Delft University of Technology
Master's Thesis in Embedded Systems

DIRECTOR: Enabling advanced driver assistance systems with predictive signalized intersection control using LSTM networks

Jan Cees van Senden

DIRECTOR: Enabling advanced driver assistance systems with predictive signalized intersection control using LSTM networks

Master's Thesis in Embedded Systems

Embedded Software Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Jan Cees van Senden
J.C.C.vanSenden@student.tudelft.nl

Thursday 31st May, 2018

Author

Jan Cees van Senden (J.C.C.vanSenden@student.tudelft.nl)

Title

DIRECTOR: Enabling advanced driver assistance systems with predictive signalized intersection control using LSTM networks

MSc presentation

Tuesday 12th June, 2018

Abstract

Traffic congestion at signalized intersections is a big economical and ecological problem. Handcrafted traffic light controllers (TLCs) are currently used to minimize the impact, but they are expensive to design and maintain and their performance degrades over time. Predictive TLCs and advanced driver assistance systems (ADAS) form a potential solution but are still unfeasible in practice today because of their computational complexity and unpredictability.

The distributed predictive TLC developed in this thesis, called DIRECTOR, is feasible and enables time to green/red and green light optimal speed advice (GLOSA) systems. DIRECTOR utilizes predictions of the arriving traffic flows and a model of the current queue length to optimize the traffic light schedule. It can operate in two modes; Ad-hoc mode, where the schedule is generated and applied right away, and fixed-ahead mode, where the schedule is fixed in advance to enable ADAS. DIRECTOR's design makes it scalable and suitable for live learning, eliminating the need for expensive (re)calibrations and improving its performance with more and better data, which will become available in the near future.

A long short-term memory recurrent neural network is developed to predict the arriving traffic flows. On a case study this network proves to be on average 4.7% more accurate than the current state-of-the-art model, which is significant for a controller's performance.

Simulations of the same case study intersection, which is currently equipped with a state-of-the-art actuated controller with green wave coordination, show that in ad-hoc mode DIRECTOR performs similar to the current controller. DIRECTOR reduces the average delay per vehicle by 1% (from 10.4s to 10.3s) at the cost of an increase of 15% in the average number of stops per vehicle (from 0.40 to 0.46) compared to the current controller. Simulations with ideal predictions show that, in ad-hoc mode, DIRECTOR has the potential to improve the average delay by 8.7% (from 10.4s to 9.5s) while keeping the number of stops equal (at 0.40).

Simulations with GLOSA show a 30% reduction in the average number of stops at the cost of a 13% increase of the travel time compared to the ad-hoc mode. Combining this with ideal predictions shows that DIRECTOR in fixed-ahead mode has the potential to keep the average delay equal compared to the current controller, which will greatly improve traffic flow.

Compared to a more typical Dutch actuated controller, DIRECTOR achieves a delay reduction of 39% in ad-hoc mode and 23% in fixed-ahead mode.

Overall, DIRECTOR is a new data-driven traffic light controller that is relatively easy to set up, reduces costs, can enable advanced driver assistance systems, is futureproof and has the potential to greatly improve traffic flow.

Preface

Throughout writing this thesis many people have been of great help. I would like to thank the most significant contributors by name.

First of all I would like to thank Eddy Verhoeven for granting me a carte blanche opportunity to conduct my research at Siemens Mobility. Eddy was patient when the project was still taking shape and energetic in getting me the right resources when the project could be accelerated.

Secondly, I must thank Noorhan Helmy, my predecessor at Siemens, for taking so much time to explain the ins and outs of her research, which really kick-started mine.

Of course many other people at Siemens Mobility have helped shape this thesis. In particular I would like to thank Linda Lanphen for her direct and constructive feedback. And of course the Talking Traffic team; Celine Krstulovic, Peter Luns, Martijn Brautigam, Alexander Koek and Hans Looijen, who were always there to answer questions, to build something I needed and to have a good laugh.

Finally I would like to thank my committee members from Delft University. Matthijs Spaan for the insights acquired during our sparring sessions. Koen Langendoen for the guidance throughout the entire process. Giving freedom in deciding which direction to head into, yet strict, detailed and constructive when it came to reviewing the progress and documents. But always with a sense of humor.

Jan Cees van Senden

Zoetermeer, The Netherlands
May 24, 2018

Contents

Chapter 1

Introduction

1.1 Background

The costs of congested traffic flow are estimated at $700 Bn [22] mainly due to travel time delays and fuel inefficiencies. Part of these losses are caused by stop-and-go behavior at signalized intersections. In an attempt to decrease these losses the efficiency of traffic light controllers has improved tremendously over the years moving from fixed-cycle to adaptive to actuated control. This section provides a brief introduction on signalized intersection control explaining the historic development, the important concepts and the ongoing efforts to further improve traffic flow in the future.

1.1.1 Signalized intersection control 101

Signalized intersections by definition have a negative impact on local traffic flow since their purpose is to control access to an intersection by stopping vehicles from conflicting directions. Nevertheless their existence is crucial for road safety. Hence, the challenge is to design signalized intersection control mechanisms that have the least negative impact on the traffic flow.

Before the different types of control can be discussed a better understanding of the terminology is required. First of all an intersection is the connection point of several links. In this thesis a link is defined as a road segment that is connected to the intersection. The intersection's purpose is to ensure a vehicle's safe passage from an origin link to a destination link. A matching origin link and destination link are called an origin-destination pair (OD).

Each link will have one or more signals assigned to it, one for each lane feeding traffic into the intersection. A signal grants or prohibits the traffic on its lane access to the intersection. In practice this is done using green and red light signals, respectively. Granting access is also called servicing. A signal group is defined as a combination of signals that are always serviced together and have the same id. Non-conflicting signals are signals that can be serviced simultaneously without the possibility of traffic intersecting with each other, i.e. without the possibility of causing an accident.

A phase group is a combination of signal groups that the traffic light controller will service simultaneously. A phase is defined as the period that a phase group is serviced. A traffic light schedule is defined as a sequence of phases. The challenge of intersection control optimization is to construct the optimal traffic light schedule with regards to the desired optimization metric(s) within the intersection's boundary conditions.

Traditionally the optimization metrics have always been the cumulative or average travel time delay, which is the time lost due to non-free flow conditions caused by for example other vehicles or signalized intersections. Other traditional metrics are the number of stops or the queue lengths at an intersection. More recently, ecological impact has gained in importance leading to optimization metrics such as the fuel efficiency and, CO2 or NOx emissions. In practice a combination of metrics is often used for optimization, leading to a trade-off when making a scheduling decision.

The boundary conditions for intersection control differ per intersection and are dependent on local legislation and the road owner's preferences. A simple example of a legislative boundary condition is the minimum duration of amber (a.k.a. orange) when a signal transitions from green to red. Another example is the maximum time that a signal group is allowed to remain unserviced. This last example is an example of both a safety and a fairness boundary condition. When a signal group is not serviced for too long a road user may think that he or she will not be serviced and drive through a red signal, possibly leading to a dangerous situation. A fairness condition ensures that, although general optimization of the intersection might benefit from a particular signal group not being serviced for a while, road users waiting at that particular signal group are not penalized unfairly.

1.1.2 Taxonomy of signalized intersection control mechanisms

There are four main types of signalized intersection control. In order of improving performance they are:

1. Fixed-time control
2. Adaptive control
3. Actuated control
4. Model predictive control

The simplest signalized intersection control mechanism is unoptimized fixed-time control. Fixed-time control refers to the fact that the schedule of the controller is fixed. The duration of each phase and the order in which signal groups are serviced is determined offline. This is done arbitrarily in the case of unoptimized control or based on historic traffic data in the case of optimized control.

Optimized fixed-time control refers to a fixed-time controller where the duration of each phase and the order of servicing phase groups is optimized offline with respect to historic traffic flow data. However, traffic flow patterns change over time and therefore the performance of the optimized fixed-time controller degrades over time.

Adaptive signalized intersection control tries to solve this problem by adapting the controller's schedule to the recent changes in traffic flow. The controller is equipped with an algorithm that determines the configuration of several control parameters based on the intensity of traffic flow on its links in the recent history. In practice a history of approximately 15 minutes is taken into account. An example of such a configuration parameter is the duration of each phase. SCOOT [42] is one of the most widely deployed examples of an adaptive controller.

Actuated control in contrast does not look at the historic arrival flows but rather acts directly based on the intersection's detectors. Intersections operated by an actuated controller are usually equipped with:

- stopline detectors located at the stopline to detect vehicles waiting at the intersection
- queue detectors located approximately 10-50 meters away from the stopline to detect the presence of longer queues at the intersection
- optionally also arrival detectors located approximately 80-100 meters away from the stopline to detect the arrival of vehicles

The actuated controller is equipped with a finite state machine that determines how to respond to the traffic detected at the intersection. It is more responsive than the previously described controller types as it evaluates and adjusts its planned schedule on a seconds timescale. The number of possible corrections and the timescale on which these adjustments can take place is called a controller's degree of flexibility.

A model predictive controller is similar to an actuated controller in terms of flexibility. The difference is that a model predictive controller uses a model to predict the vehicle arrivals and departures at the intersection. This model enables the controller to optimize the traffic flow beyond the detection horizon described above. This leads to improved performance compared to the actuated controller given a sufficiently accurate prediction model.

Table 1.1 shows an overview of the different controller types and their features. It also shows what the ideal controller would look like.

The above description of the different controller types is still very generic and there exist many different implementations of each. An actuated

Controller type	Online optimization	Flexible	Optimize beyond detection horizon	100% predictable
Fixed-time	✗	✗	✗	✓
Adaptive	✓	✗	✗	✗
Actuated	✓	✓	✗	✗
Model predictive	✓	✓	✓	✗
Ideal	✓	✓	✓	✓

Table 1.1: Overview of different traffic control mechanisms and their features. A checkmark means that the controller has the particular feature. A cross means the controller does not.

controller can for example be implemented with or without the freedom to change the order of phase servicing, which impacts both performance and system complexity.

Another important distinction is the way in which network optimization is performed. A combination of local optima can lead to an unstable network [31], an example of such a situation occurs when an intersection's outflowing link is completely occupied but the intersection still tries to feed more traffic into the link based on its local optimization. This can lead to a gridlock that completely blocks the intersection.

Network optimization is used to prevent the likelihood of such events occurring. A well-known example of such measures are green waves, where a series of connected intersections are synchronized in order to give passage without stops along a particular road segment. Such controllers are called coordinated controllers. In order to make coordination possible, traditional signalized intersection controllers operate based on cycles. Within a cycle each signal group will be serviced. The service duration of each phase group is called the split time. An offset is then used to synchronize the intersection controllers. Model predictive controllers can perform network optimization by including the interaction between intersections in the prediction model.

In practice hybrid forms of the different controller types might be used. Road owners might for example deploy a fixed-time/adaptive controller during the peak hours (when most traffic arrives at the intersection, usually in the morning and evening) to facilitate a green wave while a traffic actuated controller is used during the non-peak hours (outside the peak hours).

Much more can be said about the domain of signalized intersection control but the above description should provide sufficient insight in order to understand and evaluate the concepts described in this thesis.

1.1.3 The transition to autonomous vehicles

The introduction of connected vehicles (vehicles that are capable of communication with other vehicles, infrastructure or the internet) and fully autonomous vehicles will change the traffic landscape in the nearby future. This section discusses some of the developments and challenges in the signalized intersection domain related to the rise of these new technologies.

The development of fully autonomous vehicles is underway. Prototypes by companies like Waymo and Uber are already being tested on the road [44, 48] and it is only a matter of time until these vehicles reach a production stage and society can enjoy the promised benefits of this technological advancement [18]. This development will cause major disruption in the way traffic is organized and infrastructural changes are required to get the most out of this new technology.

A crucial question that still needs to be answered is how to transition from human-controlled vehicles to fully autonomous vehicles while maintaining safety standards and against minimal economic cost? Intersections form an important challenge here. At some point human-controlled vehicles might not be allowed on the road anymore and autonomous vehicles will be able to communicate in order to achieve safe passage through an intersection without the use of signals [15, 16, 17] or even completely without the use of an intersection controller [51]. Dresner, Stone and VanMiddleworth show that this should lead to great benefits in traffic flow efficiency [15, 16, 17, 51]. However, during the transition period an intersection controller is still required to mediate between vehicles and it should do so with the greatest achievable efficiency.

1.1.4 Advanced driver assistance systems

During the transition period connected vehicle technology can already lead to great improvements in traffic flow [2, 3, 12, 36, 37, 47, 52]. To facilitate these opportunities the Dutch government recently started the development of a 'smarter' infrastructure. They initiated the Talking Traffic project in collaboration with the industry [5]. The focus of the project is to develop a state-of-the-art infrastructure to vehicle (I2V) and vehicle to infrastructure (V2I) communication network. The technology has been developed and is currently in the process of pilot deployments.

The Talking Traffic project is very wide in scope but in particular use-case 4 of the project is of interest for signalized intersection control. Use-case 4 of the project aims to improve traffic flow through advanced driver assistance systems (ADAS) by informing road users approaching an intersection. One type of ADAS are systems that communicate the signal states and times of signal changes, also called signal phase and timing (SPaT) information. Examples are:

- Showing the time to green/red (T2G/R), which is a countdown until the signal changes to green or red. Figure 1.1 visualizes the implementation of T2G/R.

- Providing a green light optimal speed advice (GLOSA), which is a speed advice that makes sure a driver will arrive at an intersection when the signal is green [6]. Figure 1.2 visualizes the application of GLOSA.

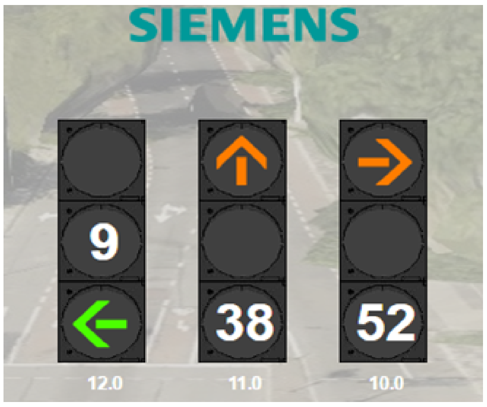


Figure 1.1: Visualization of time to green/red (T2G/R) countdowns.

Showing T2G/R countdowns at traffic lights aims to reduce the capacity drop during queue clearance when the light turns green. In theory vehicles should exit an intersection at full capacity. However, because drivers cannot anticipate when the light will turn green there is a response delay in practice. This delay is magnified throughout the queue leading to queue clearance at only 75% of the road's capacity [8].

Providing road users with GLOSA information enables them to adjust their driving profiles to avoid fuel inefficient cyclic driving, i.e. they can smoothen their trajectory and avoid unnecessary acceleration and deceleration.

1.2 Problem statement

From Sections 1.1.3 and 1.1.4 it can be concluded that road owners are facing two main challenges at signalized intersections in the near future:

1. Improve traffic flow through the use of advanced driver assistance systems use cases such as time to green/red and green light optimal speed advice in a safe way.

2. Improve traffic flow through the transition from a completely human-controlled to a completely autonomous vehicles traffic mix.

An important note is that both should be done in a safe and cost-effective manner.

At first glance challenge 1 might not seem too complicated. However, as explained in Section 1.1.2, the state-of-the-art actuated traffic controllers adjust their schedule at the last second based on the current traffic state.

Figure 1.2: Visualization of green light optimal speed advice (GLOSA) [49]. The vehicle is approaching a red signal at an intersection and receives a speed advice that ensures it arrives when the signal has turned green.

This makes sense as this flexibility improves the traffic flow. However it also makes it impossible to provide stable SPaT information for T2G/R or GLOSA applications. This can lead to unsafe situations and user frustration. As a matter of fact, as Table 1.1 shows, the only controller type that can provide reliable SPaT information is a fixed-time controller. As discussed in Section 1.1.2, a fixed-time controller is the least flexible and consequently worst performing type of signalized intersection controller. Therefore there is a need to develop a control mechanism that leads to SPaT stabilization while maintaining traffic flow efficiency.

Furthermore, given challenge 2 it would be preferred if the solution for challenge 1 would be able to accommodate the transition from human drivers to fully autonomous vehicles.

The problem can then be summarized as:

> Developing an intersection controller that can stabilize signal phase and timing information and is capable of optimizing traffic flow for human drivers, autonomous vehicles and a mix of the two in a safe and cost-effective way.

To guarantee stable SPaT information the traffic light controller must schedule and fix phase groups at least some time before the phase group is serviced. The only way to achieve this without inherently compromising performance is using a model predictive controller. The hypothesis under investigation in this thesis is that the stated problem can be solved using a model predictive controller.

1.3 Research question

Following the hypothesis from Section 1.2 the research question is formulated as:

> Can a model predictive control-based signalized intersection controller lead to stabilized signal phase and timing information that enables advanced driver assistance systems use-cases while maintaining traffic flow at least as good as a state-of-the-art actuated traffic controller?

Subquestions related to this research question are:

1. Can a model predictive controller outperform a state-of-the-art optimized actuated controller in terms of traffic flow?
2. Can data-driven approaches improve the performance of a state-of-the-art heuristics-based model predictive controller in terms of traffic flow?
3. Can a model predictive control-based intersection controller lead to stable enough signal phase and timing information to make safe application of advanced driver assistance systems possible?
4. Does the application of T2G/R and GLOSA improve performance in terms of traffic flow efficiency considering the cost required to make these applications feasible?

1.4 Research scope

When answering the research question the practical restrictions of legal and road safety boundary conditions are taken into account. Industry standards are used to evaluate whether the traffic light controller meets the requirements for practical application.

Throughout this thesis there is no distinction between detector types. The assumption is that the necessary detectors are in place and functional within industry fault tolerance margins.

The study is also not concerned with the means of getting ADAS information to the road users. The traffic light controller works independent of whether the road owner decides to use road signs, in-car displays, smartphones or another method to inform the users.

Furthermore, the study focuses on standard vehicle road users, i.e. cars and trucks. The impact of pedestrians, bicycles, public transport and blue vehicles (police, ambulance and firetrucks) is not investigated although the traffic light controller should be able to deal with these types of road users.

1.5 Report outline

To answer the research question a custom traffic light controller was developed called. The controller is called DIRECTOR (Data-driven Intersection and Road Environment Controller for Traffic Optimization in Realtime). The remainder of this report describes the design, development and evaluation of DIRECTOR.

After this introduction chapter, Chapter 2 will explore the related work in literature in more depth. Next, Chapter 3 explains the conceptual design of DIRECTOR. Chapter 4 explains the design of a new short-term traffic flow prediction algorithm. Then chapter 5 elaborates on the theory behind and implementation of DIRECTOR. Chapter 6 describes the extension of DIRECTOR's implementation from Chapter 5 to enable advanced driver assistance systems. Chapter 7 outlines the experimental setup used for the execution and evaluation of all experiments. The results of these experiments can be found in Chapter 8 followed by the conclusions and recommendations for future work in chapter 9.

Chapter 2

Related work

This chapter covers advancements in literature related to this thesis. In particular it covers the areas of signal phase and timing (SPaT) prediction, short-term traffic flow prediction, model predictive control approaches for signalized intersections, dealing with errors and uncertainty in traffic flow detection and advanced driver assistance systems (ADAS).

2.1 Signal phase and timing prediction

As the prediction of SPaT information is crucial for ADAS like GLOSA and T2G/R many papers have been published on the topic.

