2012

Driver Modeling for Risk Assessment

André Joseph Edouard Levesque
University of Windsor

DRIVER MODELING FOR RISK ASSESSMENT

by

ANDRÉ J. E. LEVESQUE

A Thesis
Submitted to the Faculty of Graduate Studies
through Mechanical, Automotive, & Materials Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science at the
University of Windsor

Windsor, Ontario, Canada
2012

DRIVER MODELING FOR RISK ASSESSMENT

by

ANDRÉ J. E. LEVESQUE

APPROVED BY:

Dr. C. Lee
Civil and Environmental Engineering

Dr. B. Minaker
Mechanical, Automotive, & Materials Engineering

Dr. J. Johrendt, Advisor
Mechanical, Automotive, & Materials Engineering

Dr. R. Barron, Chair of Defense
Mechanical, Automotive, & Materials Engineering

January 23, 2012

Declaration of Previous Publication

Chapter 2: "Literature Review", consists of "The State of the Art of Driver Model Development", as published by the Society of Automotive Engineers, 4/12/2011:

A. Levesque, J. Johrendt,(2011) The State of the Art of Driver Model Development, SAE Paper 2011-01-0432. SAE Publishers.

I certify that I have obtained a written permission from the copyright owner to include the above published material in my thesis. This permission by be referred to in Appendix A of this document. I certify that the above material describes work completed during my registration as graduate student at the University of Windsor.

I declare that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

As the Baby Boomer generation begins to age along with the advances made in modern medicine, the number of elderly people is expected to increase significantly over the course of the next several decades. As the elderly population increases, the number of elderly drivers on our roads increases as well. Driving is both a physically and cognitively intensive task, and it is a well known fact that as people age, both their physical, and cognitive abilities decrease. As a result, elderly drivers are at an increased risk of being involved in a collision. Currently, the methods to determine driver fitness are limited, and as a result, doctors are placed in a difficult situation where they must choose between protecting their client, the public, and their own reputation; or allowing their client to maintain their accustomed level of independence. While there are elderly drivers who are obviously no longer fit to drive, the problem is making a decision regarding elderly drivers whose ability has not completely deteriorated, and fit in a sort of "gray area". The following research presents the ground work for the development of an objective driver risk assessment tool. The assessment tool makes use of artificial neural networks to both model, and evaluate driver behaviour. Presented herein is the current state of driver modeling, the theory behind neural networking and vehicle dynamics, the process used to develop the model, the performance results, and finally the conclusions that were obtained from the research.

Dedication

To my parents, Marilyn and Philippe
For their understanding of the value of eduction and more importantly for all of their love & support

Acknowledgements

I would first like to thank my supervisor Dr. Jennifer Johrendt for providing with the great opportunity to study under her guidance. No matter what, it seemed as if she always had time to listen to me along with an answer to all of my questions. Whenever any difficulties arose, a short conversation with her always had me on my way again. I would also like to thank Dr. Johrendt for her encouragement and her faith in me.

I would like to thank the AUTO21 Network Centres of Excellence for funding the research that has provided me with many great opportunities, and has allowed me to extend my knowledge in a field of which I am very passionate. I must also acknowledge the GRAME resarch group at L'Université Laval in Québec City, PQ for providing me with the data for my research and offer a special thanks to Martin Lavallière for answering so many of my questions.

I must also mention my colleagues in 110C, especially Hart Honickman and Sean Maloney who, over the past year and a half, have shared in many laughs and interesting discussions about projects that have failed to materialize, provided me with advice, put up with my antics and most importantly become very good friends. I also must acknowledge Dr. Bruce Minaker and Dr. Robert Rieveley for providing me with their opinions and technical advice.

Last and most certainly not least I must give a huge thanks to all my family and friends for all of their love and support over the years and for putting up with my occasional jargon filled ramblings. It is very reassuring to know that I will always have such great people in my life to support and encourage me in my endeavours.

Contents

List of Tables

List of Figures

Notation

Label	Description
A_x	Longitudinal Acceleration
A_y	Lateral Acceleration
ANN	Artificial Neural Network
B_p	Brake Pedal Input
BFGS	Broyden-Fletcher-Goldfarb-Shanno
DUI	Driving Under the Influence
GRAME	Groupe de Recherche en Analyse du Mouvement et Ergonomie (Translated: Movement and Ergonomics Analysis Research Groupe)
L-M	Levenberg-Marquardt
LVQ	Learning Vector Quantization
MLP	Multilayer Perceptron
NARX	Nonlinear Autoregressive Network with Exogenous Inputs
NHTSA	National Highway an Traffic Safety Administration
OBD	Optimal Brain Damage
RBFN	Radial Basis Function Network
ROC	Receiver Operating Characteristics
SOM	Self Organizing Map
STI	Systems Technology Inc.
T_p	Throttle Input
TBI	Traumatic Brain Injury
V_x	Longitudinal Velocity
V_y	Lateral Velocity
V_{lim}	Speed Limit
VDANL	Vehicle Dynamics Analysis NonLinear
VDANL/RT	Vehicle Dynamics Analysis NonLinear Real Time
Y	Lateral Position
ρ	Road Curvature
θ	Steering Wheel Angle

Chapter 1

Introduction

Over the course of the next several decades, the portion of the population greater than the age of 75 is set to increase significantly. A result of the increase of the number of elderly people will be an increase in the number of elderly drivers on the road. This poses a concern to society because as people age both their physical and mental abilities deteriorate. The task of driving is one that is both cognitively and physically demanding, and deterioration of such abilities poses a risk to both the elderly drivers themselves as well as the rest of the population using the roadways. The ability to identify competent behaviour is becoming increasingly important in order to ensure that elderly drivers are capable of driving and that no warning signs of potential danger are missed.

1.1 Current Practice

In order to evaluate elderly drivers there are very few options available. Current practice, for the most part, is based on observations by a physician through a physical examination. The physical examination is performed to assess the overall health of the person along with their mental and physical functions. Using conclusions drawn from the physical examination, the doctor then makes a decision regarding the fitness of the driver. Often, what may happen is the doctor will take a conservative approach and revoke the licence for fear of repercussions should the person be involved in collision, serious or otherwise. The problem with the current practice is that it is very subjective, and drivers that are still capable of driving may have their licence revoked. This leaves the person unable to transport themselves and limits their independence as they must now rely on either family members or friends to make even the shortest of trips, which ultimately reduces the quality of life of the individual. The entire process essentially places doctors in a lose-lose situation, because on one hand they risk being found liable if the person's actions harms either themselves or someone else;

on the other hand the person may be resentful towards the doctor since they are removing much of their independence.

One tool called DriveABLE™ is currently available on the market to assist with the decision. DriveABLE™ consists of two separate tests; the first test is done in office where the person wil undergo a physical examination by an occupational therapist to evaluate head mobility, peripheral vision as well as both mobility and strength of the wrists, arms and legs. The person will then be asked to perform a series of tasks with the assistance of a computer interface, the tasks are designed to test cognitive ability, memory and reflexes. The second test is done in-vehicle, where the person will be asked to drive a predetermined course while being accompanied by the occupational therapist for evaluation. All of the driver inputs are recorded as well and compared statistically with what is considered to be acceptable. While DriveABLE™ does provide an objective means for the evaluation of elderly drivers and the people using it do speak favourably about its performance, DriveABLE™ still has some drawbacks, the first of which is cost. Each session costs approximately $400; rather than being a one-time cost to purchase the equipment, a fee must be paid each time a person is tested. Another significant drawback of DriveABLE™ is with the in office test, all of the tasks are done on a computer with the use of either a mouse or a touchscreen. While these tests are good predictors of physical capabilities such as mobility, reflexes and vision, they fail to assess the person's physical capabilities when put in an actual driving situation. The first test may also be considered questionable by some because of the fact that it is a computer-based test. The reason for this is that elderly people are often unfamiliar with and therefore uncomfortable using computers. This instantly puts them at a disadvantage because they must spend time learning the interface rather than simply being evaluated on their performance, which might ultimately make them perform poorly. In the case of the on-road test, the drawback is that a potentially unsafe driver may be allowed to drive, thus putting themselves, the occupational therapist and other road users at risk. The second portion of the test is administered regardless of the outcome of the first part.

1.2 Proposed Solution

The idea being presented herein is to provide a conclusive, objective method of evaluating drivers, with particular attention paid to elderly drivers. The current methods of evaluation mentioned earlier are either very subjective and place doctors in what amounts to being an unfair position, or they are regarded as being controversial because they may not provide an accurate assessment. The method being proposed is to take advantage of the ability of Artificial Neural Networks to produce a driver model that represents a typical, average driver. Neural Networks are mathematical models that in concept mimic the function of the human brain in that they can learn, when a series of inputs is

presented to them, to predict an output. They are capable of both function approximation and data organization. In the case of this research, there are two neural network structures that will be used; the first will be of the function approximating variety, which will be used to model a typical driver. It will be developed by presenting data from a series of test subjects gathered by the GRAME research group at l'Université Laval in Québec City, PQ; using their driving simulator. This data will be used to represent what an average driver would likely do in a given scenario. The other type of network that will be used is a clustering network, where the actual behaviour of the driver will be compared with the behaviour of the driver model, or, what the behaviour should be.

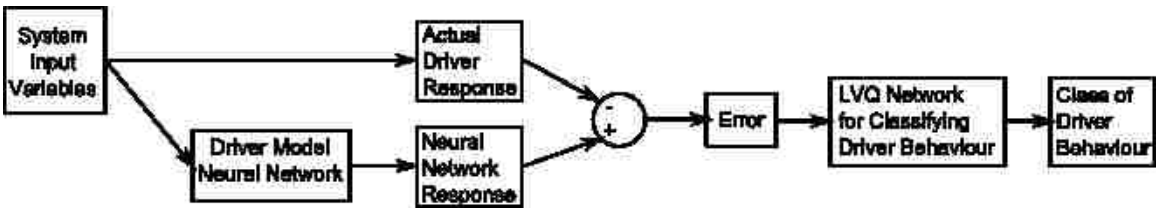


Figure 1.1: Block Diagram of Driver Assessment Tool

Ultimately a decision will be made where there are three possible outcomes; the first is that the driver's behaviour is satisfactory and in-line with what is to be expected from a typical driver. The next potential outcome is that the driver's behaviour indicates that their ability has deteriorated, but not to the point where they can no longer drive; therefore they will be allowed to keep their licence but they may only drive under limited conditions (e.g., only during the day, good weather, not on freeways, etc.). The final outcome is that the drivers ability has deteriorated considerably, and therefore they should no longer be allowed to drive. It is hoped that this tool will help keep drivers that are in the second category of limited ability on the road such that they can still retain some of their independence and be allowed to make short trips by car. At the same time, both they and the other road users will be protected by keeping them away from situations that they are no longer capable of handling due to their decreased capability. Another potential use for this model is to assist with driver retraining; in the past, the group at Laval have used their simulator to help retrain drivers, and it may be possible to use this tool to identify where the shortcomings of the driver are and thus tailor a retraining plan properly suited to their issues. In the past, attempts to retrain drivers using driving simulators have proven to be successful [1]. According to the group at Laval, drivers who were informed of their shortcomings and instructed on how to improve their driving, demonstrated an improved level of performance when they were evaluated afterwards. Of note is that several of these drivers were elderly, which offers some promise that an assessment tool could be valuable in order to help retrain elderly drivers whose abilities have diminished.

Chapter 2

Literature Review

The follwing chapter has been taken from SAE paper 2011-01-0432 entitled "The State of the Art of Driver Model Development"; the abstract, introduction and conclusion have all been omitted.

2.1 Distinction Between Model Types

According to Michon, driver models can be classified based on two dimensions: distinctions between taxonomic and functional models as well as between behavioural (input-output) and internal state (psychological) models[2, 3, 4] (Figure 2.1). Taxonomic models are those that are based purely on facts and ignore any type of interaction between different components of the model. In the case of functional models, these interactions are considered. By taking into consideration these interactions it is possible to determine how an action in one component of the model will affect what happens in all other components or how a given component is affected by what is occurring in the system as a whole. Behavioural models analyse the input to the model and the associated output. They are incapable of analysing the thought processes of the driver. Internal state models are models that are capable of analysing the thought process and they determine the psychological state of the driver.

The result of these four classifications of models and the fact that they are arranged in a two-dimensional manner yields several types of models that combine the aspects of two of the classifications. The first type of model is task analysis models which are classified as a taxonomic and behavioural model. Task analysis models separate driving into a series of tasks and subtasks, such models are built using data that was likely collected from either a driving simulator or from an instrumented vehicle. The purpose of a task analysis model is to analyse the necessary requirements of the driver for the given situation as well as the ability of the driver. The next type of model is the trait model which is a combination of taxonomic and internal state models. Trait models describe

	Taxonomic	Functional
Input-Output (Behavioural)	Task Analyses	Mechanistic Models Adaptive Control Models -Servo-Control -Information Flow Control
Internal State (Psycholgical)	Trait Models	Motivational Models Cognitive (Process) Models

Figure 2.1: Model Types According to Michon, [2]

the thought process of the driver and aim to classify the attitude and traits of the driver with the goal of determining the relative amount of risk that a given driver is likely to take. Mechanistic models are classified as behavioural and functional models and attempt to use a mechanistic system to describe driver behaviour (one example referred to by Michon is the use of hydrodynamics to model traffic behaviour). Overall, mechanistic models are said to be somewhat limited in their capability and see very little use as no consideration is given to the driver's thought process. Adaptive control models are also classified under behavioural and functional models, a further subdivision of types exists consisting of servo-controlled models and information flow control models. Servo-controlled models consider driving as a continuous or intermittent tracking task, Often these models use input signals that represent the lateral position (compensatory tracking) or road curvature (pursuit tracking). In order to represent the driver and vehicle, transfer functions are used. Information flow control models are very much like the name suggests in that they use the information that is perceived by the driver. The information is then interpreted through a flow of logical steps. Motivational models are classified as internal state and functional type models; according to Michon at the time of writing such models were limited to the discussion of the products of cognitive functions such as beliefs, emotions and intentions rather than the actual cognitive functions. When discussing motivational models, there are three different varieties that exist, which are compensation models, risk threshold theory and threat avoidance models. Compensation models postulate that drivers establish a target level of risk and attempt to maintain that target level of risk and will act to compensate for any deviation. Risk threshold models appear to have some similar characteristics to compensation models; however, they are based more on the driver attempting to maintain a balance between the perceived level of risk and the actual level of risk. Finally, threat avoidance models state that drivers attempt to avoid any and all risk. Further information with regard to motivational

models as well as some examples will be presented in the following sections.

As mentioned previously in the introduction, one method of classifying models is to distinguish between descriptive models and motivational models. These classifications are proposed by Carsten[5] and are broader than those proposed by Michon. According to Carsten, descriptive models attempt to describe either the entire driving task or some parts of it; they are analytical in nature and therefore cannot make any predictions with regard to the effect of motivation, capability or decision. Classified under descriptive models are task models, adaptive control models and production models. Details with regard to the first two types have already been previously discussed. Production models describe driving as a formal set of rules in the manner that a production system works. The definition of motivational models according to Carsten is very much the same as the one presented by Michon in that they attempt to describe the reasoning behind driver decisions based on aspects such as risk, personality, capability and so forth.

2.2 Model Hierarchy

To describe the driving task, a hierarchical structure has been put forth by Michon[2, 6]. The model described by Michon, models driving task performance, which describes the level at which a driver processes a given task. The model consists of three levels: the strategic level, the manoeuvring level and the control level (Figure 2.2). A fourth level, the behavioural level is sometimes added to the model. The behavioural level contains the most fundamental aspects of the driver, namely the attitudes and beliefs that dictate the general behaviour of the driver in everyday life. The strategic level of the hierarchical model governs general plans for a given trip, aspects such as destination, planned stops and essentially any other long term goals for the trip. The manoeuvring level, sometimes known as the tactical level, manages more immediate goals; the operations at this level are of a shorter term than those at the strategic level. Essentially the manoeuvring level governs the tactics used to arrive at given destination such as when to make a turn, or the process of overtaking a vehicle in order to arrive at the final destination quicker. Finally the last level of this model is the control level; the control level governs very short term goals to help the driver arrive at the desired destination. Goals of interest at the control level consist of basic inputs to ensure that the vehicle stays on the route, things like speed control, negotiating a curve and maintaining lane position.

While the model hierarchy proposed by Michon describes the level on which the driver carries out a task, it fails to describe the type of behaviour used to carry out the driving task. The Task Behaviour model developed by Rasmussen [4, 6] describes the driving behaviour through the use of a sort of hierarchy. Rasmussen's model describes three different types of behaviour that are used by drivers while carrying out the driving task. These three types of behaviour are knowledge-based

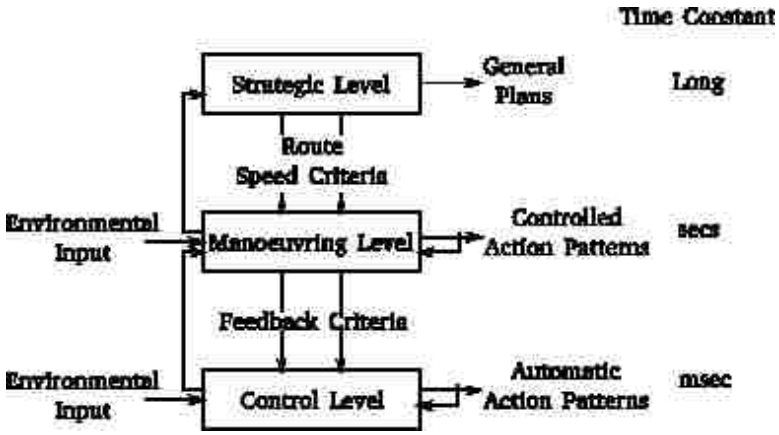


Figure 2.2: Michon's Hierarchical Structure, [2]

behaviour, rule-based behaviour and finally skill-based behaviour. Knowledge-based behaviour is used when a driver may encounter a new situation and must use knowledge from a previous situation or some form of training in order to deal with the situation at hand. Knowledge-based behaviour requires conscious thought on the part of the driver as the driver must actively search through their memory and find the appropriate knowledge to deal with the given situation. Rule-based behaviour is used when the driver has some familiarity with an event and there is a partially pre-determined method for dealing with a situation that the driver has already learned. In situations where rule-based behaviour is used, a combination of conscious and unconscious thought is used by the driver as a portion of the action is carried out automatically however the driver must still give some thought for dealing with the event. The final level of the task behaviour model involves skill-based behaviour where the task is carried out through an entirely unconscious process. Skill-based behaviour is used for generally simple tasks where the driver does not need to give any thought about the process and everything is accomplished automatically.

The final form of hierarchy involves the information processing model. While the information processing model is not necessarily a hierarchy, it can still be used in combination with the previously described hierarchies. The information processing model also consists of three different components which are perception, processing and action[4]. Perception is the point at which a driver first notices an event occurring within the driving environment; once the driver is aware of the situation, the information from the environment is processed by the driver where a decision is made with regards to the appropriate course of action. Once the appropriate course action is determined by the driver, the action is carried out to allow the vehicle to progress through the road environment with the least amount of difficulty possible. The information processing model can be viewed as

a cycle (Figure 2.3) as the driver is constantly going through this process and the actions taken by the driver have an impact on the situation at hand. The impact of the driver's actions must then be perceived by the driver in order for another action to be carried out for further progress through the road environment. To form a complete hierarchical model of driver behaviour, the three aforementioned models can be combined to form a 3-dimensional model that completely describes all aspects the driving task (Figure 2.4)[7, 8].

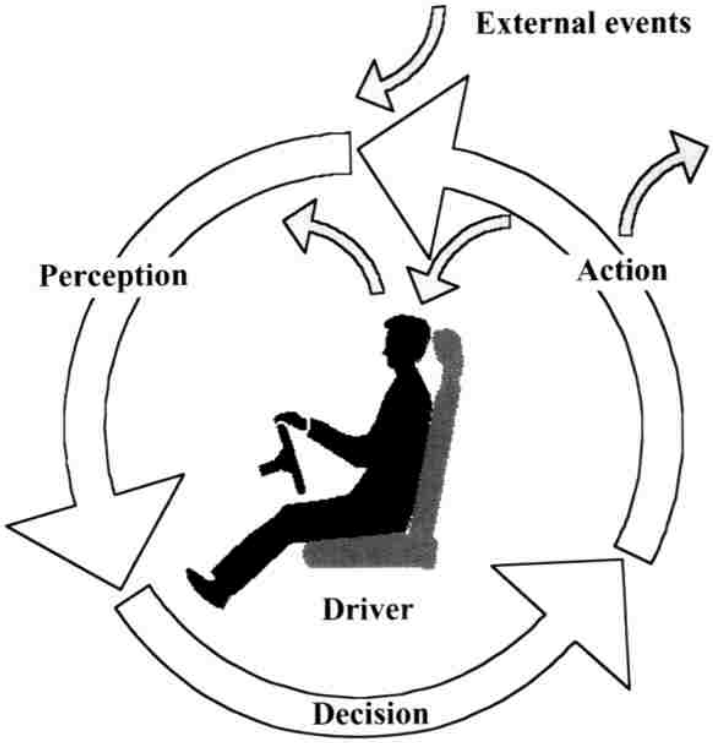


Figure 2.3: The driving process represented as a cycle, [4]

2.3 Motivation and Risk Management

Motivation is an important concept when trying to model driver behaviour, as it is a cognitive process and is determined by the driver's intentions. When modeling driver motivation there are three factors that are influential: attitude, the subjective norm and the perception of control [9]. The attitude of a driver is dictated by overall beliefs and a link between consequences and outcomes. Some drivers may believe that driving in a certain manner may yield positive consequences or

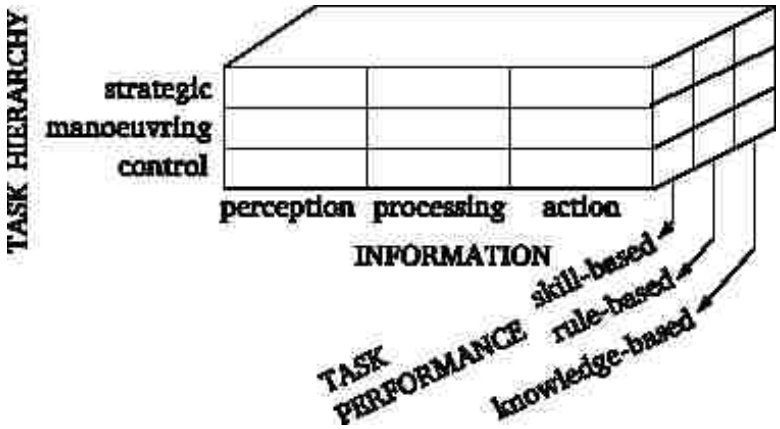


Figure 2.4: Three dimensional driver model, [7]

outcomes, while other drivers may believe that behaving in such a manner carries too much risk and that the potential positive outcomes do not outweigh the negative ones. The subjective norm is what the general population believes to be acceptable behaviour; if the majority of people behave in a certain manner, a driver may use that information to gauge how they should behave and whether or not the way in which they are behaving would be perceived favourably or unfavourably by the general population. Perception of control describes the driver's sense of ability to accomplish a task in the driving environment, which is governed by what the driver feels are their own limits as well as the limits of the vehicle.

Risk Management is another important concept when describing driver motivation and behaviour; several researchers have used risk management to establish driver models. Two notable theories that are discussed by Fuller[9] are the Risk Homeostasis Theory by Wilde and the Zero Risk Theory by Näätänen and Summala. The Risk Homeostasis Theory proposes that the intention of the driver is to maintain a constant level of risk called a target risk while driving. The target risk is influenced by a compromise between benefits and consequences. Drivers attempt to maximize the net benefit and in order to do so the amount of expected gain by the driver must be increased and thus the risk must be increased as well, subsequently increasing the amount of expected loss[10]. In order to maintain the target risk, the driver will compare it with the level of perceived risk. The perceived risk is determined by three factors according to Wilde, which are past experiences with the given situation, the perceived likelihood of an accident and the amount of confidence the driver has in their ability to make decisions and control the vehicle. An important note with regard to risk homeostasis is that Wilde stresses that the target level of risk is not constant and that drivers do in fact adjust according to the situation; however, in a given situation there will be no variation. In the

case of the Zero Risk Theory, it is proposed that the ultimate goal of the driver is to avoid any and all risk; this is made possible as drivers learn the required actions in order to avoid collisions. What is interesting with regard to the two aforementioned theories is that they are essentially a contradiction of each other, since, according to Wilde, at a state of zero risk nothing is gained or lost and thus no progress is made. In his criticisms of both methods, Vaa established an interesting concept of target feeling [11]. In his opinion, the Risk Homeostasis Theory is too rigid in its assumption of target risk being a number and that the Zero Risk theory fails to establish why drivers will feel safe traveling at a given speed, for example. According to Vaa, the target feeling is achieved by the driver in a given situation where the driver feels comfortable and without risk. In a manner of speaking, he has merged the two theories by taking the aspect that drivers attempt to achieve a target value (in this case a feeling of safety) and as a result,in the driver's opinion, there is no risk.

The Task-Capability model was proposed by Fuller [9]. The model works based on the fact that drivers have a limited amount of capability and that tasks have a demand associated with them that requires a certain amount of capability. For the majority of driving situations, the driver capability outweighs the task demand; for such situations the driver is able to maintain control of the vehicle. When the task demand exceeds the driver capability, the driver loses control of the vehicle (Figure 2.5). When loss of control occurs, the likelihood of a collision is increased significantly; however, it is possible to avert a collision either through the actions of other drivers or through a so-called "lucky escape". When describing driver capability, there are several aspects for consideration. There are physical abilities, such as speed, that affect reaction time, coordination and strength, and mental capabilities, which are determined by the driver's experiences as well as any training they may have received. To describe the task demand, there are four different categories that affect demand. The first category is the road environment, which encompasses aspects like visibility, road signs and road design (straight, curved etc.); the second category is related to traffic and the other vehicles in the driving environment; the third category is related to characteristics of the vehicle being driven; and the fourth category is the speed and direction of the vehicle. The fourth category is the only one over which the driver has any level of control.

2.4 Behavioural Adaptation

Behavioural Adaptation is a phenomenon associated with driver assistance systems. The phenomenon occurs when drivers operate vehicles equipped with these systems and become over-reliant on them. The driver assistance system is programmed to assist the driver with the driving task in order to make it easier and increase safety; however, because the driver is aware of the presence of this system, they may become inclined to take more risks. A classic example where behavioural

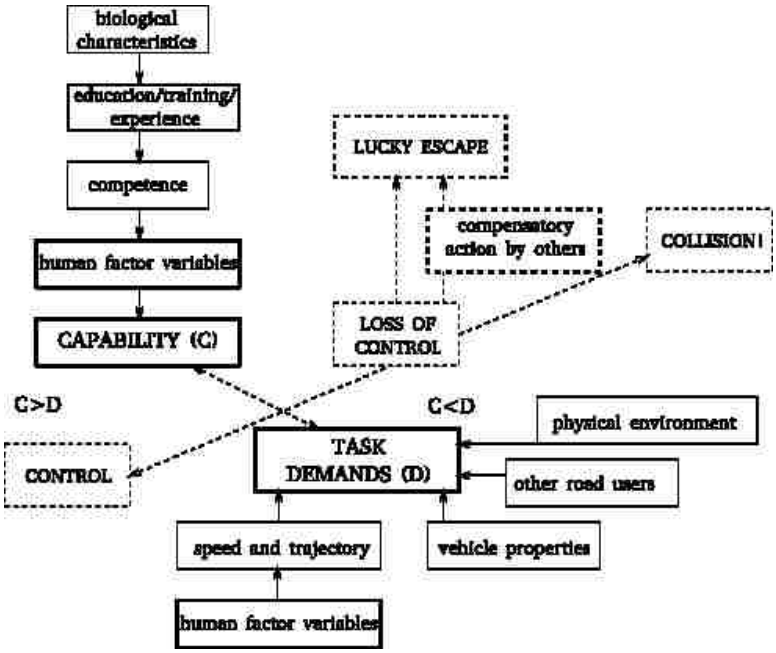


Figure 2.5: Task-Capability Model, [9]

adaptation occurs is with vehicles that use an anti-lock braking system (ABS). Some studies have observed that drivers operating vehicles with ABS will have a tendency to both drive at a higher speed and begin braking at a later point, as ABS reduces stopping distance and allows the vehicle to be steered while the brakes are being applied. Vaa discusses the details of such a study that was undertaken in Germany with taxi drivers[10]. Behavioural Adaptation is highly undesirable because it undermines the effectiveness of driver assistance systems as drivers adjust their behaviour to compensate for the supposed increase in safety offered by the system. An interesting fact is that the notion of behavioural adaptation ties in with the Risk Homeostasis theory, in that it states that drivers attempt to maintain a constant level of risk. It is essentially what occurs with behavioural adaptation since a driver assistance system works to lower the risk level; however, when drivers compensate for their effects, essentially they are increasing risk back to a level that they find to be acceptable. The works by Bengler[12], Janssen[13] and Saad[14] provide significant data to support this theory, as well as discussion on more specific effects due to behavioural adaptation.

2.5 Safety Margins

Safety margins have significant implications for driver behaviour and determining risk, as drivers create a series of imaginary regions around themselves within the driving environment. Drivers establish a safety region around themselves that consists of a risk threshold[15]. Once this threshold is breached the driver has a greater feeling of risk and that a collision may occur. The establishment of these safety margins is generally based on time and space. It has been suggested by Gibson and Crooks that drivers attempt to locate gaps in time and space within the road environment to which they refer to as the *field of safe travel*[16]. These gaps are pathways that allow the driver to progress through the road environment without being impeded (Figure 2.6).

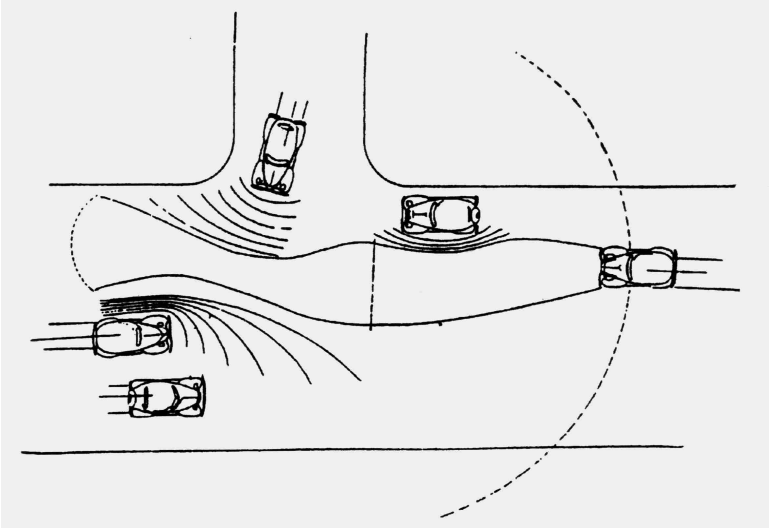


Figure 2.6: The field of safe travel as described by Gibson and Crooks, [16]

The field of safe travel is bounded by objects and features in the road environment and it resembles a sort of "tongue" that extends in front of the vehicle. According to Gibson and Crooks, acceleration is motivated by the desire arrive at a given destination while deceleration is motivated by a contraction in the field of safe travel. Another quantity discussed by Gibson and Crooks is the minimum braking distance, which is determined by a number of factors such as vehicle speed, driver reaction time, road conditions and the braking power of the vehicle. Typically, the minimum braking distance is ahead of the field of safe travel such that the driver feels safe and comfortable as they are capable of stopping before anything can impede them. It should be noted that the minimum braking distance can be extended by increasing the speed of the vehicle; however, it can only be extended as far the field of safe travel. Another factor affecting minimum braking distance is that

there is potential for the field of safe travel to be cut off abruptly to the point of cutting into the minimum braking distance and inducing the instantaneous application of maximum braking. In such a scenario a collision is highly probable.

