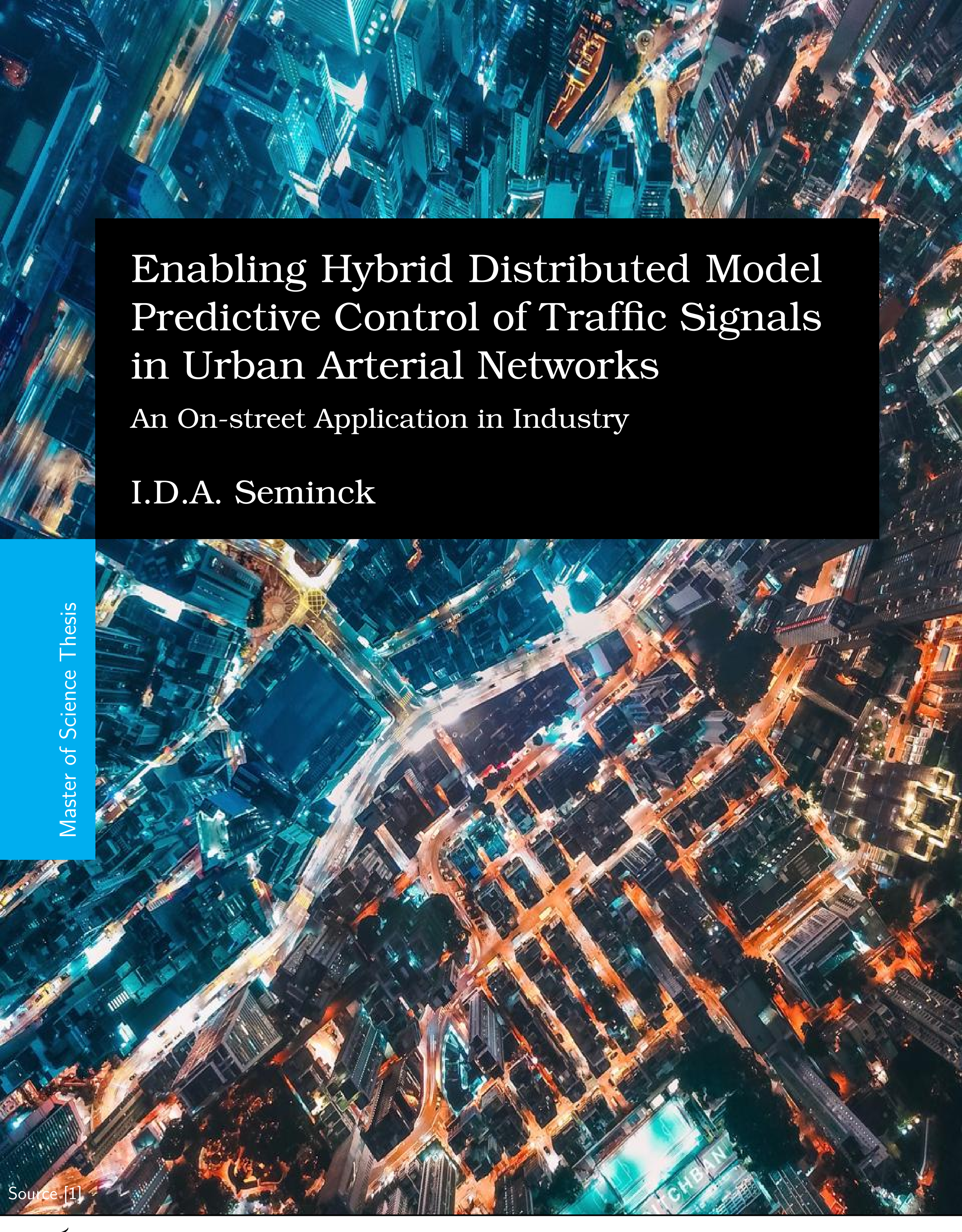


Enabling Hybrid Distributed Model Predictive Control of Traffic Signals in Urban Arterial Networks

An On-street Application in Industry

I.D.A. Seminck

Master of Science Thesis

Source [1]

Delft University of Technology

A Siemens Business

Delft Center for Systems and Control

CONFIDENTIAL

Enabling Hybrid Distributed Model Predictive Control of Traffic Signals in Urban Arterial Networks

An On-street Application in Industry

Master of Science Thesis

For the degree of Master of Science in Systems and Control
at Delft University of Technology

I.D.A. Seminck

March 16, 2022

Faculty of Mechanical, Maritime and Materials Engineering (3mE) ·
Delft University of Technology

The work in this thesis was supported by Yunex Traffic b.v. (previously Siemens Mobility).

Abstract

Road users still encounter unnecessary delays due to inefficient traffic control in urban traffic networks. These unnecessary delays are ever increasing and have large environmental and economic consequences. In order to reduce these delays, one of the lower cost alternatives to constructing new roads is to use the current infrastructure more efficiently by employing advanced traffic signal controllers at urban signalised intersections.

In the Netherlands, most intersections are controlled using actuated controllers, which respond directly to the current traffic demand obtained from vehicle detections at the intersection. This causes the signal timing to be myopic and coordination between intersections is limited. These features also make the actuated control method not suited for implementation with in-vehicle information services such as Green Light Optimised Speed Advice (GLOSA), where the signal timings of the traffic controller need to be predictable. A control method such as Model Predictive Control (MPC) is well suited to resolve the shortsightedness and limited coordination of the actuated controller. MPC is a model-based optimal control method that uses a prediction model and optimisation to compute the best current signals, while still taking the future evolution of the traffic system into account. MPC also provides many opportunities for coordination between multiple controllers in a network, since a great deal of distributed and hierarchical MPC methods have been developed.

In this thesis, a new Mixed Logical Dynamical (MLD) model of a signalised intersection is developed for controlling traffic signals in urban traffic networks with the goal of minimising vehicle delays. In addition, constraints are modelled for on-street application. The developed model and constraints are then used as the prediction model of a MPC controller and applied decentralised at an urban arterial intersection. A distributed algorithm is also developed to coordinate the signals of intersection controllers in urban traffic networks.

Simulation results show that the mean delay time per vehicle of the developed Decentralized Model Predictive Controller (DeMPC) is 24% lower than the greedy control method of Yunex and only 8% above the actuated controller. The DeMPC outperforms both actuated and Yunex's method in terms of the total number of stops by 10% and 26%, respectively . For MPC, the vehicle delay time is mainly affected by the control time step, mismatches between the MLD intersection model and the real-world traffic system, and the accuracy of the arrival prediction methods. For future work, the distributed coordination algorithm should be evaluated by simulation and the mismatches between both intersection and arrival models and the traffic system should be reduced. Additionally, the DeMPC should be evaluated in combination with GLOSA.

Table of Contents

List of Figures

List of Tables

Glossary

List of Acronyms

TU Delft	Delft University of Technology
OD	Origin-Destination
TSC	Traffic Signal Control
GLOSA	Green Light Optimised Speed Advice
iTLC	intelligent Traffic Light Controller
V2I	Vehicle-to-Infrastructure
I2V	Infrastructure-to-Vehicle
T2G/R	Time-to-Green/Red
FCD	Floating Car Data
MPC	Model Predictive Control
SPaT	Signal Phase and Timing
CTM	Cell Transmission Model
MILP	Mixed-Integer Linear Programming
LSTM	Long Short-Term Memory
I2I	Infrastructure-to-Infrastructure
MLD	Mixed Logical Dynamical
DIRECTOR	Data-driven Intersection and Road Environment Controller for Traffic Optimisation in Real-time
RMSE	Root Mean Square Error
NRMSE	Normalised Root Mean Square Error
DeMPC	Decentralized Model Predictive Controller
DiMPC	Distributed Model Predictive Controller
TLC	Traffic Light Controller
HCM	Highway Capacity Manual

CVNIP	Contactgroep Verkeersregeltechnici Nederland Internet Protocol
XML	Extensible Markup Language
V-Log	Verkeerskundinge Log
ETSI	European Telecommunications Standards Institute

Acknowledgements

This thesis is the final stage in completing my master degree in Systems and Control. As this thesis project is the result of collaboration, I want to express my appreciation to all people who have assisted me during this project.

Firstly, I would like to express my gratitude to my TU Delft supervisor, Sergio Grammatico, for the responsive and valuable feedback and enriching discussions concerning my methodology. Sergio encouraged me to be unique in my methodology while also resolving the problem posed within industry. Next to that, I would like to thank him for being understanding of the challenging implementation at Yunex. Additionally, I would like to thank PhD student Mattia Bianchi for the enlightening conversation regarding distributed optimisation techniques in the beginning of the thesis project.

Next, I would like to thank everyone within Yunex for the opportunity to pursue my thesis project within the industry. Specifically, I would like to thank my supervisor, Alexander Koek. He was available to help me anytime with the implementation of my methods within Yunex's systems and was clear and transparent in his communications about technical and organisational topics. The feedback and insights from his industry perspective were of great value to this project. Likewise, I would like to appreciate the assistance of Hans Looijen, who was always ready to support me with the implementation of the simulation environment in Aimsun.

I would also like to thank my friends for all their support, advice, and memorable times together. All the fun activities that we did helped me to clear my mind and recharge my batteries. In particular, I would like to thank Kaat, my girlfriend, for being an excellent conversation partner and for her support during this challenging project, which broadened my knowledge and deepened my technical and personal skills. Finally, I wish to express my gratitude to my parents for enabling me to pursue an excellent education at TU Delft and for their continuous guidance and encouragement.

Delft, University of Technology
March 16, 2022

I.D.A. Seminck

Chapter 1

Introduction

Urban road traffic networks are frequently not used to their full capacity as road users still encounter unnecessary delays due to inefficient traffic control [2]. These unnecessary delays result in stop-and-go driving, which causes the majority of emissions during urban trips [3]. During 2019, the road transport sector accounted for 20% of all European greenhouse gas emissions [4]. Next to the environmental impact, delays increase travel costs with congestion resulting in nearly EUR 100 billion per year within the European Union (EU) or 1% of its GDP [5]. In the Netherlands, traffic jams increased by 17% during 2019 and the number of registered vehicles and trips is ever increasing [6, 7]. It is thus clear that measures should be taken to reduce the unnecessary delays encountered in urban road traffic networks as these have large environmental and economic consequences.

There are several strategies to reduce delays in road traffic networks: constructing or designing new roads, introducing road tolls, and promoting other means of transport [8]. Yet another strategy is to use the current infrastructure more efficiently by employing a range of advanced traffic control measures. Variable speed limits, for example, are used to increase highway traffic flow by adjusting the speed limit to the prevailing traffic conditions and traffic signals at highway on-ramps to ensure that the traffic flow can stay longer below its capacity. On a larger scale, route guidance control measures are used to direct traffic away from highly congested roads and distribute the traffic over the road network more evenly [9]. In urban areas, however, the major bottleneck is the impaired flow at signalised intersections, where traffic from conflicting directions needs to be regulated by traffic signal controllers. These controllers do not only regulate the traffic flow through a single intersection, but have a profound impact on the traffic network as a whole.

Traditionally, traffic signal timings are determined using fixed-time or actuated controllers. Fixed-time traffic signal controllers are optimised off-line based on historical traffic demand data, whereas actuated controllers respond directly to the current traffic demand obtained from detectors located at the intersection. However, this makes the actuated control method also shortsighted and the coordination between intersections is limited. Most of the signalised intersections in the Netherlands are operated by actuated controllers.

In 2017, *Partnership Talking Traffic* was initiated by the Dutch Ministry of Infrastructure and Environment in order to improve the accessibility, flow and road safety of the Dutch traffic infrastructure [10]. In this partnership 30 private companies, including Siemens Mobility (now Yunex Traffic), develop services for connected transportation systems. These are transportation systems in which infrastructure-to-infrastructure, vehicle-to-infrastructure and infrastructure-to-vehicle communication is enabled. One application in such a connected system is in-vehicle information services such as Time-to-Green/Red or Green Light Optimised Speed Advice. These services inform the road user on the timing of the current and future traffic signals or the corresponding advised speed. In this way, drivers can anticipate a signal change resulting in faster queue discharge times, higher platoon ratios, and smoother vehicle profiles on urban roads. All of which leads to decreased vehicle delays and emissions [11].

To implement such services, the timing of the traffic controller's signals need to be known in advance, i.e., the timings need to be predictable. This requirement makes the actuated control method not suited for connected transportation systems as its signal timings vary depending on the vehicle detections in the vicinity of the intersection [12]. Consequently, there is a high demand in the Netherlands for traffic signal controllers which produce signal timings that are predictable for connected services to be implementable and flexible to adapt to the current traffic demand. Additionally, there is also a high demand for coordination of traffic signals in a network of intersections, since this can create green waves in which vehicles can drive through several intersections without stopping. This further decreases the delay and enhances the performance of the connected services [13].

Model Predictive Control (MPC) [14] has proven to decrease delays in urban traffic networks [15, 16]. MPC is an optimal model-based predictive control method that uses a model to predict the future evolution of the (traffic) system and an optimisation algorithm to compute the best sequence of control actions (signals) over a prediction horizon. Only the first control action in this sequence is applied to the system. In this way, MPC makes the best current control action while still taking the longer term effects into account. Next to that, this method can handle a user-defined objective function and hard safety constraints. MPC is thus well-suited to resolve the issue of providing in-vehicle information services posed above.
To control a network of intersections, not one controller (centralised control) but multiple controllers (multi-agent control) are used. Centralised control is not easily scalable, since it has reliability issues and other disadvantages involving unavailability of control and data access due to legal or commercial constraints [17]. Therefore, hierarchical, distributed, and decentralised multi-agent structures are used to control traffic networks.

This thesis focuses on the development and on-street implementation of a distributed model predictive traffic signal controller to reduce delays in urban traffic networks. The next chapter will start with a background on traffic signal control by explaining the used terminology, current traffic infrastructure, and basic traffic dynamics on urban signalised roads. In chapter 3, the related work is provided to understand the state-of-the art on optimal model-based traffic signal control in urban traffic networks. The conclusion of the related work together with the problem posed within industry sets the problem statement, objective and scope of the thesis. In chapter 4, the intersection model is developed and reformulated into a mixed-logical dynamical system. Additionally, the constraints are formulated to enable on-street application of the controller. In chapter 5, a distributed MPC algorithm is developed in order to coordinate multiple intersection controllers in urban traffic networks. Thereafter, in chapter 6, both decentralised and distributed controllers are implemented and the simulation within Yunex's simulation environment is setup. Finally, the thesis is completed with the results and discussion of the decentralised MPC simulations. After which the conclusions and recommendations for future work are formulated.

Chapter 2

Background

In this chapter, the terminology, infrastructure, and basic traffic dynamics relevant for Traffic Signal Control (TSC) are explained to provide the necessary background for the traffic models discussed in the literature review and the chapters thereafter. In section 2-1, the used terminology and basic concepts are defined. Next, in section 2-2, the current traffic infrastructure in the Netherlands is explained in order to describe the available measurements and communication possibilities for TSC. Lastly, in section 2-3, the basic traffic dynamics in signalised urban links is presented.

2-1 Terminology

In a traffic network, an intersection connects a number of links from different directions. The path that a vehicle takes when it crosses an intersection from one lane in an approaching link to a leaving lane is called an Origin-Destination (OD) pair or movement. Traffic signals control these movements to ensure safe and efficient passage of vehicles and other road users. The most basic controlled element of a traffic signal controller is a signal group, which is defined as a set of traffic signals that always display the same colour. In Figure 2-1, an example of the Dutch signal group coding scheme is depicted. A signal group usually coincides with a movement, but movements can be combined into one signal group. For example, signal group six in Figure 2-1 combines a left-turn and through-going movement. A pair of signal groups is called conflicting when the paths of their movements cross each other in the intersection area. This crossing is then called the conflict area of these two movements. The other basic controlled element is a phase group[1], which consist of a set of nonconflicting signal groups. For example, signal groups ten and twelve can be combined into a phase group, since their movements do not contain any conflicts.

The locations of four conflict areas are shown in Figure 2-1 for signal group pairs $(2, 12)$, $(2, 10)$, $(6, 12)$ and $(6, 10)$. To guarantee safety, protected conflicts prohibit two signal groups to display a green signal at the same time, as opposed to permitted conflicts, where signals can be green

[1]The Dutch TSC terminology uses signal groups and phase groups, however in the literature, signal groups and phase groups are called phases and stages, respectively. In this thesis, the Dutch terminology is used.

concurrently. Signal group pairs, such as $(6, 10)$ and $(6, 12)$, for example, are considered permitted conflicts as they contain a turn movement and a trough-going movement [13]. A signal

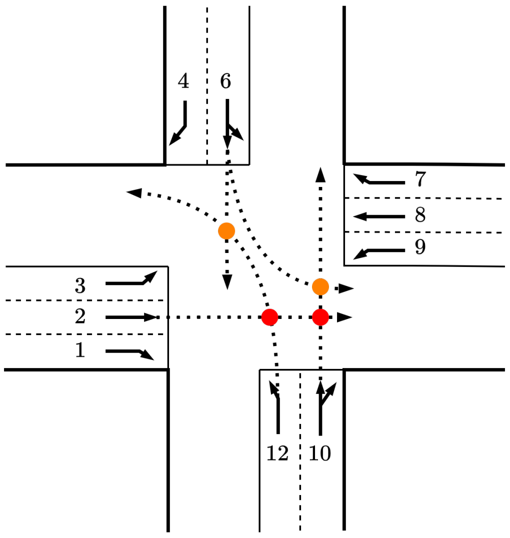


Figure 2-1: Dutch signal group coding scheme of signal group numbers from 1 to 12. The dotted arrows indicate the movement paths, while the bold arrows indicate the arrows painted on the road at the intersection. The red dots specify the protected conflicts and the amber dots the permitted conflicts.

group can safely switch to a green signal if all vehicles from the conflicting signal groups have cleared the intersection. In this way, it is ensured that vehicles can safely enter and cross the intersection without any collisions with the last vehicle(s) from the conflicting signal group. Formally, this is guaranteed by the minimum clearance time, which is defined as the time between the start of red for the conflicting (exiting) signal group and the start of green of the succeeding (entering) signal group.

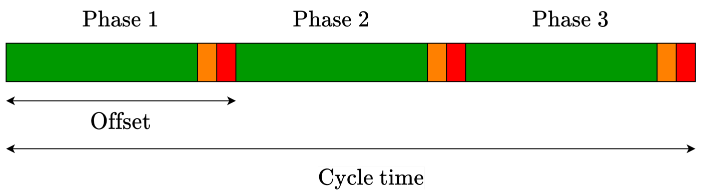


Figure 2-2: Cyclic operation of intersection traffic signals.

Lastly, traffic signals can be scheduled in a cyclic way, where each phase group can be green at most once per cycle. Then, the most basic controlled variable is the green split, which describes the duration of a green signal as a portion of the total cycle time of the intersection. The cycle time is defined as the sum of the duration of all green splits and the time between phase changes due to amber and minimum clearance times. An illustration of the cyclic operation of three phases is shown in Figure 2-2. Additionally, one can also opt to control the cycle time and the offset, which is the time difference between the start of the cycle and the start of a phase. Offsets are mostly used to enable coordination between between neighbouring intersections. Traffic signals can also be scheduled in an acyclic way, where no predefined cycle or signal sequence is set. Then, phase or signal groups switch to green at any given moment in time (taking into account the constraints posed by conflicting movements).