2.1.1 Floating car data based approaches

One approach found in literature is to predict SPaT information based on floating car data (FCD). Floating car data is data generated by road users sharing their location, e.g. for navigation purposes. Axer & Friedrich followed this approach when they recently attempted to estimate SPaT information based on low-frequency (15 seconds sampling interval) FCD [3]. Axer & Friedrich base their SPaT estimation on historic vehicle trajectories, which they split into crossing and stopping trajectories. They aggregate these trajectories into frequencies for a given time interval. The green time is then estimated by maximizing the consistency of crossing and stopping frequencies.

A major limitation of their approach is the requirement of a similar cycle length and phase sequence for the same time period of a workday. This limits the applicability of their approach. According to the authors this limitation is reasonable since actuated traffic control behaves like fixed-time control in situations with a constant high traffic demand.

An important contribution of Axer & Friedrich to the FCD-based research is that their experiments are performed using micro-simulations that take positioning errors caused by GNSS (e.g. GPS) into account [3]. The results of their simulations of a peak hour show that their average prediction errors for green times are approximately 2 and 6 seconds for the start and end times, respectively.

Fayazi et al. used sparse bus location data (every 200 meters or 10-80 seconds) to infer the fixed-time schedule of intersections in a crowded area of San Francisco (USA) [19]. Using statistical methods based on three months of historic data they are able to predict the green start times with a root mean squared error (RMSE) of 2.5 seconds and a maximum error of approximately 8 seconds. Their evaluation was done by conducting a real-time experiment at the actual intersection. A nice feature of their statistical approach is that the performance should improve when more FCD sources become available in the future.

2.1.2 Vehicle detection based approaches

Another approach is to predict the SPaT information using Markov chains based on combinations of signal states and corresponding detector counts. Barthauer & Friedrich evaluated the usefulness of such an algorithm for several I2V applications (GLOSA, a T2G countdown signal and start-stop assistance) [4]. The algorithm under evaluation is originally developed by Menig et al. [38]. Barthauer & Friedrich evaluate the performance of the algorithm as the probability that the predicted signal switching time is within the error bound for a particular application. The authors conclude that the algorithm does not meet the requirements for the described applications. These requirements were defined theoretically and therefore might be even stricter in practice. They also note that the risk of erroneous predictions is still unclear.

2.1.3 Other approaches

Mahler & Vahidi used a simple probabilistic mechanism to predict green times based on a linear relationship between the average green time, the average red time and the current state of the signals [36, 37]. A limitation of their prediction mechanism is the need for a constant cycle length. Unfortunately they don't provide an evaluation of their prediction quality since their papers are focused on fuel efficiency improvements through the application of GLOSA. Their prediction mechanism is a method to achieve this and not evaluated on its own. The authors note that information on how long the signal has been red or green already could improve their prediction accuracy.

Recently, several master theses have been dedicated to SPaT prediction. Khakhutskyy decomposed the SPaT prediction problem into five subproblems [26]:

- Traffic flow prediction (regression)
- Phase length prediction (classification)
- Phase activity prediction (classification)
- Next phase prediction (classification)
- Combining the phase predictions

Khakhutskyy performed the phase length prediction with k-nearest neighbors (kNN) and a feedforward neural network (FFNN). The phase activity prediction (will a phase end within the next couple of seconds) is done using a FFNN and decision trees. Decision trees were used for the next phase prediction. Khakhutskyy's model achieves a RMSE of less than 5 seconds for coordinated-actuated traffic light control and 11 seconds for fully-actuated mode. He concludes that this is sufficient for ADAS that require longer prediction horizons and lower accuracy like GLOSA, although the opinions on this matter differ in literature and practice.

Boyd attempted SPaT prediction using a classification and regression tree (CaRT) algorithm and a long short-term memory (LSTM) network [7]. She generated traffic flow and SPaT information using a simulator and then used the simulation outputs to train and evaluate the models. She concludes that the LSTM model outperforms the CaRT model with a top accuracy of 97.78%. However, it remains unclear how exactly this metric is defined.

Scheepjens used support vector regression (SVR) to predict the switching times (from red to green and green to red) of actuated traffic light controllers [43]. He created two separate models per signal group, one for the prediction of switching to green (best mean absolute error of 0.85 seconds and 99% within 4 seconds) and another for the prediction of switching to red (best mean absolute error of 2.42 seconds and 81% within 4 seconds).

2.2 Short-term traffic flow prediction

Helmy provides a great summary of the main analytical models for traffic flow progression [21]. She highlights the three most important models:

- Lighthill & Witham's fluid dynamic traffic model [34]
- Pacey's diffusion model
- Robertson's platoon dispersion model [41]

Pacey's model is a modification of Lighthill & Witham's model and Robertson's model is a modification of Pacey's model. Each modification fixes an invalid assumption of the older model. It is therefore not surprising that

Robertson's model has been proven to be the most accurate of these models. His model describes the downstream arrival flows as a function of the upstream departure flows and the travel time between the upstream and downstream locations. It requires accurate calibration of parameters in order to model the traffic progression properly. Many papers have been published attempting to simplify this calibration process with little success. For an overview of these papers see Helmy's thesis [21].

The analytical models suffer from a major assumption, which is the conservation of vehicles from upstream to downstream. This limits the application of these models at roads where traffic might leave the road (through a so-called sink) or enter the road (through a source) mid-way. Hence, a data-driven method might be more suitable for traffic flow prediction. Khakhutskyy therefore designed a recurrent neural network (RNN) to predict the amount of traffic flowing from one intersection to another. However due to a lack of accurate data he was unable to properly evaluate the model's performance.

Helmy was able to design and evaluate a RNN for traffic flow prediction [21]. Her model takes the upstream departure flows, the presence of a queue downstream, the traffic split between downstream lanes, the states of the downstream signals and information about the day of the week and time of the day as inputs. The model then outputs the arrival flows downstream. The model is trained with backpropagation through time (BPTT). Using a case study with real data Helmy showed that her RNN outperforms Robertson's model in terms of prediction accuracy. Her model especially outperforms the analytical model in congested traffic conditions, which is something most analytical models are not designed to cope with.

2.3 Long short-term memory networks

Recurrent neural networks have proven to be great tools for various applications [39]. Helmy confirmed this for the challenge of short-term traffic flow prediction [21]. However, for many purposes traditional RNNs are far from perfect. Hence, researchers have been looking for ways to boost their performance. In particular the invention of long short-term memory led to a great improvement.

Christopher Olah wrote a blog about the basics of recurrent neural networks (RNNs) and long short-term memory (LSTM) networks. This is a great introduction for the reader unfamiliar with these concepts [39]. The remainder of this section briefly summarizes the problem of conventional recurrent neural networks and how long short-term memory solves it.

Hochreiter showed that conventional RNNs trained with backpropagation through time (BPTT) are unable to learn long-term time dependencies. With BPTT the error signals flowing backward in time tend to either blow up, leading to oscillating weights, or vanish, leading to loss of information. This is because the backpropagated error over time exponentially depends on the size of the weights [23].

Hochreiter & Schmidhuber found a solution to this problem in the form of long short-term memory [24]. The intuition behind their concept is to replace (some) hidden neurons with memory cells. Each memory cell internally has a constant error flow, which enables the memory cell to store inputs over a longer period of time. The memory cells have input and output gates (propagation functions with weights that can be trained) that determine whether the content of the memory cell should be changed or propagated, respectively. Through training of the memory cells' weights, long-term dependencies can be recognized by the LSTM network. Through experiments with both short and long time-lag problems Hochreiter & Schmidhuber show that LSTM networks clearly outperform conventional RNNs in terms of accuracy and required training time.

2.4 Model predictive intersection controllers

Model predictive control (MPC) is an online model-based control approach in which a prediction model and optimization function are used to determine the control actions that optimize a given performance criterion over a given time horizon subject to given constraints [10, 35]. De Schutter et al. provide a good introduction to the application of MPC in the domain of traffic network optimization [13]. In the remainder of this section several implementations of a model predictive controller for signalized intersection control are explored.

2.4.1 Centralized approaches

Van Katwijk et al. describe a taxonomy of MPC algorithms for traffic control [50]. Combining the advantages and disadvantages of several algorithms from literature they propose and evaluate a hybrid algorithm. Their evaluation is done using microsimulations of a single intersection and compared to a traffic actuated controller. The authors conclude that predictive control can improve the traffic flow quite substantially (28-36% for their simulations) but that there is an important trade-off between computational complexity and performance, which depends on the geometry of an intersection, the demand at an intersection and the prediction horizon. This trade-off is important given the real-time computation requirement of traffic light controllers.

2.4.2 Distributed approaches

SURTRAC

Xie et al. found a way to reduce the computational complexity of the MPC approach to intersection control [54]. Their idea is to reduce the state space and transform the problem into a single machine scheduling problem. The authors model vehicle flows based on a simple model and then aggregate arrivals to create clusters of arriving vehicles that are handled as indivisible jobs in a forward-recursive scheduling algorithm. Their state reduction strategy leads to a computational complexity that is polynomial in the prediction horizon. The performance of their algorithm is evaluated in real-world scenario network simulations. The results show a maximum reduction of approximately 20% in terms of travel time delay compared to the current fixed-time coordinated signal timing plans.

Xie et al. acknowledge that their implementation has a prediction horizon that is limited by the travel time between two intersections [54]. Therefore they propose an extension in the form of a coordination mechanism between neighboring intersection in a follow-up paper [53]. Their solution is that intersections communicate their planned departure flows to their downstream neighbors, hence increasing the prediction horizon of those downstream intersections. To guarantee network-wide stability downstream each intersection monitors if its queues might spill back onto upstream intersections. When that happens the intersection changes its schedule to service those queues first.

The work from Xie et al. has been developed for real-world application and is called SURTRAC (Scalable URban TRAffic Control). A live implementation of the SURTRAC system has been deployed and evaluated in Pittsburgh (Pennsylvania, USA) by Smith et al. [45, 46]. The test site consists of nine intersections with link lengths ranging from 90 to 500 feet between them (on average 272 feet). SURTRAC is compared to the original traffic light controller that operated as an offline-optimized coordinated actuated controller during the peak hours and as a simple actuated controller during the remainder of the day. Evaluation of the system was done using drive-through runs along the 12 highest volume routes in the network. Each route was taken 12 times per scenario, i.e. 144 runs for the base case and 144 runs for SURTRAC. Based on these measurements the authors conclude that SURTRAC improved the traffic flow by approximately 25-40%. However, the reliability of these numbers could be questioned given the small sample size of their measurements.

Self-controlling traffic lights

Lämmer et al. describe the queueing process at an intersection as a hybrid dynamical system using a combination of first order differential equations with predicted vehicle arrival flows as the main input [30]. Their method enables the real-time computation of the queue clearance time and expected cumulative travel time delay of a particular lane. Consequently, they can optimize the traffic light control with regards to the cumulative travel time delay. An evaluation of the application of their methods is not present in their original paper [30].

In a follow-up paper Lämmer & Helbing build on this concept to create a mechanism for self-controlling traffic lights in order to optimize for travel time delay based on the predicted short-term arrival flows [31]. The traffic flows are predicted according to Lighthill and Whitham's fluid dynamic traffic model [34], which is sufficiently accurate for short-term urban road traffic flow predictions according to the authors. Lämmer & Helbing propose a non-periodic optimization technique for individual intersections that create locally optimal schedules. They note that a combination of local optima can lead to instability throughout a network [28]. The authors therefore also propose a stabilization mechanism. This mechanism makes sure that non-dominant links are sufficiently serviced and prevents spillbacks at the start of any link. Their final proposed control algorithm is a hybrid of this optimization and stabilization mechanism and leads to guaranteed stability in a traffic network as long as the incoming flows remain smaller than the saturating flow (the maximum road capacity). Lämmer & Helbing evaluate their mechanism using artificial simulations pushing their system at the boundary conditions. They compare their results to a fixed-time schedule where the offsets and cycle lengths are adjusted locally, i.e. a semi-adaptive control mechanism. Their results show that their new control mechanism significantly outperforms the semi-adaptive control algorithm in terms of average queue length and thus also cumulative travel time delay.

Lämmer & Helbing then developed a generalized version of their stabilization mechanism [32]. They propose a supervising process that adds constraints to the local optimization strategy such that it will always perform as least as good as a fixed-time strategy would. The general idea behind the stabilization mechanism is to increasingly restrict the freedom of the local optimization strategy as queues mount for any direction, hence guaranteeing in-time servicing of each direction within the bounds of a fixed-time controller. Lämmer & Helbing also show the results of a simulation of a complex network of 13 intersections in the center of Dresden, Germany [32]. They compare the result of their generalized stabilization mechanism and the optimization mechanism from their previous work [31] with the current adaptive coordinated control system, which includes green waves and public transport priority. The authors find that their system significantly outperforms the current system in terms of average travel time delay for all modes of transport (-56% for public transport, -9% for cars and trucks and -36% for pedestrians).

To show the impact of the previously described methodology Lämmer performed a live experiment controlling two of the previously simulated intersections in Dresden [29]. In this study Lämmer compares the current VS-PLUS controller to his self-controlled mechanism. Both control algorithms were observed for 3 weekdays. The detection horizon of vehicle flows was approximately 250 meters and the prediction horizon for traffic estimation was set to 2 minutes. The average vehicle flow was used for prediction beyond the detection horizon. The study's results are promising as they show a significant reduction in average travel time delay (-37.8% for pedestrians, -33.6% for bicycles, -80.4% for public transport and -38.4% for cars and trucks) compared to the base case, despite a 10% increase in traffic volume compared to the base case. Another interesting observation is that the VS-PLUS green wave leads to more stops than the self-control mechanism. Of course this experiment does not provide a general proof that the self-control methodology is always superior to other control mechanisms, but the results are promising and provide a starting point for further exploration.

2.4.3 Other approaches

Microsimulation based control

Brouwer & Van der Bijl developed a method for predictive traffic light optimization through a microscopic real-time traffic simulation model [9]. They model the progression of unique individual vehicles based on a fusion of traditional sensors (e.g. loop detectors or cameras) with FCD. Using this predicted progression they are able to optimize singular intersections or even entire networks. Simulations of their methodology show a 40% reduction in cumulative travel time delay(from 30 hours to 18 hours) for an intersection in Tilburg, the Netherlands, and a 25% reduction of delay for a saturated intersection in Rotterdam, the Netherlands, during the evening peak hours. They also claim to be able to deal with priority for special services such as public transport and to enable driver assistance use-cases. Unfortunately the work by Brouwer & Van der Bijl lacks a detailed description of the methods used, nor do they provide an evaluation of the reliability of their SPaT information for driver assistance use-cases.

Floating Car Data-based conrol

Krajzewicz et al. describe the major achievements of the COLOMBO project (cooperative self-organizing system for low carbon mobility at low penetration rates) [27]. Among other things the project aimed at optimizing traffic light control based on FCD at low penetration rates. Their schedule selection method is inspired by the ant algorithm, a method from the field of artificial intelligence. The authors claim that their method leads to travel time delay reduction compared to fixed-time traffic lights at 10% penetration and that similar performance to actuated traffic lights is achieved at 25%

penetration. At higher penetration rates their algorithm outperforms current actuated traffic lights. These results are all generated using microscopic simulations.

2.4.4 Summary

Table 2.1 summarizes the controllers covered in this section based on some of the key features.

Paper	Guaranteed network stability	Fixed computation time	Feasible today
Brouwer & van der Bijl [9]	Unknown	✗	✓
SURTRAC [45, 46]	Unknown	✗	✓
Krajzewicz et al. [27]	Unknown	✗	✗
Van Katwijk et al. [50]	✓	✗	✓
Lämmer et al. [29, 30, 31, 32]	✓	✓	✓

Table 2.1: Overview of discussed literature on model predictive intersection controllers. A checkmark means that the controller has the particular feature. A cross means the controller does not.

2.5 Dealing with errors and uncertainty

Both actuated and predictive traffic control algorithms rely heavily on sensors. Many types of sensor errors can occur in practice, examples found in literature are: double lane detectors that count only one vehicle instead of two or more [21], broken detectors that overestimate or underestimate the flow [21], sharply turning vehicles being missed by the detector [45], large vehicles being double-counted [45], influence of weather [45], etcetera. This section explores how systems can deal with some of these errors. Furthermore, it discusses how systems in literature deal with predictions beyond the upstream detection horizon, i.e. further in time than the farthest upstream detector.

2.5.1 Arrival flow correction

In order to correct for underestimation of the queue length SURTRAC uses two mechanisms [45]. Firstly, they use the link arrival/departure-ratio (AD-ratio, the number of arrivals divided by the number of departures). The idea is that (hypothetical) sources and sinks can lead to an AD-ratio < 1 if vehicles have been missed. To solve this problem the arrival flow is divided by the AD-ratio.

The queue clearance time is the time it takes to service an entire queue. SURTRAC's second mechanism is using an elastic queue clearance time t_{QC}, which is proportional to an elasticity ratio r_{QC}^{ela}. r_{QC}^{ela} is defined as a sigmoid function of the queue length [45].

2.5.2 Prediction beyond the upstream detector horizon

Lämmer et al. use the average historic traffic flow for each link as a best guess for arriving vehicles beyond the upstream detectors [31]. SURTRAC uses the expected scheduled departures from the upstream intersections [46, 45]. SURTRAC intersections communicate to all neighboring intersections therefore implicitly propagating information in a distributed fashion.

2.6 Advanced driver assistance systems

Advanced driver assistance systems enable possible travel time reduction, fuel savings and greenhouse gas emission reduction. This sections explores the requirements and benefits of such systems.

De Nunzio et al. developed a method that can calculate the sub-optimal energy-efficient route for each individual vehicle in real-time when the green time intervals of a sequence of upcoming traffic lights is known in advance [12]. In microscopic evaluations of an artificial traffic network they showed that a penetration rate of 100% leads to a 28.5% reduction in energy consumption and 4% reduction in average travel time. These results could be improved even more when V2V communication would be taken into account. Major limitations of their work are the assumption of free flow traffic conditions and up-front knowledge of reliable far future SPaT information (several intersections away).

Asadi & Vahidi utilize SPaT information to enable the concept of predictive cruise control [2]. Their solution is based on the concept of adaptive cruise control where a vehicle attempts to maintain a speed $\mathrm{v}_{\mathrm{target}}$ while keeping a safe distance from any obstacles (other vehicles) in front of the vehicle. They extend the adaptive cruise control by taking SPaT information into account to optimize the vehicles trajectory for minimal idling at traffic lights and minimal fuel consumption. Asadi & Vahidi evaluate their solution using a simulation of a real traffic network with historic SPaT data. They estimate the fuel consumption using a detailed powertrain model. Their results show a 4.2% reduction of travel time and a 41.8% reduction of fuel consumption compared to traditional adaptive cruise control. The authors note that an extensive statistical analysis is still required to determine the attainable gains of their approach and that the influence of different traffic conditions could impact their results.

Stevanovic et al. investigated the relationship between traffic actuation, signal timing optimization and the application of GLOSA [47]. Without taking safety issues of fluctuating GLOSA information into account they evaluate the impact on travel time, number of stops and fuel efficiency for eight scenarios (all possible combinations of with and without traffic actuation, signal timing optimization and GLOSA) using the VISSIM microscopic simulation software [20]. The authors conclude that:

- Signal timing optimization remains of vital importance.
- The application of GLOSA is beneficial for fixed-time signals.
- GLOSA is not beneficial for actuated signals until an accurate SPaT estimation method is found.

They found that without a reliable prediction algorithm the application of GLOSA decreases the performance.