The field of safe travel is defined by the number of different types of boundaries. The first type of boundary is the natural boundary which consists of obstacles along with environmental factors such as lighting (day or night, glare), weather and terrain. Vehicle capability also limits the field of safe travel; while a portion on the inside curve of a road may be unimpeded, it may still be impossible for the vehicle to travel along that path as the tires may not have enough grip to allow such a tight trajectory. While obstacles themselves serve to define the field of safe travel, drivers will typically not follow a path where they come in contact with the obstacle; therefore, the region around the object, known as the *clearance region* also defines the field of safe travel. Moving obstacles have clearance where the vehicle will be closest to; therefore, it may appear that an object is within the field of safe travel as it might be in front of the vehicle in its path. However, once the vehicle arrives at the point, the object will have moved. Such is the case for a pedestrian crossing the street or even a vehicle following another at higher speeds. Potential obstacles have an uncertainty attached to them; such situations occur when the driver may not be able to see part of the road environment ahead of the vehicle, as with blind corners, or when coming over a hill, the assumption is that the path is clear; however, the potential exists for there to be an obstacle. Finally, there are legal obstacles, which consist of things like speed limits or the information from signs such as a stop sign.

2.6 Error Quantification

In order to model driver performance and to determine whether or not a driver's behaviour can be considered appropriate, any deviation from desired behaviour can be viewed as an error. Driving errors have some relation to safety margins. Any behaviour that takes the driver outside of the established safety margin is likely an error of some nature, since it puts the driver at an increased risk of being involved in a collision. Driving errors can occur at all three levels of driver behaviour and are classified as either being slips/lapses or mistakes[17]. At the knowledge-based performance level of driver behaviour, errors are considered to be mistakes; these errors occur due to incorrect or limited knowledge of the driving situation, which results in the wrong course of action taken by the driver. At the rule-based level, errors are also classified as mistakes; generally these errors are the result of misapplying a certain rule to the given situation. Finally, at the skill-based level, errors are regarded as slips or lapses. Such an error is due to an action carried out by the driver unconsciously in the same manner that all activity at the skill level is carried out–without cognitive thought.

In order to quantify any errors, certain measures are required. One method of doing so is by

the use of time related measures. Several methods for time related measures are discussed by van der Horst[8]; he differentiates between methods that can be used for either lateral or longitudinal control. For lateral control of the vehicle, the most heavily discussed measure is the Time-to-Line Crossing, which, as the name suggests, gives the amount of time before a vehicle crosses the line marking a lane and wanders over into another lane. To calculate the Time-to-Line Crossing, the lateral position, heading angle and speed are used. The driver has control over these parameters through the steering angle. For longitudinal control, several measures are proposed which, are Time to Collision (TTC), Time to Intersection (TTI) and the Time to Stop Line (TTS). The Time to Collision measure, indicates the amount of time before two vehicles in the driving environment collide since one is interfering with the path of another; there are several different forms of the Time to Collision measure which describe certain aspects of the event. One form of TTC is TTC braking (TTC_{br}) which is the time remaining before a collision at the point where the driver begins to apply the brakes; there is also the minimum TTC (TTC_{min}) which describes the point in time during the event where the time before a collision was at its smallest. This measure is particularly useful for describing how close a collision was to occurring. The Time to Intersection (TTI) measures the time needed to reach a major road when approaching it from a minor one. The TTI decreases as the vehicle approaches the intersection and it is affected by deceleration, it is possible to reduce the rate that TTI decreases through gradual deceleration. The Time to Stop Line (TTS) is a similar measure to (TTI); however, rather than referencing an intersection, a specific stop line is specified that may correspond to a certain object in the driving environment is specified.

2.7 Modeling Methods

Prior to developing driver models, it is important to establish what elements should be contained; MacAdam[18] provides a comprehensive list of these essential aspects as well as some secondary ones that may be used to enhance the model. In his work MacAdam quotes Rashevsky, who is of the opinion that the model must consider the vehicle and the driver as one entity that may not be separated. He also characterises human drivers based on physical limitations and physical attributes. Physical limitations refer to input channels that humans have such as visual, vestibular, kinaesthetic, auditory and tactile channels. MacAdam ranked the channels based on importance and, in his opinion, the most important is the visual channel, as information regarding velocity and position can be obtained through vision. He also reinforces his argument by indicating the fact that regulatory agencies often place a great level of importance in visual acuity tests. Following the visual channel, the next in terms of importance are the vestibular and kinaesthetic channels, as this relates to what the driver is feeling and from which acceleration information is extracted. The final two input channels

in terms of importance are the auditory and tactile channels, with more importance being placed on the tactile channel; however, according to MacAdam, both are useful at providing additional information with regard to the situation. The list of essential components for a driver model is based on physical attributes and consists of:

- a transport delay time to consider reaction times
- the use of preview to sense lateral and longitudinal control
- adaptation provisions for changing conditions
- a linear regime "crossover model" near what is known as the crossover frequency
- the presence of an internal vehicle model to estimate future responses

There is also a list of secondary components that are not as essential; however, they may serve to further enhance the driver model. The list includes the following components:

- provisions for processing incoming signals to account for neural delays, thresholding, etc.
- neuromuscular filtering elements for output channels such as steering, braking and throttle response,
- previewed path adjustment capabilities to account for skill related abilities or preferences in paths
- the ability to adjust speed according to upcoming lateral path requirements to improve path tracking
- provisions for surprises or unexpected situations
- inclusion of skill factors to account for different skills and experience levels

Some of the earliest driver models made use of control theory. This approach has been used by a number of researchers. Control theory methods can be considered appropriate for modeling, given that a driver is a very complex controller whose task is to maintain the course of the vehicle and arrive at the desired destination. One of the early control theory models was developed by Tustin; the model focused on the linear part of driver behaviour while the non-linear portions were regarded as a remnant[19]. Attempts were made to describe the remnant; however, they were relatively unsuccessful as they can generally only be regarded as an error which is quite difficult to model mathematically. An observation with regard to the remnant is that it is relatively small and that most of the behaviour is described by the linear portion; however, employing this method is not very well

regarded as there are inaccuracies that are known to exist in the model. Ultimately, the research by Tustin revealed that modeling human behaviour through the use of mathematical equations is extremely difficult given the amount of variation that exists in human behaviour, especially from person to person.

The Quasi-Linear model is another control theory model that was developed by McRuer and Krendel as discussed by Jürgensohn[19]. In order to model driver behaviour, second-order linear differential equations are used[19]. The model uses five parameters, which are the driver reaction time, the neuromuscular delay, and a gain along with both a lead and lag factor. To create a model using this method, these parameters must be found in a catalogue established by the researchers, which greatly limits the model's capability. Another model closely related to the Quasi-Linear model that Jürgensohn discuses as well is the Crossover model, which was also developed by McRuer and Krendel. This model is a somewhat simplified form of the Quasi-Linear model that uses an integrator and a phase correction for the reaction time and requires two parameters: the reaction time and the crossover frequency. One significant property of this model is that it assumes that the driver and the vehicle are one system; the model is said to often produce a fairly precise description of driver behaviour; however, it too is limited in the same manner as the Quasi-Linear model as it requires either empirical data or parameters from a catalogue.

Common practice in the development of driver models that use control theory principles is to divide the model in two separate parts, where one portion is responsible for longitudinal control and the other portion is responsible for lateral control[20]. A model of this nature is presented by Weir and Chao (Figure 2.7). The lateral portion of the model contains two feedback loops, one that considers the heading angle of the vehicle and the other that considers the lateral lane position. In addition to the feedback loops, the model also considers a random yaw rate disturbance. While the model is presented showing consideration for the lane position in the outer loop, it is possible to consider other parameters like path angle, sideslip and lateral acceleration; however, the latter two are less desirable since they are more difficult to perceive by the driver. A dynamic model of the vehicle is required, which is not difficult to obtain as it may already be known or can be measured if necessary. For the driver model there are two describing functions that are used that relate to variables chosen for the feedback loops. When constructing this driver model a combination of different approaches is used, which consists of the crossover model, control principles and experimental data acquired from either a driving simulator or an instrumented vehicle. The other portion of the model is for longitudinal control. This model is less complex than the one used to describe the lateral position as there is only one feedback loop with one describing function, because only the throttle pedal position is considered and braking is neglected. To increase the accuracy of the longitudinal model, a speed disturbance that may simulate environmental factors such as wind and changes in

terrain is added.

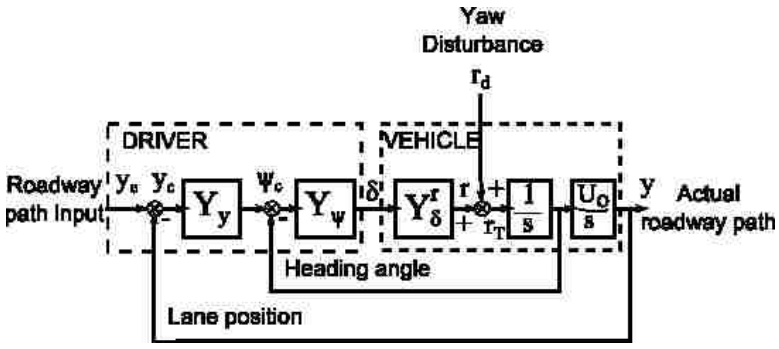


Figure 2.7: Lateral Control Model by Weir and Chao, [20]

Bengtsson et. al in conjunction with Volvo have developed a more comprehensive longitudinal model for use in the development of adaptive cruise control systems[21]. The model that they have presented considers both acceleration and braking. To construct the model, data was acquired from instrumented vehicles. For this particular research both a leading vehicle and following vehicle, as seen in Figure 2.8, are used, as adaptive cruise control systems base their vehicle speed on that of a vehicle ahead of it in traffic. Two different systems are used to consider the variables of interest as seen in Figure 2.9. The first system considers headway, velocity and differential velocity as inputs and brake pressure and throttle angle as outputs. In the case of the second system, the input is simply the velocity of the lead vehicle and the output variables are the headway, differential velocity, velocity, brake pressure and throttle angle of the following vehicle. To analyze the collected data, three different methods were used: a linear regression method, a state space model using subspace identification and a behavioural model. Little insight is given into the details of each method. The first two methods utilize the input and output variables described in system one, while the third method that is discussed uses the input and output variables described in system two. The linear regression method is used to determine if a correlation exists between any of the inputs and outputs. Ultimately this method proved unsuccessful, even with models with an order as high as 30. The residuals from the higher-order model were used to construct a pseudolinear regression model that proved to be more accurate. In the case of the state space model, a model of order of 15 was used; however, it too proved to be relatively unsuccessful at describing driver behaviour and like the linear regression method was somewhat better suited at describing the throttle angle rather than the brake pressure. Finally, the behavioural model provided a fairly accurate representation of driver behaviour. A model of order 30 is used; however, very few details with regard to this method are given.

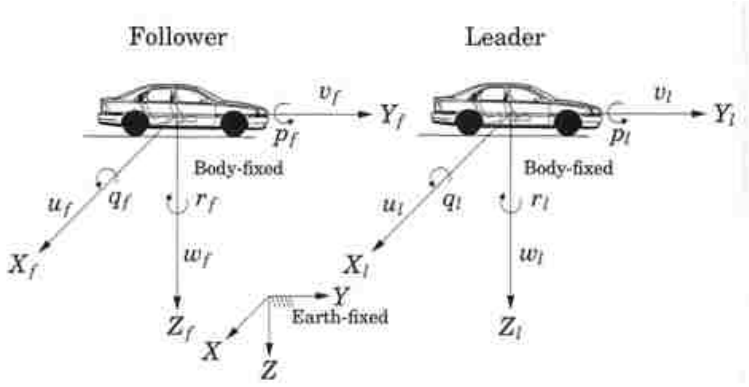


Figure 2.8: Leading vehicle and following vehicle, [21]

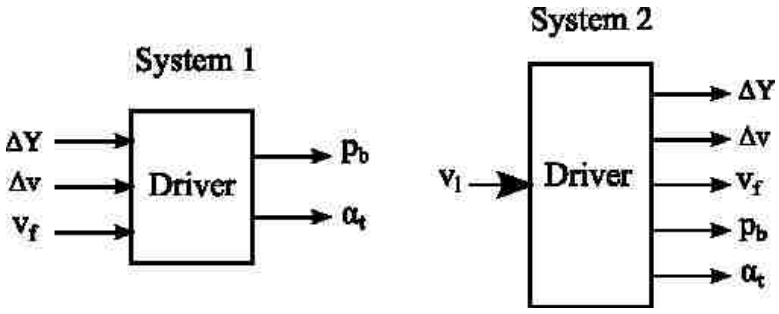


Figure 2.9: Two input and output systems used in the adaptive cruise control model, [21]

A lateral driver model based on adaptive predictive control was developed by Ungoren and Peng[22]. The basis of the model is to correct the error in the actual position of the vehicle on the road versus the desired one, as seen in Figure 2.10. The lateral position is determined as a function of vehicle state and the steering angle. The proposed model is said to contain three key components of human driving: the use of preview information, off-line adaptation and driving style. For the preview component of the model, selecting the proper preview time is critical since a longer preview window introduces an increase in tracking error; however, the vehicle is said to be more stable. This was found when validating the model and using preview times that were described as being long, medium and short in length. The initial assumption regarding steering angle was that it was fixed throughout the preview window; however, if it is adjusted throughout the preview window it reduces tracking error. Continuous adjustment of steering angle is not a realistic representation of driver behaviour; however, it is likely that some adjustment will occur. Therefore the model often accounts for one adjustment in steering angle during the preview window; this was found to have a similar effect on tracking as shortening the preview window. The model also accounts for driving style, as it was found that drivers control the vehicle based on different cues. For example, experienced drivers use yaw information more than inexperienced drivers. Ultimately, following data analysis from 22 subjects operating a driving simulator, it was found that results from the model correlated well for the most part with the data obtained from the driving simulator.

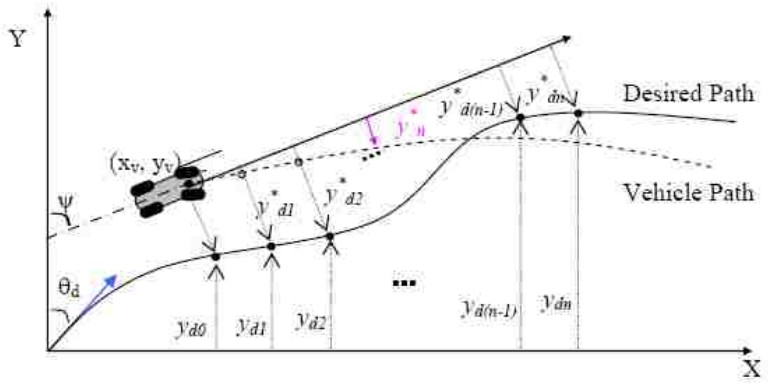


Figure 2.10: How the model corrects the actual path according to the desired one, [22]

A complete driver model using a control theory based approach is presented by Kiencke and Nielsen. The model that they have developed is termed the hybrid driver model and considers both longitudinal and lateral behaviour[23]. The longitudinal portion takes into consideration both acceleration and braking. The model is also more complete in that it considers how information is acquired and handled by the driver, the determination of reference values and finally the longitudinal

and lateral control models (Figure 2.11). For processing information, queuing theory is used and in the case of driver models, the driver is represented by a server and the incoming information is represented as clients. The queue temporarily stores these clients until they can be handled by the server. Different types of queues exist, some work simply on a first come first served basis, while some are capable of giving priority to certain clients. In the case of driver modeling, the latter type is best suited. For the driver model, two separate queues are used: one is dedicated to processing visual information, while the other is said to process vestibular information, which consists of motion that is perceived by the sense of balance. The next aspect of the model that is considered is the determination of a reference value, which, in this particular case, is the desired velocity of the vehicle. To determine the desired velocity, a finite state machine (Figure 2.12) that is based on the scenarios encountered in the road environment and consists of seven states numbered 0 through 6 is used. State 0 is the initialisation state when the automat is first called. State 1 (straight line): there is no road curvature within sight and the radius ($\rho(t+t_f)$) is equal to infinity. State 2 (approaching a curve): the driver can see a curve and may need to adjust the velocity of the vehicle, to calculate the appropriate reference velocity several parameters are used such as the curve radius, the road width and yaw angle. State 3 (braking): at this stage braking is necessary and the necessary amount of braking is determined in order to attain the desired velocity. State 4 (before the curve): at this stage the desired velocity is nearly attained; however, some light braking may still be required. State 5 (curve): the vehicle is now travelling at the desired velocity, minor adjustments may be necessary based on lateral acceleration and what the driver feels. State 6 (accelerating): the final state where the driver begins to accelerate, this may occur slightly before the curve ends. The final portion of the model is the controller itself. As with many models two separate systems are used for longitudinal and lateral control with both using a General Predictive Controller. For the longitudinal controller, the inputs are the desired and actual acceleration while the outputs are the throttle angle and the brake pedal force. To determine the appropriate amount of engine or braking torque required, the controller uses maps that contain both engine and brake torque with respect to both desired acceleration as well as vehicle velocity. While the braking portion of the controller is very similar to the accelerating portion, it also adds the ability to consider the effect of engine braking. The operating principle used for the lateral controller is to minimize the offset of the vehicle to an ideal line on the road, the output of this controller is the steering angle and the offset is measured from the centre point of the front axle.

While many of the methods used to develop driver models utilize traditional control theory methods, there are alternate methods such as neural networks or fuzzy logic, to name a few. Neural network methods have been successfully applied by Lin et. al, where the objectives of their work were to employ and compare different types of neural networks as well as to develop a driver-

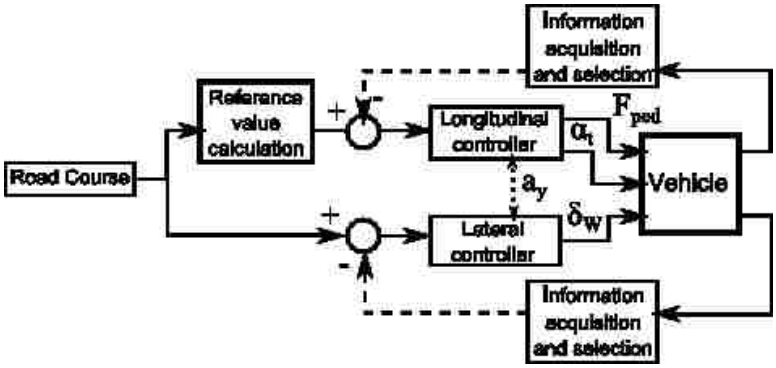


Figure 2.11: Model structure proposed by Kiencke and Nielsen, [23]

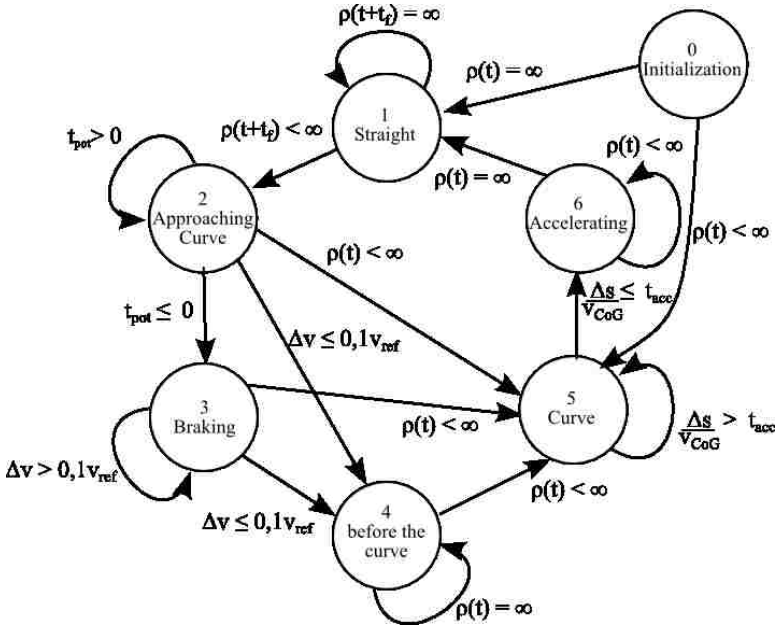


Figure 2.12: Finite State Machine used to calculate the reference velocity, [23]

vehicle-environment (DVE) simulation system[24]. To model the driver handling behaviour in the DVE both mental behaviour and physical action are considered. In their study, three different manoeuvres were considered: a single line motion which is typically used in a lane change, a double line motion as seen in the overtaking of another vehicle and sine line motion which is used while negotiating an s-curve turn. The inputs to the neural network that are used in this model are the yaw rate, lateral velocity, lateral acceleration, roll angle, roll angle velocity, lateral displacement and preview lateral offset. The outputs of the system consist of the steering wheel angle as well as the steering wheel angle velocity. Figure 2.13 presents the general architecture for feedforward neural network structures.

To implement the neural network model, three different types were used: Counter Propagation Network (CPN), Radial Basis Function Network (RBFN) and Back Propagation Network (BPN). It should be noted that the authors omitted any details with regard to the BPN development. The CPN model consists of an unsupervised training layer known as the Kohonen layer and a supervised training layer known as the Grossberg layer. The Kohonen layer deals with information compression and pattern recognition which greatly reduces training time while the Grossberg layer has the role of ensuring accuracy. In the case of the RBFN model, it is initially considered to have two layers where one layer is hidden. The hidden layer is then redefined as a non-linear function in the input layer. In order to train the neural networks, data was required; the data, in this case, was collected from instrumented vehicles. Five male drivers were the test subjects ranging in experience from one to twenty years; each driver drove five different cars alternately. The drivers were required to drive one hour each day at a required velocity between 30 and 90 km/hr. To evaluate the various models, training time, error tolerance and accuracy were measured. In terms of training time, the RBFN and CPN models were very close, with the RBFN requiring slightly less time while the BPN took substantially longer to train. For error-tolerance, the CPN model yielded the best result; this is attributed to the weight update scheme. The RBFN model was the next best, while the BPN model was the worst. In terms of accuracy, the RBFN model was the most accurate and the CPN model was the least accurate. To validate the simulation results, experimental data was collected from two drivers that drove along the prescribed motions as mentioned previously; the data collection was performed in a parking lot in order to eliminate any uncontrollable disturbances. Following comparison with the experimental data, it was determined that the simulation does in fact yield an output that is in agreement. One aspect that is noted by the authors is that the simulation often appears to lag the experimental data, which they attribute to the lateral acceleration approaching the validity limit of the vehicle dynamics model (0.4g). The conclusion reached by the authors is that the RBFN model is well suited for driver modeling and that the agreement with the experimental data represents a good fault tolerance.

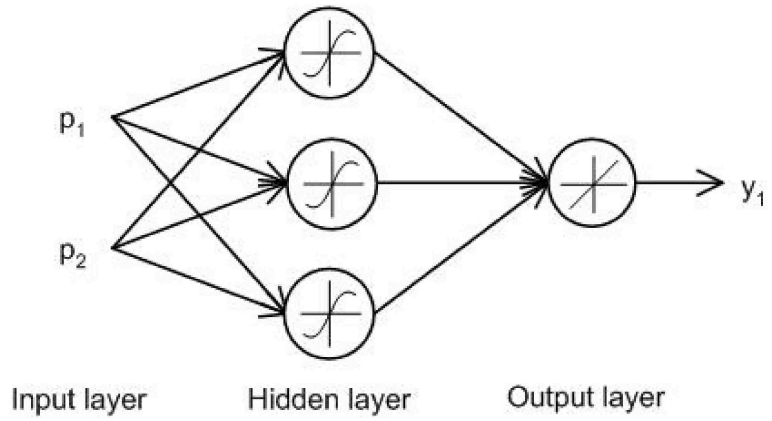


Figure 2.13: General feedforward neural network architecture, [25]

2.8 Potential Applications for Driver Models

Up to this point, most of the research into driver modeling has been for the purpose of developing driver assistance systems or driver information systems. The models are used to study both the effects of implementing these devices on driver behaviour as well as to design them to be more effective. There exists several other applications for the use of driver models, one area in particular is the optimization of drive cycles in hybrid vehicles. Hybrid vehicles utilize two methods of propulsion and as a result energy management is a significant issue in their development; driver modeling can aid in optimizing fuel economy and maintaining proper charge sustenance. To achieve this goal, a control device will determine the appropriate torque distribution. Also some systems may actually work with the driver to either alert them or assume control in order to operate in a manner that increases fuel efficiency.

The work of Ishio et. al[26] presents a driver model that was used to assess the handling characteristics of vehicles. The motivation for using a driver model in this application is that the behaviour of the vehicle may be the result of the driver rather than the vehicle characteristics, particularly in the case of subjective evaluations. According to the authors, accurate evaluation of vehicle handling performance is said to have an increased importance with the development of active chassis controls and understanding their effects on vehicle performance. By applying a driver model based on steering angle and vehicle trajectory data during a lane change, a common reference can be used to evaluate handling performance; in this particular case the effects of drive-by-wire steering and direct yaw control were evaluated. MacAdam also discusses the use of driver modeling to evaluate handling to simulate how a driver would react during a sudden tire failure during an obstacle avoidance manoeuvre[18].

The application of driver modeling for accident reconstruction has been undertaken by Jurecki & Stanczyk[27]. In their work they state that there are many factors that contribute to accidents thus making it difficult to determine the exact cause. In their opinion accident reconstruction is the next logical step for the application of driver modeling, following their use for vehicle development and then vehicle control systems development. The authors focus on the concept of risk time, which is a function of both vehicle speed and the distance to an obstacle and describes how much time a driver has to react to a scenario. A relationship between reaction time and risk time was discovered with reaction times increasing with risk time. This relationship is to be expected as drivers will have more time to react and they will be slower to do so, as seen in Figure 2.14. To characterise driver behaviour in pre-accident situations, braking and steering manoeuvres were evaluated along with risk time; by using this data in conjunction with a driver model previously developed by the authors, it is believed possible to reconstruct an accident and determine the cause, since there is now more insight on how drivers will behave when presented with an accident scenario.

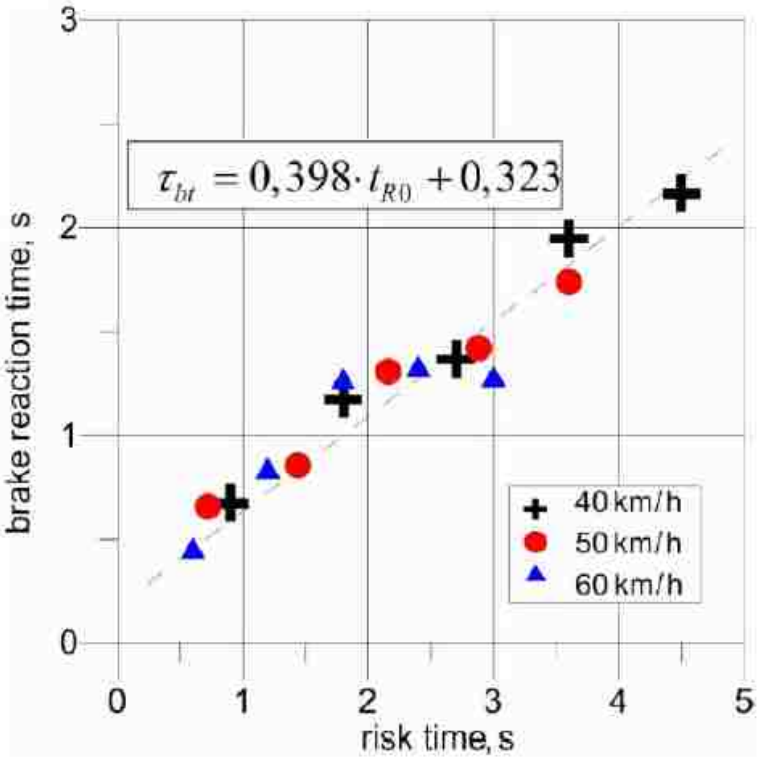


Figure 2.14: General relationship between reaction time and risk time, [27]

Another potential application of driver models is for risk assessment of drivers. Such an applica-

tion will be very useful for the medical field, as currently there are few objective methods to evaluate driver risk. This subject is particularly sensitive in the case of elderly drivers as family members who are concerned for their safety may inform their doctor that they feel such a person may be no longer fit to drive. Subsequently, doctors may take a safe approach and remove the licence on medical grounds, while the person may still be fit to drive in a limited capacity. Ultimately when the licence is revoked, the person becomes much less independent and their quality of life will suffer. The goal of the authors' research in this topic is to establish a method to evaluate drivers using a driving simulator, which will give doctors a more definitive and objective way of determining driver fitness.

Chapter 3

Theory

3.1 Artificial Neural Networks

3.1.1 General Information

While there are many possible methods to create a driver model, as described in the previous chapter, for this particular project Artificial Neural Networks were chosen. Although it is unknown if this is necessarily the best method to create a driver model, part of the goal of this research is to discover whether or not this is the case. The principle behind Neural Networks is to mimic the function of the human brain; they are capable of learning the behaviour of a certain phenomenon based on provided information. Neural Networks are capable of representing a broad range of systems but there is one major distinction in that they can be used for either function approximation or data organization. Typically the systems that are modeled using neural networks fall into one of the two aforementioned categories.

In order to develop a neural network, it must first be trained; this is done by presenting a dataset that contains both input and desired output, or target variables. The goal of training is to allow the network to be exposed to the input variables that stimulate the system along with their corresponding response; using this information the network is able to learn how the system will behave under the conditions characterized by the presented training dataset. This type of training is known as supervised training. Unsupervised training is another form of neural network training and it is typically used in clustering and pattern recognition applications. The idea behind unsupervised training is for the network to group like variables together when presented a dataset. It does this by assuming that inputs within a certain region of the input space are similar and thus represent a similar behaviour or characteristic. An important point to note is that neural networks are very effective at making generalizations and very effective at interpolating; however, it is widely accepted

that neural networks are not particularly strong at extrapolating data [28]; therefore, when selecting a dataset it is important to ensure that the number of data points is sufficient to span as large an area as possible (or practical) such that the network is trained for the widest number of possible scenarios.