2-2 Traffic Infrastructure

The road traffic infrastructure is well developed in the Netherlands, where urban roads are thoroughly designed and maintained with many measurement detectors and communication devices available. The measurement data used in this thesis are obtained from open loop detectors and communication between intersection control applications is available through Signal Phase and Timing (SPaT) messages. These available measurements and communication messages are described in subsection 2-2-1. In subsection 2-2-2, the connected traffic infrastructure is briefly outlined together with its advantages and future available measurements for TSC.

2-2-1 Measurements and Communication

Inductive loop detectors are embedded in the pavement and measure vehicle occupancy, which denotes the state and duration of a vehicle detection. When a vehicle passes over a loop detector, the inductance of the loop is briefly reduced and the state of the detector switches from *free* to *occupied* to *free*. This sequence of states is then denoted as a detection of one vehicle. When a vehicle stops within the area encompassed by the detector loop, the detector state stays *occupied*. Next to vehicle occupancy, detectors can measure different vehicle types and speeds depending on the size and configuration of the inductive loops [18]. The following types of inductive loop detectors are used:

- The stop line detector is located at the stop line of an approaching intersection lane.
- The long queue detector is located 10 to 20 meters[2] upstream of the stopline.
- The arrival line detector is located between 80 and 120 meters[2] upstream of the intersection. This detector is occasionally installed as a double loop detector, which can detect speed. These detectors can be placed per lane or overlapping several lanes.

The location of the detectors can be seen in Figure 2-3, where a real-world intersection and its detectors are depicted. Detector data is communicated through a binary Verkeerskundinge Log (V-Log) protocol message [19]. SPaT messages are used to communicate the signal timings, current state of the traffic signal controller, and info about the intersection [20]. The message includes a great deal of information, but the most relevant for TSC is

- the intersection identification number, which is used to uniquely define each intersection in a region,
- the status of the intersection controller, which indicates the operating mode of the intersection controller (e.g., fixed-time, actuated, flash amber signal or any other controller),

[2] This is dependent on the speed limit and whether the signal group contains a turn or straight-ahead movements.

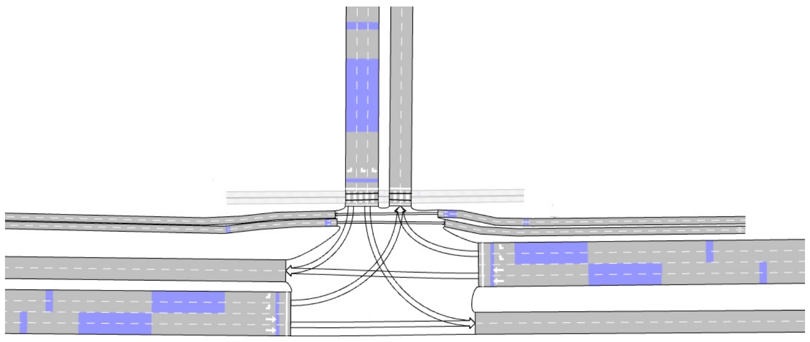


Figure 2-3: Location of stopline, queue, and arrival loop detectors (indicated by blue boxes) at signalised intersection. Movements are indicated by the white arrows on the intersection area.

- the signal timings. This includes the current signal states accompanied by its minimum, likely and maximum end time. It also includes the timings of the next signals. These vary from the next three signals to the next cycle of phases depending on the type of controller.

2-2-2 Connected Traffic System

As mentioned, the traffic infrastructure is already well developed in the Netherlands. For example, most of the current traffic signal controllers are already capable of Infrastructure-to-Infrastructure (I2I) communication of real-time data using the V-Log protocol.
Nevertheless, the Dutch government is transforming the current traffic light controllers to intelligent traffic light controllers (iTLC)s, where Infrastructure-to-Vehicle (I2V) and Vehicle-to-Infrastructure (V2I) communication is added [10]. In this new connected traffic system, the intelligent Traffic Light Controller (iTLC) is connected to a cloud service via a telecommunication network and real-time data is exchanged on a national standardised platform. Next to that, the hardware and software of iTLCs are decoupled, making software as a service possible. Furthermore, all technology and communication protocols are standardised by international ETSI standards and as a result identical for all suppliers of iTLCs, smartphones and cars. This enables the use of more advanced traffic control methods, where communication is needed between intersections, and in-vehicle information services such as Green Light Optimised Speed Advice (GLOSA). This is reflected in the iTLC's supported use cases, namely traffic flow optimisation via traffic signal control software and extra data, in-vehicle information services (e.g., GLOSA), and prioritisation of special purpose groups such as public transport and emergency vehicles.

The amount of active iTLCs in the Netherlands is currently around 550 of about 1000 ordered [21]. However, another 1000 iTLCs are planned to be ordered [10]. In the Netherlands, there are currently a total of around 5000 iTLCs. The location of the iTLCs can be seen in Figure 2-4, where most of the controllers are located on provincial arterials.

It is expected that in the next few years, the availability of traffic data such as Floating Car Data (FCD), which consists of GPS data obtained from smartphones, navigation systems, and other GPS systems, will increase. This will provide opportunities to improve the performance of traffic state detection and control greatly, even with low market penetration rates of connected devices [22, 23]. It is thus desirable to design a controller that is compatible or extendable to deal with these extra measurements and services.

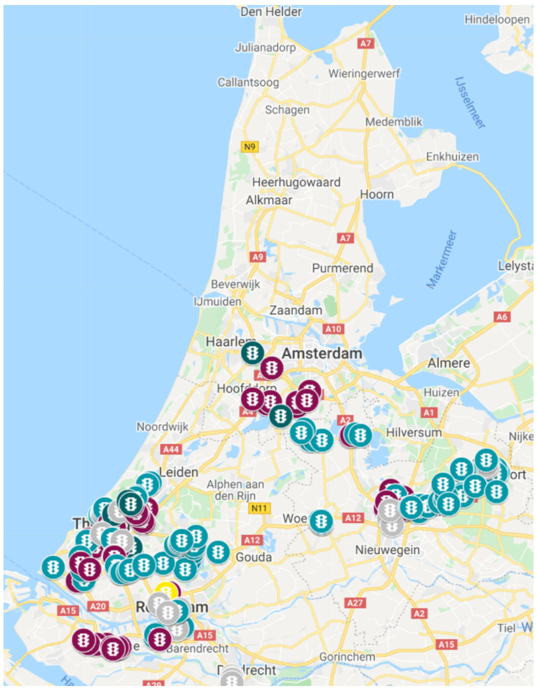


Figure 2-4: Location of active iTLCs on 15th of February 2022 in North- and South-Holland, the Netherlands. Source [21].

2-3 Traffic Dynamics

In this section, a brief outline is given of the basic traffic dynamics in urban signalised links to provide the necessary background for discussing the TSC methods. In subsection 2-3-1, the basic microscopic and macroscopic traffic variables are described and the fundamental relationship between traffic flow and density is explained. The different traffic conditions that can occur in signalised links are discussed in subsection 2-3-2. Lastly, in subsection 2-3-3, the queue dynamics at signalised intersections are explained together with the queue discharge rate.

2-3-1 Fundamental Relationship

Traffic dynamics can be described on a microscopic and macroscopic level. On the microscopic level, vehicles are described individually by basic variables such as speed, headway and space headway. The space headway is defined as the distance between a vehicle and its leader, including the length of the vehicle, i.e., the distance from the rear bumper of the leading vehicle to the rear bumper of the following vehicle. The headway is defined as the time it takes for the vehicle's front bumper to reach the position of its leader's rear bumper [13]. Variables on a macroscopic level do not describe individual vehicles, but describe aggregated variables, such as flow, density and average speed per road section. Naturally, there exists a relationship between each of the macroscopic and microscopic variables, which is summarised in Table 2-1.

Table 2-1: Microscopic and macroscopic variables and their relationship (Bar above a variable indicates mean).

Microscopic	Symbol	Unit	Macroscopic	Symbol	Unit	Relation
Headway	h	s	Flow	μ	veh/h	$\mu = \frac{3600}{\bar{h}}$
Spacing	s	m	Density	k	veh/km	$k = \frac{1000}{\bar{s}}$
Speed	v	m/s	Average speed	u	km/h	$u = 3.6\bar{v}$

There also exists a relationship between the three macroscopic variables, namely, that the flow, μ , is proportional to both the density k and the speed u: $\mu = ku$. This relationship is called the fundamental relationship and is depicted in the fundamental diagram of Figure 2-5.

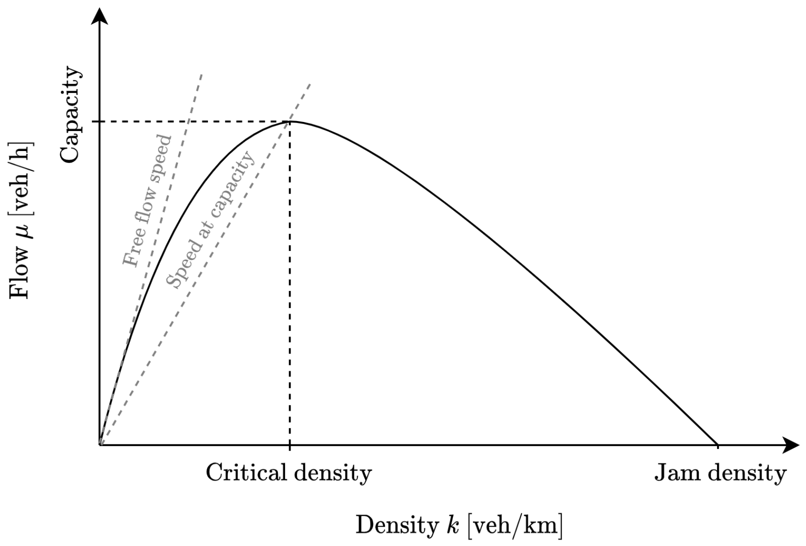


Figure 2-5: Fundamental diagram.

From this figure, it can be derived that vehicles travel at free-flow speed at low densities. When the density increases, the speed of the vehicles starts to decrease, increasing the density of the road segment. When the density reaches the critical density, the road segment reaches its saturation capacity, which is the maximum flow rate that can be maintained on that road segment. When the density further increases beyond the critical density, congestion appears and the flow decreases with further increasing density. The jam density is equal to the maximum number of vehicles that can be stored in a road segment. When the jam density is reached, the speed is zero and the maximum holding capacity of the road segment is attained.

Since real traffic data is scattered, different approaches have been taken to represent the fundamental diagram. Over the years, many mathematical expressions have have been proposed for the fundamental diagram. For example, there exist triangular shapes or shapes called inverse lambda, which also take the capacity drop after a bottleneck into account [13].

2-3-2 Traffic Conditions

In an urban signalised link, three traffic conditions can be defined [24]. The traffic condition in which a queue can still be emptied during one green signal is called the understurated condition. Here, a coupling exists between upstream and downstream intersections as an increase in the outflow of an upstream intersection can lead to a change in the outflow at a downstream intersection [2]. This can result, depending on the traffic signal controller employed, in the formation of *green waves*, where vehicles can progress through successive intersections without stopping. In the saturated condition, queues cannot be fully emptied during a green signal resulting in a residual queue, but queues do not propagate upstream. Thus, in the saturated condition, the outflow of both upstream and downstream intersections is the same. In oversaturated conditions, queues do propagate upstream and potentially spill back and block the exiting vehicle at the upstream intersection. Here, a coupling between downstream and upstream intersections exists as an increase in the outflow of a downstream intersection leads to a change in the outflow at an upstream intersection.

Coordination of Traffic Signals

Coordination of traffic signals between intersections is a complex task. Traffic flows show nonlinear and stochastic dynamics, which are coupled between intersections in an urban network. Additionally, driving behaviour and traffic demand changes over time and depends on a multitude of factors (e.g., weather, time, behaviour of other drivers, behaviour of the traffic signal controller etc.).
Considering the above definition of traffic conditions, coordination between intersections is most necessary in undersaturated and oversaturated conditions. However, when the traffic demand is very low (e.g., at night), not much coordination is needed. When intersections are located close to each other (within 800 meters), the signal timings of an upstream intersection have a large influence on the downstream traffic state and vice versa [25]. This originates from the fact that vehicles leaving an intersection travel in a platoon that slowly disperses as it flows further downstream [13]. Lastly, the more flexible the control signals are to the traffic demand, for example in acyclic operation, the more difficult coordination of these signals between intersection becomes, because of the increased amount of decision variables in acyclic operation.

2-3-3 Queue Dynamics

The traffic flow dynamics at signalised intersections are largely described by the dynamics of its queues. Queuing dynamics and vehicle trajectories can be displayed in space-time and flow-density diagrams displayed in Figure 2-6. In this figure the different queuing conditions for vehicles are displayed by zones A to C, with the stopline indicated by the green-red-green

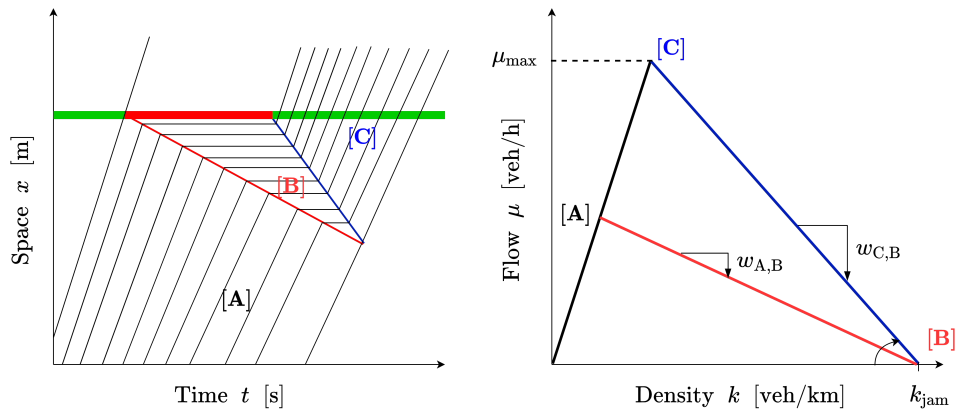


Figure 2-6: Shockwave dynamics at signalised intersection. Space-time diagram (left) and flow-density diagram (right).

bar, of which the colour represents the colour of the traffic signal. In zone A, the vehicles approach the intersection at free-flow speed and in zone B, vehicles are standing still in the queue, whereas in zone C the vehicles are released from the queue and accelerating to free-flow speed. Intuitively, the formation of queues due to a red traffic signal, results in a growing queue tail in the upstream direction. These temporal and spatial queuing dynamics can also be described by using shockwave theory [26]. The stopping shock wave at tail of the queue is formed by the interaction between the arriving traffic and the stopped traffic. It begins from the stopline at the start of the red signal and moves upstream with speed $w_{A,B}$. During queue dissipation, when a green signal is shown and when vehicles discharge until the saturation flow rate $\mu_{\max}$, a starting shockwave is formed at the head of the queue that moves in upstream direction with speed $w_{B,C}$. The speed of the shockwaves is determined from the fundamental relationship and is equal to the difference in flow rate between the two zones divided by the difference in density of the corresponding zones.

$$w_{AB} = \frac{\mu_A - \mu_B}{k_A - k_B} \tag{2-1}$$

It should be noted that in Figure 2-6, the trajectories of the vehicles are simplified. In reality, vehicle trajectories have more variation in the acceleration and deceleration due to the different driving behaviours and vehicles. For example, in Figure 2-6, the deceleration of the vehicles from free-flow speed to stand still in a queue is immediate. In reality, a smoother, slower deceleration occurs.

The departure process after a signal signal switches from red to green at a signalised intersection is displayed in Figure 2-7, where the headway measured at the stopline of an intersection is depicted per vehicle position in the queue. The headway of the first vehicles is longer, because of the driver's reaction and the acceleration limits of the vehicle. It can be seen that the headway becomes constant after the fifth vehicle crosses the stopline [27]. This constant headway is the saturation headway, which is the average headway that can be achieved by a saturated stable moving queue of vehicles passing over the stopline [27]. When the signal turns to amber, some vehicles still cross the stopline, but not the whole amber time is effectively used. There is still some amber time that could be used to cross the intersection, but drivers stop in front of the stopline during the amber signal if they can safely stop. Thus, the

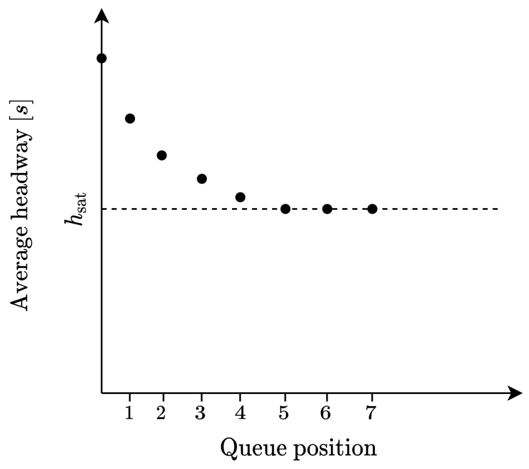


Figure 2-7: Average headway when signal switch from red to green in saturated traffic condition.

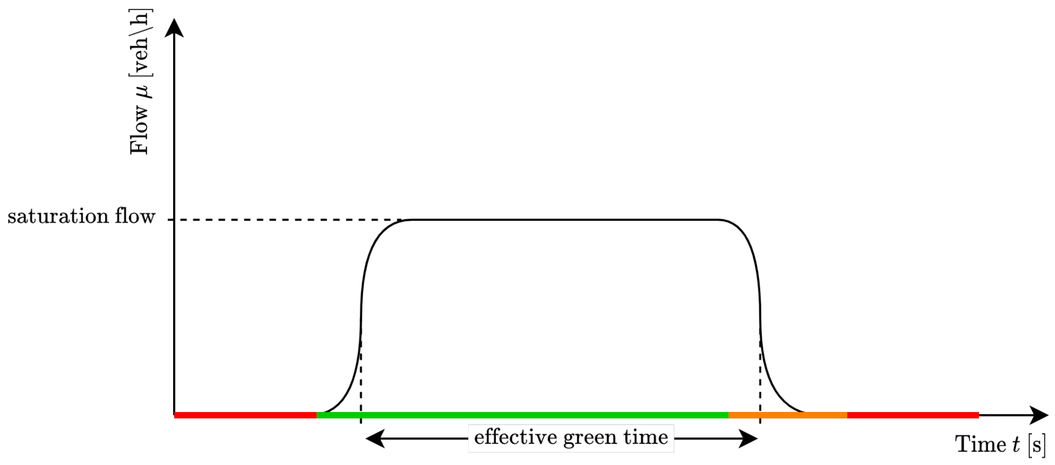


Figure 2-8: Queue discharge in saturated traffic condition.

combination of the longer headways at the start of the green signal and the partially used amber signal constitutes to the ineffective use of the green signal. The effective green time accounts for the loss at the beginning of green and the gain during amber and is illustrated in Figure 2-8. Again, it should be noted that this is a representation of driving behaviour and vehicles. In reality, variation exists within drivers, vehicles, weather conditions, etc.

Chapter 3

Related Work

First, in section 3-1, a taxonomy is provided on the different features of traffic signal controllers in order to provide the positioning of Model Predictive Control (MPC) within other types of traffic signal control methods. Next, a brief introduction to MPC and multi-agent MPC is given in section 3-2. Thereafter, the related work on distributed model-based traffic signal control methods in urban traffic networks is discussed in sections 3-3 and 3-4.
This chapter will only investigate methods that use the current infrastructure and its available measurements covered in subsection 2-2-1. In section 3-5, the problem statement will combine the insights learnt from the related work with the problem posed within industry. This sets the scope, objective and research questions of the thesis in the problem statement in section 3-5.