Mahler & Vahidi used a relatively simple prediction algorithm (as described in section 2.1.3) to enable a GLOSA implementation that should lead to optimal fuel efficiency [36, 37]. Their speed advice is based on a cost function that is the weighted sum of accelerations, decelerations and idling time at a red light. With custom simulations of fixed-time traffic lights they show that for particular configurations a 61% and on average a 16% improvement of fuel efficiency is possible using their setup. Simulations of a series of real traffic actuated intersections based on historic data show a 6% average improvement of the fuel efficiency. A clear limitation of their evaluation is that they ran simulations with singular cars, hence excluding the influence of other traffic.

Wan et al. investigated this influence of traffic [52]. They developed a fuel efficiency optimizing speed advisory system (SAS) and evaluated it for different penetration rates and different traffic densities through simulations of a series of intersections controlled by fixed-time traffic light schedules. The authors show that their mechanism achieves a fuel efficiency improvement of approximately 25% at the cost of a 2-8% decrease in traffic flow. This indicates that traffic flow and fuel optimization are not necessarily the same problem. Furthermore, the authors show that increasing penetration rates further improve the fuel efficiency of vehicles that do not receive a speed advice (at the cost of increased travel time). A limitation of the work done by Wan et al. is the lack of an evaluation in the case of traffic actuated control.

2.7 Conclusion

From the literature study described in this chapter several conclusions can be drawn:

- To the best of the author's knowledge no signal phase and timing prediction method exists that is capable of enabling advanced driver assistance systems without reducing the performance of the control algorithm.
- To the best of the author's knowledge the state-of-the-art in data-driven short-term traffic flow prediction is Helmy's recurrent neural

network [21]. Furthermore, this model has been proven to outperform a state-of-the-art analytical prediction model.

- Long short-term memory networks have been proven to outperform regular RNNs in many applications.
- Model predictive control has a great potential for traffic control but most MPC approaches suffer from a trade-off between computational complexity and real-time response.
- Distributed MPC approaches like SURTRAC and the self-organizing traffic light provide a scalable solution for traffic control. Although each has its own limitations.
- Different traffic control algorithms are tough to compare as each algorithm is evaluated in a different environment with a unique base case controller.
- Advanced driver assistance systems have a great potential to reduce the cumulative travel time delay, improve driver comfort and reduce greenhouse gas emissions at signalized intersections. However, that is only the case when reliable SPaT information is available over a sufficiently long prediction horizon.

Chapter 3

Conceptual design

This chapter describes the conceptual design of a signalized intersection controller called DIRECTOR (Data-driven Intersection and Road Environment Controller for Traffic Optimization in Real-time). The design choices are based on the observations from the literature review in Chapter 2.

From the literature study it is clear that distributed model predictive control has the greatest potential for further improvement of signalized intersection control in a scalable way. However, the current distributed controllers rely heavily on heuristics and assumptions regarding the flow of traffic. Research has shown that data-driven traffic flow methods can outperform analytical models. It therefore makes sense to create a distributed predictive controller that utilizes a data-driven traffic flow prediction model.

Advanced driver assistance systems (ADAS) have a great potential to improve traffic flow and reduce greenhouse gas emissions. However due to the lack of a reliable signal phase and timing (SPaT) prediction method it cannot be applied in practice. In order to safely use advanced driver assistance systems like T2G/R and GLOSA 100% accurate SPaT information is required. This is only possible when the the schedule is fixed ahead of time. Doing this efficiently requires insight in the future traffic situation. Since predictive controllers inherently have insight into the future situation it makes sense to develop a predictive controller to enables ADAS.

The conceptual design of such a predictive intersection controller is explained in Figure 3.1 and contains the following building blocks:

1. Short-term traffic flow prediction model (Chapter 4)
2. Queue length prediction model (Chapter 5)
3. Predictive traffic light controller (Chapters 5 and 6)
4. Advanced driver assistance systems module (Chapter 6)

Detailed descriptions of the building blocks are described in the next chapters.

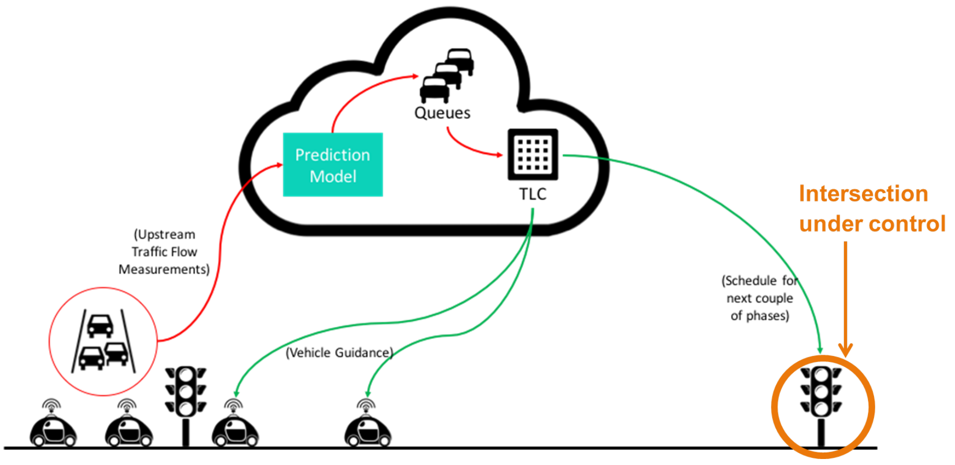

Queues
Prediction Model
TLC
Intersection under control
(Upstream Traffic Flow Measurements)
(Schedule for next couple of phases)
(Vehicle Guidance)

Figure 3.1: Illustration of DIRECTOR's conceptual design [21]. Traffic moves from the left intersection (upstream) to the right intersection (downstream). The upstream departures are counted and used as input for a prediction model. The prediction model predicts the evolution of queues downstream. Using the predicted queue length over time a traffic light controller (TLC) optimizes the schedule. This schedule is fixed some time before it is actually executed. That guarantees knowledge of the future SPaT information and thus enables advanced driver assistance systems.

DIRECTOR's conceptual design, as outlined in Figure 3.1, is also futureproof. Because of the data-driven nature of the system its performance should only increase towards the future with more and better data becoming available. Figure 3.2 shows this self-enforcing cycle.

The design enables ADAS, which leads to two consequences. Firstly, the driving behavior will become more predictable, which leads to traffic flow patterns that are easier to predict. Secondly, the use-cases will attract more users to connected vehicle technology, which leads to an increase in available, more detailed data. This increase in available, accurate data again leads to improvements in the flow prediction. Hence, DIRECTOR should be able to deal with the transition from human drivers to connected vehicles to autonomous vehicles without big, expensive changes to the infrastructure. Furthermore, its performance will improve along the way.

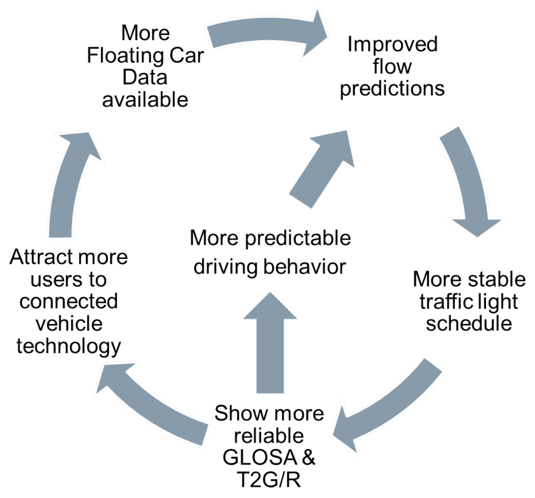

More Floating Car Data available
Improved flow predictions
More stable traffic light schedule
Show more reliable GLOSA & T2G/R
Attract more users to connected vehicle technology
More predictable driving behavior

Figure 3.2: Schematic drawing of the self-enforcing cycle of DIRECTOR's conceptual design from Figure 3.1, which leads to a stable traffic light schedule that enables GLOSA and T2G/R.

Chapter 4

LSTM network design for short-term traffic flow predictions

This chapter describes the implementation of the arrival flow prediction model. The model is inspired by the state-of-the-art in short-term traffic flow prediction, which is the recurrent neural network (RNN) architecture by Helmy [21].

4.1 State-of-the-art prediction model architecture

Helmy discovered that for short-term traffic flow prediction a RNN can outperform analytical models. She identified five input variables of influence:

- Time
- Departure flows (number of vehicles leaving upstream per time bin)
- The presence of queues downstream
- Arrival flows (number of vehicles arriving downstream per time bin)
- The signal states downstream (red, amber or green)

The output of the model is the number of vehicles arriving during a time bin Δt seconds from now. Δt is referred to as the prediction horizon.

Helmy included each of the input variables for a particular number of timesteps back in time. This number of timesteps can differ per variable. A downside of Helmy's approach is that it requires a long brute-force process to identify the best configuration of timesteps per variable.

4.2 LSTM architecture

In order to create a prediction model that is more accurate and user-friendly than Helmy's model, the model proposed in this thesis makes use of long short-term memory (LSTM) to learn the contribution of each variable over time based on the training data. Figure 4.1 visualizes the architecture of this model[1].

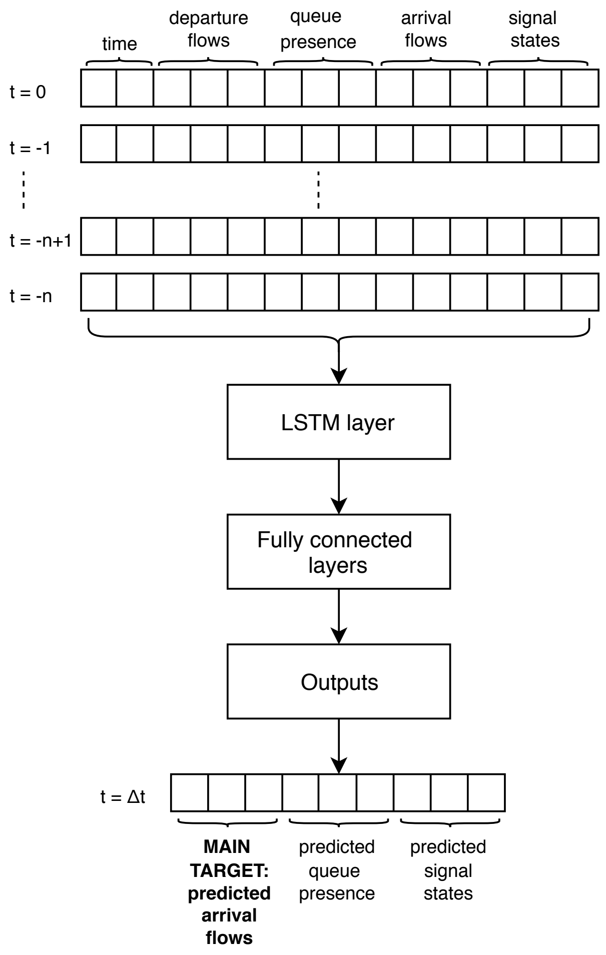


Figure 4.1: Schematic drawing of the proposed LSTM architecture. Each variable is included for n timesteps. The LSTM layer learns the importance over time for each variable.

The model includes all variables for n timesteps back in time and the LSTM layer learns the dependencies automatically based on the training data. As described in Chapter 2 this approach has been proven to outper-

[1]Appendix A visualizes the architecture of Helmy's RNN model for comparison

form conventional RNN architectures for many applications [39].

Chapter 5

Predictive controller design

This chapter describes the design of DIRECTOR's model predictive control algorithm. It is inspired by the self-organizing traffic light concept developed by Lämmer et al. [29, 30, 31, 32]. The algorithm is redesigned from the ground up to address some of the limitations, incorporate the LSTM traffic flow prediction model and enable advanced driver assistance systems.

The chapter covers the design by starting from the basic intuition behind the methodology and then continuously adding features that improve the model's reflection of reality.

5.1 Intuition behind DIRECTOR

The goal of DIRECTOR's algorithm is to minimize the average travel time delay per vehicle within the boundary conditions set by the regulating authorities and road owners. The intuition behind DIRECTOR is to use the cumulative travel time delay in the near future as the value of the cost function. The cumulative travel time delay is the sum of the travel time delays of all vehicles.

The travel time delay D_{OD} accumulated for a certain origin-destination pair OD from time t_{start} to t_{end} is given by Equation 5.1,

$$D_{OD}(t_{start}, t_{end}) = \int_{t=t_{start}}^{t_{end}} \rho_{OD}(t) \tag{5.1}$$

where ρ_{OD} is the number of queued vehicles on the link. Figure 5.1 illustrates the relation between the number of queued vehicles and the cumulative travel time delay.

For an intersection with a set of completely conflicting origin-destination pairs it is possible to periodically calculate D_{OD} for each origin-destination pair and then schedule the OD that would lead to the largest cumulative travel time delay if left unserviced.

Let us assume a scheduling decision needs to be made now at time t_0 and the next decision will be made at time t_1. The scheduled OD at time

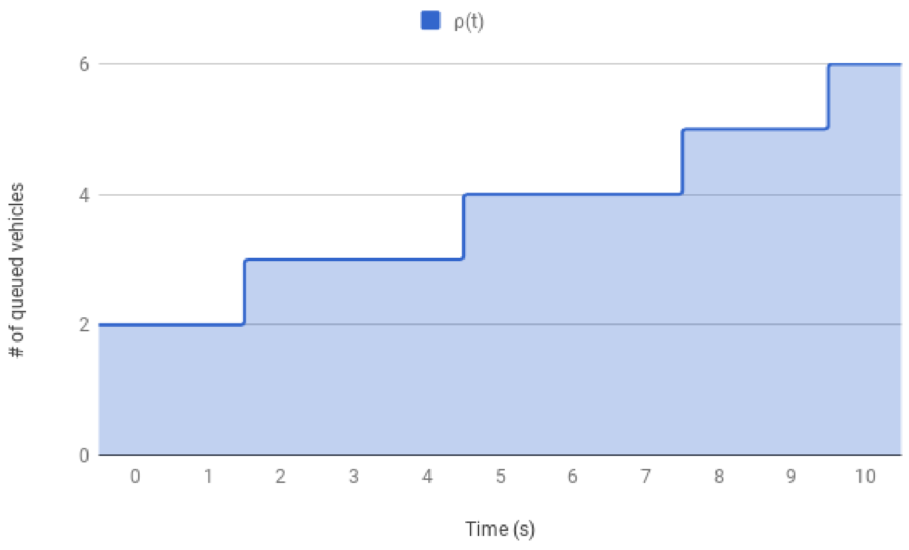


Figure 5.1: Example of how a queue evolves in practice, starting with two queued vehicles at time $t = 0$ and with new vehicles arriving at the back of the queue at $t = \{2, 5, 8, 10\}$. The blue line denotes the number of queued vehicles $\rho(t)$. The shaded area is the cumulative travel time delay $D(0, 10)$ incurred by the queue.

t_0, denoted as $\chi(t_0)$, is then the OD with the highest priority at time t_0, denoted as $\pi(t_0)$. Equation 5.2 describes this mathematically. ODs is the intersection's set of origin-destination pairs.

$$\begin{aligned}\chi(t_0) &= \underset{OD \in ODs}{\arg\max} \left\{ \pi_{OD}(t_0) \right\} \\ &= \underset{OD \in ODs}{\arg\max} \left\{ D_{OD}(t_0, t_1) \right\} \\ &= \underset{OD \in ODs}{\arg\max} \left\{ \int_{t=t_0}^{t_1} \rho_{OD}(t) \right\}\end{aligned} \tag{5.2}$$

However, this assumes that direct switching between phase groups is safe and that $\rho_{OD}(t)$ for each origin-destination pair is known. But future values of $\rho_{OD}(t)$ cannot be known exactly. The challenge therefore is to accurately estimate $\rho_{OD}(t)$ in the future.

$\rho_{OD}(t)$ can be described by Equation 5.3. $\rho_{OD}(t_0)$ is equal to the number of vehicles currently queued for origin-destination pair OD. $\Delta\rho^+_{OD}(t_0, t)$ and $\Delta\rho^-_{OD}(t_0, t)$ are the number of vehicles that will arrive at and depart from origin-destination pair OD in the interval $[t_0, t]$, respectively.

$$\rho_{OD}(t) = \rho_{OD}(t_0) + \Delta\rho_{OD}(t_0, t) = \rho_{OD}(t_0) + \Delta\rho^+_{OD}(t_0, t) - \Delta\rho^-_{OD}(t_0, t) \tag{5.3}$$

Since the goal is to service the OD that would lead to the greatest increase in travel time delay if not serviced, when calculating the cumulative travel time delay the assumption is that no vehicles depart, i.e. $\Delta\rho^-_{OD}(t_0, t) = 0$. Equation 5.3 then simplifies to Equation 5.4.

$$\rho_{OD}(t) = \rho_{OD}(t_0) + \Delta\rho^+_{OD}(t_0, t) \tag{5.4}$$

Given a prediction of the vehicle arrivals it is possible to predict the travel time delay according to Equation 5.1 and then schedule the most demanding OD according to Equation 5.2.

Vehicles can be predicted within a chosen time period or bin size according to the LSTM traffic flow prediction model described in Chapter 4. This bin size will be denoted as t_{bin}. Because the exact arrival times of the predicted vehicles in the interval $[t, t + t_{bin}]$ are unknown[1] an assumption is required. Without further information regarding the arrival patterns of vehicles a uniform distribution of arrivals will be assumed.

In order to easily calculate the travel time delay during an interval $[t, t + t_{bin}]$ a representation of the average queue length during the interval is desired. The average queue length during the interval $[t, t + t_{bin}]$ will be denoted as $\rho_{OD}[T]$, where T is the index of the bin (note the difference in notation, the average queue length during an interval is denoted with square brackets). $\rho_{OD}[T]$ is given by Equation 5.5.

$$\rho_{OD}[T] = \begin{cases} \rho_{OD}(t_0) + \frac{1}{2} \cdot \Delta\rho^+_{OD}(t_0, t_0 + t_{bin}) & T = 0 \\ \rho_{OD}[T-1] + \frac{1}{2} \cdot \Delta\rho^+_{OD}(t_0 + T \cdot t_{bin}, t_0 + (T+1) \cdot t_{bin}) & T > 0 \end{cases} \tag{5.5}$$

Figure 5.2 and Table 5.1 illustrate the transformation from $\rho(t)$ to $\rho[T]$ for time bins of size 10 seconds.

Using the average queue length per interval it is possible to calculate the cumulative travel time delay according to $\rho_{OD}[T]$. This leads to the discretized version of Equation 5.1 represented by Equation 5.6.

$$D_{OD}[T_{start}, T_{end}] = t_{bin} \cdot \sum_{T=T_{start}}^{T_{end}} \rho_{OD}[T] \tag{5.6}$$

Assuming that a new scheduling decision is made every time a new prediction is made[2], i.e. $t_1 = t_0 + t_{bin}$, Equation 5.2 can be rewritten to

[1]The LSTM network predicts the number of vehicles arriving during the time bin, not their exact arrival times within the time bin.

[2]It is possible to use a rolling horizon for the predictions, i.e. predict at a higher frequency than every t_{bin}. For simplicity this option is excluded in this thesis.

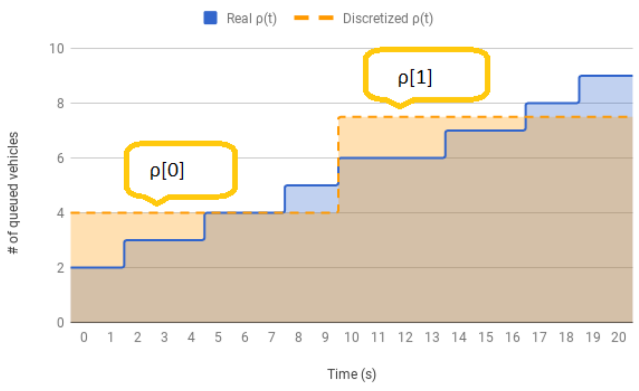


Figure 5.2: The example from Figure 5.1 extended to 20 seconds. The orange dashed line indicates the assumed average queue length according to Equation 5.5.