Artificial Neural Networks are structured very much like their biological counterparts; they contain elements known as neurons (Figures 3.1 & 3.2) which are connected to one another as well as to data. The network may contain multiple layers of neurons; however, there is a minimum of three layers: The input layer, which, like the name suggests contains the input data. The next layer is the hidden layer (or layers as often more than one may be utilized). This is where much of the generalization is performed. The final portion is the output layer where the network gives the output representing the response of the system. The hidden layers and the output layer are comprised of a series of neurons. In the case of the output layer the number is defined by the number of output variables; however, for the hidden layer the number is somewhat arbitrary. The goal is to achieve a desired level of accuracy with a minimal number of neurons.

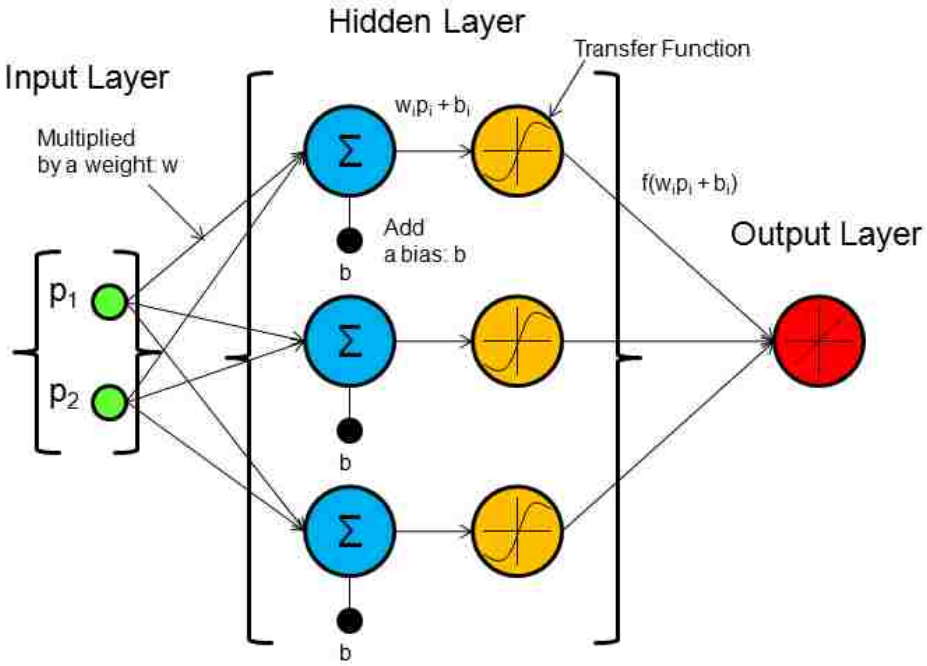


Figure 3.1: Structure of a feedforward neural network, Adapted from: [25]

In order to further understand the operation of a neural network it is important to understand what exactly a neuron is (see Figure 3.2). Neurons are comprised of three main elements: a series

of weights, a bias and a transfer function. The role of the weight is to connect network inputs to the neuron or to connect the output of one layer of neurons as inputs to another. The number of weights in each neuron corresponds to the number of inputs to that particular neuron, in the case of a single hidden layer network, a hidden layer neuron will have the same number of weights as network inputs while an output layer neuron will have the same number of weights as there are hidden layer neurons. Each weight is multiplied by its corresponding input and all of the products are summed together. The role of the bias is very similar to that of the weight; however, each neuron only contains one and in some cases a bias may not be used at all. Essentially a bias is weight that acts on an input equal to unity in the same way that a weight acts on an input from the dataset. It acts to shift the sum of the weighted inputs by the bias value. The final component of neuron is the transfer function which may also be referred to as an activation function. A variety of functions can be used as the transfer function in the neuron, the most important distinction being between linear and non-linear transfer functions. In the case of hidden layer neurons, the transfer function used is generally non-linear with the most common being either the logistic function (Figure 3.3 & Equation 3.1)[29] or the hyperbolic tangent function (Figure 3.4 & Equation 3.2)[29]. In the output layer typically a simple linear activation function is used.

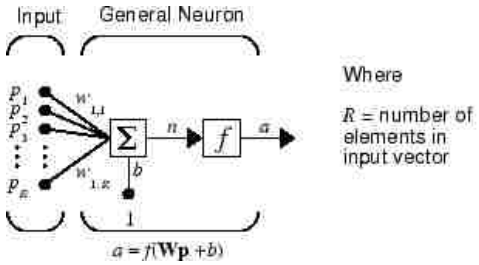


Figure 3.2: Structure of a single neuron, [28]

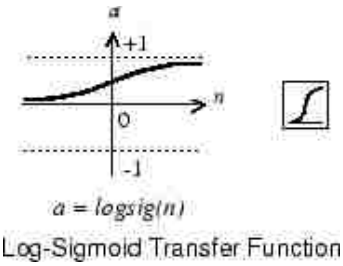


Figure 3.3: Plot of Logistic Function, [28]

$$[htpb]f(t) = \frac{1}{1+e^{-t}} \tag{3.1}$$

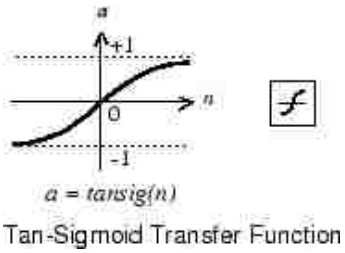


Figure 3.4: Plot of Hyperbolic Tangent Function, [28]

$$[htpb]f(t) = \frac{1+e^{-t}}{1-e^{-t}} \tag{3.2}$$

Having defined all the critical elements of the neuron it is now possible to describe how they all work together to approximate the behaviour of a particular system. First, each input will be passed to each hidden layer neuron where, as mentioned before, they will be multiplied by the corresponding weight. The next step is to sum the products of all the inputs multiplied by the weights and to add the bias. Each summation is then passed through the transfer function at, which point they become the outputs of the neurons. The same procedure is repeated through the next hidden layer or the output layer depending on the overall network structure. Updating the weights is the method by which the network learns to model the system. As an initial starting point an arbitrary set of weights is chosen and the data is passed through the network. The output of the network is then compared with the target output of the system and the error between the two outputs is calculated and then used to update the weights using one of many available training algorithms which will be discussed in greater detail in the following section. Please note that the aforementioned process is specific to supervised learning.

3.1.2 Training Algorithms

As discussed briefly in the previous section, neural networks learn by updating the weights using one of many available training algorithms. When discussing training algorithms, there is one major distinction that must first be made, which is between first and second order method methods. First order methods use the gradient of the error surface (see Figure 3.5) as a function of the weights to select a path of descent to a minimum point on the error surface corresponding to the minimum error; hence, they are often referred to as gradient descent methods. The error gradient is backpropagated

through the neural network to update the weights until a satisfactory minimum error is achieved. Several variants of gradient descent methods exist which will be discussed later in this section. In the case of second order methods, the second derivative of the error surface with respect to the weights is used to determine the curvature of said surface, which is used to update the weights. Second order methods are more powerful and have a tendency of training faster; however, they do require more computing resources. It is also important to mention that there are several ways of calculating the error; a commonly used method is the mean squared error or MSE (Equation 3.3)[29].

$$E = \frac{1}{2N}\sum_{i=1}^{N}(z_i - t_i)^2 \tag{3.3}$$

where,

$$\begin{aligned} E &= \text{the error} \\ N &= \text{the size of the training set} \\ z &= \text{the network output} \\ t &= \text{is the target output} \end{aligned}$$

The simplest form of neural network training is backpropagation learning. As the name implies the error associated with the neural networks for a given set of weights is used to update the weights for the next iteration. The formula for updating the weights using backpropagation training is given in (Equation 3.4)[29]. Essentially the direction of steepest descent is determined by noting that it is in the opposite direction of the previously calculated gradient, d_m[29]. The gradient is then multiplied by a pre-determined factor called a learning rate, ε, and it is then added to the current weight. The only parameter that the analyst controls is the learning rate which is essentially optimized through trial and error, ranging between 0 and 1. The ideal learning rate is able to quickly and efficiently determine the minimum error yet still descend the error surface in a smooth manner to the global minimum. If the error rate is too small it may take a long time to attain convergence and if it is too large it may oscillate around a solution yet never reach the minimum point. A parameter known as momentum is an additional term that can be added to the backpropagation algorithm (Equation 3.5[29]). Once again the parameter ranges between 0 and 1 and its purpose is to consider previous weight changes in order to dampen potential oscillations.

$$\begin{aligned} w_{m+1} &= w_m + \Delta w_m \\ \Delta w_m &= -\varepsilon d_m \end{aligned} \tag{3.4}$$

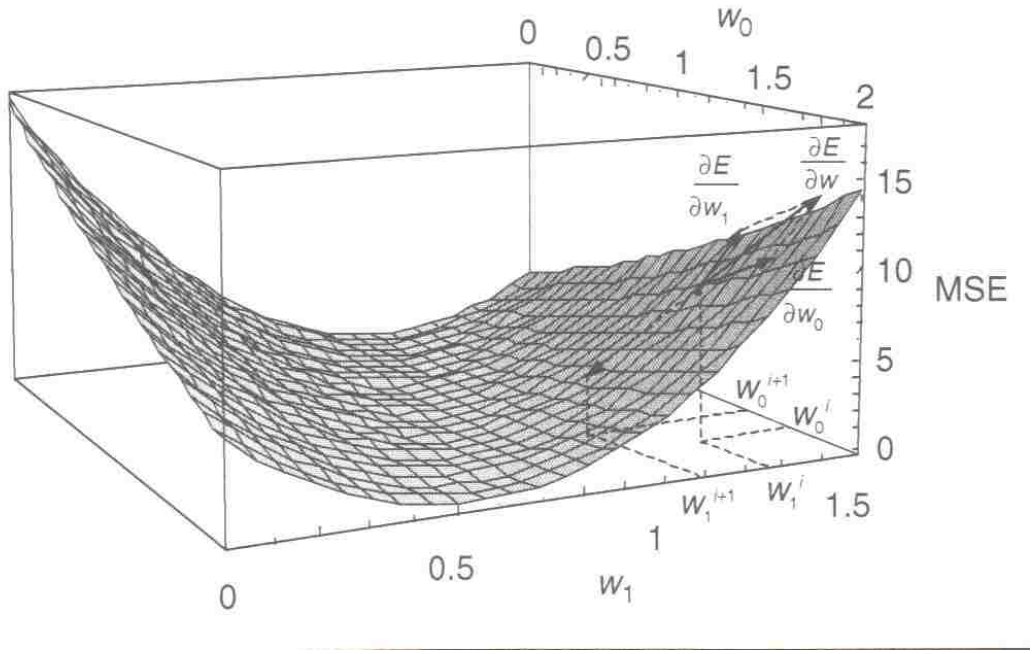


Figure 3.5: Mean Square Error as a Function of Network Weights, [29]

where,

$$w = \text{the weight}$$
$$\varepsilon = \text{the learning rate}$$
$$d_m = \text{the derivative or gradient.}$$

$$\Delta w_m = \mu \Delta w_{m-1} - (1-\mu)\varepsilon d_m^w \tag{3.5}$$

where,

$$\mu = \text{the momentum term}$$

Another commonly used set of first order training algorithms uses conjugate vectors to locate the minimum error. They are commonly referred to as conjugate gradient methods. Conjugate gradient methods differ from steepest descent methods in that descent of the error surface follows the direction of a vector that is conjugate to the steepest descent vector. In simple terms a conjugate vector is orthogonal to the steepest descent vector; see Equation 3.6 for the definition of conjugate. The advantage of conjugate gradient algorithms (Equation 3.7) is that they are supposedly capable of obtaining second order information without the need to actually calculate the second derivative. Unfortunately some of this information may be lost through a procedure called restarting where after a certain number of iterations, the solution restarts following the direction of steepest descent. Restarting must be employed to increase the rate of convergence, otherwise it is only linear[30]. There are several theories on how often the algorithm must be restarted. A discussion of such methods is beyond the scope of this thesis; further reading is available in other publications, [30, 31, 32]. The main difference between the variants of conjugate gradient methods is the manner in which β_k in Equation 3.7 is calculated, the method shown is just one example.

$$p_i^t G p_j = 0 \text{ when } i \neq j \tag{3.6}$$

where,

$$G = \text{ an arbitrary matrix}$$
$$p_i^t \ \& p_j = \text{ two vectors that are mutually conjugate to one another}$$

$$
\begin{aligned}
w_{k+1} &= w_k + \alpha_k p_k \\
p_{k+1} &= -g_{k+1} + \beta_k p_k \\
\beta_k &= \frac{y_k^t g_{k+1}}{y_k^t p_k} \\
y_k &= g_{k+1} - g_k
\end{aligned}
\tag{3.7}
$$

where,

$w =$ the weight

$g =$ the gradient.

$p =$ the search direction

$\alpha =$ appears to be the learning rate however this is not explicitly stated

$k = k_{th}$iteration

In the case of second order training algorithms, they make use of information regarding the curvature of the error surface to locate the minimum[29]. The major advantage of such methods is that they tend to be both quicker and more accurate. A significant disadvantage with second order methods is that they tend to be computationaly intensive as the Hessian matrix must be calculated. The Hessian matrix (Equation 3.8)[29] contains the partial second derivatives of the error with respect to the weight and as the size of the network increases so does the complexity of the Hessian matrix. There are two well known second order methods: the Gauss-Newton method (Equation 3.9)[29] and the Levenberg-Marquardt (L-M) method (Equation 3.10)[29]. These training algorithms are almost identical; however, the L-M method contains a conditioning term that can force it to operate like the Gauss-Newton method when the term is set to a small value, and like a first order method when it is large. The advantage of the L-M method over the Gauss-Newton method is that there may be occasion for the second derivative information to lead the solution towards a maximum, thus increasing the error. Because the L-M method can operate as a first order method, it avoids this [29]. As a result, the L-M algorithm is often viewed as being the best one as it is often the fastest and the most accurate. As a word of caution, as the network increases in size, the speed advantage of the L-M method diminishes with respect to the other algorithms and it has a tendency of requiring a significant amount of memory, particularly when using large sets of training data. Also related to the second order methods are what are called quasi-Newton methods. A common one of these methods is known as the Broyden-Fletcher-Goldfard-Shanno(BFGS) algorithm[33]. This method operates in a similar fashion to the second order methods; however, it does not need to directly compute the

Hessian matrix thus allowing it to run using fewer system resources.

$$H = \begin{bmatrix} \frac{\partial^2 E}{\partial w_i^2} & \frac{\partial^2 E}{\partial w_i \partial w_j} & \cdots & \\ \frac{\partial^2 E}{\partial w_j \partial w_i} & \ddots & & \\ \vdots & & & \\ & & & \frac{\partial^2 E}{\partial w_m \partial w_n} \end{bmatrix} \tag{3.8}$$

$$\Delta w_m = -\varepsilon \frac{d_m}{d_m^s} \tag{3.9}$$

where,

$$\begin{aligned} w_m &= \text{the weight} \\ \varepsilon &= \text{the learning rate} \\ d_m &= \text{the 1st derivative of the error surface} \\ d_m^s &= \text{the 2nd derivative of the error surface} \end{aligned}$$

$$\Delta w_m = -\frac{d_m}{d_m^s + e^{\lambda}} \tag{3.10}$$

where,

$$e^{\lambda} = \text{the conditioning term}$$

3.1.3 Types of Neural Networks

Until now, the discussion regarding neural networks has focused mainly on the multilayer perceptron or MLP, which is the most basic form of neural network; however, many different types exist with each being suitable for a different purpose. As briefly mentioned earlier in the preceding sections, there are clustering networks which are used to organize data. The way in which they work is that they will take a dataset and look for certain trends in the data. Based on these trends the data will then be grouped into various clusters in which the data exhibits similar characteristics. Such architectures are often used for pattern recognition problems in order to classify information of some sort. Examples of fields that use them are the medical field and image processing.

Another type of network is a variant of the mutlilayer perceptron network known as the the nonlinear autoregressive network with exogenous inputs or NARX; these are known as dynamic networks and they are distinguished by the fact that their training takes into consideration previous inputs, outputs and states of the network[28]. To accomplish this, the inputs are delayed and either the network output is looped back into the network as shown in (Figure 3.6) or the target value from

the same state is used instead (Figure 3.7). The latter architecture is said to be more accurate and based on the author's experience, it appears to train significantly faster.

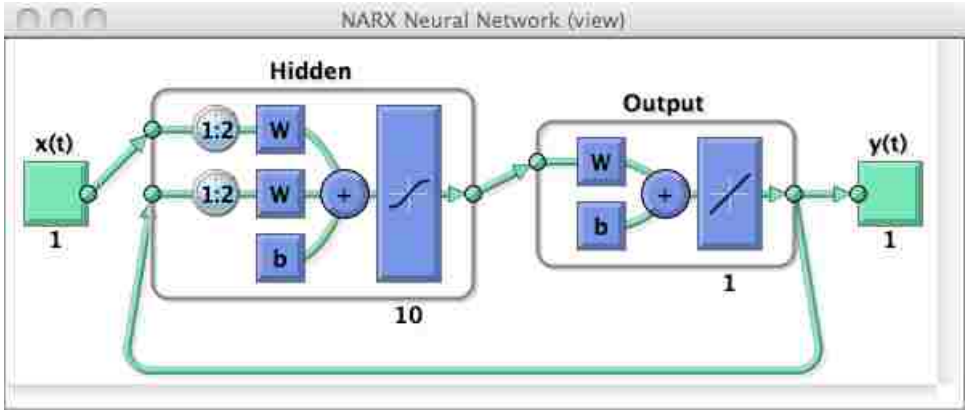


Figure 3.6: Closed loop NARX network, [28]

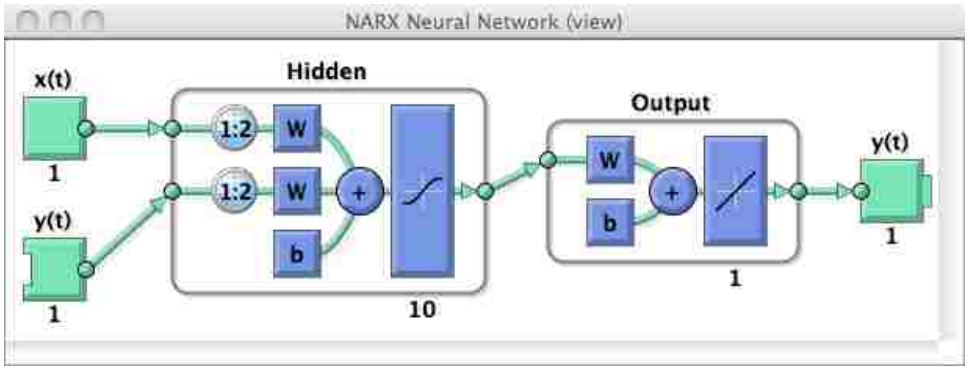


Figure 3.7: Open loop NARX network, [28]

The final type of network of interest is the Radial Basis Function Network or RBFN. The concept of the RBFN is different from other types of networks altogether. The main distinction of RBFN's is that their transfer function is a Gaussian function (Equation 3.11)[28] where the weight of the neuron acts as the center of the function in the input space[28, 34]. The distance between the input and the weight (center) is calculated; if the distance is close to 0, the output of the radial basis function is close 1 (Figure 3.8). Another property of the transfer function is called the spread which defines slope of the function around the center of neuron in the input space. A large spread means that the function will have a shallow slope and therefore more data will have a higher correlation, and small spread indicates the function will have a very steep slope around the center. When designing such a

network, the spread is the only parameter that the analyst can tune to optimize it. Another important point to note is that there exists what is called an exact RBFN where the size of the network is dictated by the size of the training set, such that there is a neuron for each data point in the input set whose center is equal to that particular input. The main drawback is that the network becomes very large, perhaps too large and inefficient. An alternative to this approach is start with one neuron and train the network by continuously adding a neuron and updating the weights until a network deemed acceptable according to the standards set forth by the analyst is created.

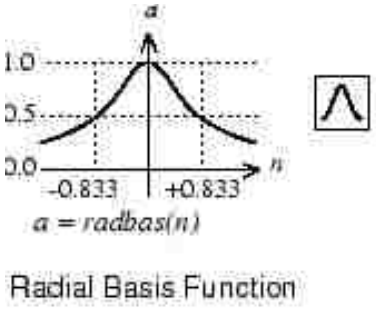


Figure 3.8: Plot of Radial Basis Function, [28]

$$f(x) = e^{-|x-w|b} \tag{3.11}$$

where,

$x =$ the network input

$w =$ the weight or neuron center in the input space

$b =$ the spread

3.2 Vehicle Dynamics

While the focus of this thesis is not on vehicle dynamics, it is the opinion of the author that it is important to discuss some elementary vehicle dynamics properties given the influence they have on vehicle handling, which ultimately influences the driver's decisions. Most of the decisions that a driver will make are either based on visual information (what the driver sees) and vestibular information (what the driver feels). Other properties are used as well; however, these are the two principal ones.

Vehicle dynamics is used to describe the overall behaviour of the vehicle. There are two significant regimes of interest which are linear and non-linear. The handling behaviour is very closely

related to the behaviour of the tire and the property known as slip. In the longitudinal sense it is known as the slip ratio and in the lateral sense it is known as the slip angle. Until a certain degree of slip the tire will behave in a linear fashion; however, beyond that point the behaviour becomes nonlinear and therefore more difficult to predict[35]. As a result, the average driver is restricted to the linear regime and the car is much more predictable. Typically, if an average driver enters the non-linear range of operation, the result is often dangerous as they typically reach the grip limit of the tires. It is only highly trained and skilled drivers that are capable of operating in this range. The data used in this study likely falls within the linear regime.

Another important aspect of vehicle dynamics is handling and the vehicle's steering tendencies. Vehicles can handle in three different manners; they will either understeer, oversteer or experience neutral steering. In the case of understeer, the vehicle will have a tendency of taking a corner with a wider turning radius if the speed is increased[35]. During oversteer the turn radius will decrease with speed, and neutral steer occurs when the vehicle keeps the same radius. Ideally neutral steering is desirable; however most vehicles are designed to have some degree of understeer. The reason for this is that understeer is a stable state such that if the driver were taking a curve too fast they would simply need to reduce their speed in order to safely negotiate it. In the case of an oversteering vehicle, it can become very unstable before the driver can safely realize it; therefore, such vehicles are prone to losing control and spinning suddenly. Once again, there are some applications where highly skilled drivers may prefer some degree of oversteer; however, its use in everyday applications is highly unrecommended and potentially dangerous. There are several factors that influence the handling characteristics of a vehicle: the first being weight distribution and the location of the center of mass, the next being the lateral force being developed by the tires which, in turn, is influenced by the properties of the tires themselves as well as by the stiffness of the suspension.

A simple model known as the bicycle model (Figure 3.9) has been developed to help illustrate the handling behaviour of a vehicle. This particular model neglects lateral weight transfer; however, it does provide some good insight on the forces affecting vehicle handling. The bicycle model illustrates that it is the lateral forces generated by the rear tires of the vehicle that provide an understeering moment as all moments act about the center of gravity. Because understeer is a stable state, the moment generated by the rear tires is sometimes referred to as a stabilizing moment. What the bicycle model illustrates is that a vehicle with with a center of gravity towards the front will tend to understeer and thus be more stable. As a result a large majority of road vehicles are designed as such. Given this fact it is safe to assume that any vehicle modeling for the development of a driver model used to evaluate road users should exhibit the aforementioned characteristics.

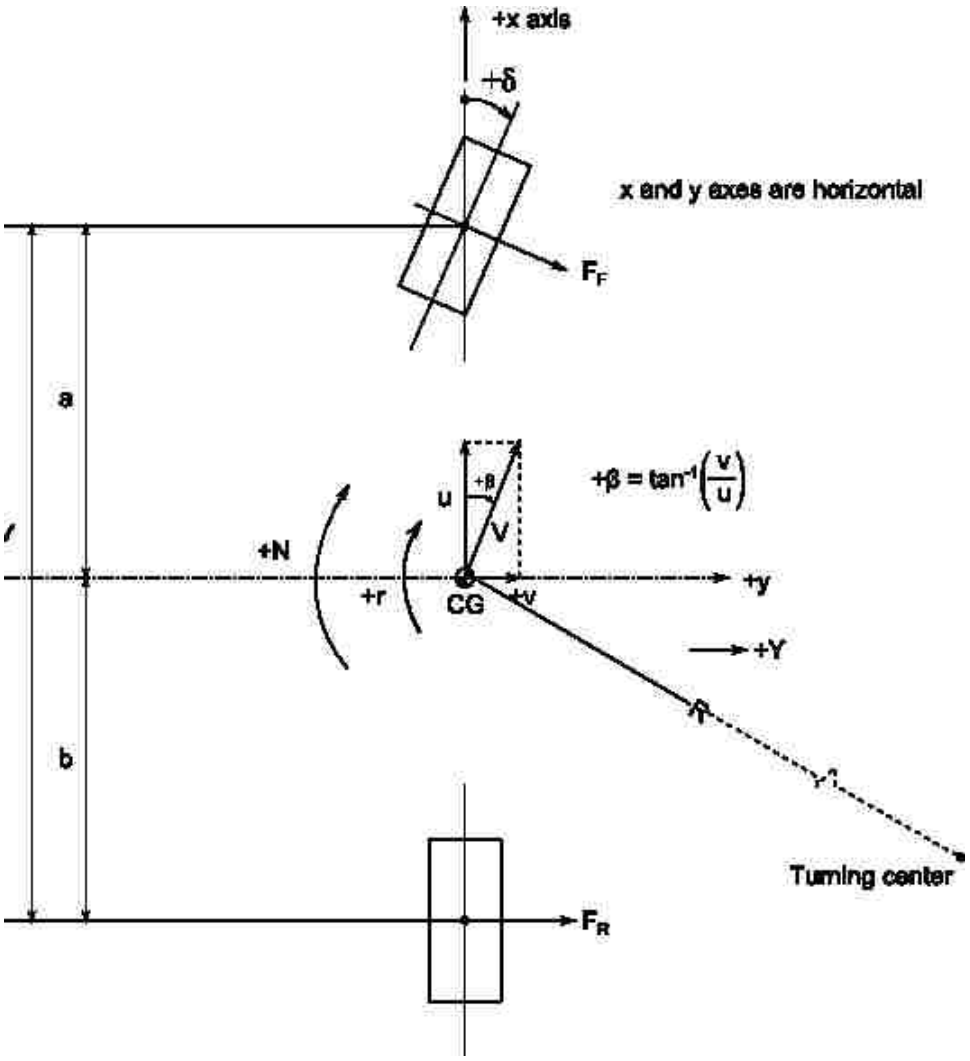


Figure 3.9: Bicycle Model, [35]

Chapter 4

Training Data

In order to develop a neural network, a set of training data must be acquired such that the network can learn the behaviour of the system, as mentioned in Chapter 3. To develop a driver model, the training data must consider three key elements, which are the driver, the vehicle and the road environment. It is important to recognize that for the purpose of this research the driver is nothing more than a controller, albeit a very sophisticated controller. As a result there will be both a set of input data as well as a set of output data. Drivers will use information of the vehicle's behaviour and information from the road environment to make their decisions on the vehicle control inputs to use. This information will constitute the input variables, while the driver's actions will be the output variables.

In order to obtain driving data, there are two methods available. The first is to instrument a vehicle and ask a selected group of people to act as test subjects and to drive a specified route in the vehicle. While this occurs their actions, along with the vehicle behaviour are recorded. The other method is to set up a driving simulator on a computer. The former method has the advantage of being more accurate since all of the factors in terms of the vehicle's performance and roadway information can be considered. The drawback to this method is that a lot more resources are required, which can increase the cost as well as the time needed to acquire the data. The latter method using a simulator is much easier and much cheaper to use, since the subjects can perform the test in an office environment and fewer resources are required. The issues with using a driving simulator are concerns with accuracy. Constructing a simulator that can accurately model the behaviour and response of a vehicle can be very difficult and costly. As an example, when driving a vehicle on the road it is likely to behave differently than in a simulator due to non-linearities and the behaviour of the components' materials, particularly in the case of the tires. The question of accuracy also calls into question the overall validity of any test administered on a driving simulator.

4.1 Driving Simulator

To acquire data, the Groupe de Recherche en Analyse du Mouvement et Ergonomie (GRAME, translated: Movement and Ergonomics Analysis Research Group) at l'Université Laval in Québec City, PQ; was contacted as they possess a driving simulator and have substantial experience running simulations, particularly for elderly drivers. The simulator is developed by Systems Technology Inc.(STI®), it is referred to as STISIM and it operates their Drive 2.0 software[36]. STI® offers a variety of simulator configurations both in terms of hardware and software; descriptions of the different types of simulators are listed in [37]. The arrangements offered range from very simple setups that are relatively inexpensive and can be used with a desktop computer and controls used for gaming to sophisticated cab style simulators with projectors that provide steering feedback to the driver. The simulator installed at Laval shown in Figure 4.1 is an open cab simulator; however, it does not offer any steering feedback to the driver. The simulator uses an image approximately 1.45m high and 2.0m wide projected onto a flat wall situated approximately 2.2m away from the simulator. The field of view generated by the projector is 40 degrees horizontally and 30 degrees vertically. The simulator is also equipped with a video camera to record the driver's actions[36]. All of the driving simulator interfaces are instrumented such that all of the steering wheel, throttle pedal and brake pedal movements are recorded. The simulator can record 50 data channels[38]; however, only a select number of the channels are used for model development. Ultimately, the channels of interest for this project are the longitudinal velocity and acceleration, V_x and A_x, respectively, the lateral position, velocity and acceleration, Y, V_y and A_y, respectively, speed limit, V_{lim} and road curvature, ρ. These seven channels will act as the input variables for the neural network. The steering wheel angle, θ, throttle pedal, T_p input and brake pedal input, B_p, will act as target variables for the network as they will be the output from the network once it is simulated.

In terms of software, the vehicle model is based on a simulation tool developed for the NHTSA known as Vehicle Dynamics Analysis NonLinear (VDANL)[37]. As the name suggests, the model provides a non-linear analysis of the vehicle performance, and it includes a sophisticated tire model referred to as STIREMODEL. The motivation, according to STI®, for using a high fidelity vehicle dynamics model is "to achieve realistic feel and motion cueing and to be able to provide hardware-in-the-loop interaction with elements such as steering and braking"[37]. STI® does acknowledge that some of the computer analysis components of the model have been disabled in order for the model to be able to run in real time, they refer to the modified model as VDANL/RT, this variation also has an additional input-output component added to it. The software used in the simulator at Laval is a simplified model that is linear and according to STI® "has no notion of individual wheels". The linear vehicle dynamics model also models the throttle and braking differently than the more advanced mode. Typically throttle position is represented between 0 and 1, where 0 is

no throttle inout and 1 is full throttle input, and braking is typically measured either using the force required to move the pedal or the brake master cylinder pressure. Instead, the linear model considers the accelerations due to throttle and braking; it does, however, keep a record of the raw inputs. This does create some complications for the development of the model that will be discussed in Section 4.3. Another complication that arises from the use of the simplified model is that because there is no notion of individual wheels, there is no steering ratio between the angle of the steering wheel and angle of the steered wheels at the road. The lack of a steering ratio is problematic when validating using a vehicle simulation program such as CarSim® as it uses the angle at which the wheels are being steered while the data from the simulator is the angle of the steering wheel; without a steering ratio there is no way of relating these two parameters. Ultimately the simplified linear model may be a satisfactory and more cost effective solution for the purposes that it is being used for by GRAME in terms of giving an estimate of driver performance from the standpoint of observing physical abilities. A non-linear model would provide a more realistic model and allow for the creation of a more accurate representation of the driver's behaviour, particularly when constructing the model using a neural network. Such a model would be more costly; however, future work should endeavor towards using a non-linear model. Further discussion regarding this idea will be made in the conclusions and recommendations chapter.

