An Evaluation of Online Reinforcement Learning Neuro-Fuzzy Traffic Signal Controllers

by

MOUDUD HASAN

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
(Civil Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

March 2006

© Moudud Hasan, 2006
Abstract

The work related to the development and evaluation of an online reinforcement learning neuro-fuzzy traffic signal controller undertaken at the University of British Columbia has been presented in this thesis. The main objective of the initiative was to advance the functionality of an earlier design developed at the same research facility which has been presented in the undergraduate thesis of Denesh Pohar. The original code has now been modified to control traffic movements at a standard four-leg arterial intersection. Its robustness was tested considering a range of traffic volume scenarios on the intersecting roads and simulated over a 90-minute period in Vissim traffic simulation software. Furthermore, the performance was compared to the operations of two existing signal control strategies, the actuated and modified FUSICO controls, which were also simulated under an identical set of conditions. Results suggest some positive changes in the intersection performance with the implementation of the online RLFNN control.
## Table of Contents

Abstract............................................................................................................ii  
Table of Contents ...............................................................................................iii  
List of Tables ......................................................................................................v  
List of Figures .....................................................................................................vi  
List of Symbols and Abbreviations.....................................................................vii  
Acknowledgement ...............................................................................................viii

1. Introduction.....................................................................................................1  
   1.1 Objective of this Research .......................................................................2  
   1.2 Structure of this Thesis ...........................................................................4

2. Literature Review ..........................................................................................5  
   2.1 Existing Traffic Signal Control Systems ................................................5  
      2.1.1 Pre-timed Control ...........................................................................5  
      2.1.2 Semi-Actuated Control ...................................................................6  
      2.1.3 Fully-Actuated Control ....................................................................7  
      2.1.4 Logic Based Control .......................................................................9  
   2.2 Fuzzy Logic Based Control .....................................................................10  
      2.2.1 Origin of Fuzzy Concepts ................................................................10  
      2.2.2 Fuzzy Set Theory ...........................................................................11  
      2.2.3 Membership Functions ....................................................................12  
      2.2.4 Fuzzy Set Operators .......................................................................16  
      2.2.5 Fuzzy Applications .........................................................................19  
      2.2.6 Fuzzy Expert Control Systems .......................................................20  
      2.2.7 A Fuzzy Signal Control Example ....................................................26  
   2.3 Neural Networks and Artificial Intelligence .............................................27  
      2.3.1 Performance Learning in the ANN ..................................................28  
      2.3.2 Back Propagation Algorithm .........................................................30  
   2.4 Reinforcement Learning ..........................................................................31  
   2.5 Neuro-Fuzzy Systems and Reinforcement Learning ............................35  
      2.5.1 Neuro-Fuzzy Systems ....................................................................37  
      2.5.2 Learning in Neuro-Fuzzy Systems ..................................................39  
   2.6 Reinforcement Learning Neuro-Fuzzy Control ........................................43  
      2.6.1 Input Term Nodes ..........................................................................44  
      2.6.2 Fuzzy Control System ....................................................................44  
      2.6.3 Fuzzy Reinforcement Predictor ......................................................44  
      2.6.4 Reinforcement Gradient Approximation and SAM .......................44  
      2.6.5 Reinforcement Learning .................................................................45  
   2.7 FUSICO and Neuro-Fuzzy Signal Controllers .......................................46  
      2.7.1 The Fuzzy Variables .......................................................................48  
      2.7.2 The FUSICO Rulebase ....................................................................50

3. Methodology ..................................................................................................53  
   3.1 The Investigation Approach .....................................................................54  
   3.2 The Reinforcement Learning Neuro-Fuzzy Signal Control .......................56  
      3.2.1 Traffic Simulator and Vehicle Detectors .........................................58  
      3.2.2 Queue Counter ...............................................................................58  
      3.2.3 Fuzzy Control System ....................................................................59
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.4</td>
<td>Delay Computation</td>
<td>60</td>
</tr>
<tr>
<td>3.2.5</td>
<td>Reinforcement History</td>
<td>60</td>
</tr>
<tr>
<td>3.2.6</td>
<td>Noise Generator</td>
<td>60</td>
</tr>
<tr>
<td>3.2.7</td>
<td>Reinforcement Gradient Approximation</td>
<td>61</td>
</tr>
<tr>
<td>3.2.8</td>
<td>Stochastic Action Modifier</td>
<td>61</td>
</tr>
<tr>
<td>3.3</td>
<td>Implementation of the Proposed Control</td>
<td>63</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Signal Timing Design</td>
<td>64</td>
</tr>
<tr>
<td>3.4</td>
<td>Alternative Signal Control Systems for Comparison</td>
<td>66</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Modified FUSICO Controller</td>
<td>67</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Fully-Actuated Controller</td>
<td>68</td>
</tr>
<tr>
<td>3.5</td>
<td>Model Development in Vissim</td>
<td>71</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Traffic Flow and Human Behaviour Model in Vissim</td>
<td>72</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Intersection Model Coding in Vissim</td>
<td>73</td>
</tr>
<tr>
<td>3.5.3</td>
<td>Simulation Traffic</td>
<td>74</td>
</tr>
<tr>
<td>3.5.4</td>
<td>Signal Controller Setup</td>
<td>75</td>
</tr>
<tr>
<td>3.5.5</td>
<td>VAP Signal Control Logic</td>
<td>75</td>
</tr>
<tr>
<td>3.5.6</td>
<td>Simulation Settings in Vissim</td>
<td>76</td>
</tr>
<tr>
<td>3.5.7</td>
<td>Measures of Effectiveness</td>
<td>81</td>
</tr>
<tr>
<td>3.5.8</td>
<td>Evaluation Files and Results</td>
<td>84</td>
</tr>
<tr>
<td>4.</td>
<td>Experimental Results</td>
<td>86</td>
</tr>
<tr>
<td>4.1</td>
<td>Results</td>
<td>86</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Influence of Traffic Flow on Controller Performance</td>
<td>87</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Influence of Simulation Length on Controller Performance</td>
<td>94</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Comparison of the Signal Control Options</td>
<td>100</td>
</tr>
<tr>
<td>5.</td>
<td>Conclusions</td>
<td>103</td>
</tr>
<tr>
<td>5.1</td>
<td>Summary of Findings</td>
<td>103</td>
</tr>
<tr>
<td>5.2</td>
<td>General Discussions</td>
<td>108</td>
</tr>
<tr>
<td>5.3</td>
<td>The Proposed Signal Control Logic</td>
<td>112</td>
</tr>
<tr>
<td>5.4</td>
<td>Recommendations for Future Research</td>
<td>114</td>
</tr>
<tr>
<td>5.5</td>
<td>Conclusions</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td>List of References</td>
<td>117</td>
</tr>
</tbody>
</table>
List of Tables

Table 2.1: Parameters defining the initial linguistic functions..........................50
Table 2.2: The original FUSICO rulebase .........................................................50
Table 2.3: The modified FUSICO rulebase .......................................................51
Table 3.1: Traffic flow conditions used for simulation......................................78
Table 4.1: Average total delay under different traffic flow conditions.................87
Table 4.2: Average stopped delay under different traffic flow conditions.............89
Table 4.3: Average number of stops under different traffic flow conditions..........90
Table 4.4: Average queue under different traffic flow conditions........................91
Table 4.5: Average travel time under different traffic flow conditions................93
Table 4.6: Average total delay under different simulation durations....................96
Table 4.7: Average stopped delay under different simulation durations...............97
Table 4.8: Average number of stops under different simulation durations............97
Table 4.9: Average number of queued vehicles under different simulation durations.98
Table 4.10: Average travel time under different simulation durations..................99
Table 4.11: Comparison of the MOE values with respect to the proposed control......101
List of Figures

Figure 2.1: Semi-actuated signal control (detectors on minor approaches only) ........................................... 7
Figure 2.2: Various types of vehicle detectors ........................................................................................................ 8
Figure 2.3: Definition of three linguistic values of fuzzy variable Speed ................................................................. 12
Figure 2.4: Shapes of different membership functions and their parameters ......................................................... 16
Figure 2.5: Fuzzy set operations on two Gaussian membership functions ............................................................. 19
Figure 2.6: Functional stages in a fuzzy traffic signal controller ............................................................................. 20
Figure 2.7: Basic architecture of a FCS ..................................................................................................................... 21
Figure 2.8: Fuzzy reasoning process with two input variables .................................................................................. 23
Figure 2.9: Fuzzy inference in a rulebase of two rules ............................................................................................ 27
Figure 2.10: Schematic diagram showing the resemblance ..................................................................................... 28
Figure 2.11: The basic training environment in a neural network ............................................................................ 32
Figure 2.12: A reinforcement-learning network with a critic .................................................................................. 34
Figure 2.13: The simplified architecture of a five-layer neuro-fuzzy system .......................................................... 36
Figure 2.14: Graphical representation of the initial linguistic functions ................................................................. 49

Figure 3.1: Steps involved under the current methodology ....................................................................................... 55
Figure 3.2: Layout of the intersection model setup in Vissim ................................................................................... 56
Figure 3.3: The functional blocks and signals within the proposed RLFNN controller ........................................... 57
Figure 3.4: The VISVAP algorithm for the RLFNN controller ................................................................................ 65
Figure 3.5: The VISVAP algorithm for the modified-FUSICO controller ................................................................. 68
Figure 3.6: Screenshot of intersection simulation in Vissim ..................................................................................... 79
Figure 3.7: Graphical representation of definitions of delay ..................................................................................... 83

Figure 4.1: Average total delay vs. varying traffic flow conditions ........................................................................... 88
Figure 4.2: Average stopped delay vs. varying traffic flow conditions ................................................................. 89
Figure 4.3: Average number of stops vs. varying traffic flow conditions ............................................................... 91
Figure 4.4: Average queue vs. varying traffic flow conditions ................................................................................ 92
Figure 4.5: Average travel time vs. varying traffic flow conditions ......................................................................... 94
Figure 4.6: Average total delay vs. varying simulation durations ........................................................................... 96
Figure 4.7: Average stopped delay vs. varying simulation durations ................................................................. 97
Figure 4.8: Average number of stops vs. varying simulation durations ............................................................... 98
Figure 4.9: Average queue vs. varying simulation durations ................................................................................ 99
Figure 4.10: Average travel time vs. varying simulation durations ................................................................. 100
Figure 4.11: Improvements in average delay .......................................................................................................... 102
## List of Abbreviations

<table>
<thead>
<tr>
<th>Term</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>Metre</td>
</tr>
<tr>
<td>km</td>
<td>Kilometre</td>
</tr>
<tr>
<td>km/h</td>
<td>Kilometres per hour</td>
</tr>
<tr>
<td>s</td>
<td>Second</td>
</tr>
<tr>
<td>sec</td>
<td>Second</td>
</tr>
<tr>
<td>vph</td>
<td>Vehicles per hour</td>
</tr>
<tr>
<td>veh/hr</td>
<td>Vehicles per hour</td>
</tr>
<tr>
<td>v/c</td>
<td>Volume to Capacity ratio</td>
</tr>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>FCS</td>
<td>Fuzzy Control Systems</td>
</tr>
<tr>
<td>FRP</td>
<td>Fuzzy Reinforcement Predictor</td>
</tr>
<tr>
<td>LOS</td>
<td>Level of Service</td>
</tr>
<tr>
<td>MOE</td>
<td>Measures of Effectiveness</td>
</tr>
<tr>
<td>NN</td>
<td>Neural Network</td>
</tr>
<tr>
<td>RL</td>
<td>Reinforcement Learning</td>
</tr>
<tr>
<td>RLFNN</td>
<td>Reinforcement Learning Fuzzy Neural Network</td>
</tr>
<tr>
<td>SAM</td>
<td>Stochastic Action Modifier</td>
</tr>
<tr>
<td>VAP</td>
<td>Vehicle Actuated Program</td>
</tr>
</tbody>
</table>
Acknowledgement

The author of this thesis would like to extend heartfelt thanks to Dr. T. Sayed for his outstanding guidance and supervision throughout this research and during the preparation of thesis. At each and every discussion, he provided valuable ideas from his vast knowledge and experience. Whenever any resources and materials were needed, Professor Sayed would let his extensive cache without any hesitation. Dr. J. Jenkins also deserves special thanks for providing important directions by reviewing the thesis.

The author is also grateful to his family and friends for their support and inspiration which have made extraordinary contributions towards the completion of this work.
1. Introduction

Most arterial roads of urban areas are congested causing significant delays, increased fuel consumption, lower productivity and increased environmental pollution. While traffic management strategies may offer long-term hope for cities with congested roads, there is a need for short-term improvements as well. The conventional approach is to relieve congestion at signalized intersections by adjusting the signal timing plans and coordination as well as by means of additional lanes, and other physical measures. Regardless of significant costs involvement, physical countermeasures may not be practical beyond a certain point. Under these constraints, improving the efficiency of signal control systems appear as a desirable approach towards relieving traffic congestion at the signalized intersections. The research presented in this thesis describes works undertaken in the development and assessment of an adaptive signal control strategy which could operate under fluctuating traffic demands.

It is expected that signal control systems would perform more efficiently, if artificial intelligence could be incorporated into their design. This “intelligence” can be useful in terms of decision-making steps such as system interpretation, application of rules, selection of optimal actions, performance evaluation and self-adjustment for future operation. A number of recent studies, as referenced in this thesis, have indicated that an appropriate combination of neural networks and fuzzy concepts can lead to “artificially intelligent” control. Such systems have already been found to be efficient in various process control
tasks. However, limited research has focused on applying this relatively new concept to the field of traffic signal control.

A signal control strategy based on neuro-fuzzy concepts is presented in this thesis. The work is part of the on-going research conducted by the transportation engineering group at the University of British Columbia. The original design concept, introduced in the work of Pohar (1) at the University of British Columbia, was followed in developing the control logic presented in this thesis. The application of fuzzy-neural network principles and computing techniques also resembles the methodology used in a research performed at The University of Technology (HUT) in Helsinki (2, 3). However, the work presented here is different from the research conducted at HUT because of the application of a different reinforcement learning algorithm and its real-time self-adaptability. Moreover, the current signal controller displays a wider functionality and is applicable to a more general traffic environment. As will be discussed in the following chapters, the control system presented here is capable of self-adjusting on a real-time basis.

1.1 Objective of this Research

The main objective of this research was to advance the functionality of a previously introduced online reinforcement learning traffic signal controller which applies fuzzy logic-neural network concepts. Possibilities of a neuro-fuzzy signal control which would perform efficiently under varying traffic demands without any external aid was looked at in particular. A similar, but simpler type of controller had previously been designed at the
University of British Columbia (1). The main purpose was to develop an extended application, which would incorporate more traffic variables that are present in the actual traffic environment. However, the current application was limited to an isolated intersection, which is similar to the previous version, but incorporated more traffic variables such as two-way movements on a four-leg multilane intersection. The intersection model subject to the proposed control also accommodated all types of turning movements. The previous fuzzy rulebase was modified to accommodate two way movements on the four legs of the intersection and to include the protected left turn movements as well. To evaluate the control performance, higher traffic volumes were now selected than those used previously. Optimum locations of upstream vehicle detectors and other parameters were established from a prior sensitivity test as a part of this project. After a thorough revision of the earlier designs, the application of principles, generation of algorithms, and implementation in a simulation program were carried out for the new signal controller. To be able to conclude on the performance of the proposed controller, the system was judged against alternative signal control options available in the field. A fully vehicle-actuated controller and a neural network-fuzzy logic based controller were selected for comparison. The neural-fuzzy logic based controller was similar to the proposed control in all aspect, except that it did not include any reinforcement-learning feature. An identical set of traffic condition was replicated including the same intersection model in the simulation of these control alternatives. Computer simulation was carried out and the output data were extracted for necessary processing and statistical analysis. A number of simulation runs with different seeds were conducted to reach a statistically
valid observation for each set of variables. Tables and graphs were prepared for the presentation of results and observations.

1.2 Structure of this Thesis

This thesis is divided into five chapters including this introductory chapter which presents a brief discussion on the subject and purpose of this research. Chapter Two, titled Literature Review, summarizes a review of literature and research related to the subject of this thesis. The chapter also looks at theories, definitions and mathematics required to develop the proposed control logic. Details of the experimental setup are given in Chapter Three - Methodology. Various components of the proposed controller, as well as other controller alternatives selected for comparison purposes are featured. The chapter also offers description of the model development process in the simulation software, Vissim, and programming in VISVAP. A summary of results and subsequent discussion are presented in Chapter Four, titled Results and Discussion. Conclusions of this research and recommendations for future work in this area are outlined in Chapter Five - Conclusion. Finally, a list of references used in this thesis is provided at the end of this document.
2. Literature Review

For a better understanding of the signal control technique presented in this thesis, a brief introduction to the existing signal control systems is provided in this chapter. Also included in this chapter are the theories, mathematics and earlier works in the field of neuro-fuzzy control.

2.1 Existing Traffic Signal Control Systems

Based on the working principle and technology involved, signal controllers can be classified as:

- Pre-timed or fixed time controllers;
- Vehicle-actuated controllers of two types (semi or fully-actuated); and
- Advanced, logic-based signal controllers.

The proposed neuro-fuzzy control shares certain characteristics with existing vehicle-actuated controllers, therefore, fully-actuated and fuzzy-logic-based controllers were selected for the comparative evaluation of the proposed control system. In this research, design procedures related to the intersection model and in the implementation of the signal controller were based on the Highway Capacity Manual (4) and Manual on Uniform Traffic Control Devices (5) guidelines. A brief description on various types of signal control is presented next.

2.1.1 Pre-timed Control

This is the earliest, simplest and widely used type of traffic signal control system. In this type of control, phase sequences, their splits, inter-green times, cycle lengths, etc. are all
pre-defined based on historic traffic movement data for the subject intersection. This information is used to determine the optimum cycle time and then to distribute it among the conflicting movement groups. The fixed-time controller can also operate based on more than one timing plan for different periods of the day. Optimum performance is achieved only when the actual traffic volumes at an intersection match those used in the original signal timing design. Although pre-timed controllers do not respond to actual traffic demands, they are suitable for establishing a coordinated signal control system along an arterial corridor or over a road network. For coordinated signal systems, it is necessary that the intersections have an identical or half or multiples of the identical cycle length. This condition is easy to satisfy with fixed-time signal control systems. Moreover, it involves less work and resources to install and maintain the fixed-time control system; therefore, it has a cost advantage over the other types described in the following sections.