3-1 Taxonomy

Traffic signal control methods mainly differ on their type, structure, and the used traffic model. In subsection 3-1-1, the types of controller are divided on how adaptable the determined traffic signals are to the prevailing traffic demand. Next, in subsection 3-1-2, a brief description of traffic flow models is given. The controller structure relates to the level hierarchy and the communication level between controllers in a network of controllers and is discussed in subsection 3-1-3. Finally, the road network topology and road type are described in subsection 3-1-4. These are used to describe the environment for which the controller is designed.

3-1-1 Type of Traffic Signal Controller

As mentioned, the type of controller is categorised based on the flexibility of the traffic signals to the current traffic demand. Three types exist, namely fixed-time, actuated, and predictive traffic signal controllers.

Fixed-time

Fixed-time controllers use a traffic model and optimise the signal timings offline for a specific traffic condition (e.g., high traffic demand) based on historical traffic flows. The more advanced versions select a signal schedule from a set of predetermined schedules for a particular time (e.g. time-of-day or weekend/weekdays). Fixed-time traffic controllers operate in a cyclic way, where the cycle time, split and order of the phases are the control inputs. Once the optimisation is done offline, the controller is deployed on the street and the signal timings repeat the same signal sequence every cycle. The disadvantage of the fixed-time controller is that it uses historical data for the determination of the signal schedules. Since traffic demand changes over time, the delay performance of these controllers degrades by 3-5% each year [28]. Therefore, fixed-time traffic controllers need to be calibrated on-site and are employed at intersections where the traffic demand is predictable [13]. Additionally, they are not adaptive to an unexpected short-term traffic demand change [25]. It is, however, relatively easy to coordinate the signal timings of intersections as the optimisation can be executed offline for a specific scenario and few control input variables need to be determined (i.e. phase order, split, offset and cycle time).
The implementation of Green Light Optimised Speed Advice (GLOSA) works well in combination with fixed-time controllers as the signal timings do not change. Therefore, these timings can be communicated reliably to the road users in advance. Some well-known fixed-time controllers in industry are SIGSET [29], MAXBAND, MUTLIBAND [30] and TRANSYT [31].

Actuated

The traffic signal timings of actuated controllers react in real-time to vehicle detections. Actuated controllers can be configured in such a way that the order of the signal groups is still fixed, but the duration of the active green signal group can vary between lower and upper bounds, based on vehicle detections. In the second type of configuration, the order of these green signals can be changed to give a nonconflicting signal group with the most traffic demand a green signal as soon as possible. The active green signal group is extended or terminated to clear the queue based on the speed limit, clearance times, minimum headway, and the distance of the detector to the stopline of the signal group. Actuated controllers are placed at intersections with unpredictable traffic demand and/or in (under)saturated traffic conditions. Actuated controllers require a small computation time for determining the signal timings, so they can operate with a small control time step of around 0.1-0.2 seconds. This small control time step results in more efficient operation of the signals due to the smaller discretisation of time. This waists less green time and the controller can react faster to vehicle detections.
The main disadvantage of actuated control is that it determines its signals in a very responsive manner by basing its signal timings directly on detections. Therefore, it is shortsighted as it does not take into account the longer-term effects of its determined signals. On top of that, most actuated controllers only take vehicle detections close to the intersection into account (see detectors in Figure 2-3). Actuated controllers do not perform well in (over)saturated traffic conditions, because the extension of the active green signal group always reaches the maximum green time and the controller then operates on a fixed schedule. Road managers

then decide to switch to a predetermined fixed-time plan tailored for high traffic demand. Coordination between traffic actuated controllers is achieved such that a specific coordinated signal group can be scheduled in time and other noncritical (non-coordinated) signal groups can be *forced-off* or terminated in time. The length of the non-coordinated signal group is limited to the duration of the predetermined split such that the coordinated signal group can be given a green signal in time.

Predictive

Predictive traffic signal controllers determine their signal timing online based upon measurements from multiple intersections and use a traffic model that predicts the future evolution of the traffic flow. Predictive controller can be designed to operate in a cyclic way where the cycle time, split and offset are optimised every two to ten minutes based on real-time detector data and its prediction model. The most well-known predictive controllers in industry are SCOOT [32], SCATS [33, 34], UTOPIA/SPOT [35] and TUC [36, 37]. Predictive controllers can also operate in an acyclic way, where the signal timings are optimised every few seconds. By basing the signal timings on predictions from a traffic model, the controller can determine the signals that anticipate the future traffic flow. Therefore, predictive controllers produce signals that take into account longer-term effects, which resolves the shortsighted control of the actuated approach. The disadvantages of predictive controllers are the larger computation times required, the complex implementation code that needs to be developed, and, in the case of more advanced control methods, road managers that are unfamiliar with this type of traffic control [38].

3-1-2 Traffic Flow Models

Traffic models can be divided according to their level of detail in which they represent the traffic system [39, 40]. The most detailed level is described by microscopic traffic models, which specify the trajectories of each individual vehicle and their interactions with other vehicles, both in the longitudinal and in the lateral direction. As mentioned, microscopic models deal with variables like headway, spacing and speed. On the other hand, macroscopic models do not describe individual vehicles, but describe aggregated variables, such as flow, density and average speed, per road section. In mesoscopic models, the characteristics of both levels of description are combined. They describe the average speed microscopically of small individual vehicle groups and vehicle flow in aggregate terms such as in probability distributions. Queue models also fit into this group of mesoscopic models. Microscopic models are mostly used to control vehicle trajectories for use in lane-changing or car-following controllers, whereas macroscopic and mesoscopic models are used for traffic control.
For predictive traffic controllers that use optimisation to determine the traffic signals, a consideration needs to be made between the level of detail the model provides and the computational complexity of the optimisation method [38].

3-1-3 Control Structure

The control structure describes the way in which one or more controllers are organised to control a traffic network. This affects the authority and communication relation between controllers and their access to detector measurements and the control of traffic signals. The different structures can be categorised under a single-agent control structure, where only one controller is used, and multi-agent control structure, where multiple controllers are deployed in a network [17].

Single-agent Control Structure

A centralised control structure contains only one controller that is able to determine all traffic signals in the network and has access to all detectors. This is illustrated in Figure 3-1, where the control agent represents the traffic signal controller, control actions the signal timings and measurements the detector data. The other parts of this MPC block scheme will be explained in section 3-2.

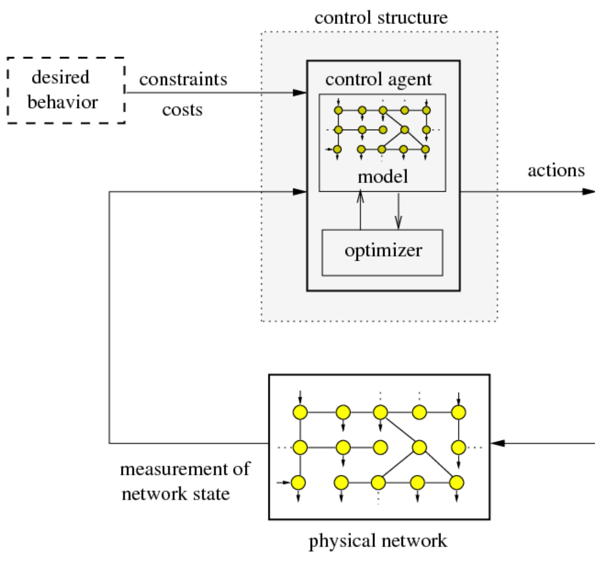


Figure 3-1: Centralised control structure. Source [17].

Centralised control structures yield the best performance, since all information of the network is included in the model, but it has several major disadvantages [17]. The centralised control structure is not easily scalable, because each time a new intersection needs to be added to the controller, the whole centralised structure needs to be adapted and the traffic model has to be replaced with an extended version, accompanying the extra intersection. Additionally, the computation time of the centralised structure increases with the number of intersections. Furthermore, the centralised structure also has reliability issues. If an error or failure occurs in the traffic control system, the operation of traffic control within the network stops. Other disadvantages involve unavailable control access and detector measurements, due to legal or commercial constraints [41].

Multi-agent Control Structure

Dividing the problem of controlling traffic signals in a network into multiple sub-problems can alleviate the disadvantages related to the centralised control structure [17]. Such a division lowers the computational burden as each local controller solves a smaller sub-problem separately. This multi-agent structure is also beneficial if there is restricted control access to traffic signals or unavailable detector measurements in some parts of the network, since local controllers only require measurements from their part of the network. This type of structure also increases reliability, because if one controller fails, the other ones are still operational. On the other hand, multi-agent structures have a lower performance than the centralised structure, because they do not contain all information of the network into their local model. There exist three multi-agent control structures, namely, decentralised, hierarchical, and distributed.

In the decentralised control structure, depicted in Figure 3-2 (without the dashed arrows), each controller only has access to its own traffic signals and detector measurements in their particular part of the network and they do not communicate or coordinate with other controllers. In a distributed structure, also shown in Figure 3-2 (including the dashed arrows), the local controllers are able to communicate and/or coordinate with each other. This naturally increases the computational and communication requirements, but it also increases the overall network performance. Additionally, decentralised and distributed control structures are easily scalable. No model needs to be replaced, only extra controllers need to be added with a different configuration from the other controllers due to the configuration of the intersections and links. There is no need to replace the model, only additional controllers with different configurations from other controllers are required because of the configuration of intersections and links.

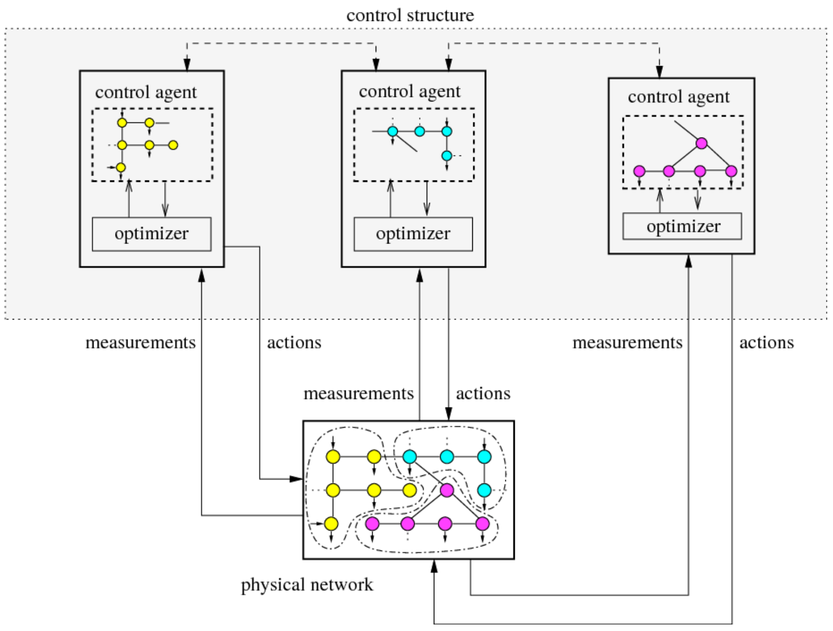


Figure 3-2: Distributed control structure with communication (dashed arrow) and decentralised control structure (without dashed arrow). Source [17].

In the hierarchical structure, shown in Figure 3-3, a higher level controller has authority over the lower level controllers, in a manner that they can inform or coordinate the lower lever controllers [17]. Specifically a higher level controller typically determines the constraints or reference points for the lower level controllers, which are typically organised in a decentralised or distributed manner. A hierarchical structure offers the possibility to make a trade-off between the high computational requirements of centralised structures and overall network performance. It should be noted that the communication and traffic model design is even more complex in a hierarchical structure than in the distributed or centralised control structures.

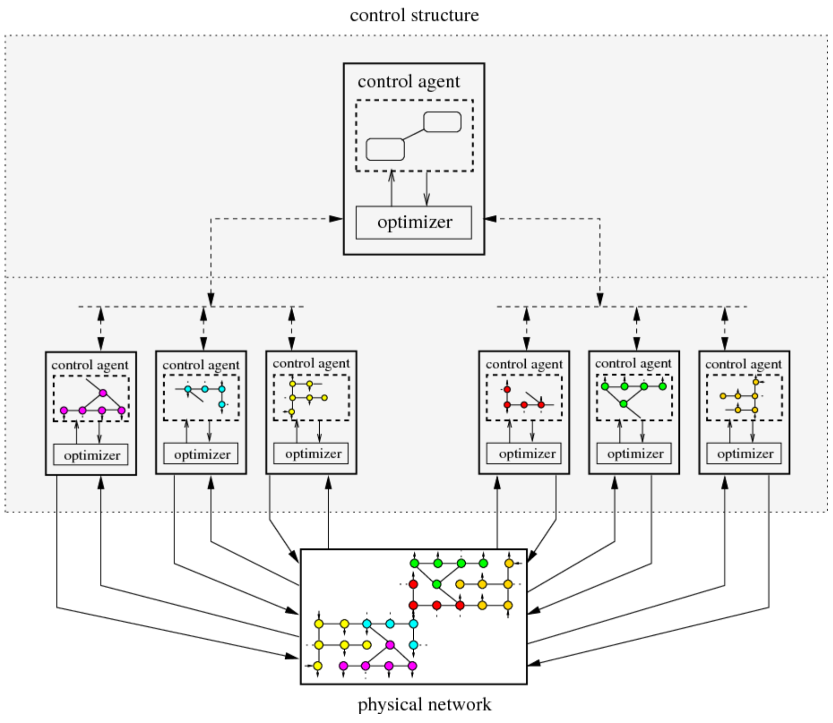


Figure 3-3: Hierarchical control structure. Source [17].

3-1-4 Network Topology and Road Type

Although traffic signal control is only applicable to signalised urban and arterial roads, for completeness, the road type is also included. There also exist methods which incorporate both a highway and traffic signal control methods into one integrated controller. The most significant topology factor is the network for which the control method is designed for. The network topology can range from an isolated single intersection to a complete network. An arterial network consists of multiple successive intersections on a main road, where the demand is higher, to which minor roads with lower traffic demand are connected. A grid network consists of intersections connected in a grid topology with equal or variable distances between the intersections. Naturally, the most realistic network topology is a real road network. Methods can be tested in a simulated (real) network or in a real-world case study.

3-2 Model Predictive Control

In this section, a brief introduction to MPC and its advantages relating to Traffic Signal Control (TSC) is given in subsection 3-2-1. Thereafter, in subsection 3-2-2, a brief introduction to multi-agent MPC is described. This will set up the categorisation used for the reviewed methods in section 3-3 and section 3-4.

3-2-1 Introduction to MPC

MPC is a model-based predictive control method that uses a prediction model subject to constraints and optimisation to minimise a certain objective. This is done to determine a sequence of control actions over a prediction horizon. Only the first action in this sequence is applied to the system. At the next time step, the optimisation is done again, but the horizon is shifted one time step in the future, this is called the receding horizon policy. This policy is illustrated in Figure 3-4, where the objective function minimises the error between the reference and the predicted output. A schematic view of the an MPC controller in a control block scheme is depicted in Figure 3-5. With regards to traffic control, MPC has several advantages [17, 15]:

- A user-defined objective function can be defined. Herein, multiple objectives can be weighted against each other, such as minimising the delay and the number of stops.
- MPC can handle hard constraints on control inputs, states and outputs. This is useful for TSC as safety constraints need to be satisfied.
- The receding horizon procedure gives MPC built-in robustness properties: MPC can quickly react to disturbances and mismatches between the model and the real system, since at each time step the initial conditions of the model are set equal to the current measurements of the system. Additionally, MPC also makes the best current control action while still keeping in mind the longer term effects.
- The predictive model can easily be replaced or adapted and few parameters need to be tuned [14].

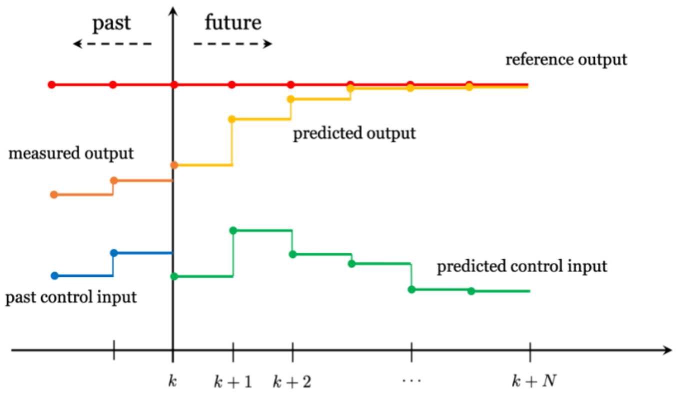


Figure 3-4: Receding horizon policy where k indicated the current time step. Source [42].

The disadvantages are that a prediction model has to be specified, because sometimes accurate models are not available [42]. Additionally, an efficient and tractable optimisation method has to be available. Therefore, a trade-off between computational complexity and modelling accuracy needs to be made [15]. Model predictive control methods mainly differ in their control structure, coordination, traffic flow model, and solution methods for the optimisation problem. Therefore, the methods will be discussed under these characteristics in the next sections.

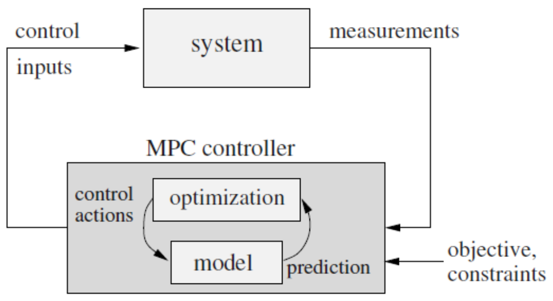


Figure 3-5: Schematic view of model predictive control. Source [17].

3-2-2 Multi-agent MPC

In Figure 3-5, a centralised control structure is shown, but there also exist methods for multi-agent MPC. The methods in multi-agent MPC can be divided into two categories, namely top-down and bottom-up. The first uses a model of the whole traffic network and then decomposes this central problem into smaller sub-problems. The bottom-up approach starts with a model of the sub-problem (i.e., starting with a decentralised controller) and adds coordination separately.
Distributed MPC methods can also differ in the coupling source between different controllers in the network (which coupling makes the overall system nonseparable: inputs, outputs, states, objective or constraints), local controller information (strictly local or partially global), iterative or non-iterative computation, cooperative or non-cooperative behaviour, serial or parallel communication, synchronous or asynchronous timing of communication. The most important theoretical feature is whether the methods guarantee optimality. If the method is optimal, then the distributed method yields the same result and performance as the centralised controller, typically after several iterations or satisfying several theoretical conditions [41].

In the next sections, only methods using decentralised and distributed control structures are covered, since centralised structures have several inherent drawbacks regarding scalability, reliability, computational complexity, and control access (see subsection 3-1-3). Hierarchical control structures are also less scalable and not preferred by Yunex, since this structure makes the coordination layer more complex with more communication and two different models required (see subsection 3-1-3). Ease of deployment and robustness to communication shortages are important goals for Yunex in their development of traffic signal control software.

3-3 Top-down Networked Control

In this section, the multi-agent or networked MPC methods using a top-down approach are discussed. As already mentioned, these methods will be described in terms of the used traffic model in subsection 3-3-1, control structure in subsection 3-3-2 and finally the used optimisation methods in subsection 3-3-3.