Figure 4.1: Setup of Driving Simulator at l'Université Laval, Photo Courtesy [39]

The data that is used in the neural network is distance-dependent as opposed to time dependent. This approach is taken because the subjects completed the driving scenario at different rates. Basing data extraction on time is not appropriate as the proper events will not be recorded; however, because the scenario is the same for all subjects, extracting the data based on the distance traveled will provide the desired events. The sample rate of the simulator is constant at 30 Hz., therefore the datasets will all be of different sizes for different subjects depending on the amount of time taken by the subject to complete the scenario. In this discussion, scenario describes the entire course that the subject is asked to follow. It is important to note that while it is not explicitly recorded, time is considered in the input dataset as each segment is a timeseries concatenated to form one large dataset. Training the neural network requires presenting it with the expected outcome for a given series inputs. Based on that expected outcome a generalization is made, therefore, the time at which each event occurs is ultimately irrelevant.

4.2 Data Acquisition

The data used for training was obtained from four different sets. Two of the sets were for a study regarding elderly drivers where the subjects were mixture of younger drivers and elderly drivers. In total there were 23 subjects involved in the study and two different visits used (datasets referred to as AUTO21 phase 2, visits 2 and 7)[36]. One of the datasets was obtained from a study regarding the effects of alcohol consumption on driving (DUI). In that particular set a baseline test was performed where no alcohol was consumed, and three other tests that recorded the effects of drinking one, two and three standard bottled servings of beer. The final set used was one where the simulator was used to retrain a person who had suffered a traumatic brain injury (TBI). In this particular case there were 12 runs recorded. Once the data was extracted according to each of the segments, it was analyzed for any anomalies such as accidents or flags. The purpose of the flag is for the people administering the test to note observed behaviour that was deemed to be unsatisfactory. Many different actions constitute unsatisfactory behaviour; however, anything that is either illegal or increases the risk of a collision is deemed unsatisfactory. Segments where such an event was present were excluded from the dataset as they do not represent the desired behaviour for the driver model. They were, however, set aside for future use. Also of note is that some of the segments were completely excluded as a result of "sim sickness" where the subject may have become nauseous. This a known issue with driving simulators and one of their shortcomings. Measures to mitigate this effect such as venting additional air directly on the subject are used; however, they are not always successful.As a result of sim sickness, the subject was unable to complete the run and no data could be acquired.

4.3 Data Processing

Once the data was acquired from the simulator extensive processing was required to prepare it to train the designed neural network. As discussed previously, there were many data channels recorded from the simulator; however, only a select few were required to construct the model. As also previously noted, segmentation of the data was required such that what was occurring in the road scenario was known and that it could be associated with the appropriate subject or test run. As mentioned, there were four datasets that were used in training the neural network. In terms of the scenario, it was the same one used for the the two visits studying elderly drivers and for the study that focused on the retraining following a traumatic brain injury. A comprehensive breakdown of the driving scenarios is given in Table 4.1.

In order to obtain the data in a useful form, a script was written in MATLAB® that would extract the desired variables for each segment. The manner in which the script functions is that the distance corresponding to the beginning of the segment is presented; the script then searches through the distance variable until it finds the first value equal to or greater than the presented value. It then identifies the "handle" of that variable and uses it to find the appropriate value for each of the variables that are to be extracted (The handle is an identifier of a variable assigned by MATLAB®, it is associated with the index of a matrix)[40]. This same process is repeated to find the end of the segment; once these two values are found they are used to set up a loop that extracts all of the desired data in between these two points. Once all of the data was extracted, it was then concatenated into one data file containing in excess of 380,000 data points. Following the concatenation, the data was then normalized between -1 and 1; this is common practice when conditioning data for use in neural networks; the process is illustrated in Figure 4.2 .

The principle reason for data scaling is to ensure that all of the variables have the same relative size in order to prevent some of the network weights from becoming too large. Variables with larger magnitudes will have a tendency of having larger weights associated with them. This in turn could cause their effect to dominate the effect of the variables with smaller magnitudes, and give an inaccurate representation of the relative influence of the variables. Scaling also helps give a more accurate representation of error for similar reasons[29, 41]. There are a few important points to note regarding the data scaling. The first is that all of the input variables were scaled with respect to their respective maximum value within the training set. The one exception is the speed variable as it was normalized with respect to the largest speed limit since the two variables are coupled together and the highest speed recorded did not exceed the highest speed limit within the dataset. Had the highest speed exceeded the highest speed limit, it would have been used. In the case of the output variables the data was normalized with respect to the global maximum of all the acquired data. This was done because typically the throttle signal is simply a value between 0 and 1. However, it was not available

Table 4.1: Breakdown of data properties used to train the neural network driver model

Dataset	Segment Type	Length	Road Curvature	Number of Subjects or Runs	Number of Segments per Subject or Run
Auto21 Phase 2 Visit 2	Straight Road	500m	Not Applicable	23	4
Auto21 Phase 2 Visit 7	Straight Road	500m	Not Applicable	23	4
Traumatic Brain Injury	Straight Road	500m	Not Applicable	12	4
Auto21 Phase 2 Visit 2	Curve Right	300m	250m	23	3
Auto21 Phase 2 Visit 7	Curve Right	300m	250m	23	3
DUI	Curve Right	234m	500m	13	12
Traumatic Brain Injury	Curve Right	300m	250m	12	3
Auto21 Phase 2 Visit 2	Curve Right	350m	250m	23	1
Auto21 Phase 2 Visit 7	Curve Right	350m	250m	23	1
DUI	Curve Right	290m	666.67m	13	1
Traumatic Brain Injury	Curve Right	350m	250m	23	1
Auto21 Phase 2 Visit 2	Curve Left	300m	250m	23	2
Auto21 Phase 2 Visit 7	Curve Left	300m	250m	23	2
DUI	Curve Left	234m	500m	13	12
Traumatic Brain Injury	Curve Left	300m	250m	23	2

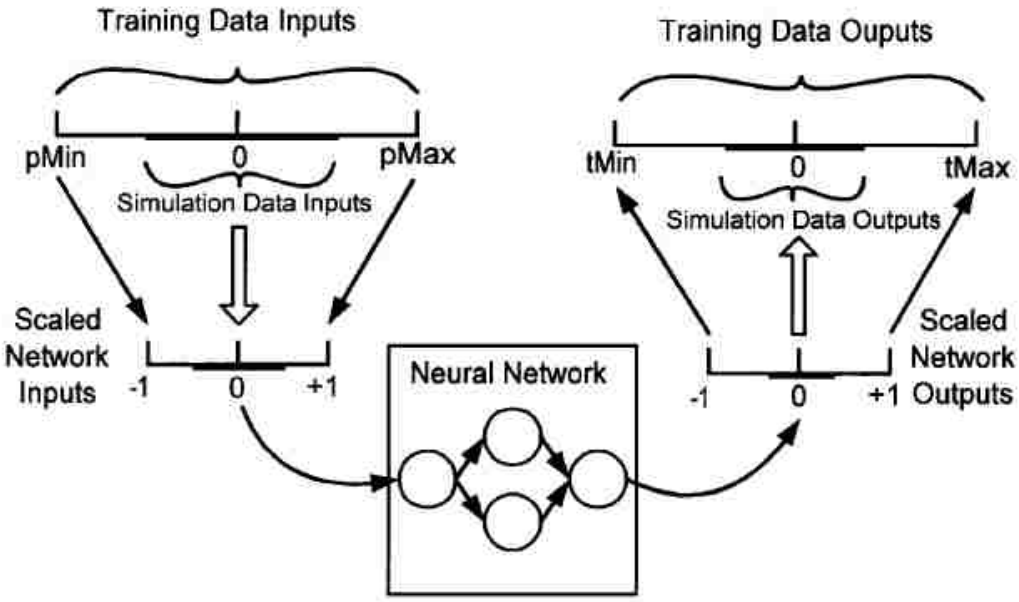


Figure 4.2: Illustration of how data is scaled down for input and often how the output will be scaled up. For the purposes of this research scaling up is not necessary unless using the model in conjunction with a vehicle simulation program., [41]

in this form because of the simplified vehicle model; therefore, only the raw signal could be used. In order to establish full throttle, the largest value that appears had to be used. The same holds true for the steering angle, as the data was normalized against the largest steering angle that appeared. In the case of the braking data, the deceleration due to braking was used. In this form the data can be manipulated in many ways; in this particular case to determine the brake master cylinder pressure. This was done in the event that the model is ever to be used in a vehicle simulation program.

4.4 Data for the Classification Network

Following the development of the driver model neural network, it was then possible to obtain data that could be used to evaluate the behaviour of the drivers. The procedure used was to take each subject, in this case both those that exhibited desirable behaviour and those that exhibited undesirable behaviour and to run simulations with their input variables through the neural network. Since this particular network had already been trained using a large dataset of acceptable driving behaviour, the output from the neural network should represent the desired driving behaviour under the given conditions. The deviation in the actual output dataset from the behaviour calculated from the neural network constitutes the error which is then divided into three categories for each variable. As a result of this there are 27 potential combinations that can arise as there are three output variables and three classes used for each variable. In order to construct the classification network, self-organizing maps (SOMs) and clustering networks were considered. Further research discovered that the appropriate network to use in this role was a Learning Vector Quantization Network (LVQ). Discussion regarding the construction of LVQ networks will be given in the following chapter. In brief, LVQ networks have target classes assigned to their input vectors and use such information to organize an SOM to classify them.

Chapter 5

Neural Network Design

When using neural networks for analysis, the first step in their design is to select the appropriate architecture for the given problem. As discussed in Chapter 3, there are several different types of neural networks available with the most important distinction being between networks used for function approximation and those used for grouping data. The following chapter discusses the design steps taken to design the neural networks used in this research and the reasoning behind the choices that were made.

5.1 Driver Model Neural Network

Initially to develop the driver model network, a simple multi-layer perceptron network was selected. There are two reasons behind the choice of such a network as an initial starting point. The first reason is that the purpose of the network is to approximate the behaviour of an ideal driver. It is not attempting to classify data; therefore, a network suited for function approximation, was used, that being the MLP. The second reason is that the MLP is the simplest architecture available for function approximation; therefore, it was deemed appropriate that it be used as a starting point and if the performance proved to be favourable no further experimentation with network architectures would be required. To evaluate the performance of the MLP in a driver modeling role, it was initially trained using only a portion of all of the available training data that was discussed in Chapter 4. This was done because a decision of how to process the entire training dataset had not yet been made. Only a portion of the dataset was also used in order establish an evaluation in a relatively short time frame, given that it was known that the MLP may eventually not be used as the final driver model.

Once the network was trained, its performance was analyzed. Upon initial examination of the output variables, it was apparent that the network was able to approximate the behaviour of the

steering and throttle inputs that the driver would be using. This, however, was not the case with the braking data. Examination of the braking data plots revealed that the error was quite large, meaning that the network was not able to accurately represent that particular behaviour. This effect also manifested itself in the linear regression plots, while the overall correlation factors were very high it was obvious that there were some outliers specifically associated with the braking data. Because a large majority of the network outputs correlated well with their target values, the effect of these outliers on the global correlation factor was limited. Nevertheless, it was still important to determine a cause and solution to this problem as the individual correlation factor for the braking behaviour was much lower than those for the steering and throttle behaviour.

The first attempt to correct this problem was to develop a NARX network (see Section 3.1.3), once again using only a portion of the dataset. It was believed that as a result of the use of time delays and feedback loops that it might help diminish the error in the braking behaviour. Once the network was generated and trained, the results were once again analyzed; however, the improvement was limited. Another factor that was considered was that the local maximum was used as the scaling factor, which caused the network to measure the relative error to that portion of the data rather than the absolute error error of the entire dataset available. This had a large effect given that the discrepancy was normalized over a smaller value and thus magnified giving rise to a larger error. A global scaling factor was then applied to the complete dataset. Both an MLP and a NARX network were trained using the rescaled dataset. The results were much more favourable as the level of error was significantly reduced which gave more validity to the method being used.

Another attempt to improve the driver model involved the creation of a radial basis function network(RBFN). An attempt to use this type of network was made as they are supposedly well suited for driver modeling as discussed in Chapter 2 and in [24]. Unfortunately, attempts to construct such a network were unsuccessful as initial attempts failed to train a simplified RBFN or simulate in the case of an exact RBFN(see Section 3.1.3 for details on the differences between the two types of RBFNs). The attempts to develop a driver model using an RBFN failed as result of the extensive computing resources required; the memory required to train the network exceeded the 12 GB that was available. There is a possibility that with the adjustment of some training parameters that an RBFN could be used; however, time constraints prevented such an investigation from taking place. While an evaluation of the performance of an RBFN would have been insightful, the fact of the matter is that the NARX network provided a model that was adequate for the purpose of the research at this time.

5.2 Neural Network Sizing

Following the selection of a neural network architecture, the next step is to size the network in terms of hidden layers and the number of neurons in the hidden layer. At this stage neural network sizing is a very open-ended question with very few rules or guidelines; therefore, much debate exists and reserach is on-going in order to establish methods that optimize neural network sizing. Typically the current practice is to overestimate the number of neurons and evaluate the performance. The process becomes iterative by reducing the number of neurons (assuming the initial network was an overestimate) until the performance of the network degrades to a point deemed to be unsatisfactory. The importance of obtaining an optimally sized network is to find a balance between accuracy and efficiency, while a larger network may be capable of more accurately modeling the behaviour of the system of interest, the benefits may not be so great that it warrants being used over a smaller network. The use of a smaller network will have the effect of reducing both the time and computing resources needed to train and simulate the neural network. Another concern regarding neural network sizing is overtraining. It is possible that a neural network may in fact capture the behaviour too well rather than make a generalization. This can be an issue particularly with noisy datasets, as an overtrained network will begin the follow the noise rather than make a generalization "through the noise"; the end results is that because the network models the noise, it is not modeling the true response of the system (Figure 5.1).

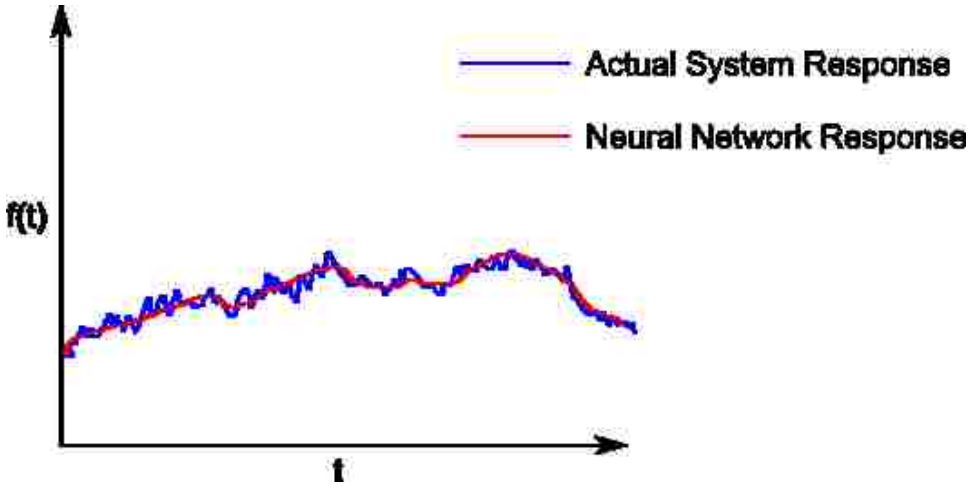


Figure 5.1: Plot of the Desired Neural Network Generalization with Noisy Data

One method to establish a starting point is to use Kolmogorov's Theorem[42], which is not uniquely used in neural networks as it is used in other computational methods. Kolmogorov's

Theorem states that any function of n variables may be represented by the superposition of a set of $2n$+1 univariate functions. In this case there are 7 input variables, therefore 15 hidden layer neurons is an appropriate starting point according to Kolmogorov's Theorem. In all likelihood a network of this size is larger than necessary. A preliminary analysis done for another project using a smaller portion of the dataset revealed that a network with as few as 8 neurons could adequately model the system. Time constraints prevented training a smaller network with the complete dataset.

While there are no established rules for sizing neural networks, methods for reducing their size following training do exist. These methods are referred to as pruning methods. Samarsinghe[29] gives an extensive discussion of two methods, the first method is called Optimal Brain Damage (OBD), and the other is based on an analysis of the variance of sensitivity to the error. Optimal Brain Damage uses a parameter known as *saliency*, which defines how important a weight is and the method itself calculates the so-called cost of setting that particular weight to zero. The saliency is computed through the Hessian matrix Equation 5.1[29] (recall Section 3.1.2).

$$H_{ij} = \frac{\partial^2 E}{\partial w_i \partial w_j} \tag{5.1}$$

In order to reduce the computational cost, a local approximation is typically used where only the diagonal terms of the Hessian matrix are considered. H_{ii} represents the acceleration of the error with respect to a weight; when it is multiplied by w_i^2 (Eqn.5.2)[29] an indication of the total effect of w_i on the error is given.

$$s_i = \frac{H_{ii} w_i^2}{2} \tag{5.2}$$

Effectively, a larger value of s_i indicates that the weight w_i has a larger influence on the error. The entire procedure for pruning using this method involves first calculating the saliency and then removing pathways associated with any weights with small saliencies and removing entire neurons. Once those weights have been removed, the simplified network is then retrained and should perform just as well as the initial network.

The second pruning method discussed in [29] uses the variance of the network sensitivity of a given parameter. In this case sensitivity is described as being the derivative or gradient of a given parameter; the parameter can be either an input, a hidden neuron activation, or a weight according to [29]. For conciseness a lengthy description of the method will be omitted; however, to give a brief overview of the method, a series of statistical properties are calculated and two hypothesis tests are performed. The results of the hypothesis tests give an indication of the relevance of the parameter. If the parameter is relevant, then its removal is not advisable; however, if it is deemed to be irrelevant, it should be removed from the network. In other work referenced in [29], it is

stated that the variance nullity measure is superior to OBD. This stems from the fact that OBD is magnitude based and typically attempts to remove small weights. In some cases the network may still be sensitive to small weights and their removal is not warranted as a result. Also of note is the fact that data scaling prevents the network from calculating large weights in the first place.

5.3 Neural Network Training

Training the neural network is the next step once the architecture has been chosen and it has been sized; this entails selecting a training algorithm. The default training algorithm in MATLAB® is the Levenberg-Marquardt (L-M) method as it is generally regarded as being the fastest and most accurate as discussed in Chapter 3. While the L-M training algorithm is regarded as being the most effective in terms of speed and accuracy, the principle disadvantage is the amount of resources required to train with it. The L-M training algorithm is very memory intensive, so much so that when using a very large dataset, the 12 GB of memory was inadequate, such that the network was unable to complete even one training iteration. As a result, an alternative training algorithm that is less resource intensive was sought. Eventually it was discovered that the L-M method could be used as MATLAB® allows for certain parameters to be altered to reduce the amount of memory required at the expense of increased training time. Another factor that allowed for the use of L-M training algorithm was switching to an open-loop NARX network rather than one of the closed-loop variety, the differences of which were discussed in section 3.1.3. In comparison with the other training algorithms, the L-M method produced the most accurate network as was expected to be the case.

5.4 Final Driver Model Neural Network Architecture

To recap the previous sections of this chapter, the final decision on the overall design of the neural network to act as the driver model is to use an open-loop NARX network with 15 hidden layer neurons, each using the tansig activation function and three linear output neurons (one for each of the output variables. The network also incoporated two time delays on the inputs and the output variables trained using the Levenberg-Marquardt training algorithm (Figure 5.2). This network produced favourable results, more details of which will be discussed in the following chapter.

5.5 Classification Network Design

As briefly mentioned in Chapter 4 the driver behaviour assessment tool requires the use of two types of neural networks. Until now the discussion in this chapter has focused solely on the network used

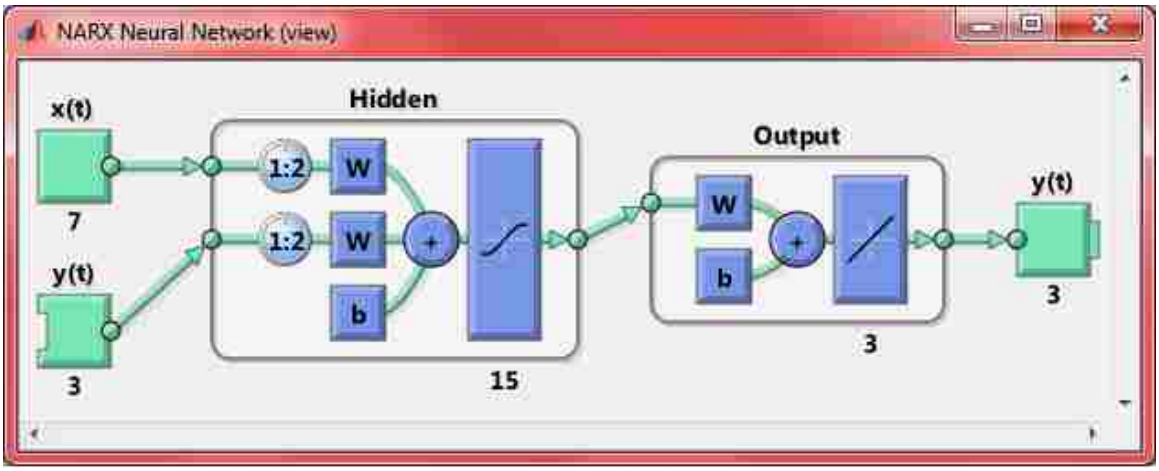


Figure 5.2: Final Architecture for the Driver Model Neural Network

to construct the driver model. The information in this section will be focused on the network used to classify each driver based on their behaviour in comparison to that of the driver models intended "typical" behaviour. In order to develop this portion of the assessment tool several types of networks that are best suited for classification were investigated. The networks for function approximation discussed in the previous sections are generally not well suited for classification.

A problem with using the standard clustering networks and self organizing maps (SOM) in this application is that many classes may exist throughout the dataset, necessitating a large quantity of neurons, while the ultimate goal of the assessment tool and in turn the classification network is to classify the drivers based on three different levels of ability. This justifies the use of a learning vector quantization network, as it makes use of an SOM to associate neurons with their appropriate location in the input space, while incorporating a layer of linear neurons with which the neurons in the SOM associate themselves. The linear neurons are each associated with one of the particular classes of behaviour prior to training. The neurons in the SOM act as subclasses that eventually, become associated with one of the linear layer neurons.

Construction of an LVQ network requires prior knowledge of what, in this case, constitutes good, marginal and poor driving behaviour. Unfortunately such parameters have yet to be established and are beyond the scope of this research; therefore, an estimation to divide the data was used in order to establish whether or not an LVQ network can be used in this role.

The training data used for the LVQ network was the absolute error of the normalized network outputs with respect to the normalized target outputs while the parameters to divide the data were based on the percentage error. As already discussed on several occasions there are three output

variables from the driver model and three possibilities in terms of classification of behaviour for each one, thus there are 27 degrees of freedom or what can be viewed as 27 sub-classes (see Figure 5.3 for a visual representation). As a result the same number of neurons are used in the SOM, indicating that theoretically there should be one neuron associated with each potential sub-class and as already mentioned there are three potential outcomes and thus three neurons in the output layer.

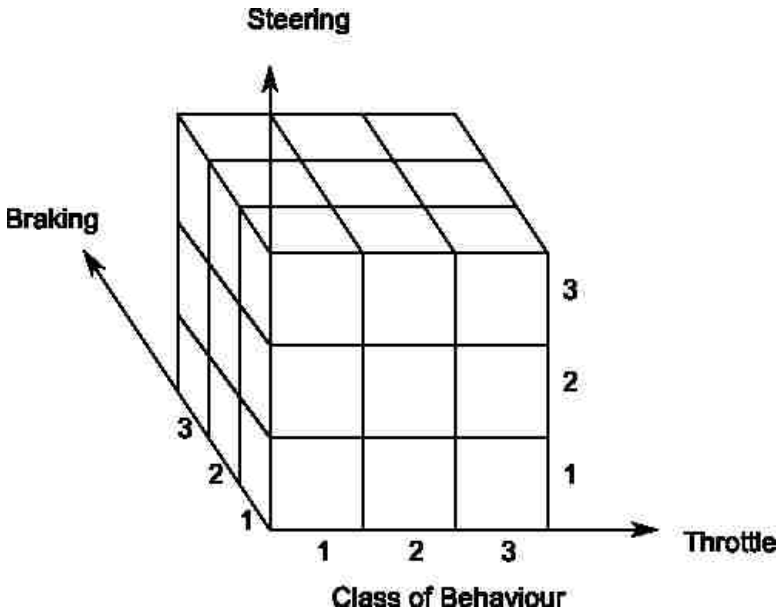


Figure 5.3: Visual Representation of the Input Space to the LVQ Network

Chapter 6

Results & Discussion

The following chapter discusses the results of the different methods that were tested in order to develop the driver assessment tool as discussed in the preceding chapter.

6.1 Driver Model Network

Performance of the driver model networks were analyzed based on the observations made from several plots which have been included in this section. The plots include the training window generated by MATLAB® during training. This window provides information such as the amount of time and number of iterations needed for training and the criteria that was met to halt training. The plots of the neural network output are presented alongside the target output in order to view a direct comparison the network's performance along with a zoomed in portion of the network output overlayed on the target output. It should be noted that these plots display the normalized target and neural network outputs versus each input; also note that the overlays and error plots are based on the same information. Overlaying the entire plots would be ideal however arranging the plots in such a manner renders it difficult to draw conclusions from them. As such, plotting them separately along with a zoomed in portion of the overlay was viewed as an adequate compromise. The plots of the error for each output is also included to provide further insight into the performance of the network. The linear regression of the network output versus the target output also provides a valuable indicator of neural network performance. In the following section the linear regressions for each output variable and for the entire network are given. A network that performs well will place most of the data in the region of a plot that follows the function y=x. The error histogram places the error for each data point in its appropriate bin based on its magnitude; in this plot it is desirable for most of the data points to be placed in bins at or near zero. Finally the training performance

plot is also presented, the purpose of this plot is to present the mean squared error of the network for the training set, the test set and the validation set at each iteration. This is significant because one of the stopping criteria is when the MSE of the training set is greater than that of the validation set for a pre-specified number of consecutive iterations. The aforementioned plots are presented for each of the networks discussed in Chapter5 that were used to develop the driver model portion of the assessment tool.

6.1.1 Multi-Layer Perceptron

As discussed in Ch.5, the first network architecture to be tested was the MLP due to its simplicity. Based on the information in the following figures it can be concluded that the network performed fairly well. Figure 6.1 (p.57) indicates that network training stopped after 37 iterations once the minimum error gradient was exceeded. According to the figure, training took almost 59 minutes to complete and a mean square error of 0.000201 was achieved.

According to Figures 6.2a & 6.2b (p.58), the MLP network was successful in establishing a relatively accurate generalization of the steering behaviour. Figures 6.2c & 6.2d indicate that a relatively accurate generalization of the throttle behaviour was made, however, not to the same degree as witnessed in the case of the steering behaviour. Figures 6.2e & 6.2f indicate a similar result for braking as was witnessed for the steering behaviour. The zoomed in portion of the overlay plots (Figure 6.3, p.59) help to further illustrate this behaviour. It is apparent that the network output follows the target output very closely for both the steering and braking behaviour (Figure 6.3a & 6.3c, respectively); however, a larger variation in the performance of the network relative to the target output is observed for the throttle behaviour (Figure 6.3b).

The error plots (Figure 6.4, p.60) further serve to enhance the observations made from the performance plots. Figures 6.4a and 6.4c illustrate the errors for the steering & braking behaviours respectively. With the exception of a few spikes, there are very few errors of significant magnitude in both plots. Figure 6.4b illustrates a far greater number of errors in the throttle behaviour in comparison with the plots for the steering and braking behaviours.

The linear regression plots in Figure 6.5 (p.61) help to reinforce the observations made from figure 6.2 as the correlation of the network output variables appears to be relatively strong with the target variables. The correlation factors for the steering, throttle and braking performance were 0.99386, 0.95787 and 0.95221, respectively, with an overall correlation factor of 0.99823. The steering regression plot illustrates that most of the data is concentrated near the line of y=x with only a small number of outliers. The throttle regression plot illustrates that most of the data lies in a larger area that follows the regression line, while the braking regression plot displays a larger number of outliers which is indicative of less accurate performance.

Figure 6.6 (p.62) is the error histogram for the MLP network. From the error histogram it can be observed that the errors are mostly clustered around zero which helps to confirm that the network did in fact perform well.

Figure 6.7 (p.63) displays the training performance of the neural network. It can be observed from this plot that the three sets of data used for training, validation and testing all seem to follow a similar level of performance at each iteration. However, the performance of the validation data never exceeded the performance of the training data often enough to stop the network from training as this was not the factor that stopped training. One of the potential criteria that the neural network toolbox uses to determine if training should be stopped is if the performance of the validation set exceeds that of the training set for a pre-specified number of iterations.

The overall assessment of the performance of the MLP network based on the information displayed in the plots is that it demonstrated the ability to accurately model the target driving behaviour. This conclusion is especially true in the case of the steering & braking behaviour, while the accuracy of the throttle behaviour leaves ample possibility for improvement.