2.1.2 Semi-Actuated Control

The semi-actuated type of signal control system is suitable when a minor road intersects a major road that has significantly higher traffic volume. It minimizes disruption of the major road traffic as the signal phase for the minor road is activated only upon detection of vehicles on the minor approaches. Semi-actuated controllers can have additional signal switching conditions besides the simple vehicle detection. For example, a minimum number of vehicles at the minor approach may be required to trigger the signal switching logic. A typical four-leg intersection with semi-actuated control is shown in Figure 2.1.
2.1.3 Fully-Actuated Control

The fully-actuated controller is another common form of vehicle-actuated systems and actuation is possible from both intersecting roads of the intersection. In this case, all four approaches are equipped with vehicle sensors in order to detect approaching vehicles. This type of control is particularly suitable for major arterial intersections that have moderate to heavy traffic volumes on both intersecting streets.

Vehicle-actuated systems can be used for uninterrupted movement of public transit services (e.g. buses, trams or light rails). They can also ensure right-of-way priority for emergency vehicles. An early and unconditional priority is provided as soon as an approaching emergency vehicle is detected.
An Evaluation of Online Reinforcement Learning Neuro-Fuzzy Traffic Signal Controllers
Chapter 2  Literature Review

Figure 2.2: Various types of vehicle detectors.  
*Courtesy: International Road Dynamics (IRD) Inc.*
As shown in Figure 2.2, a wide range of detection technology is currently available, this includes:

- intrusive detectors (e.g. axle sensors, piezoelectric sensors, quartz sensors, inductance loops);
- on-pavement/temporary detectors (e.g. pneumatic road tubes, magnetic detectors, temporary inductance loops); and
- non-intrusive/roadside or overhead mountable detectors (e.g. infrared, laser, acoustic, fibre-optic or microwave sensors, and traffic cameras).

2.1.4 Logic-Based Control

Vehicle-actuated signal controllers have been in use for almost thirty years. However, their design and application have changed significantly over the past few years leading to the development of logic-based advanced control techniques. Controllers based on vehicle-actuation or advanced application of detection technology can be grouped into three generations, each characterizing a different type of application (6):

- the first generation - crisp logic based control, “gapping out” logic;
- the second generation - single objective network control (e.g. SCOOT, SCATS); and
- the third generation - multi-objective network control (e.g. SCOOT+, SCATS+, FUSICO).

The first two generations did not include any application of logic-based or advanced control concepts and operated based on rather simple detection technologies. The control
logic presented in this thesis is based on advancing one of the third generation logic-based control techniques, specifically the FUSICO system.

For the successful application of logic-based controllers, it is important to obtain accurate traffic volume information because the green signal duration is often kept flexible, and allowed to vary according to current traffic demand conditions. For instance, the FUSICO system and its application for control strategies uses traffic volume information for not only the lane group that has the green phase but also the approaches that are queued behind the red lights. Demand conditions on the competing signal groups are compared and then assigned with optimum amounts of green time thereby minimizing vehicular delays. The theories and concepts related to such advanced control systems are elaborated in the following sections.

2.2 Fuzzy Logic Based Control

2.2.1 Origin of Fuzzy Concepts

The concept of “Fuzzy Logic” was first introduced by professor Lotfi Askerzade Zadeh, at the University of California at Berkley in 1965 (7). It was originally not presented as a control methodology, rather as a way of processing data. In his article, Zadeh introduced fuzzy sets and membership functions and extended several related definitions of set-theoretic operations. However, this approach to set theory was not applied to control systems until the seventies due to early computing limitations. Zadeh reasoned that precise numeric information inputs are not always required, and there could be highly adaptive
control with fuzzy linguistic variables. If feedback controllers could be programmed to accept noisy and imprecise input, they would be much more effective and, perhaps, easier to implement. Fuzzy logic provides a simple way to arrive at a unique control decision based on vague, imprecise, or noisy input information. The approach to control problems mimics the human decision-making process, only much faster, and based on a predefined and limited set of rules (8). Fuzzy logic system is therefore applied as a problem-solving control methodology; it lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel PC or workstation-based data acquisition and control systems. It can be implemented in hardware, software, or a combination of both. Traffic signal control is one of the early applications of fuzzy control logic. However, limited research has been conducted so far in this particular field (9, 10, 11, 12, 13).

2.2.2 Fuzzy Set Theory

Classical set theory and Boolean algebra cannot define many terms that we commonly use to describe our perceptions about things. Fuzzy sets can provide specific, mathematical interpretations of such vague, fuzzy language terms (14). For example, "small amount", "tall", "very hot", "too cold", "very heavy", etc. are only a few of such vague descriptions. Fuzzy sets are generalizations of classical or crisp sets, in the sense that fuzzy sets may contain their elements partially, too, whereas elements of crisp sets either belong to the sets or do not (i.e., binary). In conventional mathematics, the above terms would need to be associated with distinct numerical values and their respective units of measurements. The
fuzzy variables that fuzzy systems deal with are called "linguistic variables", as they are closer to natural language terms. The weight of an object, temperature of a room, length of time, and queues in traffic congestions, as perceived by human judgment, are good examples of linguistic variables, when they are defined in fuzzy terms. "Linguistic values" are the values assumed by the linguistic variables. For example, the weight of an object, a linguistic variable, can be assigned the linguistic values - light, heavy, too heavy, etc. Similarly, the temperature of a room can be cold, hot, too hot, etc. Obviously, for each linguistic variable there can be a set of several linguistic values. Linguistic values do not have definite numeric values; they are defined over a range of numerical values. For example, vehicle speed when ranging between 40 to 70 mph may be regarded as "moderate" in fuzzy terms. This is also illustrated in Figure 2.3.

\[ \begin{align*} 40 & \quad 55 & \quad 70 & \quad \text{speed (mph)} \\ \mu_{\text{Slow}} & \quad \mu_{\text{Moderate}} & \quad \mu_{\text{Fast}} \end{align*} \]

Figure 2.3: Definition of three linguistic values of fuzzy variable Speed.

2.2.3 Membership Functions

A fuzzy set may contain its elements partially, whereas an element \( x \) of a crisp set either belongs to the set or does not. The following characteristic function of a crisp set \( A \) maps the element \( x \) of a given universal set \( U \) to 0 or 1.

\[ \chi_A : U \to \{0,1\} \] (2.1)

The characteristic function is defined as
The functions of crisp sets distinguish between members and non-members of the crisp set $A \subset U$ as binary systems (15).

A fuzzy set $S$ is formally denoted as $S = \{ (x, \mu_S(x)) | x \in U \}$ (2.3)

Every linguistic variable $x$ is associated with the set of the membership function.

In contrast with the crisp set, each element $x \in U$ is assigned a degree of membership in $S$ (7). The degree of membership is measured by a function

$$\mu_S : U \rightarrow [0,1]$$ (2.4)

This explains the major difference between Fuzzy and Boolean (i.e., standard) logic; that possible values of membership range from 0.0 to 1.0 (inclusive), not just 0 and 1. For example, the fuzzy truth value (FTV) of the statement, "Bill is tall", is 0.75, if Bill is 1.8 meters tall. The statement can be formally expressed as $m(\text{TALL}(Bill's \text{ height})) = 0.75$

where, $m$ is a membership function and is the function that would map 2.0 metres to an FTV of 0.75. A formal definition of a membership function can be stated as a function that maps each element of fuzzy set $A$ to the real interval $[0, 1]$, such that as $m(A(x))$ approaches the grade of membership for $x$ in $A$ increases. $U$ (i.e., the universe of discourse) is a crisp set, and here, the focus is restricted to $x \in U$. The membership function

$\mu_S(x) = 0$, when $x$ does not belong to $S$ at all, $\mu_S(x) = 1$ when $x$ belongs to $S$ totally, and $0 < \mu_S(x) < 1$ when $x$ belongs to $S$ partially. Membership functions can be simple, or incredibly complex. For example, a relatively simple membership function could be
Some commonly used types of membership functions are triangular, trapezoidal, Gaussian, generalized bell and sigmoidal membership functions. The triangular membership function is expressed in the following terms:

$$ m(Tall(x)) = \begin{cases} 
0, & x < 3 \\
\frac{x - 3}{2}, & 3 \leq x \leq 8 \\
1, & \text{otherwise} 
\end{cases} \quad (2.5) $$

Parameters $p_1$, $p_2$, and $p_3$ define the vertices of the membership triangle. The trapezoidal membership function can be expressed in the following terms:

$$ \mu(x; p_1, p_2, p_3) = \begin{cases} 
\frac{x - p_1}{p_2 - p_1}, & x \in [p_1, p_2] \\
\frac{x - p_2}{p_3 - p_2}, & x \in [p_2, p_3] \\
1, & x \in [p_3, p_4] \\
0, & \text{otherwise} 
\end{cases} \quad (2.6) $$

Parameters $p_1$, $p_2$, $p_3$, and $p_4$ define the four corners of the trapezoid. It may be noted that triangular or rectangular functions are the two forms of trapezoidal function characterized by the values of its four parameters. Both triangular and trapezoidal membership functions are popular due to their overall simplicity. However, these membership functions are, basically, step functions and discontinuous at the corner points (16), which limits the use of such step functions in several mathematical operations (e.g. calculus). Nonlinear Gaussian, Generalized Bell and Sigmoidal membership functions are smooth and do not
suffer from discontinuity. These are the functions considered in the methodology described in this thesis. The Gaussian membership function can be expressed in the following form (14):

$$\mu(x; c, \sigma) = e^{-\frac{(x-c)^2}{2\sigma^2}}$$ (2.8)

The parameters $c$ and $\sigma$ determine the center and width of the membership function.

The Generalized Bell membership function has the following form (16):

$$\mu(x; a, b, c) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^b}$$ (2.9)

The parameters $a$, $b$ and $c$ control the width, slope and center of the membership function respectively.

The Sigmoidal or logistic membership function is described as (16)

$$\mu(x; a, c) = \frac{1}{1 + e^{-a(x-c)}}$$ (2.10)

Parameters $[a; c]$ control the slope and $x$ coordinate of the crossover point $x = c$. The Sigmoidal membership function differs from Gaussian and Generalized Bell membership functions in that it is monotone. Depending on the sign of $a$, a Sigmoidal function is open left or right and, therefore, is appropriate for representing concepts such as “very large” or “very negative” (16). In Figure 2.4 shapes of various types of commonly used membership functions are shown.
2.2.4 Fuzzy Set Operators

Common mathematical operations for fuzzy sets are complement, intersection and union.

They all have several mathematical interpretations as shown in Figure 2.5. For further
information on these operations reference can be made to Yager and Filev (15), Klir and Yuan (16), and Zimmermann (17).

- **Complement Operation** The definition of fuzzy set complement is equivalent to the complement of a classical set. The complement $S^c$ of a fuzzy set $S$ has a membership function

  \[ \mu_{S^c}(x) = 1 - \mu_S(x) \]  

  Klir and Yuan (16) note that $\mu_{S^c}(x)$ may be interpreted not only as the degree to which $x$ belongs to $S^c$, but also as the degree to which $x$ does not belong to $S$. Similarly, $\mu_S(x)$ may also be interpreted as the degree to which $x$ does not belong to $S^c$.

- **Intersection Operation** Intersections of fuzzy sets can be computed using a number of different mathematical formulae. Some of these will be discussed here. Using the traditional fuzzy intersection operator, the Minimum (7), the intersection $S = \cap S_i$ of fuzzy sets $S_i$, $i = 1,2,...,n$, has a membership function

  \[ \mu_S = \min\{\mu_{S_1}, \mu_{S_2}, ..., \mu_{S_n}\} \]  

  where $\mu_{S_i}$ is the membership function of the set $S_i$, $i = 1,2,...,n$. The Minimum is also the intersection operator for classical sets. It is not smooth, and is sometimes replaced by its differentiable alternative, the Soft Minimum

  \[ \mu_S = \text{soft min}\{\mu_{S_1}, \mu_{S_2}, ..., \mu_{S_n}\} = \frac{\sum \mu_{S_i} e^{-k\mu_{S_i}}}{\sum e^{-k\mu_{S_i}}} \]  

  where $k$ is a parameter, the soft minimum approaches the minimum asymptotically as $k \to \infty$. If $k$ is small, the soft minimum gives unexpected results at small membership.
function values. The Product Combiner is also smooth, and it is sometimes preferred in literature because of its simplicity. This is expressed as

$$\mu_S = \prod \mu_{S_i}$$  \hspace{1cm} (2.14)

The product combiner takes all the membership function values into account, whereas the minimum and soft minimum ignore information. This difference is more significant, when there are more than just two membership function values to be intersected.

- **Union Operation** The fuzzy union operation also has important mathematical interpretations. Traditionally, the union was interpreted as the Maximum (7). Using the Maximum Combiner, the membership function for the union set $S = \bigcup S_i$ of fuzzy sets $S_i, i = 1,2,... n,$ has a membership function

$$\mu_S = \max \{\mu_{S_i}, \mu_{S_2}, ..., \mu_{S_n}\}$$  \hspace{1cm} (2.15)

where $\mu_{S_i}$ is the membership function of the set $S_i, i = 1,2,... n.$ Maximum is also the union operator for classical sets. Another possible alternative is the Sum Combiner.

$$\mu_S = \sum \mu_{S_i}$$  \hspace{1cm} (2.16)

The sum retains all information, whereas the maximum does not. As the sum of membership functions may exceed one, the summation operator is usually restricted to lie between zero and one by using the bounded sum (17), whose membership function is

$$\mu_S = \min \{1, \sum \mu_{S_i}\}$$  \hspace{1cm} (2.17)

or, the probabilistic sum, whose membership function for two sets is

$$\mu_S = \mu_{S_1} + \mu_{S_2} - \mu_{S_1}\mu_{S_2}$$  \hspace{1cm} (2.18)
Figure 2.5: Fuzzy set operations on two Gaussian membership functions A and B of a linguistic variable -
(a) Complement set; (b) Intersection set using the minimum rule; and (c) Union using the maximum combiner.

2.2.5 Fuzzy Applications

Fuzzy logic can be most readily applied to expert systems whose information is inherently fuzzy. For instance, doctors, lawyers, engineers can diagnose problems quicker, if the expert system they use to diagnose the problem lists a few fuzzy solutions. Another area in which the fuzzy logic is used is handwriting recognition. In Japan, complicated Kanji strokes are detected as they are being written. Applications of fuzzy logic have also been seen in areas such as cement kiln control and financial prediction/control.

Perhaps Pappis and Mamdani (9) were the first to propose the conceptual design of a fuzzy signal controller, which took place exactly a decade after the original proposition of fuzzy logic concept. The basic motivation behind the development of such controllers was to make the allocation of green signal time more efficient by avoiding the use of crisp control inputs. The various stages in a fuzzy logic based signal control system are illustrated in Figure 2.6.
Over the last two decades several researchers have contributed significantly to the conceptual development of fuzzy logic based traffic control systems. This includes the works of Nakatsuyama et al who applied fuzzy logic in the control of two junctions in 1983, Chui investigated network-wide controls in 1992, Sayers et al explored urgency based control in 1996, Nittymaki and Pursula proposed group-based control in 1997, in the same year, Landenfeld and Cremer developed fuzzy control strategies for oversaturated urban traffic networks, and Nittymaki and Kikuchi designed control of a pedestrian crossing signal, Nittymaki performed transit priority designs and Lee et al incident detection techniques both in 1998.

### 2.2.6 Fuzzy Expert Control Systems

Fuzzy control is regarded as the most practical and successful application of fuzzy theory. An obvious superiority of fuzzy control systems over traditional systems is their ability to
incorporate expert knowledge. The knowledge can be in the form of a rule base, where the rules are propositions of the form “if \(X\) is \(A\), then \(Y\) is \(B\)”; here \(A\) and \(B\) are two fuzzy sets.

In general, a fuzzy controller consists of the following key components which is also shown in Figure 2.7:

- Fuzzifier Module (Fuzzification);
- Fuzzy Inference Engine (Fuzzy Inference); and
- Defuzzifier Module (Defuzzification).

**Figure 2.7:** Basic architecture of a Fuzzy Control System.

- **Fuzzification Module** The fuzzification module obtains input quantities from the physical system under the control objective. These measurements are transformed into independent variables for appropriate input membership functions. Moreover, the Fuzzification process can construct or modify the membership functions by itself (structure learning) for use in the fuzzy control process. In an ongoing control process, membership functions are already constructed, and only fine-tuning of the functions can be carried out additionally during the control process.
• **Fuzzy Inference Engine**  The fuzzy inference engine evaluates the control rules stored in the fuzzy rule base. Fuzzy inference includes computation of rule firing strength, fuzzy implication and rule aggregation. The process is elaborately discussed in a number of literatures referenced in this thesis. The result of fuzzy inference is made up by one or several output fuzzy sets, whose membership functions have to be defuzzified in order for the ultimate control decision to be made. Each rule in the rule base consists of two parts in its *if-then* format. The *if* part is called the antecedent, and the *then* part is known as the consequent. Weights may also be assigned to the rules to imply additional importance. Several fuzzy sets may be combined using "and" (i.e., intersection) or "or" (i.e., union) operators in the antecedent and the consequent. This is useful when the system has more than one input or output variables.

Rule firing strength is calculated by utilizing the membership function values for fuzzy sets used in the rule antecedent. Based on the input quantities, membership functions can only yield values ranging between zero and one. One or more values are then compared according to the logic operator used in the antecedent part. For example, the *minimum* operator is used to select between the values combined with "and". The *maximum* operator can be used for an "or" combiner. Once the rule firing strength is calculated, a value between 0 and 1, the fuzzy set of the rule consequent is either clipped or scaled at a level specified by the rule firing strength. Only the remaining or scaled part of this membership function is used in the rule combination process. For
each rule there will be a transformed function, which are then combined as described in the following paragraph.

Fuzzy implication results in one fuzzy set from each of the rules used in the current control process. A union set of these fuzzy sets is generated using a suitable method. The resulting union set is used in the defuzzification process in order to obtain the desired control output.

![Fuzzy reasoning process with two input variables using Mamdani's minimum implication rule.](Image)

**Figure 2.8**: Fuzzy reasoning process with two input variables using Mamdani's minimum implication rule. *Courtesy: Neural Fuzzy Systems, Lin and Lee (18)*

- **Defuzzification Module** After the union set has been defined, it must be defuzzified to obtain a numerical value for the output or the control decision. Common defuzzification methods for a union of several membership functions include the Center of Area (COA) and the Mean of Maxima (MOM) methods, as well as their
modified alternatives. Different defuzzification methods are also shown in Figure 2.8 and explained briefly as follows.