$\rho[0]$	$\rho[1]$
4	7.5

Table 5.1: Average queue length assuming uniform distribution of arrivals in 10 seconds bins for the example from Figure 5.2.

Equation 5.7.

$$\begin{aligned}\chi(t_0) = \underset{OD \in ODs}{\arg\max}\left\{\pi(t_0)\right\} &= \underset{OD \in ODs}{\arg\max}\left\{D_{OD}(t_0, t_1)\right\}\\ &\approx \underset{OD \in ODs}{\arg\max}\left\{D_{OD}[0,0]\right\}\\ &= \underset{OD \in ODs}{\arg\max}\left\{t_{bin} \cdot \sum_{T=0}^{0} \rho_{OD}[T]\right\}\\ &= \underset{OD \in ODs}{\arg\max}\left\{\sum_{T=0}^{0} \rho_{OD}[T]\right\}\\ &= \underset{OD \in ODs}{\arg\max}\left\{\rho_{OD}[0]\right\}\end{aligned} \tag{5.7}$$

The approximately equal sign indicates that the discretized version is an approximation of the true cumulative travel time delay. t_{bin} can be discarded as it is a constant multiplier in the argmax function.

Example 5.1.1. Table 5.2 shows an example situation at an intersection consisting of three completely conflicting origin-destination pairs 0, 1 and 2.

The example starts out with 2, 1 and 1 vehicles queued for ODs 0, 1 and 2, respectively. Between now (t_0) and the next decision making time (t_1) 4, 3 and 7 vehicles are expected to arrive at ODs 0, 1 and 2, respectively. Filling

OD	$\rho_{OD}(t_0)$	$\Delta\rho^{+}_{OD}(t_0, t_1)$	$\rho_{OD}[0]$	$\pi_{OD}(t_0)$
0	2	4	4	4
1	1	3	2.5	2.5
2	1	7	4.5	4.5

Table 5.2: Example situation at an intersection consisting of 3 completely conflicting origin-destination pairs.

out these details in Equation 5.7 leads to:

$$\begin{aligned}\chi(t_0) &\approx \underset{OD\in\{0,1,2\}}{\arg\max}\left\{\rho_{OD}[0]\right\}\\ &= \arg\max\left\{\rho_0[0], \rho_1[0], \rho_2[0]\right\}\\ &= \arg\max\left\{4, 2.5, 4.5\right\}\\ &= 2\end{aligned}$$

This means that OD 2 will be scheduled next. Even though the immediate queue length of OD 0 is greater, more vehicles are expected to arrive at OD 2, and hence, servicing OD 2 is preferred to minimize the cumulative travel time delay.

Equation 5.7 captures the basic idea that larger queues cause more travel time delay and that cars that are in the queue earlier contribute more to the travel time delay. Therefore Equation 5.7 forms a good basis for a signalized intersection control algorithm. However, Equation 5.7 is still slightly too simple to be used in real scenarios as some of the underlying assumptions do not hold in practice. DIRECTOR's basic approach therefore requires some extensions to live up to reality.

5.2 Extending DIRECTOR with switching penalties

One of the assumptions underlying Equation 5.7 is the possibility to instantly switch between phase groups. In reality, for safety reasons, there is at least an intergreen time t_{ig} between one signal changing from green to amber and the next signal turning green. A typical value for t_{ig} is in the range of 3 to 8 seconds. During this time no vehicles can be serviced by the intersection. Hence, switching phase groups leads to potential inefficiencies at the intersection and should therefore be incorporated in DIRECTOR's decisionmaking algorithm.

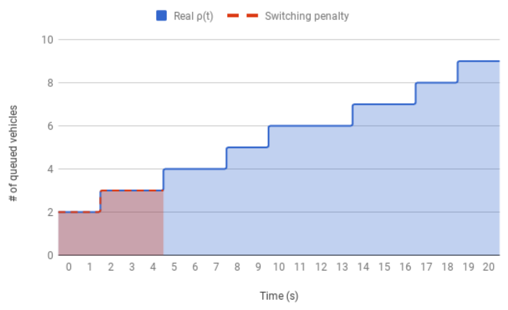


Figure 5.3: The example from Figure 5.1 revisited. The red shaded area indicates the switching penalty for the other ODs if this OD is currently being serviced and the intergreen time t_{ig} is 5 seconds.

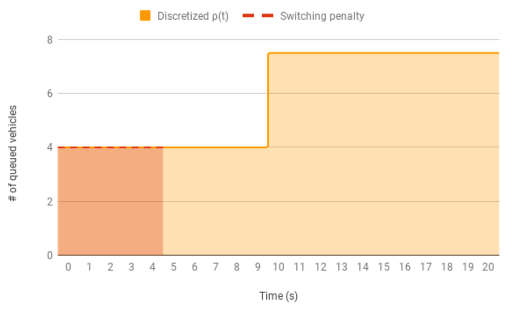


Figure 5.4: The discretized version of Figure 5.3. The red shaded area indicates the switching penalty for the other ODs if this OD is currently being serviced and the intergreen time t_{ig} is 5 seconds.

A switching penalty can be used to take switching inefficiencies into account [31]. The idea is that this penalty reduces the priority of the phase groups currently *not* being serviced and therefore increases the relative importance of the phase group currently being serviced. Equation 5.8 rewrites Equation 5.2 to include such a term. $\epsilon_{OD}(t_0)$ is the switching penalty applicable to origin-destination pair OD at time t_0.

$$\chi(t_0) = \underset{OD \in ODs}{\arg\max} \left\{ \pi(t_0) \right\} = \underset{OD \in ODs}{\arg\max} \left\{ D_{OD}(t_0, t_1) - \epsilon_{OD}(t_0) \right\} \tag{5.8}$$

The question is then how large this switching penalty should be. It should be equal to the cumulative travel time delay caused by the switch. This additional travel time delay is equal to the number of vehicles that could have been serviced during t_{ig} if a switch would not have occurred multiplied by the time they have been waiting. The switching penalty can then be described by Equation 5.9.

$$\epsilon_{OD}(t_0) = \begin{cases} 0 & \text{if } OD = \chi(t_{-1}) \\ \int_{t=t_0}^{t_{ig}} \rho_{\chi(t_{-1})}(t) & \text{if } OD \neq \chi(t_{-1}) \end{cases} \tag{5.9}$$

$\chi(t_{-1})$ is the origin-destination pair currently being serviced (scheduled at time $t_{-1} = t_0 - t_{bin}$). Figure 5.3 illustrates the application of Equation 5.9.

Using the discretized average queue length Equation 5.9 can be rewritten to Equation 5.10. This is under the assumption that $t_{ig} \leq t_{bin}$. Figure 5.4 illustrates the application of Equation 5.10.

$$\epsilon_{OD}(t_0) = \begin{cases} 0 & \text{if } OD = \chi(t_{-1}) \\ t_{ig} \cdot \rho_{\chi(t_{-1})}[0] & \text{if } OD \neq \chi(t_{-1}) \end{cases} \tag{5.10}$$

Substituting this result in Equation 5.8 the scheduling algorithm is given by Equation 5.11.

$$\begin{aligned}
\chi(t_0) &= \underset{OD \in ODs}{\arg\max} \Big\{ D_{OD}(t_0, t_1) - \epsilon_{OD}(t_0) \Big\} \\
&\approx \underset{OD \in ODs}{\arg\max} \Big\{ D_{OD}[0,0] - \epsilon_{OD}(t_0) \Big\} \\
&= \underset{OD \in ODs}{\arg\max} \left\{ \begin{cases} \left(t_{bin} \cdot \sum_{T=0}^{0} \rho_{OD}[T] \right) - 0 & \text{if } OD = \chi(t_{-1}) \\ \left(t_{bin} \cdot \sum_{T=0}^{0} \rho_{OD}[T] \right) - t_{ig} \cdot \rho_{\chi_{t-1}}[0] & \text{if } OD \neq \chi(t_{-1}) \end{cases} \right\} \\
&= \underset{OD \in ODs}{\arg\max} \left\{ \begin{cases} \sum_{T=0}^{0} \rho_{OD}[T] & \text{if } OD = \chi(t_{-1}) \\ \left(\sum_{T=0}^{0} \rho_{OD}[T] \right) - \frac{t_{ig}}{t_{bin}} \cdot \rho_{\chi_{t-1}}[0] & \text{if } OD \neq \chi(t_{-1}) \end{cases} \right\} \\
&= \underset{OD \in ODs}{\arg\max} \left\{ \begin{cases} \rho_{OD}[0] & \text{if } OD = \chi(t_{-1}) \\ \rho_{OD}[0] - \frac{t_{ig}}{t_{bin}} \cdot \rho_{\chi_{t-1}}[0] & \text{if } OD \neq \chi(t_{-1}) \end{cases} \right\}
\end{aligned} \tag{5.11}$$

The last term of Equation 5.11 intuitively makes sense as the fraction $\frac{t_{ig}}{t_{bin}}$ is the fraction of the time period t_{bin} that no vehicles can be serviced. For simplicity this term will be defined as the switching penalty $\sigma_{OD}(t_0)$. Equation 5.12 describes the explicit definition.

$$\sigma_{OD}(t_0) = \frac{\epsilon_{OD}(t_0)}{t_{bin}} = \begin{cases} 0 & \text{if } OD = \chi(t_{-1}) \\ \frac{t_{ig}}{t_{bin}} \cdot \rho_{\chi(t_{-1})}[0] & \text{if } OD \neq \chi(t_{-1}) \end{cases} \tag{5.12}$$

This simplifies Equation 5.11 to Equation 5.13.

$$\chi(t_0) \approx \underset{OD \in ODs}{\arg\max} \Big\{ \rho_{OD}[0] - \sigma_{OD}(t_0) \Big\} \tag{5.13}$$

To illustrate the influence of the switching penalty Example 5.1.1 is revisited in Example 5.2.1.

Example 5.2.1. In this example the assumptions are that OD 1 is currently being serviced, i.e. $\chi_{t_{-1}} = 1$, and that $t_{ig} = 5$ seconds. t_{bin} is still 10 seconds. Table 5.3 then shows the results according to Equations 5.12 and 5.13.

OD	$\rho_{OD}[t_0]$	$\rho^{+}_{OD}(t_0, t_1)$	$\rho_{OD}[0]$	$\sigma_{OD}(t_0)$	$\pi_{OD}(t_0)$
0	2	4	4	1.25	2.75
1	1	3	2.5	0	2.5
2	1	7	4.5	1.25	3.25

Table 5.3: Example situation at an intersection consisting of 3 completely conflicting origin-destination pairs. $\chi_{t_{-1}} = 1$, $t_{ig} = 5$ seconds and $t_{bin} = 10$ seconds.

Substituting the values from Table 5.3 in Equation 5.13 gives:

$$\begin{aligned}\chi(t_0) &\approx \underset{OD\in\{0,1,2\}}{\arg\max}\left\{\rho_{OD}[0]-\sigma_{OD}(t_0)\right\}\\ &= \arg\max\left\{(4-1.25),(2.5-0),(4.5-1.25)\right\}\\ &= \arg\max\left\{2.75, 2.5, 3.25\right\}\\ &= 2\end{aligned}$$

Hence, in this example the scheduling decision has not changed, OD 2 will still be scheduled. However, Table 5.3 shows that even though the scheduling decision has not changed, the relative priority of OD 1 did increase compared to the example without a switching penalty.

5.3 Extending DIRECTOR to create dynamic green waves

As it takes time to switch between phase groups it makes sense to prepare in advance for large platoons of vehicles that will have to stop if there is a queue that still needs to be emptied upon the platoon's arrival. Therefore it makes sense to take vehicle arrivals further into the future into account as well. This section describes the extension of DIRECTOR to include predicted arrivals from $t_2 = t_1 + t_{bin}$, where $t_1 = t_0 + t_{bin}$ as before.

Including future arrivals can be done by taking the travel time delay up to $t = t_2$ into account, which can be calculated by filling out the correct values in Equations 5.5 and 5.6. This leads to Equation 5.14.

$$\begin{aligned}D_{OD}(t_0,t_2) &\approx D_{OD}[0,1]\\ &= t_{bin}\cdot\sum_{T=0}^{1}\rho_{OD}[T]\\ &= t_{bin}\cdot(\rho_{OD}[0]+\rho_{OD}[1])\end{aligned} \tag{5.14}$$

The scheduling decision is then governed by Equation 5.15.

$$\begin{aligned}\chi(t_0) &= \underset{OD\in ODs}{\arg\max}\left\{D_{OD}(t_0,t_2)-\epsilon_{OD}(t_0)\right\}\\ &\approx \underset{OD\in ODs}{\arg\max}\left\{D_{OD}[0,1]-\epsilon_{OD}(t_0)\right\}\\ &= \underset{OD\in ODs}{\arg\max}\left\{\sum_{T=0}^{1}\rho_{OD}[T]-\sigma_{OD}(t_0)\right\}\\ &= \underset{OD\in ODs}{\arg\max}\left\{\rho_{OD}[0]+\rho_{OD}[1]-\sigma_{OD}(t_0)\right\}\end{aligned} \tag{5.15}$$

To illustrate the influence of taking arrivals further in the future into account Example 5.2.1 will be revisited in Example 5.3.1.

Example 5.3.1. In this example the same assumptions hold as in Example 5.2.1. Table 5.4 shows the results according to Equations 5.5 and 5.15.

OD	$\rho_{OD}(t_0)$	$\Delta\rho^{+}_{OD}(t_0, t_1)$	$\rho_{OD}[0]$	$\Delta\rho^{+}_{OD}(t_1, t_2)$	$\rho_{OD}[1]$	$\sigma_{OD}(t_0)$	$\pi_{OD}(t_0)$
0	2	4	4	2	5	1.25	7.75
1	1	3	2.5	10	7.5	0	10
2	1	7	4.5	2	5.5	1.25	8.75

Table 5.4: Example situation at an intersection consisting of 3 completely conflicting origin-destination pairs. OD 1 has just been serviced, $t_{ig} = 5$ seconds and $t_{bin} = 10$ seconds. The scheduling algorithm also takes travel time delay due to arrivals from the interval $[t_1, t_2]$ into account.

The scheduling decision according to Equation 5.15 then becomes:

$$\begin{aligned}
\chi(t_0) &= \underset{OD\in\{0,1,2\}}{\arg\max} \Big\{ D_{OD}(t_0, t_2) - \epsilon_{OD}(t_0) \Big\} \\
&\approx \underset{OD\in\{0,1,2\}}{\arg\max} \Big\{ \rho_{OD}[0] + \rho_{OD}[1] - \sigma_{OD}(t_0) \Big\} \\
&= \arg\max \Big\{ (\rho_0[0] + \rho_0[1] - \sigma_0(t_0)), (\rho_1[0] + \rho_1[1] - \sigma_1(t_0)), (\rho_2[0] + \rho_2[1] - \sigma_2(t_0)) \Big\} \\
&= \arg\max \Big\{ (4 + 5 - 1.25)_0, (2.5 + 7.5 - 0)_1, (4.5 + 5.5 - 1.25)_2 \Big\} \\
&= \arg\max \Big\{ 7.75, 10, 8.75 \Big\} \\
&= 1
\end{aligned}$$

Note that the outcome of the scheduling decision has changed compared to the previous examples. DIRECTOR recognizes that a big platoon of 10 vehicles will arrive at OD 1 and hence decides not to switch phase groups. DIRECTOR decides to stop the 7 incoming vehicles at OD 2 in order to service all 14 vehicles in the next 20 seconds at OD 1. Of course this decision will be reevaluated at time t_1 with the knowledge available to DIRECTOR at that point in time.

5.4 Generalizing DIRECTOR to multi-OD phase groups

In reality an intersection often has several non-conflicting ODs that can be serviced simultaneously as a phase group PG. This section explains how DIRECTOR can be generalized to deal with phase groups. The principle is rather simple. The goal is to minimize the cumulative travel time delay at the intersection. Therefore the phase group that would lead to the largest expected cumulative travel time delay if it is not serviced should be scheduled next. This can be done by summing the expected cumulative travel time delays of the ODs in each phase group and subtracting the phase group's switching penalty.

Since a schedule $\chi(t)$ now consists of a phase group, which is set of ODs instead of a single OD, the definition of the switching penalty should be adapted to this change. Equations 5.9, 5.10 and 5.12 should be rewritten to Equations 5.16, 5.17 and 5.18.

$$\epsilon_{PG}(t_0) = \sum_{OD \in \chi(t_{-1}) \wedge OD \notin PG} \int_{t=t_0}^{t_{ig}} \rho_{OD}(t) \tag{5.16}$$

$$\epsilon_{PG}(t_0) = t_{ig} \cdot \sum_{OD \in \chi(t_{-1}) \wedge OD \notin PG} \rho_{OD}[0] \tag{5.17}$$

$$\sigma_{PG}(t_0) = \frac{\epsilon_{PG}}{t_{bin}} = \frac{t_{ig}}{t_{bin}} \cdot \sum_{OD \in \chi(t_{-1}) \wedge OD \notin PG} \rho_{OD}[0] \tag{5.18}$$

The scheduling decision is then given by Equation 5.19. PGs is the intersection's set of phase groups.

$$\begin{aligned}
\chi(t_0) &= \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} D_{OD}(t_0, t_2) \right) - \epsilon_{PG}(t_0)) \right\} \\
&\approx \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} D_{OD}[0, 1] \right) - \epsilon_{PG}(t_0)) \right\} \\
&= \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} \sum_{T=0}^{1} \rho_{OD}[T] \right) - \sigma_{PG}(t_0) \right\} \\
&= \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} \rho_{OD}[0] + \rho_{OD}[1] \right) - \sigma_{PG}(t_0) \right\}
\end{aligned} \tag{5.19}$$

5.5 Guaranteeing stability

Stability in intersection control is defined as having bounded queues at all times [40] as long as the traffic demand does not exceed the intersection capacity. An intersection's capacity is equal to the maximum amount of traffic that a cyclic, fixed-time controller can handle, since such an algorithm does not suffer from inefficiencies due to extra phase group switches. Hence, if a control algorithm performs at least as good as a stable, cyclic, fixed-time controller at all times then the control algorithm can be considered stable.

The short-sightedness of a local optimization strategy, like the one described in Section 5.4, is not guaranteed to be stable as it might lead to inefficient use of the available capacity [31]. Therefore, a supervisory stabilization mechanism is required.

The idea is to have a local supervisory mechanism that selects phase groups that require service and adds them to a first in first out (FIFO) queue Ω. Whenever Ω is not empty, the next entry from Ω will be serviced. Mathematically the scheduling decision is given by Equation 5.20.

$$\chi(t_0) = \begin{cases} \text{head}\{\Omega\} & \text{if } \Omega \neq \emptyset \\ \arg\max_{PG \in PGs} \left\{ \left(\sum_{OD \in PG} \rho_{OD}[0] + \rho_{OD}[1] \right) - \sigma_{PG}(t_0) \right\} & \text{otherwise} \end{cases} \tag{5.20}$$

5.5.1 Selection of phase groups

Unstable control of the intersection through Equation 5.19 can occur when a queue spills back far enough to permanently cover the arrival detector. DIRECTOR is then unable to detect new arrivals and will underestimate the queue length, potentially leading to inefficient decisions and unbounded queues. Hence, whenever an origin-destination pair has a queue covering its arrival detectors a phase group containing the OD should be added to Ω.

