Monitoring Erratic Driving Behaviour caused by Vehicle Overtaking using Off-the-shelf Technologies

El Hosin Gazali

A dissertation submitted to the University of Dublin,
in partial fulfilment of the requirements for the degree of
Master of Science in Computer Science

October 2010

DECLARATION

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this or any other University, and that, unless otherwise stated, it is entirely my own work.

_____

El Hosin Gazali

29 October 2010

Permission to lend and/or copy

I, the undersigned, agree that Trinity College Library may lend or copy this dissertation upon request.

Signed: _____
El Hosin Gazali

29 October 2010

ACKNOWLEDGEMENTS

I would like to dedicate this dissertation to my dearest and beloved brother Dr. Muftah Gazali and his friend and supervisor Dr. Bunah who both died in car accidents.

I would sincerely like to thank my supervisor Professor Vinny Cahill for his help, support and technical expertise throughout all stages of the whole dissertation.

I would like to express my gratitude and appreciation to Dr. Siobhán Clarke for all her support and encouragement throughout the year. I am really grateful.

I would like to take this opportunity to deeply thank Miss Anne Holland for all her support and providing me with all the time and strength that I needed to make this dissertation and the whole masters happen.

Last but not least, I would like to thank my parents and all my family for helping me and giving me all the inspiration to make it this far.

Abstract

Previous studies and attempts of investigating and developing systems for monitoring erratic driving behaviour have been mainly based on using fully dedicated sensor devices such as magnetic sensors that are usually accompanied with visual sensors such as cameras; however none of these studies were actually based solely on using off-the-shelf technologies such as mobile phone built-in sensors. In this dissertation we present a study that investigates the possibility of using currently-available off-the-shelf sensor technologies to detect and report erratic driver behaviour caused by vehicle overtaking in Irish main roads and motorways and develops a proof-of-concept implementation of a solution.

The sensor-based technologies that we are using in our solution are all mobile phone built-in technologies, namely the orientation sensor and the GPS receiver. The system is also based on a number of techniques that we developed using Neural Networks, GIS Tools and map matching techniques. We primarily use the orientation sensor to detect the movement of the vehicle in the left, right and the forward directions. The system uses GPS for obtaining the coordinate location of the vehicle and its speed as it moves in time and relays this information to a GIS system on the application server to obtain real time information of this location.

The system is divided into two main parts, the mobile phone part, with all its built-in sensors, and the application server part for digital map matching and final decision-making on all detected overtaking patterns. We also define what Erratic Driving is and show what the possible driving scenarios and patterns that a vehicle can make to overtake another vehicle, and focus our attention on all the possibilities of how these scenarios and patterns could take place, and how we actually represent them digitally in our system. In addition to that we suggest a technique that we utilise to identify any overtaking patterns and distinguish them from other patterns through the use of Neural Networks. We then present a map matching technique of how to match and confirm these overtaking patterns on an actual road network and pinpoint their location through the use of GIS tools and geometric techniques.

Table of Contents

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Motivation

Car accidents happen all the time and as a result of which people lose their lives. In many cases there are no witnesses to these accidents. Driving is a dangerous activity but we tend not to consider it as such. It is very possible to make driving cars safer if the right monitoring system was put in place. According to car-accidents.com [7] there were nearly 6,420,000 auto accidents in the United States in 2008. The financial cost of these car accidents was more than 230 billion dollars. 2.9 million people were injured and 42,636 people killed. About 115 people die every day in vehicle crashes in the United States, one death every 13 minutes. Worldwide it is estimated that over one million people are killed a year and over fifty million more are injured in automobile collisions [8]. Below is a breakdown of the vehicle-based fatalities in the EU and the cost incurred by these fatalities in a single EU country.

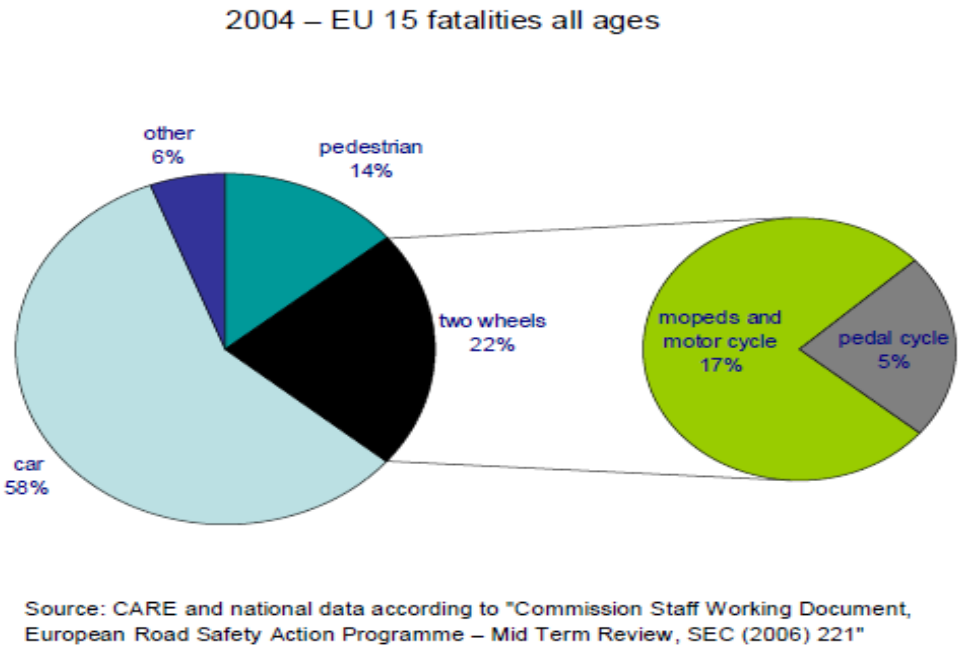


Figure 1: Vehicle-based fatalities in EU and their cost

The traditional methods of determining the causes of accidents, the speed of the vehicle before the accident happens and so on are based on studying the physical appearance of the vehicle after the accident happens, which could be caused by factors other than the accident itself. So there is a need for preventing the causes of these vehicle accidents and

monitor the driver's behaviour without any heavy reliance on the physical appearance of the vehicle after the accident, or relying on recording the position of the gas pedal and whether the brakes were actually used in the last second before the event or not. It would instead be preferable to rely on a source that is located beyond the vehicle's environment, like a remote server to which a device, such as a mobile phone, in the car could report to on a regular basis, firstly to monitor the drivers behaviour and secondly to keep a record externally for future and warning references.

Our proposed solution is a system which uses mobile phone built-in GPS and orientation sensor capabilities to monitor, detect and record erratic driving behaviour caused by a vehicle overtaking. The system reports all the detected and monitored data to a server that is capable of making a decision whether this data contains any erratic driving behaviour or not.

We believe this system can save many lives and make life easier for individuals, parents and governments. With the right regulations in place, this system could be used for convict any irresponsible drivers when accidents happen and reward the good ones, all this in an attempt to encourage people to change their attitude towards driving.

Definition of Erratic Driving Behaviour

Erratic driving behaviour events can be divided into two classes

1. The interaction between the vehicle and the road environment
2. The interaction between the vehicle and nearby vehicles.

These two classes of driving events may occur simultaneously and could lead to certain serious traffic situations. Our proposed system will analyse these two kinds of events and will determine any dangerous situations from the data collected by various sensors.

Any of the following driving scenarios is considered erratic and dangerous depending on the road speed limit and other factors.

1. Excessive Speeding
2. Overtaking
3. Aggressive turning
4. Unnecessary Weaving
5. Hard braking

6. Hard shock or flipping which is an out-of-the-ordinary movement of the vehicle when an accident occurs.

Objective

The main objective of this dissertation is to present a study that will investigate the possibility of using currently-available off-the-shelf sensor technologies to detect and report erratic driver behaviour patterns caused by vehicle overtaking and develop a proof-of-concept implementation of a solution, as well as providing a solid basis for any future work in the field.

Document Structure

This is a short description of the document, giving the overall layout of the dissertation and a short description of issues addressed in each chapter.

Introduction

It gives an overview of the subject and the motivation behind the development of the system. It also defines what erratic driving behaviour is and outlines all the driving scenarios and events that are considered erratic.

State of Art

It describes various existing technologies and approaches that are used in systems with indirect similarities to our system.

Design

Gives an overview of the overall system architecture and describes its main components and their structure as well as describes algorithms designed for our solution.

Implementation

Gives implementation details of the system, including the software, tools and techniques used in the system and how all the solutions are achieved.

Evaluation

Discusses the results of the tests carried out in order to identify the accuracy and the performance of the system as well as presents conclusions concerning further development.

Conclusions

Concludes the dissertation giving an objective opinion on the implementation of the system and outlines possible improvements and discusses future work on the solution.

CHAPTER 2: STATE OF THE ART

Sensors based Driving Behaviour Detection

A number of studies have been carried out in the area of detecting driving behaviour. In addition a number of techniques and methods have been suggested for detecting these behaviours, such as excessive speeding, hard turning and so on. Baldwin, Duncan, and West [1] have proposed a driver monitoring system as shown in figure 2 that detects over speeding and hard turns. The system is composed of a GPS receiver, a two-axis accelerometer and three video cameras. Their system was intended to be a research tool but what they concluded through their study and experiments is very important and closely related to our project as it showed how a driver monitoring system could be achieved and built through a combination of different sensors. They carried out an experiment to evaluate the driving performance of 1500 drivers, each for a 1-week period, several times over the course of a multi-year study. The results showed that they managed to detect a number of driving events, such as the acceleration of the vehicle, the heading and the coordinate location of the vehicle.

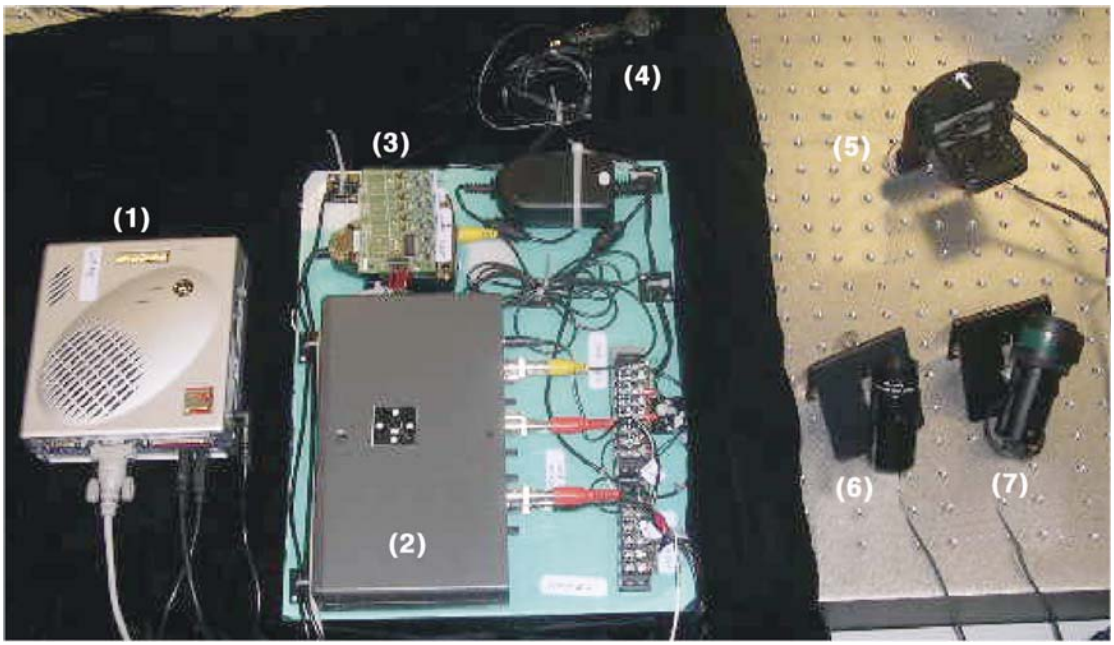


Figure 2: Driver Monitoring System [1]

Mitrovic in his study [2] presented a method for driving event recognition using Hidden Markov Models (HMMs). The method/framework he suggested, as shown in figure 3, is based on training the system to learn driving patterns such as when to do a right turn, when

to do a left turn and when to drive forward. These driving patterns will be later used to predict future driving events and issue warnings to drivers. He also believes that the complexity and cost of components used in building this kind of systems should be kept to a minimum. He developed a system consisting of accelerometers, gyroscopes, and a GPS receiver. The data is collected and processed to remove noise and reduce the data volume. A vector quantization method is used to translate analog signal sensors into discrete symbols from a finite set. After a number of tests, the vector quantization codebook size of 16 was selected as a compromise between the quantization error and HMM recognition performances. For driving event recognition, he used discrete HMMs as they better capture the dynamic characteristics of data than a general HMM. For each driving event, he selected the most appropriate number of states and trained the HMM with selected training data. He then evaluated each observation sequence by all HMMs. The highest probability is used to determine which driving event was recognized. This approach further improved the recognition rate of his system. According to him only 4 of 238 (1.7%) test events were wrongly recognized. The probabilities for correct event types were much higher than for incorrect event types, proving that his proposed method had provided encouraging results. According to him the program he developed for training and evaluation of HMMs was also successfully used for recognition of sports events from video streams.

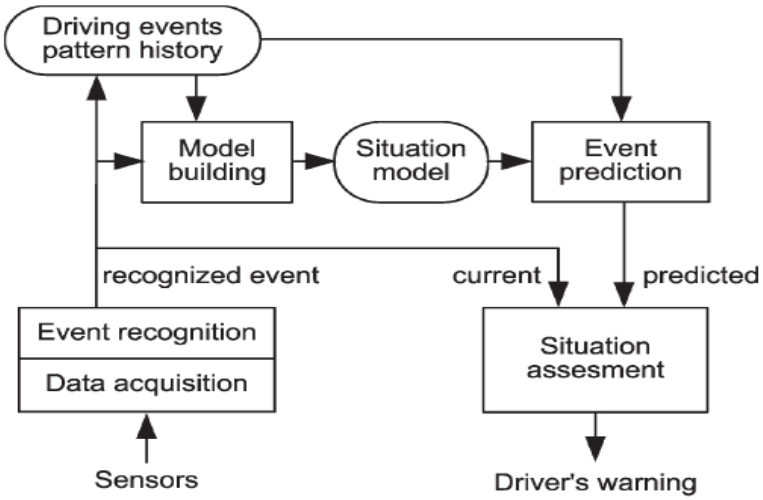


Figure 3: Driving patterns recognition system [2]

Another study carried out by Imkamon, Saensom, Tangamchit and Pongpaibool [3], describes a system, as shown in figure 4, that can record driving events and detect erratic driving behaviours using three sensors, namely an Engine Control Unit (ECU), a 3-axis

accelerometer and a camera. The three sensors are meant to represent the vehicle, the passenger and the driver's perspective. The system combines the data from these three sensors with a fuzzy inference system to identify an event of erratic driving. The output is passed through a driving risk threshold that is divided into three levels from 1-3, where 1 indicates a normal/safe level of driving. Their test results showed that their system can perform well compared with human opinions but still has a limitation of day-time operation due to constraints from the image processing algorithm they used.

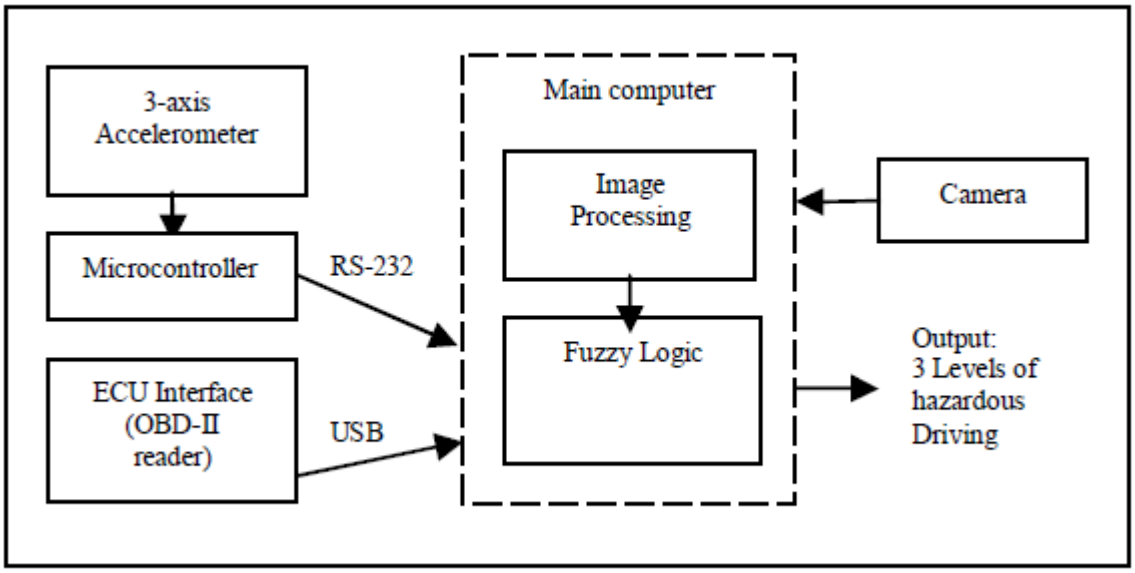


Figure 4: Hazardous Driving Detection System [3]

Another system proposed in [4] by Wang and Gao employs a Fuzzy inference system to assist in multi-sensor data fusion for detecting driving behaviour. This system does not use cameras but instead relies on using accelerometers, gyroscopes and a digital compass. The compass and gyroscope are used to provide absolute heading information relative to the magnetic north without time-accumulated errors. When the vehicle is static, the accelerometer measurement is used to directly derive pitch and roll angles without time accumulated errors. Once the motion type of the vehicle is identified, the most suitable sensor can be used to improve attitude estimation and to control the estimation error.

A system that was based on using built-in vehicle capabilities is proposed in [5] by Ngo Chon Chet. The system is termed a black box, is mainly hardware based and consists of three main parts, namely a speed sensor, a micro controller and an alarm system. The main function of

this black box is to monitor the vehicle speed and issue warnings when the speed limit is exceeded. The input to the black box is the Vehicle Speed Signal (VSS) used by the vehicle built-in speedometer. The system was designed using Maxplus-I1 software and Programmable Logic Device (PLD). The whole black box system was developed and tested using a signal generator in laboratory. Since the system is designed with digital logic and is based on programmable logic device, the output of the black box is supposed to exhibit a quick response to the changes in input signal but no experimental results were presented in the study to show that.

Another black-box style system is proposed in [6] by Lee, Jung and Lim. It consists of a processor, memory, I/O peripheral and several interfaces including a camera, accelerometer and GPS. The input to this system is the visual information from the cameras at the front and at the rear side of the vehicle as well as other driving data such as speed, braking events, seat belt status, and steering performance. The system relies on GPS for recording the location of these events on the road and saves all this data to an external memory for further processing and analysis.

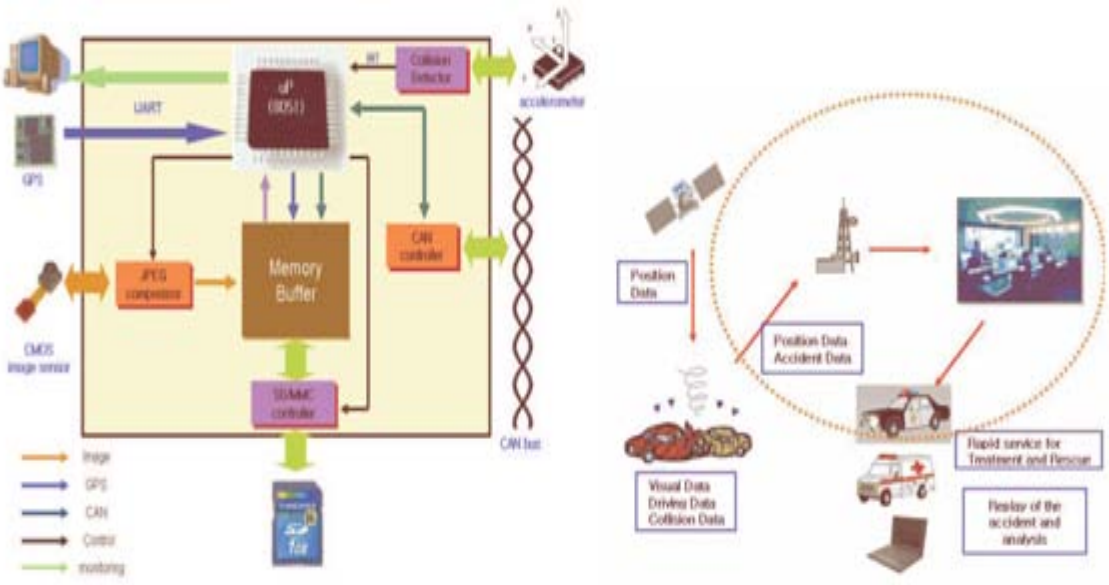


Figure 5: In-vehicle Network based Black Box system [6]

The Overtaking problem

Overtaking is the act of driving around another slower vehicle on a road. Overtaking is considered to be a risk factor and dangerous driving subtask. According to the study in [12]

overtaking accounts for 4 to 10% of all crashes, this study focuses on analysing the overtaking patterns and aims at acquiring deep insight into the overtaking behaviour and collecting enough information about the overtaking process. Through experimentation and the data collected, the study was able to show that the average time that a vehicle takes to overtake another vehicle is 8 sec and a minimum of 6 sec and a maximum of 18 seconds. The study also showed that 80% of the overtaking vehicles were exceeding the speed limit when the operation has been finished.

Neural Networks

The Artificial Neural Networks (ANN) [13] is inspired by the way the brain works. It is an information processing system that is composed of large number of neurons. These neurons are highly interconnected processing elements that make up the core structure of the ANNs, and works collaboratively to solve a certain problem. ANNs are like people they learn by example and achieve their objectivity through a learning process. They are mainly designed and configured for specific applications, such as pattern recognition and data classification where they have to be trained to achieve their objective.

The importance of Neural Networks

The importance of Neural networks lies in its ability to learn and become an expert in deriving meanings, extract patterns and detect trends that are too complex to be extracted and detected by either human or normal software logic.
Other main advantages of using ANNs include:

1. **Adaptive learning**

   ANNs are able to achieve a certain objective through the data provided during the training process.

2. **Self-Organisation**

   ANN can organise itself and create its own representation of the information received during the learning process.

3. **Real Time Operation**

ANNs have the ability to carry out their operations in real time and in parallel. Hardware manufacturers can design their hardware to take advantage of this ability.

4. **Fault Tolerance via Redundant Information Coding**

   ANNs can retain some of their computational abilities and still able to perform even during partial or major network damage or destruction.

The difference between Neural Networks and conventional computers

In order for conventional computers to solve a problem a set of instructions have to be known in advance, and unless these instructions are known the conventional computer is unable to solve the problem. This limits the conventional computers capabilities to solve any problem that we don't already understand or know how to solve. This is different than how neural networks work. Neural networks are able to solve problems that we don't already know or understand; they process information in a similar way the human brain does. Neural networks cannot be programmed in order to solve a problem but instead they have to learn by example and be well trained on how to solve the problem. The example used to train the neural network must be selected carefully or otherwise the network might be functioning incorrectly and gives faulty results.

The only disadvantage of using neural networks over conventional computing is because of the fact that the network can find out how to solve the problem automatically by itself; this can restrict us from predicting the operation of the network. The conventional algorithmic computers on the other hand are totally predictable and if anything goes wrong we can easily know if it is either due to software or hardware problem. The reason that conventional computers are predictable is because they solve kind of problems that we already know and understand and we provide them with a set of instructions which will then be converted into a high level programming language and into a code that the machine understands.

Conventional algorithmic computers and neural networks complement each other and there are certain tasks that are suited for one and not suited for the other. Both approaches could

be combined to achieve a large number of tasks. Normally conventional computers are used to supervise the neural networks.

How the Human Brain Learns

The following is a quote of how the brain works.

