (2002). Control of wheeled mobile robots: an experimental overview, *Lecture Notes in Control and Information Sciences*, 270, 181-226.
- Documentation* (n.d.). Retrieved December 25, 2016, from <http://wiki.ros.org/>

Dong, M., Wei, S., Yuhuan, Z., & Baikun, W. (2008). *Intelligent wheelchair control system based on brain-machine interface and brain-electrical signal processing method thereof*. Patent number: CN101301244 (A).

EMOTIV Epoc+ (n.d.). Retrieved December 20, 2016, from <https://www.emotiv.com/>

Folane, N. R., & Autee, R. M. (2016). *EEG Based Brain Controlled Wheelchair for Physically Challenged People*. International Journal of Innovative Research in Computer and Communication Engineering, ISSN(Online): 2320-9801

Gören, A. (2007). *Controlling a non-holonomic vehicle via artificial neural networks*. PhD Thesis, Dokuz Eylül University, İzmir.

Keij, J. J. A. M. (2003). *Obstacle avoidance for wheeled mobile robotic systems (Literature Exploration)*. Technische Universiteit Eindhoven.

Kucukyildiz, G., Ocak, H., Karakaya, S., & Sayli, O. (2017). Design and Implementation of a Multi Sensor Based Brain Computer Interface for a Robotic Wheelchair. *Journal of Intelligent & Robotic Systems*, 1-17.

Kurtzberg, J. M., & Lew J. S. (2000). *Voice-controlled motorized wheelchair with sensors and displays*. Patent number: US6108592 (A).

LaPlante, M. P. (2003). Demographics of wheeled mobility device users. *Proceedings of the Conference on Space Requirements for Wheeled Mobility*.

Latombe, J. C. (1991). *Robot motion planning*. Massachusetts: Kluwer Academic Publishers.

Muir, P. F., & Neuman, C.P. (1986). Kinematic modelling of wheeled mobile robots. *Tech. report CMU-RI-TR-86-12*. Carnegie Mellon University, Pennsylvania.

- Negenborn, R. (2003). *Robot localization and Kalman Filters on finding your position in a noisy world*. Master Thesis, Utrecht University, Utrecht.
- Nolan, D. A. (1998). *Wheelchair voice control apparatus*, patent number: US 5812978 (A).
- Oschler, R. (2010). *Mind over Matter: Brain Control Interfaces Become a Reality*, Retrieved January 03, 2017, from <http://www.pcmag.com/article2/0,2817,2359074,00.asp>
- Rotary encoder (n.d.). Retrieved December 20, 2016, from https://en.wikipedia.org/wiki/Rotary_encoder
- Roumeliotis, S., & Bekey, G. (2000). Collective localization: a distributed Kalman Filter approach. *In Proc. of the IEEE International Conference on Robotics and Automation*, 2, 1800-1087.
- Schwesinger, D., Shariati, A., Montella, C., & Spletzer, J. (2016). A smart wheelchair ecosystem for autonomous navigation in urban environments. *Autonomous Robots*, 1-20.
- SICK AG Waldkirch (2008). *LMS100/111/120/151 Laser Measurement Systems, Operating Instructions*, Retrieved November 03, 2015, http://lars.mec.ua.pt/public/Tutorials/SickLMS151/OI_LMS100.pdf.
- Siegwart, R., & Nourbakhsh, I.R. (2004). *Introduction to autonomous mobile robots*, Retrieved December 10, 2016, from <http://www.asoodread.com/Pride>.
- Simpson, R. C., LoPresti E. F., & Cooper R. A. (2008). How many people would benefit from a smart wheelchair?. *Journal of Rehabilitation Research and Development*, 45(1), 53–71.

Solanki, K. H., & Pujara, H. (2015). Brainwave Controlled Robot. *International Research Journal of Engineering and Technology (IRJET)*, 2, 609-612.

Stachniss, C., Hahnel, D., & Grisetti, W. B. G. (2005). On actively closing loops in grid-based FastSLAM. *Advanced Robotics*, 19(10), 1059-1079.

Talj, R., Tagne, G., & Charara, A. (2013). Immersion and invariance control for lateral dynamics of autonomous vehicles, with experimental validation, *European Control Conference (ECC 2013)*, pp.968-973. Zurich, Switzerland

Tamura, H., Manabe, T., Goto, T., Yamashita, Y., & Tanno, K. (2010). A study of the electric wheelchair hands-free safety control system using the surface-electromyogram of facial muscles. *Intelligent Robotics and Applications*, 6425, 97-104.

Tatsumi, K. (2002). *Navigation system using brain wave signal and navigation control method*. Patent number: JP2003247847.

Tiwari, K. P., & Dewangan, M. K. K (2013). Voice Controlled Autonomous Wheelchair. *International Journal of Science and Research (IJSR)*. International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064

APPENDICES

A. 1 Nomenclature

[OGM]	Occupancy Grid Matrix
[SOM]	Static Obstacle Matrix
[DM]	Distance Matrix
[CM]	Combined Matrix
[FCM]	Final Combined Matrix with Forbidden Areas
ASP	Almost Shortest Path
ESP	Exact Shortest Path
CSP	Curved Shortest Path
n	number of distance measurements in each scan
i	cell number in a row
j	cell number in a column
C_w	width of a cell
C_h	height of a cell
Δ	angular step of laser beam (0.25, 0.33, 0.5, or 1.0)
D	Measured distance by LIDAR
Wall	Defines the cell as obstacle (wall)
$\{C_x, C_y\}$	Center of the vehicle
$\{M_x, M_y\}$	Detected obstacle coordinates
r_c	Turning radius of the wheelchair centre [cm]
r_l	Turning radius of inner wheel [cm]
L	Distance between rear wheels [cm]
P	Position of the of the wheelchair centre [cm, cm]
D	Travelled distance between last two positions [cm]
H	Linear distance between last two positions [cm]
v	Linear velocity of the wheelchair centre [cm/s]
v_l	Linear velocity of left wheel [cm/s]
v_r	Linear velocity of right wheel [cm/s]
α_{vehicle}	Heading angle ($^\circ$)
2α	Difference between last two heading angle ($^\circ$)

θ	Angle difference between destination point and wheelchair centre (o)
t	template
f	last actively buffered signal
σf	standard deviation of actively buffered signal
σt	standard deviation of the template
$t(x,y)$	cross-correlation of a template
$f(x,y)$	actively buffered signal
n	number of pixels in $t(x,y)$ or $f(x,y)$
$\bar{f}$	the average of template (f)
$\bar{t}$	the average of last actively buffered signal (t)