The set of phase groups from which the entries for Ω are picked is an ordered set, in the order of service in the stable, cyclic, fixed-time schedule. If Ω is empty the phase group with the highest priority is selected from the set of phase groups that services the required origin-destination pair. When Ω is not empty the phase group that comes next in the ordered set after the tail of Ω is selected. This selection process guarantees that in the worst-case the phase groups will be scheduled in the same order as the stable, cyclic, fixed-time controller.

5.5.2 Duration of green time

A phase group scheduled by the supervisory mechanism remains serviced until the detector is no longer occupied or until the maximum allowed green time is exceeded. When that happens the phase group is removed from the queue. The duration of each phase group's maximum green time g_{PG}^{max} is key for the stability of the algorithm.

From now on the assumption will be that a stable, cyclic, fixed-time controller exists for the intersection. Then, if the maximum green time of phase group PG g_{PG}^{max} is equal to the green time allocated to phase group PG by the fixed-time controller, the control algorithm can be considered stable.

To prove this, consider the worst-case, which is that all arrival detectors are covered by queues. In that case Ω contains at most all phase groups of the intersection. When Ω contains all phase groups and the maximum green times of each phase group is equal to the green time allocated by a fixed-time controller, the schedule produced based on Ω will at worst be equal to the fixed-time controller schedule. Since that schedule is considered stable the predictive controller schedule can also be considered stable.

5.6 Dealing with real-world requirements

DIRECTOR is now proven to be stable and takes intergreen times into account. However, for practical purposes other real-world requirements need to be addressed as well. This section covers the most important remaining requirements:

- *Minimum green time*: A green signal should remain green for at least a minimum green time duration to provide road users enough time to safely leave the intersection.
- *Duration of amber*: The duration of amber should allow road users to safely come to a stop before the signal turns red given the applicable speed limit.
- *Minimum clearance time*: A signal should be red for at least the minimum clearance time before a conflicting signal may turn green.
- *Maximum time without service*: The maximum time that an origin-destination pair with a vehicle waiting is not serviced should not exceed a particular period in order to avoid vehicles violating the red light.
- *Priority*: Some vehicles should get priority over others, e.g. public transport or emergency service vehicles like an ambulance.

The remainder of this section addresses how these requirements can be handled by DIRECTOR.

5.6.1 Minimum green time, duration of amber & minimum clearance time

Adhering to these timing parameters is incorporated in DIRECTOR's finite state machine that controls the switching of signals. Note that the amber time and minimum clearance time together form the intergreen time.

5.6.2 Maximum time without service

In practice a *maximum time without service* is defined to avoid vehicles violating a red light. This is the maximum time period between two consecutive services while a vehicle was waiting. Hence, in order to guarantee safe operation of the intersection controller this requirement should be incorporated.

A simple way of addressing this issue is by adding another supervisory mechanism that looks after the maximum time without service. Whenever an origin-destination pair has a vehicle waiting longer than the maximum time without service a phase group containing the OD will be added to a FIFO queue Ψ. Phase groups from Ψ should be preferred over the other scheduling options to ensure safety, which would lead to a scheduling decision given by Equation 5.21.

$$\chi(t_0) = \begin{cases} \text{head}\{\Psi\} & \text{if } \Psi \neq \emptyset \\ \text{head}\{\Omega\} & \text{if } \Omega \neq \emptyset \\ \arg\max_{PG \in PGs} \left\{ \left(\sum_{OD \in PG} \rho_{OD}[0] + \rho_{OD}[1] \right) - \sigma_{PG}(t_0) \right\} & \text{otherwise} \end{cases} \tag{5.21}$$

Note that Equation 5.21 is only guaranteed to be stable when the *maximum time without service* t^{max} is at least as large as the cycle time t^{cycle} of the stable fixed-time controller on which the schedule is based, i.e. $t^{max} \geq t^{cycle}$. If this is not the case, then the worst-case fixed-time schedule might be interrupted, which would lead to additional inefficiencies that could cause instability.

5.6.3 Priority

Depending on the context and goals of the road owner some vehicles have priority over others. For example public transport or emergency services, which should not be delayed. Trucks, which cause more greenhouse gas emissions than regular cars when stopped, are another example. Or maybe the traffic going into a city center should be discouraged by getting a lower priority.

Handling priorities is not yet implemented in this thesis project. It is considered as a part of future work. Nevertheless three options for the implementation of handling priorities should be mentioned:

1. Implicit: Assign weights to different vehicle types.
2. Explicit: Add a phase group to the FIFO queues Ω or Ψ.
3. Explicit: Create an extra supervisory mechanism to handle priority requests. This mechanism should be preferred over Ω and Ψ.

Chapter 6

Enabling Advanced Driver Assistance Systems

Chapter 5 described the design of a stable, predictive controller that responds ad hoc with a scheduling decision being made every t_{bin} seconds. This chapter builds on the design from Chapter 5 and describes how this DIRECTOR implementation can and must be adjusted to enable advanced driver assistance systems (ADAS) like time to green/red (T2G/R) and green light optimal speed advice (GLOSA).

6.1 Requirements for Advanced Driver Assistance Systems

To safely utilize T2G/R and GLOSA there are two main requirements:

- 100% accurate information on signal changes
- Accurate signal phase and timing information over a sufficiently large prediction horizon (5-40 seconds)

As described before, in Chapters 1 and 2, modern intersection controllers and prediction methods from literature have failed to meet these requirements. The challenge is thus to develop DIRECTOR such that it performs well and can meet the requirements mentioned above.

6.2 Meeting the requirements with DIRECTOR

The only way to guarantee that the predictions will be 100% accurate is to fix the schedule ahead of time. Note that the earlier the schedule is fixed the larger the prediction horizon will be. But also the greater the uncertainty in the predicted inputs and hence the quality of the produced schedule.

This section describes how DIRECTOR's implementation from Chapter 5 needs to be adjusted to fix the schedule ahead of time. The starting assumption will be that the DIRECTOR's schedule will be fixed t_{bin}, i.e. one decision cycle, ahead of time. For simplicity the *stabilization* and *maximum time without service* mechanisms are excluded in the beginning.

Before a new scheduling equation can be given some assumptions that no longer hold need to be discussed.

6.2.1 Account for leaving vehicles

The assumption from Chapter 5 that the number of vehicles leaving from origin-destination pair OD in the interval $[t_0, t]$ ($\Delta\rho^-_{OD}(t_0, t)$) were ignored in order to evaluate the impact of not servicing OD is no longer valid. The interval for the assumption should be shifted by t_{bin} as $t_1 = t_0 + t_{bin}$ is the moment that the picked schedule will be applied. Under that new assumption Equation 5.3 should be rewritten to Equation 6.1 instead of Equation 5.4.

$$\rho_{OD}(t) = \rho_{OD}(t_0) + \Delta\rho^+_{OD}(t_0, t) - \Delta\rho^-_{OD}(t_0, t_1) \tag{6.1}$$

Where the number of vehicles leaving in the interval $[t_0, t_1]$ is given by Equation 6.2.

$$\Delta\rho^-_{OD}(t_0, t_1) = \begin{cases} 0 & \text{if } OD \notin \chi(t_0) \\ \min\left\{\rho(t_0) + \Delta\rho^+_{OD}(t_0, t_1), \Delta\rho^{-,max}_{OD}(t_1 - t_0 - t_{ig})\right\} & \text{if } OD \notin \chi(t_{-1}) \\ \min\left\{\rho(t_0) + \Delta\rho^+_{OD}(t_0, t_1), \Delta\rho^{-,max}_{OD}(t_1 - t_0)\right\} & \text{otherwise} \end{cases} \tag{6.2}$$

Where $\Delta\rho^{-,max}_{OD}(\Delta t)$ is the maximum number of vehicles that origin-destination pair OD can service during an interval of Δt seconds. The minimum operator is necessary in case there are more vehicles queued than the OD can service during the given interval.

In words Equation 6.2 describes the following cases:

- The first case of Equation 6.2 means that OD will not be serviced in the interval $[t_0, t_1]$. Hence, no vehicles will leave.
- The second case means that OD will be serviced in the interval $[t_0, t_1]$ but requires intergreen time first. Therefore vehicles can leave the OD but only during the interval$[t_0 + t_{ig}, t_1]$.
- The last case means that the OD will be serviced in the interval $[t_0, t_1]$ and does not require intergreen time because the OD was already being serviced before.

Following the reasoning above means that Equation 5.5, which describes the discretization process, should be adjusted to Equation 6.3.

$$\rho_{OD}[T] = \begin{cases} \rho_{OD}(t_0) + \frac{1}{2} \cdot \left(\Delta\rho^+_{OD}(t_0, t_0 + t_{bin}) - \Delta\rho^-_{OD}(t_0, t_0 + t_{bin})\right) & \text{if } T = 0 \\ \rho_{OD}[T-1] + \frac{1}{2} \cdot \Delta\rho^+_{OD}(t_0 + T \cdot t_{bin}, t_0 + (T+1) \cdot t_{bin}) & \text{if } T > 0 \end{cases} \tag{6.3}$$

6.2.2 Shift the switching penalty

Equations 5.16, 5.17 and 5.18 described the switching penalty for a switch occurring at time $t = t_0$. Now that the switch occurs at time $t = t_1$ these equations should be rewritten to Equations 6.4, 6.5 and 6.6.

$$\epsilon_{PG}(t_0) = \sum_{OD \in \chi(t_0) \wedge OD \notin PG} \int_{t=t_1}^{t_1+t_{ig}} \rho_{OD}(t) \tag{6.4}$$

$$\epsilon_{PG}(t_0) = t_{ig} \cdot \sum_{OD \in \chi(t_0) \wedge OD \notin PG} \rho_{OD}[1] \tag{6.5}$$

$$\sigma_{PG}(t_0) = \frac{\epsilon_{PG}}{t_{bin}} = \frac{t_{ig}}{t_{bin}} \cdot \sum_{OD \in \chi(t_0) \wedge OD \notin PG} \rho_{OD}[1] \tag{6.6}$$

6.2.3 Find schedule ahead of time

With a proper description of the queue length and the switching penalty it is now possible to find the schedule t_{bin} seconds ahead of time by shifting Equation 5.19 by t_{bin}. This leads to Equation 6.7.

$$\begin{aligned} \chi(t_0 + t_{bin}) &= \chi(t_1) \\ &= \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} W_{OD}(t_1, t_3) \right) - \epsilon_{PG}(t_0) \right\} \\ &\approx \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} W_{OD}[1, 2] \right) - \epsilon_{PG}(t_0) \right\} \\ &= \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} \sum_{T=1}^{2} \rho_{OD}[T] \right) - \sigma_{PG}(t_0) \right\} \\ &= \underset{PG \in PGs}{\arg\max} \left\{ \left(\sum_{OD \in PG} \rho_{OD}[1] + \rho_{OD}[2] \right) - \sigma_{PG}(t_0) \right\} \end{aligned} \tag{6.7}$$

6.2.4 Guaranteed stability

Of course, similar to the ad-hoc implementation from Chapter 5, a supervising mechanism is required to guarantee stability. Since the only change made was a shift in response time of t_{bin} seconds the same mechanism as in Chapter 5 can be used. The only difference will be that the controller responds t_{bin} seconds later. Nevertheless, the worst-case performance is still equal to that of a stable fixed-time controller. Hence, a controller scheduling according to Equation 6.8 will guarantee stability.

$$\chi(t_1) = \begin{cases} \text{head}\{\Omega\} & \text{if } \Omega \neq \emptyset \\ \arg\max_{PG \in PGs} \left\{ \left(\sum_{OD \in PG} \rho_{OD}[1] + \rho_{OD}[2] \right) - \sigma_{PG}(t_0) \right\} & \text{otherwise} \end{cases} \tag{6.8}$$

6.2.5 Real-world requirements

Fixing the schedule ahead of time also affects the responsiveness of the supervisory mechanism dealing with the *maximum time without service*. Since the schedule is decided t_{bin} seconds ahead, DIRECTOR knows t_{bin} seconds ahead of time if an origin-destination pair will not be serviced within the maximum time interval and can add it to the FIFO queue Ψ if required. The scheduling algorithm is then similar to the algorithm from Chapter 5:

$$\chi(t_1) = \begin{cases} \text{head}\{\Psi\} & \text{if } \Psi \neq \emptyset \\ \text{head}\{\Omega\} & \text{if } \Omega \neq \emptyset \\ \arg\max_{PG \in PGs} \left\{ \left(\sum_{OD \in PG} \rho_{OD}[1] + \rho_{OD}[2] \right) - \sigma_{PG} \right\} & \text{otherwise} \end{cases} \tag{6.9}$$

6.2.6 Limitations

DIRECTOR's process described to fix the schedule t_{bin} seconds ahead of time can theoretically be repeated recursively to fix the schedule any multiple of t_{bin} seconds ahead of time. However, some practical limitations influencing the *maximum schedule fixing horizon* exist.

- The prediction horizon of the vehicle arrival predictions is limited. At some point no (accurate) predictions are available anymore.
- The delay in stabilizing measures might become too large. When the time between an origin-destination pair OD being added to Ω and actually servicing that OD becomes too large, queues might spill back onto previous intersections leading to a gridlock and hence an instable network.
- Fixing the schedule ahead of time further than the *maximum time without service* would lead to a violation of the *maximum time without service* and hence this is a hard limit for the schedule fixing horizon.

6.3 Advanced Driver Assistance Systems implementation

DIRECTOR is now able to deliver 100% accurate predictions for a prediction horizon of at least t_{bin} seconds. This section describes how this information can be used to utilize T2G/R and GLOSA. First the T2G/R information is derived. This information is then used to calculate the right GLOSA.

6.3.1 T2G/R

Every time a new schedule is fixed the T2G/R information must be updated. The signal state of every origin-destination pair OD can be characterized by three variables:

- the current signal color $signal\_color_{OD}$
- the T2G countdown $T2G_{OD}$
- the T2R countdown $T2R_{OD}$

$signal\_color_{OD}$ is either *RED*, *AMBER* or *GREEN*. The T2G/R countdowns are either a strictly positive integer number (i.e. > 0) or unknown. Unknown in this case means that DIRECTOR has not planned any change for the OD in the coming schedule decision.

Throughout this reasoning the assumption will remain that the schedule is fixed t_{bin} seconds ahead of time and hence the T2G/R countdowns are limited in horizon by t_{bin}.

When an OD is red at the start of the current schedule ($\chi(t_0)$) but will turn green in the next schedule ($\chi(t_1)$) the T2G countdown at time t_0 is $T2G_{OD} = t_{bin} + t_{ig}$ because the schedule becomes active at time $t_0 + t_{bin}$ and then it takes an intergreen time t_{ig} until the signal actually turns green.

If the duration of a signal being amber is defined as t_{amber}. Then when an OD is green at the start of the current schedule but will turn red in the next schedule the T2R countdown at time t_0 is $T2R_{OD} = t_{bin} + t_{amber}$ because the schedule becomes active at time $t_0 + t_{bin}$ turning the signal from green to amber. And then it takes an amber period t_{amber} until the signal actually turns red.

Using this logic the countdown can be set and then counted down until it reaches 0 and the signal change occurs.

6.3.2 GLOSA

The T2G/R countdowns from above can be used to calculate the green light optimal speed advice for the road users. Listing B.1 shows the pseudo code of the relatively basic GLOSA implementation used in this thesis.

The idea is that the system calculates the speed at which the vehicle will catch the green light when the information from the T2G/R countdowns is available. The optimal speed advice is then compared to the minimum and maximum speed limits to guarantee that the speed advice is within the safety bounds. The minimum speed is determined by the road owner to avoid dangerous situations. When no countdown information is available the vehicle will be advised to follow the minimum or maximum speed depending on whether the signal is red or green, respectively. In Listing B.1 `distance` is the distance between the vehicle and the stopline. Note that this implementation does not take the queue length into account.

Furthermore, it might be desirable to advise the vehicle to arrive several seconds before the signal change to green. This can be done using a correction in the T2G used for the calculations in Appendix B. An example of such an implementation is available on Github[1].

[1] `https://github.com/janceesvansenden/thesis`. Access granted on request.

Chapter 7

Experimental setup

This chapter explains the experimental setup used to evaluate DIRECTOR's different components and the final complete controller design. It explains the case study, data preparation, tools and simulation details.

7.1 Case study

The case study used for this thesis is the intersection N205-N201 near Hoofddorp (The Netherlands), which has three directions and connects the N205 and N201 provincial roads. It has three intersections preceding it, each at approximately thirty seconds driving distance. The availability of all required sensors and the lack of influential sources and sinks make this intersection a good candidate for DIRECTOR. For more details on the case study location see Helmy's thesis [21].

Figure 7.1 is a schematic drawing of the intersection showing all lanes, possible vehicle movements and the detector configuration.

The intersection setup is quite complex with each origin link mapping to two destination links and a highly adaptive control algorithm currently controlling the signals. This complexity makes it hard to accurately predict the vehicle flows and to improve upon the currently implemented control algorithm.

7.2 Data

Historic data from the intersection is used for machine learning and system evaluation in order to best reflect reality.

7.2.1 Data origin

The Dutch traffic light controllers (TLCs) log their data in the V-Log data format [14]. Among the data stored are the detectors' states, signals' states

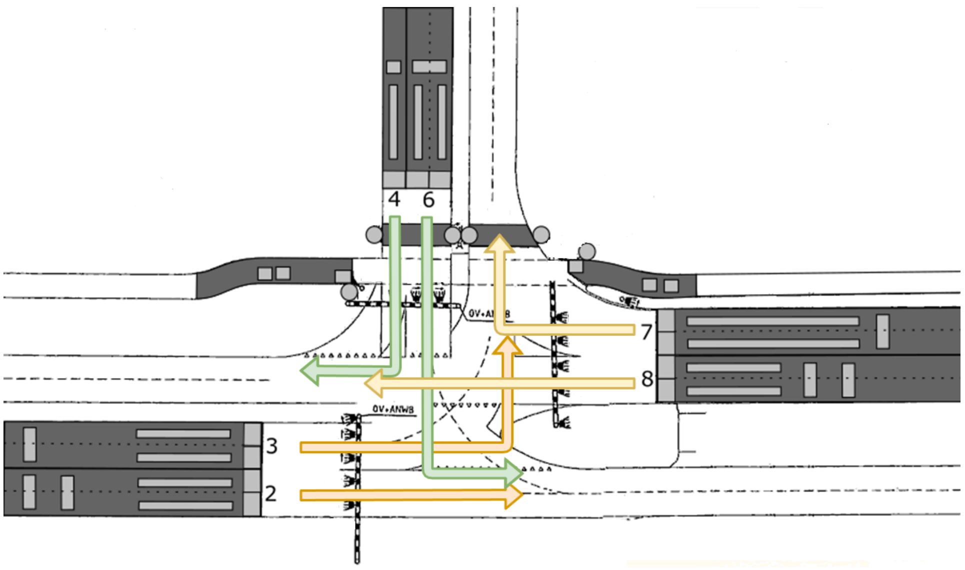


Figure 7.1: Schematic overview of the case study intersection. The arrows indicate the movements of traffic. The intersection has three directions (indicated by the color of the arrows) with two origin-destination pairs each (indicated by the arrows). The grey rectangles represent the loop detectors. Each OD pair carries an ID according to Dutch traffic conventions, which is why 1 and 5 are not present.

and the internal system state of the TLC. It should be noted that the V-Log format only records changes of the system's values and status.

The data used for the experiments in this study dates from January 2017 to May 2017. This data is compiled from the V-Log format to numerical data that can be used for machine learning and simulation purposes.