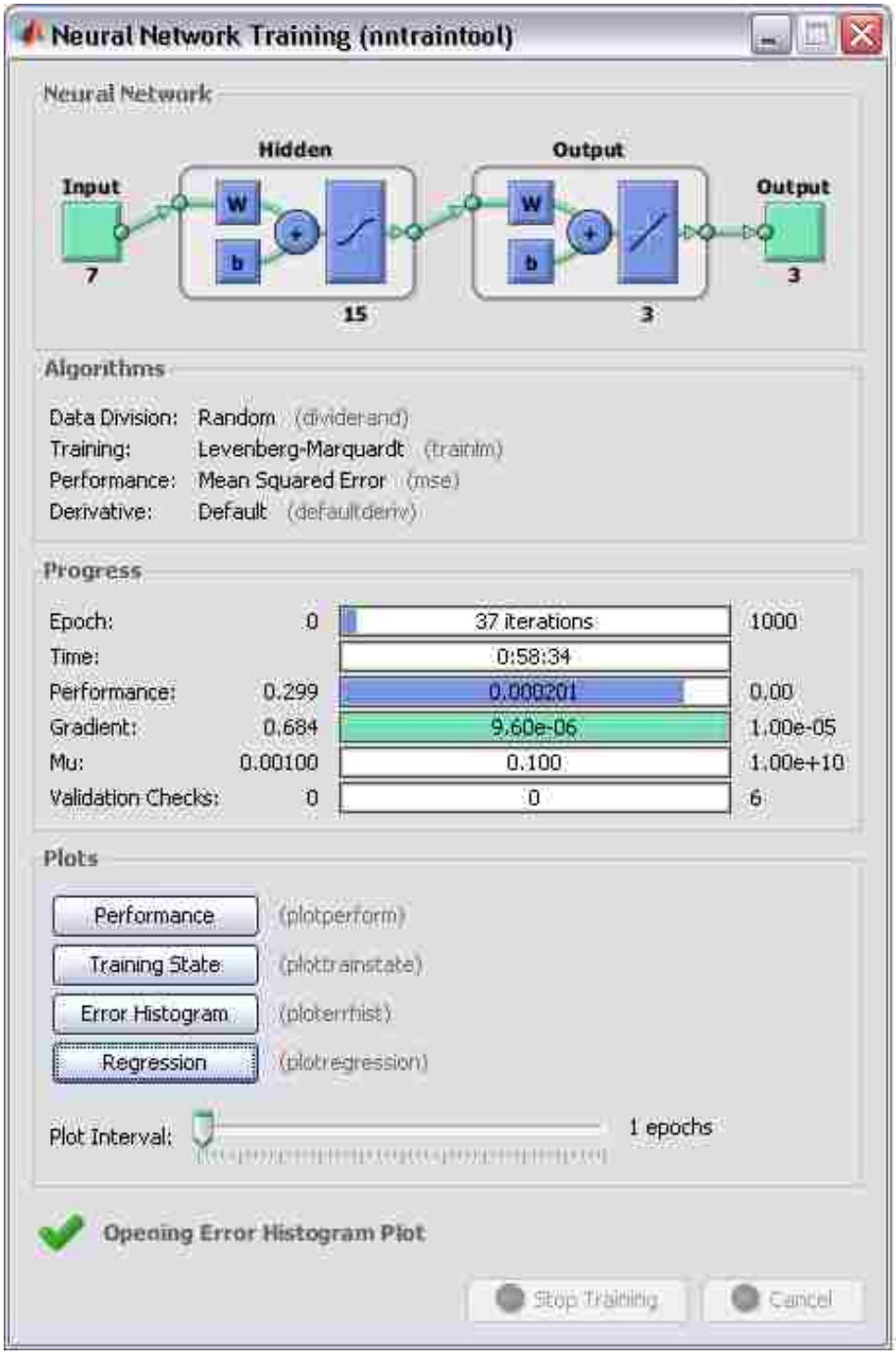
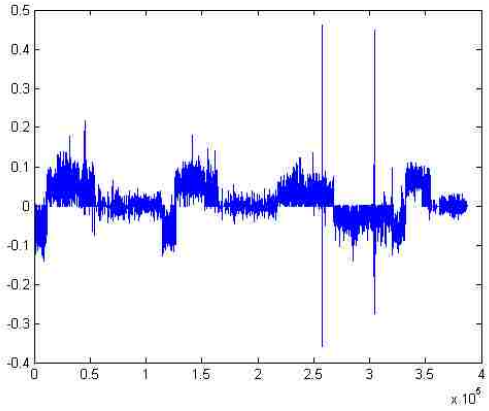
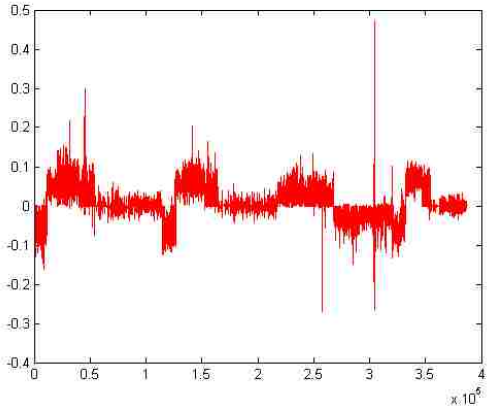


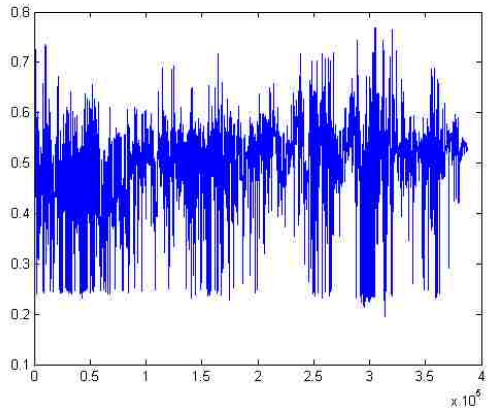
Figure 6.1: Training Window for MLP network



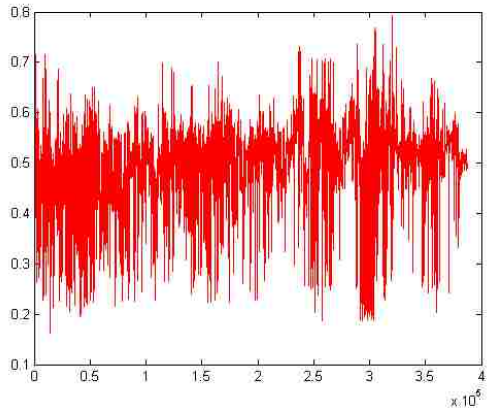
(a) Steering Target Output



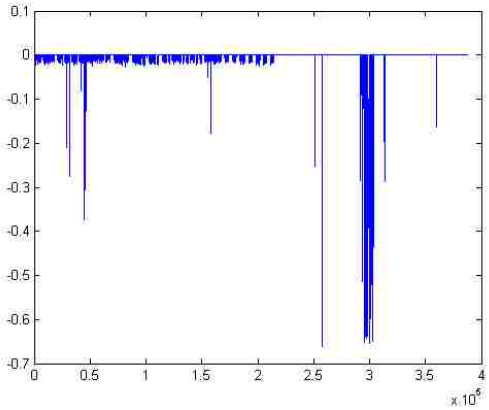
(b) Steering Network Output



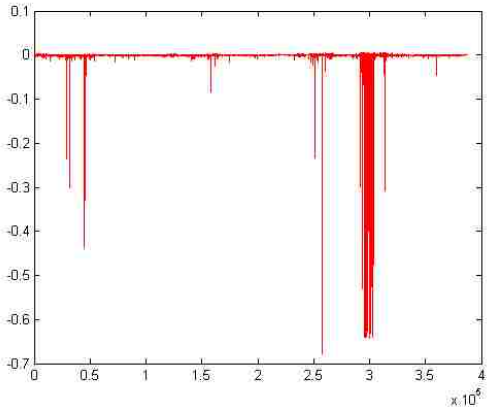
(c) Throttle Target Output



(d) Throttle Network Output

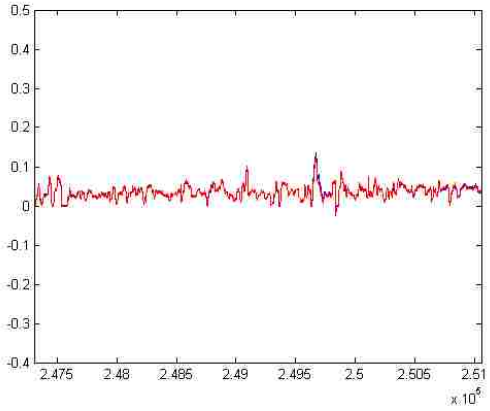


(e) Braking Target Output

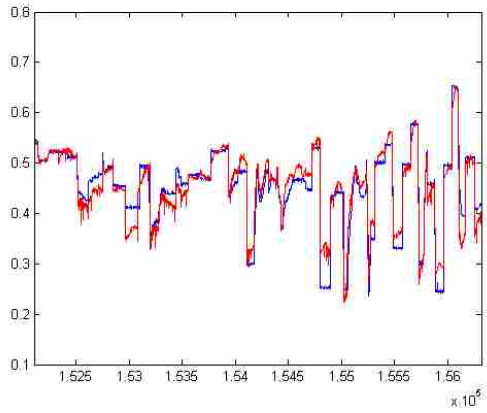


(f) Braking Network Output

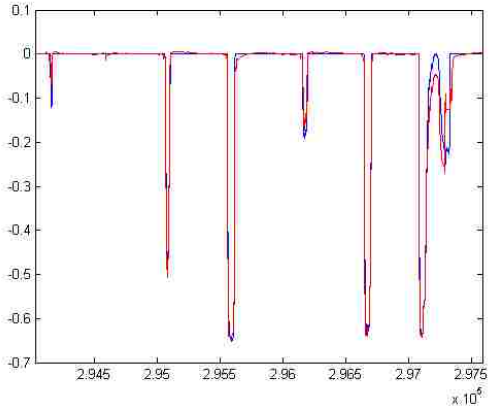
Figure 6.2: Comparison of the performance of the MLP network alongside the target output



(a) Steering Overlay

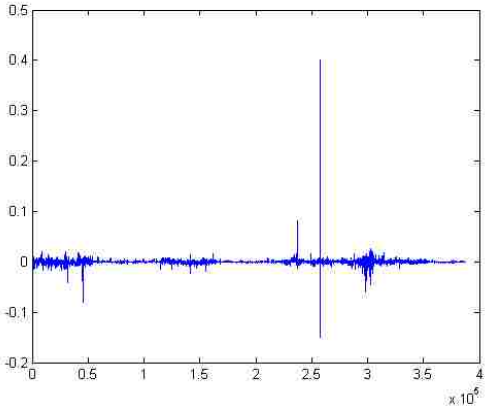


(b) Throttle Overlay

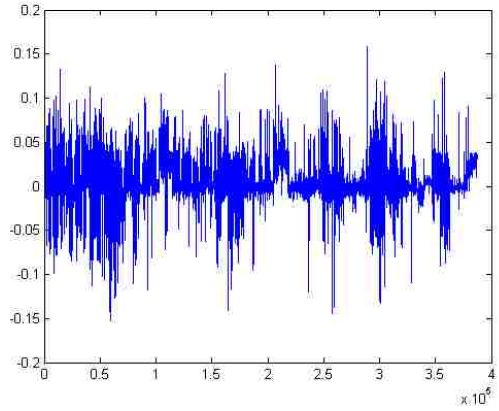


(c) Braking Overlay

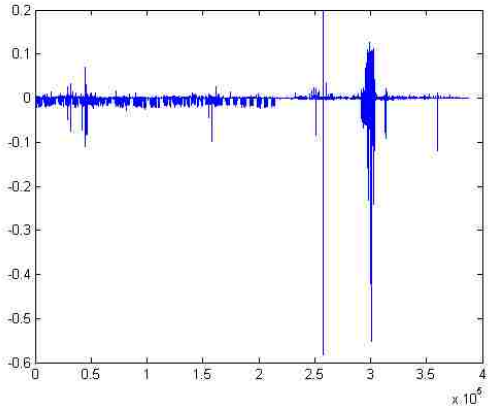
Figure 6.3: Magnified Overlays of Target Output(Blue) and Network Output(Red)



(a) Steering Error

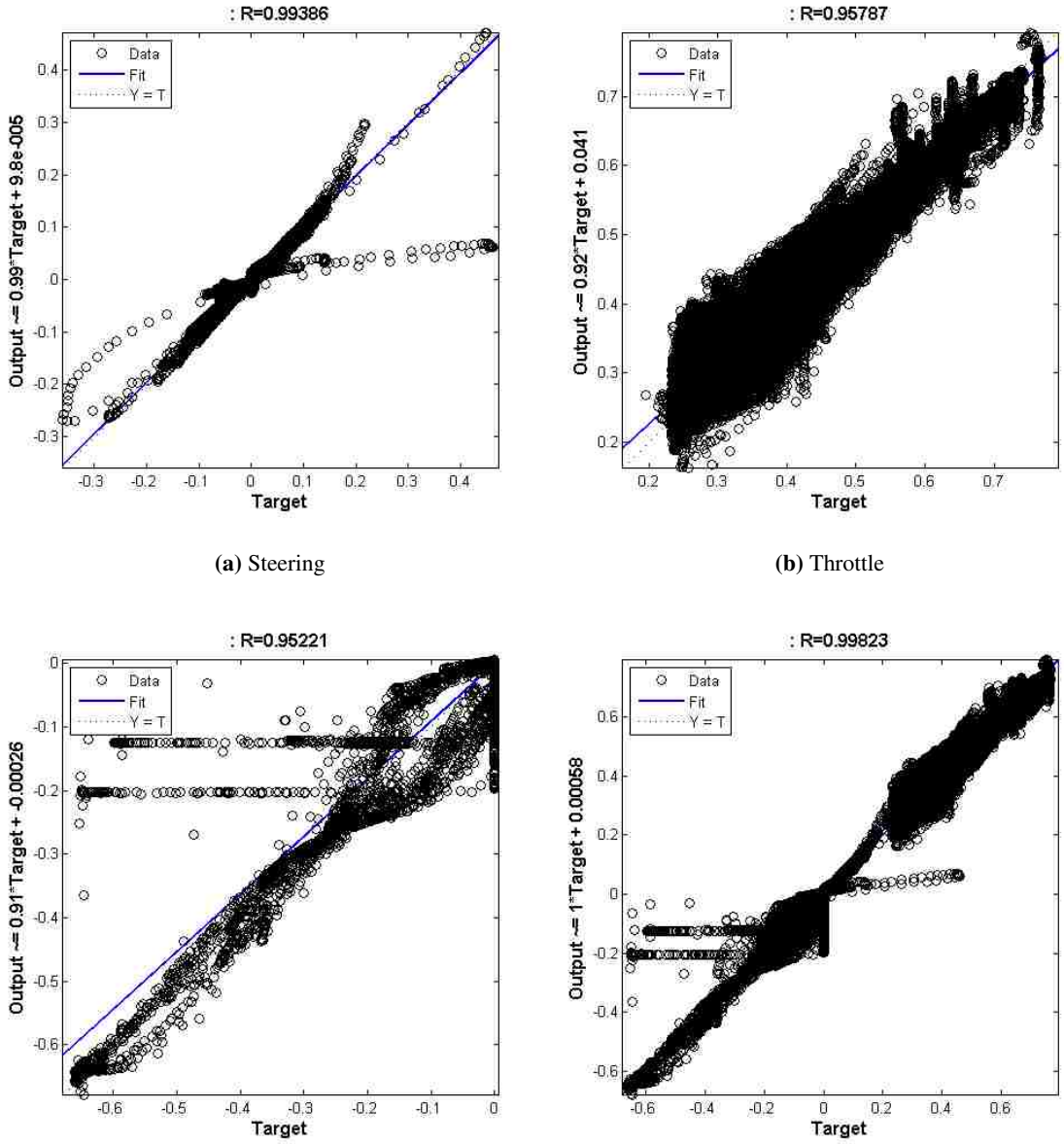


(b) Throttle Error



(c) Braking Error

Figure 6.4: Error Plots for Each Network Output



(a) Steering

(b) Throttle

(c) Braking

(d) Overall Regression

Figure 6.5: Regression plot for each individual variable and overall regression using an MLP neural network.

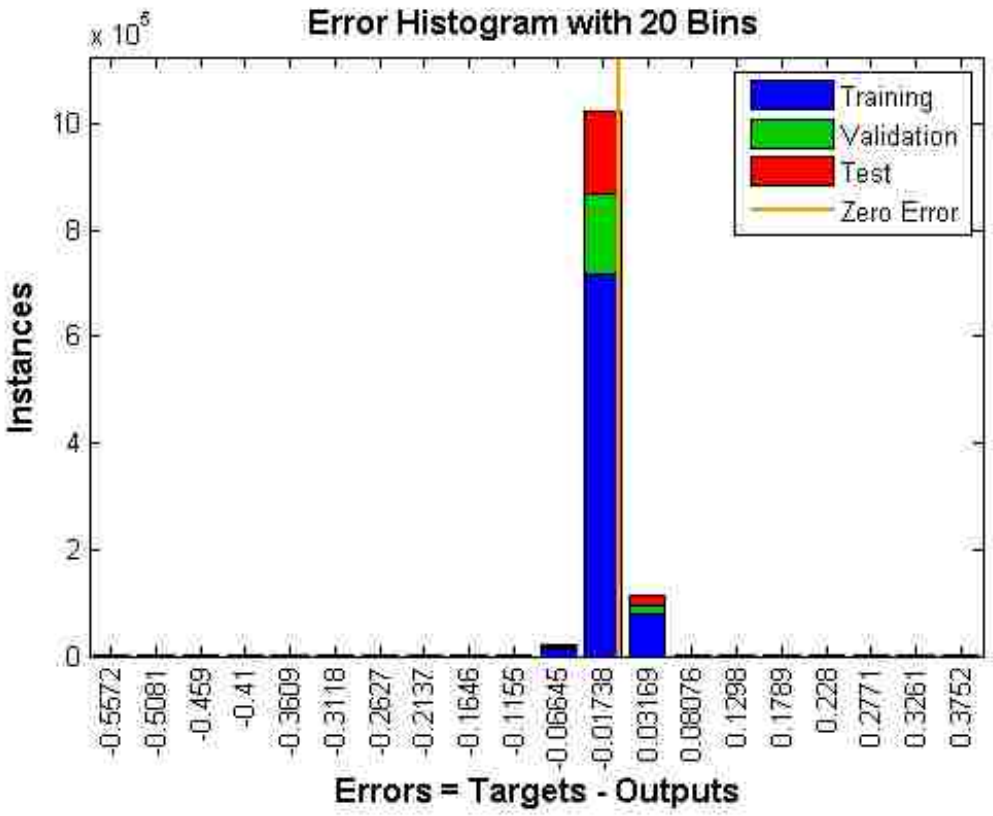


Figure 6.6: Error Histogram for MLP network

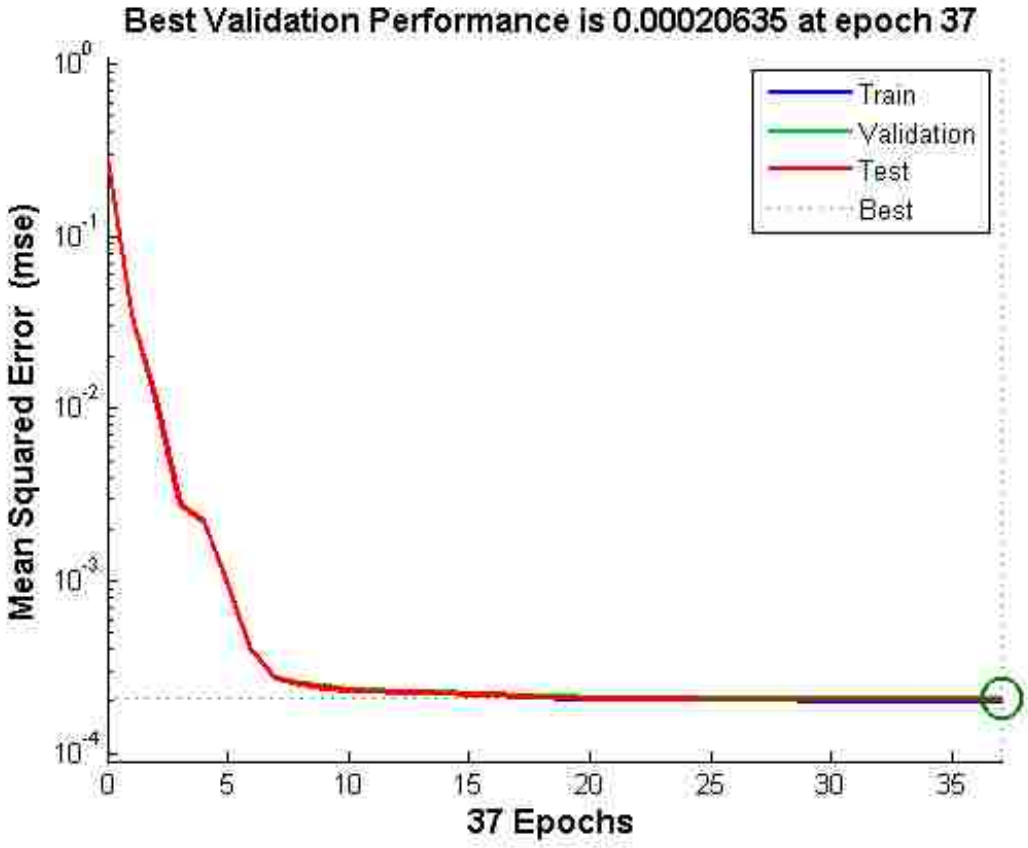


Figure 6.7: Training performance for MLP network

6.1.2 Nonlinear Autoregressive Network with Exogenous Inputs

While the MLP neural network was capable of accurately modeling the driving behaviour, an even greater level of accuracy was sought. As a result a NARX network was designed and trained. A preliminary network was trained with a smaller dataset of approximately 10,300 data points as discussed in Section 5.1, and it performed satisfactorily. In this section the results of both the preliminary network and the final network using the full dataset are presented. The main difference in the network structures is that the preliminary network used a closed loop while, the final network used an open loop in order to facilitate training. The difference between the two network structures is discussed in Section 3.1.3.

Preliminary NARX network results

The training window (Figure 6.8, p.65) indicates that training was stopped after 220 iterations based on having failed 6 consecutive validation checks. Training lasted almost 2 hours and 46 minutes and mean squared error of 0.00303 was attained.

Figures 6.9a & 6.9b (p.66) indicate that the network was able to make a relatively accurate model of the steering behaviour while Figures 6.9c & 6.9d illustrate a similar result for the throttle behaviour. Figures 6.9e & 6.9f indicate that this was not the case for the braking performance as the network had difficulty accurately learning the braking behaviour. As previously discussed in Section 5.1, much of this poor performance is attributable to the scaling factor that was used when normalizing the data. The overlay plots (Figure 6.10, p.67) help to further illustrate the accuracy of the network. Figures 6.10a and 6.10b display how both the steering & throttle performance follow the target performance closely, with the steering performance almost matching the exact target performance while the braking peformance (Figure 6.10b) leaves much to be desired.

The error plots (Figure 6.11, p.68) serve to further display the same conclusions drawn from the performance plots. Figures 6.11a & 6.11b display errors for the steering & throttle performance, respectively, are mostly of a relatively small magnitude. As expected, the same cannot be said for the braking performance (Figure 6.11c) as the errors are mostly of a significantly larger magnitude.

Figure 6.12 (p.69) illustrates the linear regression for each output variable as well as the overall linear regression. The aforementioned plots help reinforce the deductions made from the output plots as the linear regressions for both the steering and throttle behaviour are concentrated around a line representing y=x while most of the braking outputs lie very far from this region. The steering, throttle and braking data had correlation factors of 0.9997, 0.99047 and 0.17635 respectively, while the overall correlation was 0.97386.

The error histogram (Figure 6.13, p.70) illustrates that, for the most part, the errors are concentrated around zero, however there are some errors in the outlying bins which is consistent with what was observed in the performance and regression plots. The deduction that was made from the error histogram is that the NARX network offered some promise of accurately modeling driver behaviour but further investigation was required.

The training performance plot (Figure 6.14, p.71) indicates that the performance of the validation set was better than that of the training set which is consistent with the criteria used to stop training.

The overall performance of the preliminary network served as a starting point to illustrate that, at least to some degree, neural networks, particularly NARX networks, are suitable for driver modeling. This conclusion was drawn based on the performance for the steering and throttle behaviour. While the braking performance was poor, it was acknowledged that much of the poor performance

was attributable to the scaling factor used for the data. As a result, further investigation into the abilities of the NARX network using a global scaling factor and the complete dataset was warranted.

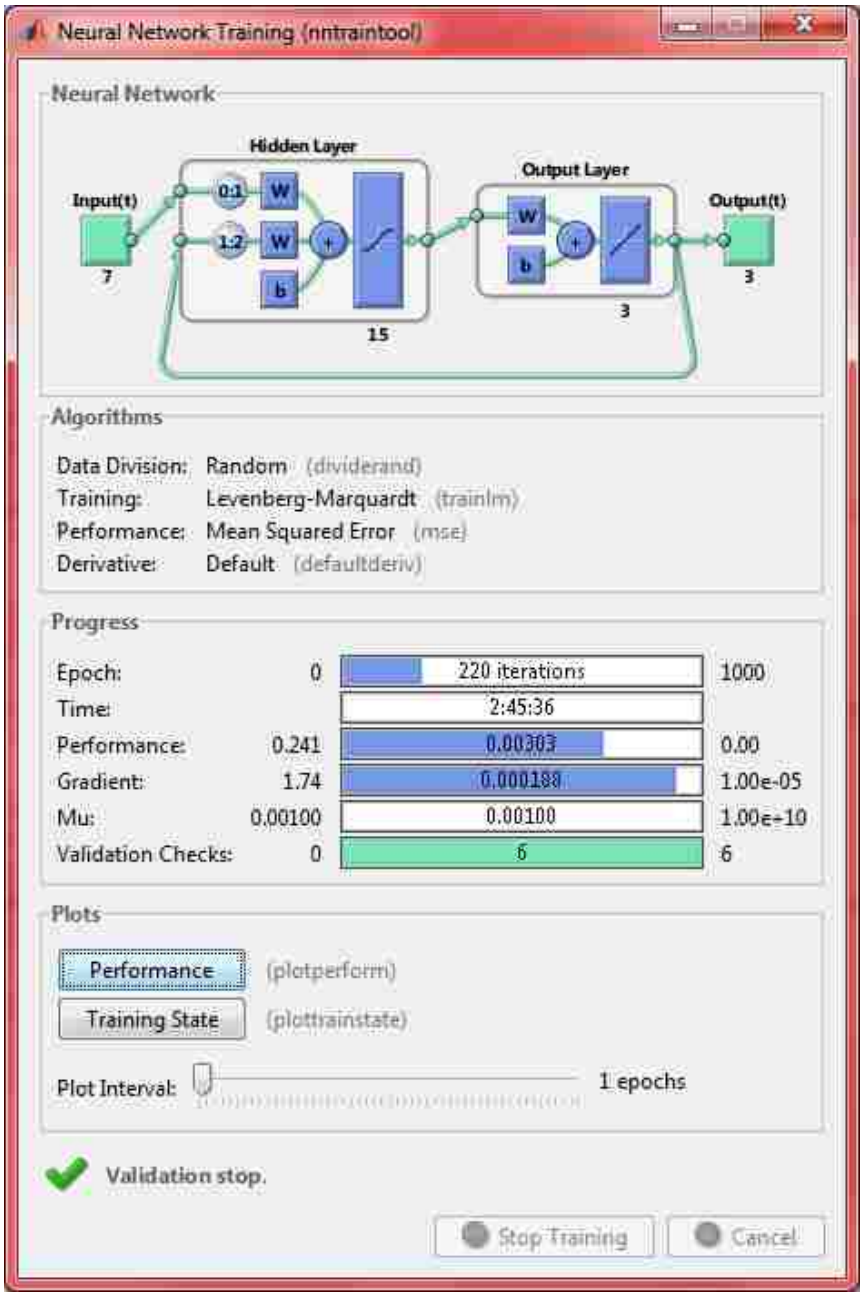
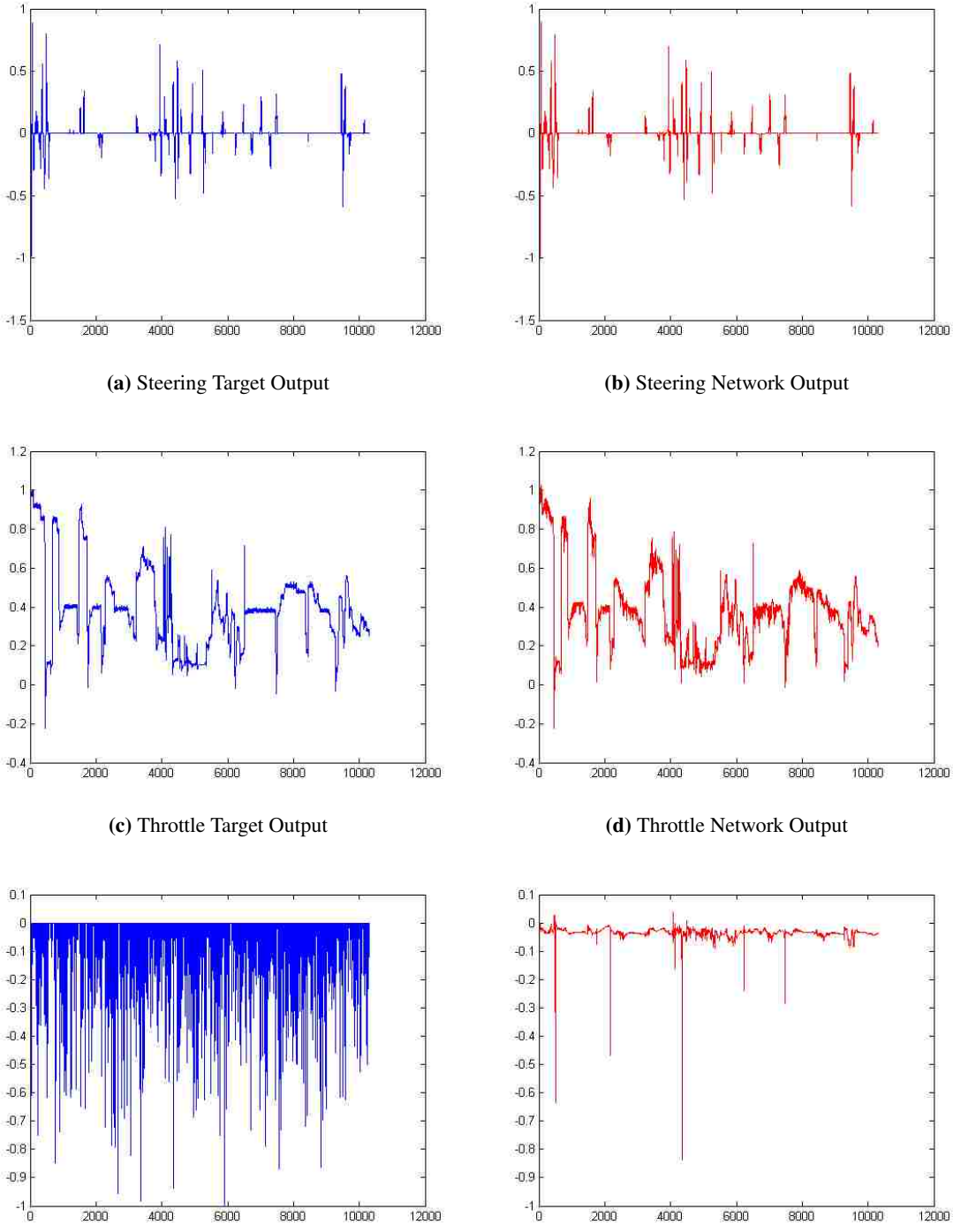