The COA defuzzification method determines the centroid of the fuzzy union set area using the following equation:

\[ y' = \frac{\int_0 y \mu(y) dy}{\int_0 \mu(y) dy} \]  \hspace{1cm} (2.19)

where, \( y \) and \( \mu(y) \) describe the range of output values (on x-axis) and their membership function values (on y-axis), respectively. For the discrete membership functions the following equation may be used:

\[ y' = \frac{\sum y_i \mu(y_i)}{\sum \mu(y_i)} \]  \hspace{1cm} (2.20)

Zimmermann (17), Jang and Sun (19) and Kosko (20) proposed another approach, which is actually a variation of the original COA method. In their approach, called the Local Center of Area (LCOA), the centroid of the union set is expressed as a convex sum of the centroids of the consequent sets. The total output fuzzy set is not needed, instead, each rule is defuzzified separately. The defuzzified output is

\[ y' = \frac{\sum m_i w_i v_i y_i}{\sum m_i w_i v_i} \]  \hspace{1cm} (2.21)

where \( m_i \) is the weight of the individual rules, \( w_i \) is the rule’s firing strength, \( v_i \) is the set volume, and \( y_i \) is the centroid of the individual sets. A drawback of the LCOA defuzzification is that large sets are given more emphasis than small sets. The wider the fuzzy set, the more uncertainty it often possesses. Thin sets represent more predictable
information and they should be given more weight than wide sets. This problem can be mitigated by eliminating the volume term in Equation 2.21.

In the MOM defuzzification method, the defuzzified value is defined as the average between the smallest value and the largest value of y for which $\mu(y)$ reaches its maximum.

$$y' = \frac{\text{Infima}_M + \text{Suprema}_M}{2}$$  \hspace{1cm} (2.22)

$\text{Infima}_M$ and $\text{Suprema}_M$ are respectively the minimum and maximum values of y.

Originally, Berenji and Khedkar (21) proposed the Local Mean of Maxima (LMOM) method of defuzzification. This variation of the MOM method is applied to defuzzify each output fuzzy set separately, instead of applying the MOM to the union of the output sets. The output $y'$ is defined as the centroid of the set of those y for which $\mu(y)$ exceeds the rule firing strength:

$$y'_i = \frac{\text{Infima}_M + \text{Suprema}_M}{2}$$  \hspace{1cm} (2.23)

The output of the fuzzy system is a weighted average of the individual rule outputs, where the weights are the rule firing strengths $w_i$, and rule importance weights $m_i$ as follows:

$$y' = \frac{\sum m_i w_i y'_i}{\sum m_i w_i}$$  \hspace{1cm} (2.24)

The use of rule importance weights $m_i$ is optional.
2.2.7 A Fuzzy Signal Control Example

An example of a fuzzy control system is illustrated in Figure 2.9. The system has two input variables, one output variable and two rules.

![Fuzzy inference in a rule base of two rules](image)

The input variables

- $\textit{APP} = \text{the approaching traffic from the green direction within the detectors}$
- $\textit{QUE} = \text{the queue in the red direction within the detectors}$

The output variable

- $\textit{EXT} = \text{the green signal extension for the active approach}$
The rulebase

If \( APP \) is zero and \( QUE \) is a few, then \( EXT \) is zero \hspace{1cm} \text{(rule 1)}

If \( APP \) is medium and \( QUE \) is long, then \( EXT \) is short \hspace{1cm} \text{(rule 2)}

An input measurement pair \((x_1, x_2)\) of the approaching traffic and the queue is obtained at a given time. The membership function values, "zero\((x_1)\)" and "a few\((x_2)\)" in rule 1, and "medium\((x_1)\)" and "long\((x_2)\)" in rule 2, are computed. The rule firing strength of each rule depends on these membership function values. Outputs of the two rules are combined and defuzzified to produce a numerical output from the system. The fuzzy control is deterministic, such that a given set of inputs always result in a unique output. The same output is produced if the control logic and its fuzzy membership functions remain unchanged. However, adaptive controllers use a learning algorithm to allow stochastic behaviour. In such algorithms, the output of the controller changes by a random amount. This stochastic behaviour may lead to better control performance. A reinforcement learning algorithm is applied in the control system presented in this thesis.

2.3 Neural Networks and Artificial Intelligence

Artificial Neural Networks (ANN) have functional similarities to the nervous systems of intelligent forms of life. The similarity is characterized by the parallel data processing capabilities of biological nervous systems which are composed of intricate networks of neurons. Structures of these two systems are illustrated in Figure 2.10. The study of neural networks originated in the field of neurobiology but has certainly deviated in its current applications this being the field of intelligent control.
The evolution of neural networks was elaborated in the literature of Hertz et al (22) and Rumelhart and McClelland (23). Generally speaking, ANNs are structured networks of interconnected simple information processing elements called neurons or cells. Each connection of such networks can be assigned a strength that is expressed as a network parameter, or weight.

Neural networks are a wide class of dynamical systems that can interpret a system without a rule base or analytical knowledge about the system. Using a large data set, a network capable of predicting the behaviour of the system can be constructed. However, the performance of the model depends largely on the quality of the data, i.e., how well the data represents the actual behaviour of the system and the structure of the network.

### 2.3.1 Performance Learning in the ANN

Learning in neural networks may be supervised or unsupervised. Under Supervised Learning, the network is presented with an input value and a desired output value. A
learning algorithm then compares the NN's output with the target output, and modifies network weight parameters in a manner that will decrease the output error. A special type of supervised learning is Reinforcement Learning discussed in section 2.4. Under Unsupervised Learning, no target output is provided. Instead, the neural network classifies inputs according to certain features that they share. The network's Output Signal should correspond with the category of the input. In other words, the network should display some self-organization. An important NN architecture of this kind is Kohonen's self-organizing map. Feed-forward networks are the most commonly used neural networks such as the Multi-Layer Perceptron (MLP) network and the Radial Basis Function (RBF) network. This type of network consists of an input layer, zero or more hidden layers and an output layer. Each layer can have one or more cells linked to the immediate layers. The input layer receives the input variables \( x_1, x_2, \ldots, x_n \) from the physical system. In the first hidden layer, each cell receives weighted input variables. The hidden layer output is computed using the hidden layer activation function, which is generally the same for every cell of the layer. Expressions for the activation functions of MLP at input and output layers, respectively, are as follows:

\[
\begin{align*}
    z_j &= f\left( \sum_i (a_{ij} x_i) \right) \\
    v_k &= f\left( \sum_j (c_{kj} z_j) \right)
\end{align*}
\]

(2.25)  
(2.26)

where, \( z \) and \( v \) are the outputs from the input and output layers, respectively, expressed as functions of linear combinations of layer input quantities. Usually, the activation function must be differentiable and it must saturate at both extremes. The following are two
representative Gaussian functions successfully used as activation functions for the MLP networks:

**Sigmoidal function**

\[ f(x) = \left(1 + e^{-x}\right)^{-1} \]  

(2.27)

**Gaussian function**

\[ f(x) = e^{-\frac{x^2}{2}} \]  

(2.28)

Among the different types of functions, only Hyperbolic and Sigmoidal functions fulfill the requirements of differentiability and saturation criteria, thus they are the most popular.

In the last hidden layer, each cell is connected to output layer cells with a certain weight. The size of the output layer should be the same as the number of output variables in the system under control.

### 2.3.2 Back Propagation Algorithm

A **Multi-Layer Perceptron** can be trained using the back propagation algorithm or the generalized delta rule, which was proposed by Rumelhart and McClelland (23). The idea is to update the weights of the network using the gradient descend method. Minimization is done. The error function measures the difference between the network output and the desired value at each observation as a Euclidean distance as

\[ E = \frac{1}{2} ||v - d||^2 \]  

(2.29)

where,  

\[ E = \text{error function} \]

\[ v = \text{output from the current control, and} \]

\[ d = \text{desired or target output} \]

In the gradient descent approach, each weight parameter \( \omega \) can be updated as
\[ \omega(t + 1) = \omega(t) - \eta \frac{\partial E}{\partial \omega} \]  
\hspace{1cm} (2.30)

where,  
\[ \omega(t + 1) = \text{updated weight at time step } (t+1), \]
\[ \omega(t) = \text{existing weight at previous time step } t, \text{ and} \]
\[ \eta = \text{rate of learning, or the learning parameter} \]

However, this procedure was not applied in the methodology of this research and the fuzzy membership function parameters were adjusted by means of on-line reinforcement learning. This concept is discussed briefly in section 2.4.

2.4 Reinforcement Learning

Artificially intelligent control systems use a special feature called machine learning which enables fine-tuning of the control process to produce more desirable results. There are several approaches to machine learning, including:

- supervised learning;
- unsupervised learning; and
- reinforcement learning.

Supervised learning of neural-fuzzy control systems is possible only if arrays of input-output pairs are available from the earlier training. Supervised learning is possible based on examples provided by an external agent/supervisor. In interactive problems, it is often impossible to establish examples of desired behaviour that are both accurate and representative of all likely situations in which the agent has to act. The supervised learning is not suitable where learning is to be based on learning from interaction (24). The system can also learn to produce desired outputs without prior training data such as in the case of
unsupervised learning. The method is essentially empirical where a model is fit to observations. In unsupervised learning, a data set of input objects is first gathered which are then typically treated as a set of random variables. A joint density model is then built for the data set (24). In the real-world applications training data is not always available. In reinforcement learning, the controller has to learn behaviour through trial and error interactions with a dynamic environment. Therefore, this method is suitable for control problems where input-output data is not available. In reinforcement learning, a controller receives a feedback from the system indicating the performance of the preceding control action as illustrated in Figure 2.11. The controller's goal is to maximize reward over time, by producing an effective mapping of states to actions called "policy".

![Figure 2.11: The basic training environment in a neural network.](image)

*Courtesy: Neural Fuzzy Systems, Lin and Lee (18)*

Reinforcement learning is based on the knowledge of rewarding actions that had successful consequence (e.g., shorter queue lengths, reduced vehicular delay, etc). The objective is to strengthen or reinforce the preference towards those actions in the later stages of the
control process (e.g. credit assignment problem). If the performance feedback of a control action can only be received several time steps later, the system needs to make a prediction of the immediate performance.

The concept of reinforcement learning has evolved from the observations of animal learning process. The process now has extensive applications in the adaptive optimal control process. The history of reinforcement learning is considered to be divided into two stages. The first stage can be traced in the 1950's, when analytical models of animal learning were first developed. It was in the later stage which happened in the 1980's, when the modern concept of associative reinforcement learning was investigated. At this stage, the input patterns were associated with characteristic output patterns, based on reinforcement signals returned from the physical system. There are two types of reinforcement learning systems: actor-critic type learning, and Q-learning (18). An actor-critic system contains two subsystems; one is for choosing the optimal control action at each state (an actor), and the other for estimating the long-term utility of each state (a critic). This actor-critic structure of RL method as shown in Figure 2.12 was applied in this research so that the signal controller could produce better results.
In a neuro-fuzzy system, a fuzzy control system is the *actor*, and a neural network system works as the *critic*. The *actor* determines the control action, and the *critic* evaluates the performance of the action chosen by the fuzzy system. The *critic* participates in the learning process.

Q-learning is a relatively recent concept of RL algorithm that does not need a model of its environment and can be used on-line (24, 25). Q-learning algorithms work by estimating the values of state-action pairs. The quantity $Q(s,a)$ is defined to be the expected discounted sum of future payoffs obtained by taking action $a$ from state $s$ and following an optimal policy thereafter. After these values have been determined, the optimal action from any state is the one with the maximum Q-value. The process is initialized to arbitrary numbers, Q-values are then estimated based on experience in the following manner:

**Step 1.** From the current state $s$, an action $a$ is selected. Consequently, an immediate payoff $r$ is received, and used to determine the next state $s'$. 

![Figure 2.12: A reinforcement-learning network with a critic. Courtesy: Neural Fuzzy Systems, Lin and Lee (18)](image-url)
Step 2. $Q(s,a)$ is updated based upon this experience, such that small changes in $Q(s,a) = x[r + y \cdot \max Q(s',b) - Q(s,a)]$ where $x$ is the learning rate and $0 < y < 1$ is the discount factor.

Step 3. Repeat.

This algorithm converges to the correct Q-values with the probability one if the environment is stationary and is dependent on the current state and the action taken in it; called Markovian (Markovian Decision Process), a lookup table is used to store the Q-values, every state-action pair continues to be visited, and the learning rate is decreased accordingly over time. This exploration strategy does not specify which action to select at each step. In practice, a method for action, called the Boltzmann Distribution strategy, is usually chosen that will ensure sufficient exploration while still favoring actions with higher value estimates. Experiments with Q-learning agent have been done in the past with favorable results as presented in the literatures of Watkins (26), Barto et al (27), and Littman and Boyan (28), and. It is apparent that sufficient research has been done and even more underway to achieve the goal of "learning" artificial intelligence.

2.5 Neuro-Fuzzy Systems and Reinforcement Learning

The term neuro-fuzzy refers to a hybrid of fuzzy logic and artificial neural networks or simply a neuro-fuzzy system is a fuzzy inference system presented as a neural net. There are typically three or five interconnected layers which include the input and output layers plus one or more of the hidden layers. However, in a neuro-fuzzy system, all possible connections among the cells in between two adjacent layers of the neural network may not
be present. Therefore, in a neuro-fuzzy network, the number of connections is often smaller. Only those cells that form a rule are linked to each other and one can find out the rulebase by tracing the connections. Berenji and Khedkar (21) presented a simple architecture for a neuro-fuzzy system. An example of a neuro-fuzzy system is shown in Figure 2.13.

![Layer 1: Input Layer 2: Input Term Nodes Layer 3: Rulebase Layer 4: Output Term Nodes Layer 5: Defuzzification](image)

Figure 2.13: The simplified architecture of a five-layer neuro-fuzzy system.

A successful combination of a fuzzy inference system and a neural system offers several advantages over conventional systems, particularly in the process control and machine learning tasks. Fuzzy systems are flexible and lend themselves to human-like reasoning. On the other hand, few modeling and learning theories exist for fuzzy systems. A neural network is able to learn from data and to efficiently perform parallel processing, but functions as a black-box approach, and therefore is not easy to interpret. A combination of these systems may have both qualitative and quantitative interpretations and may avoid the
drawbacks of a solely fuzzy or neural approach. Enhancement of fuzzy rule processing functions with the parallel processing abilities of neural networks can result in high computational efficiency. The number of rules, the formulation of the rules, and the shapes of the membership functions affect the control performance of the system, and all of these can be tuned using a neural learning scheme. Although the fuzzy linguistic representation adds nothing to the modeling abilities of the network, it is still a significant contribution because it allows the designer to include expert knowledge and check if an automatically generated system structure is feasible (24). Thus, expert knowledge can be easily incorporated in a fuzzy system. In a neural network, expert knowledge can usually be used only in choosing the initial values of the network parameters. The functional layers of a neuro-fuzzy system and signals processed in them will be briefly discussed in the following section. This information has already been presented in the work of Pohar (1).

2.5.1 Neuro-Fuzzy Systems

The neuro-fuzzy system that will be discussed here has been applied in this research and is similar to the one included in the earlier research carried out by Pohar (1). The system has a five-layer FNN architecture as illustrated in Figure 2.14. Each of these individual layers and their computational features will be discussed here as follows:

- **Layer One – Input** This layer receives the system state inputs directly from the physical system as well as the control inputs from the control system. The input is arranged in the form of the following array: $x = [x_1, x_2, \ldots, x_n]^T$. The subscript $i$ denotes
the scalars in layer one. The layer has no functional operation other than transferring
the scalar quantities to the next layer.

- **Layer Two – Input Term Nodes** Each membership function is evaluated to determine
  the degree of membership of each input to its respective linguistic values. Each node
  in layer two evaluates a single input membership function for direct application in a
  rule antecedent. Each single input has a weight of unity. The node’s activation
  function is the membership function of the linguistic value in question. For the bell-
  shaped membership functions, the output is

\[
\tilde{x}_j = \exp\left( -\frac{(x_i - m_{2,j})^2}{\sigma_{2,j}^2} \right)
\]

where, \( \tilde{x}_j \) = node output for input \( x_i \);
\( m_{2,j} \) = centroid of the membership function; and
\( \sigma_{2,j} \) = width of the membership function.

- **Layer Three – Rulebase** The rule firing strengths are calculated in this layer based on
  the outputs of layer two. Every single node in this layer is assigned to evaluate the
  firing strength of an individual rule. The rule antecedents are combined using the
  “AND” operator and the output and the activation function is expressed as

\[
r_k = \min(\tilde{x}_{j_{k_1}}, \tilde{x}_{j_{k_2}}, \ldots, \tilde{x}_{j_{k_s}})
\]

where, \( \{ j_{k_1}, j_{k_2}, \ldots, j_{k_s} \} \) is the set of indices of layer two nodes that contribute to the
rule antecedent of rule \( k \).
Layer Four – Output Term Nodes

This layer combines the rule firing strengths of each rule that produces the same rule-consequent. The combination of the firing strength are done in the following manner as suggested by Lin (18):

\[ \tilde{u}_i = \sum_k r_k \]  

(2.33)

where, \( r_k \) = output of layer three.

Layer Five – Defuzzification

All output terms are defuzzified in this layer based on corresponding rule firing strengths and membership function parameters by a suitable method. In this case, the Centre of Area method is selected and output from this layer may be expressed as

\[ u = \frac{\sum_i m_{s_j} \sigma_{s_j} \tilde{u}_i}{\sum_i \sigma_{s_j} \tilde{u}_i} \]  

(2.34)

where, \( m_{s_j} \) = centre of output membership function; \( \sigma_{s_j} \) = width of output membership function; and \( \tilde{u}_i \) = firing strength.

2.5.2 Learning in Neuro-Fuzzy Systems

As previously noted in section 2.4, there are several neuro-fuzzy training algorithms, which include supervised, hybrid supervised/unsupervised, and reinforcement learning approaches (18). Learning algorithms have been proposed to perform simultaneous structure and parameter tuning for the system. The structure learning corresponds to fuzzy rulebase tuning, while parameter learning corresponds to fuzzy membership function...
training. Structure learning is most useful in the development of control systems in which an adequate rulebase may not be available. In this research, an existing fuzzy rulebase was applied and therefore, structure learning was not required for rulebase generation purposes. The fuzzy membership functions were trained using the on-line reinforcement learning algorithm based on the concepts proposed by Lin (13). The parameter training tasks for this reinforcement learning problem can be reduced to supervised learning problems which may then be solved using Lin's supervised learning FNN training algorithm. The back propagation-based supervised learning algorithm was applied and the details of the implementation of this algorithm are presented next.