7.2.2 Data preparation

Before machine learning or simulations can be attempted the numerical data needs to be prepared for these purposes. This process is to a large extent similar to Helmy's data preparation process [21]. As noted in Chapter 4, five input variables are required by the model:

- **Time:** The timestamp and day of the week are used as input variables. The timestamp is included as the bin index from that day starting with 0 at noon (12 PM). The day of the week is supplied as an integer number from 1 to 7, where 1 means Sunday and 7 means Saturday.
- **Departure flows:** In order to predict vehicle flows it is required to aggregate the loop detection values to flows of vehicles per time unit. This is done by collecting the timestamps of vehicles leaving

or arriving at detectors and aggregating this data over the desired time bin size. Departure flows are determined by checking when the upstream stopline detectors change status from 'Car present' to 'No car present'.

- **Presence of queues:** A queue is marked as present in a time bin when at the end of the time bin the queue detector is occupied and has been occupied for at least 3 seconds. 'Queue present' is represented by a 1 and 'No queue present' is represented by a 0.

- **Arrival flows:** Similar to the departure flows. Arrival flows are measured by checking when the downstream arrival detectors change status from 'No car present' to 'Car present'.

- **Signal states:** The phase is included in the dataset by checking what the status of the signals is at the end of the bin, i.e. red, green, amber. Each color is represented by an integer value 0, 1 or 2, respectively.

Because of erroneous detector data only the first three weeks of the dataset can be used for machine learning and simulations (see [21] for details).

Before using the data for machine learning the dataset is centered around zero by subtracting the mean of each variable and then scaled between -1 and 1. According to common practice this dataset (containing the first three weeks of data) is split into 60% training data, 20% validation data and 20% test data.

7.3 Tools

Several off-the-shelf packages and programs were used to develop the experimental setup. The components are linked together with custom code. All the code developed for this thesis is available on Github[1]. Access to the repository and data will be granted on request provided that, if the requested information is proprietary, all owners of the requested information agree to the request.

7.3.1 Machine learning

The Keras interface [11] for Tensorflow [1] was used to implement the machine learning experiments. Several methods from the Scipy package [25] were used for data preparation and performance evaluation.

[1]`https://github.com/janceesvansenden/thesis`

7.3.2 Traffic simulations

Vissim [20] is the main traffic simulation tool used by Dutch road owners. The road owners already have Vissim models of most intersections, and hence, it made sense to use Vissim[2] as the simulation tool for the traffic flow experiments.

Vissim is intended to be used via its graphical interface, but the software also provides a COM (component object model) interface that enables control using custom scripts written in, for example, python. The COM interface has been the main method to control the vehicle inputs and signals, and also to read the detector states during simulations.

For the N205-N201 case study a Vissim model was already available and supplied by the province of Noord-Holland. The Vissim model supplied by the province was adjusted to be able to replay historical data recorded on the street in order to evaluate the control algorithms as accurately as possible.

7.4 Simulation details

Figure 7.2 shows a screenshot of the Vissim simulation model and explains the most important features.

7.4.1 Vehicle inputs

Throughout the simulation vehicles approaching the intersection are added based on the recorded historical data. Every ten seconds the historically recorded flows are reproduced arriving at a random time within that ten-second bin. The randomization is done by Vissim.

To properly evaluate the influence of GLOSA a second model was created where the vehicle inputs are located ten seconds (average driving time) further upstream. This increases the time that a vehicle is able to respond to GLOSA. The downside is that the behavior of the vehicles in the simulation leads to a greater spread of arrival times, which means that the predictions are less accurate. This inaccuracy in the second model is the reason why the first model (as shown in Figure 7.2) was used to evaluate the scenarios without GLOSA.

7.4.2 DIRECTOR

DIRECTOR gets the prediction information from pre-calculated traffic flow predictions based on the historical data and live detector information from the vehicle detectors in the simulation, which are shown as blue rectangles in Figure 7.2. Every 100 milliseconds DIRECTOR reads the states of the

[2]Version 10.06

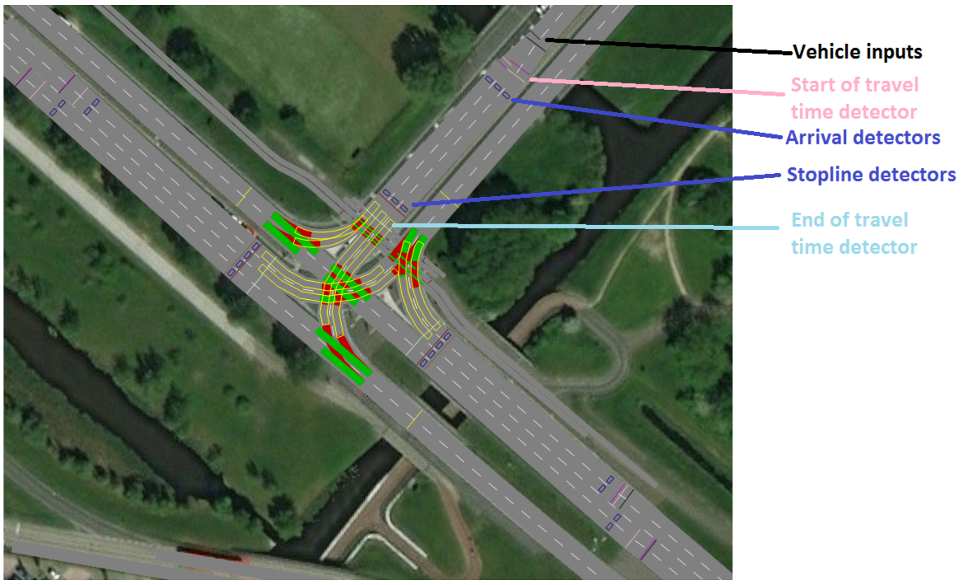


Figure 7.2: A screenshot of the Vissim simulation model. The black lines are the vehicle inputs, i.e. the location where vehicles are created in the simulation. The vehicle inputs are located right in front of the arrival detectors. The detectors are the blue rectangles in the screenshot. The cumulative travel time delay is measured via 'vehicle travel times'-detectors. A pink line indicates the start of a 'vehicle travel times'-detector and a light blue line indicates the end.

detectors and uses that information to count arriving and departing vehicles and estimate the queue length for each origin-destination pair.

DIRECTOR can directly manipulate the traffic lights in the simulation and update a signal's state every second. This update frequency is a limitation of the simulation software. In reality signal states can be updated every 100 milliseconds.

7.4.3 Measurements

Travel times per vehicle are measured using 'vehicle travel times'-detectors. A 'vehicle travel times'-detector measures the time it takes a vehicle to move from one location to the next. It also calculates the delay that each passing vehicle experiences compared to when the vehicle would have been able to continue under free flow conditions, i.e. without a red signal or queue in its way. The 'vehicle travel times'-detector also measures the number of times a vehicle has to stop[3]. The detectors measure the travel time from right after

[3]Most of the vehicles stop only 0 or 1 times. However, there are situations when a vehicle may need to stop 2 or more times because of a long queue with short green time intervals.

the vehicle input to right behind the stopline as can be seen in Figure 7.2. A pink line indicates the start of a travel time detector and a light blue line indicates the end.

Chapter 8

Results

8.1 LSTM network accuracy

To evaluate the performance of the LSTM short-term traffic flow prediction model it is compared to the state-of-the-art RNN model by Helmy for all three directions (all six origin-destination pairs). The metrics used for the evaluation of the prediction models are:

- the root mean squared error (RMSE) for the absolute accuracy in vehicles per 10 seconds.
- the normalized RMSE (NRMSE) for the relative accuracy. A NRMSE of 1.00 means the model is as accurate as randomly drawing a sample from the data. Anything smaller than 1.00 means the model shows predictive abilities better than random guessing.

Table 8.1 shows the results of the evaluation per origin-destination pair. Remember that there is one prediction model per intersection direction not one per OD, i.e. one for ODs 2 & 3, one for 4 & 6, and one for 7 & 8.

	Origin-Destination pairs					
	2	3	4	6	7	8
NRMSE (original)	0.43	1.00	0.82	0.74	0.74	0.42
NRMSE (LSTM)	0.40	0.97	0.82	0.73	0.71	0.39
RMSE (original)	1.38	0.72	0.70	0.82	0.92	1.16
RMSE (LSTM)	1.29	0.69	0.70	0.81	0.88	1.07
Difference (LSTM vs. original)	-6.6%	-3.6%	+0.3%	-1.2%	-4.0%	-7.6%
Avg. # vehicles/10s time bin	1.29	0.46	0.54	0.66	0.71	1.15

Table 8.1: Performance metrics of the short-term traffic flow prediction models for each origin-destination pair in the case study. The RMSE is given in vehicles per 10 seconds. The datapoints are the averages of 10 runs. The prediction horizon is 30 seconds.

Several things stand out from these results:

- DIRECTOR's LSTM model outperforms the original state-of-the-art model for every OD except OD 4, where the difference could be called negligible. The weighted average, where the number of vehicles per time bin are taken as the weights, shows that the LSTM network outperforms the original RNN by 4.7%.
- Where the state-of-the-art model was not able to reach a NRMSE below 1.00 for one OD pair (OD pair 3), the LSTM model does reach that performance for all OD pairs.
- The models find it relatively easier to predict the traffic flows for the ODs that handle the most traffic (the ODs going straight: 2 and 8). This makes sense as it is harder for the model to predict the rarer occasion that a vehicle will take the turn.
- The performance improvement is more significant for the directions that handle the most traffic (ODs 2 and 8). Apparently the LSTM layer excels at recognizing this type of pattern.

8.2 Predictive controller performance

When evaluating the traffic flow for different traffic light controllers three main metrics are of interest:

- the average delay or travel time per vehicle
- the average number of stops per vehicle
- the cumulative distribution function (CDF) of delay per vehicle

8.2.1 Switching Penalty and Dynamic Green Waves

To evaluate the concept of the switching penalty and dynamic green waves 24-hour simulations of ad-hoc control scenarios with perfect predictions were conducted for various scenarios (using the ad-hoc simulation model with vehicle inputs right behind the arrival detectors as explained in Chapter 7). The average delay and number of stops in each scenarios are given in Table 8.2. The best performing scenario is highlighted in bold. Figures 8.1a and 8.1b show the CDF for the most and least dominant OD, respectively.

Three things stand out from Table 8.2:

- The controller with the switching penalty, but without the dynamic green waves, leads to the best average performance.
- As expected, the switching penalty leads to fewer stops per vehicle.

Scenario				
Prediction type	switching penalty	dynamic green waves	Avg. delay/vehicle (s)	Avg. # stops/vehicle
Perfect knowledge	✓	✓	10.0	0.40
Perfect knowledge	✗	✓	10.4	0.42
Perfect knowledge	✗	✗	9.5	0.42
Perfect knowledge	✓	✗	**9.5**	**0.40**

Table 8.2: Results of 24-hour simulations of the case study intersection for different traffic light controller scenarios to evaluate the impact of the switching penalty and dynamic green waves. The ad-hoc simulation model with vehicle inputs right behind the arrival detectors is used. A ✗ indicates the absence of a feature, a ✓indicates the use of a feature.

- The dynamic green waves concept does not lead to better performance on average. When looking at the individual ODs however[1], the dynamic green waves do lead to better performance for all ODs (see for example Figure 8.1a) except OD 6, which takes a big hit (see Figure 8.1b). That leads to a worse intersection performance overall. This makes sense given the fact that OD 6 services relatively less traffic and conflicts with the two busiest ODs (2 and 8). OD 6 and the average intersection performance would therefore benefit from more short service intervals for OD 6 during platoon gaps at ODs 2 and 8. The dynamic green wave concept reduces the options for such intervals.

Because the results in this section show that the concept of dynamic green waves does not lead to average performance improvements the remainder of the simulation results describes controller implementations without this concept[2].

8.2.2 DIRECTOR's performance

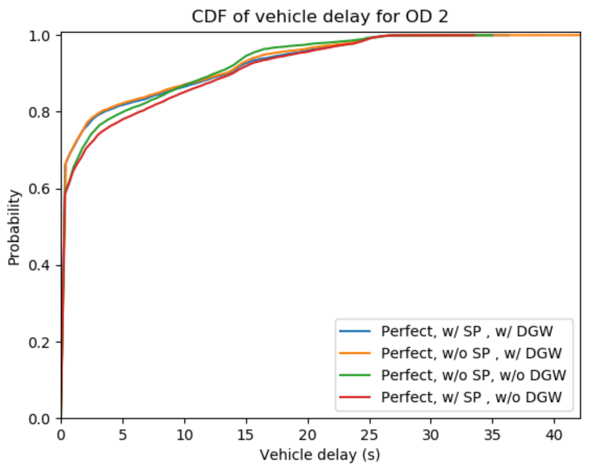
More 24-hour simulations were conducted to evaluate DIRECTOR's performance in comparison to an ideal scenario, the current on-street implementation and a more typical Dutch vehicle actuated controller (a CCOL implementation). The current on-street controller is a state-of-the-art handcrafted actuated controller with green wave coordination. Table 8.3 and Figures 8.2a and 8.2b show the results of these simulations.

The most important takeaways from these results are:

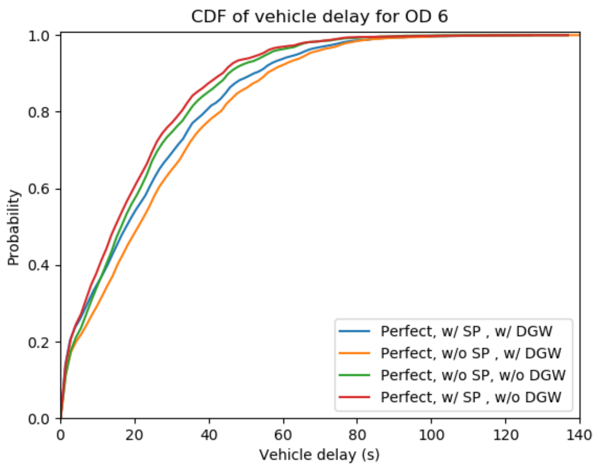
- The ad-hoc scenario with LSTM predictions (as would be used in a deployment on the street) performs only slightly worse than with perfect knowledge of future arrival flows.

[1]Data available in Appendix C.

[2]Other experiments including the use of LSTM predictions and GLOSA conducted with dynamic green waves confirm that the dynamic green waves concept does not improve the overall performance in these circumstances either.



(a) OD 2.



(b) OD 6.

Figure 8.1: Cumulative distribution function (CDF) of delay in several ad-hoc control scenarios for the most and least dominant ODs (ODs 2 and 6, respectively). SP is an acronym for Switching Penalty and DGW for Dynamic Green Waves. Note the different horizontal scales when comparing figures (a) and (b).

- In comparison to the current on-street implementation, the ad-hoc implementation of DIRECTOR performs similar. DIRECTOR performs 1% better in terms of average delay at the cost of a 15% increase in the average number of stops. The perfect predictions case, however, shows DIRECTOR's potential to improve the average delay by 8.7% at an equal average number of stops.

- One difference between DIRECTOR and the current on-street TLC that stands out from Figures 8.2a and 8.2b is the worst-case delay, which is a lot bigger for the current on-street TLC. This can have two causes: 1) The simulation replay is not perfect meaning that some vehicles do not catch a green signal in the simulation when they would in reality. 2) The on-street TLC uses a different trade-off between overall performance and fairness.

- The fixed-ahead scheduling mode suffers more from decreasing prediction accuracy than the ad-hoc mode. This makes sense as the relative importance of the predictions is greater in the fixed-ahead scheduling mode.

- Both DIRECTOR and the current on-street controller strongly outperform the typical vehicle actuated controller. Table 8.3 shows that

DIRECTOR achieves a delay reduction of 39% in ad-hoc mode and 23% in fixed-ahead mode. This shows the added value of green wave coordination.

- Figure 8.2 shows that DIRECTOR achieves green wave coordination similar to the current on-street controller. However, DIRECTOR achieves this implicitly instead of through handcrafting such a mechanism explicitly.

Scenario			
Prediction type	Scheduling mode	Avg. delay/vehicle (s)	Avg. # stops/vehicle
Perfect	Ad-hoc	9.5	0.40
LSTM	Ad-hoc	10.3	0.46
Perfect	Fixed-ahead	10.4	0.36
LSTM	Fixed-ahead	12.9	0.43
Current on-street implementation		10.4	0.40
Vehicle actuated (CCOL)		16.8	0.43

Table 8.3: Results of 24-hour simulations of the case study intersection for different traffic light controller scenarios to evaluate the impact of the scheduling mode and the prediction quality. The ad-hoc simulation model with vehicle inputs right behind the arrival detectors is used. Fixed-ahead means the schedule was fixed 10 seconds in advance.

8.2.3 Stabilization

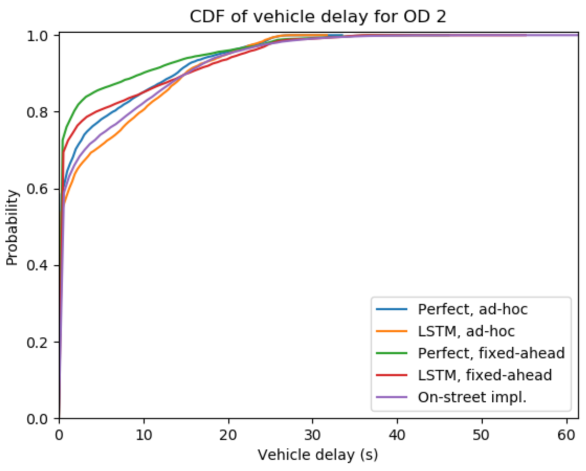
All simulation results in this chapter include the use of the stabilization mechanism. Simulations that were run without the stabilization mechanism showed that instability occurs because of two reasons:

1. Too many vehicles arriving from all directions during the peak hours, which leads to an oscillating schedule that is unable to service all arriving vehicles.
2. Vehicles being missed by the dual lane arrival detectors, which leads to incorrect queue length estimations.

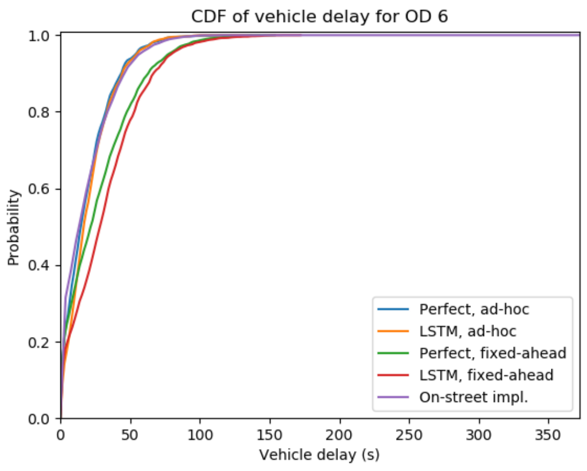
The stabilization mechanism resolves both problems and leads to the results in this chapter.

8.3 Advanced driver assistance system performance

Table 8.4 and Figures 8.3a and 8.3b summarize the results of the simulations that show DIRECTOR's performance when fixing the schedule ahead



(a) OD 2.



(b) OD 6.

Figure 8.2: Cumulative distribution function (CDF) of delay in several scenarios for the most and least dominant ODs (ODs 2 and 6, respectively). Note the different horizontal scales when comparing figures (a) and (b).

and applying GLOSA. When calculating the GLOSA a minimum speed of 20 km/h and a maximum speed of 50 km/h was applied. The GLOSA was updated every 5 seconds.

Since the target speed of the vehicles varies in some of these scenarios the delay measured by the 'vehicle travel times'-detectors is no longer accurate. Therefore the travel time is taken as the metric for traffic flow instead in these scenarios.