3-3-1 Traffic Models

Many traffic models for use in optimal model-based traffic signal controllers have been developed over the years. The most well-known models are the stage-and-forward model [43], S-model [44] and the cell transmission model [45]. Next to that, models from hybrid systems theory are also used. These are all macroscopic traffic models, since they model the whole traffic network.

Stage-and-forward Model

The stage-and-forward model [46, 43] describes the queue discharge flow of an intersection as a continuous flow and averages this flow over each cycle. This transforms the discontinuous traffic flow at an intersection into a continuous one, where the switches of the traffic signals are not taken into account. Thus, queue spillback and queue oscillations due to signal switches cannot be described and control decisions can only be implemented every cycle, which varies from 40 to 120 seconds. Due to the assumption of constant traffic demand, this model performs well only in (over)saturated traffic conditions. The control inputs are the green splits within a fixed cycle time [46].

S-model

To overcome the disadvantages posed by the stage-and-forward model, S. Lin proposed the S-model [44], which is able to describe the traffic flow in all conditions. Here, the queue discharge flow rate is taken as the minimum of the number of queued vehicles, the saturation capacity, and the holding capacity of the downstream links. The S-model also describes spatial extent of queues by multiplying the number of vehicles in a queue by the average vehicle length. Queue shockwaves are not described by the S-model and traffic dynamics are still averaged within each cycle, so queue spillback and oscillations within the cycle still cannot be described. The S-model is part of a class of discrete-time urban traffic models that are sampled spatially into road segments equal to the link length between intersections and temporally into sampling times equal to the cycle time.

Cell Transmission Model

In the Cell Transmission Model (CTM), a road segment is spatially sampled into cells of equal length, which is mostly set to the travel length of a vehicle at free-flow speed. The CTM [45] is a discrete time approximation of the LWR model [47], where it is assumed that the flow is only a function of the density in the fundamental diagram. The dynamics within each cell only depend on the previous cell. Each cell receives a number of vehicles from the previous cell and sends the number of vehicles to the next cell. The sending and receiving number of vehicles are determined by taking into account the vehicles going into the cell, the holding capacity of the cell, the maximum flow rate and the backward moving shockwave. The CTM is thus able to describe the traffic flow in all conditions and also describes the spatial and temporal dynamics of queues, i.e., the shockwave dynamics. To model an intersection, merge, diverge and signalised cells are defined. Several modifications to the CTM exits, such as the link transmission model [2] and the node transmission model [48], which is used to model freeway traffic. Most of the MPC methods using the CTM average the dynamics per cycle to transform the model into a model with real-valued control inputs and states. Then, flow rates are send between cells, instead of the number of vehicles. The CTM can also be reformulated into a linear model with discrete variables for the signal inputs and real-valued states [49].

Hybrid Systems Modelling

Yet another way to model urban traffic networks is to use models from hybrid systems theory, which studies the behaviour of systems, where the dynamics can be characterised by a hybrid of continuous or discrete time and a time-driven or event-driven evolution of the system states [50]. The traffic flow at signalised intersections can be conveniently modelled as a hybrid system, where the traffic flow is described by means of a time-driven model with continuous states and the traffic signal dynamics are represented by a discrete-event model and discrete states.

Recently, Di Liu et al. [51] formulated a multi-intersection traffic flow model as a switched system, which allows the controller to operate in an acyclic way. Others used the hybrid petri net formulation, where the vehicle flow is represented by a time-driven model and the influence of the traffic signals as an event-driven model. Hybrid petri nets model the interplay between these dynamics as an untimed, logical representation. This can be extended to a timed petri net by requiring that the transition between the logical elements of the model has to be executed within a certain time interval [50].
However, real-world implementation and performance of petri net models in traffic control has been rated average [52]. Petri nets are more used for modelling, analysis, and simulation of traffic systems and controllers. It is used, for example, to analyse behavioural properties, such as liveliness, reversibility, reachability, and safeness, which assist in analysing and ensuring safe and reliable traffic control [52].

3-3-2 Control Structure

In top-down networked control, MPC is first designed as a centralised control method where the model describes the traffic flow dynamics of the whole traffic network. From this point of view, centralised MPC can achieve optimal coordination on a network scale. This, however, comes at a cost of longer computation times. Therefore, methods are developed that use a distributed structure to make the traffic controller more scalable and less computationally demanding.
The methods in top-down networked TSC all operate in a cyclic way, which is not preferred as road users in the Netherlands are already used to the acyclic and flexible behaviour of the traffic signal controller. Additionally, Yunex already has solutions that use cyclic operation. Therefore, only distributed methods with potential for acyclic operation will be discussed.

In his PhD thesis, Negenborn developed an augmented Lagrangian-based distributed MPC method where the overall network control problem for energy networks is modelled [53]. Since the method can be applied to TSC, it is explained using traffic terminology. The overall problem in Negenborn's thesis is decomposed per intersection, where each local intersection problem minimises its own delay and has interconnecting constraints with its neighbours, i.e.,

the output flow of a neighbour must be equal to the input flow of the controlled intersection. Then, the controller performs an iterative Lagrangian optimisation procedure, where the local controllers have to agree on the value of the interconnecting variables (output and input flow). More recently, Mendes [54] proposed a faster optimisation procedure using a binary search algorithm for Negenborn's method. This enabled it to be applied to systems with binary control variables, in this case it could be used to model the traffic signals for TSC in acylic operation. Opposite to the Lagrangian-based optimisation procedure, convergence to the optimal solution is not guaranteed by Mendes' method. Next to that, Lagrangian-based distributed optimisation requires a lot of iterations and has slow convergence properties [54]. Additionally, there exist problems with non-convergence of the binary inputs [53].
If this method would be applied to TSC, the dynamics linking the sub-problems, i.e. link dynamics between two intersections, become nonlinear, so the central problem becomes fairly complicated to decouple. Specifically, the nonlinear link equation describing the arrivals at a particular time step is based on the vehicles entering the link at that time step minus the link delay. This link delay depends on the queue length, which in itself is again dependent on the arriving and entering vehicles. This link equation has to be linear to decouple the dynamics. One option is to make the link delay constant (based on a constant queue length), another is to define different constant delays based on different constant queue lengths and formulate a piecewise affine formulation of this equation [44]. Unfortunately, an extra binary variable is introduced per specified delay in the piecewise affine formulation, which increases the solution time.

The cell-transmission model [49, 45] can be directly formulated into a mixed-integer linear programming problem, where the traffic signals are modelled as binary variables. This allows for acyclic operation of the controller. To this end, Mehrabipour and Hajbaie [55], developed a real-time distributed control method based on the cell-transmission model, where the resulting optimisation problem can be solved within two seconds. This is done by decomposition of the central Mixed-Integer Linear Programming (MILP), where most cells in the middle of a link between intersections are omitted and replaced by dummy cells. This decomposition results in a decentralised MILP problem per intersection. The central CTM model is used for simulation in order to generate and pass critical information between intersections to account for the omitted cells. In particular, the CTM simulation will generate information on the number of vehicles passed between the omitted cells. This information will be passed to the dummy cells of the decentralised problem to be used in the next optimisation step. Results indicated the solutions found by the distributed approach where at most 1% different from the solutions of the central problem and at most 5.7% below the theoretical upper bound obtained by a relaxed MILP problem, where optimisation is done using real-valued inputs which are then rounded to obtain discrete variables.

3-3-3 Solution Methods

When the models used are convex and quadratic, quadratic programming can be used. When the models are nonlinear non-convex in nature, optimisation is done via multi-start local optimisation methods such as sequential quadratic programming, interior point methods, and evolutionary algorithms such as genetic algorithm [15, 56]. Solution methods of integer optimisation problems will be discussed in **??**

3-4 Bottom-up Networked Control

In this section, the MPC methods using a bottom-up approach are discussed. These methods start with the formulation of a decentralised controller and add communication separately in order to obtain coordination. Therefore, only queue and vehicle arrival models are used, which are discussed in subsection 3-4-1. Methods using decentralised and distributed structures are covered in subsection 3-4-2

3-4-1 Traffic Models

In section 3-4-1, queue models are discussed, which model the effect of the traffic signals, departures, and arrivals on the evolution of the queue. Thereafter, in section 3-4-1, the arrival models are reviewed, which model the delay and platoon dispersion within a link between two intersections.

Queue Models

Intersection queue models can be divided into two types, namely, vertical and horizontal queue models. In the vertical queue model, arriving vehicles travel at free-flow speed until the stopline, where they stop instantaneously and are added to a vertical queue. Vertical queue models thus occupy no horizontal space. This is why they are often called point queue models. A visual representation of vertical and horizontal queues can be seen in Figure 3-6. Vertical queues can also be modelled in such a way that vehicles travel at free-flow speed until the end of the queue then stop instantaneously and are added to the vertical queue. In this case, the arrival prediction model needs to take into account the current queue length and its dynamics. Vertical queue models thus cannot describe the shockwave dynamics of queues. Horizontal queue models describe the spatio-temporal dynamics of queues, including the schockwave dynamics, and have larger computational requirements [57]. Therefore, almost all bottom-up MPC methods use a vertical queuing model instead of horizontal, which is mostly relevant for modelling queue spillback and blocking effects (when a queue exceeds the link length and blocks other traffic) in oversaturated conditions and to model the nonlinear discharge dynamics of queues [57], described in subsection 2-3-3.

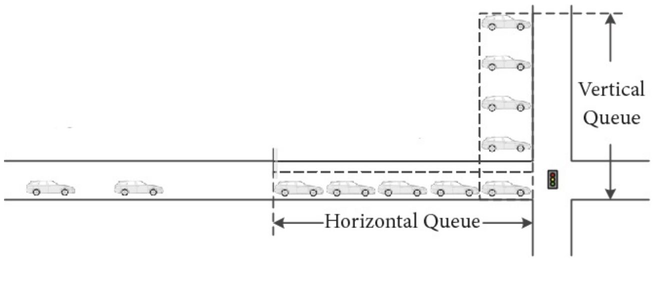


Figure 3-6: Vertical and horizontal queue models. Source[58].

Arrival Prediction Models

Since bottom-up networked control methods start with a decentralised model, prediction models are used to predict the arrivals at the intersection. This can be done using simple rules such as dividing the free-flow speed by the distance between a far away arrival detector and the stopline detector or by taking into account the shockwave speeds at the tail and head of the queue. The latter has been developed in a recent paper by Chen and Sun [59]. Some methods additionally use detector measurements and the signal timings of each upstream intersection. These simple rules are chosen to make a trade-off between modelling accuracy and computation time, since the models are included in the (distributed) optimisation methods.

Complex analytical models also exist in literature. The three most well-known are Lighthill & Witham's fluid dynamic traffic model [47], diffusion model of Pacey [60] and Robertson's platoon dispersion model [26]. These models all try to describe the progression of vehicle platoons in links. An illustration of a platoon progressing through a link can be seen in Figure 3-7. Most of these complex analytical models are used in traffic controllers that optimise the signals timings every 5 to 10 minutes. Robertson's platoon model is the most recent model and describes the progression of a platoon through a link by an empirical recursive relationship. These models produce very accurate predictions, but a lot of parameters need to be calibrated.

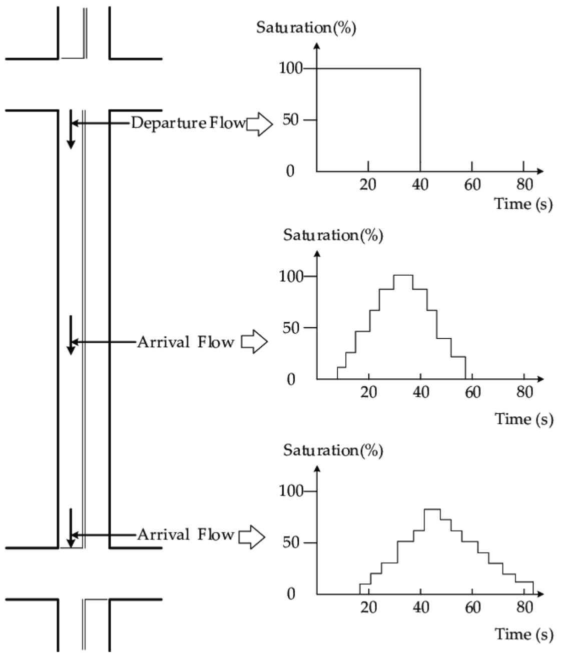


Figure 3-7: Platoon dispersion in a linke between two intersections in saturated traffic condition. Source [61].

Recently, more complex arrival prediction models have been developed [62, 61]. These models range from stochastic arrival models or learnt prediction models using machine or deep learning. To this end, Helmy [63], Van Senden [64] and Glastra [62], former Siemens Mobility thesis interns, developed a short-term data-driven traffic flow progression model to predict arrivals at the arrival loop detector of the controlled intersection. By using a data-driven model, no explicit assumptions have to be made on the link topography, driving behaviour, number of exiting vehicles between links, etc. The method employs a type of recurrent neural network, namely an Long Short-Term Memory (LSTM) network. This type of neural network is often used in time-series forecasting applications, since conventional recurrent neural networks have problems with exploding or vanishing gradients during back-propagation training on data with long-term dependencies [65]. Therefore, LSTM networks are designed with different gates to decide which dependencies in the time series to keep and which to forget or update. The LSTM method showed more accurate prediction performance, measured in terms of the normalised root mean square error than Robertson's model [63].

3-4-2 Control Structure

As mentioned, in the bottom-up networked control, the decentralised controller is first developed and then coordination is added separately. Therefore, in section 3-4-2, the decentralised and distributed controllers are discussed and in the following section 3-4-2 only the coordination part of those controllers is covered.

Decentralized Structure

Most of the decentralised methods model the intersection with binary or integer control inputs for the traffic signal colours and integer values for the queue, departure and arriving vehicle variables. However, there exists some variation in terms of which switches are allowed to reduce the number of variables that has to be taken into account during optimisation. The controllers in OPAC [66] and PRODYN [67] both decide to either extend the current phase or to go to the next phase. The cycle lengths and the sequences of the different phases remain the same. These are the same decision variables used in actuated traffic signal control. The only difference here is that a prediction model is used to anticipate arrivals more proactively and that the optimal decision is made over a prediction horizon. COP [68] takes a different approach by employing phase skipping. In this approach, the phases are still arranged in a fixed sequence, but by allowing phase skipping, any phase sequence can be obtained. Van Katwijk [57] took another approach, where a stage does not represent a single group but a block of signal groups corresponding to the Dutch RWSC actuated control configuration. Chen [59] uses a comparable NEMA [69] approach, which is used as a standard in the US.

Haddad et al. [70] formulated a steady- state control problem in which an isolated intersection is modelled as a discrete-event max-plus system. The optimal switching sequence, which minimises the queue length, is obtained through solving a linear programming problem analytically. For steady state control, it is assumed that the arrival and departure rates are constant and so is the cycle length. De Schutter [71, 72] modelled the evolution of the queue length at an isolated intersection as an extended linear complemantarity system. The authors also derive suboptimal and relaxed problems which can be computed efficiently. Although this formulation allows acyclic operation, constant average arrival and departure rates per phase and continuous queue length were assumed. In general, the above methods by Haddad and De Schutter focus on optimal control under (over)saturated traffic conditions at isolated intersections. The authors do not provide a multi-intersection approach other than proposing a complete decentralised control in traffic networks.

Van Senden [64], a graduate intern at Siemens Mobility, developed a traffic signal controller based on the self-organizing controller proposed by Lämmer [73, 74]. The controller is called DIRECTOR, which stands for Data-driven Intersection and Road Environment Controller for Traffic Optimisation in Real-time. The controller uses detector and signal data from neighbouring intersections as an input to an LSTM neural network, which predicts the vehicle arrivals at the arrival loop of the controlled intersection. These arrivals are added to a vertical queuing model. The objective is to minimise the average delay per vehicle. Therefore, every control time step, the controller gives a green signal to the phase group that has the maximum cumulative predicted delay, taking into account the switching penalties for phase changes. In essence, the controller can be said to be "greedy" in its determination of which signal to schedule as only the phase with the maximum predicted delay is chosen. The controller thus tends to allocate the green signal to the phase that has the longest queue length over the prediction horizon. The controller does not use optimisation, but performs maximisation. Therefore, not all possible signal timings are considered and an extension to a coordination optimisation procedure is limited as the authors of the original paper intended the controller to be solved decentralised. Strictly speaking, DIRECTOR is a distributed controller since it receives information from its neighbours, but it is not designed to be extended to coordinate with its neighbours. Further information about this method can be found in the thesis by van Senden [64] or Glastra [62] and in Appendix A.

Distributed Structure

In the later stages of its development, PRODYN [67] adds communication exchange between intersections to allow for coordination, thus resulting in a distributed structure. This is achieved through the exchange of information regarding pending arrivals between the adjacent controllers. The objective function of the local controller consists of a part that only depends on the local states and an additional part that signifies the contribution of the outputs of the local controller to the network. The variables in the second part are communicated by a supervisor. The procedure is as follows. The current control actions are simulated by a supervisor. This supervisor communicates the arrivals and additional variables to the local controller. Then the local controller computes its optimal control signals and sends its outputs to the supervisor. This is done iteratively until the control actions of the local controller do not change any more.

Van Katwijk [57], proposed a microscopic coordination procedure for traffic signals and a macroscopic coordination procedure for different traffic control instruments, such as ramp metering and variable speed signs. The high-level coordination procedure is the following. Upstream agents communicate their planned outflow to downstream agents, who calculate the costs of the planned outflows on their performance. This cost is then communicated to the upstream agents, who analyse if this cost outweighs their own locally optimised cost. If the downstream cost is smaller then the own cost, the own performance of the upstream agents can only be altered by regulating the inflow. This is done by incorporating the costs of the downstream inflow into the cost of the upstream agents. The above procedure is iterative until no changes in signal timings occur. The local controller developed by van Katwijk was altered to not only incorporate predicted arrivals, but also information from downstream agents. Simulations on an arterial road and a grid network found that the coordination procedure is able to generate "green waves" and adapted to changing volumes and platoon ratios. Unfortunately, the exact microscopic coordination procedure, the addition of the downstream cost into the local controller, and the specific results of the simulations are not specified.

It should be noted that the methods described above were first developed 20 years ago and only the first development of the controllers is shared in publicly available papers. Any further developments were carried out internally within private companies. Therefore, not all other distributed versions of the methods described in section 3-4-2, are discussed.

3-4-3 Solution Methods

In traffic signal control, the decision of which signals to implement can be formulated into a general decision problem. This can be done to represent any integer optimisation problem. The optimisation problem can then be represented by a search tree, where an optimal path of decisions must be found over a prediction horizon. The most important approaches for finding the (optimal) path of decisions in a decision tree are mixed integer (non)-linear programming, constraint logic programming, dynamic programming, or heuristic search techniques [15]. Most of the bottom-up approaches use dynamic programming or mixed integer (non)-linear programming. Heuristic search techniques include simulated annealing, random search, tabu search, genetic algorithms, and neural networks [56]. These techniques obtain good near-optimal solutions at reasonable computational costs. However, heuristic search techniques cannot guarantee feasibility (i.e., satisfaction of constraints) nor optimality. The length of the prediction horizon is one to two minutes. Most systems use an control time step of up to 5 seconds [75].
The main drawback of these integer optimisation problems is complexity, which is classified as NP-hard. This means that the number of potential solutions of the integer optimisation problem grows exponentially with the problem size and so does the computation time of the optimisation. The problem size depends on the number of variables in the model, which in turn depends on the level of detail of the model, the intersection topology, control time step, control horizon, and the prediction horizon. However, the computational complexity is less of an issue as optimisation solvers are getting faster and efficient as are processors. Nonetheless, the computational complexity still grows exponential with the problem size.