- **Output Term Training** The learning process begins at the output layer with the objective of minimizing the difference between the actual system output \( u \) and the desired output \( u_d \). More specifically, the width and centroid parameter that define the output fuzzy membership functions are fine tuned in the process of data training. The error term, \( e \) is defined as the square of the difference in the two output values as follows:

\[
e = \frac{(u - u_d)^2}{2}
\]  

In order to minimize the error, the term is differentiated as in the gradient descend method, which then yields,

\[
\frac{\partial e}{\partial u} = (u - u_d)
\]  

The two parameters are then trained in the following manner:
Chapter 2 Literature Review

\[ m_{s_j}(t+1) = m_{s_j}(t) - \eta \frac{\partial e}{\partial m_{s_j}} \]

also expressed as,

\[ m_{s_j}(t+1) = m_{s_j}(t) - \eta \frac{\partial e}{\partial u} \frac{\partial u}{\partial m_{s_j}} \]  \hspace{1cm} (2.37)

\[ \sigma_{s_j}(t+1) = \sigma_{s_j}(t) - \eta \frac{\partial e}{\partial \sigma_{s_j}} \]

also expressed as,

\[ \sigma_{s_j}(t+1) = \sigma_{s_j}(t) - \eta \frac{\partial e}{\partial u} \frac{\partial u}{\partial \sigma_{s_j}} \]  \hspace{1cm} (2.38)

The value of \( \frac{\partial e}{\partial u} \) comes directly from Equation 2.36, \( \frac{\partial u}{\partial m_{s_j}} \) and \( \frac{\partial u}{\partial \sigma_{s_j}} \) are derived from Equation 2.34, and \( \eta \) is the selected learning rate. Therefore, at the end of each time step the quantities are updated based on these two relationships:

\[ m_{s_j}(t+1) = m_{s_j}(t) - \eta \frac{\partial e}{\partial u} \sum_i \frac{\sigma_{s_j} \bar{u}_i}{\sigma_{s_j} \bar{u}_i} \]  \hspace{1cm} (2.39)

\[ \sigma_{s_j}(t+1) = \sigma_{s_j}(t) - \eta \frac{\partial e}{\partial u} \left( m_{s_j} \sum_i \sigma_{s_j} \bar{u}_i - \sum_i m_{s_j} \sigma_{s_j} \bar{u}_i \right) \left( \sum_i \sigma_{s_j} \bar{u}_i \right)^2 \]  \hspace{1cm} (2.40)

Input term Training Parameters of the input membership functions are updated in a similar manner according to the following equations:

\[ m_{s_j}(t+1) = m_{s_j}(t) - \eta \frac{\partial e}{\partial m_{s_j}} \]

which may also be expressed as

\[ m_{s_j}(t+1) = m_{s_j}(t) - \eta \frac{\partial e}{\partial u} \sum_k \left( \sum_i \frac{\partial u}{\partial \bar{u}_i} \frac{\partial \bar{u}_i}{\partial \bar{x}_k} \right) \frac{\partial \bar{x}_k}{\partial m_{s_j}} \]  \hspace{1cm} (2.41)

\[ \sigma_{s_j}(t+1) = \sigma_{s_j}(t) - \eta \frac{\partial e}{\partial \sigma_{s_j}} \]

which may also be expressed as
\[ \sigma_{2,j}(t+1) = \sigma_{2,j}(t) - \eta \frac{\partial e}{\partial u} \sum_k \left( \sum_i \frac{\partial u}{\partial u_i} \frac{\partial \tilde{u}_i}{\partial r_k} \frac{\partial r_k}{\partial \tilde{x}_j} \right) \frac{\partial \tilde{x}_j}{\partial \sigma_{2,j}} \] (2.42)

The summation with index \( k \) is over all \( k \) rules that have input term \( j \) in its rule antecedent. Similarly, the summation with index \( l \) is over all \( l \) output terms that are rule consequents of rule \( k \). The partial derivatives on the right-hand side are in the process described earlier except \( \frac{\partial u}{\partial \tilde{u}_i} \) which needs to be computed by taking partial derivative of Equation 2.34 as follows:

\[ \frac{\partial u}{\partial \tilde{u}_i} = \frac{\sigma_{5,i} \left( m_{5,i} \sum_i \sigma_{5,i} \tilde{u}_{5,i} - \sum_i m_{5,i} \sigma_{5,i} \tilde{u}_{5,i} \right)}{\left( \sum_i \sigma_{5,i} \tilde{u}_{5,i} \right)^2} \] (2.43)

Also, the following can be derived based on Equations 2.33 and 2.34:

\[ \frac{\partial \tilde{u}_i}{\partial r_k} = \begin{cases} 1 \\ 0 \end{cases} \] (2.44)

The value is one if the output term \( l \) is a rule consequent of rule \( k \) or zero otherwise.

\[ \frac{\partial r_k}{\partial \tilde{x}_j} = \begin{cases} 1 \\ 0 \end{cases} \] (2.45)

The value is one if the input term \( j \) is the least of the inputs of rule \( k \).

Finally, the input term partial derivatives arrive from Equation 2.31 as follows:

\[ \frac{\partial \tilde{x}_j}{\partial m_{2,j}} = \frac{2(x_j - m_{2,j})}{\sigma_{2,j}} \] (2.46)

\[ \frac{\partial \tilde{x}_j}{\partial \sigma_{2,j}} = \frac{2(x_j - m_{2,j})^2}{\sigma_{2,j}^2} \] (2.47)
However, for the special bell-shaped membership functions, such as for the one-sided, "less than" and "greater than" functions, the partial derivatives need to be defined differently.

- For one sided functions: no change when input values < centroid, otherwise set to zero;
- For greater than functions: set to zero when input values < centroid, otherwise no change;
- For less than functions: sign change when input values < centroid, set to zero otherwise.

The online reinforcement learning neuro-fuzzy control will be presented next which involves the application of the theories and techniques presented so far in this chapter.

2.6 Reinforcement Learning Neuro-Fuzzy Control

The reinforcement learning fuzzy neural network presented by Lin is a good example of reinforcement learning neuro-fuzzy applications (13, 18). The multiple input, single output control system proposed by Lin has two separate neural networks tied together to perform with an actor-critic architecture. There is a fuzzy control system (FCS) performing as the actor and there is a fuzzy reinforcement predictor (FRP) which performs as the critic in the system. The various components of this system as well as their functions are described as follows.
2.6.1 Input Term Nodes

This includes layers one and two of a basic FNN and these are common for both the FCS and FRP in this case. Its function is to convert the control inputs $x$ into fuzzy input linguistic values $\bar{x}$ for use in the fuzzy inference system.

2.6.2 Fuzzy Control System

Tasks of fuzzy inference and defuzzification layers are carried out in this block. Therefore, similar to the functions of layers three to five it uses inputs $\bar{x}$ to determine the control action $u$. The rulebase is predefined in the form of expert knowledge.

2.6.3 Fuzzy Reinforcement Predictor

This block operates in the same manner as the FCS block but it produces a different final output which is the current predicted value $p(t)$ of the next external reinforcement signal $r(t+1)$.

2.6.4 Reinforcement Gradient Approximation and SAM

The main objective of reinforcement learning is to maximize the reinforcement signal $r$. In order to determine how to adjust the control output $u$ so that $r$ is maximized, the gradient of reinforcement is taken with respect to the output. This is expressed as

$$\nabla r = \frac{\partial r}{\partial u}$$  (2.48)
This gradient is not known in most cases and the system would have to approximate its value as described later in this section. On the other hand, the Stochastic Action Modifier (SAM) seeks to explore the input-output space by adjusting the control output $u$ by a Gaussian random variable. In a single output case, as described by Lin (13, 18),

$$\hat{u}(t) = N(u(t), \sigma(t))$$

(2.49)

where, $u =$ the original output and the mean of the Gaussian random variable $\hat{u}$, and $\sigma =$ standard deviation of the Gaussian random variable which changes inverse proportionately with the predicted reinforcement. Therefore, the control action is adjusted more when the reinforcement signal has a lower value indicating greater opportunities of improvement. On the other hand, less exploration would be necessary if the action already had a better reinforcement signal. The reinforcement gradient can be approximated after the reinforcement signal is received using the following relationship (18):

$$\frac{\partial r}{\partial u} = (r(t) - p(t-1))\left(\frac{\hat{u}(t-1)-u(t-1)}{\sigma(t-1)}\right)$$

(2.50)

### 2.6.5 Reinforcement Learning

The reinforcement learning task takes place in the system in three different ways which are as follows:

- **Fuzzy Control System Training**  The FCS is trained using the gradient approximation method described in the preceding section. When the system learns the influence of changes on the control output, the problem reduces to supervised parameter learning and this method is already discussed in section 2.4. The error gradient given in Equation 2.36 is replaced by the reinforcement gradient approximated using Equation 2.45.
2.50. Furthermore, the subtraction signs in Equations 2.37 to 2.40 are replaced by additions to maximize the reinforcement signal.

- **Fuzzy Reinforcement Predictor Training** The FRP is trained by tuning the input and output term membership function parameters so that the reinforcement prediction error is minimized. The error gradient is expressed as

\[
\frac{\partial e}{\partial p} = (p(t-1) - r(t))
\]  

(2.51)

The supervised learning methods are applied replacing the error gradient in Equation 2.36 by that in Equation 2.50.

- **Input Term Training** The first two layers of the FCS and FRP networks are common, and as discussed above each of their learning schemes attempts to adjust the input membership functions at the same time. To avoid this situation, the sum or the average of the two changes may be considered to be applied to the input membership functions.

### 2.7 FUSICO and Neuro-Fuzzy Signal Controllers

The term FUSICO is the abbreviation of FUzy Signal COntrol which is originally the name of a three year project (1996-1998) undertaken at the Helsinki University of Technology/Laboratory of Transportation Engineering and funded by the Finnish Academy to investigate the potential of fuzzy signal control for isolated intersections and to develop a fuzzy signal controller for field trials. At the research facility, the traffic micro-simulation package Helsinki University of Technology Simulator (HUTSIM) and the fuzzy signal controller FUSICO were developed which were used in the works of
Niittymäki (11, 12) and Bingham (2, 3). FUSICO contains fuzzy rules and membership functions, and it evaluates the rules using fuzzy set operations. Traffic flow inputs are received from the detectors, and based on this information, the control system can extend an active green phase by a few additional seconds as determined by the system. FUSICO is considered as a promising application of fuzzy logic based signal control which attempts to optimize the green time allocation for an isolated, signalized intersection. Its objective is based on an extension principle which determines the optimal green signal time, and is similar to the basic principle of the control method presented in this thesis.

The current work and that undertaken by Pohar (1) is similar in many aspects to the research carried out by Bingham (2) at the Helsinki University of Technology. All three are based on reinforcement learning neuro-fuzzy control concepts and apply the original FUSICO rulebase and variable definitions. However, the control system presented in the work of Bingham and that presented later by Pohar are different in many key aspects.

The system architectures applied in the two systems are different. Pohar employed the RL FNN architecture proposed by Lin (13, 18) whereas Bingham followed the Generalized Approximate Reasoning-based Intelligent Control (GARIC) presented by Berenji and Khedkar (21). Although both of these follows the actor-critic style control, the critic in GARIC is a two-layer artificial neural network which is a fuzzy neural network in the Lin’s system. However, both Bingham (2, 3) and Pohar (1) used tuning of membership function parameters which is considered as a parameter learning process. Structure-
learning was not performed by any of these systems. Furthermore, short simulation runs were used in the parameter learning process in Bingham’s controller which was an online continuous process in Pohar’s design (1). In addition to this, the stochastic modification function was a new feature incorporated into the subject controller. The fuzzy system definitions that were originally applied in the FUSICO controller were reapplied in the neuro-fuzzy signal controllers presented in the works of Bingham (2) and Pohar (1). The variable definitions, parameter definitions as well as the FUSICO rulebase are described in the following three subsections.

### 2.7.1 The Fuzzy Variables

Two separate fuzzy variables were used to describe the traffic demand conditions on the two cross streets of an intersection. The numbers of vehicles stored within a pre-defined distance on both the green and red signal approach of the intersection were recorded. This was done at every time step by the use of a pair of detectors placed approximately 70 metres apart on each travel lane. The FUSICO controller applies only two input variables, called – APP and QUE. The fuzzy input variable APP was defined as the number of vehicles within the paired detectors for the active signal group, i.e., green signal at that time step. Again, the crisp value QUE represented the number of stopped vehicles queued at the red signal at the same time step. The fuzzy output variable EXT represented the amount of green extension allocated by the system to an active phase. Definition of the fuzzy linguistic functions used in the proposed neural-fuzzy control algorithm for the two
input variables and the output fuzzy variable is given in Table 2.1 and illustrated in Figure 2.14.

![Graphical representation of the initial linguistic variable functions - APP (shown at the top), QUE (in the middle) and EXT (at the bottom).](image)

**Figure 2.14:** Graphical representation of the initial linguistic variable functions - *APP* (shown at the top), *QUE* (in the middle) and *EXT* (at the bottom).

The bell-shaped functions shown in Figure 2.14, were chosen to represent membership functions for each of the three fuzzy variables. The equations for the bell-shaped functions were given in Equation 2.9. However, to overcome limitations in the VAP coding which
cannot accommodate exponential functions; it was necessary to use equivalent Taylor polynomials in place of exact bell-shaped exponential functions.

### Table 2.1: Parameters defining the initial linguistic functions

<table>
<thead>
<tr>
<th>Fuzzy Variable</th>
<th>Linguistic Value</th>
<th>Center of function, ( m )</th>
<th>Width of function, ( \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>APP (input 1)</td>
<td>Zero</td>
<td>0.0</td>
<td>1.5</td>
</tr>
<tr>
<td></td>
<td>A few</td>
<td>3.0</td>
<td>1.5</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>6.0</td>
<td>1.5</td>
</tr>
<tr>
<td></td>
<td>Many</td>
<td>9.0</td>
<td>1.5</td>
</tr>
<tr>
<td>QUE (input 2)</td>
<td>A few</td>
<td>5.0</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>10.0</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Too many</td>
<td>15.0</td>
<td>3.0</td>
</tr>
<tr>
<td>EXT (output)</td>
<td>Zero</td>
<td>0.0</td>
<td>3.0</td>
</tr>
<tr>
<td></td>
<td>Short</td>
<td>3.0</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>6.0</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Long</td>
<td>9.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>

Note: numbers in the above table represent the number of vehicles counted and stored within a pair of detectors.

#### 2.7.2 The FUSICO Rulebase

The basic fuzzy variable definitions and rulebase were adopted from those applied in the FUSICO controller at the Helsinki University of Technology. The main reason was to avoid the requirements of validating any new rule or variable definitions. It is therefore important to include the fuzzy variables and the FUSICO Rulebase prior to introducing the proposed neuro-fuzzy signal controller. The fuzzy input and output membership functions have already been defined in section 2.2.3 and the rulebase applied in FUSICO is given in Table 2.2.

### Table 2.2: The original FUSICO rulebase

<table>
<thead>
<tr>
<th>After ( G_{\text{min}} ) (right after minimum green):</th>
<th>APP</th>
<th>QUE</th>
<th>empty</th>
<th>zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>Many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>APP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>empty</td>
<td></td>
<td>EXT: zero</td>
<td></td>
<td>EXT: medium</td>
<td></td>
<td>EXT: long</td>
<td></td>
</tr>
<tr>
<td></td>
<td>a few</td>
<td></td>
<td>EXT: short</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
After the first unit green extension:

<table>
<thead>
<tr>
<th>QUE1</th>
<th>APP→</th>
<th>empty</th>
<th>zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>EXT: zero</td>
<td>EXT: long</td>
<td>EXT: medium</td>
<td>EXT: short</td>
<td></td>
</tr>
</tbody>
</table>

After the second unit green extension:

<table>
<thead>
<tr>
<th>QUE1</th>
<th>APP→</th>
<th>empty</th>
<th>zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>EXT: zero</td>
<td>EXT: short</td>
<td></td>
<td>EXT: long</td>
<td></td>
</tr>
</tbody>
</table>

| LT medium |          | EXT: medium | EXT: long |

After the 3rd unit green extension:

<table>
<thead>
<tr>
<th>QUE1</th>
<th>APP→</th>
<th>Empty</th>
<th>Zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EXT: zero</td>
<td>EXT: long</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| LT a few | EXT: short |       |       |       |       |       |     |
| a few    | EXT: medium |       |       |       |       |       |     |
| LT medium |          |       |       |       |       |       |     |
| too long | EXT: zero |       |       |       |       |       |     |

After the 4th unit green extension:

<table>
<thead>
<tr>
<th>QUE1</th>
<th>APP→</th>
<th>Empty</th>
<th>Zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EXT: zero</td>
<td>EXT: long</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| LT a few | EXT: short |       |       |       |       |       |     |
| a few    |          | EXT: medium |       |       |       |       |     |
| too long | EXT: zero |       |       |       |       |       |     |

Note: MT = More Than, LT = Less Than

Some modifications to the original rulebase were necessary to ensure that at least one rule applied under any combination of inputs. The modified rulebase is presented in Table 2.3.

Bingham (2) demonstrated that with the original FUSICO rulebase, there could be input scenarios for which there would be no rules applicable directly.

**Table 2.3:** The modified FUSICO rulebase

<table>
<thead>
<tr>
<th>QUE1</th>
<th>APP→</th>
<th>empty</th>
<th>zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>EXT: zero</td>
<td>EXT: long</td>
<td>EXT: medium</td>
<td>EXT: short</td>
<td></td>
</tr>
</tbody>
</table>

After $G_{min}$ (right after minimum green):

<table>
<thead>
<tr>
<th>QUE1</th>
<th>APP→</th>
<th>empty</th>
<th>zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>EXT: zero</td>
<td>EXT: long</td>
<td>EXT: medium</td>
<td>EXT: short</td>
<td></td>
</tr>
</tbody>
</table>

After the first unit green extension:

<table>
<thead>
<tr>
<th>QUE1</th>
<th>APP→</th>
<th>empty</th>
<th>zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>EXT: zero</td>
<td>EXT: long</td>
<td>EXT: medium</td>
<td>EXT: short</td>
<td></td>
</tr>
</tbody>
</table>

| LT medium |          | EXT: medium | EXT: long |

51
After the second unit green extension:

<table>
<thead>
<tr>
<th>QUE1 APP</th>
<th>Empty</th>
<th>Zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td>empty</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LT medium</td>
<td>EXT: zero</td>
<td>EXT: short</td>
<td>EXT: medium</td>
<td>EXT: long</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After the third unit green extension:

<table>
<thead>
<tr>
<th>QUE1 APP</th>
<th>Empty</th>
<th>Zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td>empty</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LT a few</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LT medium</td>
<td>EXT: zero</td>
<td>EXT: short</td>
<td>EXT: medium</td>
<td>EXT: long</td>
<td></td>
<td></td>
</tr>
<tr>
<td>too long</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After the fourth unit green extension:

<table>
<thead>
<tr>
<th>QUE1 APP</th>
<th>Empty</th>
<th>Zero</th>
<th>a few</th>
<th>MT a few</th>
<th>medium</th>
<th>many</th>
</tr>
</thead>
<tbody>
<tr>
<td>empty</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LT a few</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a few</td>
<td>EXT: zero</td>
<td>EXT: medium</td>
<td>EXT: long</td>
<td>EXT: short</td>
<td></td>
<td></td>
</tr>
<tr>
<td>too long</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: MT = More Than, LT = Less Than
3. Methodology

The goal of this research was to establish a functional real-time adaptive signal control routine applying the fuzzy logic and artificial neural networking concepts. In order to conclude on the performance of the proposed control system, a comprehensive evaluation was necessary. For this purpose, the performance of the subject control was compared with two existing signal control techniques that also involve control logic in their operation albeit simpler than the proposed system. Vissim traffic simulation program was used to create a wide range of actual traffic conditions that may exist at an isolated four-leg intersection. This work was undertaken to advance an ongoing research activity at the University of British Columbia aimed at developing better signal control techniques. In this particular research, the domain for the controller application was limited to isolated multilane four-approach arterial intersections.