Scenario					
ID	Prediction type	Control type	GLOSA penetration rate	Avg. travel time/vehicle (s)	Avg. # stops/vehicle
1	LSTM predictions	Ad-hoc	n\a	25.4	0.50
2	LSTM predictions	Fixed-ahead	0%	29.0	0.50
3	Replay scenario 2 w/ GLOSA		100%	28.7	0.35
4	LSTM predictions	Fixed-ahead	100%	28.6	0.36

Table 8.4: Results of 24- hour simulations of the case study intersection for different traffic light controller scenarios. All scenarios use the GLOSA model, where the vehicle inputs are located further upstream of the arrival detectors. NB: the difference between scenario 3 and 4 is that scenario 3 replays the recorded schedule from scenario 2. Scenario 4 performs live control while using GLOSA, i.e. the schedule is generated during the simulation similar to scenarios 1 and 2.

Some important observations can be drawn from Table 8.2:

- Fixing the schedule ahead leads to degrading performance (up to 14%

increase in travel time). This makes sense as it requires more (possibly invalid) assumptions to fix the schedule in advance. This is also in accordance with the results from Section 8.2.2.

- Scenarios 3 & 4 show that applying a very simple form of GLOSA with DIRECTOR reduces the average number of stops per vehicle by 30% at the cost of a 13% increase in travel time compared to the ad-hoc control. Note that this is for a very simple implementation of GLOSA. The implementation assumes that there are no vehicles queued and that acceleration and deceleration is instant. Hence, the final result with a more accurate GLOSA implementation will be better, reducing the travel time and number of stops.

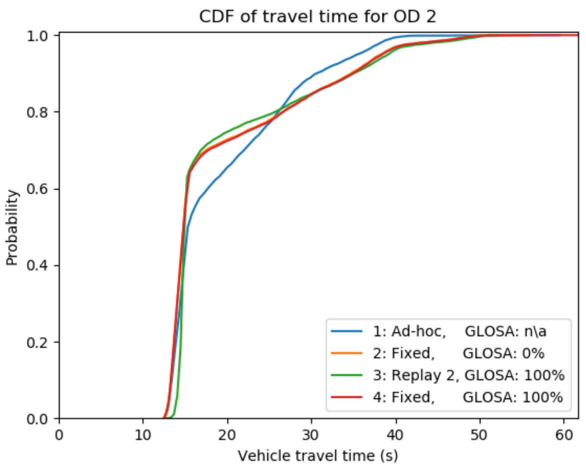
- The fact that the results from scenarios 3 and 4 differ shows that the scheduling decision making is affected by the use of green light optimal speed advice. Simulations show that this is because the vehicle arrivals are influenced by the green light optimal speed advice. Because the prediction model is not trained on data with GLOSA applied its predictions are inaccurate. DIRECTOR therefore takes different scheduling decisions, which lead to different performance results. In practice online learning could be used to teach the model the new arrival flow patterns with GLOSA. This can be expected to improve performance according to the self-enforcing cycle in Figure 3.2.

8.4 Discussion

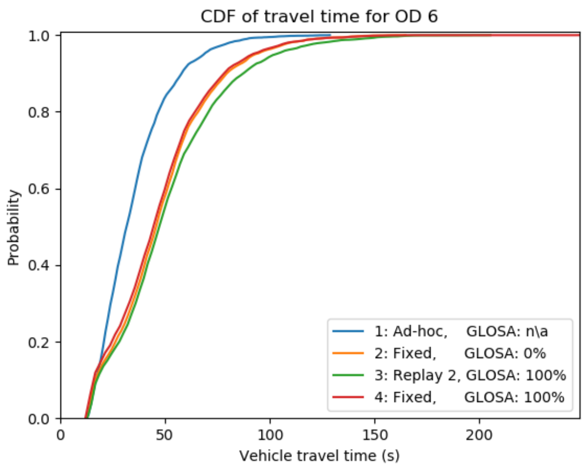
To place the results from the case study into context and evaluate how well DIRECTOR can be expected to perform under different circumstances several experiments have been performed.

To investigate the influence of the prediction quality on the performance, simulations have been run where Gaussian noise is added to the perfect predictions in order to simulate noisy predictions. Table 8.5 shows the results of these simulations. The distribution of the noise is centered around zero and the standard deviation of the distribution is equal to the multiplication factor (second column of the table) times the standard deviation of the historic vehicle arrivals of that origin-destination pair. Negative predictions are corrected to zero to comply with real-life circumstances.

The results show that in ad-hoc mode DIRECTOR's performance degrades only a little and, hence, the controller is quite robust against prediction inaccuracies. For very little noise (0.5 times the standard deviation) the performance is even negligibly better, but that is most likely a coincidence caused by the modeling heuristics. In fixed-ahead scheduling mode DIRECTOR can deal quite well with small prediction inaccuracies, but performance degrades a lot faster for increasingly noisy predictions compared



(a) OD 2.



(b) OD 6.

Figure 8.3: Cumulative distribution function (CDF) of travel time in the control scenarios from Table 8.4 for the most and least dominant OD (ODs 2 and 6, respectively). GLOSA indicates the GLOSA penetration rate. Note the different horizontal scales when comparing figures (a) and (b).

to ad-hoc mode. This makes sense given the heavy reliance on accurate predictions in the fixed-ahead scheduling mode.

Experiments were also conducted to evaluate the impact of dual lane versus single lane arrival detectors. The case study is equipped with dual lane arrival detectors, which have the downside that they occasionally miss vehicles, leading to underestimation of the queue length. Table 8.6 describes the results of simulations both with single and with dual lane detectors. The results show a minor impact. This can be explained by the fact that, although the single lane detectors do not miss vehicles, the single lane detectors occasionally double count vehicles. This double-counting leads to overestimation of the queue length. Thus, one inaccuracy is replaced by another and the results remain approximately the same.

The LSTM model that predicts the vehicle arrivals relies on upstream detectors being available to count the upstream vehicle departures. There are many intersections where such upstream arrival detectors are not yet available. In such cases a different prediction mechanism might be attempted of which one of the simplest would be to use a moving average (MA) of the past arrivals as a prediction for future arrivals. Such a scenario was simulated and compared to the other prediction mechanisms. Table 8.7 shows the results. These show that it would likely be possible to operate DIRECTOR in such cases, but for improved performance upstream detectors

Scheduling mode	**Std. dev. mult. factor**	**Avg. delay/vehicle (s)**	**Avg. # stops/vehicle**
Ad-hoc	0	9.5	0.40
Ad-hoc	0.5	9.4	0.40
Ad-hoc	1	9.7	0.41
Ad-hoc	2	10.2	0.43
Ad-hoc	4	11.3	0.46
Fixed-ahead	0	10.4	0.36
Fixed-ahead	0.5	10.7	0.37
Fixed-ahead	1	11.3	0.40
Fixed-ahead	2	12.9	0.44
Fixed-ahead	4	14.9	0.48

Table 8.5: Results of simulations with predictions where Guassian noise is added to perfect predictions. The distribution of the noise is centered around zero and the distribution's standard deviation is equal to the multiplication factor times the standard deviation of the historical vehicle arrivals.

Scheduling mode	**Predictions**	**Detector**	**Avg. delay/vehicle (s)**	**Avg. # stops/vehicle**
Ad-hoc	Perfect	Dual	9.5	0.40
Ad-hoc	Perfect	Single	9.7	0.39
Ad-hoc	LSTM	Dual	10.3	0.46
Ad-hoc	LSTM	Single	10.4	0.44
Fixed-ahead	Perfect	Dual	10.4	0.36
Fixed-ahead	Perfect	Single	10.0	0.36
Fixed-ahead	LSTM	Dual	12.9	0.43
Fixed-ahead	LSTM	Single	12.7	0.44

Table 8.6: Results of simulations evaluating the impact of using single lane arrival detectors versus dual lane arrival detectors.

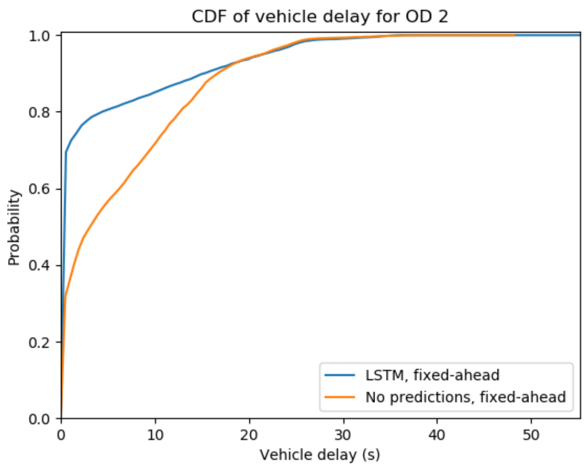
would be preferred. Another downside of using such a simple prediction mechanism compared to the LSTM model is that the LSTM model would improve over time according to the cycle from Figure 3.2, which would be unlikely for the simpler model.

Another interesting test case from Table 8.7 is the scenario with failing detectors or connections that would lead to predictions being unavailable. In that case the schedule would be based purely on the measured queue length. This scenario is simulated by setting all predictions equal to zero. Table 8.7 shows that for fixed-ahead scheduling such a situation would have a relatively small impact on the delay (it actually improves compared to the LSTM predictions), but a major impact on the number of stops. Scheduling in fixed-ahead mode purely based on the measured queue length means that the controller responds with a delay and without implicit green waves. This leads to more intermediate phase group switches servicing the less crowded origin-destination pairs more often. This frequent switching and delay lead to an increase in the number of stops. Figure 8.4 confirms this.

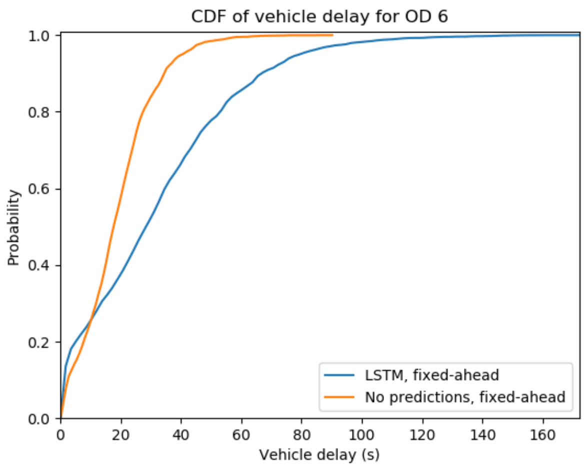
The experiments in this section show that DIRECTOR is quite robust

Predictions	**Avg. delay/vehicle (s)**	**Avg. # stops/vehicle**
Perfect	10.4	0.36
LSTM	12.9	0.43
MA	13.4	0.48
No predictions	12.1	0.59

Table 8.7: Comparison of different prediction mechanisms. All scenarios were run in fixed-ahead scheduling mode. MA predictions means that a 5-minute moving average was used as predictions.



(a) OD 2.



(b) OD 6.

Figure 8.4: Cumulative distribution function (CDF) of vehicle delay in the control scenarios from Table 8.7 for the most and least dominant OD (ODs 2 and 6, respectively). Note the different horizontal scales when comparing figures (a) and (b).

against inaccurate predictions and that there is good reason to believe that DIRECTOR will also perform well at other intersections.

Chapter 9

Conclusions and Future Work

Traffic congestion at signalized intersections is a big economical and ecological problem. Traffic congestion at signalized intersections is a big economical and ecological problem. Handcrafted traffic light controllers (TLCs) are currently used to minimize the impact, but they are expensive to design and maintain and their performance degrades over time as the traffic situation changes. Predictive TLCs and advanced driver assistance systems (ADAS) form a potential solution but are still unfeasible in practice today because of their computational complexity and unpredictability. The question is therefore whether a model predictive control-based signalized intersection controller can lead to stabilized signal phase and timing information that enables advanced driver assistance systems use-cases while maintaining traffic flow efficiency at least as good as a state-of-the-art actuated traffic controller?

The distributed predictive controller developed in this thesis, called DIRECTOR, is feasible and enables time to green/red and green light optimal speed advice (GLOSA) systems. DIRECTOR utilizes predictions of the arriving traffic flows and a model of the current queue length to optimize the traffic light schedule. It can operate in two modes; Ad-hoc mode, where the schedule is generated and applied right away, and fixed-ahead mode, where the schedule is fixed in advance to enable ADAS. DIRECTOR's design makes it scalable and suitable for live learning, eliminating the need for expensive (re)calibrations and improving its performance with more and better data, which will become available in the near future through the availability of floating car data and the use of ADAS.

DIRECTOR reevaluates its traffic light schedule at a fixed frequency (a 10-second interval is used throughout this thesis). When deciding which lights to turn green or red DIRECTOR calculates the expected cumulative delay the vehicles at the intersection will experience based on the measured queue lengths and predicted vehicle arrivals. Instead of predicting the individual vehicle arrivals, which is complex and computationally intensive, DIRECTOR predicts the vehicle flows per time interval. To account for the time that no vehicles can be serviced due to signal changes, a switching penalty is introduced. Stability is guaranteed through a stabilization mechanism, which guarantees worst-case operation equal to a stable, fixed-time traffic light controller.

To predict the arriving traffic flows a long short-term memory recurrent neural network is developed. On a case study intersection that connects two provincial roads this network proves to be on average 4.7% more accurate than the current state-of-the-art model, which is significant for the controller's performance.

Simulations of the case study intersection, which is currently equipped with a state-of-the-art actuated controller with green wave coordination, show that in ad-hoc mode DIRECTOR performs similar to the current controller. DIRECTOR reduces the average delay per vehicle by 1% (from 10.4s to 10.3s) at the cost of an increase of 15% in the average number of stops per vehicle (from 0.40 to 0.46) compared to the current controller. Simulations with ideal predictions show that, in ad-hoc mode, DIRECTOR has the potential to improve the average delay by 8.7% (from 10.4s to 9.5s) while keeping the number of stops equal (at 0.40).

Simulations in fixed-ahead mode, with the schedule fixed 10 seconds in advance and using a simple GLOSA implementation, show a reduction in the average number of stops of 30% at the cost of a 13% increase of the average travel time compared to the ad-hoc mode. Combining this with ideal predictions shows that DIRECTOR in fixed-ahead mode shows the potential to keep the average delay equal compared to the current on-street controller while significantly reducing the number of stops, which would greatly improve traffic flow ecologically and economically.

Compared to a more typical Dutch actuated controller, DIRECTOR achieves a delay reduction of 39% in ad-hoc mode and 23% in fixed-ahead mode.

From these results the research question cannot be answered conclusively. On the one hand DIRECTOR performs approximately equal to the current state-of-the-art controller on the street and shows great potential for improving the traffic flow in ad-hoc mode. On the other hand in fixed-ahead mode, which is required to enable ADAS, DIRECTOR performs worse in terms of delay, but better in terms of stops. Simulations do show DIRECTOR's potential to outperform the current state-of-the-art controller given more accurate predictions, which are likely to be achieved in the near future given the rise of floating car data.

Several things are still worth investigating as future work. For example, some decisions are still based on heuristics where they could be data-driven. The queue dispatch rate is currently fixed but could be made data-driven or based on a sigmoid function like in the SURTRAC system [46]. DIRECTOR's scheduling model could be extended with priority weights as described in Chapter 5. An automated way of generating the optimal weights could be using a genetic algorithm where only the boundary conditions are set by the road owner.

Also, the current stabilization mechanism only looks at the stability at the arrival detectors. To avoid gridlocks in dense networks the system should also look at the queues beyond the intersection. A possible solution would be to integrate the concept of self-healing networks [33].

To further improve the prediction accuracy floating car data (FCD) could be included as model inputs. When sufficient FCD is available the prediction performance is likely to approach ideal predictions. The inclusion of signal timing information in the model could also be of value. Another important contribution would be to apply online learning in order to continuously tune the prediction model.

When further evaluating DIRECTOR in simulations it would be interesting to investigate its performance in different circumstances, e.g. in dense city networks or very remote and relatively quiet locations. Furthermore, it would be worth investigating how a network of DIRECTOR-controlled intersections performs and if it is possible to expand the prediction horizon without compromising performance to enable ADAS over longer distances.

A practical evaluation would also be highly valuable to better evaluate DIRECTOR's performance and get a better understanding of the impact of road user behavior. At the time of writing, preparations for a factory acceptance test[1] are in progress and a road owner has expressed his interest in conducting a pilot on the street.

Overall, DIRECTOR is a new, distributed, data-driven traffic light controller that is easy to set up, reduces costs, can enable advanced driver assistance systems, is futureproof and has the potential to greatly improve traffic flow.

[1]Prerequisite for on-street deployment.

Bibliography

[1] Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from `https://www.tensorflow.org/`.

[2] Behrang Asadi and Ardalan Vahidi. Predictive cruise control: Utilizing upcoming traffic signal information for improving fuel economy and reducing trip time. *IEEE transactions on control systems technology*, 19(3):707–714, 2011.

[3] Steffen Axer and Bernhard Friedrich. Estimating signal phase and timing for traffic actuated intersections based on low frequency floating car data. In *IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2059–2064, 2016.

[4] Mirko Barthauer and Bernhard Friedrich. Evaluation of a signal state prediction algorithm for car to infrastructure applications. *Transportation Research Procedia*, 3:982–991, 2014.

[5] Platform Beter Benutten. Talking traffic. `https://www.beterbenutten.nl/talking-traffic`. Last seen on: 8 February 2018.

[6] Platform Beter Benutten. Talking traffic use cases. `https://www.beterbenutten.nl/nieuws/1013/gebruikstoepassingen-talking-traffic-de-use-cases`, 2016. Last seen on: 7 February 2018.

[7] Priscilla Nagashima Boyd. A study of machine learning algorithms and their suitability for predicting traffic signal timing. Msc thesis, Kellogg College, University of Oxford, 2017.

[8] Werner Brilon and Hendrik Zurlinden. *Überlastungswahrscheinlichkeiten und Verkehrsleistung als Bemessungskriterium für Straßenverkehrsanlagen*. Bundesministerium für Verkehr, Bau-und Wohnungswesen, Abteilung Straßenbau, Straßenverkehr, 2003.

[9] Jeroen Brouwer and Bas van der Bijl. Cutting through the big data hype - Big data and intelligent traffic light controllers for predictive traffic management services. In *Proceedings of the 25th ITS World Congress*, Copenhagen, Denmark, September 2018.

[10] Eduardo F. Camacho and Carlos Bordons. *Model predictive control in the process industry: Advances in industrial control*. Springer, London, 1995.

[11] François Chollet et al. Keras. `https://github.com/keras-team/keras`, 2015. Last seen on: 31 May 2018.

[12] Giovanni De Nunzio, Carlos Canudas Wit, Philippe Moulin, and Domenico Di Domenico. Eco-driving in urban traffic networks using traffic signals information. *International Journal of Robust and Nonlinear Control*, 26(6):1307–1324, 2016.

[13] Bart de Schutter, Hans Hellendoorn, Andreas Hegyi, M. van den Berg, and Solomon K. Zegeye. *Intelligent Infrastructures*, chapter 11, pages 277–310. Springer, 2010.

[14] J. Douma. V-log protocol en definities. Technical report, Vialis B.V., March 2017.

[15] Kurt Dresner and Peter Stone. Multiagent traffic management: A reservation-based intersection control mechanism. In *Proceedings of the third international joint conference on autonomous agents and multiagent systems*, volume 2, pages 530–537. IEEE Computer Society, 2004.

[16] Kurt Dresner and Peter Stone. Multiagent traffic management: An improved intersection control mechanism. In *Proceedings of the fourth international joint conference on autonomous agents and multiagent systems*, pages 471–477. ACM, 2005.