"The human brain consists of a large number (more than a billion) of neural cells that process information. Each cell works like a simple processor and only the massive interaction between all cells and their parallel processing makes the brain's abilities possible. As the figure indicates, a neuron consists of a core, dendrites for incoming information and an axon with dendrites for outgoing information that is passed to connected neurons. Information is transported between neurons in form of electrical stimulations along the dendrites. Incoming information that reaches the neuron's dendrites is added up and then delivered along the neuron's axon to the dendrites at its end, where the information is passed to other neurons if the stimulation has exceeded a certain threshold. In this case, the neuron is said to be *activated*. If the incoming stimulation had been too low, the information will not be transported any further. In this case, the neuron is said to be *inhibited*. The connections between the neurons are *adaptive*, what means that the connection structure is changing dynamically. It is commonly acknowledged that the learning ability of the human brain is based on this adaptation."[32]

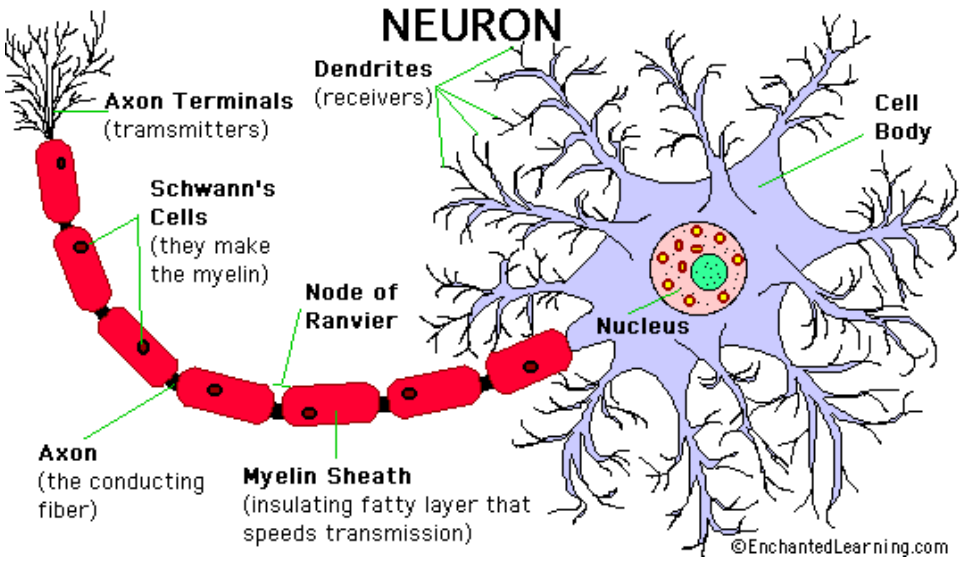


Figure 6: Human neuron structure

From Human Neurones to Artificial Neurones

The computer power and the knowledge we currently have about how the human brain works are still limited, hence we can only create models that approximately imitate the real neurons and design computer programs that simulate them with deduced features.

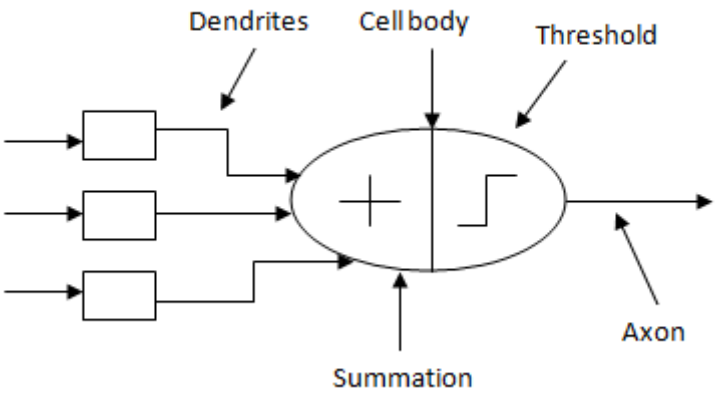


Figure 7: Artificial neuron

A simple neuron

The simplest form on an artificial neuron is a device with a number of inputs and one output that has two modes of operation:

(1) Training Mode

The neuron is trained to go on or off depending on the input pattern

(2) Using Mode

The associated output of the taught input becomes the current output when a taught input is detected.

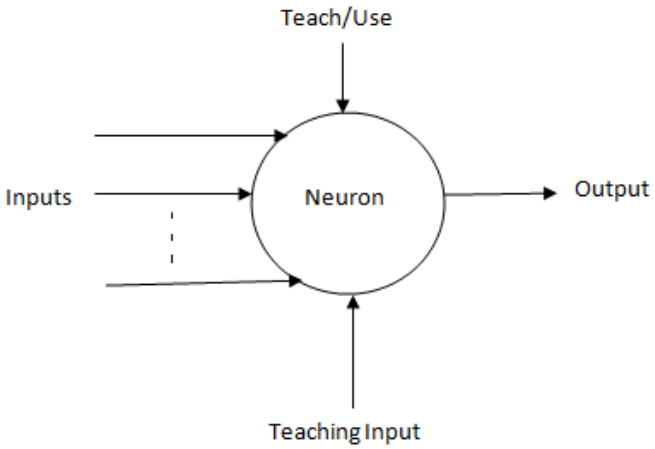


Figure 8: Simple neuron

Pattern Recognition

One of the important applications of neural networks is the pattern recognition using feed-forward neural network where the network is trained to recognise the input pattern and tries to output the expected output. One powerful feature of the neural networks is that it can still give an output even if the input pattern has no associated output with it that the network wasn't trained to recognise, the network, in this case, chooses an output that corresponds to a taught input pattern that is least different from the given pattern.

Architecture of neural networks

Feed-forward networks

Feed-forward neural networks are straight forward networks with the signal travel one way from input to output and there is no feedback. Their main use is in pattern recognition applications. They are also known as bottom-up or top-down networks.

Feedback networks

Feedback neural networks are networks with the signal travelling in both directions, they are very powerful networks and their architecture can get very complicated. Feedback networks are dynamic in nature and their state changes continuously until they reach a steady state point, and they remain in this steady state point until a new one needs to be found or the input changes. They are also known as interactive or recurrent networks.

Network layers

The most common type of ANNs is typically consists of three layers of neurons, the input layer, the hidden layer and the output layer (see figure 9).

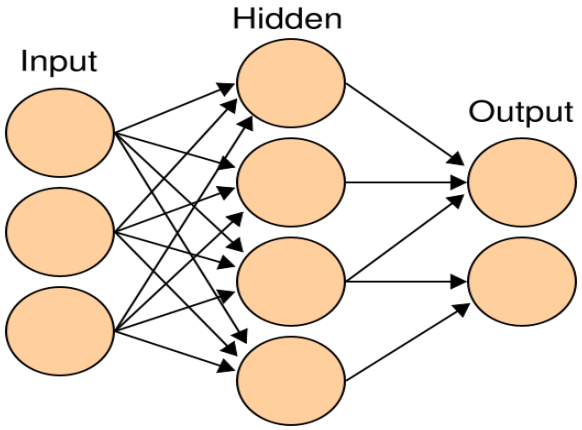


Figure 9: Multi-layer neural network

The input to the input layer represents the raw data or the information that is fed into the network. The weights between the hidden layer and the input layer help in determining the activity of every hidden neuron in the hidden layer. These weights, together with the hidden neurons activities determine the behaviour of the output layer.

The weights between the input layer and the hidden layers can determine when each hidden neuron should be active and by modifying them the network can choose what to represent.

In most cases we only need a single-layer network as it provides more computational power than the multi-layer ones. The reason behind that is that all the input neurons are connected directly to the output neurons and there is no need for the hidden layer.

The Learning Process

All learning methods used for adaptive neural networks can be classified into two major categories:

- **Supervised learning** The aim in supervised learning is to minimise the error between the calculated and the desired output values, we provide the network with a set of inputs and tell the network of the desired output (the teacher signal), the network keeps changing the weights of the hidden layers until a minimum error between the desired and calculated output values is achieved. A well known method that is used in supervised learning is the least mean square convergence.

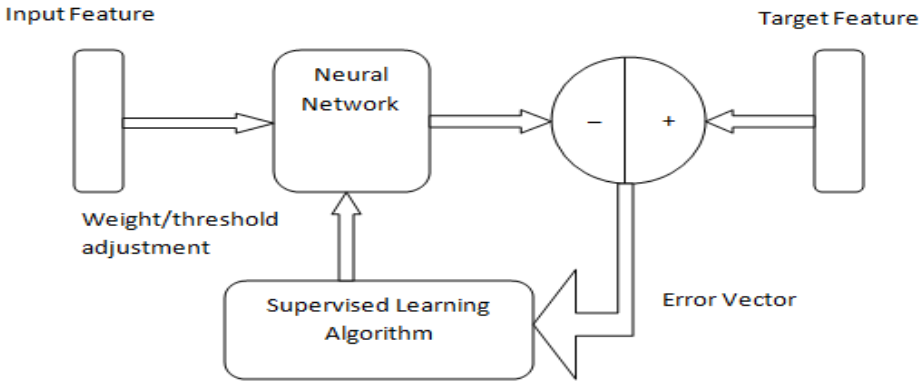


Figure 10: Supervised learning neural network

- **Unsupervised learning** The training set includes input training set only and the network is trained without any teacher signal. The network is self-organising and learns from collected experience gained from previous training patterns.

Vehicle driving system based on Neural Networks

The system in [14] proposes a mechanism for intelligent neural network (MIND) based driving system that is capable of identifying curves in the road and obstacles in its path and takes appropriate action. MIND is an advanced automated transport system that uses multi-layer-forward neural networks with back propagation and other technologies such as Global Positioning System (GPS), Geographic Information System (GIS) and laser ranging. MIND is able to guide a vehicle, after being well trained, through a hostile and unfamiliar domain, and able to negotiate turns and implement lane-changing and overtaking. The system is domain independent, uses less processing overhead and more functionality, comparing to other systems that don't use neural networks.

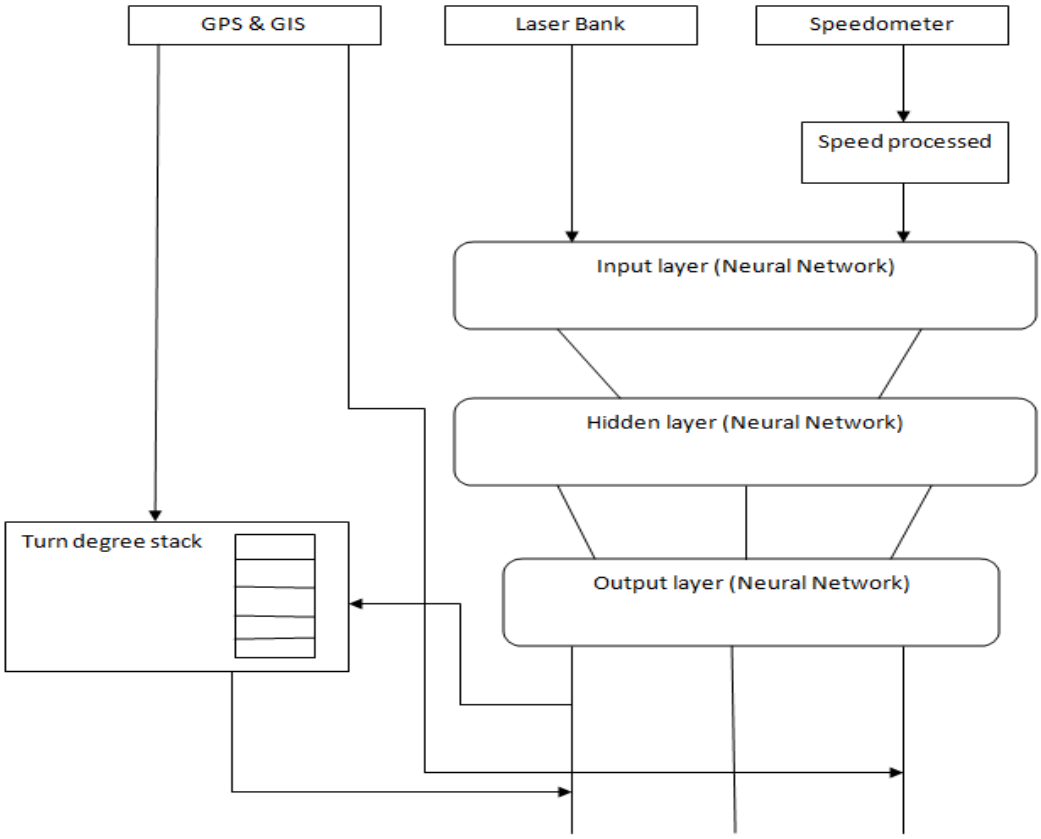


Figure 11: Neural network driving based system

The system use GPS for obtaining the coordinate location of the vehicle on the road and relay this information to the GIS to obtain a real time information based on this location, information such as the shape of the road location and the road angle. The use of neural networks in MIND is mainly to guide the vehicle through the right direction at the appropriate speed. The input to the neural network comprises of the speedometer reading and the information collected from the laser beams that are necessary for detecting and avoiding all sort of obstacles in front of the system. The output of MIND on the other hand comprises of the steering direction, accelerator pressure and brakes pressure.

Map Matching

Map matching techniques and algorithms can be categorised into four main groups, geometric based algorithm [15], topological based algorithms [16] and other methods.

The geometric based map matching methods are methods that mainly depend on the geometric shape of the road. On the other hand the topological methods are based on the use of the topology of the map and the connectivity of the road information. The probabilistic method is based on the use of probabilities in determining the exact matched point on the road. The other methods that are used in map matching are methods that utilise the use of Kalman Filters [17], fuzzy logic [18] and Hidden Markov Models [19, 20]; these methods are more refined methods and used in applications with high map matching accuracy requirements.

The Algorithms of map matching techniques can also be categorised into two main categories when matching the trajectories:

- Local Algorithms

  Used to find the best matched local point based on orientation and similarities. The algorithm in [21] uses a constant-depth recursive look-ahead strategy that is based on locally matching geometries. The algorithm proposed in [22] uses the Dijkstra algorithm to find the shortest path on local free space graph. Another algorithm described in [23] uses road segment-based matching algorithm for matching high-confidence segments first and then match the low-confidence segments using previous matched segments. The local algorithms in general are used in online map

matching applications as they perform better and faster than the Global ones especially when the sampling rate of the location points is high.

- Global Algorithms

  They are usually used in offline map matching applications as they require the overall track trajectory information. Most of the global algorithms applications are based on the uses Frechet Distance [26] or Weak Frechet distance. The paper in [24] proposed an algorithm that applies parametric search over all critical values and then solve the problem of decision making by finding a path in the free space from the lower left corner to the upper right corner. A more refined version is proposed in paper [21] by using the average Frechet distance. An algorithm that is similar to the average Frechet is proposed in [25] that uses a weighted graph representation of the road network.

GeoTools

GeoTools is an open source Java GIS toolkit that consists of a collection of implementations of many Open Geospatial Consortium (OGC) specifications and the GeoAPI project. It is organised into specialised modules and provides an architecture that can be easily extended to incorporate additional functionality [27].

Among the most important features of GeoTools is the support of OGC grid coverage implementation and coordinate reference system and transformation. Another important feature is the support for Java Topology Suite (JTS) and the support of OGC Filter Encoding. GeoTools also support different type of data formats, such as Vector format and Raster format.

CHAPTER 3: DESIGN

Overview of the system architecture and functionality

The system consists of several parts that have different functionalities and carry out different roles in the overall design. The system is divided into two main tiers, the client tier and the server tier. The client tier consists of the phone and its built-in GPS and orientation sensor. The server tier consists of the application server along with all the GIS and digital map matching tools.

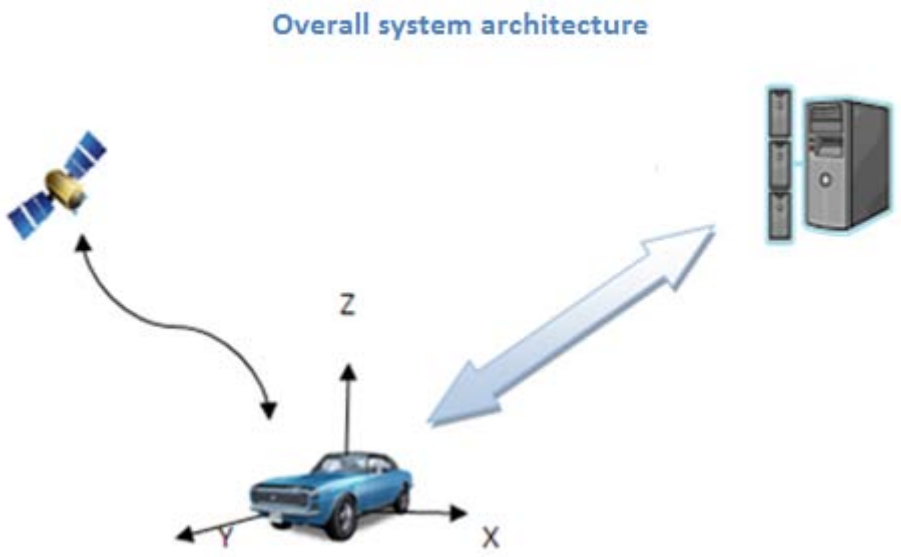


Figure 12: Overall system architecture

The system main designed functionality is to detect any overtaking manoeuvre sequences and send these sequences for further analysis and confirmation to the server along with the GPS location data of these sequences. The server has access to a digital map of the Irish roads and equipped with all the necessary tools for map matching of all the GPS data, received from the phone, on actual road network of the Irish roads.

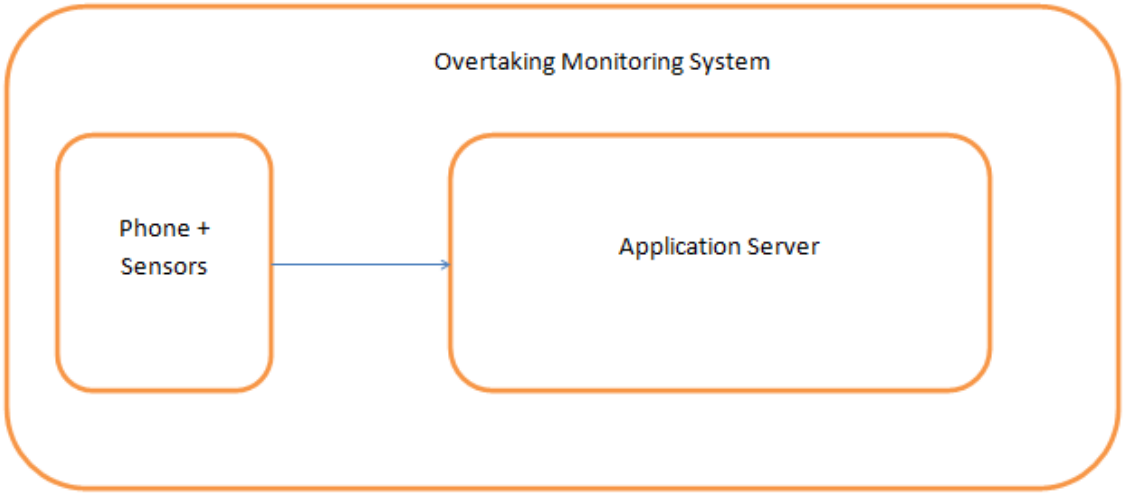


Figure 13: The system two tiers

Phone Side

The phone side design is divided into five main components that are responsible for detecting, filtering, deciding on and sending any overtaking data to the application server. These components are designed to be loosely coupled and cohesive so it can be changed and replaced with other components without affecting the functionality of the system. The phone side can also be upgraded and more components can be added and plugged-in if it is required to detect any other type of driving behaviour events or sequences in the future.

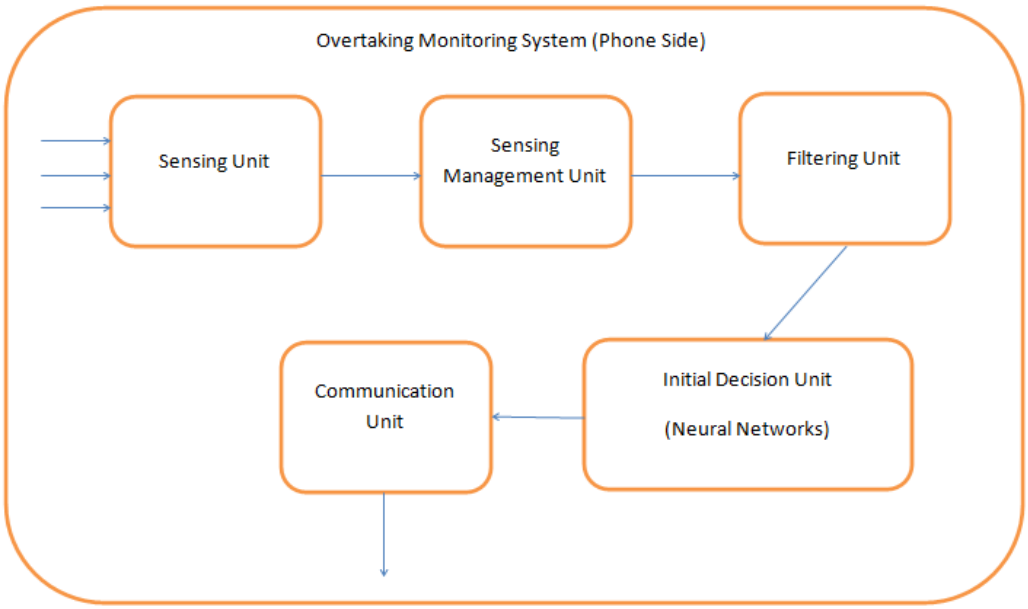


Figure 14: Phone side block diagram

Sensing Unit

- Sensing the movement and reporting the trajectory of the vehicle in the X-Plane
- Reporting the vehicle's GPS location.

Sensing Management Unit

- Controls and organises the sensing process
- Manages the sensors in the sensing unit
- Responsible for queuing and organising all the sensors data that comes from the sensors
- Organise the parallel distribution of the sensors data among the sensing filters in the Filtering Unit

Filtering Unit

- Based on the filter design pattern
- Group of pluggable filters that could be plugged and unplugged according to the required sensing functionality
- Responsible for filtering the required orientation and GPS sensors data
- Reports any overtaking sequences to the Initial Decision Unit
- Reports any left or right movements to the Initial Decision Unit

Initial Decision Unit

- Group of well trained, feed-forward, back-propagation and supervised learning Neural Networks
- Responsible of determining the initial overtaking sequences
- Responsible for determining the left and right movements
- Reports any initial overtaking sequences to the Communication Unit

Communication Unit

- Maintains a reliable communication with the server
- Sends all the initial overtaking sequence data along with its GPS location data to the server for further analysis and confirmation
- Ensures data delivery to the server

Class Diagram

The design of the class structure of the phone side is mainly based on the filter design pattern. This allows the class structure to be extended and upgraded easily. The advantage of using the filter design pattern is that it allows for adding any new movement sensing classes to be plugged into the system as filters and add more functionality to it without the need to change anything in the system structure, and also unplugging them at anytime if they are not needed.

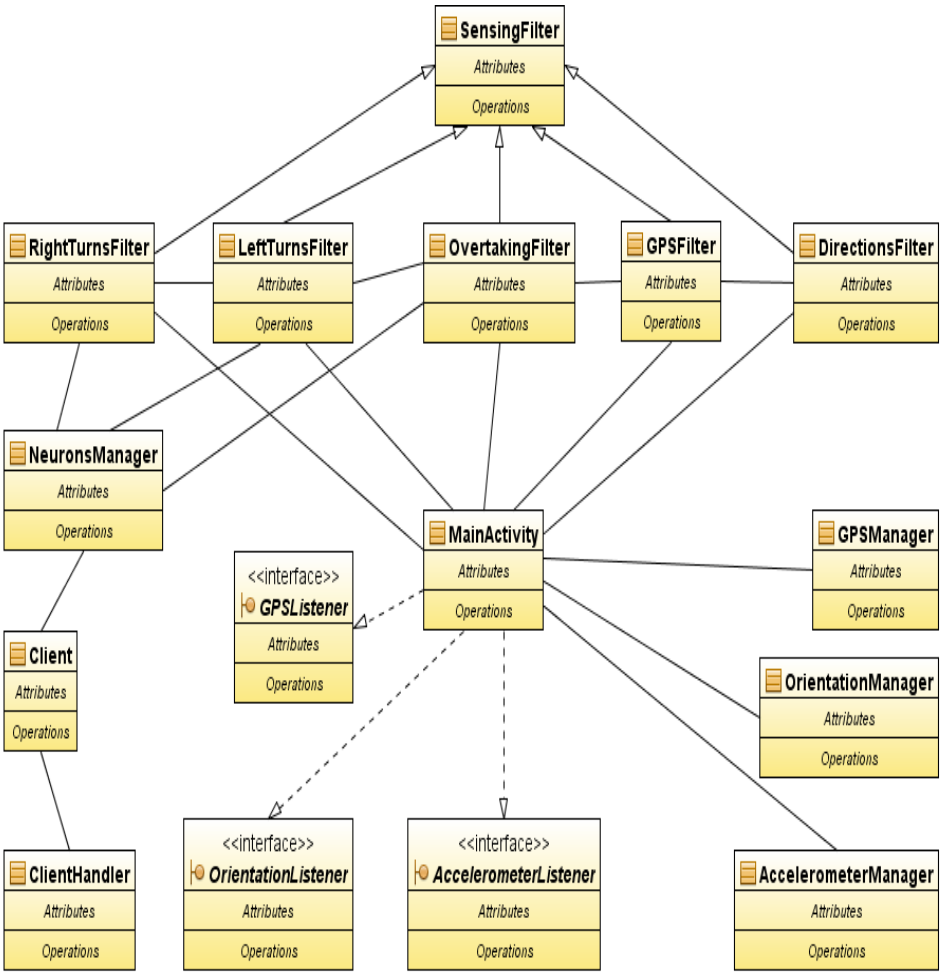


Figure 15: Phone side class diagram

Sensing filter class

An Abstract class that is designed to be extended and it is the main class in the filter design pattern. It contains two methods for getting and setting the next filter class.

Movements and GPS filter classes

Group of specialised classes where each class is a subclass of the SensingFilter Class and each is responsible for filtering and detecting only the movement that is set to filter and detect, for example the RightTurnFilter is specialised in detecting and filtering only the right turns movement data that is coming from the orientation sensor, the LeftTurnFilter is only specialised in detecting and filtering the left turn movements and the OvertakingFilter is specialised in detecting and filtering the overtaking sequences. Among these filter classes there is also the GPSFilter class that is specialised in obtaining the GPS location data when an overtaking pattern is detected.