The program code for the proposed signal control was first established in VISVAP accommodating all necessary sub-routines. The code was then imported/incorporated into the main simulation program and implemented in a four-leg intersection. Performance of the controller was tested under different traffic flow conditions. The same conditions were reapplied in examining performances of the alternative controllers. Finally, evaluation of the proposed system was carried out based on outputs obtained at the end of simulation runs which represented a number of performance measures. The queuing conditions and traffic flow within the subject intersection model were also observed visually on the computer screen throughout the simulation runs.
The traffic simulation program, Vissim, described later in this chapter, was used for the implementation of the proposed signal control system. For the purpose of implementation and testing of the proposed control algorithm, traffic environment at an urban, arterial intersection with multiple lanes on all four approaches was simulated. Other than the traffic input variables of interest, all geometric and traffic flow parameters in Vissim were predefined with standard or default values before the signal control code was imported for simulation. A description of these will be presented later. For the purpose of a comprehensive evaluation, each of the alternative signal control programs were executed for a wide range of traffic flow conditions. Details of the steps involved in the entire process are given in the following paragraphs.

3.1 The Investigation Approach

A number of tasks were undertaken in order to fulfill the objective of this research. The key steps involved in the process of the development and evaluation of a real time self-refining fuzzy-neural signal control strategy are shown in Figure 3.1. A preliminary design of an adaptive neural-fuzzy signal controller was available from the previous research conducted at the University of British Columbia (1). The controller design implemented by Pohar (1) was reviewed, revised and modified to serve a typical four-leg intersection. Two-way traffic flow and all turning movements were included in the intersection model. Furthermore, the proposed control system was subjected to additional traffic flow scenarios to test the robustness of its performance. A couple of alternative signal control strategies were used for the purpose of comparison. Once the algorithm was established
for the proposed control, it was coded in VISVAP, a component application that was available in the Vissim package. The program code was generated in VAP (Vehicle Actuated Program) programming language which could be linked as external signal control logic in Vissim. Prior to its successful implementation in Vissim, the code underwent extensive iterations of trial-error-debugging procedures. Controller codes for the two alternative signal controllers were similarly generated in VISVAP and then implemented in Vissim.

Figure 3.1: Steps involved under the current methodology.
Besides the programming activities, an intersection model was setup in Vissim. The subject model included a standard, isolated, arterial intersection with four approaches and multiple lanes as shown in Figure 3.2. Different signal control logics were referenced as external signal control codes for the same model alternatively. A wide range of traffic flow scenarios were simulated for the subject model, under the operation each of the three controllers. A detailed description of the intersection configuration and traffic movement is presented in the following sections.

**Figure 3.2:** Layout of the intersection model used in Vissim (Vehicle detectors appear as small dark boxes)

### 3.2 The Reinforcement Learning Neuro-Fuzzy Signal Control

The online reinforcement learning neuro-fuzzy traffic signal controller proposed by Pohar (1) was modified to include additional intersection and traffic variables. An attempt
was made to apply the proposed system to a more general traffic environment and assess the performance with respect to other common types of signal control systems. Several other traffic flow conditions were examined and the results were subject to detailed statistical analysis in the current research. In order to incorporate additional traffic input variables; such as two-way movements on each road link, turning movements, etc., the signal control logic was slightly modified as well. The features and design of the current control system are presented in the following chapter of this thesis. The conceptual design which includes the functional blocks and signal communication within the control system is presented in Figure 3.3, and is briefly presented in the following paragraphs. It may be noted that the design is very similar to what was presented by Pohar (1).

![Figure 3.3: The functional blocks and signals within the proposed RL FNN controller. Courtesy: adapted from Pohar (1).]
3.2.1 Traffic Simulator and Vehicle Detectors

This functional block includes the entire physical system that simulates the actual environment, captures traffic flow and signal state inputs and also receives and implements the control decisions. An available traffic simulation and signal controller coding software package called Vissim was applied in this work. Vehicles entering at the upstream detector and leaving the stop bar detector are recorded at every time step. The detectors can also be used to obtain information on vehicular speed, delay, occupancy, etc., which are also useful in the control task. The signal that this block generates is an array of vehicle counts expressed as

\[ X(t) = \begin{bmatrix} n_{\text{in},1} & \ldots & n_{\text{in},N} \\ n_{\text{out},1} & \ldots & n_{\text{out},N} \end{bmatrix} \]  

(3.1)

where the subscript \( in \) represents the number of vehicles arriving at the upstream detector and \( out \) represents the number of vehicles that leave the stop bar detector at a certain time step. The numbers \( l \) to \( N \) are used to identify the approaches. Here, \( t \) denotes the corresponding time step.

3.2.2 Queue Counter

Vehicles captured between the upstream and downstream detectors at any given time step are the primary input variables in the current control system. This is calculated using the raw data obtained from the upstream and the stop bar detectors. The raw detector data is interpreted into the input variables in this block as

\[ x(t) = \begin{bmatrix} APP \\ QUE \end{bmatrix} \]  

(3.2)
where \( APP \) and \( QUE \) are the input variables for the signal control system defined in section 2.2.7. The length of queue is calculated in terms of numbers of vehicles according to the following equation:

\[
n(t) = n(t-1) + n_{in}(t) - n_{out}(t)
\]  

(3.3)

The equation takes accumulation of queues from preceding time steps into account, this is represented as \( n(t-1) \).

### 3.2.3 Fuzzy Control System

This block performs the tasks of layers one to five of the fuzzy neural network with the input signal received from the queue counter. The output it produces is the recommended control output, this is expressed as \( u(t) = [\text{EXT}] \)

(3.4)

where, the output \( \text{EXT} \) is the recommended amount of green signal extension, in seconds.

The block marked Q in Figure 3.3 simply performs mathematical quantization tasks so that the output always assumes real integer values. This was necessary because of the simulator program that was used in this work. Another key assumption in the implementation of the current neuro-fuzzy control system was to simplify the exponential bell-shaped fuzzy membership functions. The exponential function was approximated using the first three terms of Taylor’s expansion as given in Equation 3.5.

\[
\bar{x}_j = \exp \left[ - \left( \frac{x_i - m_{2,j}}{\sigma_{2,j}} \right)^2 \right] = \frac{1}{1 + \left( \frac{x_i - m_{2,j}}{\sigma_{2,j}} \right)^2 + \left( \frac{x_i - m_{2,j}}{\sigma_{2,j}} \right)^4}
\]

(3.5)
3.2.4 Delay Computation

Delay is computed based on the number of vehicles and the number of time steps that these vehicles have to wait in the queue. This is computed from the equation

$$\bar{d}(t) = \frac{\sum_{i=1}^{m} \sum_{a=1}^{n} d_{i,a}(t)}{\sum_{i=1}^{m} n_{i}(t)}$$

which actually is the average of vehicular delays $d_{i,a}(t)$ over all $m$ queued vehicles and $i$ approaches at time $t$. The negative value of average vehicular delay was selected by Pohar as the reinforcement signal for the subject signal control logic (1). Bingham also applied average vehicular delay to form the reinforcement signal for the neuro-fuzzy control system (2). It may be noted here that maximization of the reinforcement signal would mean minimization of delay as they are related as:

$$r(t) = -\bar{d}(t)$$

3.2.5 Reinforcement History

This block serves to store past reinforcement signals as well as original control decisions. It serves the noise generation or stochastic action modification and the reinforcement gradient approximation tasks. The input signals that flow into this node are therefore, $u(t)$ and $r(t)$ while the outputs are represented as $r(t_a)$ and $r(t_{a'})$.

3.2.6 Noise Generator

This block performs the stochastic action modification of the original control decision based on current and past reinforcement signals. A general description of the SAM process
was provided in section 2.6. It may be noted that the output signal from this block also passes through the quantization process to ensure real integer outputs.

### 3.2.7 Reinforcement Gradient Approximation

This functional block enables the system to know the effect of changes in control output on the reinforcement signals. This approximation is important in the initiation of the back propagation learning of the FCS and FRP modules. The reinforcement gradient approximation process has been described earlier in section 2.6.4.

### 3.2.8 Stochastic Action Modifier

It is the most important component of the Noise Generator block that seeks to explore the input-output space by modifying the control output \( u \) by a Gaussian random variable. In this work, a previously generated array of random numbers was provided to the control system. The random numbers were generated using the MATLAB function with a standard normal distribution. However, noise scale factors were applied to these random numbers before applying those to modify the control decision. The scaling factor, which is based on another new parameter called performance index, helps determine the desired degree of exploration of the action modification and this factor is calculated as

\[
\sigma(t) = \begin{cases} 
\frac{1 + \sigma_{\text{max}}(r')^2}{1+(r')^2} & \text{if } r' < 0 \\
\frac{1}{1+(r')^2 + (r')^2} & \text{otherwise} 
\end{cases}
\]

where, \( \sigma(t) \) = the output noise scale factor;
r' = delay-based performance index (described in the following paragraphs); and

\( \sigma_{\text{max}} \) = the maximum allowable noise scale factor (in the current work, 1.7).

The performance index \( r'(t) \) is estimated from the following matrix multiplication:

\[
\dot{r}'(t) = \beta (1 - \beta) \begin{bmatrix}
\dot{r}(t_u) - r(t_u) \\
\dot{r}(t_u) - \bar{r}(t) \\
\phi(\bar{r}(t))
\end{bmatrix}
\]

where, the first matrix corresponds to the weights applied to the three quantities in the second matrix. While the weight factor \( \beta \) is assumed to be 0.5, the quantities in the second matrix are respectively:

- change in reinforcement due to recommended control output,
- difference between the reinforcements due to recommended control output and mean reinforcement, and
- an average reinforcement performance index function, expressed as

\[
\phi(\bar{r}(t)) = \begin{cases} 
\frac{\phi_{\text{max}} - r_u}{r_u - (\bar{r}(t) + d_{\text{min}})} & \text{if } \bar{r}(t) < -d_{\text{min}} \\
\phi_{\text{max}} & \text{otherwise}
\end{cases}
\]

where, \( \phi_{\text{max}} \) = the maximum value of the function (in this case assumed as 2.5), and \( r_u = 90 \) and \( d_{\text{min}} \) is a time measurement assumed here to be 6.0.

It may be noted from Figure 3.3 that the outputs from this node also undergo the quantization function to ensure that the numbers are always real integers. This is a requirement imposed by the signal control simulator.
3.3 Implementation of the Proposed Control

For a better understanding of the proposed online neural-fuzzy signal controller performance, a variety of traffic flow conditions were simulated in Vissim. Performances of the signal control alternatives were reviewed after being simulated in Vissim under similar traffic conditions. An isolated intersection with four symmetrical approaches was designed with all standard geometric features. It accommodated all turning movements with dual through lanes and a left turn lane on each approach of the intersection. Actuation based protected left turn phase was provided for all control alternatives. For modeling simplicity, right turn on red was not permitted, and the turning vehicles had to share the curb lane with through traffic during the protected phase. It was also assumed that traffic volumes on two opposite directions on each of the intersecting streets would be the same under all test scenarios. Different traffic flow levels were simulated on each link to examine the traffic operation under the proposed control algorithm. For any total approach volume, it was assumed that twenty-percent of the vehicles would turn left, another twenty percent would turn right and the remaining sixty percent would travel straight through on each of the four legs. A pair of loop detectors was placed on each travel lane of all four approaches. In the case of the neural-fuzzy controllers implemented in this work, detectors were used to count the number of vehicles within a pair of detectors at every subsequent time step. These numbers were added up for each approach in order to estimate the signal demand on the competing approaches and to determine whether vehicles were approaching or stalled. Based on this up to date flow information, the proposed control algorithm would allocate green times to the competing signal groups.
Researchers have made attempts to enhance the performance of the FUSICO controller by adding adaptability features that would fine-tune the fuzzy-neural network parameters, in order to obtain minimum delays. The works undertaken by Bingham (2) and Pohar (1) are closely related to the objective of the work presented here. However, all previous applications were limited to simple and hypothetical intersection of two single-lane cross streets. Only one-way traffic movement on each of these links were assumed without any provisions for intersection turning movements. In this work, signal control system for a typical four-leg intersection was designed not only with the proposed reinforcement learning neuro-fuzzy system, but also with the modified FUSICO and vehicle actuated controllers. Various design aspect of the proposed and alternative signal control systems are discussed in the following sections.

3.3.1 Signal Timing Design

The algorithm for the proposed controller is presented in Figure 3.4. An earlier control algorithm developed at the University of British Columbia was revised and modified for the current application; it involved a more realistic level of traffic operations. The current design catered to the online reinforcement learning fuzzy-neural logic based control for typical four-leg multilane intersection configuration, and also accommodated all turning movements. The concepts and theories involved in the development process of this system are presented in chapter two of this thesis. The operational design is focused on the form of basic signal timing design and the final implementation in the traffic simulation program. The signal timing calculations and assumptions are outlined in the following
paragraphs. The signal operation code was generated in VAP language from the original graphical format in VISVAP.

![Flowchart Diagram]

**Figure 3.4:** The VISVAP algorithm for the real-time reinforcement learning neural fuzzy controller

The proposed signal control system was designed for an isolated intersection. One through protected phase was designed for traffic movements in the opposite directions on each road. Actuated protected left-turn phases were provided only when vehicles were detected...
on the left turn lanes. The right turning vehicles on the shared through lane also shared the same signal phase and no right turn was permitted during the red signal.

All-red clearance interval of one second after the amber signal of four seconds was provided for all protected phases. The through phases were defined with a minimum green duration of eight seconds and a maximum green of 45 seconds, and a three-second gap time was assumed. For the timing related computations, an average approach speed of 60 km/h was assumed. A pair of detectors was positioned on each travel lane of all four approaches of the intersection. The downstream detector was positioned at the stop bar, and the other detector was located 65 metres upstream. The detectors were configured to count the number of vehicles entering and leaving the area bounded by each pair of detectors at every time step during the simulation. Detectors were also placed at the beginning of each left turn storage bay in order to receive calls from the left-turning vehicles.

3.4 Alternative Signal Control Systems for Comparison

In the process of examining the performance of the proposed control system comprehensively, a comparative evaluation was important. Therefore, performances of two alternative signal control options were compared to that of the proposed signal controller. The two alternative logic based controllers selected and implemented were:

- neuro-fuzzy controller, without any online reinforcement learning (also referred to as the modified FUSICO); and
fully-actuated controller (without any neural-fuzzy application also not adaptive).

A brief discussion on the design and implementation of these controllers is given in the following sections.

3.4.1 Modified FUSICO Controller

This signal control logic was based on fuzzy-neural network concepts and green extension rules identical to the proposed system, but it lacked any real-time self-optimization schemes. It was in fact, a variation of the original FUSICO controller algorithm developed and tested at the Helsinki University of Technology, Laboratory of Transportation Engineering (2, 3, 11). In the current work, the original concept and rulebase of the fuzzy controller was adapted for its application in a general four-leg multi-lane intersection accommodating all turning movements. The algorithm for the modified FUSICO controller is shown in VISVAP format in Figure 3.5. The algorithm uses a rule base identical to that of the proposed reinforcement-learning neuro-fuzzy controller.

The modified FUSICO control has functional modules that are similar to the proposed control algorithm, but it does not include the steps including Reinforcement Gradient Approximation, Fuzzy Reinforcement Prediction and Stochastic Action Modification (SAM) blocks which were described in section 3.2.
3.4.2 Fully-Actuated Controller

Most of the controllers used on the roads today are pre-timed, semi or fully actuated. In this work, a fully-actuated controller was designed based on the methodology recommended by McShane and Roess (29), and Khisty and Lal (30). However, some parameters are discretionary, and the designer has to select those to suit the individual applications, geometric configurations and expected traffic conditions. Some of the design details for the basic actuated controller implemented in the current investigation will be

---

`Figure 3.5: The VISVAP algorithm for the neural fuzzy controller (modified FUSICO)`
outlined in the following paragraphs. As before, the signal code was generated in VAP language, in VISVAP.

In determining the signal phasing, protected phases were provided for through traffic movements and also for left turn movements when actually needed. Vehicles waiting in the storage lanes triggered protected left-turn phases. Like the other control types, an all-red clearance interval of one second preceded by an amber signal of four seconds was defined for all protected phases. The through phase lengths were defined with a minimum green duration of eight seconds, a maximum green of 45 seconds, and a gap time assumed to be three seconds. For the timing related design calculations, an average approach speed of 60 km/h was assumed. A single detector was positioned on each travel lane of all four approaches of the intersection. It was located at 40 metres upstream of the stop bar for each travel lane. These detectors were configured to detect vehicles approaching the intersection at every time step. Detectors were also placed at the beginning of each left turn storage lane to receive signal recalls for the left turn vehicles. In the following paragraph the basic design steps of a simple fully-actuated controller design are outlined.