Figure 6.8: Training Window for Preliminary NARX network



(a) Steering Target Output

(b) Steering Network Output

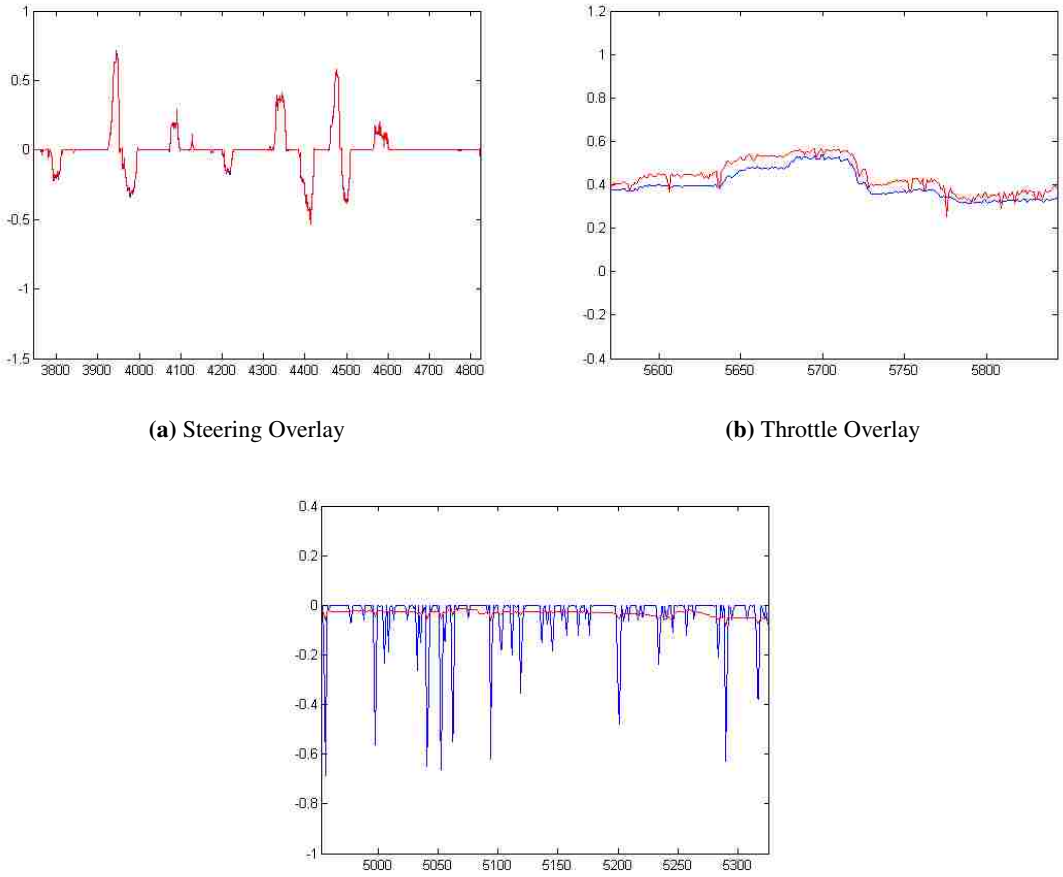
(c) Throttle Target Output

(d) Throttle Network Output

(e) Braking Target Output

(f) Braking Network Output

Figure 6.9: Comparison of preliminary NARX network alongside the target output

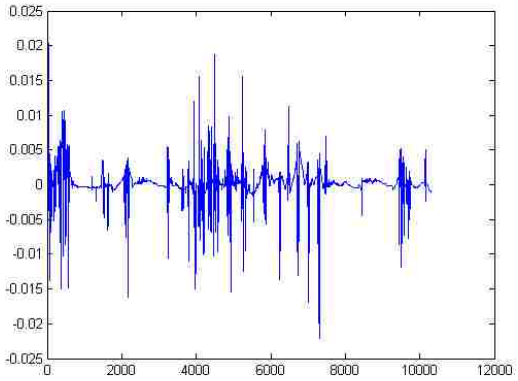


(a) Steering Overlay

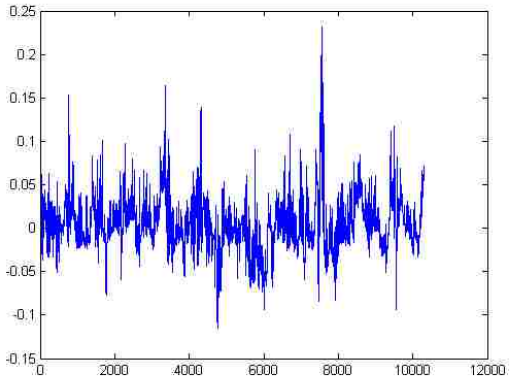
(b) Throttle Overlay

(c) Braking Overlay

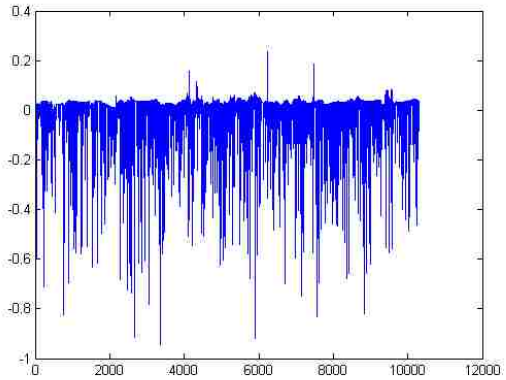
Figure 6.10: Magnified Overlays of Target Output(Blue) and Network Output(Red)



(a) Steering Error

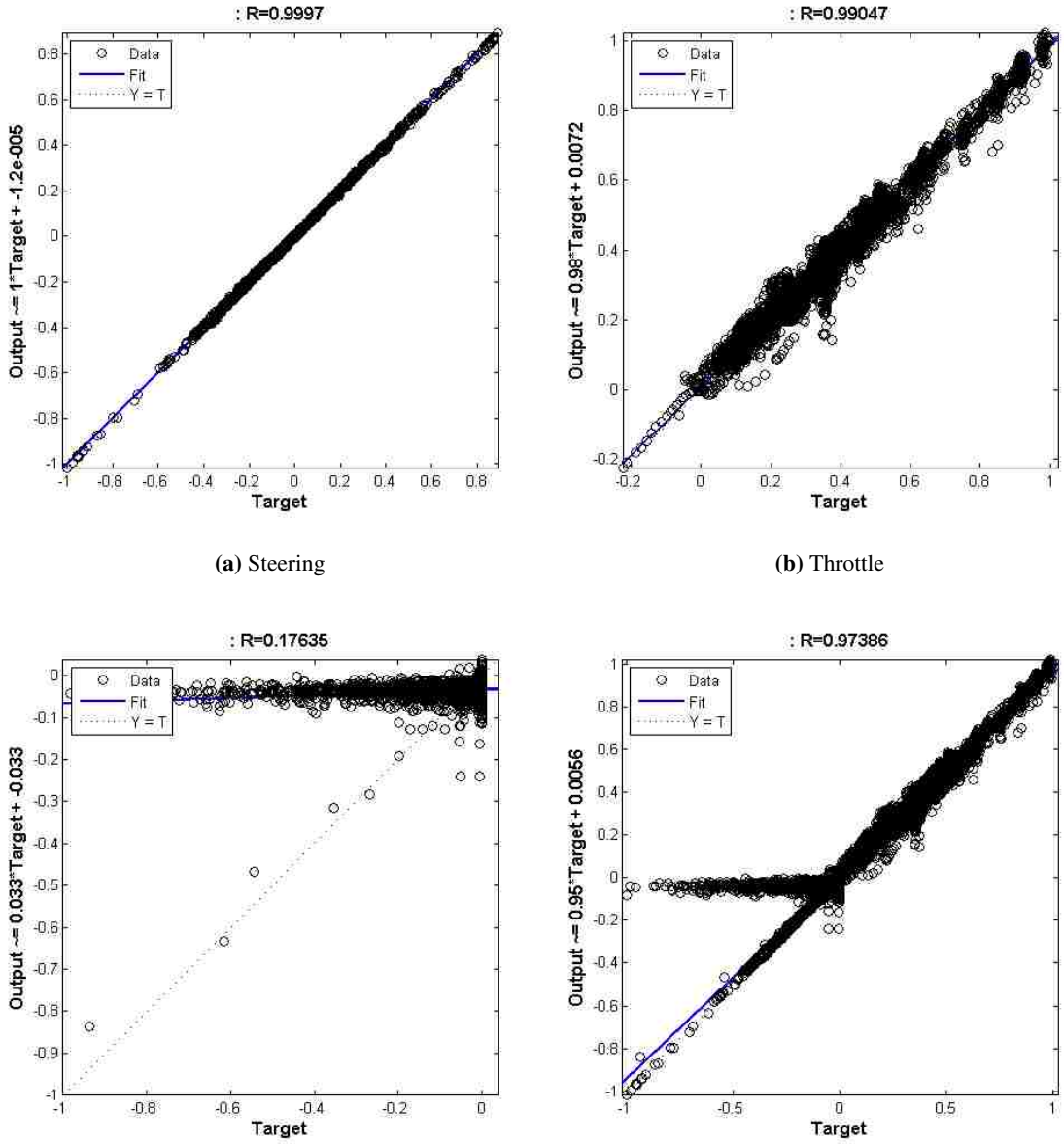


(b) Throttle Error



(c) Braking Error

Figure 6.11: Error Plots for Each Network Output



(a) Steering

(b) Throttle

(c) Braking

(d) Overall Regression

Figure 6.12: Regression plot for each individual variable and overall regression for preliminary NARX network

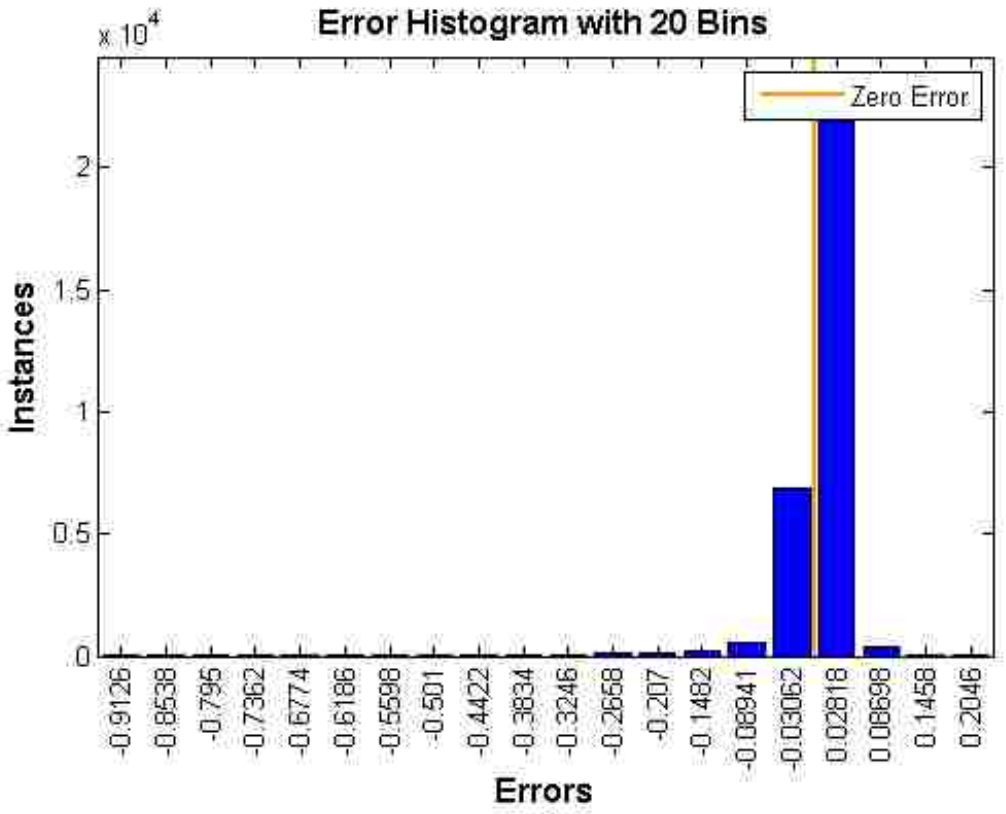


Figure 6.13: Error Histogram for Preliminary NARX network

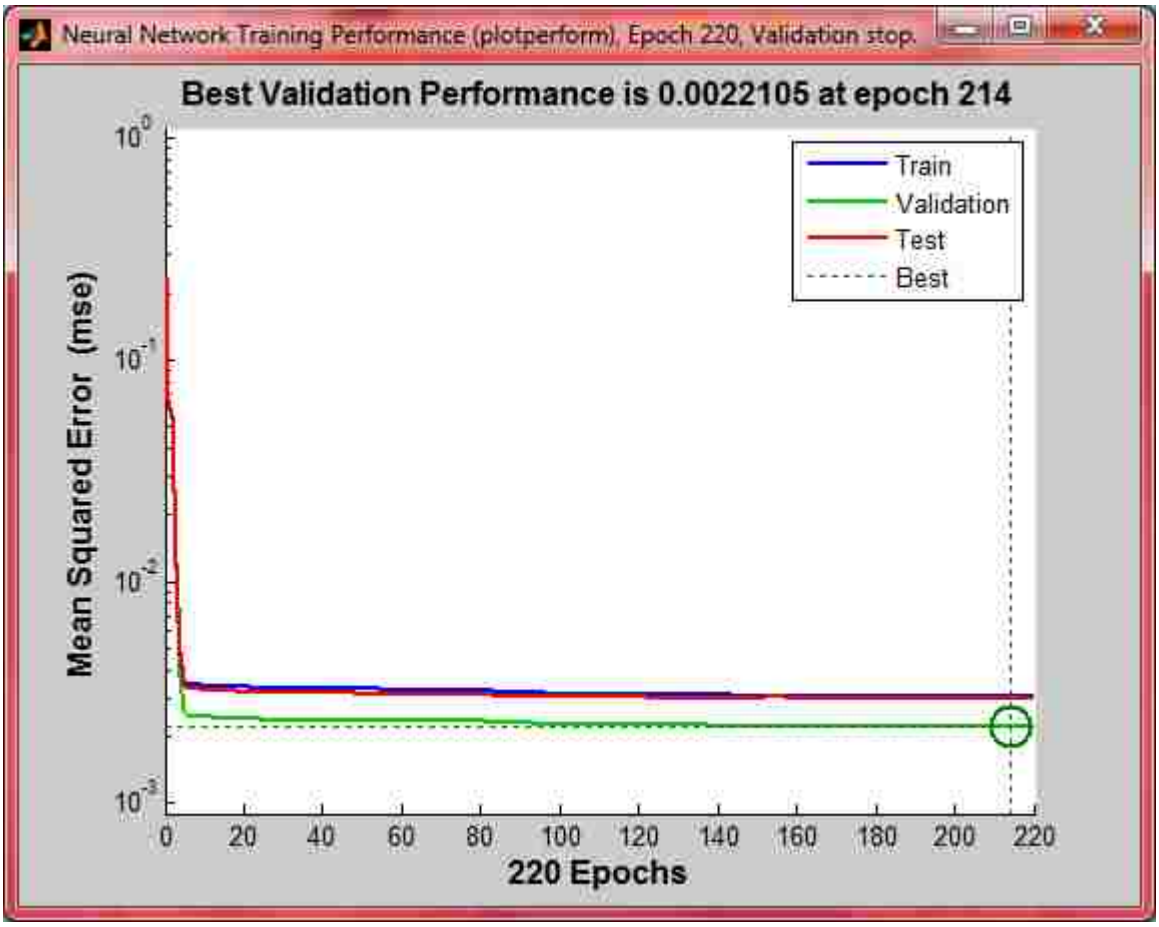


Figure 6.14: Training performance for Preliminary NARX network

NARX network results using the full dataset

As discussed, following an analysis of the performance of the the preliminary NARX network, further development of this type network was warranted using the full dataset, the results of which are presented below.

The training window (Figure 6.15, p.73) indicates that training was terminated after 44 iterations once the minimum error gradient was exceeded. According to the training window, the training time was approximately 13 hours and a mean squared error of $2.31\text{x}10^{-5}$ was attained.

According to Figures 6.16a & 6.16b (p.74) the NARX network was able to model the steering behaviour. Figures 6.16c & 6.16d illustrate that the NARX network was able to accurately represent the throttle behaviour as well, however, not to the same degree as witnessed with the steering behaviour. Figures 6.16e & 6.16f illustrate an accurate representation of the braking behaviour as

well. Overall, the output plots of the NARX network seem to validate two hypotheses which are: that the NARX network is better suited for driver modeling over an MLP network and that using a global scaling factor as opposed to a local scaling factor for normalizing the data would yield better network performance. All three overlay plots in figure 6.17 (p.75) help to further illustrate the accuracy of the network for all three outputs as network output plots are almost indistinguishable from those of the target output.

The error plots (Figure 6.18, p.76) further serve to illustrate the quality of the performance as all three plots display errors of a relatively small magnitude. It is apparent from these plots however, that the throttle performance(Figure 6.18b) is inferior to that of both the steering & braking performances displayed in Figures 6.18a & 6.18c, respectively. The same characteristics regarding the performance of each individual output was also observed in the performance of the MLP network.

The regression plots (Figure 6.19, p.77) help to validate the observations made based on the output plots. The steering regression (Figure 6.19a) plot illustrates a very strong correlation with very few outliers, the throttle regression (Figure 6.19b) plot demonstrates that an acceptably strong correlation was achieved however there are more outliers than for the regression plot for the steering behaviour, finally Figure 6.19c demonstrates a very strong correlation with the braking data and large improvement over the performance witnessed in the preliminary network. The correlation factors for the steering, throttle and braking behaviour were 0.99613, 0.99577 and 0.99665 respectively and the overall correlation factor was 0.9998, this information indicates that from a mathematical perspective, the trained NARX network is very accurate and would produce a high fidelity model of a driver.

The error histogram (Figure 6.20, p.78) for the NARX further serves to reinforce the quality of the performance of the NARX network as most of the errors have been placed in the bin closest to zero.

The training performance plot (Figure 6.21, p.79) indicates that the performance of all three datasets (training, validation and test) all followed a similar level of performance at each iteration. This is consistent with the stopping criteria that was used to terminate network training as the performance of the validation dataset never exceeded the performance of the training dataset often enough to halt network training.

Overall, following an examination of the plots, it can be concluded that the performance of the NARX network was very good and, as expected, it exceeded that of the MLP network. The performance of the NARX network illustrates the importance of proper data scaling in comparison to the performance of the preliminary network.

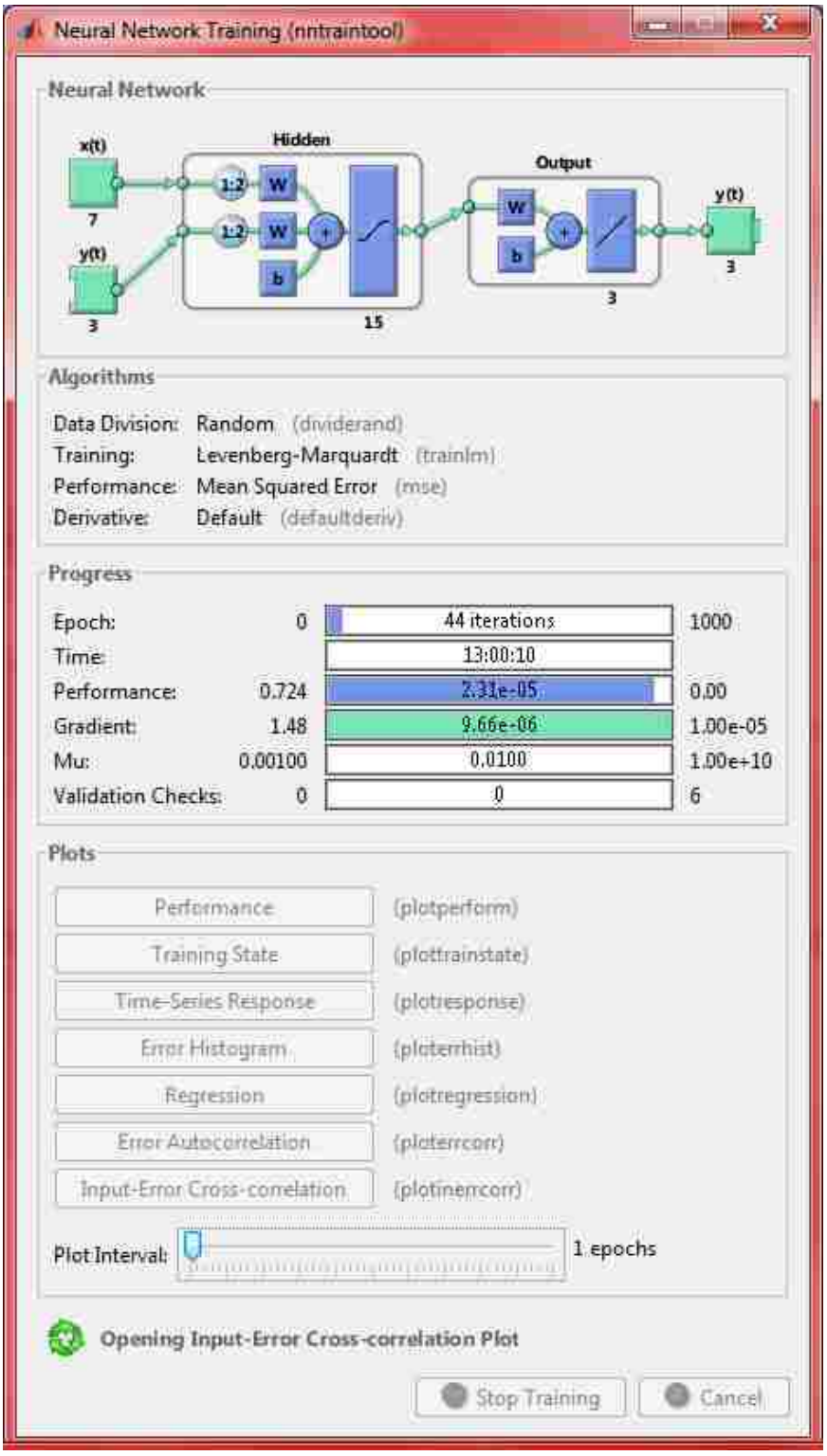
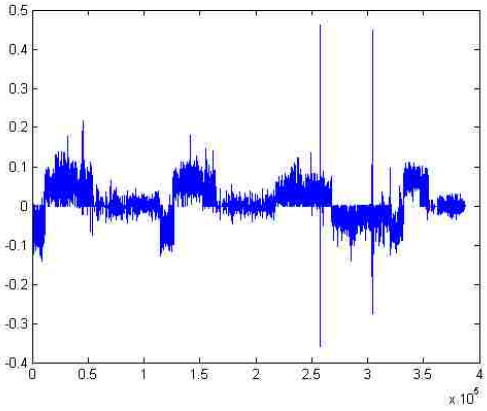
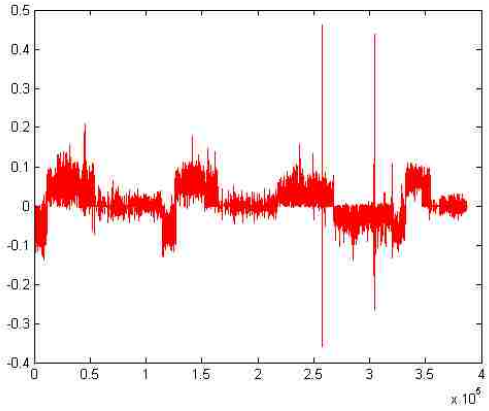


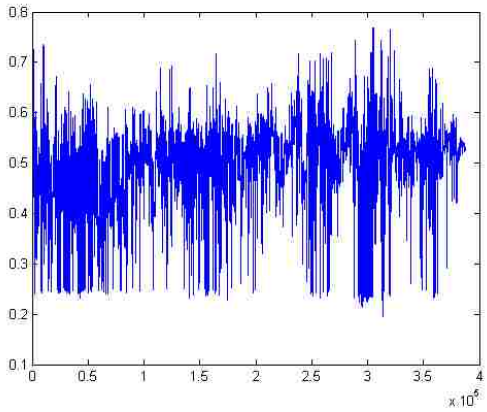
Figure 6.15: Training Window for NARX network



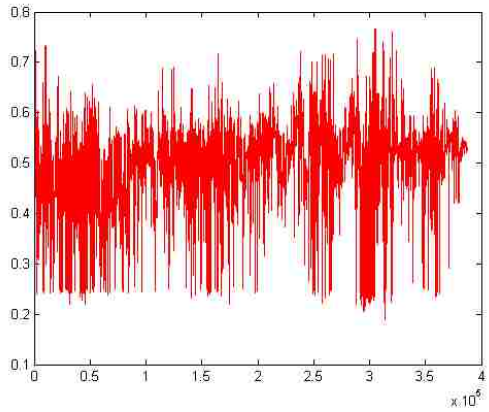
(a) Steering Target Output



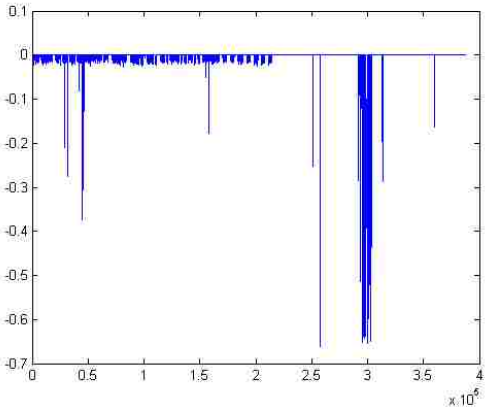
(b) Steering Network Output



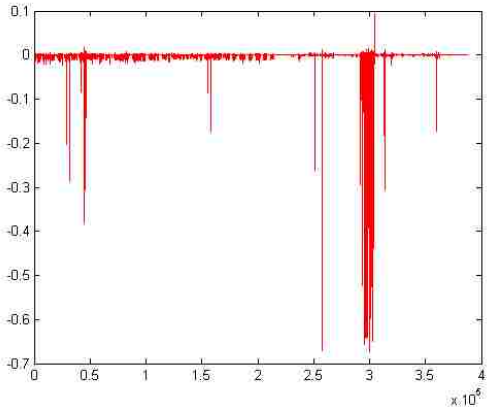
(c) Throttle Target Output



(d) Throttle Network Output

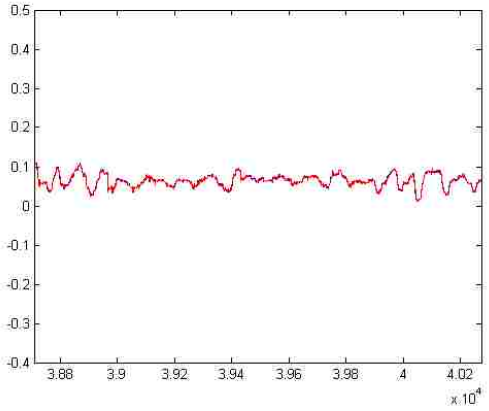


(e) Braking Target Output

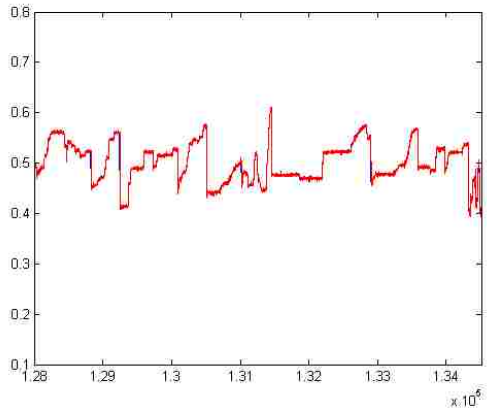


(f) Braking Network Output

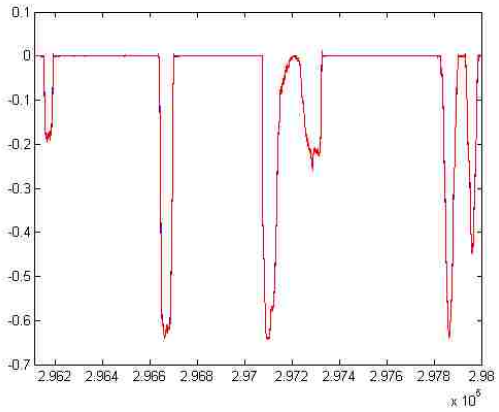
Figure 6.16: Comparison of NARX network alongside the target output



(a) Steering Overlay

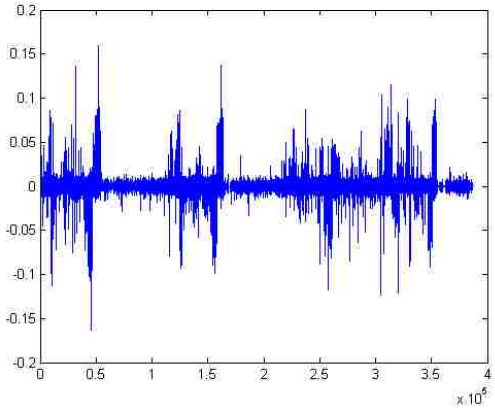


(b) Throttle Overlay

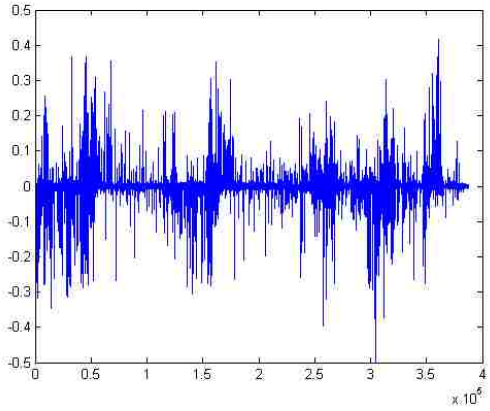


(c) Braking Overlay

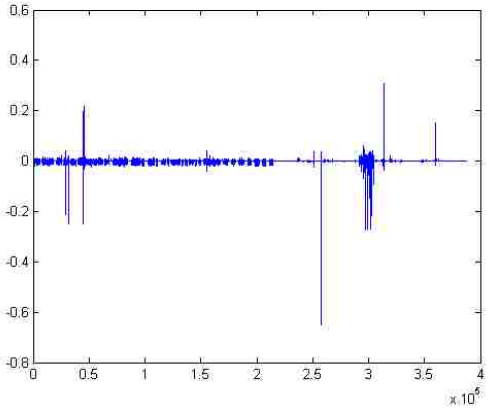
Figure 6.17: Magnified Overlays of Target Output(Blue) and Network Output(Red)