3-5 Problem Statement

In this section, the insights from literature describe the opportunities within distributed MPC for traffic signal control. This is combined with the problem of enabling in-vehicle information services, described in subsection 3-5-2, in order to set the objective, scope and research questions in subsection 3-5-3.

3-5-1 Insights From Literature

Coordination between intersections using actuated controllers is limited, due to its local information based on intersection detectors and its high flexibility to these detections [76]. MPC can resolve this issue by relying on prediction models and optimisation to take into account the predicted arrivals and the longer-term effects of the signal timings on the traffic system. The performance and predictability of traffic signal controllers in a network of urban intersections can be further extended by coordinating intersection controllers.

Most of the recent research regarding distributed MPC applied to traffic signal control is focused on the top-down approach. Top-down networked MPC methods have a shown track record of improving the traffic flow and delay compared to fixed-time controllers and perform best in medium to (over)saturated traffic conditions [15]. The models used are macroscopic models that are intended to represent not just one intersection, but the whole traffic network. Coordination is achieved though various optimal distributed optimisation techniques. Therefore, these methods provide good coordination between intersections (or areas) and some even attain the same results as centralised MPC methods, after many iterations or when several theoretical conditions are met. Top-down distributed MPC methods have a control time step equal the cycle time and a prediction horizon of 5 to 15 minutes. Therefore, they are a less predictable, but more flexible than the fixed-time controllers. This is illustrated in Figure 3-8, where the discussed types of controllers are shown in terms of flexibility of the signal timings to the traffic demand versus the predictability of these timings.
However, most of the methods use fixed cycle times and control the green splits within these cycles, which is not preferred by road users and Yunex (see subsection 3-3-2). Additionally, formulating a top-down distributed control problem with acyclic operation results in a distributed optimisation scheme with discrete inputs. This means that the optimisation problem becomes an integer programming problem, further complicating the distributed optimisation techniques.

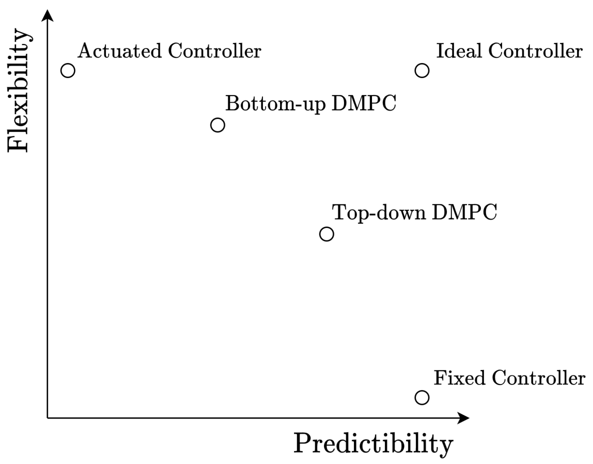


Figure 3-8: Types of controllers compared in terms of flexibility to the current traffic demand and predictability of the signal timings.

Decentralised MPC methods are well developed for single intersections. They have comparable performance to the actuated control approach in terms of delay and perform well in low to (under)saturated traffic conditions. These methods operate in an acyclic way and use queue and arrival prediction models. Coordination is mostly done through communication of planned signals and coordination parameters. Bottom-up distributed MPC methods are thus more predictable than actuated controllers, but also slightly less flexible to the traffic demand.

However, the bottom-up methods have been first developed 20 years ago and only the first development of the controllers is shared in publicly available papers. Over the last decade, however, a lot of new distributed bottom-up MPC methods have been developed [41]. This provides opportunities to pursue the bottom-up approach with new distributed bottom-up control methods.

3-5-2 Enabling In-vehicle Information Services

For the implementation of in-vehicle information services, such as GLOSA, the signal timings need to be predictable for such services to perform well [63]. This predictability of the signal timings only needs to be within a certain horizon, which is dependent on the speed of the vehicle, the current state of the traffic signal and the distance to the intersection [77]. For example, it was found that GLOSA was only effective when activated from a distance of 300 meters to the intersection at a speed limit of 50 km/h [77]. Most GLOSA methods are implemented with fixed-time traffic controllers [63]. When implemented with actuated controllers, GLOSA showed negligible effects on the number of stops and emmissions [12]. This originates from the frequently changing signal timings based on detections, which makes it difficult to predict the signal timings in advance and communicate these timings to the road users reliably [63].

When implemented correctly, GLOSA results in higher queue discharge flows as road users are informed of the starting green signal. GLOSA also results in smoother vehicle speed profiles along links as road users mostly follow the advised speed, depending on driver acceptance and market penetration of drivers using GLOSA. This also results in lower platoon dispersion, which further increases the efficiency of traffic signal control, as more vehicles can pass during a green signal if platoons are denser. GLOSA therefore has more use in traffic conditions from under-saturated to saturated demand, because drivers in oversaturated urban conditions already drive slowly since the holding capacity of the road is almost reached. This can be seen in the fundamental diagram of Figure 2-5, where the flow rate decreases once the critical density is reached which corresponds to a saturated traffic condition.

3-5-3 Objective, Research Questions and Scope

In this subsection, the objective of the thesis is advocated after considering the findings from literature and the implementation of in-vehicle information services. Then, the control problem with the optimisation objective and the hard and soft constraints for on-street implementation is covered. Finally, the scope and research questions are discussed.

Objective

Predictability and flexibility of the signal timings are contradicting goals. To obtain the ideal controller, as shown in Figure 3-8, the signal timings have to be predictable and adaptive or flexible to the traffic demand. This controller would then use a highly accurate traffic flow and intersection control model to describe the whole traffic network. Next to that, a tractable optimisation method must be available to find the optimal signals in this highly accurate model in real-time. To this day, this has not been the case. Therefore, methods have been developed for specific scenarios, however distributed MPC methods do make the gap smaller to the ideal controller.

Considering the above discussed summaries from the literature and implementation of GLOSA, the bottom-up distributed MPC method lends itself best for both goals of coordinating intersections and enabling GLOSA. Bottom-up distributed MPC methods perform better in (under)saturated conditions, which is also the best application use case of GLOSA. Next to that, there exists a great deal of distributed MPC methods which enable coordination of signal timings between intersections.
Therefore, the objective of the thesis is to develop, implement, and analyse a distributed model predictive traffic signal controller to reduce vehicular delays in urban traffic networks. This will be done using a bottom-up approach, where a decentralised version is first developed and analysed and then communication and coordination is introduced to obtain a distributed control structure.

The performance will be analysed in terms of delay, number of stops, and computation time of the optimisation technique. Since the delay represents the difference between the free-flow travel time and the actual travel time, it is a good measure to evaluate the performance of the controller. Delays thus represent any effect the traffic signal controller has on the travel time of vehicles. Number of stops is chosen as a performance indicator as more stops result in more acceleration and deceleration, increased driver discomfort, and increased greenhouse gas emissions on urban roads [63]. Lastly, since optimisation is used, it is useful to track the computation time, since this gives an insight into how the method is extendable to different kinds of intersection typologies and controller configurations, such as small time steps or longer prediction horizons.

Control Problem Definition

The control objective of the thesis is to minimise the delay in an urban arterial network. Next to the objective, several hard and soft constraints need to be satisfied in order for the controller to be safe for on-street implementation and fair for all drivers. The first and foremost hard safety constraint is that two conflicting signal groups cannot be given a green signal at the same time. Next to that, a minimum intergreen time must be satisfied. The minimum intergreen time is defined as the sum of the fixed amber time and the minimum clearance time between conflicting signals groups. To ensure safety and that drivers have enough time to react to a signal switch, a minimum duration of the red and green signal time is also demanded for on-street implementation.

Next to safety constraints, fairness for the road users of the intersection must also be addressed. Ideally, to ensure fairness, the time each vehicle spends in a queue should be limited such that drivers should not wait too long in a queue. This is called starvation and leads to drivers violating the red signal. Therefore, vehicles cannot wait longer than the maximum time without service, which represents the maximum wait time during red. Fairness can also be incorporated into the objective function as soft constraints. However, road managers prefer setting hard constraints on the minimum duration of the green and red signal times, and a maximum duration on the wait time for each signal group [78]. The following control problem summarises the objective, including the hard constraints posed by on-street implementation of the controller. The delay is equal to the time difference between the time a vehicle takes to cross the intersection at free-flow speed and to the actual time it takes for the vehicle to cross the section.

- **Optimisation objective:** Minimise delay
- **Constraints:**
  - Minimum inter-green time
  - Minimum green time
  - Minimum red time
  - Maximum time not served
  - Signal group conflicts

Scope

This thesis focuses first on the development, on-street implementation, and analysis by simulation of a decentralised model predictive traffic signal controller to reduce vehicular delays at urban intersections. Thereafter, the design of a distributed model predictive traffic signal controller is developed for urban traffic networks and implemented for on-street application. Unfortunately, due to issues with the simulation environment within Yunex Traffic, an analysis by simulation of the distributed version could not be carried out within the project time frame. To make the project feasible within the time frame, only vehicular signal groups will be used.

Research Questions

Since most of the distributed methods are older, a new intersection model is developed and a distributed optimisation method is first adapted for use in TSC. The new intersection model uses a vertical queuing model to compare the decentralised controller to DIRECTOR which also uses a vertical queuing model. This will provide a fair comparison and an insight into what the added benefit of optimisation is, which is valuable in industry as optimisation requires complex modelling, optimisation solvers, and larger computing times than the actuated control method. The decentralised controller will also be compared to the actuated controller, currently most used in the Netherlands. This provides a well-known baseline of performance. The first research question of this thesis can therefore be formulated as:

> How does a decentralised model predictive traffic signal controller perform compared to a greedy self-organising controller and an actuated controller deployed at an urban arterial intersection in terms of average delay time per vehicle, number of stops per vehicle and solution time of the used optimisation technique?

Moreover, several added features are investigated. Firstly, the decentralised controller is simulated with the LSTM prediction method described in subsection 3-4-1. This is compared to a linear arrival prediction model to assess if the linear prediction method, based on constant link delay, achieves acceptable accuracy to be used as a link equation in top-down distributed MPC methods. The second research question can be formulated as follows:

> What is the difference between a linear delay prediction arrival model and an LSTM arrival prediction model when used in conjunction with a decentralised MPC traffic signal controller in terms of average delay per vehicle, average number of stops, and normalised root mean square error?

The decentralised controller is also simulated with two different control time steps to investigate the computation time if the model size grows. This also gives an insight into its extendability to different intersection configurations and longer horizons.

> What is the influence of using different control time steps in a decentralised MPC traffic signal controller in terms of average delay per vehicle, average number of stops, and solution time of the optimisation technique?

Chapter 4

Decentralised Model Predictive Control

This chapter starts in section 4-1 with modelling the dynamics of the queues, arrivals, and constraints that are used in the prediction model of the model predictive controller. The developed model and constraints are then reformulated into a Mixed Logical Dynamical (MLD) model. Thereafter, in section 4-2, the controller design is described in which the objective function is formulated and the resulting control problem formulation including the optimisation objective, prediction model, and constraints is defined.

4-1 Modelling

First, the dynamics at urban intersections is formulated using a vertical queue model in subsection 4-1-1. Then, in subsection 4-1-2, the used arrival prediction models are briefly explained. In subsection 4-1-3, the safety, fairness and initial condition constraints are modelled in order to model the dynamics on the intersection area and enable on-street implementation of the controller. In order to make the optimisation method more efficient, the queue model and constraints are reformulated into an MLD system in subsection 4-1-4.

4-1-1 Queue Model

Since DIRECTOR uses a vertical queue model, the to be developed Decentralized Model Predictive Controller (DeMPC) will also employ such a vertical queue model. This is to provide a fair comparison between the two methods and reveal the true benefit of optimisation by using the same vertical queue model. A vertical queue model also has other benefits. As mentioned, the prediction model used in Model Predictive Control (MPC) has to be formulated where a trade-off has to be made between computational complexity of the optimisation technique and the accuracy of the model in representing the real-world dynamics. The requirement of acyclic operation of traffic signals requires modelling the signal switches at any time step over the prediction horizon, which leads to integer optimisation. Unfortunately, the computation time of integer programming problems grows exponentially with the problem size. Therefore, a vertical queue model is beneficial, because it has less variables than a horizontal queue model [57]. Additionally, the decentralised intersection model should be designed to be compatible with coordination algorithms used in the distributed controller, which also adds extra computation time due to a possibly larger model and/or iterations for seeking consensus. However, it should be noted that computation time is not such a limiting problem anymore due to faster solvers and processors [42].

When a signal group indicates a green signal, the number of leaving vehicles, i.e. departures, can be modelled as the minimum of two terms, namely the sum of the number of cars waiting in the queue plus the arrivals and the saturation holding capacity during the simulation period. The departures leaving a queue, $n_s^{\mathrm{d}}(k)$, during the simulation period $[kT_{\mathrm{c}}, (k+1)T_{\mathrm{c}})$, where k indicates the control time step counter and T_{c} indicates the control time step, can thus be described as

$$n_s^{\mathrm{d}}(k) = \begin{cases} 0 & \text{if } u_s(k) = 0 \\ \min\left(n_s^{\mathrm{q}}(k) + n_s^{\mathrm{a}}(k), \mu_s^{\mathrm{sat}} T_{\mathrm{c}}\right) & \text{if } u_s(k) = 1, \end{cases} \tag{4-1}$$

for all signal groups s in the set $\mathcal{S}_i$, which denotes the set of signal groups of intersection i. The saturation flow rate (expressed in vehicles/second) is represented by μ_s^{sat} and the number of queued vehicles is denoted as $n_s^{\mathrm{q}}(k)$ with its number of arrivals as $n_s^{\mathrm{a}}(k)$. In Equation 4-1, a traffic signal is modelled by a binary variable, $u_s(k)$, where a green signal corresponds to one and red to zero. The two terms in the min() operator correspond to undersaturated and saturated traffic conditions respectively. Equation 4-1 is inspired on the BXL model by M. van den Berg [79]. However, here, the dynamics are not averaged over the cycle time and no oversaturated term is taken into account as the model is meant for undersaturated to saturated traffic conditions. The queue is represented as a vertical queue and its evolution is described by

$$n_s^{\mathrm{q}}(k+1) = n_s^{\mathrm{q}}(k) + n_s^{\mathrm{a}}(k) - n_s^{\mathrm{d}}(k). \tag{4-2}$$

A graphical representation of the arrival, queue and departure variables at an intersection can be seen in Figure 4-1.

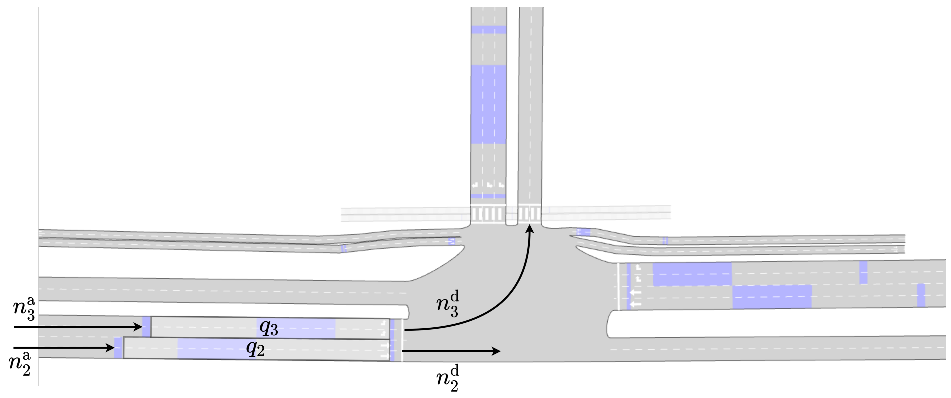


Figure 4-1: Arrival, queue and departure variables for signal groups 2 and 3 at intersection.

The arrivals n_s^{a} are predicted at the location of the arrival loop detector by a Long Short-Term Memory (LSTM) neural network. After the arrivals have been predicted, they are added to the queue. The vertical queue model is thus not placed at the stopline, but at the the location of the arrivals loops. The LSTM neural network is trained to predict vehicle arrivals at the arrival loop, since data of arrivals is only available at that location in the link between intersections.

Several modelling assumptions are taken in the departure and queue modelling:

1. When in saturated traffic condition and a signal group switches from red to green, vehicles depart immediately at the discharge saturation flow rate. The modelled flow profile using the binary formulation for the traffic signal is illustrated for saturated the traffic flow condition in Figure 4-2, where the actual flow profile is also shown.
2. Vehicles arriving at the end of a queue in simulation period $[kT_\mathrm{c}, (k+1)T_\mathrm{c})$ are allowed to depart in that same period, provided a green signal is indicated and the sum of the arriving vehicles and the number of vehicles in the queue is smaller than the number of vehicles that can leave during the saturated condition [44].
3. The arrivals are predicted at the location of the arrival loops after which they are added to the queue.

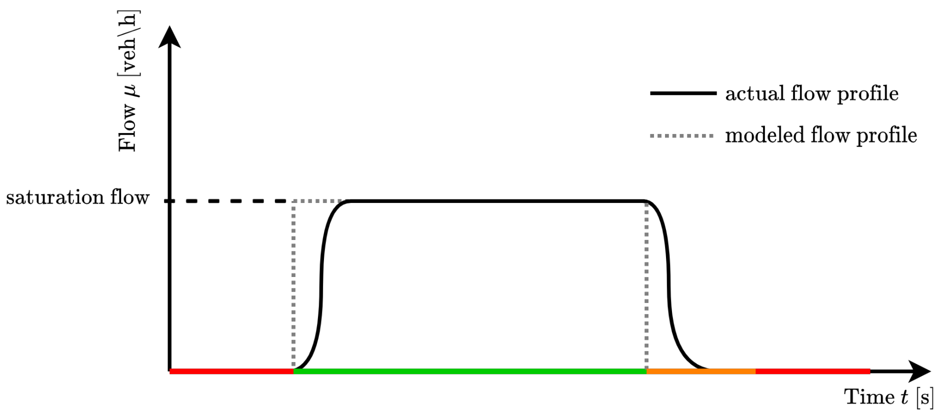


Figure 4-2: Modelled and actual flow profile in saturated condition. Red, green and amber bars indicate the signal group colour.

4-1-2 Arrival Prediction Models

In this subsection, the two arrival prediction methods will be explained. The first is an LSTM neural network developed by previous interns and the second is the linear prediction method based on a constant link delay.