Movements and GPS manager classes

Group of Manager classes that implements number of interfaces that are used by the MainActivity class for initialising, starting and managing the phone sensors.

Neurons manager class

The NeuronsManager class is responsible for all the initialisation and access to all the neural networks classes and provides the necessary methods to the filter classes to recognise and determine the left, right and overtaking patterns. The NeuronsManager class is also responsible for forwarding any initial overtaking data to the Client class which will then send it to the server for further analysis and map matching.

Client and client handler classes

These two classes are mainly networking classes to ensure reliable communication and data delivery of the GPS and initial overtaking data between the phone and the server.

Operational Design

As it can be seen in the flowchart in figure [16] the system on the phone side begins its operation by initialising and starting all the sensors managers and the neural networks, after that it enters a waiting state and waits for the sensors data to arrive. When the sensors data arrives the system queues this data in a buffer and starts filtering any data that is not needed. The data is then split into equal copies and passed in parallel through three specialised filters for detecting and recognising overtaking patterns, left turns and right turns patterns. The data that is passed through these filters is also queued and filtered and any detected patterns, left or right or overtaking patterns, will be recognised by specialised neural networks. These neural networks are trained in advance to recognise these specific

patterns. The output from these neural networks is used as an input to another neural network that is trained to recognise an overtaking patterns and decides whether this is actually a full overtaking pattern or not. If the pattern is indeed an overtaking, the system will consider this as an initial overtaking that needs to be confirmed by the server through digital map matching and hence it prepares a message that consists of the overtaking and GPS data and sends it to the server.

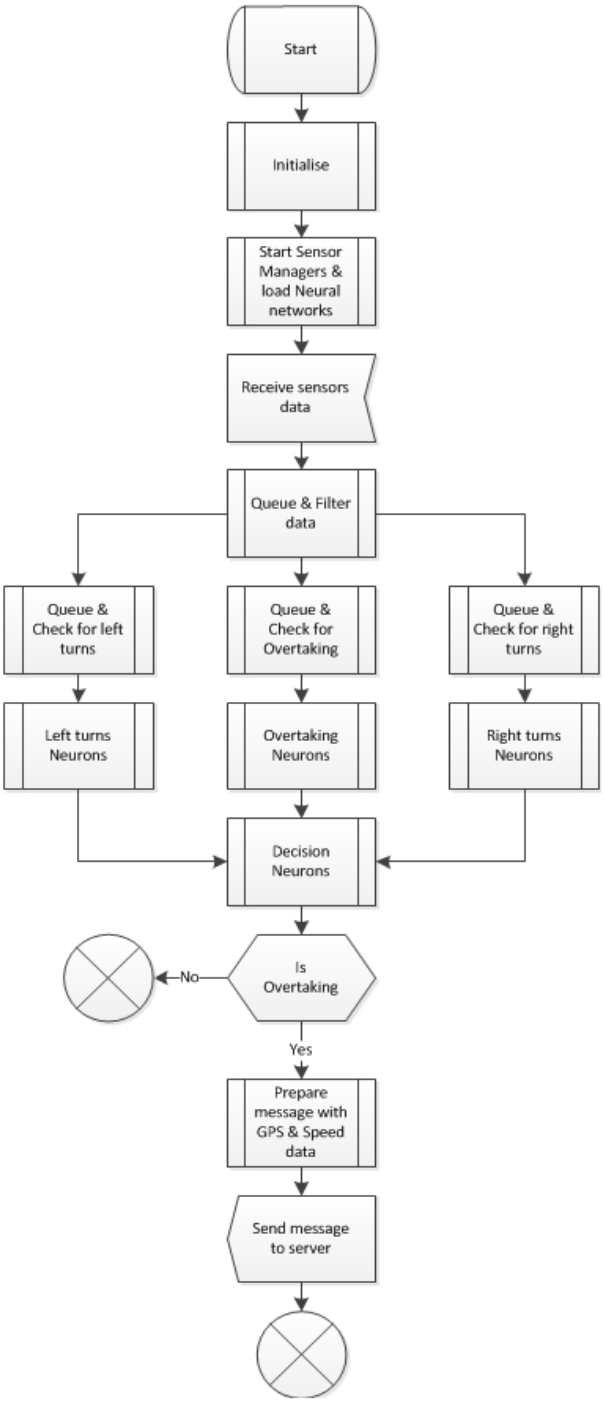


Figure 16: Phone side operational flowchart

Server Side

The server side design is divided into two main components, the IO component and the Digital map manager component. The IO component consists of the IoService, the IoFilterChain and the IoHandler. The design of these three components is based on the Java NIO capabilities that are provided by Apache Mina server API. The IoService acts as the communication gateway for the server and looks after the connection establishment and termination with clients, the IoFilterChain is a chain of filters that can be plugged and unplugged according to the need, these filters could be logging filters or codec filters such as Object Serialisation filter or any type of filters that the user can choose or implement. These filters allow us to design our own communication protocol and provide the system with the ability to add or remove these filters statically or dynamically at run time. The IoHandler is where all the connection handling and the validation of the client data take place. All these IO components are injected into the system as SPRING Beans through the use of SPRING framework that allows for easy configuration and alteration of these components and provides us with a better performance and scalable system.

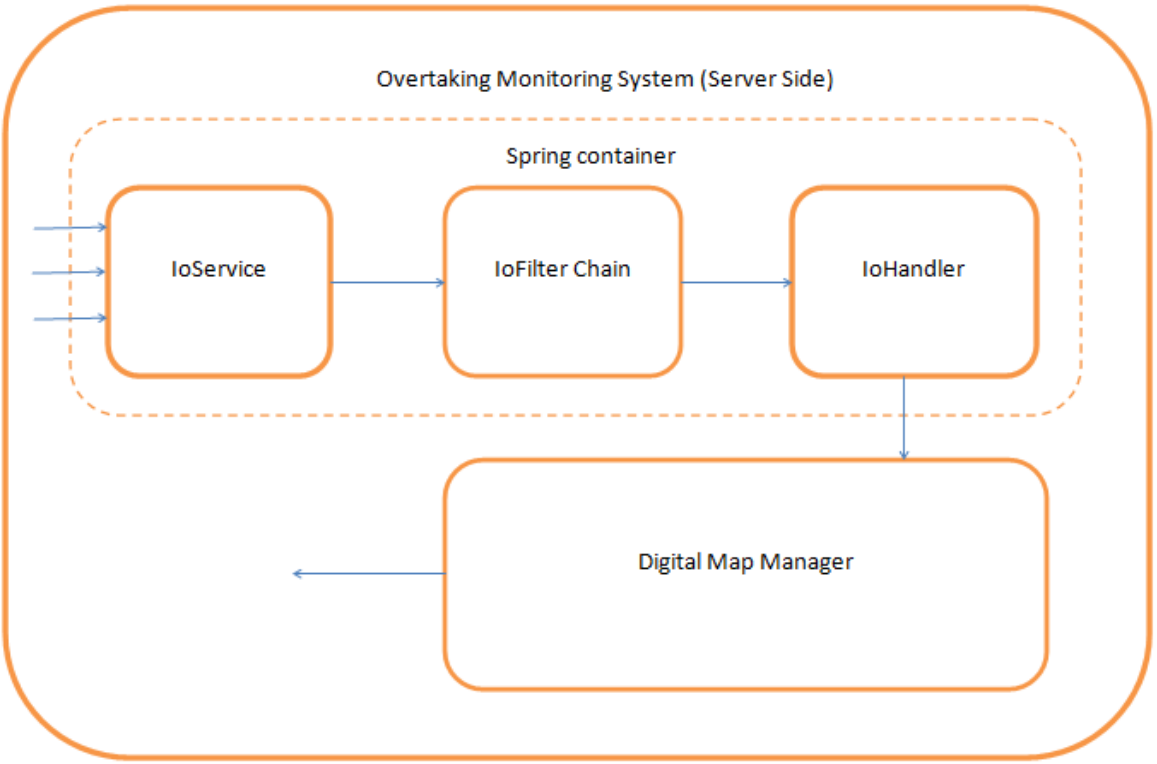


Figure 17: Server side block diagram

The other main component in the system is the Digital Map manager component, it is here where we carry out the main work and do the digital map matching of the overtaking data that is provided by clients.

Class Diagram

The most important classes in the class design of the system on the server side are the Server class and the MapManager class. These two classes represent the core of the system on the server side. The Server class with the help of the IoFilter class, the IoService class and the ServerHandler class provides all the functionalities that are needed to start the server and handle all the connections with clients.

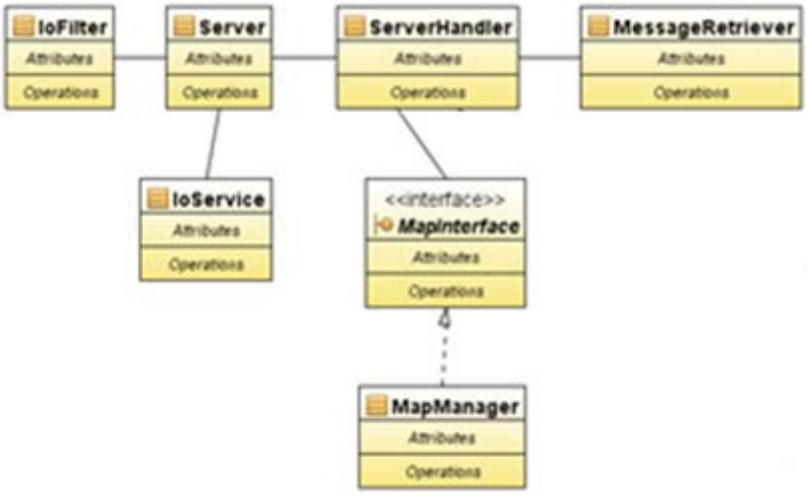


Figure 18: Server side class diagram

The MapManager class on the other hand provides all the functionalities for the digital map matching of all the overtaking data that is received from clients; it provides an interface class, the MapInterface class, for all these necessary map matching and GIS operations. Another class, the MessageRetriever class, is used by the ServerHandler to retrieve the client messages and validate them before carrying out any further processing on the data contained in the message.

Operational Design

The system on the server side starts by initialising the server and loading the digital map into memory; it then loads and starts the IO Filter chain and IO Service. After that the server

enters a waiting state until a data is received from a client, when any data is received from a client the server verifies this data and checks whether it is valid or not, if it is valid then the server starts the process of map matching and compares the received data with the actual data from the digital map, if this data after the comparison is an overtaking data then the server stores this data and goes back to into the waiting state to wait for another client data to arrive. If the received data after the map matching is not an overtaking data then it is considered a right turn data and it is fully discarded.

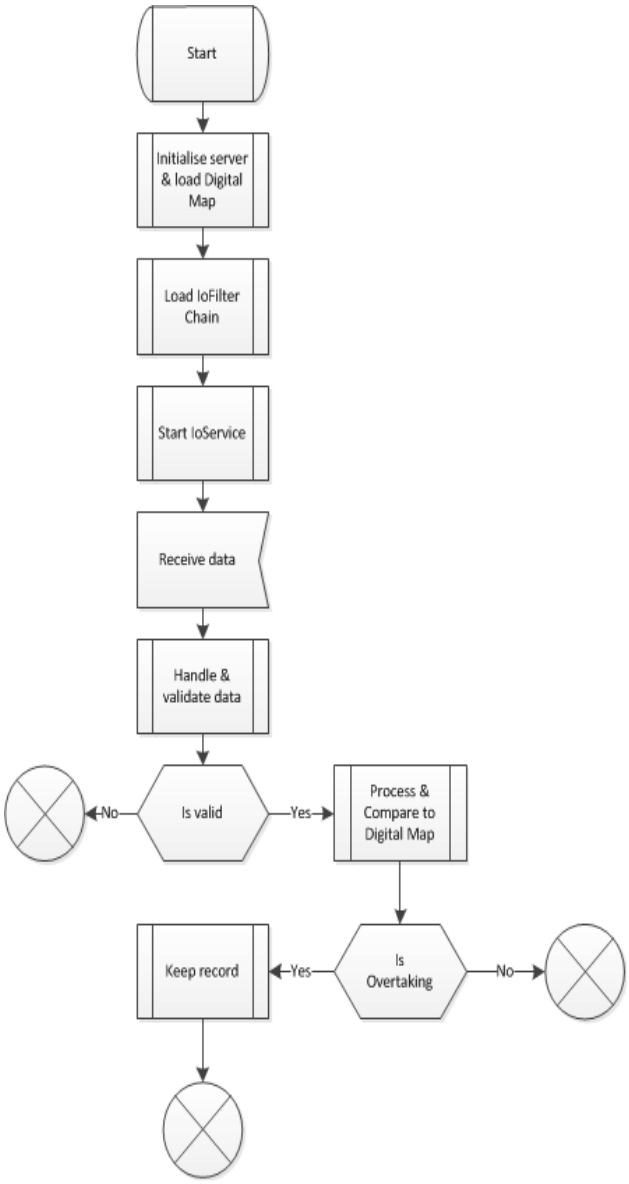


Figure 19: Server side operational flowchart

CHAPTER 4: IMPLEMENTATION

Technologies and Software Environment

All of the technologies that we used in implementing the system, on the client side and on the server side, are Java-based technologies.

Client side (The mobile phone)

All the development on the client side is carried out on an Android operating system [10] based mobile phone (i.e. HTC Desire mobile phone [11]). We used the Android SDK 2.1 and its library packages; in particular, we focused on using the Sensors package and the GPS package. We also made use of the networking API and other APIs when needed. The core of the application on the client side is running in multi-threaded background services that will detect and report any possible erratic driving behaviour to the server side. For the neural network development we used Neuroph 2.0 [30], a lightweight Java neural network framework that fits and runs on android for developing common neural network architectures.

Server side

The Java EE 5 SDK update 8 (with JDK 6 U18) packages is used for the main development on the server side. The server is based on Apache Mina 2.0 [28]. Mina is a network application framework, which helps users develop high performance and high scalability network applications. Mina provides an abstract event-driven asynchronous API over various transports such as TCP/IP and UDP/IP via Java NIO [28]. We also used SPRING 2.5.6 [29] for its performance and significant benefits in a J2EE environment through providing better leverage, enabling POJO [29,31] programming, dependency injection [29], which helps testability and simplify programming. For our map matching developed algorithm we used GeoTools java GIS library [27] for pinpointing the vehicle GPS coordinates on real road network and determine the geometry of the road where an overtaking takes place.

Methodology

The methods and the techniques that we developed for detecting the movement of the vehicle are all based on the roll, the yaw and the pitch readings from the phone orientation sensor. These angular movements, known also as Tait–Bryan angles, give a clear indication of the movement of the vehicle in any possible direction. The roll movement is defined as the movements about the y-axis, the pitch movement is defined as the movement about the x-axis and the yaw movement is defined as the movement about the z-axis.

Most modern mobile phones are equipped with sensors that can detect movements in different directions and Android is one of them. We focus on the orientation sensor's reading of the yaw/azimuth movement, this reading gives us an indication that the heading angle of the vehicle relative to the true north has changed. We use this orientation angle to determine the direction of the vehicle, either to the right or to the left direction (more details on this in the following sections). In the following sections we present our methods and techniques for detecting erratic driving behaviour caused by overtaking, and show how we actually use the orientation sensor, the neural networks and GIS tools to achieve that.

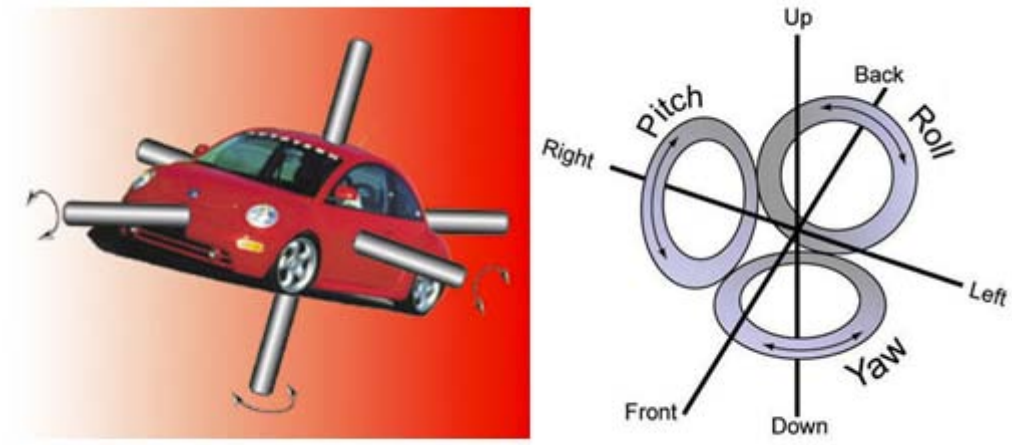


Figure 20: Yaw, Pitch and Roll angles

Detection Methods and techniques

The detection process through all the sensors is carried out on continuous timely bases where every move of the vehicle at anytime counts i.e. not a discrete fashion. The data from these sensors is initially analysed and checked for possible erratic driving behaviour patterns caused by overtaking. Listeners and filters are created and tuned to detect this kind of

driving behaviour patterns and events. The output from these listeners and filters is then used as an input to neural networks where it will be initially decided if the detected pattern is a possible overtaking pattern and worth sending to the server for further analysis and confirmation or not. The use of the event listeners, the filters and the neural networks on the phone side is explained further in the following sections.

In order to get a digital representation of all the possible trajectory movements of the vehicle that are sensed by the orientation sensor, we need to represent each of these movements with a digital number, for example if we represent the right movement (one degree in the clock wise direction) of the vehicle with digit 1 and represent the left movement (one degree in the anti-clock wise direction) with digit 2 and represent the straight line movement (no change in direction) with the digit 0 as shown in figure [21], then we can eventually represent all the possible movements of the vehicle in a digital format.

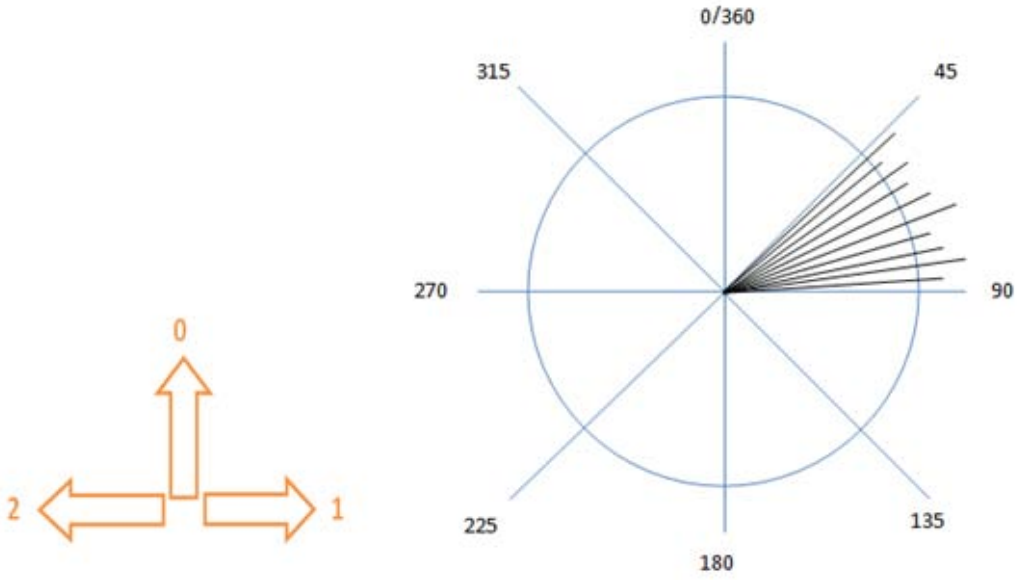


Figure 21: Digital representation of vehicle movements

Detecting Speed

Every time the location of the vehicle changes in time the system calculates the distance between the current and the previous location GPS points and divides that by the elapsed time to determine the average speed of the vehicle. The speed calculation process is actually carried out only when the vehicle is moving and there is a possible overtaking

pattern detected. This speed data is then combined with the overtaking data and sent to the server.

Detecting overtaking patterns

Detecting overtaking patterns is not a task that can be achieved easily with normal logic programming, overtaking patterns take many forms and there are numerous possibilities of how they could happen. It is correct to assume that a full overtaking pattern consists of a right turn followed by a left turn and straight line and then another left turn followed by a right turn and another straight line. If the vehicle does only a right turn followed by a left turn and straight line, we consider this pattern as an overtaking pattern as well. Detecting the overtaking pattern involves detecting different trajectory movements of the vehicle (right, left and straight) and all their combined possibilities.

After having the entire possible vehicle movement coming from the orientation sensor represented digitally, we split this data into three equal copies and pass one of them in parallel through a specialised overtaking filter that is tuned to only pass any movement that is considered to be a possible overtaking manoeuvre, this filtered data is then passed as an input to a specialised multi-layer feed-forward back-propagation neural network, designed and trained by us to recognise and confirm all initial overtaking manoeuvres. Figure [22] shows some of the inputs to the overtaking neural network and their outputs along with the actual overtaking neural network (see appendix A for full table of these inputs and outputs). The output of digit 1 from this neural network for recognising overtaking patterns represents “overtaking” and the output of digit 0 represents “right turn/weaving”.

Input 1	Input 2	Input 3	Input 4	Input 5	Input 6	Output
0	1	0	0	0	0	0
0	1	0	0	0	1	0
0	1	0	0	0	2	1
0	1	0	0	1	0	0
0	1	0	0	1	1	0
0	1	0	0	1	2	1
0	1	0	0	2	0	1

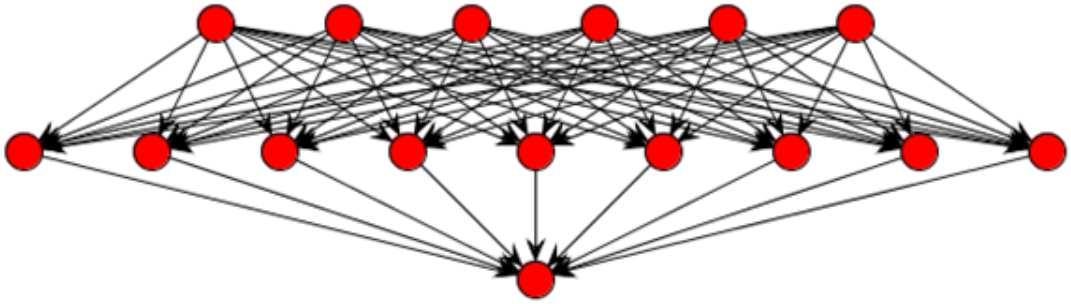


Figure 22: Overtaking neural network and some of its inputs and outputs

The neural network that we use here provides us with the artificial intelligent we need to decide whether a certain manoeuvre is actually an overtaking manoeuvre or not.

Detecting right and left turns

The same method and concept that is used for detecting overtaking patterns is used here for detecting the right turns and left turns. After the sensing data arrives from the orientation sensor, we split this data into three equal copies and pass one of these copies in parallel through the filter that is designed for only passing any manoeuvres that is possible to be a right turns, and pass another copy through the filter that is specialised in passing any manoeuvres that is possible to be a left turn manoeuvres. The output of these filters is used as an input to the neural networks that are specialised in recognising only right turn and left turn patterns. Figure [23] shows some of the inputs to the right turn neural network and their outputs along with the actual right turn neural network (see appendix A for full table). The output of digit 1 from the neural network for recognising right turn patterns represents a “right turn” and the output of digit 0 represents “weaving”.

Input 1	Input 2	Input 3	Input 4	Input 5	Input 6	Output
1	0	0	0	0	0	1
1	0	0	0	0	1	1
1	0	0	0	0	2	0
1	0	0	0	1	2	1
1	0	0	0	1	0	1
1	0	0	0	1	1	1
1	0	0	0	2	0	0

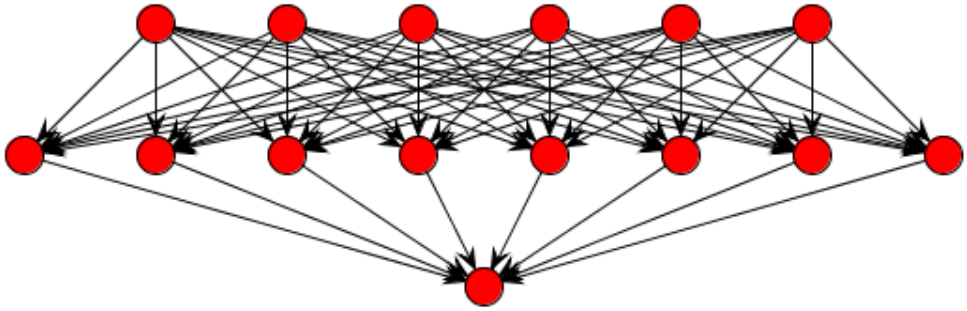


Figure 23: Right turn neural network and some of its inputs and outputs

The same concept is shown in figure [24] for the left turn neural network where output of digit 1 means “left turn” and output of digit 0 means “weaving”.