- **Determination of Minimum Green, \( G_{\text{min}} \)** The strategy is to provide the minimum time required to clear the vehicles stored in the space between detector and stop line. The following equation can be used for its computation (7, 8):

\[
G_{\min} = t_{sd} + \left( \frac{A}{l + S_o} \right) \left( \frac{1}{q_s} \right)
\]

(3.11)

where,

- \( G_{\min} \) = initial green allocation, second,
- \( t_{sd} \) = startup delay, second [assumed as 4.0 second],
$A = \text{distance of the up-stream detector from the between the stop line, m},$

$S_0 = \text{average distance between the closest points of successive vehicles, m},$

$l = \text{average vehicle length, m } [l + S_0] \text{ may be assumed as 6 m},$ and

$q_s = \text{saturation flow, pcu/hr } [\text{can be assumed as 1800 pcphgl}].$

- **Allowable Gap, $h$** This is calculated as the time taken by a vehicle to reach the stop line from the detector; it is also the maximum time headway to retain the green for an approach. Generally, the value lies in the range of two to eight seconds. It is directly related to the position of the detector and the approach speed. Detector location is approximated to facilitate the calculation of $G_{\text{max}}$ and maximum allowable gap. The gap should be lower for approaches with higher magnitude of flow (2.0 to 3.0 seconds for saturation flow) and speed (30). The following equation can be used to calculate the gap time (29):

$$h = \frac{A}{v} \tag{3.12}$$

where,

$h = \text{allowable gap, second},$

$A = \text{distance between the stop line and the detector, m, and}$

$v = \text{average approach speed, m/s}.$

- **Maximum Green, $G_{\text{max}}$** It is the time maximum allowable green time for an approach since the moment of call from the competing approach (31). $G_{\text{max}}$ can be approximated based on optimum cycle length calculation. The green splits are calculated based on Webster’s formula for optimum cycle length, or by using the trial-and-error methodology described in the Highway Capacity Manual (4). The green times thus computed are multiplied by a factor of 1.25 or 1.50 to find the maximum greens (29).
Detector Location, $A$. This can be calculated from Equation 3.11 by approximating a desired minimum green time and measuring the approach speed. With the assumption of saturation flow of 1800 pcphgpl, a start up delay of four second and 7 m spacing between vehicles, detector distance from stop line is estimated using

$$A = (G_{\text{min}} - t_{sd}) \times q_s \times (l + S_o)$$  \hspace{1cm} (3.13)

Another form of Equation 3.2 can be used,

$$A = v \times h$$  \hspace{1cm} (3.14)

where, $v =$ average approach speed, m/s [say 60 km/h or 16.7 m/s], and $G_{\text{min}} =$ minimum green time desired, second [say 10 sec].

Finally, computation of inter-green times is similar to that for pre-timed operation, which is based on safety considerations (e.g. non-dilemma zone clearance and safe pedestrian crossing time).

### 3.5 Model Development in Vissim

Vissim 3.60 released by PTV AG Germany, is a time step and behaviour based, microscopic simulation software, and was used in this research. Vissim is a decision support tool for traffic engineers and transportation planners. The following features of this software were of particular interest in the context of this research (32, 33, 34):

- Various types of signal controllers, such as fixed-time controllers, vehicle actuated, pedestrian push-button or other logic (i.e. similar to NEMA) based control systems can be implemented in the Vissim traffic system.
- The simulation package is available with two additional tools, namely, VAP and VISVAP, that can be used for external signal state generation. Virtually any kind of signal controller logic, including SCATS, SCOOT, can be coded using VAP language and modeled or simulated in Vissim. VISVAP is a window based program for coding the control algorithm in a graphical flow-chart format. After checking the compilation VISVAP generates the program code in VAP language format.

- Prior to the final simulation runs, the controller design can be tested for performance with the aid generated traffic flow, or by initiating vehicle detectors manually. Triggering of the detectors is reported in macro files, which can be used for running identical test scenarios with alternative values of signal control parameters.

- Vissim can model various types of intersection layouts and control strategies. Signalized, non-signalized or mixed roundabouts, as well as junctions for multi-modal traffic flow including pedestrian activities.

Details on the intersection model development in Vissim will be presented in the following sections.

### 3.5.1 Traffic Flow and Human Behaviour Model in Vissim

Application of a reliable traffic flow model helps a traffic simulator generate a more realistic traffic simulation. Vissim uses a discrete, stochastic, time step based (simulation second) microscopic traffic flow model, with Driver-Vehicle-Units (DVU) as single entities. A psychophysical car following model, based on the original work of Wiedemann (35) and Sparmann (36) who did the lane change model, has been incorporated in Vissim.
The model outlines longitudinal vehicle movements associated with a rule-based algorithm for lateral movements (e.g. lane changing). In any traffic simulation, the question is how well it captures unpredictable interactions between drivers, vehicles, and the road environment. Longitudinal and lateral position of a vehicle in the stream, minimum space margins, gap acceptance, lane selection behaviour, sense of time, etc. are among the aspects addressed in the human behaviour models used in a simulation application. In the model originally developed by Wiedemann in 1974, a driver is assumed to be in either of the following possible driving modes at any given instant of time: free driving, approaching, following, and braking. For each mode, acceleration is described as a function of speed, speed difference, distance and the individual characteristics of driver and vehicle. The driver shifts from one mode to another, as he reaches a predefined threshold. This threshold is expressed as a combination of speed difference and distance between the two cars. To reflect the differences among a DVU group, stochastic distributions of speed and spacing thresholds are used. Vissim’s traffic simulator uses both longitudinal gap and lateral clearance on multilane roads to determine the driver actions of each vehicle. In addition, higher alertness of an approaching driver is assumed when a signalized intersection is only 100 metres ahead (37).

3.5.2 Intersection Model Coding in Vissim

The Network Editor toolbar available in Vissim main program window, allowed tracing and defining the desired road links, nodes and connectors to the necessary details such as number of lanes, lane widths, etc. Subsequent to this, desired hourly traffic flow, duration
of the flow, traffic composition, etc. were configured for each individual link. Default vehicle dimension and mechanical properties, driving rules, speed and acceleration distribution, etc., were maintained for the purpose of this experiment. Position and lengths of various link connector elements were carefully chosen, to ensure that all vehicles travel through the intersection and are captured by the measurement and detection devices at all time steps of the simulation. A pair of detectors were traced and configured for each lane of all four approaches of the desired intersection model. Loop detectors were placed on each of the approach lanes, and were defined with their corresponding signal groups, measurement instructions to match the configuration used in the signal control logic. Segments were defined by start and end points to obtain the delay based performance data from the travel time devices. Queue counters were placed at the stop bars on each approach lane of the intersection.

3.5.3 Simulation Traffic

A wide variety of vehicle class and their different sub-categories can be defined and modeled in Vissim. Cars, trucks, buses, trams, pedestrians and bikes are some examples. Vehicle properties can be created or modified for each of these classes using the Vehicle Editor tool. The vehicle properties are expressed in terms of suitable distributions. For instance, make year, mileage, weight, engine power, dwell time, desired speed and accelerations are usually defined by defining stochastic distributions. The Traffic Composition tool is used to describe the mix proportions of different vehicle classes for simulation, and their corresponding speed profiles. General configuration is set to “All
vehicle types" for the current work. Traffic volumes are entered for each individual link, on a per hour basis. Exact number of vehicles could also be generated in simulation runs but was not used in this case. Vissim is capable of producing different flow rates and vehicle composition at different time intervals of a simulation run. However, this feature was not selected in this experiment.

3.5.4 Signal Controller Setup

Various types of signal control options, including unsignalized, priority based, fixed-time or vehicle actuated controllers, and transit priority based control, etc. can be implemented in Vissim. For the logic-based controls, the program logic has to be coded, first, in VAP language, and then incorporated into Vissim. Signal groups, inter-green times, cycle lengths (pre-timed case only), minimum and maximum greens, gaps, inter-stages, etc., have to be defined either in the Signal Settings menu or using additional external codes (*.vap and *.pua). Separate calculations for the determination of cycle length and optimum phase splits are necessary as Vissim does not have any signal optimization capability.

3.5.5 VAP Signal Control Logic

Phase or stage based signal operation logic for signal controllers can be coded in VAP language using a component application called VISVAP. As already mentioned, VISVAP is a useful, Windows-based application, where the control algorithm can be coded in the form of simple flow-charts. That is advantageous in coding complex algorithms, and when iterative code modification is expected. Once the flowchart is established, VISVAP can
check, compile and generate the actual controller code in VAP language automatically.

Despite the obvious advantages in understanding the simple conventions and commands of VISVAP, the program lacks the capability of assuming exponential or logarithmic mathematical functions. Therefore, any exponential or logarithmic forms of equations have to be approximated in terms of equivalent polynomial forms.

### 3.5.6 Simulation Settings in Vissim

Vissim simulation parameters including *Simulation Time, Random Seed, Time Steps per Simulation Second, Rate of Simulation* are customized prior to simulation runs, in order to meet its specific needs. The most important of these is, perhaps, the duration of simulation. A longer duration of simulation may produce a result that is, statistically, more reliable as longer simulation represents a larger sample of DVUs. However, to reflect the dynamic and constantly varying nature of traffic conditions, it is inappropriate to simulate traffic operation for a long period with a constant set of physical conditions. Vissim does allow the input of different traffic flow scenarios for different time intervals within a single simulation period. It is possible to extract the evaluation measurements for the different time intervals corresponding to various traffic conditions. For evaluation purposes, simulation runs should be longer than the period of data extraction. A “warming up” of the road network or intersection model is necessary, as the empty network is gradually being filled up with specified flow levels of traffic, at the actuation of the simulator. Data extraction, right from the beginning of a simulation run, may capture the traffic conditions within a model, when it is empty or partially used. *Rate of Simulation* is used to select the
speed at which the simulation will run. This feature is useful in visually examining the operation of the system during the progression of a simulation run. *Time Steps per Simulation Second* parameter defines the number of times the vehicle position and other computations are performed by Vissim. The greater the number of steps, the smoother is the flow of vehicles in the simulated network. The *Random Seed* value is used to reproduce or modify the traffic arrival/platooning pattern in the simulation runs; the pattern does not change in successive simulation runs as long as the simulation seed is unchanged. This feature is important in conducting a meaningful comparative evaluation among a number of signal controllers tested for identical base conditions.

The three logic-based signal control alternatives were implemented and tested in Vissim by using the program codes that were generated in VISVAP. The control logics were incorporated into Vissim by referencing the respective code files as External Signal Control logics. A separate program code, called the Signal *Interchange Definition* file (*.pua), was also used with the main signal control codes for additional signal phasing configurations.

In order to assess the performance of the proposed control algorithm, a simulated dynamic traffic environment was created. Additional study was carried out to assess the relative performance of the new signal control logic with respect to its alternatives. As already noted, an isolated intersection accommodating all turning movements was used as the subject model for the implementation of the new controller. The intersection was designed
with four approaches, each accommodating two travel lanes, and a separate left turn lane. Adequate storage length and tapering for left turn lanes was provided for the turning vehicles. Detectors were placed at the beginning of each left turning bay for phase actuation purpose only.

A wide range of traffic flow scenarios was simulated to examine how robust was the controller operation. Traffic volumes, turning splits, durations of different traffic flow levels, etc., had to be defined and configured accordingly, in the model, before the simulation runs. Identical base conditions were applied to compare the operations of different signal control strategies. Traffic volumes had to be defined for each of the links individually, to match the traffic flow scenario under investigation. The flow conditions used in the simulator to examine the control algorithm are given in Table 3.1.

Table 3.1: Traffic flow conditions used for simulation

<table>
<thead>
<tr>
<th>Simulation Scenario</th>
<th>N/S Approach Volume (vph)</th>
<th>E/W Approach Volume (vph)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LT</td>
<td>ST</td>
</tr>
<tr>
<td>1.</td>
<td>100</td>
<td>300</td>
</tr>
<tr>
<td>2.</td>
<td>100</td>
<td>300</td>
</tr>
<tr>
<td>3.</td>
<td>200</td>
<td>600</td>
</tr>
<tr>
<td>4.</td>
<td>260</td>
<td>780</td>
</tr>
</tbody>
</table>

Note: LT = left turn; ST = straight through, and RT = right turn traffic volume

A mixed mode Traffic Composition was defined in the subject model for the signal controller simulation. Pedestrian movements or crosswalks were not considered in simulating the intersection operation. However, the minimum green timings were provided based on general safety considerations to address the minimum pedestrian crossing requirements. In order to determine the values of other parameters to be used in the Vissim
intersection model, a brief parametric study was conducted prior to final simulation. In this process, suitable locations of the detectors and measurement devices were also determined to finalize the intersection model. The intersection geometry and laning configuration used to simulate different traffic flow scenarios is shown in Figure 3.2 while the simulation screenshot is presented in Figure 3.6.

![Figure 3.6: Screenshot of intersection simulation in Vissim (plan view).](image)

For each approach lane of the subject intersection, a pair of detectors was used, one near or at the stop line, and the other upstream from it. At every time step, the number of vehicles
that entered over the upstream detector, and left from the stop bar detector was recorded. The difference of these two numbers was used to calculate the net number of vehicles that entered the area bounded by the detectors, at that time step. For the purpose of a satisfactory comparison among the signal control alternatives, it was necessary to ensure identical base conditions for all cases. To achieve this, the model established in Vissim for the proposed controller was reused for the applications of the other two options, and simulated with identical set of flow scenarios.

Vissim uses the Poisson distribution in modeling the vehicle arrival process and an arrival profile is exactly reproduced in different simulations, as long as the simulation seed remains unchanged. Simulation parameters including the seed need to be defined prior to each simulation run. This feature in Vissim is important in the comparison of alternative traffic operation options as it allows the exact reproduction of a traffic flow condition. Several preliminary simulation runs and measurements were carried out as a part of brief sensitivity test. The purpose was to determine the optimal values/ appropriate ranges of various parameters to be used in the Vissim model setup. With all these quantities being defined the base model was established which was applied to all three controller types under consideration. The brief study included determination of suitable positions of the travel time and delay measurement devices along different lanes, suitable range of traffic volumes to be used, proportionate volumes of various turning movements, duration of simulation, etc. A brief sensitivity study was undertaken to determine the optimal values
or appropriate ranges of all these quantities which formed the base conditions regardless of controller types in the final simulation.

3.5.7 Measures of Effectiveness

Several Measures of Effectiveness (MOE) can be used to assess the operational performance of an intersection under different signal control options. The same indicators can also be used in conducting a comparative evaluation of the alternative control strategies. Among the different tools for analyzing operational performance are: total delays, total travel time, queues, intersection overall level of service, number of vehicle throughput, number of stops, average approach speed, fuel consumption, emissions, etc. Delays resulting from inefficient intersection operation are probably the most important of these indicators, at least from the road user's point of view. Also, it is not appropriate to consider just a single Measure of Effectiveness as the most "appropriate" for all traffic study is, therefore, more than one measurement is recommended. Delay, with its different components, is an important feature, and it is correlated to some of the other measures, such as queue length, travel time, etc. Some of the commonly used measures in analyzing traffic conditions of signalized intersections were selected for evaluation purposes in this work. These MOEs are described briefly in the following paragraphs.

- **Travel Time Delay**  Delay is an important performance index, and is closely related to the operating traffic conditions at an intersection or a roadway segment. Delay increases the travel time proportionately, and has economic consequences. Other
disruptions to free-flow conditions may occur at roadway merge and weaving areas, as well as due to incidents or emergencies. It is relatively easier to quantify and obtain measurements of delay in the field. Travel time delay is defined as the difference between the actual time and the theoretical or average time required to cover a certain distance through the intersection. It can be expressed as

\[ \Delta t = \frac{x}{v} - t \]  

(3.5)

where,
- \( \Delta t \) = Travel time delay,
- \( x \) = distance that includes the intersection,
- \( v \) = average approach free flow speed, and
- \( t \) = actual time of traveling the distance, \( x \).

Travel time delay can also be seen as a combination of various types or components of delay – deceleration time while approaching an intersection control, stopped time in queue or red light, and start up or acceleration delay. These individual components of delay are illustrated in Figure 3.7.

Stopped time delay is defined as the time a vehicle is completely stopped or queued at an intersection. A vehicle may be stopped due to the control system, or congested traffic conditions. Approach delay includes the time lost as vehicles may decelerate from their ambient speed to a stop, stopped time, and also the time they take accelerating from the stop back to their ambient speed.
- **Queue length**  Length of queue at any given time is another useful measure of effectiveness of intersection operation. Usually, the 50th or 95th percentiles of queue length are used in evaluation purposes. That is particularly useful in determining when an intersection will begin to impede the discharge from neighbouring intersections.

- **Average Number of Stops**  The average number of stops made by each vehicle to clear an intersection is another good measure of effectiveness. Stop sign controlled intersections, shorter cycle lengths of signal controllers or heavy traffic volumes cause vehicles to experience greater number of stops. The average number of stops is also an important input variable for the air quality models.
Among the selected Measures of Effectiveness (MOE) for the current evaluation process were average stopped delay, travel time delay, number of stops, queue lengths, and total vehicle throughput. To extract these results from the simulation runs, measurement devices were setup at suitable measurement locations and configured in the subject intersection model. Queue Counters were placed near the stop bars. Travel Time Measurements required two sections (origin and destination) to be defined in the links that compose the target route. Definition of the queue condition can be customized in Vissim by specifying a maximum speed (default is at 10 km/h). Queue Counters count all vehicles upstream from the stop bar in queue when it operates at a lower speed.

### 3.5.8 Evaluation Files and Results

Vissim generates a set of output data files at the end of each simulation run. Each of these files corresponds to an evaluation criterion which is also referred to as the Measure of Effectiveness (MOE) or, Performance Measure. For example, Travel Time Delay and Queue Lengths on separate approaches were reported in two separate output files. A list of desired evaluation data has to be preconfigured in Vissim prior to the simulation so that the MOE’s are reported automatically at the end of each run. Simulation was conducted five times by altering the simulation seed in Vissim for a given test scenario. The same set of simulation seeds was used for all three signal control options. The average results from these five runs were considered in final evaluations and these will be presented in the following chapter. Traffic flow gradually starts to reach the defined level, as the simulation is started and continues at this level till the end of each run, for the specified period of
time. A total simulation length of 4200 seconds was used to examine the performance of the three controllers under different traffic volumes. Durations of the simulation runs were limited to 2700, 3600, 4200 and 5400 seconds for each test scenario to examine the effect of time duration on controller performance. Selection of such relatively short time period reflects the underlying assumption that traffic conditions may remain fairly unchanged over such short periods only. Information on signal controller performance was obtained in two manners from the simulation runs. Traffic operation was visually examined from the on-screen display during the simulation runs, as well as from the output files generated at the end of simulation run. The output/evaluation files were then imported into spreadsheet programs for data processing, and preparation of graphs or tables. Vissim generated the output files corresponding to the desired measures of effectiveness, which included average travel times, stops, travel and stopped delays, approach speeds, queue lengths, total throughputs during a specified time period, and vehicle emissions data. The appropriate points of data collection in the subject model, desired evaluation types, and the duration of measurements had to be defined prior to the simulation run, in order to obtain reliable results. This, however, included the warm up time at the beginning and end of each simulation run, when network “warms up” and no data collection is done. Furthermore, variable trace commands were used in the VAP signal control code, so that the modified membership function parameters were recorded every 500 seconds. Values of control delays and external reinforcement values were also recorded applying corresponding trace commands in VAP.
4. **Results and Discussion**

Performance of the subject online reinforcement learning neuro-fuzzy signal control was compared with performances of the two alternative control systems. Evaluation files containing the raw output data were produced at the end of each simulation run in Vissim. The data were processed in spreadsheets followed by statistical analysis. Finally, average results of the simulation runs were presented in the form of tables and plots. The results, evaluation, and comparison of the performances of the three controllers based on the final simulation outputs are presented in the following sections.