LSTM Prediction Model

In this model, the arrivals are predicted by LSTM neural network first developed by N. Helmy [63] and later improved upon by J.C. Van Senden [64] and T. Glastra [62]. The input matrix of the LSTM model consists of the the following data: Day of week, time of day, number of upstream departures from the neighbouring intersection, queue detections, number of arrivals

arrivals and signal states. The input and the neural network architecture for one approaching link to an intersection is depicted in Figure 4-4. The network consists of one LSTM layer of 50 neurons, two fully connected layers of 50 and 100 neurons respectively and lastly an output layer where the amount of neurons is dependent on the number of signal groups of the approach. The fully connected layers have a tanh activation function and the output layer has a linear activation function. The input data is gathered for the last 100 seconds, i.e., from the previous time step ($T = -1$) up until $T = -n$, where n is the number of time steps corresponding to 100 seconds. A neural network is trained per approaching link and per prediction horizon. The location of the input detectors for one approaching link to

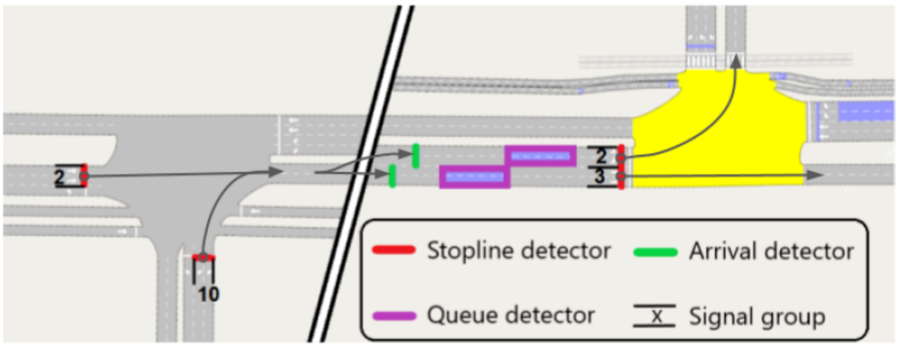


Figure 4-3: Location of detector inputs for arrival prediction. Source [62].

an intersection can be seen in Figure 4-3. For more information on the architecture of the neural network, refer to the thesis by J.C. Van Senden [64]. For information regarding data aggregation and preprocessing steps of the data, refer to the thesis by T. Glastra [62].

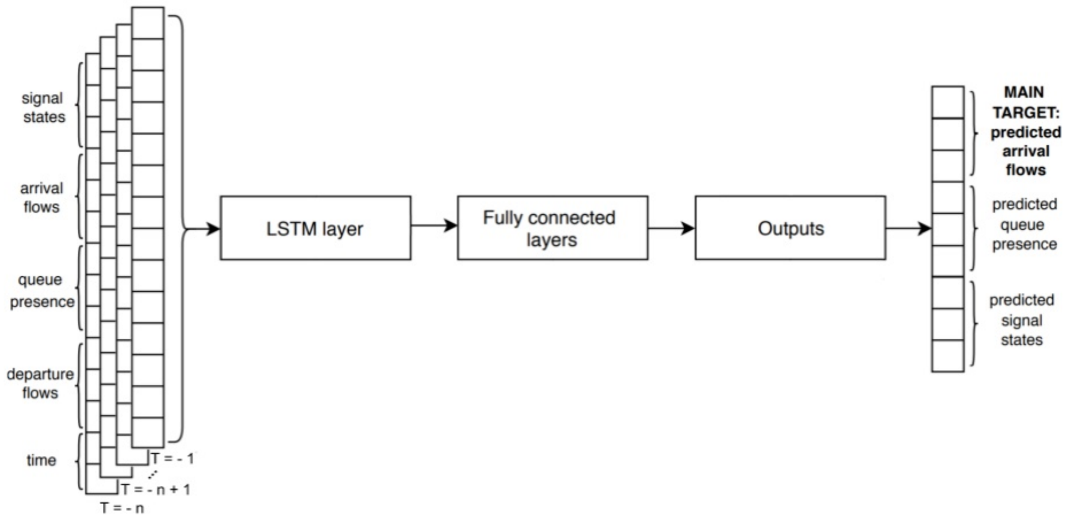


Figure 4-4: Input, output and LSTM architecture. Source [64].

Linear Arrival Prediction Model

In the linear prediction model, the arrivals are calculated using a fixed link delay, based on a constant free-flow speed which is set to the speed limit. This is done using the following equation

$$n_s^{\mathrm{a}}(k+N_{\mathrm{p}}) = p_s^{\mathrm{turn}} n_{ji}^{\mathrm{d}}(k - D_{j,i}^{\mathrm{link}} + N_{\mathrm{p}}) \tag{4-3}$$

where p_s^{turn} indicates the turning percentage of signal group $s \in \mathcal{S}_i$ and $n_{ji}^{\mathrm{d}}(k - D_{j,i}^{\mathrm{link}} + N_{\mathrm{p}})$ denotes the sum of the departures of neighbouring intersection j leading to intersection i at time step k minus the link delay $D_{j,i}^{\mathrm{link}}$, which is defined as

$$D_{j,i}^{\mathrm{link}} = \left\lfloor \frac{L_{j,i}^{\mathrm{link}}}{v_{j,i} T_{\mathrm{c}}} \right\rfloor \tag{4-4}$$

with link length $L_{j,i}^{\mathrm{link}}$ in meters and free-flow speed equal to the speed limit $v_{i,j}$ expressed in meters per second. This arrival prediction model was chosen to investigate if it would obtain an acceptable accuracy and performance for use in distributed optimisation techniques that require decomposition of the link equation between two intersections. This link delay is also used in the S-model for cyclic operation by S. Lin [44].

4-1-3 Constraint modelling

In order to represent the dynamics on the intersection area, several hard constraints need to be defined on the timings of the binary variable $u_s \in \{0, 1\}$. These constraints are important to guarantee safety and fairness for the road users and fulfil the on-street implementation requirements. Unless specified otherwise, the following constraints all hold for time steps $k \in [0, N_{\mathrm{p}} - 1]$, where N_{p} represents the prediction horizon of the model. Next to that, all used symbols for signal groups belong the the set of signal groups of intersection i, i.e., signals $r, s \in \mathcal{S}_i$.

Conflict Constraints

The first and foremost hard safety constraint ensures that two conflicting signal groups cannot be given a green signal at the same time. This can formulated as

$$\sum_{s \in \mathcal{C}_n} u_s(k) \leq 1 \quad \forall\ \mathcal{C}_n \in \mathcal{C}, \tag{4-5}$$

where the conflict set $\mathcal{C}$ is composed of subsets $\mathcal{C}_n$ defined as

$$\mathcal{C} := \{\mathcal{C}_n,\ n = 1, ...|\mathcal{C}|\}. \tag{4-6}$$

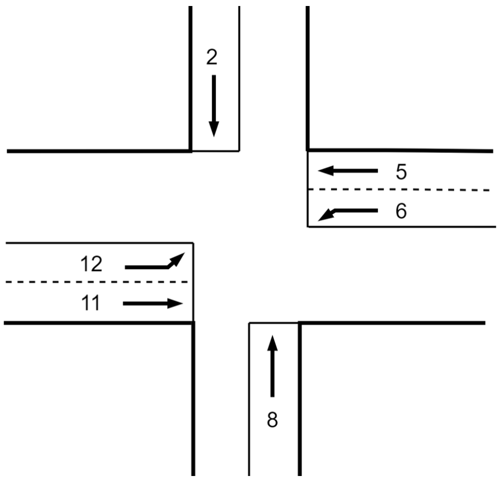
It is important how this set of conflicts is defined. The set can be defined as containing all the sets of pairs of conflicting signal groups. Using the intersection as illustrated in Figure 4-5a and its corresponding conflicts in Figure 4-5b, the set $\mathcal{C}$ would then become

$$\begin{aligned} \mathcal{C} = \{&\{2,5\}_1, \{2,6\}_2, \{2,11\}_3, \{2,12\}_4, \{5,8\}_5, \\ &\{5,12\}_6, \{6,8\}_7, \{6,11\}_8, \{8,11\}_9, \{8,12\}_{10}\}. \end{aligned} \tag{4-7}$$

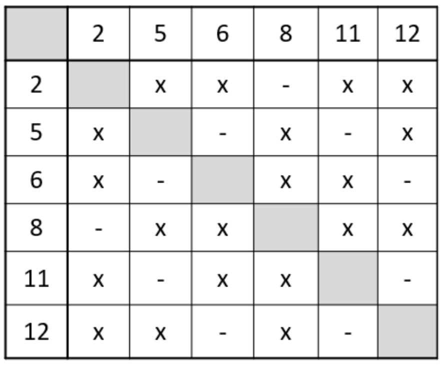
However, there is a more efficient way to define the conflict subsets by combining the conflict pairs into pairwise different subsets. Following the same example, the set would then become

$$\mathcal{C} = \{\{2,5,12\}_1, \{2,6,11\}_2, \{5,8,12\}_3, \{6,8,11\}_4\}\,. \tag{4-8}$$

This has the implication that fewer conflict constraints of Equation 4-6 need to be defined. Additionally, it also has benefits when solving the optimisation problem using Mixed-Integer Linear Programming (MILP). Using the formulation in Equation 4-8, the feasible region of the relaxed linear programming problem in the branch-and-bound method is closer to the convex hull of the original MILP. Then, integer feasible solutions are found at corners of the feasible region. This leads to the reduction in branches as the solution of the linear programming problem also satisfies MILP constraints. This is shown and proved by S. Göttlich et al. [80].



(a) Example intersection with numbers indicating signal groups.



▨	2	5	6	8	11	12
2	▨	x	x	-	x	x
5	x	▨	-	x	-	x
6	x	-	▨	x	x	-
8	-	x	x	▨	x	x
11	x	-	x	x	▨	-
12	x	x	-	x	-	▨

(b) Symmetrical conflict matrix, where conflicts between signal groups are indicated with X and non-conflicting signal groups with -.

Figure 4-5: Intersection and its corresponding conflict matrix

Minimum Green Time

The green signal must at least last longer than the minimum green time. If the green time is too short, driver's may violate the red signal. In more detail, if a signal switches from zero at time step k to one at time step $k+1$, then the next $T_s^{g_{\min}}$ time steps should also be equal to one. This can be represented by the following logical statement

$$[u_s(k) = 0 \wedge u_s(k+1) = 1] \implies [u_s(k+1) = \ldots = u_s(k + T_s^{g_{\min}}) = 1] \tag{4-9}$$

The above can be ensured when the following constraints are satisfied

$$\sum_{\tau=k+1}^{k+T_s^{g_{\min}}} u_s(\tau) \geq T_s^{g_{\min}}\left(u_s(k+1) - u_s(k)\right) \quad \forall\, k \in [0, N_p - T_s^{g_{\min}} - 1] \tag{4-10a}$$

$$\sum_{\tau=k+1}^{N_p-1} u_s(\tau) \geq |T|\left(u_s(k+1) - u_s(k)\right) \quad \forall\, k \in [N_p - T_s^{g_{\min}}, N_p - 1) \tag{4-10b}$$

with $T = [k, N_\mathrm{p} - 1]$. The minimum green constraints can be easily checked by filling in the left side of Equation 4-9 and recalling that $u_s \in \{0, 1\}$. The minimum green constraints are only implemented when the control time step T_c is smaller than than $T_s^{g_\mathrm{min}}$. The minimum green time steps $T_s^{g_\mathrm{min}}$ is defined as the minimum green time $t_s^{g_\mathrm{min}}$ divided by the control time step and then ceiled to the nearest integer. The minimum green time and control time steps are both defined in seconds.

$$T_s^{g_\mathrm{min}} = \left\lceil \frac{t_s^{g_\mathrm{min}}}{T_\mathrm{c}} \right\rceil \tag{4-11}$$

Minimum Red Time

For the same reason as the minimum green time, the minimum red time is required. If the the duration of a signal group is shorter than the minimum red time, drivers may think the traffic signal is broken. The constraint can also be represented by the following logical statement

$$[u_s(k) = 1 \wedge u_s(k+1) = 0] \implies [u_s(k+1) = \ldots = u_s(k + T_s^{r_\mathrm{min}}) = 0], \tag{4-12}$$

which can be modelled by the following constraints

$$\sum_{\tau=k+1}^{k+T_s^{r_\mathrm{min}}} u_s(\tau) \leq (T_s^{r_\mathrm{min}} + 1)\,(1 - u_s(k) + u_s(k+1)) \quad \forall\, k \in [0, N_p - T_s^{r_\mathrm{min}} - 1] \tag{4-13a}$$

$$\sum_{\tau=k+1}^{N_p-1} u_s(\tau) \leq (|T| + 1)\,(1 - u_s(k) + u_s(k+1))\,, \quad \forall\, k \in [N_p - T_s^{r_\mathrm{min}}, N_p - 1) \tag{4-13b}$$

with $T = [k, N_\mathrm{p} - 1]$. Again, the minimum red constraints can be easily checked by filling in the left side of Equation 4-12 and recalling that $u_s \in \{0, 1\}$. The minimum red constraint is only implemented when the control time step T_c is smaller than than $T_s^{r_\mathrm{min}}$. The minimum red time steps $T_s^{r_\mathrm{min}}$ is defined as the minimum red time $t_s^{r_\mathrm{min}}$ divided by the control time step and then ceiled to the nearest integer.

$$T_s^{r_\mathrm{min}} = \left\lceil \frac{t_s^{r_\mathrm{min}}}{T_\mathrm{c}} \right\rceil \tag{4-14}$$

Maximum Wait Time

Next to safety constraints, fairness for the road users of the intersection must also be addressed. Ideally, to ensure fairness, the time each vehicle spends in a queue should be limited such that drivers should not wait too long in a queue. Unfair control can happen, for example, when traffic demand is unbalanced such as on an arterial road, where the main road has a high traffic demand and the minor road has a low demand. If in this case the objective is to minimise the delay and there are no fairness constraints, then the vehicles on the minor road will not receive a green signal and are kept waiting forever. This is called starvation and leads to drivers violating the red signal. Therefore, vehicles cannot wait longer than the maximum time without service or here described as maximum wait time. As mentioned, fairness can also be incorporated into the objective function as soft constraints. However, road owners prefer setting hard constraints on minimum duration on the green and red signal times and a maximum duration on the wait time. To avoid starvation a maximum wait time is imposed and is implemented using the following constraints

$$\sum_{\tau=k+1}^{k+T_s^{\text{wmax}}+1} u_s(\tau) \geq 1 \quad \forall\, k \in [0, N_p - T_s^{\text{wmax}} - 1] \tag{4-15a}$$

$$\sum_{\tau=k+1}^{N_p-1} u_s(\tau) \geq 1 \quad \forall\, k \in [N_p - T_s^{\text{wmax}}, N_p - 1) \tag{4-15b}$$

with $T = [k, N_\text{p} - 1]$ and where T_s^{wmax} denotes the maximum time steps a vehicle can wait in front of a red signal and is defined as

$$T_s^{\text{wmax}} = \left\lfloor \frac{t_s^{\text{wmax}}}{T_\text{c}} \right\rfloor \tag{4-16}$$

The constraint is thus only implemented if there is a red signal and there is at least one vehicle waiting in the queue. Then the above maximum wait time constraint forces the signal to be at least once set to one by the time the maximum wait time is passed.

Minimum Intergreen Time

When a signal group switches to green, the minimum intergreen time must be satisfied between that signal group and all its conflicting signal groups. The minimum intergreen time is defined as the sum of the fixed amber time and the minimum clearance time between every ordered conflicting signals groups. The minimum clearance time and intergreen time are illustrated in Figure 4-6. The minimum intergreen time is defined for each ordered pair of conflicting signal groups, since the clearance time is dependent on the location where the movement paths of the conflicting signal groups cross. The ordered conflict set $\mathcal{C}_\text{o}$ is defined as all combinations of two different conflicting signal group pairs. For example, in Figure 4-5b, every X in the conflict matrix corresponds to one ordered conflicting signal group pair. The constraint on

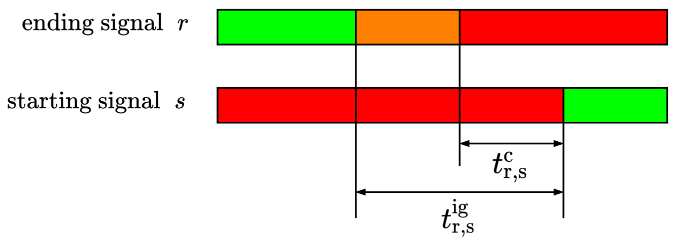


Figure 4-6: Clearance time $t^{\text{c}}_{r,s}$ and intergreen time $t^{\text{ig}}_{r,s}$ between two conflicting signals $(r, s) \in \mathcal{C}_\text{o}$. The green, amber and red bars indicate the signal group colours.

the intergreen time can be formulated as a logical statement as follows,

$$[u_r(k) = 1 \wedge u_r(k+1) = 0] \Longrightarrow \left[u_s(k+1) = \cdots = u_s(k + T^{\text{ig}}_{r,s}) = 0\right] \quad \forall\, (r, s) \in \mathcal{C}_\text{o} \tag{4-17}$$

This can be modelled into the following two constraints

$$\sum_{\tau=k+1}^{k+T^{\mathrm{ig}}_{r,s}} u_s(\tau) \leq \left(T^{\mathrm{ig}}_{r,s}+1\right)(1-u_r(k)+u_r(k+1)) \quad \forall\, k \in \left[0, N_p - T^{\mathrm{ig}}_{r,s}-1\right] \tag{4-18}$$

$$\sum_{\tau=k+1}^{N_p-1} u_s(\tau) \leq (|T|+1)(1-u_r(k)+u_r(k+1)) \quad \forall\, k \in \left[N_p - T^{\mathrm{ig}}_{r,s}, N_p-1\right) \tag{4-19}$$

with $T=[k, N_{\mathrm{p}}-1]$ and where $T^{\mathrm{ig}}_{r,s}$ denotes the minimum time steps that must pass between the ending of conflicting signal r and the beginning of signal s. This is defined as

$$T^{\mathrm{ig}}_{r,s} = \left\lceil \frac{t^{\mathrm{ig}}_{r,s}}{T_{\mathrm{c}}} \right\rceil \tag{4-20}$$

Initial Conditions Constraints

The receding horizon policy of MPC, displayed in Figure 3-4, ignores past dynamics and only takes the current initial conditions (obtained from measurements) and the prediction model into account. However, in Traffic Signal Control (TSC), some information has to be known about the past signal timings in order to ensure the minimum intergreen time, minimum red, minimum green and maximum wait time are satisfied. Therefore, the initial condition constraints take the last implemented signal timings into account and impose additional constraints to the model dynamically. In order to maintain the minimum red time, the difference in number of time steps between the current red time, t^{r}_s, and the minimum red time is defined as,

$$\Delta^{\mathrm{r}}_s = \left\lceil \frac{t^{r_{\min}}_s - t^{\mathrm{r}}_s}{T_{\mathrm{c}}} \right\rceil \tag{4-21}$$

to formulate the following minimum red initial conditions constraint,

$$\sum_{\tau=0}^{k+\Delta^r_s} u_s(\tau) \leq 0. \tag{4-22}$$

The same can be done for maintaining the minimum green time using the difference in time steps between the current green time t^g_s and the minimum green time in the following initial condition constraint,

$$\sum_{\tau=0}^{k+\Delta^{\mathrm{g}}_s} u_s(\tau) \geq \Delta^{g,i}_s, \tag{4-23}$$

where Δ^{g}_s is defined as

$$\Delta^{\mathrm{g}}_s = \left\lceil \frac{t^{g_{\min}}_s - t^g_s}{T_{\mathrm{c}}} \right\rceil \tag{4-24}$$

Lastly and most importantly, the minimum intergreen time must be satisfied at all times. Similarly to the previous constraints, this is done by tracking the time since the last green (in seconds) of the conflicting signal t^{ig}_r. This is then used in the following constraint

$$\sum_{\tau=0}^{\Delta^{\mathrm{ig}}_{r,s}} u_s(\tau) \leq 0, \quad \forall\, (r,s) \in \mathcal{C}_o, \tag{4-25}$$

where Δ_s^{ig} is defined as

$$\Delta_s^{\mathrm{ig}} = \left\lceil \frac{t_s^{ig_{\min}} - t_s^{ig}}{T_{\mathrm{c}}} \right\rceil \tag{4-26}$$

Note that the initial condition constraints to maintain the minimum green and minimum red time are only implemented when the control time step is larger then the minimum green and red times.