Input 1	Input 2	Input 3	Input 4	Input 5	Input 6	Output
2	0	0	0	0	0	1
2	0	0	0	0	1	0
2	0	0	0	0	2	1
2	0	0	0	1	0	0
2	0	0	0	1	1	0
2	0	0	0	1	2	0
2	0	0	0	2	0	1

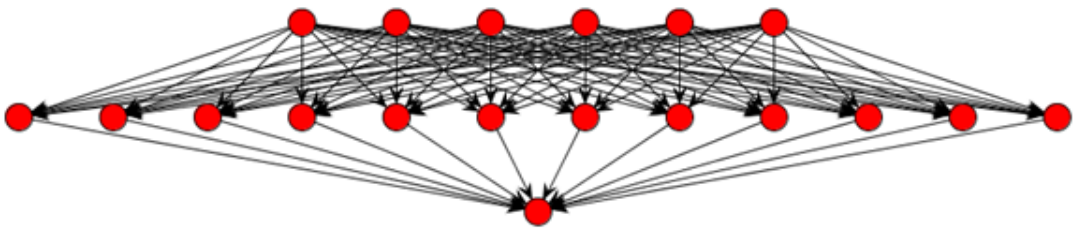


Figure 24: Left turn neural network and some of its inputs and outputs

Special cases

The overtaking pattern and time both depend on number of factors; among these factors is the speed of the vehicle, the distance between two the vehicles and the type of the road (one lane or multi-lane road). There are cases when an overtaking pattern cannot be fully detected and represented by 6 digits. The vehicle could take a long right turn before turning to the left to overtake another vehicle or change lanes, and the road itself could be a multi-lane road. To cater for this situation or similar situations, the system first reads the 6 first right turn moves, keeps a 6 digit representation of it in a buffer and ignores the rest, it then waits for the left turn to happen within 18 seconds, if the left turn happens within 18 seconds then it consider this as an overtaking pattern and carries on as normal, if not then it considers this as a right turn instead and ignores it.

The Initial overtaking concept

The output from the overtaking neural network, along with the output from the right turn and the left turn neural networks, are used as inputs to a final neural network. This neural network is specialised in deciding whether the detected pattern should be considered as an initial overtaking that is worth sending to the server or not. All we are sure of, when we decide that the pattern is initially an overtaking pattern, is that the vehicle has definitely made a manoeuvre that is initially considered to be an overtaking manoeuvre, but we still don't have any definite knowledge of whether this manoeuvre is indeed an overtaking manoeuvre, or it is just the shape of the road that happened to look like an overtaking pattern (i.e. the road is a right turn followed by a left turn and a straight line). To confirm this we construct a message with the GPS coordinates of the event along with the speed data where the even happened (the overtaking event) and send this message to the server for final confirmation. Figure [25] below shows all the possible outputs from the initial overtaking neural network and all the possible input combinations from the overtaking neural network, the right turn and the left turn neural networks, the figure also shows the corresponding neural network layout.

This neural network (the initial overtaking network) is mainly used to confirm that the vehicle had indeed made a manoeuvre, that should be considered initially, as overtaking and not to miss any manoeuvres that should be considered as such.

Notice that the output of the initial overtaking neural network is always 1 which means "initial overtaking" whenever the input value of the overtaking is 1 regardless of what the other input values are and it is 0 whenever the value of the overtaking is 0 which means "definitely not initial overtaking", with one exception to this rule, and that when the overtaking input value is 0 but the other values are 1. This means a right turn was detected and a left turn was also detected but the overtaking pattern wasn't fully detected (for more details see the special cases section above). There are cases shown in the table below that by design can never happen under normal circumstances but they are still possibilities that need to be counted for. For example the cases when the overtaking value is 1 and either the right turn value or the left turn value or both are 0's, this in fact is not possible to happen under normal circumstances as the system is designed to have the same replica of the data sent to each neural network (right turn, left turn and overtaking neural networks), therefore if there is an overtaking detected by the overtaking neural network it implicitly means a right turn and a left turn are going to be detected by their specialised neural networks too and produce a value of 1 each.

Right	Left	Overtaking	Output
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

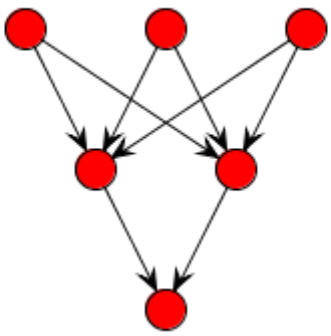


Figure 25: Initial overtaking neural network and all possible inputs and outputs

Server side map matching and processing

The most important aspect in the server side implementation is the map matching and processing that take place when an overtaking data is received from the client. As it can be seen from the flowchart in figure [26], the system first loads the shapefile into memory and then waits for the data from clients to arrive. After the data arrives from a client, the server validates the data and puts it in a queue, then creates an envelope (i.e. search window) and starts searching the digital map for matched features for each GPS point, if it happens that a feature couldn't be found then the system increases the search window and starts again from that particular GPS point. When a match is found the system starts to extract only the features with the shortest distances to each GPS point. A GPS point can lie between several features (i.e. roads) but the system will only retain the feature with the shortest distance to the point. After this stage is over and all the points are matched to their specific road features, the system enters into a voting stage and elects only the feature with the highest number of short distances to the GPS points and creates a line segment from it. At this stage the system needs to create a line segment from all the GPS points as well. Due to GPS limited accuracy it is difficult to assume that all the GPS points will form a straight line, therefore the system calculates the RMS value for all the GPS points.

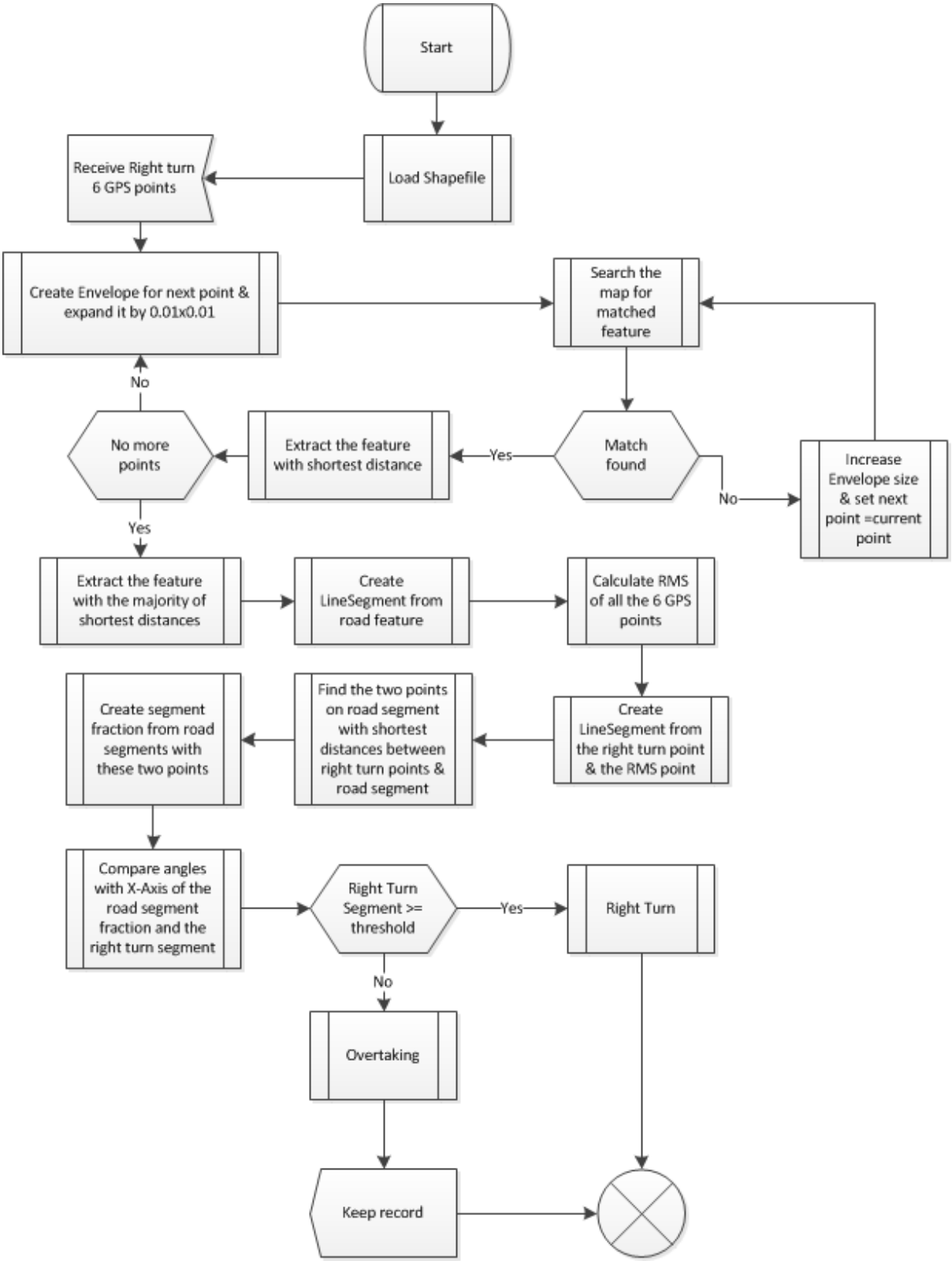


Figure 26: Map matching algorithm

We use this RMS value point together with the original first GPS point in the group to create a road segment fraction of two points that represents the right turn part of the overtaking manoeuvre. The system after that creates a segment fraction from the actual road with

these two points, it does that by finding the nearest two points on the road segment to these to these two points and creates a road segment fraction out of that. Now all the system needs to do is compare the two angles, the road segment fraction angle and the right turn angle, and determine the difference between them. The manoeuvre is considered to be a definite overtaking if the angle of the right turn segment fraction is less than the angle of the actual road segment fraction. If the angle of the right turn segment fraction is greater than or equal to the actual road angle then the manoeuvre is considered to be a right turn manoeuvre so the system discards it. All the angles are calculated with respect to the x-axis see figure [27] for more details. Threshold is taken as one degree in the positive direction and one degree in the negative direction, so a difference of one degree between any two angles is considered to be zero.

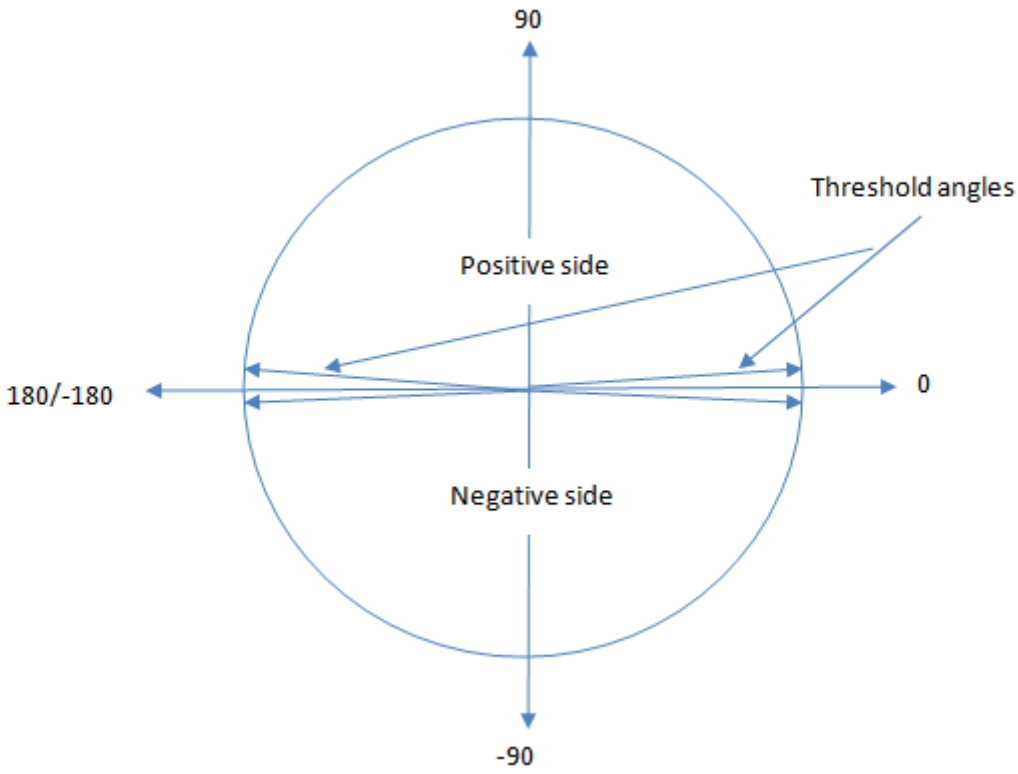


Figure 27: angles with respect to x-axis and the allowed threshold angles

Voting algorithm

The algorithm is simple. GPS coordinates of a vehicle travelling on the road, taken over a period of time, in most cases can't form a perfect straight line. This can introduce real difficulties in determining the right road segment where the overtaking had taken place,

among these difficulties is when a GPS point lies with equal distance between two segments and when some of the GPS points lie closer to a segment and that segment is in fact the wrong segment where the overtaking had happened. To solve this problem and ease all the difficulties that can be raised due to the limited characteristics of the GPS accuracy and the layout of the road network, we use a voting algorithm that imitate the normal voting algorithm in real life, the segment with the majority votes gets elected. The votes here are the number of shortest distances from the GPS points to a segment. Figure [28] shows two situations at the same location with one when a vehicle is doing an overtaking and the other when the vehicle is doing a right turn.

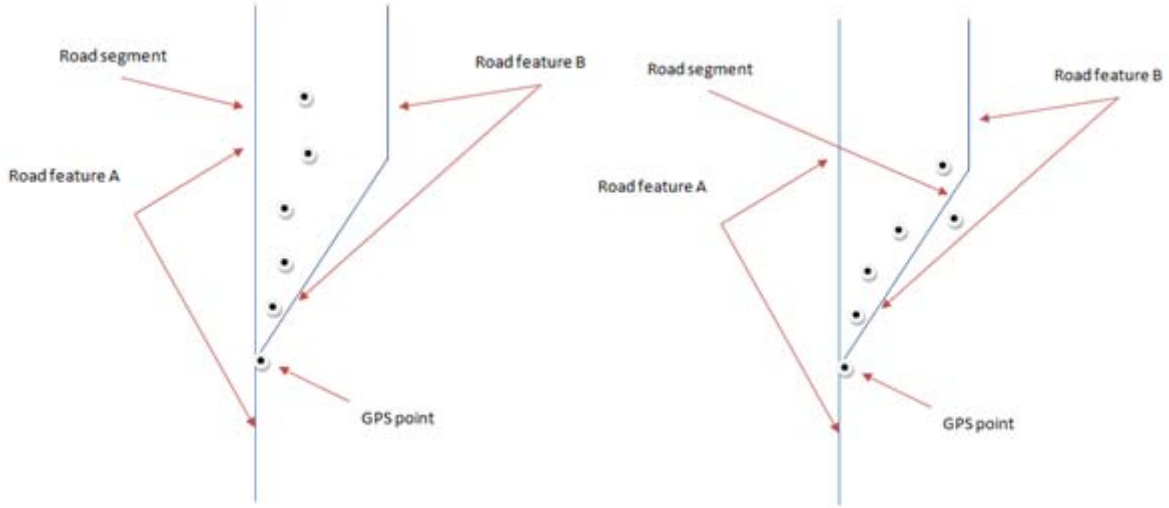
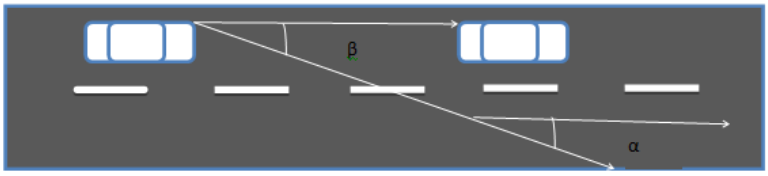


Figure 28: Overtaking and a right turn GPS actual coordinates

Determining Overtaking and right turns

After the voting is carried out and the RMS point is calculated, we end up with two points that represent the vehicle's path. To determine the overtaking we calculate the difference between this path's angle and the actual road's angle, if this difference is less than the allowed threshold then it is an overtaking and if the difference is more than the threshold then it is a right turn. Figure [29] shows a general scenario with all the angles that a vehicle makes when it tries to overtake another vehicle. The angle β is the angle that a vehicle makes when it turn to the right to do an overtaking, and the angle α is the angle that a vehicle makes when it turns to the left to straighten itself and the angle Ω is difference angle between the two. It also shows how the road segment fraction AB is determined by

mapping the points C and D on the digital map. Figure [30] shows an overtaking scenario and the difference between an overtaking angle and the road angle.



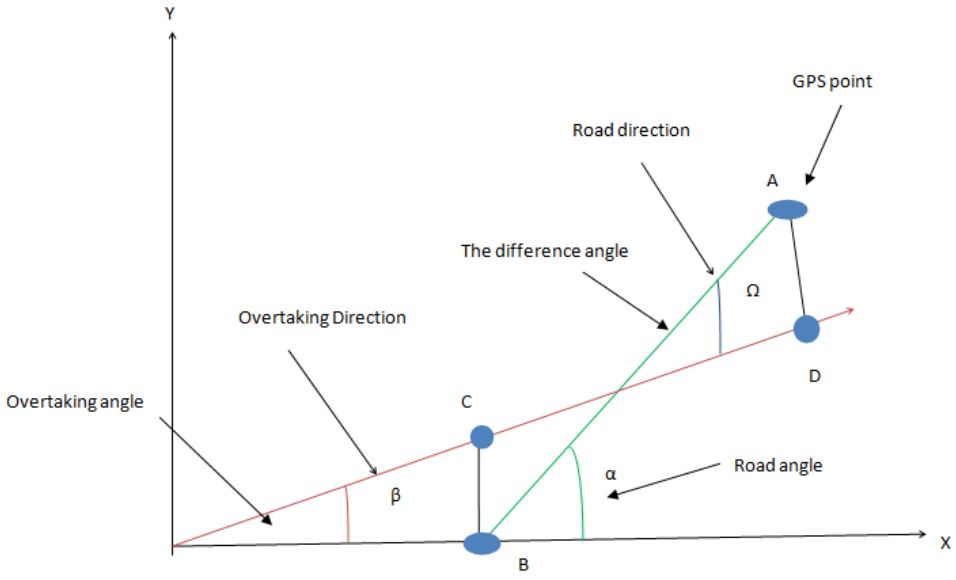


Figure 29: General scenario with the vehicle and road angles

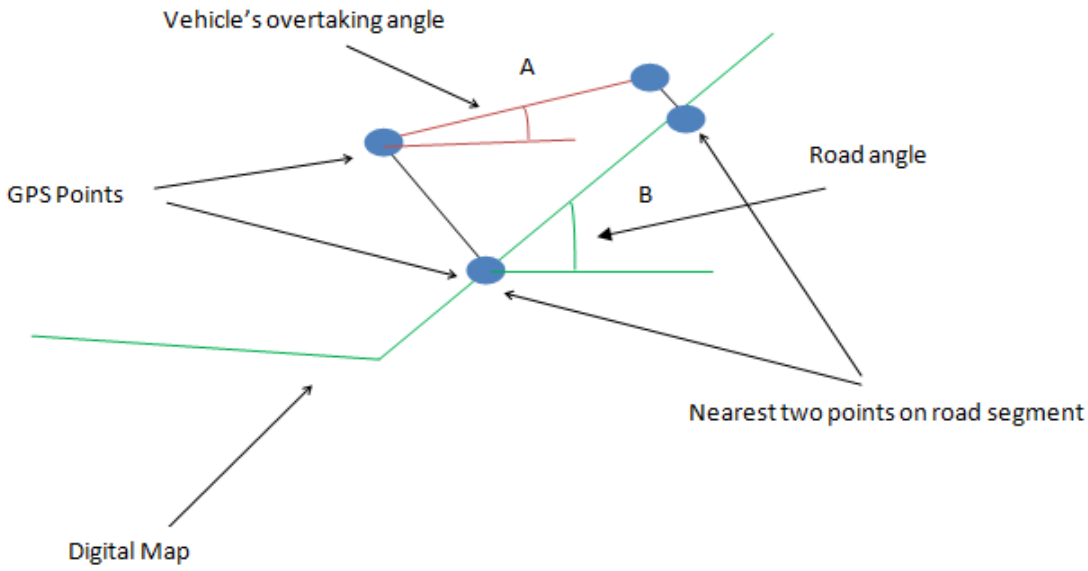


Figure 30: Overtaking angle and the road angle

Figure [31] and [32] show two possible scenarios of a right turn, one when the vehicle turns to the right at exactly the beginning of a segment and the other when the right turn happens before the beginning of the segment. In both scenarios the vehicle is making an angle that is greater than the road angle which indicates that it is indeed a right turn not an overtaking.

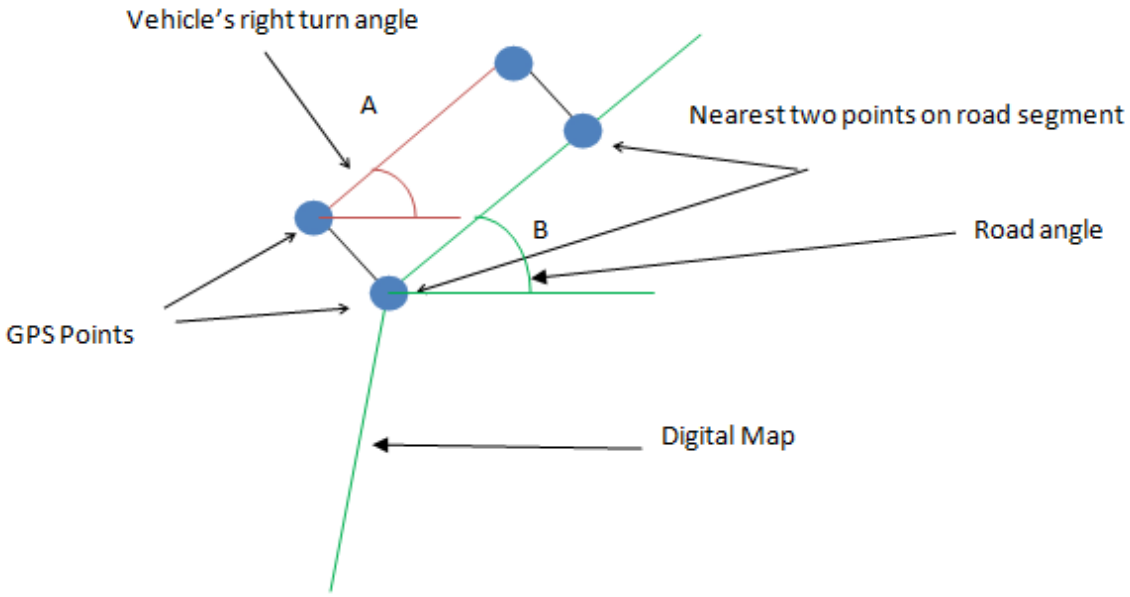


Figure 31: Right turn with an angle equals to the road angle

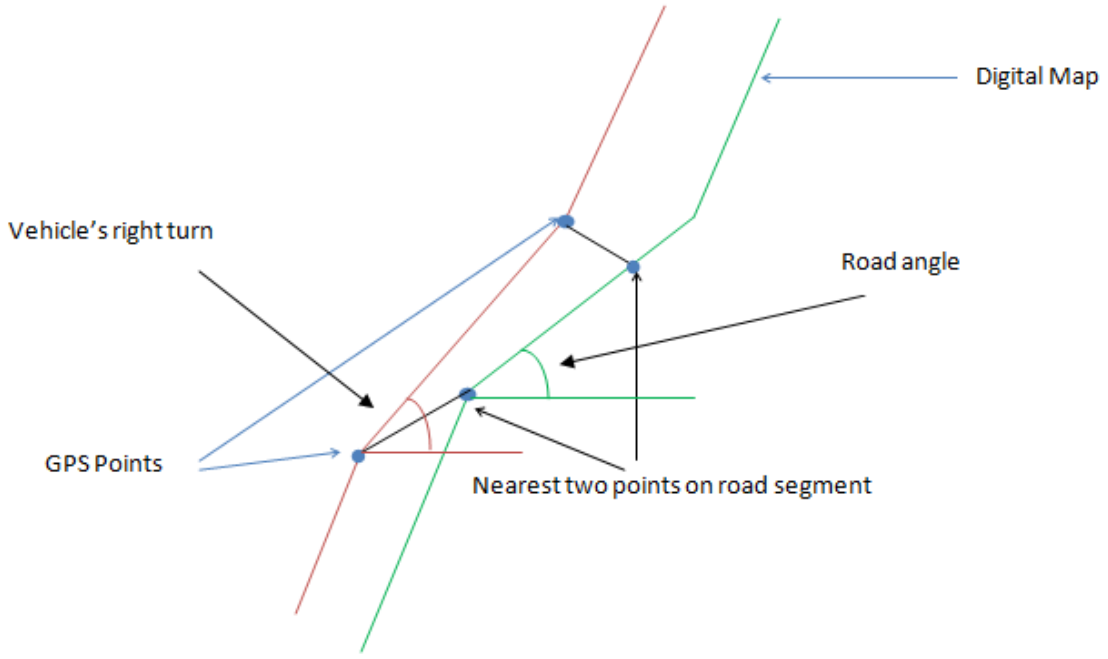


Figure 32: Right turn with an angle greater than the road angle

CHAPTER 5: EVALUATION

Evaluation goals

An essential part of our study was to evaluate the suitability of off-the-shelf technologies in detecting vehicle's movement and test their performance, as well as checking their usability in detecting erratic driving behaviour caused by overtaking. In addition to that we also aimed to evaluate the compatibility of GPS in this kind of applications and test any schemes or algorithms that we came up with to compensate for its limited accuracy. Another essential part of our study was to investigate and show the benefits of using GIS systems as a confirmation tool for all the detected erratic driving behaviour events and get a feel of its reliability in providing real time information. The following sections will show how these aims were achieved and evaluated.