(a) Steering Error

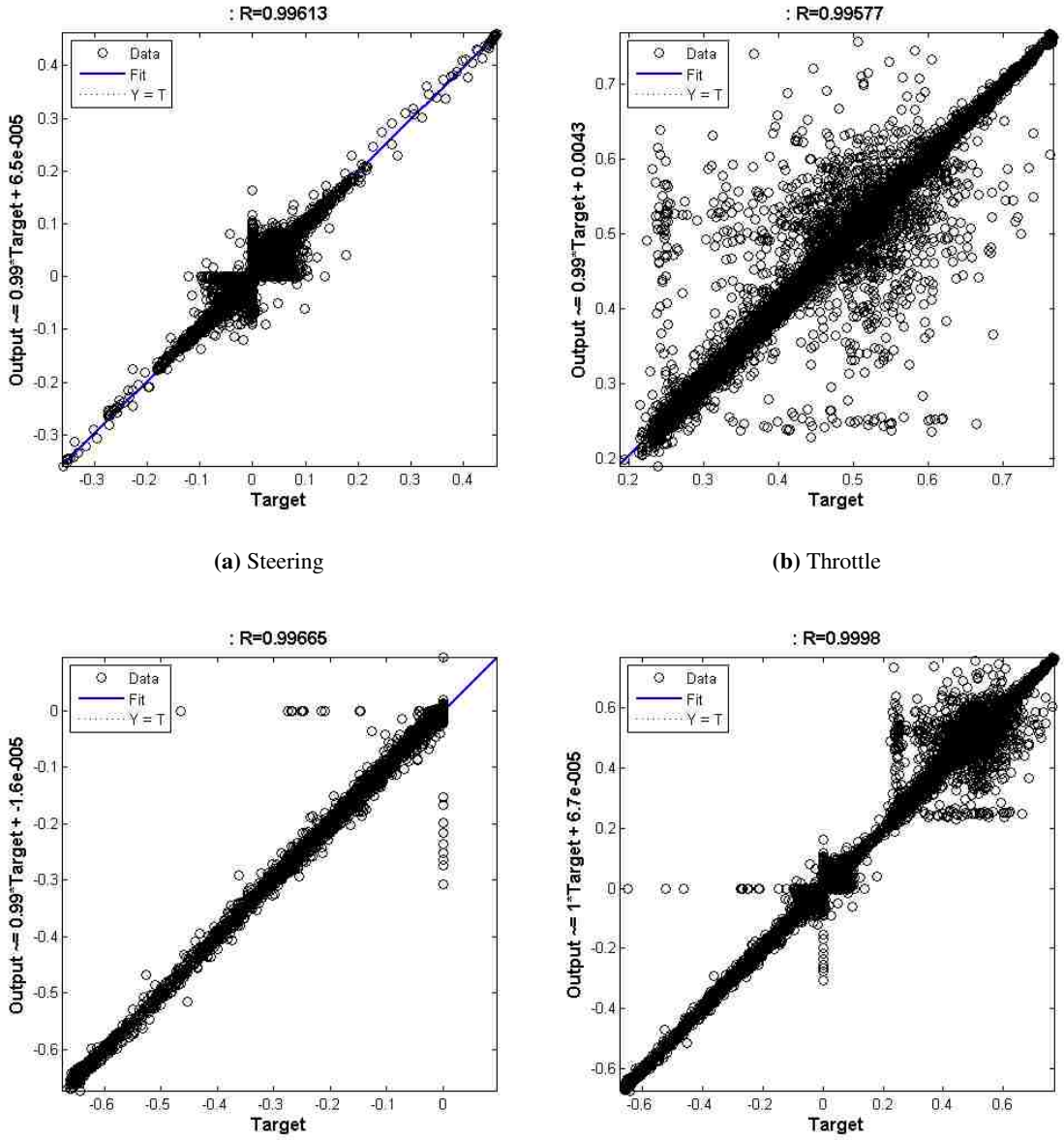


(b) Throttle Error



(c) Braking Error

Figure 6.18: Error Plots for Each Network Output



(a) Steering

(b) Throttle

(c) Braking

(d) Overall Regression

Figure 6.19: Regression plot for each individual variable and overall regression for NARX network

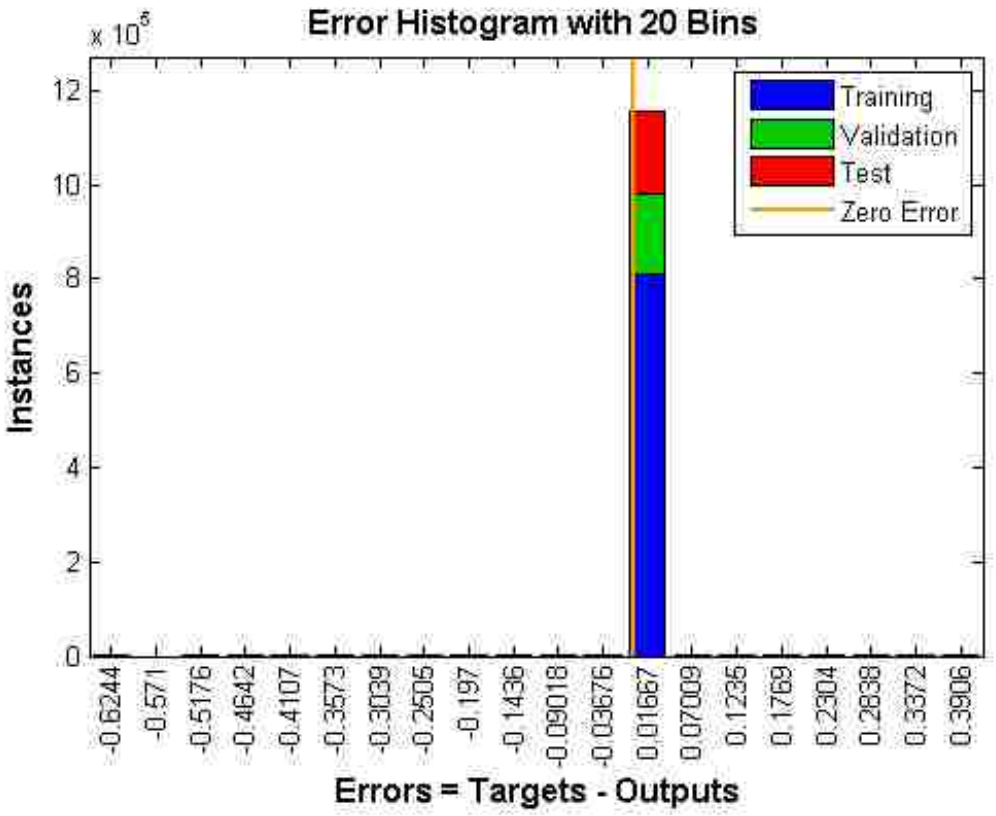


Figure 6.20: Error Histogram for Preliminary NARX network

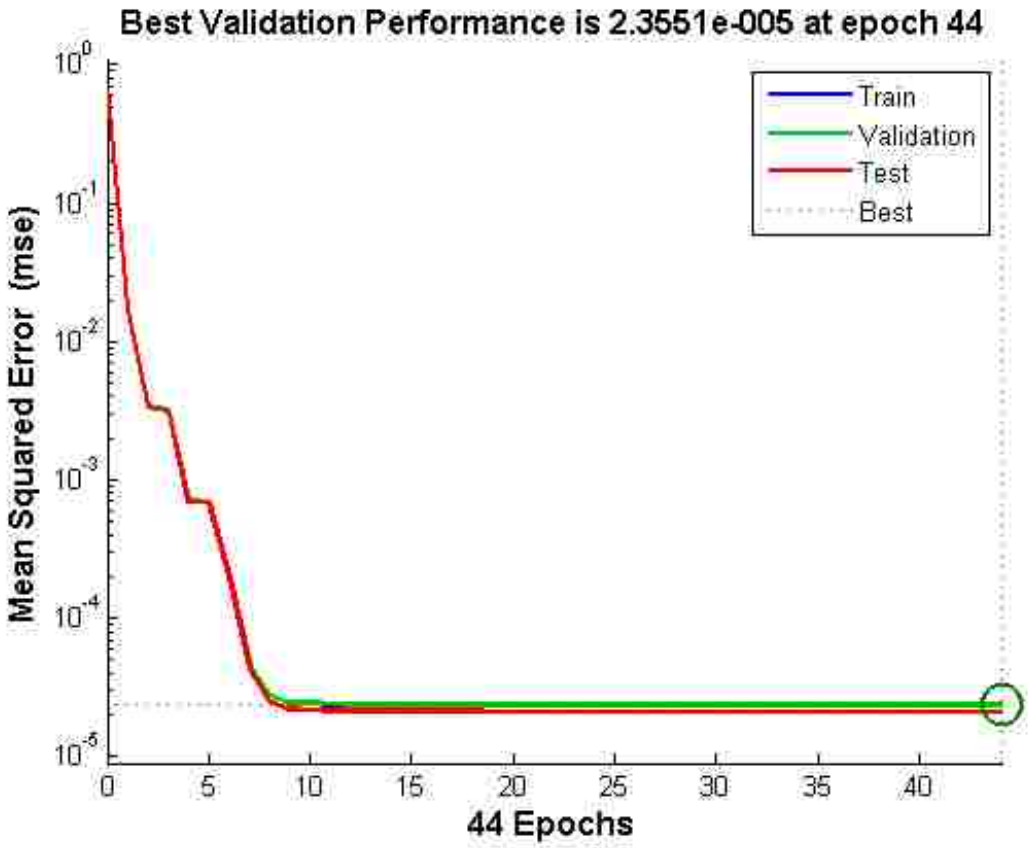


Figure 6.21: Training performance for Preliminary NARX network

6.2 Classification Network

In the same manner that the performance of the neural networks used for the driver model were assessed, the performance of the networks developed for classifying driver behaviour are analyzed based on several criteria displayed in a series of plots. It should be noted that when examining the plots, that class 1 is associated with acceptable behaviour, class 2 is associated with marginal behaviour, and class 3 is associated with unacceptable behaviour. The first plot presented is the training window generated by MATLAB®. This is the same window generated for the driver model network discussed in Section 6.1. The next figure presented is what is called the confusion matrix. The confusion matrix presents the class that the input vector was associated with in the training dataset and the class that the network associates that particular input vector with once it has been trained. Evidently, if the trained network is capable of associating each input vector to its correct

class, it is indicative that it has been properly trained and can be trusted to function properly when used in its desired application. The next figure used to assess the performance of the classification network is what is called the receiver operating characteristics (ROC); the MATLAB® help files [40] describe how the ROC parameter calculates the true positive ratio which it describes as "the number of outputs greater or equal to the threshold, divided by the number of one targets" and the false positive ratio which it describes as "the number of outputs less than the threshold, divided by the number of zero targets"; the MATLAB® help files [40] also discuss how it is desirable for the ROC plot to "hug" the left and top edges of the plot. According to Witten[43], the ROC characterizes the tradeoff between the "hit rate" and the "false alarm rate". The vertical axis plots the number of positives in the sample expressed as a percentage of the total number of positives with respect to; the number of negatives in the sample as a percentage of the total number of negative on the horizontal axis. When plotting the ROC chart, movement in the vertical direction corresponds to that particular input being positive; while movement on the horizontal direction corresponds to that particular input being negative. There are some important points that must be stressed regarding ROC which are, that the magnitude of displacement is not important given that they are expressed as percentages, ideally the plot should be close to the top lefthand corner, and that a diagonal line from the lower lefthand corner corresponds to the behaviour of a random dataset. The final plot used to assess the performance of the classification network is the training performance, which, in this case, plots the means squared error at each iteration.

6.2.1 Learning Vector Quantization Network Using the Full Dataset

Figure 6.22 (p.82) displays the training window for the classification network that was trained using the full dataset. Training was stopped by the user as the window indicates that the mean squared error experienced no change following 292 iterations that took in excess of 117 hours to complete. The training window indicates that a mean squared error of 0.00957 was achieved. Initial observation appears to indicate that an acceptable network was trained. Further investigation into network performance reveals that training did not produce a network that functions to an acceptable level.

The confusion matrix (Figure 6.23, p.83) reveals significant information regarding the quality of the network. Essentially the confusion matrix indicates that following training, the network classified all of the input vectors in the same class, which is not a desirable outcome. One observation made was that an overwhelmingly large portion of the data belonged to the same class; therefore, if the rest of the data was misclassified it would still have very little effect on the performance of the network. It was also believed that the manner in which the data was divided prevented the network from being able to classify each vector properly and as a result a modified input dataset was created. The performance results of this network are not presented as MATLAB® destabilized which

made it impossible to obtain vital performance information. It should be noted that the observed performance of the network prior to MATLAB® destabilizing exhibited extremely minimal benefits.

The receiver operator characteristics (Figure 6.24, p.84) also serve to reinforce the observation that the network failed to perform as desired. While the MATLAB® help files [40] state that it is desirable for the plots to "hug" the left and top edges of the plot, the plots in Figure 6.24 follow a line of y=x, according to Witten[43], this indicates that the performance corresponds to that of a random dataset.

The training performance plot in Figure 6.25 indicates that network performance did not improve to any extent during training. As a result, it is unlikely that the performance would have improved at all or without an exorbitant amount of training. A possible explanation for this is that the error surface may have had a very shallow gradient as other networks were trained using the L-M training algorithm and their training was quickly terminated as the minimum error gradient was very quickly reached.

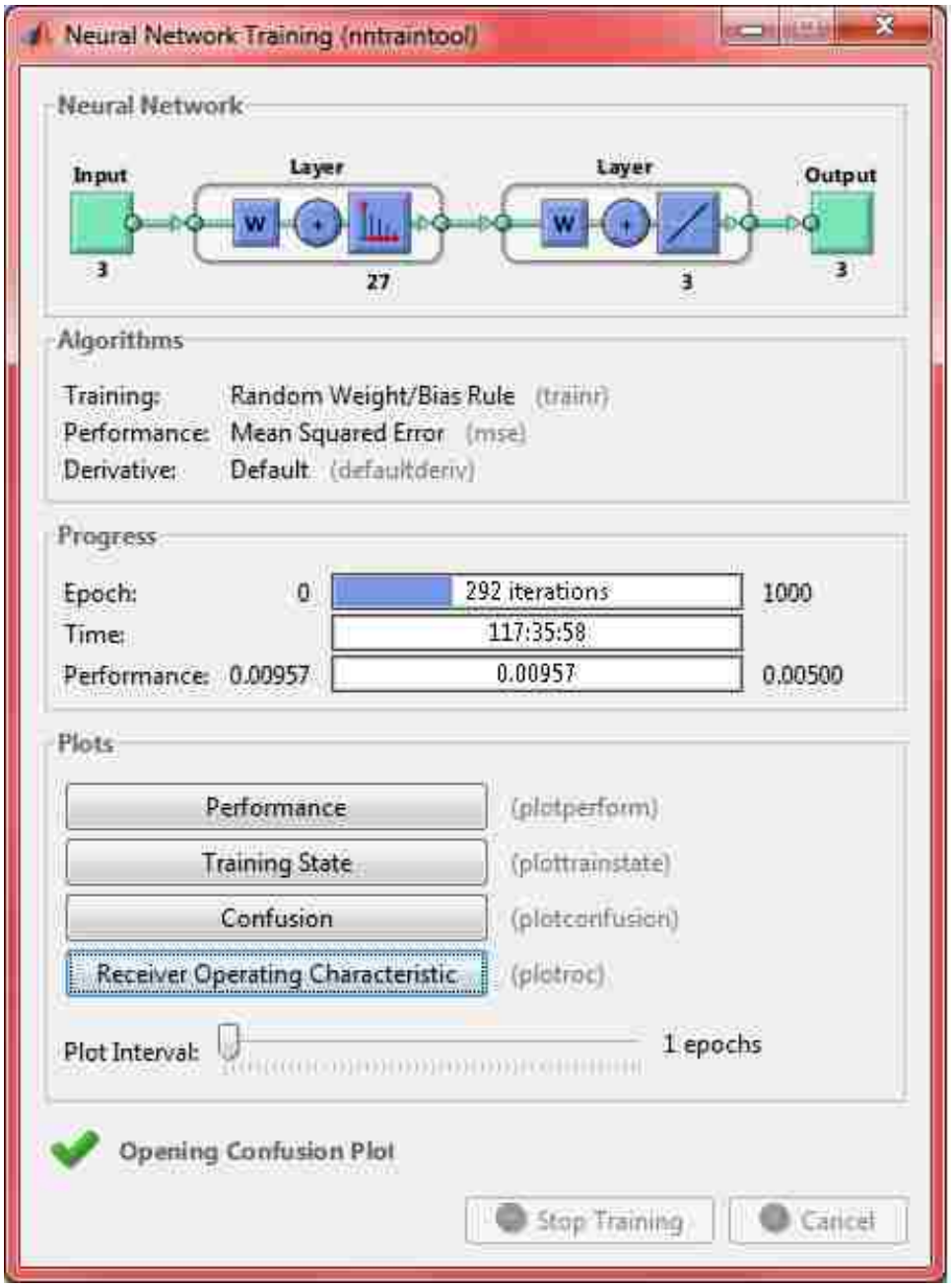


Figure 6.22: Training Window for LVQ network using the full dataset

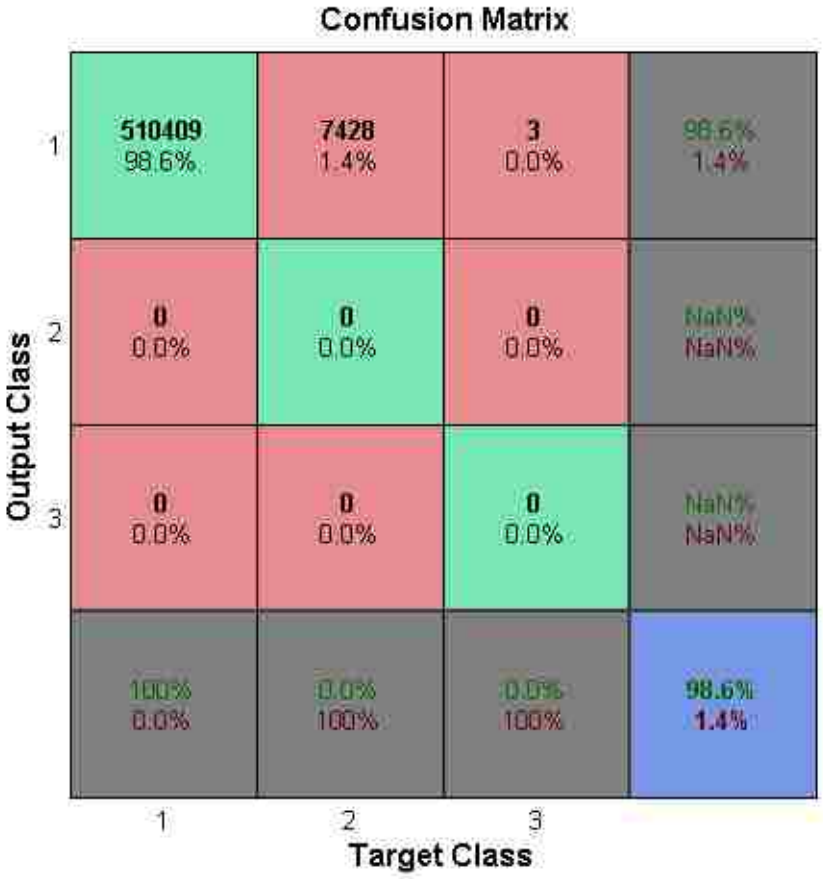
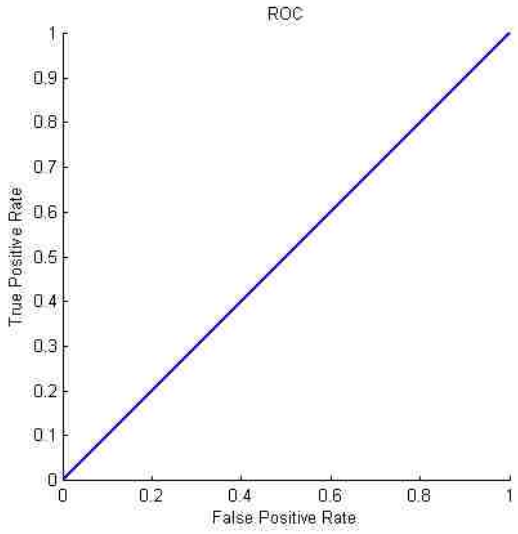
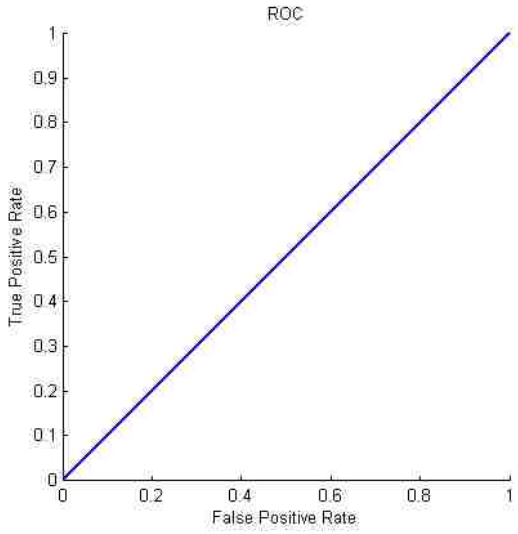


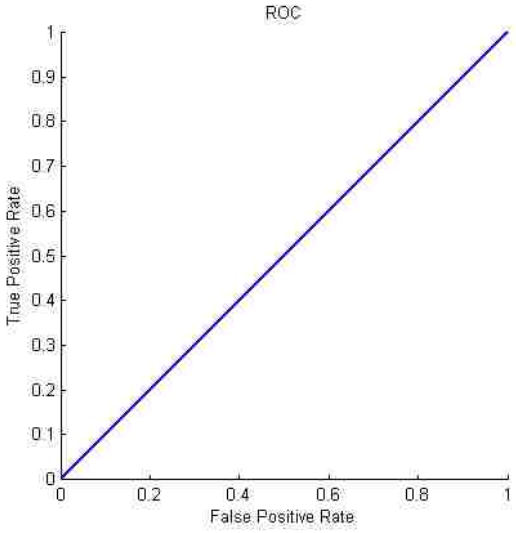
Figure 6.23: Confusion Matrix for LVQ network using the full dataset



(a) Class 1



(b) Class 2



(c) Class 3

Figure 6.24: Receiver Operating Characteristics for Each Class of the LVQ Network Using the Full Dataset

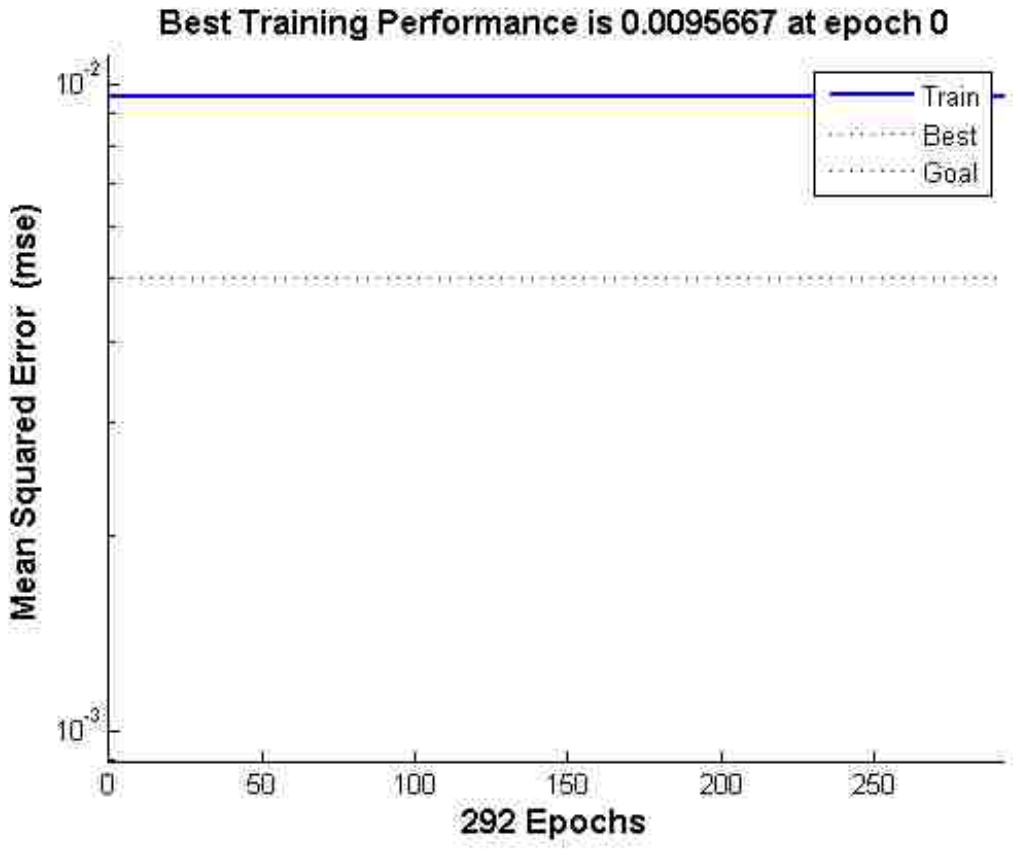


Figure 6.25: Training performance for LVQ network using the full dataset

6.2.2 Learning Vector Quantization Network Using a Portion of the Dataset

In order to try and provide evidence that an LVQ network can be used to classify driver behaviour, an investigation of the performance using a portion of the dataset was carried out. The training window (Figure 6.26, p.87) indicates that training was terminated following 1000 iterations taking nearly 16 hours to do so. Following training, the network was able to attain a mean squared error of 0.0923.

The confusion matrix (Figure 6.27, p.88) helps to reinforce the observation made from the training window that the network performance is not satisfactory, while the network was able to associate some of the input vectors with their appropriate class, there were many vectors that it failed to properly classify. This was especially noticeable with input vectors belonging to class 3, which is associated with behaviour deemed to be unacceptable, in which it failed to properly identify any of the associated input vectors. It can be noted, however, that the fact that it was able to

at least properly classify data for more than just one class represents an improvement over the first LVQ network developed.

The receiver operator characteristic plots (Figure 6.28, p.89) represent a validation of the observations made from the confusion matrix as the plots move closer to the left and top edges of the figure for classes 1 and 2. The plot for class 3 is identical to the plot generated from the network using the entire dataset in that it forms a line of y=x; this is also consistent with the observations made from the confusion matrix as there was no improvement in performance for identifying that particular class of behaviour.

Finally, the training performance plot (Figure 6.29, p.90) illustrates that as training progressed, the mean squared error was reduced from the first iteration; however, it appears as if it is oscillating within a region of the plot indicating that the network was perhaps searching in different directions on the error surface in which the error could be minimized. The plot may also indicate that further training with this particular dataset may not yield any improvement in network performance.

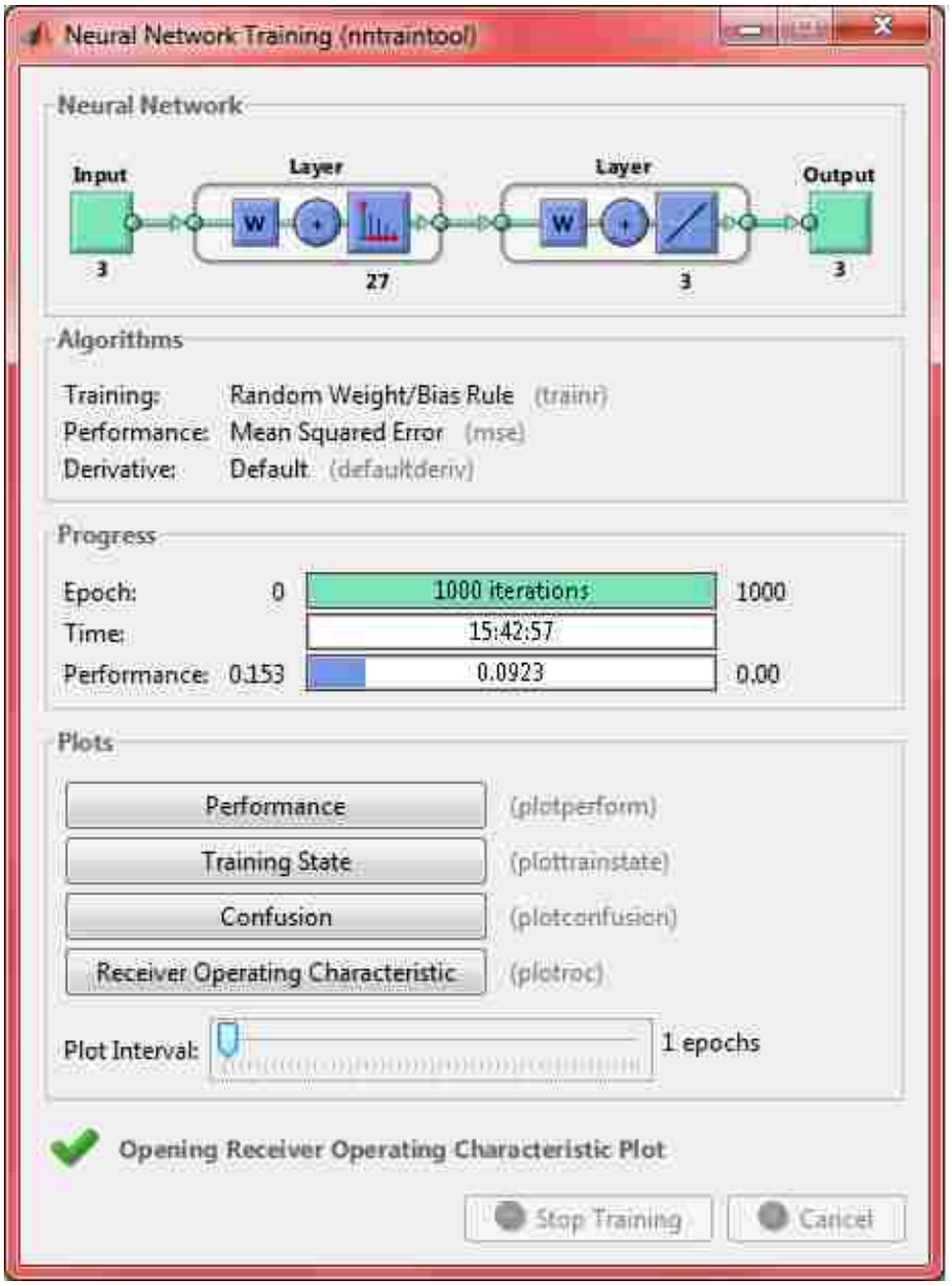


Figure 6.26: Training Window for LVQ network using a portion of the dataset

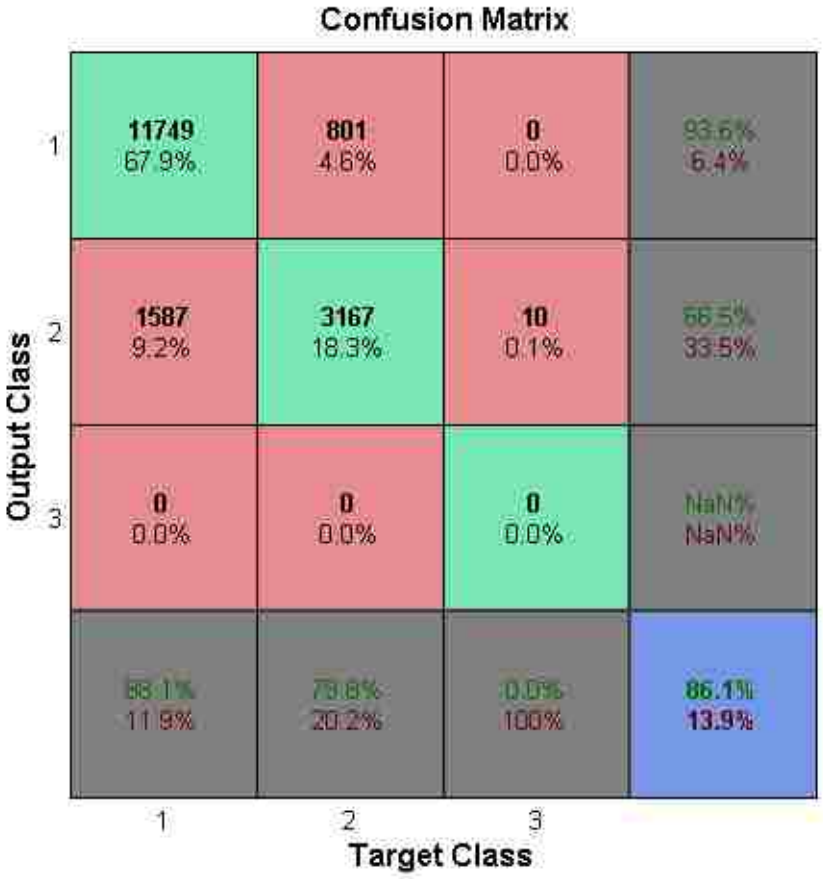
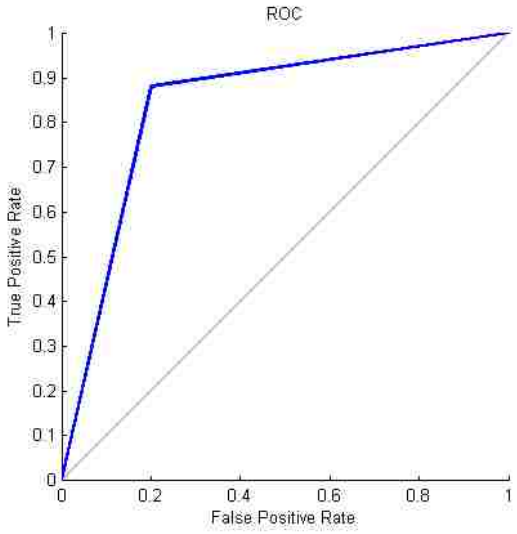
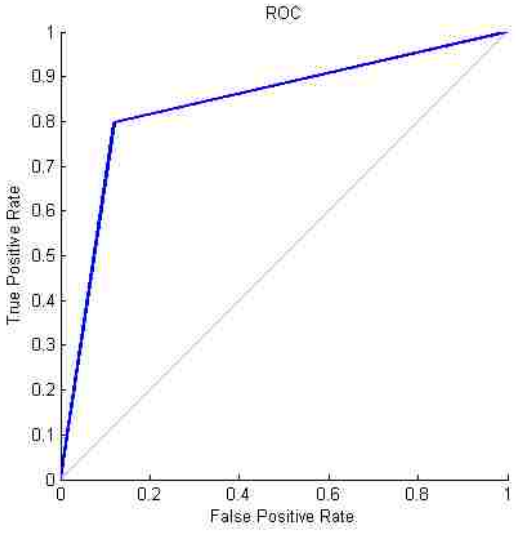