[17] Kurt Dresner and Peter Stone. A multiagent approach to autonomous intersection management. *Journal of artificial intelligence research*, 31:591–656, 2008.

[18] Darrell Etherington. Uber CEO hopes to have self-driving cars in service in 18 months. `https://techcrunch.com/2018/01/23/uber-ceo-hopes-to-have-self-driving-cars-in-service-in-18-months/`, 2018. Last seen on: 8 February 2018.

[19] S. Alireza Fayazi, Ardalan Vahidi, Grant Mahler, and Andreas Winckler. Traffic signal phase and timing estimation from low-frequency transit bus data. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):19–28, 2015.

[20] PTV Group. Vissim. `http://vision-traffic.ptvgroup.com/en-us/products/ptv-vissim/`. Last seen on: 29 May 2018.

[21] Noorhan Helmy. An intelligent traffic flow progression model for predictive control applications. Msc thesis, Delft University of Technology, November 2017.

[22] Nicolaus Henke, Jacques Bughin, Michael Chui, James Manyika, Tamim Saleh, Bill Wiseman, and Guru Sethupathy. The age of analytics: Competing in a data-driven world. *McKinsey Global Institute report*, 2016.

[23] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. Msc thesis, Technische Universität München, 1991.

[24] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[25] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python. `http://www.scipy.org/`, 2001–. Last seen on: 29 May 2018.

[26] Valeriy Khakhutskyy. Signal phase and timing prediction for intelligent transportation systems. Msc thesis, Technische Universität München, November 2011.

[27] Daniel Krajzewicz, Robbin Blokpoel, Alessio Bonfietti, Jérôme Härri, Stefan Hausberger, and Jérémie Dubois-Lacoste. Traffic management based on vehicular communication at low equipment rates. In *Proceedings of the 22nd ITS World Congress*, number ITS-2826, Bordeaux, France, October 2015.

[28] P. R. Kumar and Thomas I. Seidman. Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems. *IEEE Transactions on Automatic Control*, 35(3):289–298, 1990.

[29] Stefan Lämmer. Selbst-gesteuerte lichtsignalanlagen im praxistest. *Straßenverkehrstechnik*, 60(3), 2016.

[30] Stefan Lämmer, Reik Donner, and Dirk Helbing. Anticipative control of switched queueing systems. *The European Physical Journal B*, 63(3):341, 2008.

[31] Stefan Lämmer and Dirk Helbing. Self-control of traffic lights and vehicle flows in urban road networks. *Journal of Statistical Mechanics: Theory and Experiment*, 04(P04019), 2008.

[32] Stefan Lämmer and Dirk Helbing. Self-stabilizing decentralized signal control of realistic, saturated network traffic. *Santa Fe Institute Working Paper*, (10-09-019), 2010.

[33] Stefan Lämmer and Martin Treiber. Self-healing networks - Gridlock prevention with capacity regulating traffic lights. Workshop on technologies for the organisation, adaptation and simulation of transportation systems. In *IEEE Sixth International Conference on Self-Adaptive and Self-Organizing Systems Workshops*, pages 61–65, September 2012.

[34] M.J. Lighthill and G.B. Whitham. On kinematic waves II. A theory of traffic flow on long crowded roads. In *Proceedings of the Royal Society A*, volume 229, pages 317–345, 1955.

[35] Jan Marian Maciejowski. *Predictive control with constraints.* Pearson education, 2002.

[36] Grant Mahler and Ardalan Vahidi. Reducing idling at red lights based on probabilistic prediction of traffic signal timings. In *American Control Conference (ACC)*, pages 6557–6562, June 2012.

[37] Grant Mahler and Ardalan Vahidi. An optimal velocity-planning scheme for vehicle energy efficiency through probabilistic prediction of traffic-signal timing. *IEEE Transactions on Intelligent Transportation Systems*, 15(6):2516–2523, 2014.

[38] Cornelius Menig, Robert Hildebrandt, and Robert Braun. Der informierte Fahrer - Optimierung des Verkehrsablaufs durch LSA-Fahrzeug-Kommunikation. In *HEUREKA '08 - Optimierung in Verkehr und Transport*, 2008.

[39] Christopher Olah. Understanding LSTM networks. `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`, August 2015. Last seen on: 26 March 2018.

[40] James R. Perkins and P.R. Kumark. Stable, distributed, real-time scheduling of flexible manufacturing/assembly/diassembly systems. *IEEE Transactions on Automatic Control*, 34(2):139–148, 1989.

[41] Dennis I. Robertson. TRANSYT: A traffic network study tool. Technical Report LR 253, Road research laboratory, Crowthorne, United Kingdom, 1969.

[42] Dennis I Robertson and R David Bretherton. Optimizing networks of traffic signals in real time-the scoot method. *IEEE Transactions on vehicular technology*, 40(1):11–15, 1991.

[43] Robert Scheepjens. Algorithm design for traffic signal timings predictions of vehicle-actuated controlled intersections using support vector regression. Msc thesis, Delft University of Technology, May 2016.

[44] Nicolas Shield. Waymo and GM are far ahead in self-driving car tests. `http://www.businessinsider.com/waymo-gm-far-ahead-self-driving-car-tests-2018-2`, 2018. Last seen on: 8 February 2018.

[45] Stephen F. Smith, Gregory J. Barlow, Xiao-Feng Xie, and Zachary B. Rubinstein. Smart urban signal networks: Initial application of the SURTRAC adaptive traffic signal control system. In *Proceedings of the Twenty-Third*

International Conference on Automated Planning and Scheduling, pages 434–442, 2013.

[46] Stephen F. Smith, Gregory J. Barlow, Xiao-Feng Xie, and Zachary B. Rubinstein. SURTRAC: Scalable urban traffic control. In *Transportation research board 92nd annual meeting*, number 13-0135, 2013.

[47] Aleksandar Stevanovic, Jelka Stevanovic, and Cameron Kergaye. Comparative evaluation of benefits from traffic signal retiming and green light optimized speed advisory systems. *93rd Annual Meeting of the Transportation Research Board, Washington, DC*, (1476), 2014.

[48] Uber. Our road to self-driving vehicles. `https://www.uber.com/blog/our-road-to-self-driving-vehicles/`, 2017. Last seen on: 8 February 2018.

[49] Volvo Car USA. Car2car green light optimum speed advisory. `https://www.media.volvocars.com/us/en-us/media/photos/49886`, July 2013. Last seen on: 31 May 2018.

[50] Ronald T. van Katwijk, Bart de Schutter, and Hans Hellendoorn. Look-ahead traffic-adaptive signal control. In *Proc. 6th European Congress on Intelligent Transport Systems and Services (ITS07)*, 2007.

[51] Mark VanMiddlesworth, Kurt Dresner, and Peter Stone. Replacing the stop sign: Unmanaged intersection control for autonomous vehicles. In *Proceedings of the seventh international joint conference on autonomous agents and multiagent systems*, volume 3, pages 1413–1416, 2008.

[52] Nianfeng Wan, Ardalan Vahidi, and Andre Luckow. Optimal speed advisory for connected vehicles in arterial roads and the impact on mixed traffic. *Transportation Research Part C: Emerging Technologies*, 69:548–563, 2016.

[53] Xiao-Feng Xie, Stephen F. Smith, and Gregory J. Barlow. Schedule-driven coordination for real-time traffic network control. In *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling*, pages 323–331, 2012.

[54] Xiao-Feng Xie, Stephen F. Smith, Liang Lu, and Gregory J. Barlow. Schedule-driven intersection control. *Transportation Research Part C: Emerging Technologies*, 24:168–189, 2012.

Acronyms

AD-ratio arrival/departure-ratio. 19

ADAS advanced driver assistance systems. iii, 5, 6, 8, 9, 11, 13, 21, 23, 24, 31, 45, 67–69

BPTT backpropagation through time. 14

CaRT classification and regression tree. 13

CDF cumulative distribution function. 58, 60, 62, 64, 66

COLOMBO cooperative self-organizing system for low carbon mobility at low penetration rates. 18

COM component object model. 54

DIRECTOR Data-driven Intersection and Road Environment Controller for Traffic Optimization in Real-time. iii, 9, 23–25, 31, 35, 38, 39, 41, 42, 45, 46, 48, 49, 51, 54, 55, 58–61, 63–69

FCD floating car data. 11, 12, 18, 67–69

FFNN feedforward neural network. 13

FIFO first in first out. 41, 42

GLOSA green light optimal speed advice. iii, 5–8, 11–13, 20, 21, 23, 25, 45, 48, 49, 54, 59, 62–64, 67, 68

GNSS global navigation satellite system. 12

GPS global positioning system. 12

I2V infrastructure to vehicle. 5, 12, 78

kNN k-nearest neighbors. 13

LSTM long short-term memory. iii, 13–15, 28, 31, 33, 57–59, 64, 65, 68

MPC model predictive control. 8, 11, 15, 16, 22, 23, 31, 67

NRMSE normalized root mean squared error. 57, 58

OD origin-destination pair. 1, 31–42, 46, 48, 49, 52, 57–60, 62–66, 83, 84

PG phase group. 39, 41

RMSE root mean squared error. 12, 13, 57

RNN recurrent neural network. iii, 14, 15, 21, 22, 27–29, 57, 58, 68, 79

SAS speed advisory system. 21

SPaT signal phase and timing. 5, 7, 8, 11–13, 18, 20–24, 45, 67

SVR support vector regression. 13

T2G time to green. 12, 49

T2G/R time to green/red. iii, 5–8, 11, 23, 25, 45, 48, 49, 67

T2R time to red. 49

TLC traffic light controller. iii, 7–9, 15, 16, 23, 24, 51, 52, 58–62, 67–69

V2I vehicle to infrastructure. 5, 78

V2V vehicle to vehicle. 20

Glossary

cycle Period within which every signal group will be serviced. 4, 11, 12, 17, 43

floating car data Data generated by road users sharing their location. 11

green wave Traffic control mechanism where a series of connected intersections are synchronized in order to give passage without stops along a particular road segment. 4

intergreen time Minimum time between one signal changing from green to amber and the next signal turning green. This time is chosen such that the intersection is clear before the next signal turns green. 35, 36, 42, 46, 49

intersection Location where multiple roads intersect/connection point of several links. 1

link Road segment that is connected to the intersection. 1, 31, 51

model predictive control Online model-based control approach in which a prediction model and optimization function are used to determine the control actions that optimize a given performance criterion over a given time horizon subject to given constraints. 15

non-conflicting signals Signals that can be serviced simultaneously without the possibility of traffic intersecting with each other. 1

non-peak hour Time outside of the peak hours. 4

offset Coordinated traffic control parameter that is used for synchronization between neighboring intersections. 4, 17

origin-destination pair Pair of matching origin and destination links. 1, 31, 32, 34, 36, 37, 39, 41, 42, 46, 48, 52, 55, 57

peak hour Time of the day when the most traffic is on the road. 4, 12, 16, 18, 61, 77

penetration rate Percentage of users using an application. 18–21

phase Period that a phase group is serviced. 2, 11, 13, 53, 78

phase group A combination of signal groups that the traffic light controller will service simultaneously. 2, 7, 32, 35, 36, 38–43

queue clearance time Time it takes to service an entire queue. 19

saturating flow Maximum vehicle flow (vehicles/time unit) that a road is able to service, i.e. the maximum road capacity. 17

schedule A sequence of phases. iii, 2, 18, 23–25, 40, 41, 43, 45–49, 61–63, 65, 67, 68

servicing Granting a signal group access to the intersection. 1

signal Means to grant or prohibit the traffic on its lane access to the intersection. 1, 51, 53, 55

signal group Combination of signals that are always serviced together and have the same id. 1, 2

sink Location where road users can leave the road. 14, 51

source Location where road users can enter the road. 14, 51

split time The split time of a phase group is the duration of service within a cycle allocated to the phase group. 4

Talking Traffic Project initiated by the Dutch government in collaboration with the industry to develop and deploy the infrastructure required for vehicle to infrastructure and infrastructure to vehicle communication. 5

traffic light controller System that controls the light states of the signals at an intersection. 1, 2, 78

travel time delay Time lost during travel due to non-free flow conditions caused by for example other vehicles or signalized intersections. 2, 16–18, 22, 31–36, 38–40, 55

Appendix A

Helmy's RNN architecture

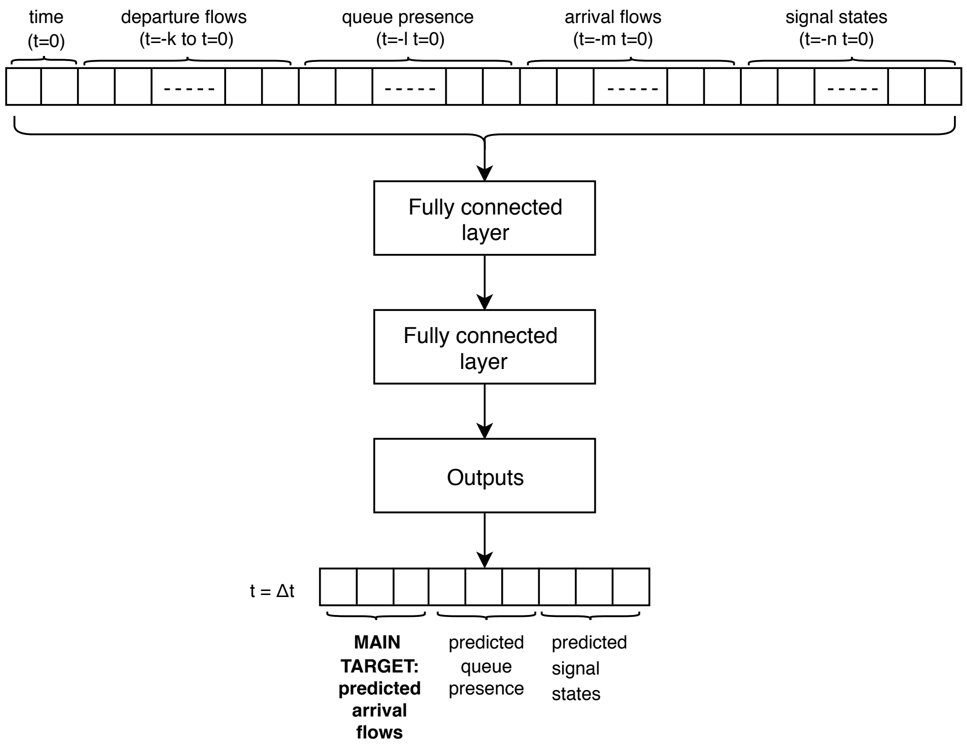


Figure A.1: Schematic drawing of the RNN architecture by Helmy [21]. Each variable is included for a finetuned number of timesteps (indicated by the integers k, l, m and n). Note that the number of included timesteps may differ for each variable.

Appendix B

GLOSA implementation

Listing B.1: Pseudo code of GLOSA implementation. `signal` is the target signal of the vehicle. `distance` is the vehicle's distance to the stopline. `T2R` and `T2G` are the target signal's countdowns.

```
def get_GLOSA(signal, distance, T2R, T2G):
  if (signal is GREEN and T2R is unknown):
    GLOSA = max_speed
  elif (signal is RED and T2G is unknown):
    GLOSA = min_speed
  elif (signal is RED and T2G is known and T2R is unknown):
    GLOSA = min(distance / T2G, max_speed)
  elif (signal is not RED and T2R is known and T2G is unknown):
    speed_to_catch_green_before_red = distance / T2R
    if (speed_to_catch_green_before_red > max_speed):
      GLOSA = min_speed
    else:
      GLOSA = max_speed
  elif (both T2G and T2R are known):
    if (signal is GREEN):
      speed_to_catch_green_before_red = distance / T2R
      if (speed_to_catch_green_before_red < max_speed):
        GLOSA = max_speed
      # If the current GREEN signal cannot be caught
      else:
        speed_to_catch_next_green_after_red = distance / T2G
        if (speed_to_catch_next_green_after_red < min_speed):
          GLOSA = min_speed
        else:
          GLOSA = min(speed_to_catch_next_green_after_red, max_speed)
    # If signal is RED
    else:
        speed_to_catch_green_before_next_red = distance / T2R
        if (speed_to_catch_green_before_next_red > max_speed):
          GLOSA = min_speed
        # If vehicle is able to catch next GREEN before it turns RED again
        else:
          GLOSA = max(speed_to_catch_green_before_next_red, min_speed)
  return GLOSA
```

Appendix C

Simulation results per OD

Scenario	Avg. vehicle delay per OD						Avg. # stops/vehicle per OD					
	2	3	4	6	7	8	2	3	4	6	7	8
Perfect, w/ SP, w/ DGW	3.0	21.5	7.7	22.6	4.0	10.7	0.16	0.71	0.40	0.75	0.19	0.45
Perfect, w/o SP, w/ DGW	2.8	22.1	7.6	25.2	3.9	10.8	0.15	0.76	0.42	0.84	0.20	0.46
Perfect, w/o SP, w/o DGW	3.0	18.6	6.7	20.6	4.2	11.1	0.16	0.73	0.38	0.80	0.22	0.50
Perfect, w/ SP, w/o DGW	3.4	18.4	6.7	19.0	4.3	11.4	0.18	0.66	0.36	0.71	0.21	0.48

Table C.1: Detailed results per OD for the ad-hoc simulation scenarios in Table 8.2. Several shortcuts and acronyms are used to describe the scenarios; Perfect means perfect predictions, SP is an acronym for Switching Penalty and DGW for Dynamic Green Waves.

Scenario	Avg. vehicle delay per OD						Avg. # stops/vehicle per OD					
	2	3	4	6	7	8	2	3	4	6	7	8
Perfect, Ad-hoc	3.4	18.4	6.7	19.0	4.3	11.4	0.18	0.66	0.36	0.71	0.21	0.48
LSTM, Ad-hoc	4.2	19.3	7.7	20.8	5.0	11.6	0.23	0.73	0.44	0.79	0.27	0.54
Perfect, Fixed-ahead	2.4	24.8	8.9	27.4	3.4	8.7	0.12	0.70	0.40	0.77	0.15	0.34
LSTM, Fixed-ahead	3.4	30.9	12.5	31.8	4.7	10.2	0.17	0.79	0.52	0.80	0.23	0.44
Curr. on-street impl.	4.1	22.8	8.4	18.9	5.3	11.2	0.21	0.70	0.38	0.64	0.24	0.45
Vehicle actuated	5.0	40.1	17.7	49.3	4.0	9.3	0.19	0.82	0.54	0.85	0.19	0.37

Table C.2: Detailed results per OD for the simulation scenarios in Table 8.3.

Scenario	Avg. vehicle delay per OD						Avg. # stops/vehicle per OD					
	2	3	4	6	7	8	2	3	4	6	7	8
Perfect	2.4	24.8	8.9	27.4	3.4	8.7	0.12	0.70	0.40	0.77	0.15	0.34
LSTM	3.4	30.9	12.5	31.8	4.7	10.2	0.17	0.79	0.52	0.80	0.23	0.44
MA	4.6	30.5	12.2	31.9	5.2	11.2	0.23	0.84	0.57	0.88	0.26	0.49
No predictions	6.3	18.1	7.3	18.7	7.0	17.4	0.34	0.82	0.44	0.85	0.41	0.79

Table C.3: Detailed results per OD for the simulation scenarios in Table 8.7. The tables shows the results of a comparison of different prediction mechanisms. All scenarios were run in fixed-ahead scheduling mode. MA predictions means that a 5-minute moving average was used as predictions. Zeros means that all predictions are set to 0.