Due to circumstances that were beyond our control, all our tests were carried out in an indoor environment, which means there are no real GPS data available for our tests and all the driving was carried out virtually by using the phone as a steering wheel.

The Tests

Client side

The orientation sensor

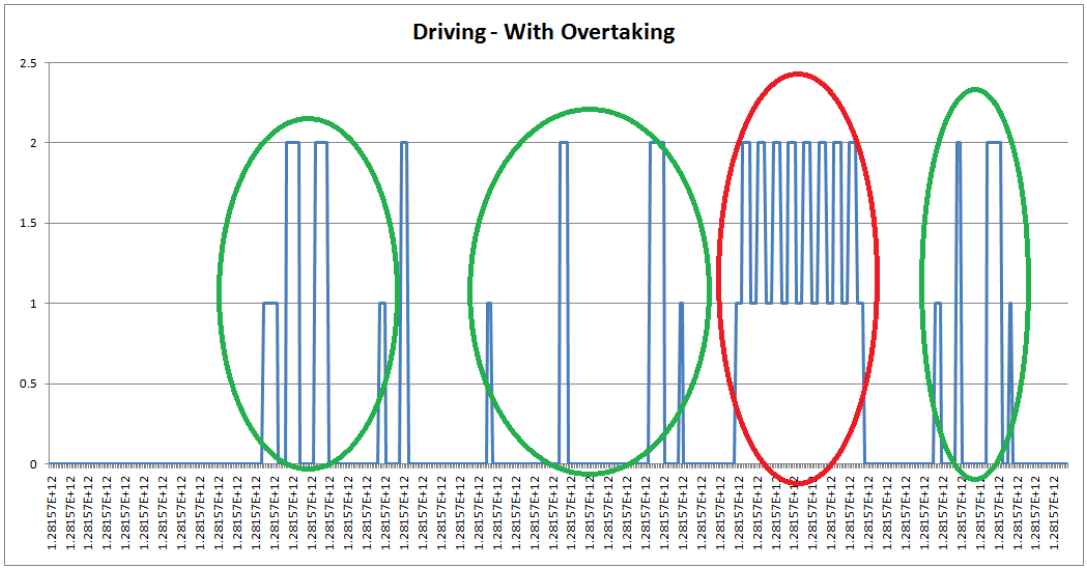


Figure 33: Overtaking and weaving patterns detected by the system

A value of 0 on the vertical axis in figure [33] above indicates a straight line movement, a value of 1 indicate a right turn and a value of 2 indicates a left turn, the horizontal axis represents the system elapsed time in milliseconds. The test is carried out in an indoor environment by holding the phone as a steering wheel and does a virtual driving. As it is shown in the graph a number of different types of overtaking patterns are detected by the system, all marked in green. By taking a closer look at each one of these overtaking patterns and focus on the time that each one had taken, we can see that the first one to the right had actually taken less time to finish than the others, which indicates a quick overtaking. The overtaking pattern in the middle is obviously taking larger time than all the others, which indicates a slow overtaking, notice the large time gap between the first right turn and the first left turn. The pattern marked in red is a weaving pattern as it is all right turns followed immediately by left turns.

Initial overtaking

Regular overtaking

Regular overtaking is an overtaking that is finished within 18 seconds and the first right left turn movement takes a minimum of 3 seconds and a maximum of 8 seconds. Below are the data of a regular overtaking detected during the virtual driving with no GPS enabled.

```
10-24 17:03:11.453: INFO/System.out(25580): Right -->  (Start of the right turn)
10-24 17:03:11.893: INFO/System.out(25580): Right -->
10-24 17:03:12.113: INFO/System.out(25580): Right -->
10-24 17:03:12.333: INFO/System.out(25580): Right -->
10-24 17:03:12.553: INFO/System.out(25580): Right -->
10-24 17:03:12.774: INFO/System.out(25580): Right -->
10-24 17:03:12.995: INFO/System.out(25580): Right -->
10-24 17:03:13.223: INFO/System.out(25580): Right -->
10-24 17:03:13.443: INFO/System.out(25580): Right -->
10-24 17:03:13.663: INFO/System.out(25580): Right -->
10-24 17:03:13.883: INFO/System.out(25580): Right -->
10-24 17:03:14.104: INFO/System.out(25580): Right -->
10-24 17:03:14.333: INFO/System.out(25580): Right -->
10-24 17:03:15.204: INFO/System.out(25580): Right -->
10-24 17:03:15.643: INFO/System.out(25580): Right -->
10-24 17:03:17.405: INFO/System.out(25580): <-- Left  (Start of the left turn)
10-24 17:03:17.853: INFO/System.out(25580): <-- Left
10-24 17:03:18.293: INFO/System.out(25580): <-- Left
10-24 17:03:18.733: INFO/System.out(25580): <-- Left
10-24 17:03:18.953: INFO/System.out(25580): <-- Left
10-24 17:03:19.173: INFO/System.out(25580): <-- Left
```

10-24 17:03:19.393: INFO/System.out(25580): <-- Left

10-24 17:03:20.053: INFO/System.out(25580): <-- Left

10-24 17:03:20.713: INFO/System.out(25580): <-- Left

10-24 17:03:21.373: INFO/System.out(25580): <-- Left

10-24 17:03:21.593: INFO/System.out(25580): <-- Left

10-24 17:03:22.253: INFO/System.out(25580): <-- Left

10-24 17:03:22.693: INFO/System.out(25580): <-- Left

10-24 17:03:22.913: INFO/System.out(25580): <-- Left

10-24 17:03:23.133: INFO/System.out(25580): <-- Left (End of initial overtaking pattern)

10-24 17:03:23.143: INFO/System.out(25580): Initial - Overtaking

10-24 17:03:23.143: INFO/System.out(25580): 0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0 (Data sent to server)

Notice that the right turn had taken around 6 seconds and the whole overtaking pattern had taken around 12 seconds to finish. Figure [34] show a snapshot of this data taken from the android debugger.

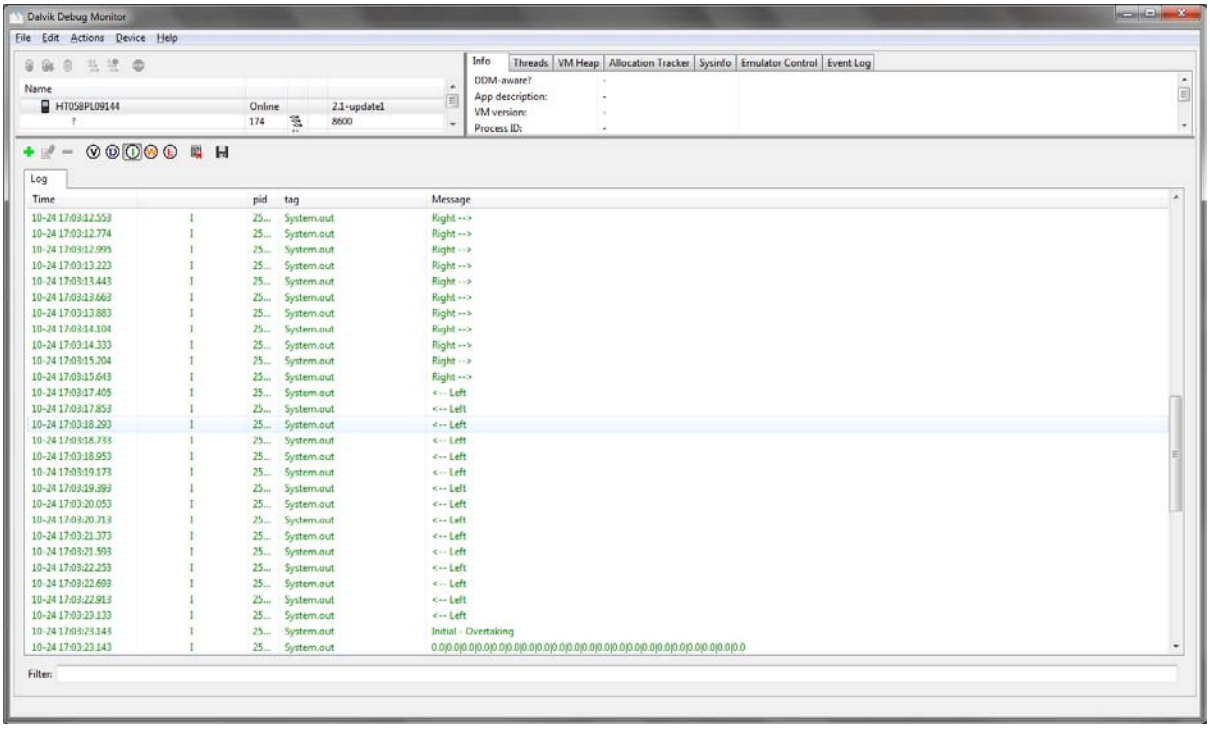


Figure 34: Snapshot of regular overtaking

Quick overtaking

Quick overtaking is an overtaking that is finished within 18 seconds and the first right left turn movement takes a minimum of 2 seconds and a maximum of 4 seconds. Below are the data of a quick overtaking detected during the virtual driving with no GPS enabled.

10-24 17:38:36.573: INFO/System.out(25956): Right --> (Start of the right turn)

10-24 17:38:36.793: INFO/System.out(25956): Right -->

10-24 17:38:37.013: INFO/System.out(25956): Right -->

10-24 17:38:37.233: INFO/System.out(25956): Right -->

10-24 17:38:37.454: INFO/System.out(25956): Right -->

10-24 17:38:37.893: INFO/System.out(25956): Right -->

10-24 17:38:38.113: INFO/System.out(25956): <-- Left (Start of the left turn)

10-24 17:38:38.334: INFO/System.out(25956): <-- Left

10-24 17:38:38.557: INFO/System.out(25956): <-- Left

10-24 17:38:38.783: INFO/System.out(25956): <-- Left

10-24 17:38:39.873: INFO/System.out(25956): <-- Left

10-24 17:38:40.103: INFO/System.out(25956): <-- Left

10-24 17:38:40.323: INFO/System.out(25956): <-- Left

10-24 17:38:40.543: INFO/System.out(25956): <-- Left

10-24 17:38:41.203: INFO/System.out(25956): <-- Left

10-24 17:38:41.423: INFO/System.out(25956): <-- Left

10-24 17:38:41.643: INFO/System.out(25956): <-- Left

10-24 17:38:41.866: INFO/System.out(25956): <-- Left (End of the initial overtaking pattern)

10-24 17:38:44.103: INFO/System.out(25956): Initial - Overtaking

10-24 17:38:44.113: INFO/System.out(25956): 00.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.00 (Data sent to server)

Notice that the right turn had taken around 2 seconds and the whole overtaking pattern had taken around 6 seconds to finish. Figure [35] show a snapshot of this data taken from the android debugger.

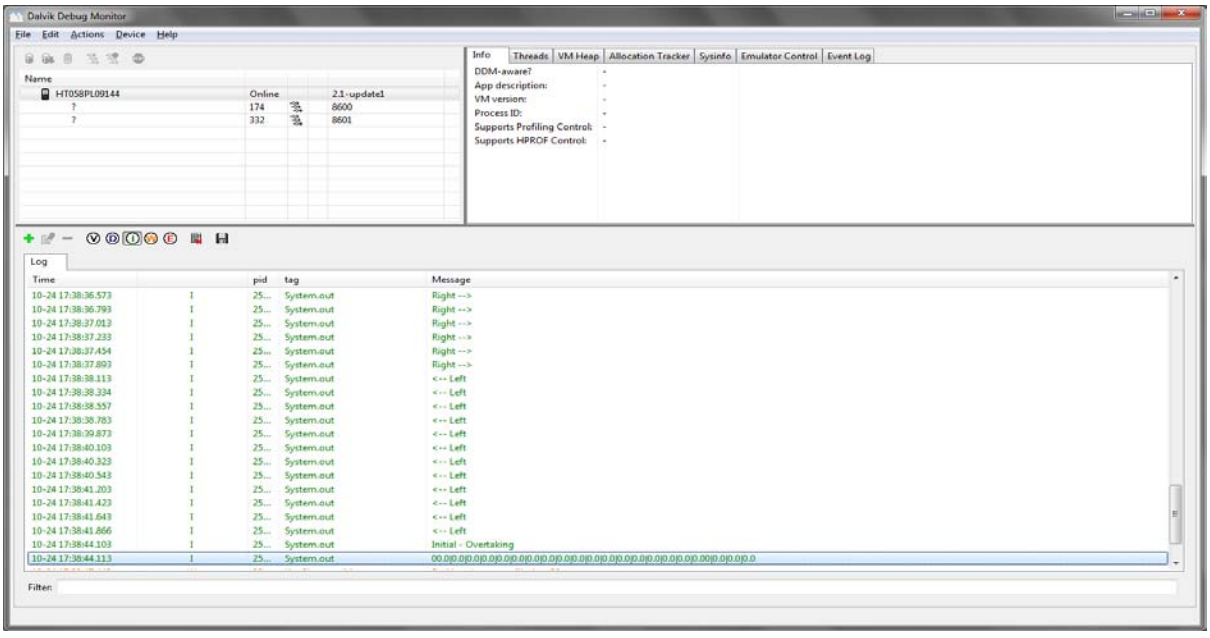


Figure 35: Snapshot of quick overtaking

Long overtaking

Long overtaking is an overtaking that takes more than 18 seconds to finish and the first right left turn movement takes a minimum of 13 seconds and a maximum of 18 seconds. Below are the data of a long overtaking detected during the virtual driving with no GPS enabled.

```
10-24 17:29:23.413: INFO/System.out(25956): Right --> (Start of the right turn)
10-24 17:29:23.634: INFO/System.out(25956): Right -->
10-24 17:29:23.863: INFO/System.out(25956): Right -->
10-24 17:29:24.083: INFO/System.out(25956): Right -->
10-24 17:29:24.303: INFO/System.out(25956): Right -->
10-24 17:29:24.523: INFO/System.out(25956): Right -->
10-24 17:29:24.743: INFO/System.out(25956): Right -->
10-24 17:29:24.963: INFO/System.out(25956): Right -->
10-24 17:29:25.183: INFO/System.out(25956): Right -->
10-24 17:29:25.403: INFO/System.out(25956): Right -->
10-24 17:29:25.623: INFO/System.out(25956): Right -->
10-24 17:29:25.844: INFO/System.out(25956): Right -->
10-24 17:29:36.614: INFO/System.out(25956): <-- Left (Start of the left turn)
10-24 17:29:36.835: INFO/System.out(25956): <-- Left
10-24 17:29:37.056: INFO/System.out(25956): <-- Left
10-24 17:29:37.283: INFO/System.out(25956): <-- Left
10-24 17:29:37.503: INFO/System.out(25956): <-- Left
10-24 17:29:37.723: INFO/System.out(25956): <-- Left
10-24 17:29:37.943: INFO/System.out(25956): <-- Left
10-24 17:29:38.163: INFO/System.out(25956): <-- Left
10-24 17:29:38.383: INFO/System.out(25956): <-- Left
10-24 17:29:38.603: INFO/System.out(25956): <-- Left
10-24 17:29:38.823: INFO/System.out(25956): <-- Left
10-24 17:29:39.043: INFO/System.out(25956): <-- Left
10-24 17:29:39.265: INFO/System.out(25956): <-- Left
10-24 17:29:39.706: INFO/System.out(25956): <-- Left
10-24 17:29:39.927: INFO/System.out(25956): <-- Left
10-24 17:29:40.373: INFO/System.out(25956): <-- Left
10-24 17:29:40.813: INFO/System.out(25956): <-- Left
10-24 17:29:41.684: INFO/System.out(25956): <-- Left
10-24 17:29:42.124: INFO/System.out(25956): <-- Left ( End of the initial overtaking)
10-24 17:29:42.133: INFO/System.out(25956): Initial - Overtaking
10-24 17:29:42.133: INFO/System.out(25956): 00.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.0|0.00 (Data sent to
server)
```

Notice that the right turn had taken around 16 seconds and the whole overtaking pattern had taken over 18 seconds to finish. Figure [36] show a snapshot of this data taken from the android debugger.

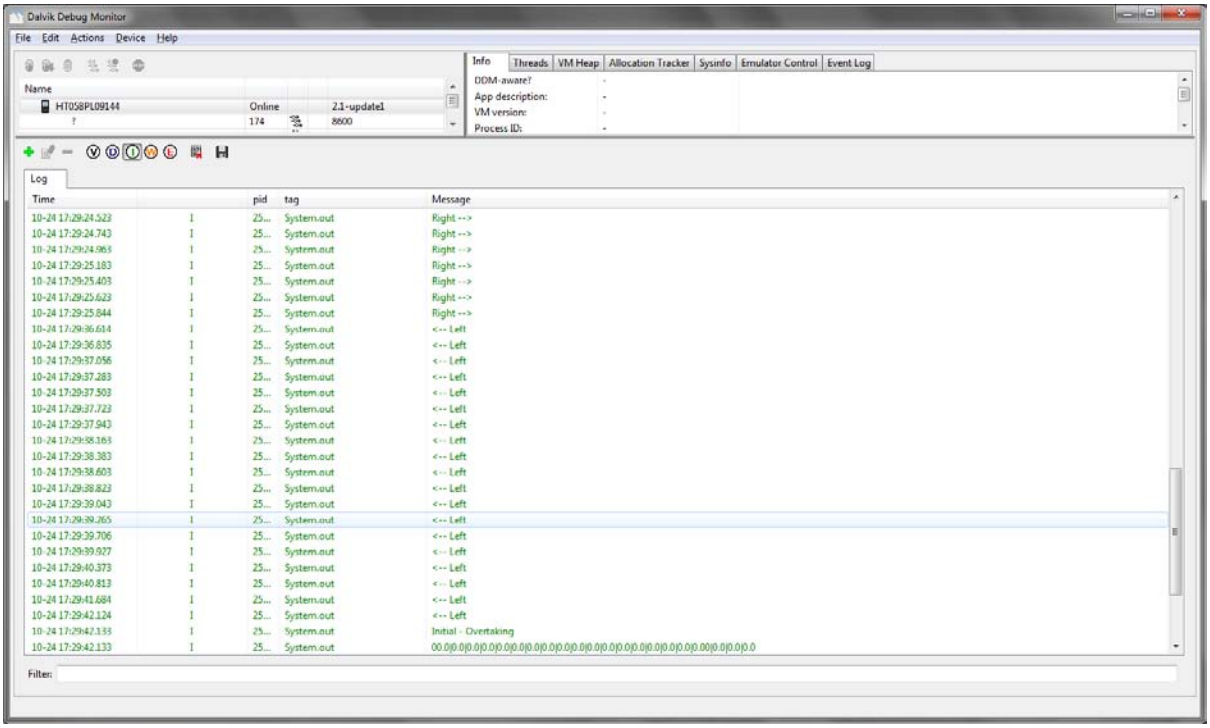


Figure 36: Snapshot of long overtaking

Server side

All the tests on the server side were carried out on the digital map shown in figure [37]. This digital map is not up to date but still serves our evaluation purposes well and fits perfectly with our aims. All the GPS data that we used here are virtual and taken directly from the digital map and not from a real data on an actual road.

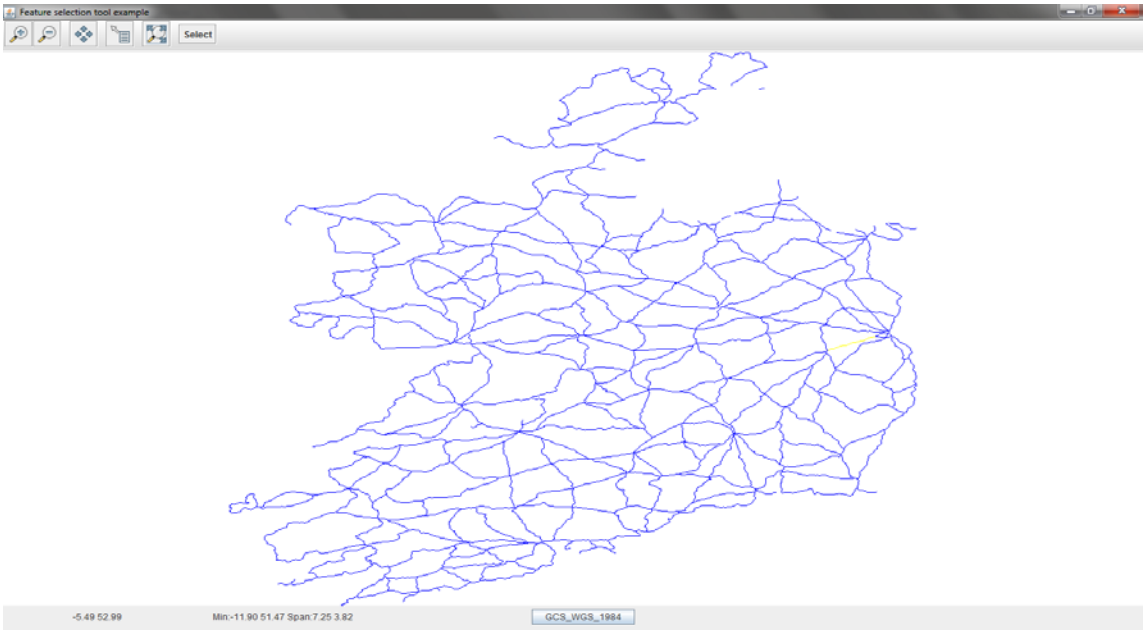


Figure 37: Digital map of Irish roads network

Voting algorithm

The best test case scenario to test our voting algorithm is to choose a road segment that lies very close to another two road segments, and then pick a number of GPS points on this segment that some of them lay either close to the other segments or lay perfectly between the chosen segment and one of the other two. The reason behind that is to see if the voting algorithm indeed works and elects the right segment. Figure [38] shows the GPS points and the chosen road segment.

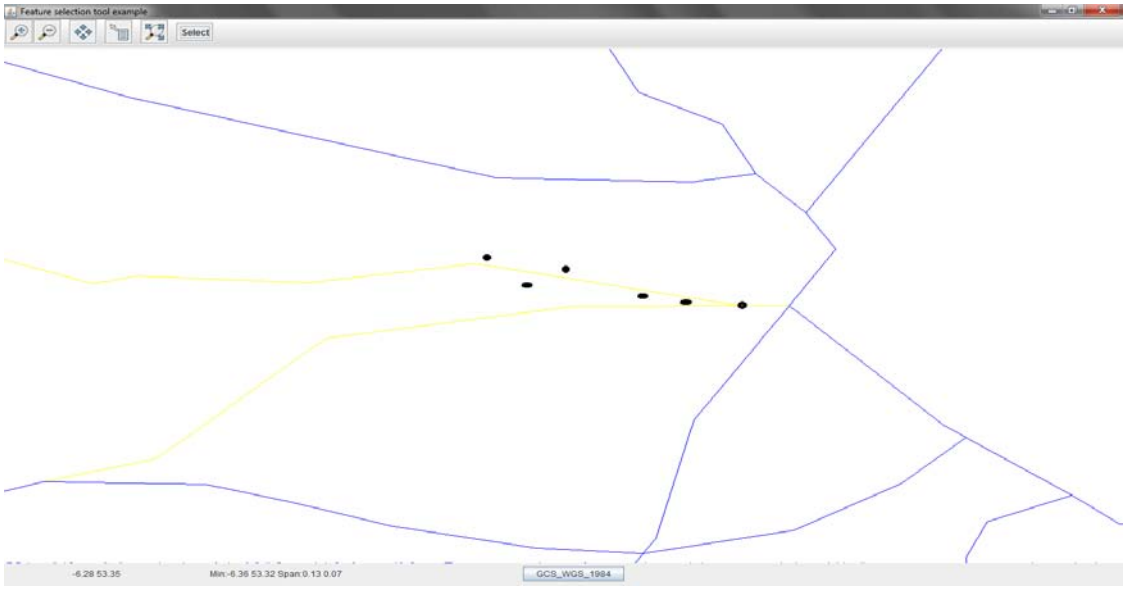


Figure 38: GPS location points for testing the voting algorithm

The GPS points we picked for this test are as follows:

Latitudes	-6.2773	-6.2813	-6.2904	-6.3009	-6.3035	-6.3072
Longitudes	53.3501	53.3503	53.3522	53.3528	53.3567	53.3563

The results we got from our system are as follows:

```
Shapefile is loaded ...
Best match result: Road ID =IRL_roads.195 Closest Distance =1.655441275553146E-5 Segment = LINESTRING( -6.277500231262874
53.35008233075617, -6.271999764273158 53.35011294875611)
Best match result: Road ID =IRL_roads.201 Closest Distance =2.3094813945773822E-4 Segment = LINESTRING( -6.277500231262874
53.35008233075617, -6.29274988523436 53.350029032756275)
Best match result: Road ID =IRL_roads.188 Closest Distance =2.3821385498748102E-4 Segment = LINESTRING( -6.307111239207509
53.35549925974605, -6.277500231262874 53.35008233075617)
Best match result: Road ID =IRL_roads.188 Closest Distance =0.0015374836954669781 Segment = LINESTRING( -6.307111239207509
53.35549925974605, -6.277500231262874 53.35008233075617)
Best match result: Road ID =IRL_roads.188 Closest Distance =0.0018309816696876413 Segment = LINESTRING( -6.307111239207509
53.35549925974605, -6.277500231262874 53.35008233075617)
Best match result: Road ID =IRL_roads.188 Closest Distance =7.733848730389571E-4 Segment = LINESTRING( -6.307916568206003
53.35564044274578, -6.307111239207509 53.35549925974605)
Road Id: IRL_roads.188 count: 4
Road Id: IRL_roads.195 count: 1
Road Id: IRL_roads.201 count: 1
Elected Road Id = IRL_roads.188
```

The results show that there were three road segments involved, IRL_roads.188 got 4 votes, IRL_roads.195 got 1 vote and IRL_roads.201 got also 1 vote, and the one that was elected is the road segment with an Id of IRL_roads.188 with the majority of 4 votes. This means that road IRL_road.188 is the one with the majority of shortest distances from most of the GPS points, and it is the one that will be used in later stages in the map matching process. The result also shows the 6 GPS chosen points and their distances from the best matched segment. Figure [39] shows a snapshot of the same system output data.