4.1 **Results**

The results summarizing the average performances of the three signal control options are presented in Tables 4.1 to 4.11 and also illustrated in Figures 4.1 to 4.11. Total delay, Stopped Delay, Number of Stops, and Queue Conditions were selected as the criteria to assess the operational performance of the three control systems.

The membership function parameter adjustments as a result of the real-time learning process were also examined for each simulation period. The resulting plots showed shifts in the centroids of the three membership functions over the duration of simulation. These were found to be similar to those identified in the work of Pohar (1), reference can be made to Pohar's undergraduate thesis or additional information.
4.1.1 Influence of Traffic Flow on Controller Performance

Measurements were recorded for the following quantities under varying traffic flow levels (i) average total delay, (ii) average stopped delay, (iii) average number of stops, (iv) average travel time, and (v) average queue. The results are presented in Figure 4.1 to 4.5, and in Table 4.1 to 4.5.

Average Total Delay is one of the Measures of Effectiveness to assess the controller performance and to compare the three controllers used in this research. That includes all sorts of delay experienced by traffic flowing through the intersection i.e., as they decelerate, stop at the signal, then start up and accelerate. The average total delay is expressed per single vehicle and is measured in seconds. Figure 4.1 which is based on the delay values given in Table 4.1, shows that the total delay gradually increases with the increase of traffic flow at the intersecting streets.

Table 4.1: Average total delay under varying traffic flow conditions

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Approach Traffic Volumes, vph</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500X500</td>
</tr>
<tr>
<td>RLFNN</td>
<td>14.96</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>15.56</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>16.97</td>
</tr>
</tbody>
</table>

Note: in the above table delay is expressed in seconds per vehicle

With the proposed controller in operation, average total delay per vehicle was only 15 seconds at 500 vehicles per hour each way on the two-lane intersecting streets. The delay...
increased to 38 seconds as the volume increased to 1500 vehicles per hour in each direction.

In addition to the above, the result also reveals that there was a sharp increase in delay as the traffic flow in each direction reached beyond 1300 vehicles per hour. With high traffic volumes on all competing lane groups, it is difficult for any controller to distribute the green time in such a manner that no congestion occurs.

Average stopped delay per vehicle is another satisfactory way to measure the effectiveness of a signal control option; and is included in the total delay computations. Total delay, however, also includes additional delays besides the stopped time. Stopped delay values are, therefore, smaller than the total delay values. Average stopped delays per vehicle at different flow levels, for the proposed real-time adaptive controller, are given in Table 4.2 and illustrated in Figure 4.2. As may be expected, it is seen that the delay values continued
to increase, as the traffic volume increased. However, there was a sharp increase in delay at flow rates above 1300 vph, on all four legs of the intersection.

Table 4.2: Average stopped delay under varying traffic flow conditions

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Approach Traffic Volumes, vph</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500X500</td>
</tr>
<tr>
<td>RL FNN</td>
<td>10.13</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>10.60</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>11.91</td>
</tr>
</tbody>
</table>

Note: in the above table, delay is expressed in seconds per vehicle.

Figure 4.2: Average stopped delay per vehicle plotted against varying traffic flow conditions.

Number of stops is another measure of controller performance. It is not desirable that travelers have to stop too many times, in order to pass through a signalized intersection. Such a “stop-go” situation may arise in queue conditions and shorter cycle lengths or green splits. Many vehicles cannot clear the intersection after the signal light turns green. Rear-end collisions may be more frequent under such circumstances. Average number of stops per vehicle is, therefore, considered a good tool in evaluating a traffic control system. Stopped delay and number of stops may not reflect the same condition. Even a small
number of stops can have longer waiting time in a stopped condition due to longer cycle time and longer green splits. On the other hand, shorter cycle lengths can cause more frequent stops and less stopped time. However, shorter cycles have greater time loss over a long time period, and also greater total delays. The average numbers of stops with the proposed controller are given in Table 4.3 and illustrated in Figure 4.3. Again the controller was simulated to operate under different traffic conditions, i.e., from low to high traffic volumes, keeping all other variables constant. The increase in number of stops made by an average vehicle was more or less steady up to a flow level of 1300 vph. Above this volume, average number of stops increased considerably, going to above two stops per vehicle.

Table 4.3: Average number of stops under varying traffic flow conditions

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>500x500</th>
<th>500x1000</th>
<th>1000x1000</th>
<th>1300x1300</th>
<th>1500x1500</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL FNN</td>
<td>0.65</td>
<td>0.72</td>
<td>0.79</td>
<td>1.01</td>
<td>2.23</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>0.68</td>
<td>0.75</td>
<td>0.79</td>
<td>1.04</td>
<td>2.72</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>0.70</td>
<td>0.76</td>
<td>0.78</td>
<td>0.93</td>
<td>2.78</td>
</tr>
</tbody>
</table>

Longer queues indicate an undesirable level of service in a traffic control system. Usually, queues are encountered, when roadways reach their capacity to serve traffic in an intersection. However, queues may also result from inefficient distribution of green times in a signal-controlled junction.
The intersection model used in this research had separate left turn lanes, which were about 60 meters long. Right turn on red was not permitted, and right-turning vehicles shared the curb lane with through traffic. Recorded queues on all three travel lanes, on an approach, were weighted to obtain the average queue value. Results of these queue measurements are given in Table 4.4 and illustrated in Figure 4.4. Similar to the other measures of effectiveness, queue increased with traffic volume. The increase was pronounced, when approach volumes of 1500 vph was defined under the proposed controller.

**Table 4.4:** Average number of queued vehicles under varying traffic flow conditions

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Approach Traffic Volumes, vph</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500X500</td>
</tr>
<tr>
<td>RL FNN</td>
<td>4.64</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>4.79</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>5.10</td>
</tr>
</tbody>
</table>

Note: Queue values are given in number of vehicles.
The average time taken by vehicles to traverse a selected distance can reflect the traffic condition within that segment of route. Vehicles can have a free flow state, if there is no interruption along their course. Yet, travel time increases due to various types of delays such as traffic signals, sharing of right-of-way or incidents. To determine average travel times on the modeled roadway, start and end sections were defined. Same distance along three possible routes, i.e., left, through and right, was considered from each approach. The distance covered the subject intersection keeping it in the middle and captured the delay caused by the control measures. Volume based weights were applied to obtain the average of left, through and right turning traffic along respective routes. The results are given in Table 4.5 and illustrated in Figure 4.5. The time to travel the predefined distance increased with the increase in traffic volume. The intersection performance degraded due to the high number of vehicles waiting to be served. Left turning vehicles overflowed the designated storage lane. The resulting queues even blocked the center through lane significantly,
constraining the intersection capacity. The system got into a gridlock scenario with high approach volumes, like 1500 vehicles per hour. The high values of average travel times, at this stage, varied greatly. As mentioned in the methodology section, robustness of each signal controller was tested under various traffic flow scenarios. At each level of traffic volume, the simulation seed in Vissim was altered five times and the result of these five runs were averaged. However, for certain random seeds, the resulting traffic flow was significantly unstable with lack of platoons at times and constant heavy platoons at rest of the times. This sometimes resulted in congestions leading to a failure as a large wave of vehicles entered the model within a short interval and at some other times there were absolutely no vehicle arriving at the intersection. This situation was experienced regardless of which controller was being tested and therefore, these runs were not included in subsequent calculations or presentation of results.

Table 4.5: Average travel time under varying traffic flow conditions

<table>
<thead>
<tr>
<th>Controller</th>
<th>500x500</th>
<th>500x1000</th>
<th>1000x1000</th>
<th>1300x1300</th>
<th>1500x1500</th>
</tr>
</thead>
<tbody>
<tr>
<td>RLFNN</td>
<td>37.82</td>
<td>40.38</td>
<td>42.19</td>
<td>47.93</td>
<td>61.08</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>38.41</td>
<td>40.58</td>
<td>42.23</td>
<td>47.98</td>
<td>65.10</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>39.79</td>
<td>41.90</td>
<td>43.98</td>
<td>50.42</td>
<td>70.86</td>
</tr>
</tbody>
</table>

Note: in the above table travel time is expressed in seconds per vehicle
4.1.2 Influence of Simulation Length on Controller Performance

The proposed real-time reinforcement learning control algorithm has a time related learning system. The membership function parameter and reinforcement arrays are fine tuned or updated with every time step. Therefore, in the course examining its operational performance, the influence of duration in each simulation run was determined. However, the learning in the control algorithm was not designed to begin with the simulation; it commenced only after 500 or 1000 seconds/time steps of the total run time. The adequate duration of simulation was thought to be 4200 seconds, equivalent to 1 hour and 10 minutes. Longer durations may not represent the real-world situations experiencing dynamic traffic flow conditions. Some initial and final time was spared from data collection because the intersection model in Vissim was partially full, i.e., the traffic feed into the network. Vehicles were gradually fed into and taken out of the system, and that
does not correspond to the desired flow rate. Otherwise, an empty network may, 
eventually, be included in the data collection and computations of delay.

It is expected that a real time adaptive controller would perform better, if it had a longer 
training period. However, over a long period of time, traffic conditions did not stay the 
same. To address this issue, further investigation using varying durations of simulation 
was carried out. Four different lengths of simulation run, 2700, 3600, 4200, and 5400 
seconds, were considered for this purpose. The total traffic volumes on each of the four 
approaches were maintained constant, at 1300 vehicles per hour. The five Measures of 
Effectiveness, as applied before, were again selected. This time, the length of simulation 
was the only variable. Mean and variance of these results corresponding to five different 
seeds were then computed to form the basis of a comparative evaluation of the three 
controllers. Findings are discussed in the following paragraphs. The results are also 
presented in Tables 4.6 to 4.10 and illustrated in Figures 4.6 to 4.10.

The average total delay results, for the proposed controller, form the first set of results and 
are presented in Table 4.6 and graphically presented in Figure 4.6. Values in the table 
suggest that the average total delay is about 25 seconds per vehicle, regardless of how long 
the simulation runs. The curve appears to have a crest with a peak at about 3600 seconds 
mark. Also, the value is smallest at the longest duration of simulation. That may not 
necessarily establish the fact that longer training or learning serves better results, as the 
delays in the case of other controllers also experience their minimum at 5400-second runs.
In a brief statistical analysis it was revealed that the differences in delay values are insignificant and purely random in nature.

**Table 4.6:** Average total delay under different simulation durations

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Length of Simulation, s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2700 s</td>
</tr>
<tr>
<td>RL FNN</td>
<td>25.03</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>25.00</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>26.97</td>
</tr>
</tbody>
</table>

Note: delay is expressed in seconds per vehicle.

![Graph showing average total delay per vehicle plotted against varying simulation durations](image)

**Figure 4.6:** Average total delay per vehicle plotted against varying simulation durations (Average values of five simulation runs under constant traffic volumes of 1300x1300 vph)

Results on average delay due to stops are given in Table 4.7 and illustrated in Figure 4.7. The stopped delays and simulation duration appears to have a correlation similar to that shown in the plots for total delay and discussed in the preceding paragraphs. However, values of the stopped delay times are less than the total delay and this is in agreement of their definition. The average stopped delay remains more or less steady at 15 seconds per vehicle and drops to a minimum at 5400-second runs.
Table 4.7: Average stopped delay under different simulation durations

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Length of Simulation, s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2700 s</td>
</tr>
<tr>
<td>RL FNN</td>
<td>16.25</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>15.97</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>18.71</td>
</tr>
</tbody>
</table>

Note: in the above table delay is expressed in seconds per vehicle

Figure 4.7: Average stopped delay per vehicle plotted against varying simulation durations
(Average values of five simulation runs under constant traffic volumes of 1300x1300 vph)

The number of stops that a vehicle had to make is about one, on average, as it cleared the travel time measurement section. The values are given in Table 4.8, and shown in Figure 4.8. As will be noted, there is very little to no difference in the average number of stops due to variations in the simulation duration. This lead to the conclusion that average numbers of stops is not affected by length of simulation runs.

Table 4.8: Average number of stops under different simulation durations

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Length of Simulation, s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2700 s</td>
</tr>
<tr>
<td>RL FNN</td>
<td>0.97</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>1.01</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>0.91</td>
</tr>
</tbody>
</table>
Simulation results for average queues were obtained by testing the system under different durations of simulation. A fixed flow rate of 1300 vph from all approaches was selected for that purpose. The results are presented in Table 4.9, and the plots are given in Figure 4.9. The proposed controller served with an average queue of 21 vehicles. The shape formed by the plot of queue values seems to have a peak at around a 3600-second simulation run. The differences in the average queues were not found to be significant due to the variations in the simulation length.

**Table 4.9:** Average number of queued vehicles under different simulation durations

<table>
<thead>
<tr>
<th>Controller Type</th>
<th>Length of Simulation, s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2700 s</td>
</tr>
<tr>
<td>RL FNN</td>
<td>20.85</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>21.22</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>21.49</td>
</tr>
</tbody>
</table>
Travel time variation against different lengths of simulation was also examined for an overall satisfactory evaluation. The same set of simulation lengths was used in this phase of investigation. The resulting travel time values for the proposed controller along with the two other types are given in Table 4.10, and shown in Figure 4.10. The average travel time ranged from 46 to 48 seconds for the four different runs. The proposed real-time reinforcement learning based controller seems to be independent of simulation runs in the selected range of duration.

Table 4.10: Average travel time under different simulation durations

<table>
<thead>
<tr>
<th>Controller</th>
<th>Length of Simulation, s</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL FNN</td>
<td>2700 s</td>
</tr>
<tr>
<td></td>
<td>47.96</td>
</tr>
<tr>
<td></td>
<td>3600 s</td>
</tr>
<tr>
<td></td>
<td>48.48</td>
</tr>
<tr>
<td></td>
<td>4200 s</td>
</tr>
<tr>
<td></td>
<td>47.93</td>
</tr>
<tr>
<td></td>
<td>5400 s</td>
</tr>
<tr>
<td></td>
<td>46.01</td>
</tr>
<tr>
<td>Modified FUSICO</td>
<td>2700 s</td>
</tr>
<tr>
<td></td>
<td>47.95</td>
</tr>
<tr>
<td></td>
<td>3600 s</td>
</tr>
<tr>
<td></td>
<td>49.05</td>
</tr>
<tr>
<td></td>
<td>4200 s</td>
</tr>
<tr>
<td></td>
<td>47.98</td>
</tr>
<tr>
<td></td>
<td>5400 s</td>
</tr>
<tr>
<td></td>
<td>46.16</td>
</tr>
<tr>
<td>Vehicle Actuated</td>
<td>2700 s</td>
</tr>
<tr>
<td></td>
<td>49.92</td>
</tr>
<tr>
<td></td>
<td>3600 s</td>
</tr>
<tr>
<td></td>
<td>50.47</td>
</tr>
<tr>
<td></td>
<td>4200 s</td>
</tr>
<tr>
<td></td>
<td>50.42</td>
</tr>
<tr>
<td></td>
<td>5400 s</td>
</tr>
<tr>
<td></td>
<td>49.27</td>
</tr>
</tbody>
</table>

Note: In the above table travel time is expressed in seconds per vehicle
Results and Discussion

Figure 4.10: Average travel time per vehicle plotted against varying simulation durations (Average values from five simulation runs with constant traffic volumes of 1300x1300 vph)

4.1.3 Comparison of the Signal Control Options

Results obtained for the two competing control strategies are compared to those for the proposed online RLFNN signal control strategy and suggest positive changes in the intersection performance with the implementation of the RLFNN control. Percentage changes in the average values of the measures of effectiveness are shown in Table 4.11. Negative signs in front of the tabulated values represent a decrease i.e., improvement in terms of average delay, number of stops, travel time, or queue, when a RLFNN controller was used. The differences in the results appear to be higher at or near the lower and heavier traffic volumes. The average total delay, stopped delay, number of stops, queue and travel times experience improvements with the implementation of the proposed controller over both FUSICO and Actuated Controllers. However, improvements using the RLFNN
controller are found to be more pronounced when compared to the actuated signal controller.

**Table 4.11:** Comparison of the MOE values with respect to the proposed control

<table>
<thead>
<tr>
<th>MOE</th>
<th>Controller</th>
<th>500x500</th>
<th>500x1000</th>
<th>1000x1000</th>
<th>1300x1300</th>
<th>1500x1500</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg Total Delay</td>
<td>Modified FUSICO</td>
<td>-4%</td>
<td>-1%</td>
<td>0%</td>
<td>0%</td>
<td>-10%</td>
</tr>
<tr>
<td></td>
<td>Vehicle Actuated</td>
<td>-12%</td>
<td>-8%</td>
<td>-9%</td>
<td>-9%</td>
<td>-20%</td>
</tr>
<tr>
<td>Avg Stopped Delay</td>
<td>modified FUSICO</td>
<td>-4%</td>
<td>-1%</td>
<td>-1%</td>
<td>1%</td>
<td>-9%</td>
</tr>
<tr>
<td></td>
<td>Vehicle Actuated</td>
<td>-15%</td>
<td>-11%</td>
<td>-13%</td>
<td>-16%</td>
<td>-29%</td>
</tr>
<tr>
<td>Avg Number of Stops</td>
<td>Modified FUSICO</td>
<td>-4%</td>
<td>-4%</td>
<td>0%</td>
<td>-2%</td>
<td>-18%</td>
</tr>
<tr>
<td></td>
<td>Vehicle Actuated</td>
<td>-6%</td>
<td>-5%</td>
<td>1%</td>
<td>9%</td>
<td>-20%</td>
</tr>
<tr>
<td>Avg Queue</td>
<td>Modified FUSICO</td>
<td>-3%</td>
<td>-1%</td>
<td>0%</td>
<td>-2%</td>
<td>-12%</td>
</tr>
<tr>
<td></td>
<td>Vehicle Actuated</td>
<td>-9%</td>
<td>-12%</td>
<td>-19%</td>
<td>-17%</td>
<td>-20%</td>
</tr>
<tr>
<td>Avg Travel time</td>
<td>Modified FUSICO</td>
<td>-2%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>-6%</td>
</tr>
<tr>
<td></td>
<td>Vehicle Actuated</td>
<td>-5%</td>
<td>-4%</td>
<td>-4%</td>
<td>-5%</td>
<td>-14%</td>
</tr>
</tbody>
</table>

Average total delay decreased by 4% at 500 vph, when the proposed controller replaced a Modified FUSICO. The decrease diminishes with increasing volumes and at 1000 to 1300 vph approach volume, and the performances become indifferent. On the other hand, differences in delay are greater between the proposed and the actuated controller. A 12% decrease in delay was observed between the two at a low traffic flow condition. The difference in the delays was less during medium traffic flow level. Under heavy traffic volumes, i.e., 1500 vph flow, there was a high margin of difference in performances of the three control mechanisms. The proposed RLFNN controller performed better in comparison to the other two types. The resulting difference curves with respect to the RLFNN controller is shown in Figure 4.11. However, results for the heavy traffic
conditions resulted in a high variability and the system even failed for certain simulation seeds. This has been discussed earlier in this section. Vissim, the simulation tool used in this research, has a technique that removes a vehicle from the model, if it has to wait for over a minute to move to its desired lane for turning. Features like that may sometimes lead to unrealistic results, in severe queue conditions. Again, the traffic volumes used in this research were probably slightly on the high side, with respect to the selected intersection capacity.