4-1-4 Model Reformulation

If the previously developed queue model of Equations 4-1 and 4-2 and its constrains defined in subsection 4-1-3 would be used as the prediction model in MPC, then the optimisation problem would become nonlinear and non-convex. The min() operator of Equation 4-1 and the dependency of this operator on the the traffic signal $u_s \in \{0, 1\}$ introduce the nonconvex and non-linear dynamics. Nonlinear and nonconvex optimisation problems can be solved using nonlinear integer optimisation methods such as mixed-integer nonlinear programming, genetic algorithms, simulated annealing, tabu search, or other heuristic algorithms [81]. These optimisation methods all require a large number of evaluations of the model and the objective function. This increases the solution time of the optimisation algorithm [82]. Nonlinear and nonconcvex optimisation problems have an exponentially growing computational complexity when the size of the model grows. The model size depends on its accuracy in describing the real-world dynamics, the intersection topology, control time step, control horizon, and the prediction horizon.
Therefore, the model is reformulated using the MLD framework introduced by Bemporad and Morari [83]. This reformulation will result in a MILP programming optimisation problem, where the complexity still grows exponentially with the problem size, but for these problems very efficient solvers exist [56]. The MLD framework is developed by reformulating model equations that combine physical dynamics, logical rules, and operating constraints [83]. This fits the description of controlling traffic signals as the physical dynamics are here represented by the traffic flow dynamics, logical rules by the traffic signal switches, and the operating constraints on those signal switches.
The reformulation will be done by transforming the nonlinear nonconvex equations into mixed-integer linear inequality constraints. The resulting model then constitutes of linear equations subject to linear mixed-integer inequalities, in which both continuous and binary variables are present. The formulation is inspired upon the reformulation of the by S. Lin [44], but heavily adapted to fit this use case.

Used Rules for Reformulation

The following rules are used to reformulation logical statements into mixed-integer linear inequalities involving both continuous variables $x \in \mathbb{R}^n$ and logical variables $\delta \in \{0, 1\}$. The explanation of the rules is entirely based upon the paper by Bemporad and Morari [83].

Consider the function $f : \mathbb{R}^n \to \mathbb{R}$ and assume that $x \in \mathcal{X}$, where $\mathcal{X}$ is a finite set, and define,

$$M = \max_{x \in \mathcal{X}} f(x), \quad m = \min_{x \in \mathcal{X}} f(x) \tag{4-27}$$

Theoretically, an over-estimate (under-estimate) of M (m) suffices for our purpose. However, more realistic estimates provide computational benefits [84]. It is proven in [83] that the following logical statement is equivalent to two mixed integer inequalities,

$$([f(x) \leq 0] \Leftrightarrow [\delta = 1]) \Longleftrightarrow \begin{cases} f(x) \leq M(1-\delta) \\ f(x) \geq \varepsilon + (m-\varepsilon)\delta \end{cases} \tag{4-28}$$

where ε is a very small tolerance, typically the machine precision. Moreover, $\delta f(x)$ can be replaced by the auxiliary real variable $z = \delta f(x)$, which satisfies $[\delta = 0] \implies [z = 0]$ and $[\delta = 1] \implies [z = f(x)]$. Then, $z = \delta f(x)$ is equivalent to

$$z = \delta f(x) \Longleftrightarrow \begin{cases} z \leq M\delta \\ z \geq m\delta \\ z \leq f(x) - m(1-\delta) \\ z \geq f(x) - M(1-\delta) \end{cases} \tag{4-29}$$

Reformulation of Queue Model

Below, it is shown how the model can be reformulated as mixed-integer linear equations and inequalities using the above equivalent reformulation rules. Recall, the number of departures leaving a queue is defined as,

$$n_s^{\mathrm{d}}(k) = \begin{cases} 0 & \text{if } u_s(k) = 0 \\ \min\left(n_s^{\mathrm{q}}(k) + n_s^{\mathrm{a}}(k), \mu_s \cdot T_{\mathrm{c}}\right) & \text{if } u_s(k) = 1, \end{cases} \tag{4-30}$$

For readability, in the following reformulation, the dependence of the variables on the time step $k \in [0, N_p - 1]$ and the signal $s \in \mathcal{S}_i$ is not stated.
To start the reformulation define the following

$$\begin{aligned} a &= n^{\mathrm{q}} + n^{\mathrm{a}} \\ b &= \mu T_{\mathrm{c}} \\ d &= \min(a, b) \end{aligned} \tag{4-31}$$

and let

$$f_1 = a - b. \tag{4-32}$$

Next to f_1, define the binary variable δ as

$$\delta = \begin{cases} 1 & \text{if } f_1 \leq 0 \\ 0 & \text{if } f_1 > 0. \end{cases} \tag{4-33}$$

where the undersaturated traffic condition corresponds with $\delta = 1$ and the saturated traffic condition with $\delta = 0$. Then, d can be written as

$$d = b + (a-b)\delta = b + f_1\delta \tag{4-34}$$

Define the product in Equation 4-34 as

$$z = f_1\delta \tag{4-35}$$

According to the equivalent transformation rules of Equations 4-28 and 4-29, Equation 4-35 and Equation 4-33 are equivalent to the inequality constraints

$$\begin{aligned} &f_1 \leq M_1 (1-\delta) \\ &f_1 \geq \varepsilon + (m_1 - \varepsilon)\,\delta \\ &z \leq M_1 \delta \\ &z \geq m_1 \delta \\ &z \leq f_1 - m_1 (1-\delta) \\ &z \geq f_1 - M_1 (1-\delta) \end{aligned} \tag{4-36}$$

Then Equation 4-30 becomes,

$$n^{\mathrm{d}} = \begin{cases} 0 & \text{if } u = 0 \\ b + z & \text{if } u = 1, \end{cases} \tag{4-37}$$

The above equation can be written as

$$n^{\mathrm{d}} = (b+z)u. \tag{4-38}$$

Similarly to $z = f_1\delta$, the above equation can be reformulated into the following linear inequality constraints, where $f_2 = b + z$,

$$\begin{aligned} &n^{\mathrm{d}} \leq M_2 u \\ &n^{\mathrm{d}} \geq m_2 u \\ &n^{\mathrm{d}} \leq f_2 - m_1 (1-u) \\ &n^{\mathrm{d}} \geq f_2 - M_1 (1-u)\,. \end{aligned} \tag{4-39}$$

As defined in Equation 4-27, M_1 and m_1 are the maximum value and the minimum value of f_1, and M_2 and m_2 are the maximum value and the minimum value of f_2. Since $f_1 = a - b$ and $f_1 = b - z$, to define maximum and minimum of these functions, the upper bounds and lower bounds of a, b must first be clarified as

$$\begin{aligned} &a_{\min} = 0 \leq n^{\mathrm{q}} + n^{\mathrm{a}} \leq C = a_{\max} \\ &b_{\min} = 0 \leq \mu \cdot T_{\mathrm{c}} \leq \mu T_{\mathrm{c}} = b_{\max} \end{aligned} \tag{4-40}$$

where all lower bounds are zero, since the traffic flow rate cannot be negative. The upper bound for a is the maximum holding capacity of the approaching link, C. Naturally, the upper bound for b is the saturation flow rate during the control time step. With the upper bounds and lower bounds, the values for M and m can be derived as

$$\begin{aligned} &M_1 = \max f_1 = a_{\max} - b_{\min} = C \\ &m_1 = \min f_1 = a_{\min} - b_{\max} = -\mu T_{\mathrm{c}} \\ &M_2 = \max f_2 = b_{\max} + z_{\max} = \mu T_{\mathrm{c}} + C \\ &m_2 = \min f_2 = b_{\min} - z_{\min} = 0 \end{aligned} \tag{4-41}$$

Recall, the introduced variables are defined per signal s and dependent on time step k. For clarification, the reformulated equations will be summarised in the formulation where they are implemented. The min operator of Equation 4-30 is reformulated into

$$\begin{aligned} &f_{s,1}(k) \leq M_{s,1}(k)\,(1 - \delta_s(k)) \\ &f_{s,1}(k) \geq \varepsilon + (m_{s,1} - \varepsilon)\,\delta_s(k) \end{aligned} \tag{4-42}$$

using the newly introduced auxiliary binary variable $\delta_s(k)$ and real variables $f_{s,1} = n_s^{\mathrm{q}}(k) + n_s^{\mathrm{a}}(k) - \mu_s^{\mathrm{sat}} T_{\mathrm{c}}$, then $z_s(k) = f_s(k)\delta_s(k)$ is reformulated as

$$\begin{aligned} z_s(k) &\leq M_{s,1}\delta_{s,1}(k) \\ z_s(k) &\geq m_{s,1}\delta_{s,1}(k) \\ z_s(k) &\leq f_s(k) - m_{s,1}\left(1 - \delta_s(k)\right) \\ z_s(k) &\geq f_s(k) - M_{s,1}\left(1 - \delta_s(k)\right). \end{aligned} \tag{4-43}$$

Consequently, the dependency of the min operator on the binary traffic signal $u_s(k)$ in the departure model of Equation 4-30 is reformulated in

$$\begin{aligned} n_s^{\mathrm{d}}(k) &\leq M_{s,2}u_s(k) \\ n_s^{\mathrm{d}}(k) &\geq m_{s,2}u_s(k) \\ n_s^{\mathrm{d}}(k) &\leq f_{s,2}(k) - m_{s,2}\left(1 - u_s(k)\right) \\ n_s^{\mathrm{d}}(k) &\geq f_{s,2}(k) - M_{s,2}\left(1 - u_s(k)\right) \end{aligned} \tag{4-44}$$

using the newly introduced auxiliary real variable $f_{s,2} = \mu_s^{\mathrm{sat}} T_{\mathrm{c}} + z_s$ and with $M_{s,1} = C$, $m_{s,1} = -\mu_s^{\mathrm{sat}} T_{\mathrm{c}}$, $M_{s,2} = \mu_s^{\mathrm{sat}} T_{\mathrm{c}} + C$ and $m_{s,2} = 0$.

Reformulation of Constraints

The constraints defined in subsection 4-1-3 can be directly integrated into an MLD formulation as they are already formulated as linear inequalities.

4-1-5 Resulting MLD Model

In this section the general MLD prediction model description will be covered, after which this description will be adapted to the developed reformulated model.

General MLD System

The general MLD system description is as follows,

$$\begin{aligned} x(k+1) &= Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) \\ y(k) &= Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k) \\ E_1 x(k) + E_2 u(k) &+ E_3 \delta(k) + E_4 z(k) \leqslant g_5 \end{aligned} \tag{4-45}$$

where the state vector $x(k)$ is partitioned into a real-valued and binary component as,

$$x(k) = \begin{bmatrix} x_r(k) \\ x_b(k) \end{bmatrix}, x_r(k) \in \mathbb{R}^{n_r}, x_b(k) \in \{0,1\}^{n_b}, n = n_r + n_b. \tag{4-46}$$

The controlled input is also similarly partitioned

$$y(k) = \begin{bmatrix} y_r(k) \\ y_b(k) \end{bmatrix}, y_r(k) \in \mathbb{R}^{n_r}, y_b(k) \in \{0,1\}^{n_b}, l = l_r + l_b, \tag{4-47}$$

as is the control input $u(k)$

$$u(k) = \begin{bmatrix} u_c(k) \\ u_b(k) \end{bmatrix}, u_c(k) \in \mathbb{R}^{m_r}, u_b(k) \in \{0,1\}^{m_b}, m = m_r + m_b. \tag{4-48}$$

The real-valued auxiliary variables are $z(k) \in \mathbb{R}^{r_\mathrm{r}}$ and binary auxiliary variables $\delta(k) \in \{0,1\}^{r_\mathrm{b}}$. The matrices B_2 B_3, D_2 and D_3 are used to include the effects of the auxiliary variables into the dynamics, while the inequality constraints are given through E_1,E_2,E_3,E_4 and the vector g_5. All matrices are of appropriate dimension.

Reformulated MLD Model

In the case of the reformulated model of intersection i, the MLD system description is as follows,

$$x_i(k+1) = A_i x_i(k) + B_{1,i}u_i(k) + B_{2,i}\delta_i(k) + B_{3,i}z_i(k) + B_{4,i}v_{j,i}(k) \quad \forall\ k \tag{4-49}$$

$$E_{1,i}x_i(k) + E_{2,i}u_i(k) + E_{3,i}\delta_i(k) + E_{4,i}z_i(k) + E_{5,i}v_{j,i}(k) \leqslant E_{5,i} \quad \forall\ k \tag{4-50}$$

$$\mathbf{E}_{6,i}\mathbf{u}_i(k) \leqslant \mathbf{E}_{7,i} \tag{4-51}$$

The state vector is defined as

$$x_i(k) = \begin{bmatrix} n_i^q(k) \\ n_i^d(k) \end{bmatrix} \tag{4-52}$$

with queue and departure vectors $n_i^q(k), n_i^d(k) \in \mathbb{R}_+^{m_i}$ and m_i indicating the number of signals of intersection i, $m_i = |\mathcal{S}_i|$. The arrival state is here described as a separate exogenous input variable $v_{j,i} = n_i^a(k) \in \mathbb{R}_+^{m_i}$, since these are predicted by the LSTM neural network. Subscript j indicates that the arrivals originated from neighbour j which is part of the set of neighbouring intersections, $j \in \mathcal{N}_i$. The binary control inputs are defined as $u(k) \in \{0,1\}^{m_i}$ and auxiliary variables as $\delta_i(k) = \delta(k) \in \{0,1\}^{m_i}$ and $z_i(k) = z(k) \in \mathbb{R}_+^{m_i}$. The system outputs the state of the intersection to the controller. This defines the output of the system as $y(k) = x(k)$. Equation 4-51 incorporates the safety and fairness constraints. These are not defined per time step, but across the whole prediction horizon. Therefore the bold input vector is defined over the whole horizon, $\mathbf{u}_i(k) = [u_i(0) \dots u_i(N_p-1)]^\top$. All matrices are of appropriate dimension to the size of the vectors.

4-2 Controller Design

As mentioned, MPC can be formulated as an optimisation problem, where an objective function is defined subject to constraints. In subsection 4-2-1, the objective function will be defined. Thereafter, in subsection 4-2-2,the optimisation problem is specified to give a formal overview of the developed controller.

4-2-1 Objective Function

The objective function follows the objective of the controller defined in subsection 3-5-3, which is to minimise the delay of the vehicles passing through the intersection. While the delay cannot be directly measured or calculated using the variables defined in the model, it can easily be calculated by integrating the queue length over the prediction horizon. Since vehicles incur most of their delay standing in a queue, minimising the queue length can be used to approximate the delay. In Figure 4-7, the difference between the control delay and the stopped or queue delay can be seen. The control delay is defined as is the difference between the free-flow travel time and the actual incurred travel time. The queue delay only represents the delay time incurred from standing in the queue.

With the prediction horizon length N_p and discrete time step $k \in [0, N_\mathrm{p} - 1]$, the objective function of intersection i is given by,

$$J_i(k) = \sum_{k=0}^{N_\mathrm{p}-1} c_i^\top n_i^\mathrm{q}(k), \tag{4-53}$$

where $c_i = [\; c_{i,1} \;\; \dots \;\; c_{i,m_i} \;]^\top$ and $n_i^\mathrm{q}(k) = [\; n_{i,1}^\mathrm{q}(k)) \;\; \dots \;\; n_{i,m_i}^\mathrm{q}(k) \;]^\top$ denote the weight and queue length vectors with m_i the number of signals of intersection i. The objective function of Equation 5-1 is a linear where every vehicle waiting within the queue is weighted the same amount. A quadratic cost function can also be used, in which the longer queues have a much higher cost than shorter ones, giving more priority to these longer queues. This can be an alternative to setting hard constraints on the wait time to avoid starvation.

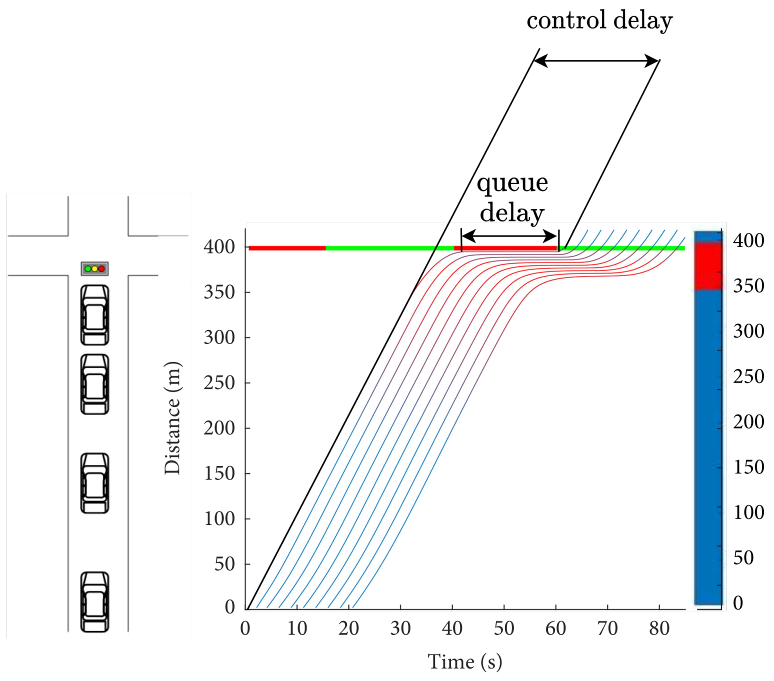


Figure 4-7: Space time diagram indicating control and queue delay. Figure adapted from [85].

4-2-2 Decentralised MPC Problem

The developed prediction model, constraints and objective function is summarised in the following optimisation problem of intersection i.

$$\min_{\mathbf{u}_i(k)} \quad J(\mathbf{x}_i(k), \mathbf{u}_i(k)) = \sum_{k=0}^{N_p-1} c_i^\top n_i^q(k) \tag{4-54a}$$

$$\text{s.t.} \quad x_i(k+1) = A_i x_i(k) + B_{1,i} u_i(k) + B_{2,i}\delta_i(k) + B_{3,i} z_i(k) + B_{4,i} v_{i,j}(k), \quad \forall\, k \tag{4-54b}$$

$$E_{1,i} x_i(k) + E_{2,i} u_i(k) + E_{3,i}\delta_i(k) + E_{4,i} z_i(k) + E_{5,i} v_{j,i}(k) \leqslant E_{5,i}, \quad \forall\, k \tag{4-54c}$$

$$\mathbf{E}_{6,i}\mathbf{u}_i(k) \leqslant \mathbf{E}_{7,i} \tag{4-54d}$$

with appropriately defined matrices and state variable $x_i(k) = \left[n_i^q(k), n_i^d(k)\right]^\top$ and $u_i(k), \delta_i(k) \in \{0,1\}^{|\mathcal{S}_i|}, z_i(k) \in \mathbb{R}_+^{|\mathcal{S}_i|}, x_i(k) \in \mathbb{R}_+^{2|\mathcal{S}_i|}$ and control inputs $\mathbf{u}_i(k) = [u_i(0) \ldots\ u_i(N_p-1)\ ]^\top$. All other bold variable are also defined over the prediction horizon in the same manner as $\mathbf{u}_i(k)$. By combining the variables, the above problem formulated as an MILP problem with appropriately defined matrices,

$$\min_{\mathbf{u}_i(k)} \quad \mathbf{c}_i^\top \boldsymbol{n}_i^{\mathrm{q}}(k) \tag{4-55a}$$

$$\text{s.t.} \quad \mathbf{F}_i \boldsymbol{\xi}_i\,(k) \leq \mathbf{G}_i \tag{4-55b}$$

with $\boldsymbol{\xi}_i(k) = \Big[\ \underbrace{\mathbf{u}_i(k)}_{\text{Control variables}}\ ,\underbrace{\mathbf{n}_i^q(k), \mathbf{n}_i^d(k),}_{\text{State variables}}\ \underbrace{\boldsymbol{\delta}_i(k)\ \mathbf{z}_i(k)}_{\text{Auxiliary variables}}\ ,\ \underbrace{\hat{\mathbf{n}}_i^a(k)}_{\text{Exogenous inputs}}\ \Big]^\top.$

Chapter 5

Distributed Model Predictive Control

This chapter directly starts in section 5-1 with the controller design, since the Distributed Model Predictive Controller (DiMPC) is based on the bottom-up approach, no added modelling step is required. In the controller design the objective function is formulated, in subsection 5-1-1. Thereafter, in subsection 5-1-2, the resulting distributed control problem formulation including the optimisation objective, prediction model, and constraints is presented, as well as the distributed coordination algorithm.