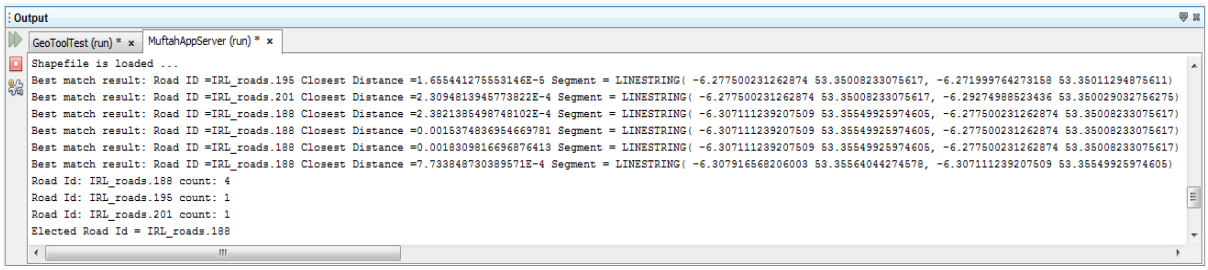


Figure 39: Snapshot of voting algorithm system output

Overtaking

The first test case scenario we are going to test here is when an overtaking is carried out on a straight road. The road segment we chose along with all the GPS points are shown in figure [40].

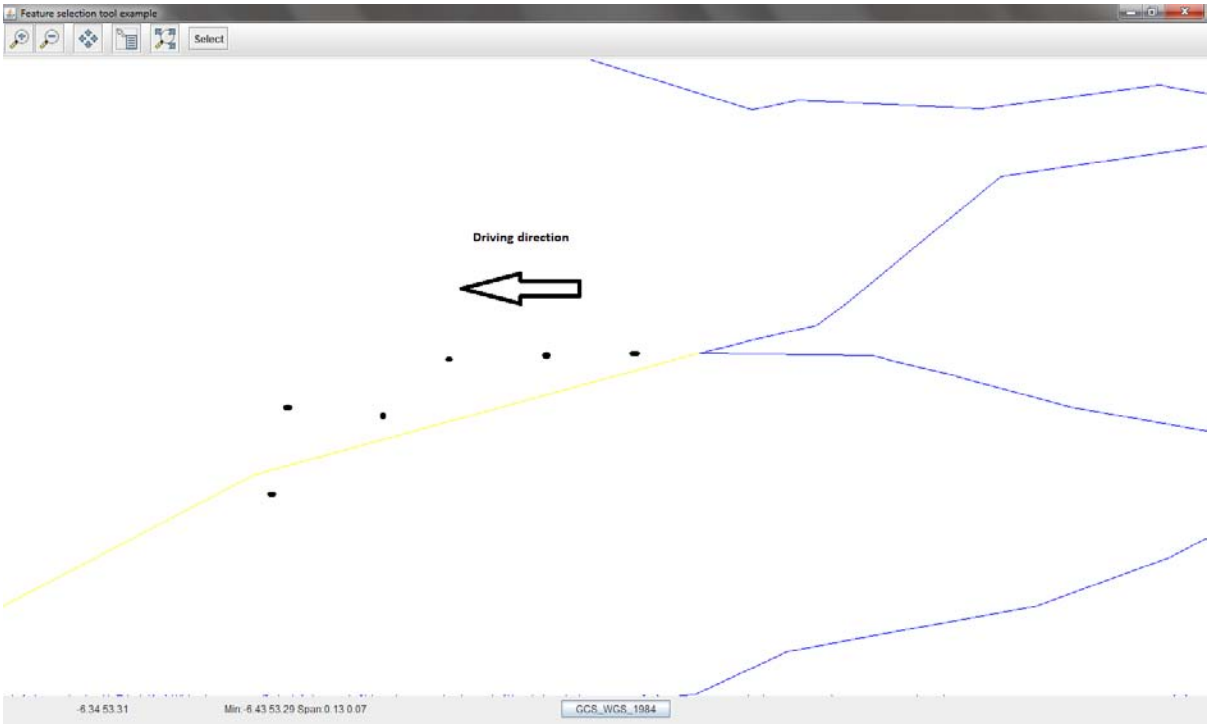


Figure 40: Overtaking event on straight road

The GPS data we picked for this particular overtaking event is as follows:

Latitudes	-6.3657	-6.3739	-6.3840	-6.3869	-6.3975	-6.4033
Longitudes	53.3265	53.3262	53.3266	53.3211	53.3223	53.3131

The results we got out of our system are as follows:

```
Shapefile is loaded ...
Best match result: Road ID =IRL_roads.229 Closest Distance =0.0017236420464667671 Segment = LINESTRING( -6.356972274114281
53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.003606716104822593 Segment = LINESTRING( -6.356972274114281
53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.006668129743720799 Segment = LINESTRING( -6.356972274114281
53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.0021329165267686308 Segment = LINESTRING( -6.356972274114281
53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.006098206859348905 Segment = LINESTRING( -6.356972274114281
53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.0012365335318987164 Segment = LINESTRING( -6.356972274114281
53.327110325799126, -6.404694585025053 53.31399920682364)
Road Id: IRL_roads.229 count: 6
Elected Road Id = IRL_roads.229
GPS RMS point (-6.385229602371398, 53.32263354705329, NaN)
Closest road segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Closest road segment fraction = LINESTRING( -6.365243370423086 53.324837943643004, -6.4036275841319075 53.31429235221811)
Vehicle's path segment fraction (RMS) = LINESTRING( -6.3657 53.3265, -6.385229602371398 53.32263354705329)
Segment fraction Angle with X-Axis = -164
Vehicle's angle with X-Axis = -168
----- Overtaking -----
```

The output result shows the 6 GPS points and their closest distance to the best matched road segment. It also shows that there was only one road segment involved in the search and there were no GPS points lie anywhere near any other segments. The elected road segment was IRL_roads.229, it is the only segment involved so it is natural that it gets all the votes. An important value to notice in the output is the RMS/effective value of all the GPS points; this point formulates a perfect straight line with the first point of the right turn in the overtaking pattern. The line will have a certain angle and this angle will be used later as the vehicle's effective angle. Finally are the angle of the road segment fraction and the angle of the vehicle itself. The result shows that the vehicle had made an angle that is 4 degrees less than the road angle and confirms that this is indeed an overtaking. Figure [41] shows a snapshot of the same system output data.

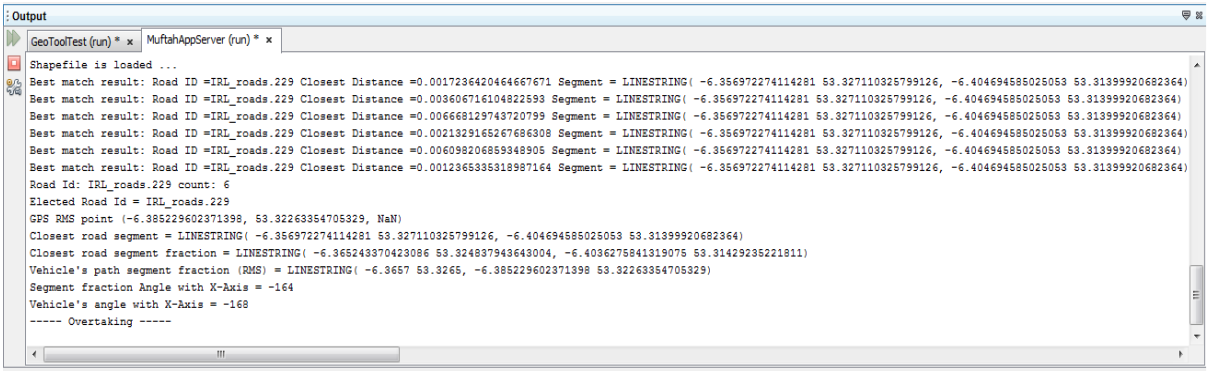


Figure 41: Snapshot of overtaking output on a straight road

The second test scenario that we are going to test is when an overtaking takes place immediately after a right turn and involves two road segments. Figure [42] shows the road segments and the GPS points where this overtaking had taken place.

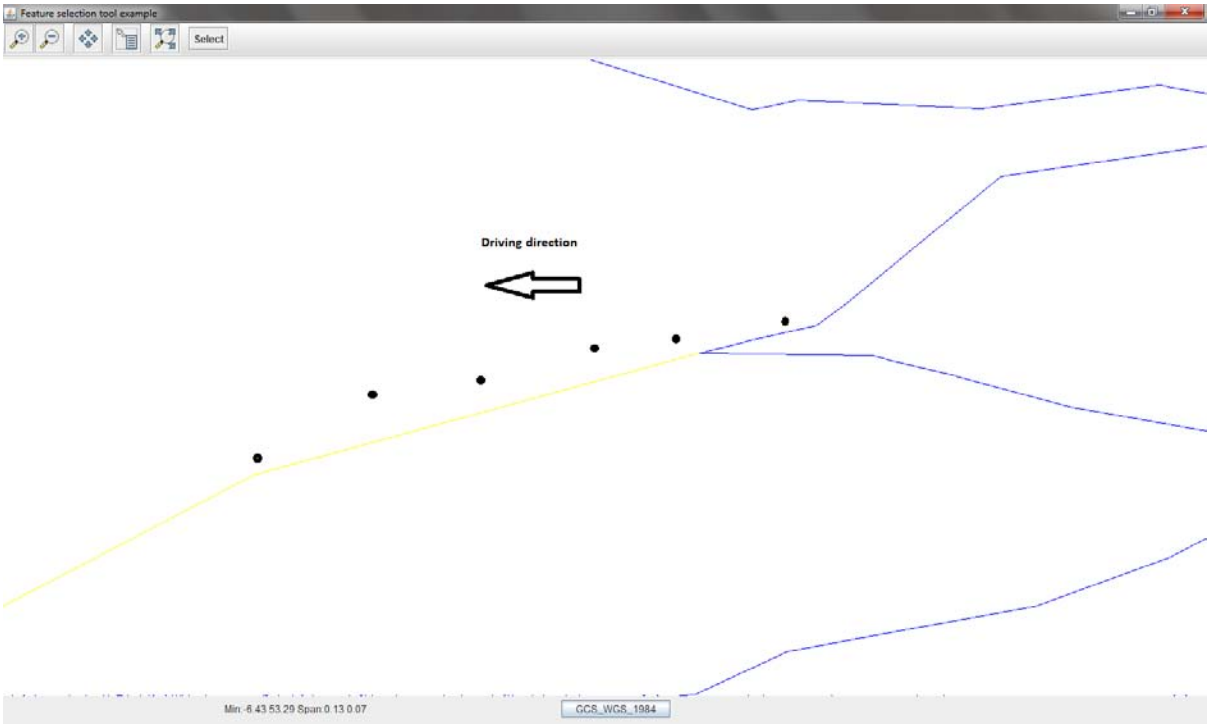


Figure 42: Overtaking immediately after a right turn with two segments involved

The GPS data we picked for this particular overtaking event is as follows:

Latitudes	-6.3469	-6.3621	-6.3728	-6.3848	-6.3947	-6.4042
Longitudes	53.3312	53.3276	53.3252	53.3232	53.3190	53.3162

The results we got out of our system are as follows:

```
Shapefile is loaded ...
Best match result: Road ID =IRL_roads.201 Closest Distance =0.0016663656351282566 Segment = LINESTRING( -6.344361249137861 53.33008197279357, -6.352083222123423 53.32827758979694)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.0018306222145203735 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.0023510325967323674 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.003601548742698503 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.002174336002689256 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =0.001991132756078321 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Road Id: IRL_roads.229 count: 5
Road Id: IRL_roads.201 count: 1
Elected Road Id = IRL_roads.229
GPS RMS point (-6.377612832269872, 53.32373357133451, NaN)
Closest road segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Closest road segment fraction = LINESTRING( -6.356972274114281 53.327110325799126, -6.403672506423331 53.314280010370105)
Vehicle's path segment fraction (RMS) = LINESTRING( -6.3469 53.3312, -6.377612832269872 53.32373357133451)
Segment fraction Angle with X-Axis = -164
Vehicle's angle with X-Axis = -166
----- Overtaking -----
```

The output shows the two segments involved and which one was actually elected. The output also confirms that this is indeed an overtaking event and shows that the angle of the vehicle is 2 degrees less than the road angle. Figure [43] shows a snapshot of the same system output data.

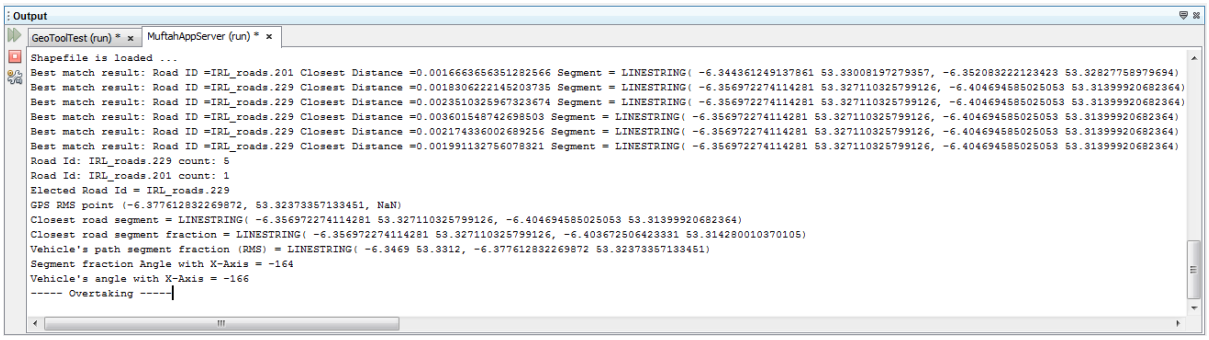


Figure 43: Snapshot of overtaking with two road segments involved

Right turns (Not overtaking)

What we mean here by right turns is that when a client sends an initial overtaking data to the server to be confirmed the server through map matching realise this is actually a right turn and not overtaking. In order to test this we sat up two scenarios, one with the right turn happens on one road segment and the other with a right turn happens on two segments. Figure [44] show the first scenario and all the GPS points.

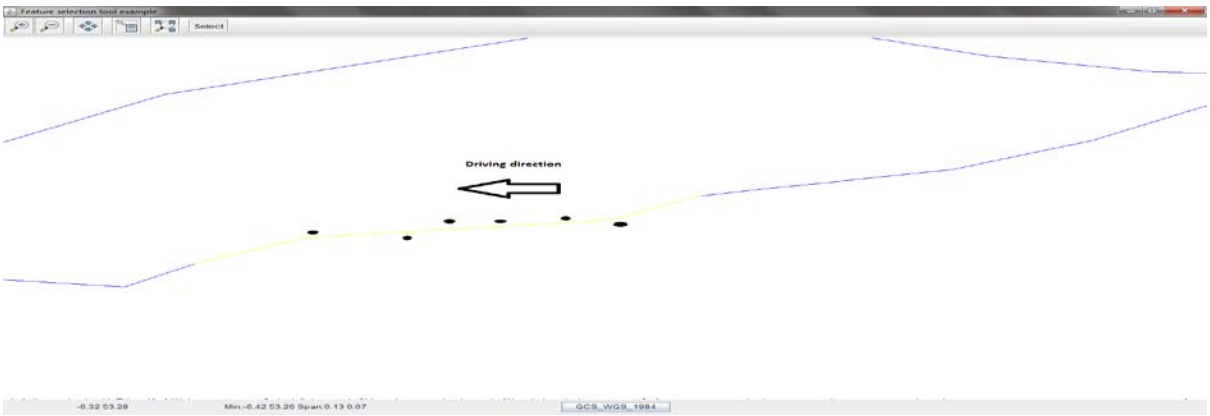


Figure 44: Right turn with one segment involved

The GPS data we picked for this particular overtaking event is as follows:

Latitudes	-6.3568	-6.3635	-6.3684	-6.3767	-6.3815	-6.3889
Longitudes	53.2909	53.2914	53.2889	53.2898	53.2873	53.2884

The results we got out of our system are as follows:

Shapefile is loaded ...

Best match result: Road ID =IRL_roads.215 Closest Distance =5.355703133895174E-5 Segment = LINESTRING(-6.3573054811136585 53.29072196686716, -6.347777802131473 53.2951927618588)

Best match result: Road ID =IRL_roads.215 Closest Distance =0.0013080085471651529 Segment = LINESTRING(-6.389527713053411 53.287418435873334, -6.359444583109659 53.290500647867574)

Best match result: Road ID =IRL_roads.215 Closest Distance =6.795498401306421E-4 Segment = LINESTRING(-6.389527713053411 53.287418435873334, -6.359444583109659 53.290500647867574)

Best match result: Road ID =IRL_roads.215 Closest Distance =0.0010617235486410487 Segment = LINESTRING(-6.389527713053411 53.287418435873334, -6.359444583109659 53.290500647867574)

Best match result: Road ID =IRL_roads.215 Closest Distance =9.360271316657936E-4 Segment = LINESTRING(-6.389527713053411 53.287418435873334, -6.359444583109659 53.290500647867574)

Best match result: Road ID =IRL_roads.215 Closest Distance =9.124740795565979E-4 Segment = LINESTRING(-6.389527713053411 53.287418435873334, -6.359444583109659 53.290500647867574)

Road Id: IRL_roads.215 count: 6

Elected Road Id = IRL_roads.215

GPS RMS point (-6.372642641270051, 53.28945001885145, NaN)

Closest road segment = LINESTRING(-6.389527713053411 53.287418435873334, -6.359444583109659 53.290500647867574)

Closest road segment fraction = LINESTRING(-6.359444583109659 53.290500647867574, -6.388806997968096 53.28749227781347)

Vehicle's path segment fraction (RMS) = LINESTRING(-6.3568 53.2909, -6.372642641270051 53.28945001885145)

Segment fraction Angle with X-Axis = -174

Vehicle's angle with X-Axis = -174

----- Right turn -----

The output result shows that there was only one segment involved, which is road segment IRL_roads.215, and it got all the votes. The result also shows that this is actually not an overtaking but instead it is a right turn. This is clearly shown by the zero difference between the road angle and the vehicle's angle. Figure [45] shows a snapshot of the same output data.

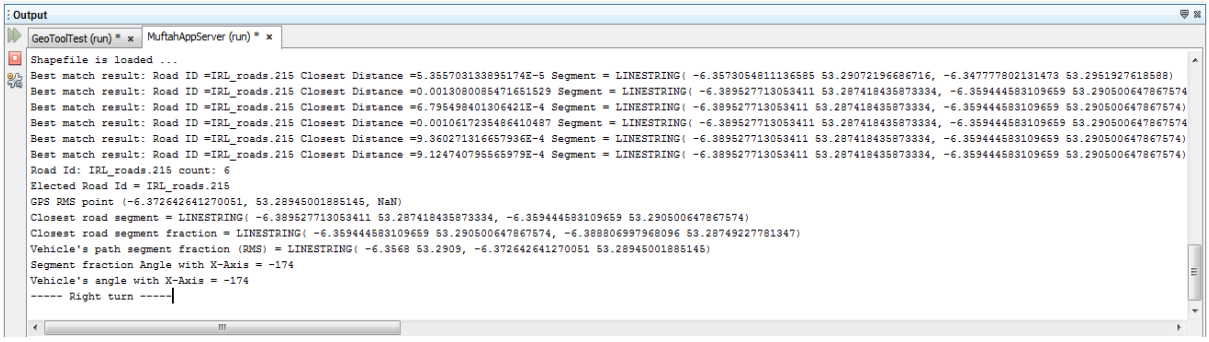


Figure 45: Snapshot of the right turn with one segment involved

The second scenario we are going to test is when a right turn happens along two road segments. Figure [46] shows the road segments that were chosen for this scenario and all the GPS points.

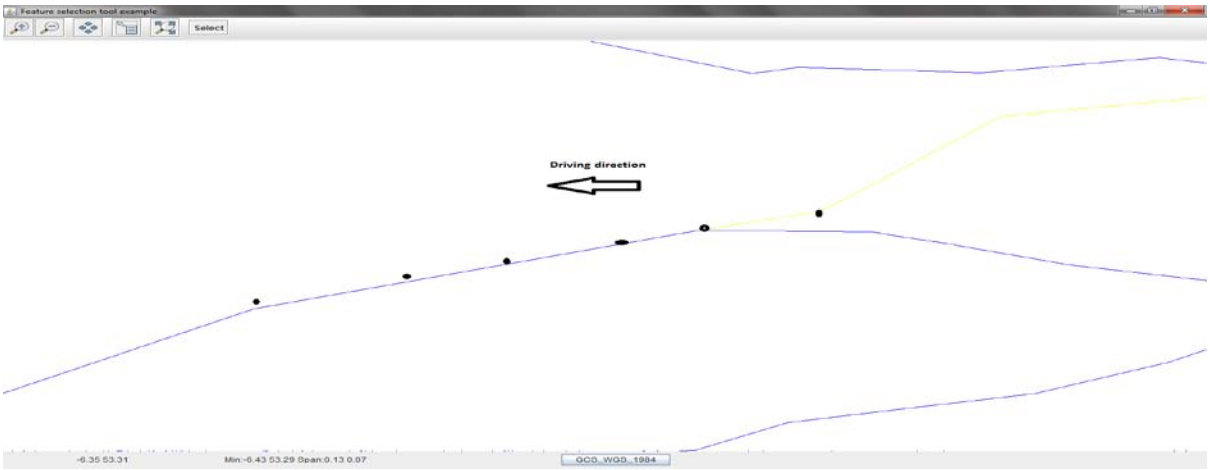


Figure 46: Right turn with two segments involved

The GPS data we picked for this particular overtaking event is as follows:

Latitudes	-6.3448	-6.3567	-6.3653	-6.3756	-6.3881	-6.4041
Longitudes	53.3298	53.3273	53.3249	53.3221	53.3185	53.3143

The results we got out of our system are as follows:

```
Shapefile is loaded ...
Best match result: Road ID =IRL_roads.201 Closest Distance =1.74743290134283E-4 Segment = LINESTRING( -6.344361249137861 53.33008197279357, -6.352083222123423 53.32827758979694)
Best match result: Road ID =IRL_roads.201 Closest Distance =1.2130993407379627E-4 Segment = LINESTRING( -6.354750201118436 53.32764047079813, -6.356972274114281 53.327110325799126)
Best match result: Road ID =IRL_roads.229 Closest Distance =7.484146876409056E-5 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =1.0357530171106131E-4 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =5.627991551627727E-5 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Best match result: Road ID =IRL_roads.229 Closest Distance =1.3252757610369 64E-4 Segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Road Id: IRL_roads.229 count: 4
Road Id: IRL_roads.201 count: 2
Elected Road Id = IRL_roads.229
GPS RMS point (-6.372463675847827, 53.3228169243555, NaN)
Closest road segment = LINESTRING( -6.356972274114281 53.327110325799126, -6.404694585025053 53.31399920682364)
Closest road segment fraction = LINESTRING( -6.356972274114281 53.327110325799126, -6.404064890615699 53.31417220763105)
Vehicle's path segment fraction (RMS) = LINESTRING( -6.3448 53.3298, -6.372463675847827 53.3228169243555)
Segment fraction Angle with X-Axis = -164
Vehicle's angle with X-Axis = -165
----- Right turn -----
```

The output result confirms the fact that there were actually two segments involved and the one that got the majority of votes is the IRL_roads.229. it also confirms that when there is only degree difference between the road angle and the vehicle's angle i.e. within the threshold limit, the angles are considered to be the same and the event is considered to be a right turn and not overtaking.