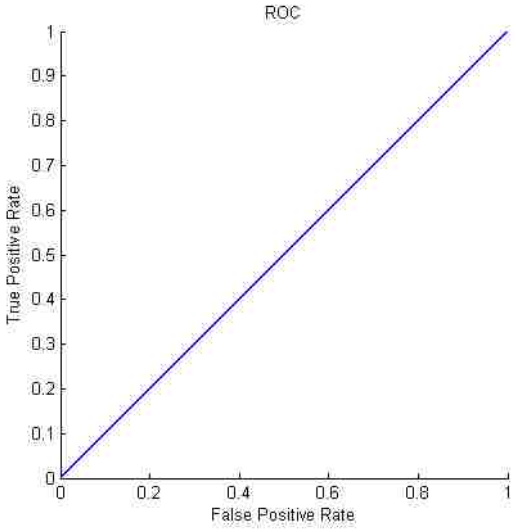
Figure 6.27: Confusion Matrix for LVQ network using a portion of the dataset



(a) Class 1



(b) Class 2



(c) Class 3

Figure 6.28: Receiver Operating Characteristics for Each Class of LVQ Network Using a portion of the Dataset

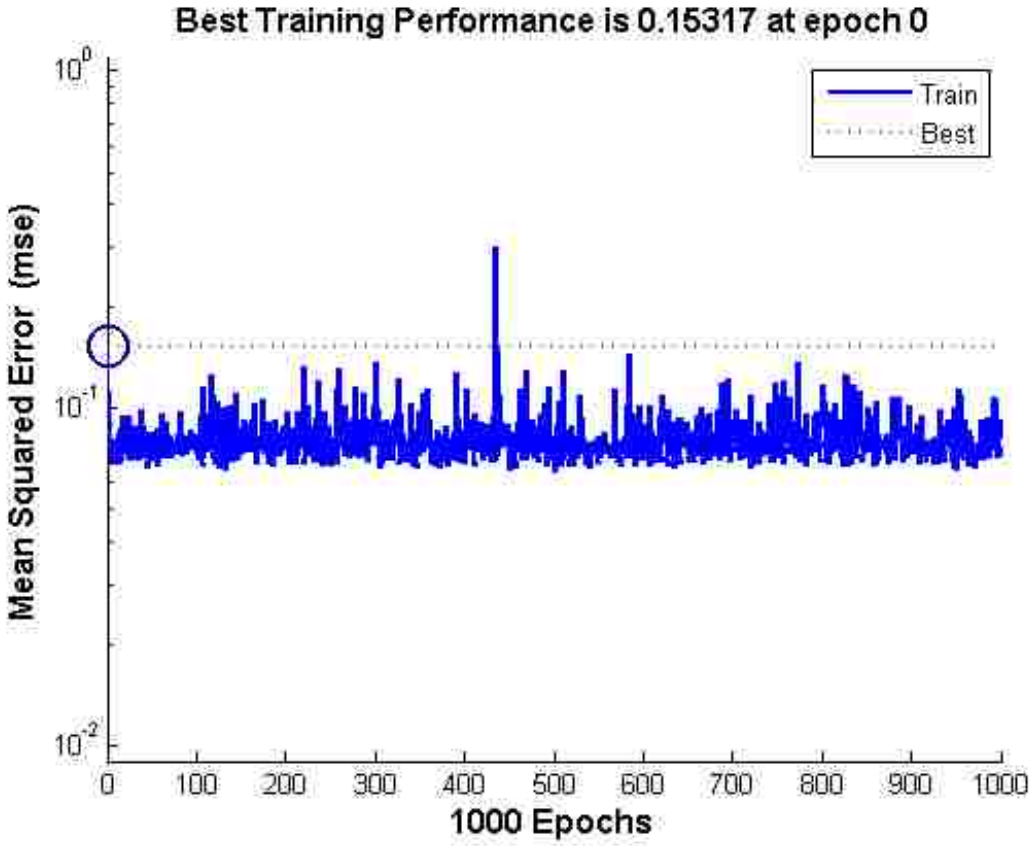


Figure 6.29: Training performance for LVQ network using a portion of the dataset

Chapter 7

Conclusions & Recommendations

The research presented reveals many conclusions regarding driver modeling and how it can be applied to assess driver behaviour. First of all the review of literature and past work reveals that a significant amount of work has been accomplished in the field of driver modeling and that the ultimate goal appears to be the development of a universal model. While the development of such a model is beyond the scope of this research, the work does represent an application of some of the information that has been learned and published. The previous accomplishments also served as a guide for developing the driver assessment tool, especially in the area of employing neural networks in the driver modeling role.

The conclusions that can be drawn from the actual research undertaken is that based on the results discussed in Chapter 6, a functional driver model was developed using neural networks. According to the standard methods used to evaluate neural network performance, the driver model accurately modeled the driver behaviour. Another conclusion that was reached was that the NARX network outperformed the MLP neural network in the role as a driver model.

While the network for the driver model portion of the assessment tool appeared to perform well the same cannot be said for the classification portion. Since the network must be designed to organize data, the use of an SOM is appropriate, and because the ultimate classification for the input vectors in the training data is known, the LVQ network is in fact the appropriate structure. The main issue with the development of this portion of the assessment tool appears to lie in the training data. At this point a metric to differentiate between acceptable, marginal and unacceptable behaviour has yet to be developed; therefore, it is difficult to properly divide the data. Combined with the fact that the data was more or less arbitrarily divided, the end result was that most of the input vectors were classified as being in the acceptable range, and the network had difficulty learning how to distribute the data.

Based on the conclusions for each portion of the assessment tool, the overall conclusion that can be reached is that the work represents a successful first attempt in the development of an elderly driver assessment tool. While the classification network did not meet expectations, a significant amount of information regarding how to proceed in future developments was learned along with many pitfalls that must be avoided, many of which are discussed in the recommendations section.

7.1 Recommendations

Based on the conclusions reached following the work, the recommendations for future work are as follows:

- Acquire a new dataset from a more sophisticated driving simulator.

  The data used to develop this initial model was acquired from a simulator that employs a simplified linear vehicle model and the user interface provides no feedback to the driver. While the subjects of interest for this study operate the vehicle in the linear regime it is a well known fact that there are many non-linearities that exist in vehicle behaviour, particularly where the tires are concerned; the vehicle model used in the simulator fails to account for tire behaviour as there is no notion of individual wheels as discussed in Section 4.1. Vehicle feedback to the driver is also an important aspect, as drivers base many of their decisions on what they feel, particularly when steering the vehicle where the driver will feel a resistant tire force as they turn the wheel. Similarly, the wheels will also have a tendency of correcting themselves once the driver ceases to steer the wheel. This attribute is incorporated intentionally and is known as return steer; however, any further discussion of its behaviour is beyond the scope of this research. While highly sophisticated driving simulators with several degrees of freedom in terms of vestibular feedback do exist, they are excessive for the purpose of the development of the assessment tool. It would, however, be very beneficial to have access to a simulator that at least employs a high fidelity vehicle model, provides steering feedback to the subject and that uses clearer sensors for the throttle and braking inputs.

- Ensure that the scenario presented to the subjects requires them to deal with a wide variety of situations.

  The road situations that the subjects were exposed to are listed in Table 4.1 in Chapter 4. After examining this table it is evident that the number of situations that the subjects were exposed to are limited, in that there are no situations other than straight and curved roads and, where the curved roads are concerned, there is very little variety in road curvature. Ideally,

the driving scenario used to acquire the data should incorporate a variety of different road curvatures, lane changes and intersections to name some examples.

- Establish a proper metric to distinguish acceptable, marginal and unacceptable driving behaviour.

  As was discussed previously, when dividing the training data for the classification network between acceptable, marginal and unacceptable behaviour, there was no definitive method to do so, and as a result the division of data was little more than arbitrary. It is understood that the establishment of such a metric would require an extensive quantity of work in which the subjects behaviour would be observed. The observations would then be correlated with the data in order to determine the appropriate thresholds for each of the output variables, namely the steering, throttle and braking performance in accordance with each of the three classes of behaviour. The development of said metric would represent a very large step forward in the development of the driver assessment tool.

- Explore the incorporation of time based measures as discussed in Section 2.6 and gap acceptance theory into the driving behaviour metric.

  The literature review discovered several time based methods that can be used to quantify error. These methods could be used to enhance the metric to differentiate the behaviour using a quantitative method. Gap acceptance theory is a measure that is frequently employed in traffic engineering and could potentially be useful in assisting to provide a quantitative method to differentiate driving behaviour.

- Validation of the driver model portion using a vehicle simulation program such as CarSim®.

  While the driver model displayed acceptable performance in accordance with the methods typically used to assess the performance of neural networks, it's performance when exposed to new driving scenarios remains untested. By developing a co-simulation using MATLAB® and a vehicle simulator such as CarSim® it would be possible to present new scenarios to the driver model, where it would be required to respond with the appropriate commands for the steering, throttle and braking all while using a high fidelity vehicle model. Also neural network performance only reveals the response to the given inputs. In reality those inputs may not be adequate and more may be required; a co-simulation with a vehicle simulation program would reveal this. Theoretically, a proper driver model should be able to control the vehicle for virtually any reasonable scenario presented to it.

- Exploring the use of different types of neural networks for the driver model as well as optimizing the network's size.

While a NARX network using 15 hidden layer neurons and two delays was deemed to be the best network for driver modeling of the ones tested, the fact of the matter is there might be better options both from a performance and efficiency standpoint. In all likelihood, 15 hidden layer neurons is more than necessary and reducing the number of hidden layer neurons would decrease both training and simulation time as well as the amount of computing resources required without significantly reducing performance. Unfortunately time constraints prevented network simplification from taking place. It is also recommended that the use of an RBFN to act as a driver model be revisited, according to Lin [24] such networks performed very well in the driver modeling role.

- Train the classification network used in the second portion of the assessment tool using the data divided with a proper metric.

  Finally it goes without saying that once a proper metric has been established, it should be employed in order to properly divide the data according based on behaviour.

While the implementation of all the recommendations may seem somewhat excessive, they serve as a guide as to where to proceed with the development of the elderly driver assessment tool. The next step in the development of the model should be to prioritize the recommendations and implement them as deemed fit.

Bibliography

[1] L. D. T. M. N. N. S. M. T. N. Lavallière, M., "Multiple-session simulator training for older drivers and on-road transfer of learning," *Proceedings of the Fifth International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design*, 2009.

[2] J. Michon, "A critical review of driver behaviour models: What do we know, what should we do?," in *Human Behaviour and Traffic Safety* (R. S. L.A. Evans, ed.), pp. 487–525, New York: Plenum, 1985.

[3] H. E. Engstrom, J., "A general conceptual framework for modelling behavioural effects of driver support functions," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 61–84, London: Springer, 2007.

[4] N. L. Peters, B., "Modelling the driver in control," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 85–104, London: Springer, 2007.

[5] O. Carsten, "From driver models to modelling the driver: What do we really need to know about the driver?," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 105–120, London: Springer, 2007.

[6] B. E. P. V. Panou, M., "Modelling driver behaviour in european union and international projects," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 3–25, London: Springer, 2007.

[7] G. T. Cody, D., "Workshop on driver models: A step towards a comprehensive model of driving?," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 26–42, London: Springer, 2007.

[8] R. van der Horst, "Time-related measures for modelling risk in driver behaviour," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 235–252, London: Springer, 2007.

[9] R. Fuller, "Motivational determinant of control in the driving task," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 165–188, London: Springer, 2007.

[10] G. Wilde, "Target risk," September 2010.

[11] T. Vaa, "Modelling driver behaviour on basis of emotions and feelings: Intelligent transport systems and behavioural adaptations," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 208–232, London: Springer, 2007.

[12] K. Bengler, "Subject testing for evaluation of driver information systems and driver assistance systems – learning effects and methodological solution," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 123–134, London: Springer, 2007.

[13] W. Janssen, "Modelling driver's risk taking behaviour," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 135–146, London: Springer, 2007.

[14] F. Saad, "Dealing with behavioural adaptations to advanced driver support systems," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 147–161, London: Springer, 2007.

[15] H. Summala, "Towards understanding motivational and emotional factors in driver behaviour: Comfort through satisficing," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 189–207, London: Springer, 2007.

[16] C. L. Gibson, J.J., "A theoretical field-analysis of automobile-driving," *The American Journal of Psychology*, vol. 51, no. 3, pp. 453–471, 1938.

[17] D. Parker, "Driver error and crashes," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 266–274, London: Springer, 2007.

[18] C. MacAdam, "Understanding and modeling the human driver," *Vehicle System Dynamics*, vol. 40, no. 1-3, pp. 101–134, 2003.

[19] T. Jürgensohn, "Control theory models of the driver," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 277–292, London: Springer, 2007.

[20] C. K. Weir, D.H., "Review of control theory models for directional and speed control," in *Modelling Driver Behaviour in Automotive Environments* (P. Cacciabue, ed.), pp. 293–311, London: Springer, 2007.

[21] J. R. S. A. Bengtsson, J., "Modeling of drivers' longitudinal behaviour," in *Nonlinear and Hybrid Systems in Automotive Control* (R. A. Johansson, R., ed.), pp. 41–58, London: Springer, 2003.

[22] A. Ungoren, "An adaptive lateral preview driver model," *Vehicle System Dynamics*, vol. 43, no. 4, pp. 245–259, 2005.

[23] N. L. Kiencke, U., *Automotive Control Systems*. New York: Springer, 2000.

[24] T. P. Z. W. Y. Q. Lin, Y., "Artificial neural network modeling of driver handling behaviour in a driver-vehicle-environment system," *International Journal of Vehicle Design*, vol. 37, no. 1, pp. 24–45, 2005.

[25] F. P. M. M. Johrendt, J.L., "Streamlining automotive product development using neural networks," *International Journal of Vehicle Design*, vol. 47, no. 1-4, pp. 19–36, 2008.

[26] I. H. K. Y. A. M. Ishio, J., "Vehicle-handling quality evaluation through model-based driver steering behaviour," *Vehicle System Dynamics*, vol. 46, no. Suppl., pp. 549–560, 2008.

[27] S. T. Jurecki, R., "Driver model for the analysis of pre-accident situations," *Vehicle System Dynamics*, vol. 47, no. 5, pp. 589–612, 2009.

[28] H. M. D. H. Beale, M.H., *Neural Network Toolbox User's Guide*. The MathWorks, Inc., 3 Apple Hill Drive Natick, MA, 2011.

[29] S. Samarsinghe, *Neural Network for Applied Sciences and Engineering*. New York: Auerbach, 2007.

[30] M. Powell, "Restart procedures for the conjugate gradient method," *Mathematical Programming*, vol. 12, pp. 241–254, 1977.

[31] M. Moller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, pp. 525–533, 1993.

[32] R. Battiti, "First- and second-order methods for learning: Between steepest descent and newton's method," *Neural Computation*, vol. 4, no. 2, pp. 141–166, 1992.

[33] M. W. W. M. Gill, P.E., *Practical Optimization*. New York: Academic Press, 1981.

[34] M. Orr, "Introduction to redial basis function networks," technical report, Centre for Cognitive Science, University of Edinburgh, 2, Buccleuch Place, Edinburgh, Scotland.

[35] M. D. Milliken, W.F., *Race Car Vehicle Dynamics*. 400 Commonwealth Drive, Warrendale, PA: Society of Automotive Engineers, 1995.

[36] T. N. T. M. N. N. S. M. L. D. Lavallière, M., "Visual inspections made by young and elderly drivers before lane changing," *Advances in Transportation Studies an international Journal*, vol. Special Issue, pp. 23–31, 2007.

[37] R. T. A. B. K. D. A. F. H. J. C. J. Allen, R.W., "A low cost pc based driving simulator for prototyping and hardware-in-the-loop applications," *SAE Technical Paper Series*, vol. SP 1361, pp. 35–47, 1998.

[38] Systems Technology, Inc., *STISIM Drive (SDL) - BSAV - Begin Block Save*.

[39] M. Robitaille.

[40] The MathWorks, Inc., 3 Apple Hill Drive Natick, MA, *MATLAB Help Files*, 2011.

[41] J. Johrendt, *Optimizing Road Test Simulation Using Neural Network Modeling Techniques*. PhD thesis, University of Windsor, 401 Sunset Ave., Windsor, Ontario.

[42] K. Swingler, *Applying Neural Networks: A Practical Guide*. San Francisco: Academic Press, 2001.

[43] F. E. Witten, I.H., *Data Mining: Networks: Practical Machine Learning Tools and Techniques*. San Francisco: Morgan Kaufmann Publishers, 2005.

Appendix A

Permission from SAE to Reprint Paper 2011-01-0432

Dear Mr. Levesque,
Thank you for your correspondence requesting permission to reprint SAE paper number 2011-01-0432 - which you co-authored - in your thesis to be submitted to the University of Windsor.

Permission is hereby granted and subject to the following conditions:

- Permission is for this one time use only. New requests are required for further use or distribution of the SAE material.
- The following credit statement must appear on the paper: "Reprinted with permission from SAE Paper No. 2011-01-0432 ©2011 SAE International."
- This permission does not cover any third party copyrighted work which may appear in the material requested. Permission must be obtained from the original source.

Again, thank you for contacting SAE for this permission.

Best regards,

Terri Kelly
Intellectual Property Rights Administrator
SAE International
Phone: 001.724.772.4095; Fax: 001.724.776.9765
E-mail: terri@sae.org

Appendix B

M-Code for Elderly Driver Assessment Tool

```
clear
clc
%Values used to scale the data
maxi=[112,0.6314,4.6105,8.9386,0.5122,112,0.0040];
maxo=[148,4169,768033];
%Load the driver model neural network
load('NARX15.mat');
%Load the path to the file to be simulated
load(File);
%For loops to scale the input and output variables
for k=1:7;
    %Divide by the maximum value to normalize
    input(k,:)=input(k,:)./maxi(k);
end
for k=1:3;
    %Divide by the maximum value to normalize
    output(k,:)=output(k,:)./maxo(k);
end
%Neural Network toolbox commands to convert the data to a form usable by
%the neural network
inputSeries = tonndata(input,true,false);
outputSeries = tonndata(output,true,false);
%Command to prepare the data specifically for use specifically by a NARX
%network
[inputs,inputStates,layerStates,outputs] =
```

```
preparets(NARX_15,inputSeries,{},outputSeries);
%Simulate using the driver model neural network
netoutputs=NARX_15(inputs,inputStates,layerStates);
%Convert
netoutput=cell2mat(netoutputs);
%De-scale the data to its original values
for k=1:3;
    %Multiply by the maximum value to reverse normalizing
    netoutput(k,:)=netoutput(k,:).*maxo(k);
    output(k,:)=output(k,:).*maxo(k);
end
%Determine the length of output vectors
l=length(output);
%Calculate the error for use by the classification network
error=netoutput-output(:,3:l);
load('Class_net.mat');
%Simulate using the classification network
Class=Class_net(error);
```

Appendix C

Neural Network Weights

This appendix displays the weight matrices calculated by neural networks. The first section displays those calculated the driver model network while the second section displays the weights calculated for the classification network.

C.1 Driver Model NARX Network Weights

The first matrix displays the weights that the connect the input layer to the hidden layer; note that there are two sets of weights, one for the current input, and another for the time-delayed input. The second matrix displays the weights connecting the target outputs acting as feedback to the hidden layer; once again there are two sets of weights as a result of the time-delayed input. The third matrix contains the weights that connect the hidden layer to the output layer. The fourth matrix contains the hidden layer biases, while the fifth layer contains the output layer biases.

$$
\begin{bmatrix}
0.063431 & -1.6579 & 0.56944 & -0.35398 & 0.28763 & -0.32735 & -0.23292 & -0.24609 & 1.4722 & -0.52196 & 0.51981 & -0.055204 & 0.30619 & 0.074383 \\
-0.80491 & -3.0945 & 0.98677 & -0.19714 & 1.8459 & -0.092181 & 0.36425 & 0.058431 & 1.4499 & -1.2418 & 0.36373 & 0.4922 & -0.11479 & -0.13945 \\
-0.46826 & 5.2848 & -0.54876 & -0.62593 & 1.7431 & -1.7592 & -0.28097 & 0.59844 & -2.4078 & 0.67874 & 2.772 & 2.7466 & 1.9841 & -0.1002 \\
-1.7271 & -1.4765 & -0.74862 & -1.1922 & 1.4922 & -0.13231 & 0.034068 & 1.6203 & 0.23091 & 0.65476 & 1.4371 & -0.74168 & 0.29142 & -0.11924 \\
0.0098652 & -0.0076296 & -0.024515 & 0.055789 & -0.087481 & 0.0020188 & 0.0015823 & -0.010553 & 0.011682 & 0.019567 & -0.031249 & 0.078882 & -0.0014488 & 0.00067619 \\
-1.9608 & -0.71074 & 0.11642 & -1.4745 & 1.9705 & 0.03782 & -0.053847 & -0.051121 & 2.1961 & -0.35779 & 0.3108 & -3.633 & 0.54632 & -0.64753 \\
-1.0042 & -4.0652 & -0.66758 & 1.9074 & 2.1238 & 1.4901 & 0.02214 & -0.68921 & 0.28019 & 1.0987 & -1.906 & -0.042424 & -1.003 & 0.42743 \\
0.0068227 & 0.026447 & 0.012752 & 0.005935 & -0.0011857 & 0.0010017 & -0.0024009 & -0.0067846 & -0.0017354 & -0.0090549 & -0.011739 & -0.0012353 & -0.0032472 & 0.0030169 \\
0.20778 & -0.44714 & -1.3133 & 2.0151 & -2.5195 & 0.12416 & 0.27389 & -0.26159 & 0.37542 & 1.1221 & -1.1771 & 2.374 & -0.063179 & -0.24486 \\
0.93771 & 0.19551 & -0.59127 & -0.24919 & 3.4272 & -0.54811 & 0.78918 & -0.64247 & 0.093028 & 0.41553 & 0.23519 & -3.7513 & 0.63012 & -0.35637 \\
-0.75014 & -2.5028 & 0.30229 & -1.3919 & 2.0647 & -0.24321 & 0.071564 & -0.14683 & 1.5015 & 0.18497 & 0.4669 & -2.5855 & -0.17011 & 0.029873 \\
-0.33632 & 0.95252 & -0.67082 & 0.12239 & 2.1966 & 0.76143 & 0.66901 & 0.62442 & -0.25511 & 0.71416 & -0.77501 & -1.7723 & -0.91146 & -0.68624 \\
0.3732 & 1.5508 & 1.2925 & -0.43086 & -0.30711 & 0.10572 & -2.1904 & -0.45704 & -4.2163 & -1.6676 & 0.11734 & 4.0154 & -0.45006 & 0.7666 \\
-0.088957 & 0.8256 & -0.60003 & 0.68516 & -0.44711 & 0.042761 & 0.14021 & 0.26851 & -0.78646 & 0.66806 & -0.74695 & 0.39071 & 0.047896 & -0.10062 \\
-0.045599 & 0.12007 & -0.058563 & 0.059972 & -0.17497 & 0.042045 & -0.0034566 & 0.062892 & -0.098134 & 0.063565 & -0.075773 & 0.14475 & -0.037022 & 0.0067891
\end{bmatrix}
$$

$$\begin{bmatrix}
1.1233 & -1.495 & 1.6507 & -1.0349 & 0.19595 & -0.59902 \\
-7.3099 & 0.23549 & -3.7109 & 3.8732 & 0.034451 & 4.9377 \\
-3.5843 & -1.6451 & -0.46341 & -2.427 & -0.67276 & 0.34096 \\
-3.1047 & 5.143 & -1.3193 & 1.7067 & -4.7023 & 1.7829 \\
0.46435 & 0.019927 & -0.051561 & -0.15457 & 0.009323 & 0.0031842 \\
-6.2458 & 0.86217 & -0.35976 & 2.7225 & 0.44349 & -4.0117 \\
-4.0842 & 1.9048 & 1.3463 & -2.2301 & -0.7313 & 1.0775 \\
-0.045634 & 0.17726 & 0.08027 & 5.4683e-005 & 0.033068 & -0.027749 \\
3.3255 & -0.22003 & -1.5662 & -2.7792 & -0.046829 & 0.096868 \\
-10.4391 & 0.053746 & 1.485 & 9.5949 & 1.5678 & 1.7972 \\
-0.59142 & 1.3412 & -2.692 & 2.8128 & -1.4153 & 4.3584 \\
-7.5665 & 2.226 & 12.5959 & 8.0584 & -0.67568 & -9.8119 \\
-4.4322 & -3.0533 & -1.0432 & 0.51664 & 2.1297 & 3.2661 \\
-0.87126 & 0.69384 & -4.9729 & 0.50349 & -0.56659 & 3.6906 \\
0.2997 & 0.34827 & 0.85446 & -0.28859 & -0.10339 & -0.20556
\end{bmatrix}$$

$$\begin{bmatrix} 0.011403 & 0.020687 & 0.0032139 & -0.0003804 & 3.3121 & -0.00052869 & 0.0026239 & -0.66041 & -0.093218 & -0.0052272 & -0.0012909 & -0.0029661 & 0.0012897 & -0.00031034 & 0.28184 \\ -0.10833 & 0.021391 & -0.0081246 & 0.10205 & 0.75541 & 0.0015572 & -0.008515 & 4.4091 & 0.0011926 & -0.028132 & 0.0035026 & 0.011927 & -0.0033119 & 0.0029706 & -0.24258 \\ 0.15143 & 0.25169 & 0.0010438 & -0.0047328 & -0.053633 & 0.00056062 & 0.00037404 & -0.38157 & 0.0004425 & 0.046429 & -0.00047979 & 0.098839 & -4.5022e-005 & -0.163351.2762 & \end{bmatrix}$$

$$\begin{bmatrix} -2.1775 \\ -3.6441 \\ 0.56235 \\ -0.032586 \\ 0.083431 \\ 5.9626 \\ -1.611 \\ -0.16966 \\ 1.9133 \\ -2.1529 \\ -1.868 \\ -0.45744 \\ -1.4949 \\ -0.6845 \\ 0.40505 \end{bmatrix}$$

$$\begin{bmatrix} -0.34084 \\ 0.59173 \\ 0.037431 \end{bmatrix}$$

C.2 Learning Vector Quantization Classification Network Weights

The first matrix contains the weights for the self organizing map. The second matrix contains the weights for the output layer.

$$\begin{bmatrix}
0.0018105 & 0.00047161 & 0.00050199 \\
0.019023 & 0.033874 & 0.0082657 \\
0.021561 & 0.033434 & 0.012187 \\
0.017895 & 0.034978 & 0.0098261 \\
0.020637 & 0.032699 & 0.0088568 \\
0.019923 & 0.033001 & 0.0091834 \\
0.020942 & 0.033494 & 0.010029 \\
0.019398 & 0.033701 & 0.0084037 \\
0.017882 & 0.032959 & 0.0076943 \\
0.018924 & 0.034867 & 0.0097376 \\
0.021113 & 0.032659 & 0.011337 \\
0.019951 & 0.032679 & 0.0096178 \\
0.018823 & 0.032918 & 0.0082023 \\
0.017852 & 0.033532 & 0.0091547 \\
0.020043 & 0.032659 & 0.0097307 \\
0.019863 & 0.033028 & 0.0092399 \\
0.019292 & 0.032701 & 0.0075479 \\
0.019523 & 0.033463 & 0.01015 \\
0.020061 & 0.033372 & 0.0090213 \\
0.02081 & 0.032462 & 0.009047 \\
0.0013736 & 0.0032768 & 0.00091852 \\
0.0043192 & 0.01787 & 0.0022095 \\
0.006351 & 0.007639 & 0.0029926 \\
0.0028729 & 0.034441 & 0.00096939 \\
0.0028924 & 0.0036826 & 0.012971 \\
0.0041653 & 0.0045081 & 0.0016133 \\
0.022366 & 0.030866 & 0.007674
\end{bmatrix}$$

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

Vita Auctoris

André Levesque was born in Sault Sainte Marie, Ontario, Canada, in 1987 where he attended Notre-Dame-des-Grands-Lacs highschool from 2001 to 2005. From there André went on to the University of Windsor where he obtained his Honours Bachelor of Applied Science in Mechanical Engineering with Automotive Option in 2009. He anticipates obtaining his Master of Applied Science in Mechanical Engineering in January 2012.