![Figure 4.11: Improvements in average delay when the proposed RLFNN controller shown under different flow conditions replaces the other controllers.](image)
5. Conclusions

The design and assessment of real-time adaptive signal control logic was presented in this thesis. Based on the preliminary research, literature review, experiments, and analysis of results, following is the discussion on the key findings and conclusions.

5.1 Summary of Findings

Simulation in Vissim was carried out to assess the performance of the subject controller in comparison to the two other alternative control options: the modified FUSICO, and the fully-actuated controller. Results and evaluation outputs were generated for each simulation run. Five simulation runs using different random number seeds were carried out under each scenario. Average results from these runs were tabulated and have been presented in chapter four of this thesis. From an analysis of these results already presented, following is a summary of the key findings on the performance of the proposed online reinforcement learning signal control:

The proposed system was found to deliver improvements in terms of delay and queue reductions under varying traffic volumes. Percentage reductions in the queue and delays by the proposed controller were compared with the actuated and modified FUSICO controls. Positive percentage reductions in average total delay were observed for all traffic volumes, when an online reinforcement learning neuro-fuzzy controller was used. However, the margins of the better performance diminished with the increase in traffic volumes on the
intersection approaches. The superiority of the proposed controller was pronounced when compared to the Actuated controller.

Performance of the actuated and modified FUSICO controller was found to equalize at higher (1300x1300 vph) traffic volumes. With regard to the average queue lengths, the proposed controller had significantly lower queues than with the actuated control, particularly at higher traffic volumes (10% to 20%). The difference in the performances of the proposed and the modified FUSICO controller were less than 4% at all traffic volumes. Moreover, when the conflicting traffic directions (e.g. north-south and east-west approaches) have different volumes of traffic, the FUSICO controller slightly outperformed the subject controller.

The results also indicated percentage reductions in the two MOEs, i.e., delay and queue lengths, for varying duration of simulation runs. A higher length of simulation represents a longer learning cycle for the proposed reinforcement-learning controller. For the modified FUSICO and the actuated controllers, the duration of simulation did not have significant effects. However, a longer simulation (e.g. 5400 seconds, or 1 hour 30 minutes) time also assumed a constant traffic flow condition throughout the period and may not reflect practical conditions. It was shown that both percentage reductions in average total delays and queue lengths increase as the duration of controller implementation (i.e., simulation duration) increases. That is another consequence, which may be expected from the working principle of a neuro-fuzzy controller based on online reinforcement training. The differences in the reductions in average total delay of the three controllers were small in the varying lengths of simulation.
Stochastic exploration into the input-output space to modify the control output should ideally help the controller find better controls decisions. However, as the controller cannot distinguish the “sound” control actions the Stochastic Action Modifier (SAM) module continues to modify even the “appropriate” outputs in search of optimality. This may, in fact, affect the optimal performance of otherwise satisfactory control actions.

Due to its inherent demand based algorithm, the control system presented is expected to achieve a more efficient allocation of green times in signal operation. For real-time demand information, traffic is counted at short time intervals, and the control effectiveness is measured in terms of resulting delay. The system has the ability to influence the decision mechanism in order to yield control outputs that produce better results, i.e., minimum delays. The adaptation procedure is based on real time measurements of traffic input variables, so it suits the dynamic traffic conditions in the real world. The modified FUSICO control algorithm used for comparison purposes does not use a real-time reinforcement-learning module. The actuated controller does not count the number of vehicles waiting to be served, it continues to serve a phase if vehicles continue to reach the detector within a minimum “gap”, unless a recall mode or maximum threshold is specified. At relatively low and heavy traffic volumes, the fully-actuated or the modified FUSICO control may not perform in an efficient manner. Under such conditions, a demand based, delay responsive allocation and extension of green times among the candidate phases becomes necessary.
The control system presented in this thesis was implemented in the Vissim traffic simulation software package with the control algorithm was originally coded in VISVAP, a user-friendly programming tool applying the VAP language. Both of these applications have some limitations which affected the current research objectives to some extent. For example, programming limitations in VAP restricts the use of exponential functions in the development of desired signal control code. Exponential functions, such as Gaussian, Bell, etc. were present in the fuzzy membership functions in the original concept but needed to be replaced with Taylor's polynomial, used as an approximation technique.

Some parameters of the subject control algorithm, viz. delay (reinforcement signal computations), unit green extensions (control outputs), could only assume discrete integer values in Vissim. This perhaps made the functions of the SAM, Reinforcement Predictor and the reinforcement learning process as a whole less effective to some extent due to rounding-off of output values from those program modules.

In microscopic simulators including Vissim, it takes several simulation time steps for traffic volumes on various links of the intersection model to attain the desired flow rates as the simulator starts to gradually fill in the empty network. This initial "warm-up" time should be carefully excluded from the time period during which evaluation data is extracted. Any performance measurement at this time would represent unrealistic performances as the intersection traffic volumes are actually much lower than that originally defined.
Use of a different simulation seed in Vissim changes the traffic generation, i.e. vehicle arrival pattern in the simulation. Traffic arrival pattern or platooning does not change as long as the seed is not manually changed. Performances of the three signal control options considered in this study were examined under identical traffic conditions. For each set of test scenario, five consecutive simulation runs were undertaken each time using a different random seed to create a variety in the traffic conditions. The average result from these five runs was considered in this thesis so that any variability issues related to the performance under random traffic flow patterns could be addressed.

An ideal traffic environment was simulated in Vissim where for the signal controller implementation. In the model, any external interference or variability was controlled and accurate, real-time traffic data and delay information was available with simple program instructions. In field application, however, this may be slightly difficult to establish, as there would be numerous detectors involved, as well as influence of uncontrollable variables and due to lack of precise traffic counts. Furthermore, the initial test runs of controller simulation indicated that the farthest the upstream detectors were located on an approach, the better the results were. In the field, as the intersection approaches can be distinctly isolated only to a certain maximum distance beyond the influence of adjacent driveways and intersections and on-street parking, the detectors would have to be placed within a limited distance.

Separate right hand turn lanes were not included in the intersection model used in Vissim in this research. Therefore, all right turn movements on red signals also had to be restricted so
that the vehicle counts obtained from the pair of detectors included all vehicles queued behind the red signals.

5.2 General Discussions

Most of the existing signal control systems use pre-configured timing and phasing plans that are based on historical traffic movement data at an intersection. Such systems cannot respond to dynamic and fluctuating traffic demands and leads to unnecessary delays, frequent stops and long back-ups. To eliminate these deficiencies, traffic engineers remain keen to implement signal control systems that can serve critical, traffic demands on a real-time basis. Controllers currently used in the field are primarily of two main categories: pre-timed and adaptive. These signal controllers are sometimes coordinated over a network of intersections for a more efficient and smooth traffic operation. Adaptive systems, which are relatively new in the field, are newer variations of vehicle actuation based control technology and are more suitable for fluctuating traffic conditions. Under traffic adaptive control, including the system presented here, the strategy is to optimize and update signal timings based on actual traffic demands as frequently as practicable.

A number of earlier research works have indicated that the application of fuzzy-neural networks in real-time control tasks can be very efficient. For example, in a physical system that has input variables of a linguistic nature, it cannot be handled by the conventional Boolean algebra in the control logic. Again, a mathematical model of the physical process may not even exist, or if it does exist, is rather difficult or complex to function in a dynamic
real-time operation. In such cases, neuro-fuzzy control systems have been shown to perform better. These systems take the advantage of incorporating expert knowledge into their control algorithm. Also, the integrated system of fuzzy systems and neural networks can learn from the information extracted from the environment. Advantages of neuro-fuzzy systems can be better understood if the two complementary systems are looked at separately. In general, following are the key advantages of a fuzzy logic based control:

- Fuzzy systems use linguistic variables which is suitable for describing vague, fuzzy environments, such as traffic variables like queue length, delay, amount of green extension, etc.
- Allow the use of imprecise/contradictory inputs, e.g., in low traffic volume conditions a queue of “5 to 10” stopped vehicles may be perceived as “long”, whereas in heavier traffic volumes up to “10 to 20” vehicles may be considered as “medium”. Furthermore, different individual linguistic variable definitions for “queue length” may be used for the major and minor streets and thereby indicate priorities.
- Permit fuzzy/overlapping threshold values for input and output variables.
- Can reconcile conflicting objectives.
- Fuzzy rulebase or sets can be easily modified to suit specific conditions, e.g., a fuzzy rule of type “if x is a and y is b, then z is c”, can be altered by modifying any of the linguistic functions a, b or c.

Besides this, as an important part of the integrated system, neural network systems can contribute with the following features when integrated into the system:
Based on the prior training data, neural networks can adjust themselves to reproduce the desired output in the learning phase.

- Have the capability of self-optimization.
- Have a connectionist structure.
- Can offer high computational efficiency.
- Can assign and modify weights of various network connections, and thus, signifying certain conditions or rules.

Combination of fuzzy and neural systems can therefore deliver several advantages over conventional traffic control systems. One key feature of fuzzy systems is that they are flexible and lend themselves easily to human-like reasoning. However, few modeling and learning theories exist for fuzzy systems. A neural network, on the other hand, is capable of learning from data and performing extensive parallel processing, but being a black-box type of application, a neural network system is sometimes difficult to interpret. A combination of these systems may have both a qualitative and a quantitative interpretation, and may mitigate drawbacks of a solely fuzzy or neural network based system. Enhancing the fuzzy rule processing with the parallel processing abilities of neural networks gives high computational efficiency. In addition, neural networks can be used to find a feasible structure for a fuzzy controller. While the number of rules, rulebase definition, linguistic values and shapes of the membership functions, etc. influence the performance of the control system, all these can be adjusted, or tuned, with the application of neural learning schemes. Neuro-fuzzy learning can be in terms of structure, as well as parameter learning which means either updating the
network weight parameters or fuzzy membership function parameters or selection or modification of the rulebase itself. Although the fuzzy linguistic representation does not enhance the modeling capabilities of the network, it allows the designer to incorporate expert knowledge and determine if a self-generated system structure is feasible. Thus, expert knowledge can easily be used in a fuzzy system, in terms of formulating the rule base and choosing the initial membership function parameters.

Finally, the real-world traffic environment is very complex and dynamic to be handled by the traditional signal control applications. Flow conditions at a four-approach intersection with all turning movements, could be highly unpredictable in nature. Such a physical environment can produce constantly changing input variables for the signal control system. Self-refining, demand responsive control logics similar to the logic presented in this thesis are anticipated to produce better results under most traffic conditions. This advantage is achievable with the successful application of intelligent control concepts that have already been proven to perform efficiently for other machine control systems. It has been suggested from recent works that the application of fuzzy-neural networks can be very efficient in many real-world situations. For example, a research team at the Helsinki University of Technology introduced a fuzzy logic to describe the human perception and control vehicle movement. Fuzzy logic was found to be a competent method for modeling human decision-making as the rules can easily be formulated in linguistic form. Fuzzy rulebase or decision-making is based on abstracting the studied situation and then expressing decision using vague information. For transportation engineering and especially microscopic modeling of traffic, fuzzy logic can be the appropriate
method. It was observed that the FUSICO signal control algorithm is robust and can perform efficiently under various traffic conditions when applied to an isolated intersection. It was also shown to have a superior performance over the traditional pre-timed and simple actuated controllers.

In this research, the performance of the subject reinforcement learning neuro-fuzzy control was evaluated in comparison with alternative signal control systems which included a modified FUSICO and a simple vehicle-actuated controller. Pre-timed or fixed-time controllers were not considered in the comparison purposes.

5.3 The Proposed Signal Control Logic

The proposed system applied the principles of fuzzy control logic associated with neural networking. The system was designed to learn while in operation and tend to select control decisions that had better results historically, all these without any external assistance.

The basic task of the RLFNN control algorithm was to determine and distribute optimum green signal times among various candidate signal groups based on actual traffic demands. Real-time traffic counts and delay information at the intersection approaches are obtained from vehicle detection devices. The system either extends or terminates an active green signal based on the number of queued vehicles on the different approaches at the intersection. The system also receives information on delays resulting from its past control actions from the detectors. The proposed control therefore has a closed-loop system that receives inputs on
traffic demands from the detectors, uses it to determine appropriate outputs and then it reviews the performance feedback in terms of vehicular delay in a continuous and iterative manner.

The underlying assumption, which also motivated this research, was that optimum allocation of green times can maximize controller efficiency by minimizing vehicular delays and maximizing the intersection capacity. Efforts were made in several earlier research works to develop signal control logics with objectives similar to this research. Of these, the works of Pohar (1) at the University of British Columbia and Bingham (2) at the Helsinki University of Technology are most resembling albeit with a different type and scope of control system. The design presented here extended the application of the original control algorithm presented in Pohar’s undergraduate thesis. The available VAP program code was revised to incorporate additional features including capabilities of controlling a typical four-leg intersection of multilane arterial roads with two-way traffic flow. The control logic was also evaluated under a wide range of traffic flow conditions and the performance was compared with those of two alternative types of signal control systems.

The program code for the proposed control was generated using the VISVAP module of the traffic simulation package called Vissim which was then implemented into the micro-simulator. The VAP code was actually used as external signal control logic in the traffic simulation program. Performance of the signal control logic was examined under various
simulated traffic flow scenarios. Finally, desired output data were obtained at the end of simulation runs which were then processed for final results presented earlier in this thesis. The ultimate objective of the ongoing research at the University of British Columbia is to establish an online neuro-fuzzy traffic signal controller capable of handling traffic at an intersection that experiences varying traffic demand conditions. The subject signal control algorithm was designed to operate efficiently under actual field conditions which include many issues, such as unpredictable and varying conditions and complex flow patterns. In the current work, however, scope and application of the proposed control was limited to a single, isolated intersection only. The proposed RLFNN control concept is superior to the conventional ones in the way that it does not require any manual adjustments or supervision to serve fluctuating traffic demand conditions. The neuro-fuzzy algorithm adjusts the initial model parameters according to the changing environment. This eliminates the need for constant monitoring, maintenance and updating tasks so that appropriate signal timing plans are present to meet variable traffic demands. It therefore has the potential to reduce resources in terms of its maintenance and operation. However, like other adaptive signal control systems, it requires extensive deployment of traffic detectors, advanced equipment and the cost associated with hardware installation and maintenance increases.

5.4 Recommendations for Future Research

So far the application of neuro-fuzzy signal control has been limited to isolated or uncoordinated, individual intersections only. The scope may be extended to establish a network-wide online reinforcement learning optimal signal control.
Under the proposed control system, traffic approaching the intersection on the four legs is recorded by a pair of detectors. This information is used to determine the degree of urgency to receive or retain a green signal. The phasing plan was predefined where only the left turn phases were non recalling meaning they could be skipped when not necessary. The phasing plan was based on the assumption that traffic volumes in two opposite directions on each link are equal with only small percentage of left and right turns. Possibilities of alternative or critical traffic movement situations can be examined where the controller would be capable of selecting suitable phasing plans, their orders, as well as their timings.

The reinforcement learning in the subject neuro-fuzzy signal control was based on parameter training/updating only. Efforts may be undertaken to incorporate structure learning as well so that the system would have greater flexibility to suit to changing traffic conditions.

Incident detection and handling of emergency situations may be incorporated in the control logic so that it qualifies as a robust and competent traffic signal control equipped with practical applications.

The original FUSICO rulebase may be modified so that more traffic variables can be handled by the system including priority signals and network optimization.
5.5 Conclusion

The proposed control algorithm presented in this thesis, investigates the possibility of a neuro-fuzzy signal control with real-time adaptability. The work extended the scope and capabilities of an earlier design which was available at the University of British Columbia. Its performance was also examined under a wide range of traffic flow scenarios at a typical four-leg intersection of arterial roads. Experiments undertaken in this research suggested positive results to some extent although the concept should be considered in a preliminary stage. Current shortcomings may be satisfactorily overcome with suggested improvements in the areas outlined above. It is believed that the proposed concept can serve as a base for a reliable, real-time self-adjusting intelligent signal control strategy to meet future needs.
LIST OF REFERENCES

1. Pohar, Denesh, Online Reinforcement Learning in Neuro-Fuzzy Traffic Signal Control, 


11. Niittymaki, Jarkko, P., Isolated Traffic Signals-Vehicle Dynamics and Fuzzy Control,  
12. Pursula, Matti, & Niittymaki, Jarkko, Evaluation of Traffic Signal Control with Simulation—A  
   Comparison of the Pappis-Mamdani Fuzzy Control vs. Vehicle Actuation with the Extension  
13. Lin, Chin-Teng, & Lee, C. S. George, Reinforcement Structure/Parameter Learning for  
   Neural-Network-Based Fuzzy Logic Control Systems. *IEEE Transactions on Fuzzy Systems*,  
   2, June 1995.  
   February 1994.


32. VISSIM Technical Description, Website: www.ptv.de

34. Leonard II, John D., An Overview of Simulation Models in Transportation *Website Article*,
   http://www.sisostds.org/webletter/siso/iss_79/art_429.htm


37. SMARTEST, *Website Article*, http://www.its.leeds.ac.uk/projects/smartest/finrep.PDF