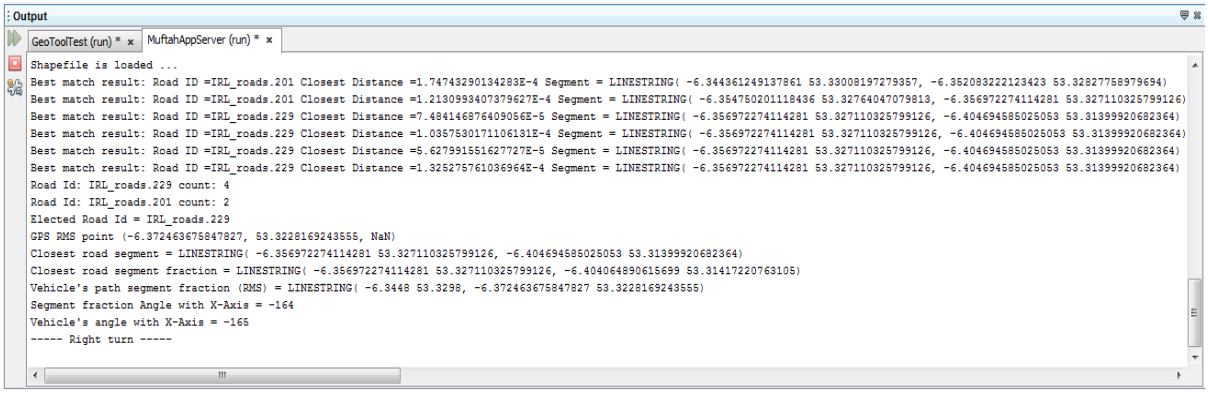


Figure 47: Snapshot of right turn with two segments involved

CHAPTER 6: CONCLUSION

General Conclusions

In this dissertation we investigated the possibility of using a mobile phone as an off-the-shelf technology to detect erratic driving behaviour caused by vehicle overtaking in Irish main roads and motorways. We looked into how this could be achieved and implemented through the built-in phone capabilities and what are the limits of these capabilities. We then developed methods and techniques to enhance these capabilities even further and provide a proof-of-concept solution that we believe it has the potential to provide a good foundation for future studies and developments in this field.

We started our investigation on the client side by looking into how the phone built-in orientation sensor could be used to detect vehicle's trajectory movements and what level of accuracy it can provide. We found that the orientation sensor is indeed capable of detecting all possible movements of the vehicle and with the right choice of the sensing level and speed, that are provided by the phone itself, and the right choice of sensing threshold level, that should be set by us, to filter any unnecessary sensors jitters and glitches, we can actually obtain a very good level of reliability out of these sensors. Even though, all the tests we carried out on the orientation sensor were in fact a theoretical and in an indoor environment, that doesn't dismiss the fact that an object that behaves like a steering wheel with the phone attached to it or using the phone itself as a steering wheel in an indoor environment cannot represent a real steering wheel in an actual vehicle.

In order to decide whether the vehicle has moved to the left or to the right or in a straight line direction we developed a number of movement filters that are specifically designed to only filter any of these movements and create a digital set that digitally represents them. These digital sets are then used as inputs to specialised artificial neural networks that are also developed and designed by us and well trained to decide if any of this gathered sets of movements is actually a possible right turn or left turn or initially an overtaking. The use of neural networks in our system have served us greatly and provided us with a form of artificial intelligence that we won't be able to come up with easily using only the normal software logic. Their abilities of recognising patterns and becoming an expert when they are well trained and provided with the right full set of inputs and expected outputs had made

the process of detecting and recognising all the possible overtaking patterns a lot easier than we first anticipated. The tests we carried out have proven this fact and shown a correct detection and recognition of all the overtaking patterns.

After all the detection of any possible overtaking patterns have been covered, it was time for us then to move on to the server side and look into all the possible available technologies that can help us in obtaining real time information about the location of the overtaking, and confirm whether this event is actually an overtaking or it is just the shape of the road that happens to look like an overtaking manoeuvre i.e. a right turn followed by a left turn and a straight line. We used GIS along with a digital map of the Irish roads network to work out where the event of overtaking had happened and what exactly was the angle of the vehicle when it had made the right turn to overtake another vehicle. We used GeoTools to geometrically obtain the angle of the vehicle and compare it to the angle of the road segment itself where the overtaking event had happened. Through this comparison of angles we were able to determine whether this event was actually an overtaking or it was just a right turn. We noticed that during a normal right turn the vehicle's angle is always either equal to the road angle or slightly greater, but during overtaking the vehicle's angle is always less than the road angle. The difference between the two angles has to be either greater or less than a threshold angle which is one degree from both sides of the road.

Due to the fact that the GPS accuracy is not very reliable in providing exact location coordinates on the road of the vehicle's full right turn movement, we decided to use the effective value of all the alternating GPS points about the road segment, and utilise this value along with the first GPS point of the right turn to form a perfect straight line that represents the vehicle's path during the right turn part of the overtaking. Another defect the could be caused by GPS limited accuracy is that in certain situations any of the reported coordinates by the GPS could lie exactly between two road segments or some of them could in fact lie closer to one segment than the other, especially when that segment is actually the wrong segment. This had led us to develop a technique to overcome this difficulty. The technique is simply a voting algorithm that only elects the road segment with the majority of short distances from each point to the segment itself. After testing this technique with different scenarios we found that this technique was proven to be effective, even though in

motorways it is very rare to find two different roads that are continuously close and in perfectly parallel to each other but that doesn't mean we shouldn't count for it.

Fulfilled Objectives and Contributions

Regardless of the fact that all the tests that we have carried out are all done in an indoor environment with the phone being used as a steering wheel and performing a virtual driving, we believe this could give a very good and almost reliable indication of how the phone will be used in a real environment as both movements of the phone in a real environment or a virtual one, are literally, the same.

We believe that our investigation has achieved its goals and provided a study that could open doors to the possibility of using off-the-shelf technologies in detecting erratic driving behaviours.

A good number of methods and techniques have been developed in this project that could contribute, even a little, to the research and development community in the field of Smart Roads and Intelligent Motorways.

Future Work

Future work and further development is definitely needed to eliminate any dependency on GPS, as it can't be available everywhere under every condition, even though all we needed from GPS in this study is an indication of where the vehicle is heading, nonetheless it still lacks a proper accuracy for more location sensitive applications and require more work from developers to compensate for that. More work is required to eliminate the need for deploying and running GIS frameworks on dedicated servers which causes clients to generate unnecessary traffic, even if that traffic is actually small but still could be avoided through relying on cheaper and easier means, such as deploying and running the GIS tools on ad-hoc based system. The use of vehicle-to-vehicle ad-hoc based communication to get rid of any reliance on dedicated network infrastructures is a must. Ad-hoc vehicle-to-vehicle communication could also be used to determine if the vehicle is actually overtaking another vehicle and it is not just trying to avoid an object on the road, or simply had made a manoeuvre that looked like an overtaking manoeuvre due to road work diversion in the road.

APPENDECES

Appendix A: Neural networks inputs and outputs

A.1 Right turn neural network inputs and outputs

1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0	1.0 0.0 1.0 1.0 1.0 2.0 0.0	1.0 1.0 1.0 2.0 1.0 0.0 0.0	1.0 1.0 0.0 0.0 0.0 2.0 1.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0	1.0 0.0 1.0 1.0 2.0 0.0 0.0	1.0 1.0 1.0 2.0 1.0 1.0 0.0	1.0 1.0 0.0 0.0 0.0 2.0 2.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 2.0 0.0	1.0 0.0 1.0 1.0 2.0 1.0 0.0	1.0 1.0 1.0 2.0 1.0 2.0 0.0	1.0 1.0 0.0 1.0 0.0 0.0 0.0 1.0
1.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0	1.0 0.0 1.0 1.0 2.0 2.0 0.0	1.0 1.0 1.0 2.0 2.0 0.0 0.0	1.0 1.0 0.0 1.0 0.0 0.1 1.0
1.0 0.0 0.0 0.0 0.0 1.0 1.0 1.0	1.0 0.0 1.0 2.0 0.0 0.0 0.0	1.0 1.0 1.0 2.0 2.0 1.0 0.0	1.0 1.0 0.0 1.0 0.0 0.2 0.0
1.0 0.0 0.0 0.0 0.0 1.0 2.0 0.0	1.0 0.0 1.0 2.0 0.0 1.0 0.0	1.0 1.0 1.0 2.0 2.0 2.0 0.0	1.0 1.0 0.0 1.0 1.0 0.0 1.0
1.0 0.0 0.0 0.0 0.0 2.0 0.0 0.0	1.0 0.0 1.0 2.0 0.0 2.0 0.0	1.0 2.0 0.0 2.0 0.0 0.2 0.0	1.0 1.0 0.0 1.0 1.0 1.0 1.0
1.0 0.0 0.0 0.0 0.0 2.0 1.0 0.0	1.0 0.0 1.0 2.0 1.0 0.0 0.0	1.0 2.0 0.0 2.0 1.0 0.0 0.0	1.0 1.0 0.0 1.0 1.0 2.0 0.0
1.0 0.0 0.0 0.0 0.0 2.0 2.0 0.0	1.0 0.0 1.0 2.0 1.0 1.0 0.0	1.0 2.0 0.0 2.0 1.0 1.0 0.0	1.0 1.0 0.0 1.0 2.0 0.0 0.0
1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0	1.0 0.0 1.0 2.0 1.0 2.0 0.0	1.0 2.0 0.0 2.0 1.0 2.0 0.0	1.0 1.0 0.0 1.0 2.0 1.0 0.0
1.0 0.0 0.0 1.0 0.0 0.1 1.0 1.0	1.0 0.0 1.0 2.0 2.0 0.0 0.0	1.0 2.0 0.0 2.0 2.0 0.0 0.0	1.0 1.0 0.0 1.0 2.0 2.0 0.0
1.0 0.0 0.0 1.0 0.0 2.0 0.0	1.0 0.0 1.0 2.0 2.0 1.0 0.0	1.0 2.0 0.0 2.0 2.0 1.0 0.0	1.0 1.0 0.0 2.0 0.0 0.0 0.0
1.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0	1.0 0.0 1.0 2.0 2.0 2.0 0.0	1.0 2.0 0.0 2.0 2.0 2.0 0.0	1.0 1.0 0.0 2.0 0.0 0.1 0.0
1.0 0.0 0.0 1.0 1.0 1.0 1.0 1.0	1.0 0.0 2.0 0.0 0.0 0.0 0.0	1.0 2.0 1.0 0.0 0.0 0.0 0.0	1.0 1.0 0.0 2.0 0.0 0.2 0.0
1.0 0.0 0.0 1.0 1.0 2.0 0.0	1.0 0.0 2.0 0.0 0.0 0.1 0.0	1.0 2.0 1.0 0.0 0.0 0.1 0.0	1.0 1.0 0.0 2.0 1.0 0.0 0.0
1.0 0.0 0.0 1.0 2.0 0.0 0.0	1.0 0.0 2.0 0.0 0.0 2.0 0.0	1.0 2.0 1.0 0.0 0.0 2.0 0.0	1.0 1.0 0.0 2.0 1.0 1.0 0.0
1.0 0.0 0.0 1.0 2.0 1.0 0.0	1.0 0.0 2.0 0.0 1.0 0.0 0.0	1.0 2.0 1.0 0.0 1.0 0.0 0.0	1.0 1.0 0.0 2.0 1.0 2.0 0.0
1.0 0.0 0.0 1.0 2.0 2.0 0.0	1.0 0.0 2.0 0.0 1.0 1.0 0.0	1.0 2.0 1.0 0.0 1.0 1.0 0.0	1.0 1.0 0.0 2.0 2.0 0.0 0.0
1.0 0.0 0.0 2.0 0.0 0.0 0.0	1.0 0.0 2.0 0.0 1.0 2.0 0.0	1.0 2.0 1.0 0.0 1.0 2.0 0.0	1.0 1.0 0.0 2.0 2.0 1.0 0.0
1.0 0.0 0.0 2.0 0.0 1.0 0.0	1.0 0.0 2.0 0.0 2.0 0.0 0.0	1.0 2.0 1.0 0.0 2.0 0.0 0.0	1.0 1.0 0.0 2.0 2.0 2.0 0.0
1.0 0.0 0.0 2.0 0.0 2.0 0.0	1.0 0.0 2.0 0.0 2.0 1.0 0.0	1.0 2.0 1.0 0.0 2.0 1.0 0.0	1.0 1.0 1.0 0.0 0.0 0.0 1.0
1.0 0.0 0.0 2.0 1.0 0.0 0.0	1.0 0.0 2.0 0.0 2.0 2.0 0.0	1.0 2.0 1.0 0.0 2.0 2.0 0.0	1.0 1.0 1.0 0.0 0.0 1.0 1.0
1.0 0.0 0.0 2.0 1.0 1.0 0.0	1.0 0.0 2.0 1.0 0.0 0.0 0.0	1.0 2.0 1.0 1.0 0.0 0.0 0.0	1.0 1.0 1.0 0.0 0.0 0.2 0.0
1.0 0.0 0.0 2.0 1.0 2.0 0.0	1.0 0.0 2.0 1.0 0.0 1.0 0.0	1.0 2.0 1.0 1.0 0.0 1.0 0.0	1.0 1.0 1.0 0.0 1.0 0.0 1.0
1.0 0.0 0.0 2.0 2.0 0.0 0.0	1.0 0.0 2.0 1.0 0.0 2.0 0.0	1.0 2.0 1.0 1.0 0.0 2.0 0.0	1.0 1.0 1.0 0.0 1.0 1.0 1.0
1.0 0.0 0.0 2.0 2.0 1.0 0.0	1.0 0.0 2.0 1.0 1.0 0.0 0.0	1.0 2.0 1.0 1.0 1.0 0.0 0.0	1.0 1.0 1.0 0.0 1.0 2.0 0.0
1.0 0.0 0.0 2.0 2.0 2.0 0.0	1.0 0.0 2.0 1.0 1.0 1.0 0.0	1.0 2.0 1.0 1.0 1.0 1.0 0.0	1.0 1.0 1.0 0.0 2.0 0.0 0.0
1.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0	1.0 0.0 2.0 1.0 1.0 2.0 0.0	1.0 2.0 1.0 1.0 1.0 2.0 0.0	1.0 1.0 1.0 0.0 2.0 1.0 0.0
1.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0	1.0 0.0 2.0 1.0 2.0 0.0 0.0	1.0 2.0 1.0 1.0 2.0 0.0 0.0	1.0 1.0 1.0 0.0 2.0 2.0 0.0
1.0 0.0 1.0 0.0 0.0 0.0 2.0 0.0	1.0 0.0 2.0 1.0 2.0 1.0 0.0	1.0 2.0 1.0 1.0 2.0 1.0 0.0	1.0 1.0 1.0 1.0 0.0 0.0 0.0 1.0
1.0 0.0 1.0 0.0 0.1 0.0 0.0 1.0	1.0 0.0 2.0 1.0 2.0 2.0 0.0	1.0 2.0 1.0 1.0 2.0 2.0 0.0	1.0 1.0 1.0 1.0 0.0 0.1 0.0 1.0
1.0 0.0 1.0 0.0 1.0 1.0 1.0 1.0	1.0 0.0 2.0 2.0 0.0 0.0 0.0	1.0 2.0 1.0 2.0 0.0 0.0 0.0	1.0 1.0 1.0 1.0 0.0 0.2 0.0
1.0 0.0 1.0 0.0 1.0 2.0 0.0	1.0 0.0 2.0 2.0 0.0 0.1 0.0	1.0 2.0 1.0 2.0 0.0 0.1 0.0	1.0 1.0 1.0 1.0 1.0 0.0 0.0 1.0
1.0 0.0 1.0 0.0 2.0 0.0 0.0	1.0 0.0 2.0 2.0 0.0 2.0 0.0	1.0 2.0 1.0 2.0 0.0 2.0 0.0	1.0 1.0 1.0 1.0 1.0 1.0 1.0
1.0 0.0 1.0 0.0 2.0 1.0 0.0	1.0 0.0 2.0 2.0 1.0 0.0 0.0	1.0 2.0 1.0 2.0 1.0 0.0 0.0	1.0 1.0 1.0 1.0 1.0 2.0 0.0
1.0 0.0 1.0 0.0 2.0 2.0 0.0	1.0 0.0 2.0 2.0 1.0 1.0 0.0	1.0 2.0 1.0 2.0 1.0 1.0 0.0	1.0 1.0 1.0 1.0 2.0 0.0 0.0
1.0 0.0 1.0 1.0 0.0 0.0 0.0 1.0	1.0 0.0 2.0 2.0 1.0 2.0 0.0	1.0 2.0 1.0 2.0 1.0 2.0 0.0	1.0 1.0 1.0 1.0 2.0 1.0 0.0
1.0 0.0 1.0 1.0 0.0 0.1 0.0 1.0	1.0 0.0 2.0 2.0 2.0 0.0 0.0	1.0 2.0 1.0 2.0 2.0 0.0 0.0	1.0 1.0 1.0 1.0 2.0 2.0 0.0
1.0 0.0 1.0 1.0 0.0 2.0 0.0	1.0 0.0 2.0 2.0 2.0 1.0 0.0	1.0 2.0 1.0 2.0 2.0 1.0 0.0	1.0 1.0 1.0 2.0 0.0 0.0 0.0
1.0 0.0 1.0 1.0 1.0 0.0 0.0 1.0	1.0 0.0 2.0 2.0 2.0 2.0 0.0	1.0 2.0 1.0 2.0 2.0 2.0 0.0	1.0 1.0 1.0 2.0 0.0 0.1 0.0
1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0	1.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0	1.0 2.0 2.0 0.0 0.0 0.0 0.0	1.0 1.0 1.0 2.0 0.0 0.2 0.0
1.0 1.0 2.0 0.0 0.0 0.0 0.0 0.0	1.0 1.0 2.0 2.0 0.0 0.0 0.0	1.0 2.0 2.0 0.0 0.0 0.1 0.0	1.0 2.0 0.0 0.0 2.0 2.0 0.0

1.0 1.0 2.0 0.0 0.0 1.0 0.0	1.0 1.0 2.0 2.0 0.0 1.0 0.0	1.0 2.0 2.0 0.0 0.0 2.0 0.0	1.0 2.0 0.0 1.0 0.0 0.0 0.0
1.0 1.0 2.0 0.0 0.0 2.0 0.0	1.0 1.0 2.0 2.0 0.0 2.0 0.0	1.0 2.0 2.0 0.0 1.0 0.0 0.0	1.0 2.0 0.0 1.0 0.0 1.0 0.0
1.0 1.0 2.0 0.0 1.0 0.0 0.0	1.0 1.0 2.0 2.0 1.0 0.0 0.0	1.0 2.0 2.0 0.0 1.0 1.0 0.0	1.0 2.0 0.0 1.0 0.0 2.0 0.0
1.0 1.0 2.0 0.0 1.0 1.0 0.0	1.0 1.0 2.0 2.0 1.0 1.0 0.0	1.0 2.0 2.0 0.0 1.0 2.0 0.0	1.0 2.0 0.0 1.0 1.0 0.0 0.0
1.0 1.0 2.0 0.0 1.0 2.0 0.0	1.0 1.0 2.0 2.0 1.0 2.0 0.0	1.0 2.0 2.0 0.0 2.0 0.0 0.0	1.0 2.0 0.0 1.0 1.0 1.0 0.0
1.0 1.0 2.0 0.0 2.0 0.0 0.0	1.0 1.0 2.0 2.0 2.0 0.0 0.0	1.0 2.0 2.0 0.0 2.0 1.0 0.0	1.0 2.0 0.0 1.0 1.0 2.0 0.0
1.0 1.0 2.0 0.0 2.0 1.0 0.0	1.0 1.0 2.0 2.0 2.0 1.0 0.0	1.0 2.0 2.0 0.0 2.0 2.0 0.0	1.0 2.0 0.0 1.0 2.0 0.0 0.0
1.0 1.0 2.0 0.0 2.0 2.0 0.0	1.0 1.0 2.0 2.0 2.0 2.0 0.0	1.0 2.0 2.0 1.0 0.0 0.0 0.0	1.0 2.0 0.0 1.0 2.0 1.0 0.0
1.0 1.0 2.0 1.0 0.0 0.0 0.0	1.0 2.0 0.0 0.0 0.0 0.0 0.0	1.0 2.0 2.0 1.0 0.0 1.0 0.0	1.0 2.0 0.0 1.0 2.0 2.0 0.0
1.0 1.0 2.0 1.0 0.0 0.1 0.0	1.0 2.0 0.0 0.0 0.0 0.1 0.0	1.0 2.0 2.0 1.0 0.0 2.0 0.0	1.0 2.0 0.0 2.0 0.0 0.0 0.0
1.0 1.0 2.0 1.0 0.0 2.0 0.0	1.0 2.0 0.0 0.0 0.0 0.2 0.0	1.0 2.0 2.0 1.0 1.0 0.0 0.0	1.0 2.0 0.0 2.0 0.0 0.1 0.0
1.0 1.0 2.0 1.0 1.0 0.0 0.0	1.0 2.0 0.0 0.0 0.1 0.0 0.0	1.0 1.0 0.0 0.0 0.0 0.1 1.0	1.0 2.0 2.0 1.0 1.0 1.0 0.0
1.0 1.0 2.0 1.0 1.0 1.0 0.0	1.0 2.0 0.0 0.0 0.1 0.1 0.0	1.0 1.0 0.0 0.0 0.0 0.2 0.0	1.0 2.0 2.0 1.0 1.0 2.0 0.0
1.0 2.0 0.0 0.0 2.0 0.0 0.0	1.0 2.0 0.0 0.0 0.1 0.2 0.0	1.0 1.0 0.0 0.0 1.0 0.0 1.0	1.0 2.0 2.0 1.0 2.0 0.0 0.0
1.0 2.0 0.0 0.0 2.0 1.0 0.0	1.0 1.0 0.0 0.0 0.2 0.0 0.0	1.0 1.0 0.0 0.0 0.1 1.0 1.0	1.0 2.0 2.0 1.0 2.0 1.0 0.0
1.0 1.0 2.0 1.0 1.0 2.0 0.0	1.0 2.0 2.0 2.0 1.0 2.0 0.0	1.0 1.0 0.0 0.0 1.0 2.0 0.0	1.0 2.0 2.0 1.0 2.0 2.0 0.0
1.0 1.0 2.0 1.0 2.0 0.0 0.0	1.0 2.0 2.0 2.0 2.0 0.0 0.0	1.0 2.0 2.0 2.0 0.0 2.0 0.0	1.0 2.0 2.0 2.0 0.0 0.0 0.0
1.0 1.0 2.0 1.0 2.0 1.0 0.0	1.0 2.0 2.0 2.0 2.0 1.0 0.0	1.0 2.0 2.0 2.0 1.0 0.0 0.0	1.0 2.0 2.0 2.0 0.0 1.0 0.0
1.0 1.0 2.0 1.0 2.0 2.0 0.0	1.0 2.0 2.0 2.0 2.0 2.0 0.0	1.0 2.0 2.0 2.0 1.0 1.0 0.0	

A.2 Left turn neural network inputs and outputs

2.0 0.0 0.0 0.0 0.0 0.0 1.0	2.0 0.0 1.0 1.0 1.0 0.0 0.0	2.0 0.0 2.0 2.0 2.0 2.0 1.0	2.0 1.0 1.0 1.0 1.0 1.0 0.0
2.0 0.0 0.0 0.0 0.0 0.1 0.0	2.0 0.0 1.0 1.0 1.0 1.0 0.0	2.0 1.0 0.0 0.0 0.0 0.0 0.0	2.0 1.0 1.0 1.0 1.0 2.0 0.0
2.0 0.0 0.0 0.0 0.0 0.2 1.0	2.0 0.0 1.0 1.0 1.0 2.0 0.0	2.0 1.0 0.0 0.0 0.0 0.1 0.0	2.0 1.0 1.0 1.0 2.0 0.0 0.0
2.0 0.0 0.0 0.0 0.1 0.0 0.0	2.0 0.0 1.0 1.0 2.0 0.0 0.0	2.0 1.0 0.0 0.0 0.0 0.2 0.0	2.0 1.0 1.0 1.0 2.0 1.0 0.0
2.0 0.0 0.0 0.0 0.1 1.0 0.0	2.0 0.0 1.0 1.0 2.0 1.0 0.0	2.0 1.0 0.0 0.0 0.1 0.0 0.0	2.0 1.0 1.0 1.0 2.0 2.0 0.0
2.0 0.0 0.0 0.0 0.1 0.2 0.0	2.0 0.0 1.0 1.0 2.0 2.0 0.0	2.0 1.0 0.0 0.0 0.1 1.0 0.0	2.0 1.0 1.0 2.0 0.0 0.0 0.0
2.0 0.0 0.0 0.0 0.2 0.0 1.0	2.0 0.0 1.0 2.0 0.0 0.0 0.0	2.0 1.0 0.0 0.0 0.1 0.2 0.0	2.0 1.0 1.0 2.0 0.0 1.0 0.0
2.0 0.0 0.0 0.0 0.2 0.1 0.0	2.0 0.0 1.0 2.0 0.0 1.0 0.0	2.0 1.0 0.0 0.0 0.2 0.0 0.0	2.0 1.0 1.0 2.0 0.0 2.0 0.0
2.0 0.0 0.0 0.0 0.2 0.2 1.0	2.0 0.0 1.0 2.0 0.0 2.0 0.0	2.0 1.0 0.0 0.0 0.2 0.1 0.0	2.0 1.0 1.0 2.0 1.0 0.0 0.0
2.0 0.0 0.0 1.0 0.0 0.0 0.0	2.0 0.0 1.0 2.0 1.0 0.0 0.0	2.0 1.0 0.0 0.0 0.2 0.2 0.0	2.0 1.0 1.0 2.0 1.0 1.0 0.0
2.0 0.0 0.0 1.0 0.0 0.1 0.0	2.0 0.0 1.0 2.0 1.0 1.0 0.0	2.0 1.0 0.0 1.0 0.0 0.0 0.0	2.0 1.0 1.0 2.0 1.0 2.0 0.0
2.0 0.0 0.0 1.0 0.0 0.2 0.0	2.0 0.0 1.0 2.0 1.0 2.0 0.0	2.0 1.0 0.0 1.0 0.0 0.1 0.0	2.0 1.0 1.0 2.0 2.0 0.0 0.0
2.0 0.0 0.0 1.0 1.0 0.0 0.0	2.0 0.0 1.0 2.0 2.0 0.0 0.0	2.0 1.0 0.0 1.0 0.0 0.2 0.0	2.0 1.0 1.0 2.0 2.0 1.0 0.0
2.0 0.0 0.0 1.0 1.0 1.0 0.0	2.0 0.0 1.0 2.0 2.0 1.0 0.0	2.0 1.0 0.0 1.0 1.0 0.0 0.0	2.0 1.0 1.0 2.0 2.0 2.0 0.0
2.0 0.0 0.0 1.0 1.0 2.0 0.0	2.0 0.0 1.0 2.0 2.0 2.0 0.0	2.0 1.0 0.0 1.0 1.0 1.0 0.0	2.0 1.0 2.0 0.0 0.0 0.0 0.0
2.0 0.0 0.0 1.0 2.0 0.0 0.0	2.0 0.0 2.0 0.0 0.0 0.0 1.0	2.0 1.0 0.0 1.0 1.0 2.0 0.0	2.0 1.0 2.0 0.0 0.0 0.1 0.0
2.0 0.0 0.0 1.0 2.0 1.0 0.0	2.0 0.0 2.0 0.0 0.0 0.1 0.0	2.0 1.0 0.0 1.0 2.0 0.0 0.0	2.0 1.0 2.0 0.0 0.0 0.2 0.0
2.0 0.0 0.0 1.0 2.0 2.0 0.0	2.0 0.0 2.0 0.0 0.0 2.0 1.0	2.0 1.0 0.0 1.0 2.0 1.0 0.0	2.0 1.0 2.0 0.0 1.0 0.0 0.0
2.0 0.0 0.0 2.0 0.0 0.0 0.0 1.0	2.0 0.0 2.0 0.0 1.0 0.0 0.0	2.0 1.0 0.0 1.0 2.0 2.0 0.0	2.0 1.0 2.0 0.0 1.0 1.0 0.0
2.0 0.0 0.0 2.0 0.0 1.0 0.0	2.0 0.0 2.0 0.0 1.0 1.0 0.0	2.0 1.0 0.0 2.0 0.0 0.0 0.0	2.0 1.0 2.0 0.0 1.0 2.0 0.0
2.0 0.0 0.0 2.0 0.0 2.0 1.0	2.0 0.0 2.0 0.0 1.0 2.0 0.0	2.0 1.0 0.0 2.0 0.0 1.0 0.0	2.0 1.0 2.0 0.0 2.0 0.0 0.0
2.0 0.0 0.0 2.0 1.0 0.0 0.0	2.0 0.0 2.0 0.0 2.0 0.0 1.0	2.0 1.0 0.0 2.0 0.0 2.0 0.0	2.0 1.0 2.0 0.0 2.0 1.0 0.0

2.0 0.0 0.0 2.0 1.0 1.0 0.0	2.0 0.0 2.0 0.0 2.0 1.0 0.0	2.0 1.0 0.0 2.0 1.0 0.0 0.0	2.0 1.0 2.0 0.0 2.0 2.0 0.0
2.0 0.0 0.0 2.0 1.0 2.0 0.0	2.0 0.0 2.0 0.0 2.0 2.0 1.0	2.0 1.0 0.0 2.0 1.0 1.0 0.0	2.0 1.0 2.0 1.0 0.0 0.0 0.0
2.0 0.0 0.0 2.0 2.0 0.0 1.0	2.0 0.0 2.0 1.0 0.0 0.0 0.0	2.0 1.0 0.0 2.0 1.0 2.0 0.0	2.0 1.0 2.0 1.0 0.0 1.0 0.0
2.0 0.0 0.0 2.0 2.0 1.0 0.0	2.0 0.0 2.0 1.0 0.0 1.0 0.0	2.0 1.0 0.0 2.0 2.0 0.0 0.0	2.0 1.0 2.0 1.0 0.0 2.0 0.0
2.0 0.0 0.0 2.0 2.0 2.0 1.0	2.0 0.0 2.0 1.0 0.0 2.0 0.0	2.0 1.0 0.0 2.0 2.0 1.0 0.0	2.0 1.0 2.0 1.0 1.0 0.0 0.0
2.0 0.0 1.0 0.0 0.0 0.0 0.0	2.0 0.0 2.0 1.0 1.0 0.0 0.0	2.0 1.0 0.0 2.0 2.0 2.0 0.0	2.0 1.0 2.0 1.0 1.0 1.0 0.0
2.0 0.0 1.0 0.0 0.0 0.1 0.0	2.0 0.0 2.0 1.0 1.0 1.0 0.0	2.0 1.0 1.0 0.0 0.0 0.0 0.0	2.0 1.0 2.0 1.0 1.0 2.0 0.0
2.0 0.0 1.0 0.0 0.0 2.0 0.0	2.0 0.0 2.0 1.0 1.0 2.0 0.0	2.0 1.0 1.0 0.0 0.0 0.1 0.0	2.0 1.0 2.0 1.0 2.0 0.0 0.0
2.0 0.0 1.0 0.0 1.0 0.0 0.0	2.0 0.0 2.0 1.0 2.0 0.0 0.0	2.0 1.0 1.0 0.0 0.0 2.0 0.0	2.0 1.0 2.0 1.0 2.0 1.0 0.0
2.0 0.0 1.0 0.0 1.0 1.0 0.0	2.0 0.0 2.0 1.0 2.0 1.0 0.0	2.0 1.0 1.0 0.0 1.0 0.0 0.0	2.0 1.0 2.0 1.0 2.0 2.0 0.0
2.0 0.0 1.0 0.0 1.0 2.0 0.0	2.0 0.0 2.0 1.0 2.0 2.0 0.0	2.0 1.0 1.0 0.0 1.0 1.0 0.0	2.0 1.0 2.0 2.0 0.0 0.0 0.0
2.0 0.0 1.0 0.0 2.0 0.0 0.0	2.0 0.0 2.0 2.0 0.0 0.0 1.0	2.0 1.0 1.0 0.0 1.0 2.0 0.0	2.0 1.0 2.0 2.0 0.0 1.0 0.0
2.0 0.0 1.0 0.0 2.0 1.0 0.0	2.0 0.0 2.0 2.0 0.0 1.0 0.0	2.0 1.0 1.0 0.0 2.0 0.0 0.0	2.0 1.0 2.0 2.0 0.0 2.0 0.0
2.0 0.0 1.0 0.0 2.0 2.0 0.0	2.0 0.0 2.0 2.0 0.0 2.0 1.0	2.0 1.0 1.0 0.0 2.0 1.0 0.0	2.0 1.0 2.0 2.0 1.0 0.0 0.0
2.0 0.0 1.0 1.0 0.0 0.0 0.0	2.0 0.0 2.0 2.0 1.0 0.0 0.0	2.0 1.0 1.0 0.0 2.0 2.0 0.0	2.0 1.0 2.0 2.0 1.0 1.0 0.0
2.0 0.0 1.0 1.0 0.0 0.1 0.0	2.0 0.0 2.0 2.0 1.0 1.0 0.0	2.0 1.0 1.0 1.0 0.0 0.0 0.0	2.0 1.0 2.0 2.0 1.0 2.0 0.0
2.0 0.0 1.0 1.0 0.0 2.0 0.0	2.0 0.0 2.0 2.0 1.0 2.0 0.0	2.0 1.0 1.0 1.0 0.0 0.1 0.0	2.0 1.0 2.0 2.0 2.0 0.0 0.0
2.0 2.0 0.0 0.0 0.0 0.0 1.0	2.0 0.0 2.0 2.0 2.0 0.0 1.0	2.0 1.0 1.0 1.0 0.0 0.2 0.0	2.0 1.0 2.0 2.0 2.0 1.0 0.0
2.0 2.0 0.0 0.0 0.0 1.0 0.0	2.0 0.0 2.0 2.0 2.0 1.0 0.0	2.0 1.0 1.0 1.0 1.0 0.0 0.0	2.0 1.0 2.0 2.0 2.0 2.0 0.0
2.0 2.0 0.0 0.0 0.0 2.0 1.0	2.0 2.0 1.0 2.0 0.0 2.0 0.0	2.0 2.0 1.0 0.0 0.0 2.0 0.0	2.0 2.0 2.0 1.0 2.0 1.0 0.0
2.0 2.0 0.0 0.0 1.0 0.0 0.0	2.0 2.0 1.0 2.0 1.0 0.0 0.0	2.0 2.0 1.0 0.0 1.0 0.0 0.0	2.0 2.0 2.0 1.0 2.0 2.0 0.0
2.0 2.0 0.0 0.0 1.0 1.0 0.0	2.0 2.0 1.0 2.0 1.0 1.0 0.0	2.0 2.0 1.0 0.0 1.0 1.0 0.0	2.0 2.0 2.0 2.0 0.0 0.0 1.0
2.0 2.0 0.0 0.0 1.0 2.0 0.0	2.0 2.0 1.0 2.0 1.0 2.0 0.0	2.0 2.0 1.0 0.0 1.0 2.0 0.0	2.0 2.0 2.0 2.0 0.0 1.0 0.0
2.0 2.0 0.0 0.0 2.0 0.0 1.0	2.0 2.0 1.0 2.0 2.0 0.0 0.0	2.0 2.0 1.0 0.0 2.0 0.0 0.0	2.0 2.0 2.0 2.0 0.0 2.0 1.0
2.0 2.0 0.0 0.0 2.0 1.0 0.0	2.0 2.0 1.0 2.0 2.0 1.0 0.0	2.0 2.0 1.0 0.0 2.0 1.0 0.0	2.0 2.0 2.0 2.0 1.0 0.0 0.0
2.0 2.0 0.0 0.0 2.0 2.0 1.0	2.0 2.0 1.0 2.0 2.0 2.0 0.0	2.0 2.0 1.0 0.0 2.0 2.0 0.0	2.0 2.0 2.0 2.0 1.0 1.0 0.0
2.0 2.0 0.0 1.0 0.0 0.0 0.0	2.0 2.0 2.0 0.0 0.0 0.0 1.0	2.0 2.0 1.0 1.0 0.0 0.0 0.0	2.0 2.0 2.0 2.0 1.0 2.0 0.0
2.0 2.0 0.0 1.0 0.0 0.1 0.0	2.0 2.0 2.0 0.0 0.0 1.0 0.0	2.0 2.0 1.0 1.0 0.0 1.0 0.0	2.0 2.0 2.0 2.0 2.0 0.0 1.0
2.0 2.0 0.0 1.0 0.0 2.0 0.0	2.0 2.0 2.0 0.0 0.0 2.0 1.0	2.0 2.0 1.0 1.0 0.0 2.0 0.0	2.0 2.0 2.0 2.0 2.0 1.0 0.0
2.0 2.0 0.0 1.0 1.0 0.0 0.0	2.0 2.0 2.0 0.0 1.0 0.0 0.0	2.0 2.0 1.0 1.0 1.0 0.0 0.0	2.0 2.0 2.0 2.0 2.0 2.0 1.0
2.0 2.0 0.0 1.0 1.0 1.0 0.0	2.0 2.0 2.0 0.0 1.0 1.0 0.0	2.0 2.0 1.0 1.0 1.0 1.0 0.0	2.0 2.0 0.0 2.0 0.0 0.0 1.0
2.0 2.0 0.0 1.0 1.0 2.0 0.0	2.0 2.0 2.0 0.0 1.0 2.0 0.0	2.0 2.0 1.0 1.0 1.0 2.0 0.0	2.0 2.0 0.0 2.0 0.0 1.0 0.0
2.0 2.0 0.0 1.0 2.0 0.0 0.0	2.0 2.0 2.0 0.0 2.0 0.0 1.0	2.0 2.0 1.0 1.0 2.0 0.0 0.0	2.0 2.0 0.0 2.0 0.0 2.0 1.0
2.0 2.0 0.0 1.0 2.0 1.0 0.0	2.0 2.0 2.0 0.0 2.0 1.0 0.0	2.0 2.0 1.0 1.0 2.0 1.0 0.0	2.0 2.0 0.0 2.0 1.0 0.0 0.0
2.0 2.0 0.0 1.0 2.0 2.0 0.0	2.0 2.0 2.0 0.0 2.0 2.0 1.0	2.0 2.0 1.0 1.0 2.0 2.0 0.0	2.0 2.0 0.0 2.0 1.0 1.0 0.0
2.0 2.0 2.0 1.0 1.0 1.0 0.0	2.0 2.0 2.0 1.0 0.0 0.0 0.0	2.0 2.0 1.0 2.0 0.0 0.0 0.0	2.0 2.0 0.0 2.0 1.0 2.0 0.0
2.0 2.0 2.0 1.0 1.0 2.0 0.0	2.0 2.0 2.0 1.0 0.0 1.0 0.0	2.0 2.0 1.0 2.0 0.0 1.0 0.0	2.0 2.0 0.0 2.0 2.0 0.0 1.0
2.0 2.0 2.0 1.0 2.0 0.0 0.0	2.0 2.0 2.0 1.0 0.0 2.0 0.0	2.0 2.0 1.0 0.0 0.0 0.0 0.0	2.0 2.0 0.0 2.0 2.0 1.0 0.0
2.0 2.0 0.0 2.0 2.0 2.0 1.0	2.0 2.0 2.0 1.0 1.0 0.0 0.0	2.0 2.0 1.0 0.0 0.0 1.0 0.0	

A.3 Overtaking neural network inputs and outputs

0.0 1.0 0.0 0.0 0.0 0.0 0.0	0.0 1.0 2.0 0.0 1.0 2.0 0.0	0.0 1.0 1.0 1.0 0.0 1.0 0.0	0.0 1.0 0.0 1.0 2.0 1.0 0.0
0.0 1.0 0.0 0.0 0.0 1.0 0.0	0.0 1.0 2.0 0.0 2.0 0.0 1.0	0.0 1.0 1.0 1.0 0.0 2.0 1.0	0.0 1.0 0.0 1.0 2.0 2.0 1.0
0.0 1.0 0.0 0.0 0.0 2.0 1.0	0.0 1.0 2.0 0.0 2.0 1.0 1.0	0.0 1.0 1.0 1.0 1.0 0.0 0.0	0.0 1.0 0.0 2.0 0.0 0.0 1.0
0.0 1.0 0.0 0.0 1.0 0.0 0.0	0.0 1.0 2.0 0.0 2.0 2.0 1.0	0.0 1.0 1.0 1.0 1.0 1.0 0.0	0.0 1.0 0.0 2.0 0.0 1.0 1.0
0.0 1.0 0.0 0.0 1.0 1.0 0.0	0.0 1.0 2.0 1.0 0.0 0.0 0.0	0.0 1.0 1.0 1.0 1.0 2.0 1.0	0.0 1.0 0.0 2.0 0.0 2.0 1.0
0.0 1.0 0.0 0.0 1.0 2.0 1.0	0.0 1.0 2.0 1.0 0.0 1.0 0.0	0.0 1.0 1.0 1.0 2.0 0.0 0.0	0.0 1.0 0.0 2.0 1.0 0.0 0.0
0.0 1.0 0.0 0.0 2.0 0.0 1.0	0.0 1.0 2.0 1.0 0.0 2.0 0.0	0.0 1.0 1.0 1.0 2.0 1.0 0.0	0.0 1.0 0.0 2.0 1.0 1.0 0.0
0.0 1.0 0.0 0.0 2.0 1.0 0.0	0.0 1.0 2.0 1.0 1.0 0.0 0.0	0.0 1.0 1.0 1.0 2.0 2.0 1.0	0.0 1.0 0.0 2.0 1.0 2.0 0.0
0.0 1.0 0.0 0.0 2.0 2.0 0.0	0.0 1.0 2.0 1.0 1.0 1.0 0.0	0.0 1.0 1.0 2.0 0.0 0.0 0.0	0.0 1.0 0.0 2.0 2.0 0.0 0.0
0.0 1.0 0.0 1.0 0.0 0.0 0.0	0.0 1.0 2.0 1.0 1.0 2.0 0.0	0.0 1.0 1.0 2.0 0.0 1.0 0.0	0.0 1.0 0.0 2.0 2.0 1.0 0.0
0.0 1.0 0.0 1.0 0.0 1.0 0.0	0.0 1.0 2.0 1.0 2.0 0.0 0.0	0.0 1.0 1.0 2.0 0.0 2.0 0.0	0.0 1.0 0.0 2.0 2.0 2.0 0.0
0.0 1.0 0.0 1.0 0.0 2.0 1.0	0.0 1.0 2.0 1.0 2.0 1.0 0.0	0.0 1.0 1.0 2.0 1.0 0.0 0.0	0.0 1.0 1.0 0.0 0.0 0.0 0.0
0.0 1.0 0.0 1.0 1.0 0.0 0.0	0.0 1.0 2.0 1.0 2.0 2.0 0.0	0.0 1.0 1.0 2.0 1.0 1.0 0.0	0.0 1.0 1.0 0.0 0.0 1.0 0.0
0.0 1.0 0.0 1.0 1.0 1.0 0.0	0.0 1.0 2.0 2.0 0.0 0.0 0.0	0.0 1.0 1.0 2.0 1.0 2.0 0.0	0.0 1.0 1.0 0.0 0.0 2.0 1.0
0.0 1.0 0.0 1.0 1.0 2.0 1.0	0.0 1.0 2.0 2.0 0.0 1.0 0.0	0.0 1.0 1.0 2.0 2.0 0.0 1.0	0.0 1.0 1.0 0.0 1.0 0.0 0.0
0.0 1.0 0.0 1.0 2.0 0.0 0.0	0.0 1.0 2.0 2.0 0.0 2.0 0.0	0.0 1.0 1.0 2.0 2.0 1.0 1.0	0.0 1.0 1.0 0.0 1.0 1.0 0.0
0.0 1.0 2.0 2.0 2.0 0.0 0.0	0.0 1.0 2.0 2.0 1.0 0.0 0.0	0.0 1.0 1.0 2.0 2.0 2.0 0.0	0.0 1.0 1.0 0.0 1.0 2.0 1.0
0.0 1.0 2.0 2.0 2.0 1.0 0.0	0.0 1.0 2.0 2.0 1.0 1.0 0.0	0.0 1.0 2.0 0.0 0.0 0.0 1.0	0.0 1.0 1.0 0.0 2.0 0.0 0.0
0.0 1.0 2.0 2.0 2.0 2.0 0.0	0.0 1.0 2.0 2.0 1.0 2.0 0.0	0.0 1.0 2.0 0.0 0.0 1.0 1.0	0.0 1.0 1.0 0.0 2.0 1.0 0.0
0.0 1.0 2.0 0.0 1.0 0.0 0.0	0.0 1.0 2.0 0.0 1.0 1.0 0.0	0.0 1.0 2.0 0.0 0.0 2.0 1.0	0.0 1.0 1.0 0.0 2.0 2.0 1.0
			0.0 1.0 1.0 1.0 0.0 0.0 0.0

BIBLOGRAPHY

[1] The Driver Monitor System: A Means of Assessing Driver Performance. Johns Hopkins, APL Technical Digest. Volume 25, Number 3 (July—September 2004), pp. 269-277. Authors: Kevin C. Baldwin, Donald D. Duncan, and Sheila K. West.

[2] Reliable method for driving events recognition. IEEE Transactions on Intelligent Transportation Systems. Volume. 6, Number 2, June 2005, pp. 198-205. Author: Dejan Mitrovic.

[3] Detection of Hazardous Driving Behavior Using Fuzzy Logic. Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2008. ECTI-CON 2008. 5th International Conference. Volume. 2, May 2008, pp. 657-660. Authors: T. Imkamon, P. Saensom, P. Tangamchit, P.Pongpaibool.

[4] Multi-sensor data fusion for land vehicle attitude estimation using a fuzzy expert system. Data Science Journal. Volume 4, 2005. pp. 127-139. Authors: Jau-Hsiung Wang and Yang Gao.

[5] Design of Hack Box for Moving Vehicle Warning System. Student Conference on Research and Development (SCOReD) 2003 Proceedings, Putrajaya, Malaysia. Author: Ngo Chon Chet, Department of Electrical and Electronic Engineering Faculty of Engineering, University Putra Malaysia 43400 UPM, Serdang, Selangor, Malaysia.

[6] System on Chip design of Embedded Controller for Car Black Box. Proceedings of the 2007 IEEE Intelligent Vehicles Symposium Istanbul, Turkey, June 2007. Pp. 13-15. Authors: Dae Geun Lee, Se Myoung Jung , Myoung Seob Lim, Dept. of Electronics & Information Engineering, Chonbuk University, Jeonju, Korea.

[7] http://car-accidents.com/ last accessed on 01/03/2010.

[8]http://www.who.int/violence_injury_prevention/publications/road_traffic/world_report/en/index.html last accessed on 04/03/2010.

[9] http://www.veronica-project.net/ last accessed on 04/03/2010.

[10] http://www.android.com/ last accessed 08/03/2010.

[11] http://www.htc.com/www/product/desire/overview.html last accessed on 03/10/2010

[12] Analysis of the Overtaking Behaviour of Motor Vehicle Drivers, Research project, Institute for Transport Sciences, Budapest (1999). Author: Mocsari Tibor.

[13] http://www.statsoft.com/textbook/neural-networks/ last accessed on 01/10/2010

[14] Mechanism for an Intelligent Neural network based Driving system, *E-Tech 2004*, vol., no., pp. 53- 60, 31 July 2004. Author: Srinivasan, T, Jonathan, J.B.S., Chandrasekhar, A.

[15] Matching GPS Observations to Locations on a Digital Map, in proceedings of the 81st Annual Meeting of the Transportation Research Board, Washington D.C, 2002. Author: J. S. Greenfield.

[16] A Weight-based Map Matching Method in Moving Objects Databases1, in proceedings of the International Conference on Scientific and Statistical Database Management (SSDBM'04), vol. 16, pp. 437-410, 2004. Author: H. B. Yin and O. Wolfson.

[17] Fusion of Map and Sensor Data in a Morder Car Navigation System, Journal of VSLI Signal Processing 45, pp. 112-122, 2006. Authors: D. Obradovic, H. Lenz and M. Schupfner.

[18] A High Accuracy Fuzzy Logic-based Map-Matching Algorithm for Road Transport, Journal of Intelligent Transportation Systems: Technology, Planning, and Operations, vol. 10(3), pp. 103-115, 2006. Authors: M. A. Quddus, W. Y. Ochieng and R. B. Noland.

[19] O. Pink and B. Hummel, "A Statistical Approach to Map Matching Using Road Network Geometry, Topology and Vehicular Motion Constraints", in proceedings of the 11th International IEEE Conference on Intelligent Transprotation Systems, 2008.

[20] P. Newson and John Krumm "Hidden Markov Map Matching Through Noise and Sparseness", in Proceedings of ACM SIGSPATIAL Conference on Geographical Information Systems (ACM GIS 2009).

[21] S. Brakatsouls, D. Pfoser, R. Salas and C. Wenk, "On Map-Matching Vehicle Tracking Data", In 31st International Conference on Very Large Data Bases (Trondheim, Norway,2005). VLDB ‟05.

[22] C. Wenk, R. Salas and D. Pfoser, "Addressing the Need for Map-Matching Speed: Localizing Globalb Curve-Matching Algorithms", In SSDBM‟ 06: Proceedings of the 18th international Conference on Scientific and Statistical Database Management, 2006

[23] S.S. Chawathe, "Segment-based Map Matching", In IEEE Symposium on Intelligent Vehicles, 2007.

[24] H. Alt, A. Efrat, G. Rote, and C. Wenk, "Matching Planar Maps", J. of Algorithms, vol. 49: pp.262–283, 2003.

[25] H. B. Yin and O. Wolfson, "A Weight-based Map Matching Method in Moving Objects Databases1", In proceedings of the International Conference on Scientific and Statistical Database Management (SSDBM'04), vol. 16, pp. 437-410, 2004.

[26] Sook Yoon; Hyouck Min Yoo; Sang Hoon Yang; Dong Sun Park; , "Computation of discrete Fréchet distance using CNN," *Cellular Nanoscale Networks and Their Applications (CNNA), 2010 12th International Workshop on* , vol., no., pp.1-6, 3-5 Feb. 2010

[27] http://www.geotools.org/ last accessed on 01/10/2010

[28] http://mina.apache.org/ last accessed on 03/10/2010

[29] http://static.springsource.org/spring/docs/2.5.x/reference/index.html Last accessed on 03/10/2010

[30] http://neuroph.sourceforge.net/ last accessed on 03/10/2010

[31] http://www.c2.com/cgi/wiki?PlainOldJavaObject last accessed on 03/10/2010

[32]http://fbim.fh-regensburg.de/~saj39122/jfroehl/diplom/e-11-text.html#TheHumanBrain Last accessed on 04/10/2010