5-1 Controller Design

In subsection 5-1-1, the objective function of the Decentralized Model Predictive Controller (DeMPC) is extended for use in a distributed controller. Next, subsection 5-1-2, the extended objective function is used in the distributed Model Predictive Control (MPC) problem formulation, together with the Mixed Logical Dynamical (MLD) and constraints developed in chapter 4. The design of the DiMPC and its coordination algorithm is based upon the paper by M. A. Müller et al. [86], where a cooperative controller is developed for dynamically decoupled systems with coupled cost or constraints. This fits the current use case, because the DeMPC is not dynamically coupled to its neighbouring intersections. However, there is an implicit coupling through the arrival predictions made by the Long Short-Term Memory (LSTM) and linear arrival prediction methods.

5-1-1 Objective Function

If the DeMPC would be deployed at each intersection in a network, each controller would minimise its own delay. However, in a network of intersections, the objective is to minimise the combined overall delay. This must be reflected in the design of the controller and therefore, the objective function of the decentralised controller is extended. Recall, the objective function of the decentralised controller for intersection i is given by

$$J_i\left(\mathbf{x}_i(k), \mathbf{u}_i(k)\right) = \sum_{k=0}^{N_\mathrm{p}-1} c_i^\top n_i^\mathrm{q}(k), \tag{5-1}$$

with prediction horizon length N_p, discrete time step $k \in [0, N_\mathrm{p}-1]$, $c_i = [\; c_{i,1} \;\; \ldots \;\; c_{i,m_i} \;]^\top$ and $n_i^\mathrm{q}(k) = [\; n_{i,1}^\mathrm{q}(k)) \;\; \ldots \;\; n_{i,m_i}^\mathrm{q}(k) \;]^\top$ denote the weight and state variables with m_i the number of signals of intersection i. Notice that the objective function has subscript i meaning that it takes into account the cost of its own state and signal variables. Therefore, the objective function is dependent only on $\mathbf{x}_i(k)$ and $\mathbf{u}_i(k)$. This objective function is extended to include more information of its neighbouring intersections as follows,

$$J_i^\mathrm{dist}\left(\mathbf{x}_i(k), \mathbf{u}_i(k), \mathbf{x}_j(k), \mathbf{u}_j(k)\right) = J_i\left(\mathbf{x}_i(k), \mathbf{u}_i(k)\right) + \sum_{j \in \mathcal{N}_i} J_{ij}\left(\mathbf{x}_j(k), \mathbf{u}_j(k)\right) \tag{5-2}$$

The distributed objective function is depended not only on the states and signals from its own intersection, but also on the states and signals from all its neighbours, i.e., $\mathbf{x}_j(k), \mathbf{u}_j(k)$, $\forall\; j \in \mathcal{N}_i$, where $\mathcal{N}_i$ represents the set of neighbouring intersections of intersection i. The extended objective function is comprised of the decentralised part, $J_i\left(\mathbf{x}_i(k), \mathbf{u}_i(k)\right)$ and the part belonging to the neighbours $J_{ij}\left(\mathbf{x}_j(k), \mathbf{u}_j(k)\right)$ which contains the states and inputs of one of i's neighbours $j \in \mathcal{N}_i$

$$J_i\left(\mathbf{x}_i(k), \mathbf{u}_i(k)\right) = \sum_{k=0}^{N_p-1} c^\top n_i^q(k) \tag{5-3}$$

$$J_{ij}\left(\mathbf{x}_j(k), \mathbf{u}_j(k)\right) = \sum_{k=0}^{N_p-1} c^\top n_j^q(k) \tag{5-4}$$

5-1-2 Distributed MPC Problem

Combining the objective function above with the model and constraint defined in chapter 4, the distributed control problem $\mathbb{P}_i^\mathrm{dist}$ becomes,

$$\min_{\mathbf{u}_i(k)} \; J_i^\mathrm{dist}(\mathbf{x}_i(k), \mathbf{u}_i(k), \mathbf{x}_j(k), \mathbf{u}_j(k)) \tag{5-5}$$

$$\text{s.t.} \quad x_i(k+1) = A_i x_i(k) + B_{1,i} u_i(k) + B_{2,i}\delta_i(k) + B_{3,i} z_i(k) + B_{4,i} v_{j,i}(k), \quad \forall\, k \tag{5-6}$$

$$E_{1,i} x_i(k) + E_{2,i} u_i(k) + E_{3,i}\delta_i(k) + E_{4,i} z_i(k) + E_{5,i} v_{j,i}(k) \leqslant E_{5,i}, \quad \forall\, k \tag{5-7}$$

$$\mathbf{E}_{6,i}\mathbf{u}_i(k) \leqslant \mathbf{E}_{7,i} \tag{5-8}$$

The distributed controller is thus not dynamically coupled, since the decentralised model is used, but it is coupled through the objective function. The above problem will be solved multiple times during each time step to achieve coordination between intersections. Below the coordination algorithm is explained and summarised.

First, at the beginning of a new time step $k \in [kT_c, (k+1)T_c)$, the initial conditions are set for the states, $x_i(0)$, signals, $u_i(0)$, the exogenous inputs $v_{ji}(k)\ \forall\ k$, which are the predicted arrivals over the prediction horizon, the number of iterations l and solution time of

the distributed coordination algorithm t^{sol} in (a). Then, in step (b), the extended part of the cost function, J_{ij} is calculated using the received queue lengths from all neighbours $j \in \mathcal{N}_i$, resulting from the most recent optimised signals and states of all neighbours. Next, all information is obtained to solve the distributed problem $\mathbb{P}_i^{\text{dist}}$ in (c). After solving the optimisation problem, the resulting planned queue lengths $\mathbf{n}_i^{q,l}(k)$, obtained from the optimised signals, are send to i's neighbours in step (d). Thereafter, the iteration count is updated in (e) and then step (b) is executed again.
These iterations are performed until the total solution time (including iterations) of the distributed controller, t^{sol} is larger or equal than the limit on the solution time t^{lim} or the objective function of iteration l does not change compared to the previous iteration $l-1$. If one of these criteria is satisfied, the signals of the first time step of the last iteration $\mathbf{u}_i^{*l}(1)$ are set to the desired signals of the traffic system $u_i^{\text{sys}}(k)$. Finally, the time step counter is updated and during the next time step the same coordination algorithm is performed. At the same time the other controllers in the network also perform the distributed coordination algorithm.
As mentioned, the coordination algorithm is inspired upon M. A. Müller et al. [86] and is here first applied to Traffic Signal Control (TSC). The algorithm features iterative and parallel computations with asynchronous timing of the communication. These types of multi-agent features can be seen in Figure 5-1, where the arrows indicate the communication of variables and the dotted lines indicate the implementation of the signal to the system. The black bold line indicates that the agent is busy optimising.

Distributed coordination algorithm of intersection i

```
1. While   $t^{\text{sol}} < t^{\text{lim}}$ or $J_i^{\text{dist},l} \neq J_i^{\text{dist},l-1}$:
    (a) Set       $x_i(0), u_i(0), v_{j,i}(k)\ \forall\, k, l = 0, t^{\text{sol}} = 0$
    (b) Receive   $J_{ij}^l \left(\mathbf{x}_j^l(k), \mathbf{u}_j^l(k)\right) = \mathbf{n}_j^{q,l}(k) \quad \forall\, j \in \mathcal{N}_i$
    (c) Solve     $\mathbb{P}_i^{\text{dist}} \to \mathbf{x}_i^{*l}(k), \mathbf{u}_i^{*l}(k)$
    (d) Send      $J_j^l \left(\mathbf{x}_i^{*l}(k), \mathbf{u}_i^{*l}(k)\right) = \mathbf{n}_i^{q,l}(k) \quad \forall\, j \in \mathcal{N}_i$
    (e) Set       $l = l + 1$ and go to (b).
2. Set   $u_i^{\text{sys}}(k) = \mathbf{u}_i^{*l}(1)$
3. Set   $k = k + 1$ and go to 1.
```

The limit on the solution time should be set such that after that there is enough time to implement the signal. A good starting point for this is 0.3 seconds, since this is the time it takes to implement a signal to the traffic system in the worst case scenario [78]. The implementation of the DiMPC will be discussed in section 6-2.

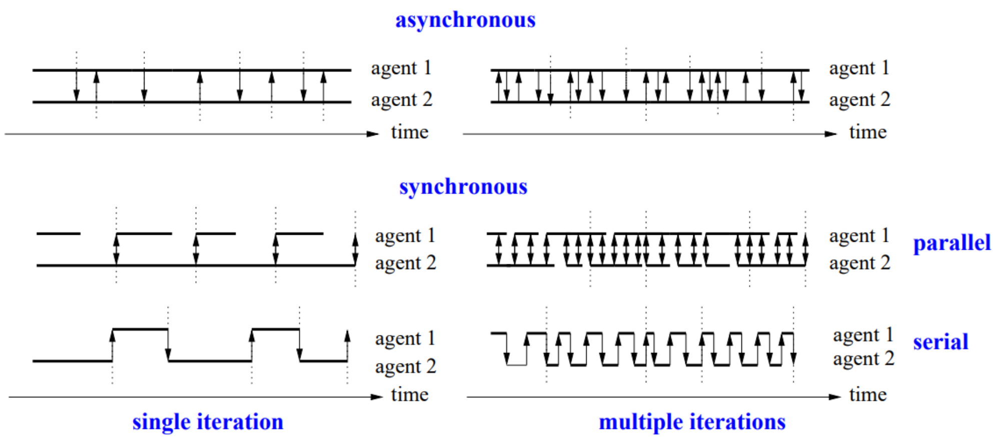


Figure 5-1: Computation, communication and implementation types between two agents. Source [53].

Chapter 6

Simulation Experiments

The simulation experiments will first describe the simulation design, implementation, and parameter settings of the decentralised controller for a single arterial intersection in section 6-1. In section 6-2, the same topics will be covered for the simulation of the distributed and decentralised controllers applied to an arterial network.

6-1 Arterial Intersection

First, in subsection 6-1-1, the intersection, simulation scenarios, and control system will be discussed. Thereafter, in subsection 6-1-2, the parameters of the controller are determined. Finally an overview of the simulated scenarios is given in subsection 6-1-3.

6-1-1 Simulation Setup and Implementation

The design of the simulation experiments entails the modelling of the intersection in a simulation network, determining the simulation scenario and traffic input demand, as well as the implementation of the control system.

Simulated Network

Since the controller is developed for on-street application, it will be evaluated in a simulation network that models a real-world network. The chosen intersection, named 201234, is located in an arterial with multiple intersections between 'Drie Merenweg' and the provincial highway N201 in North-Holland. Intersection 201234 has three neighbouring intersections, namely, 201231, 201239, and 201295, located 315, 340, and 580 meters from intersection 201234, respectively. The simulated network of the intersection and its neighbours can be seen in Figure 6-1. The configuration and topology of the intersections were provided by the province of North-Holland. The controlled intersection 201234 is depicted in more detail in Figure 6-2, where the topology of the intersection including the vehicle signal group identification

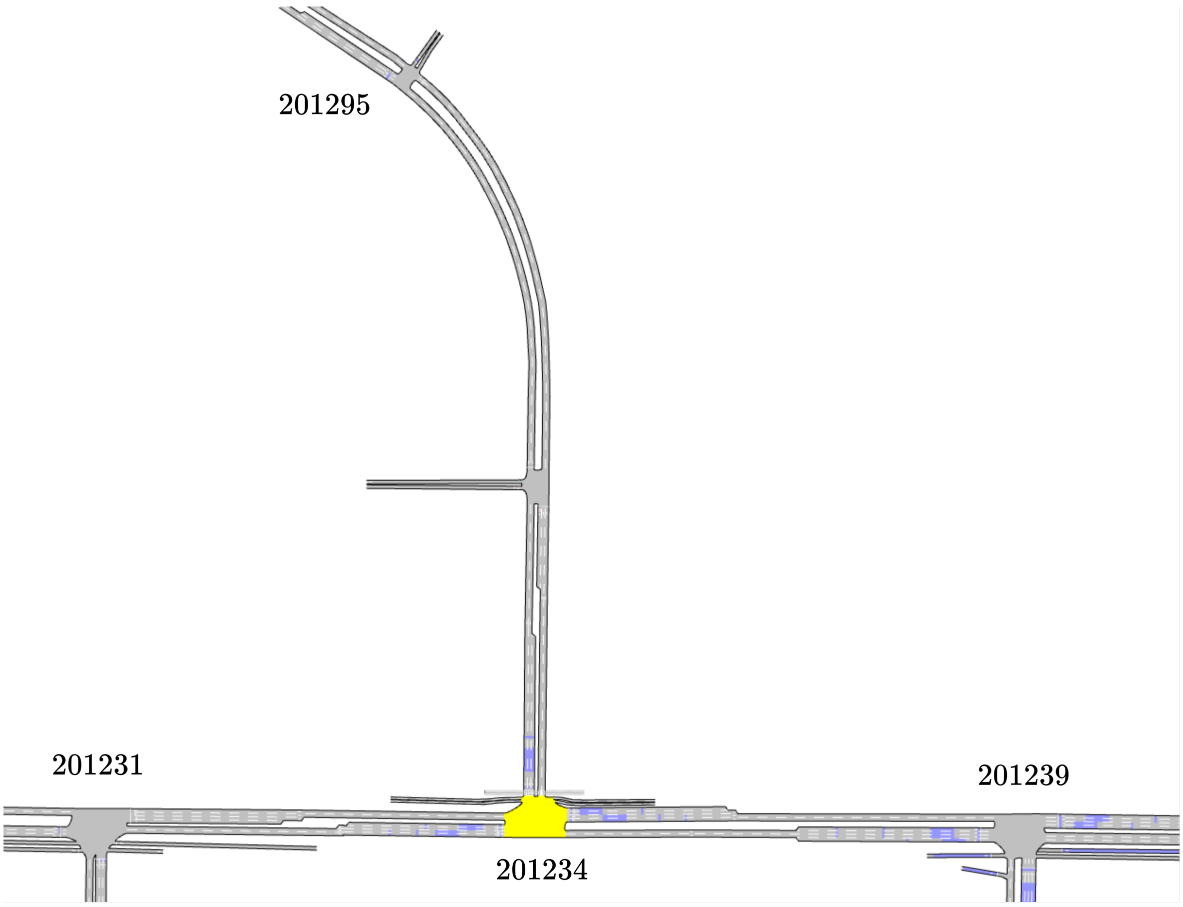


Figure 6-1: Simulation network of intersection 201234 and neighbouring intersections 201231, 201239 and 201295 on N201 in North-Holland.

numbers, and detector locations are also shown. The bike lanes and pedestrian crossings are also modelled, but these are not used during the performed simulations. It should be noted that intersection 201234 has double lane arrival detectors for all signal groups. These detect around 10% less vehicles, because when two vehicles pass the double lane detector next to each other, only one vehicle instead of two is detected by the inductive loop covering two lanes [62]. More information about the modelling of the network and the parameter calibration of the microscopic simulation model can be found in the thesis of T. Glastra [62].

Simulation Scenario and Traffic Demand

To reduce the difference between the real-world and the simulation even more, the date and time of the simulation scenario and the input traffic flow demand will be based on real-world data. Data sets were provided by the province of North-Holland, which contain detector measurements and signal timings in V-Log protocol recorded in 2019 [19]. The simulation control system is run in real-time. Therefore, it is not achievable within the project time frame to simulate for multiple days or longer periods. For that reason, only one simulation date will be selected. The chosen date of the simulation scenario is Tuesday 15th of October 2019. Firstly, a weekday is selected as the traffic flow during the week includes every traffic

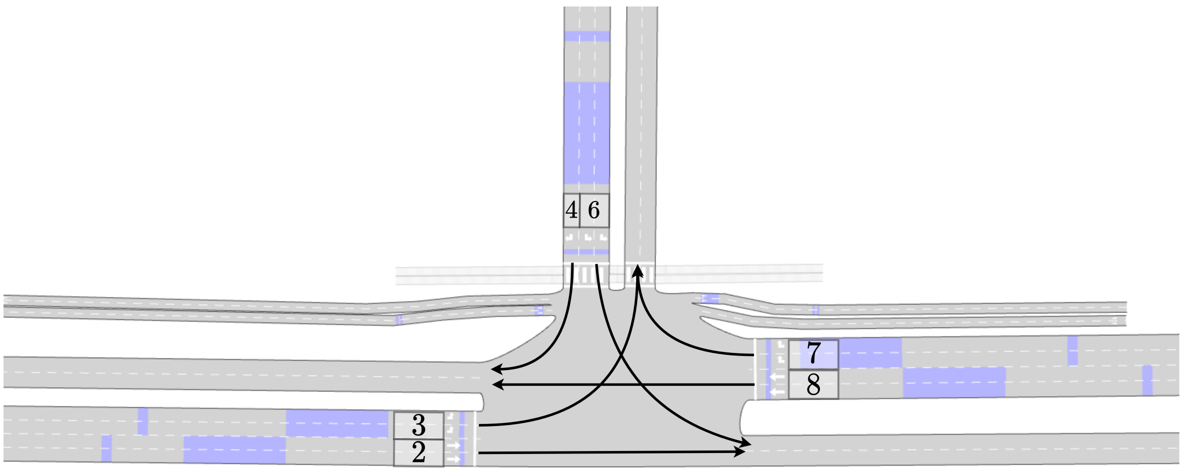


Figure 6-2: Topology, vehicle signal groups identification numbers and detectors of real-world intersection in North-Holland modelled in Aimsun [87].

condition, whereas during the weekends traffic flow is much lower. Next to that, the date is chosen such that the traffic flow at intersection 201234 is representative for most weekdays. This can be seen in Figure 6-3, where the mean and 95% confidence interval of the traffic flow at the stopline detectors of intersection 201234 is shown during the weekdays of the three weeks after 15/10/2015. It can be seen that the simulation date is thus representative of the traffic flow during the period of October 2019. Each simulation will be run from 4 am until 10 am. During this period the traffic condition transitions from undersaturated up until saturated, including the morning peak. The traffic input flow to intersection 201234 can be seen in Figure 6-4 for every signal group.

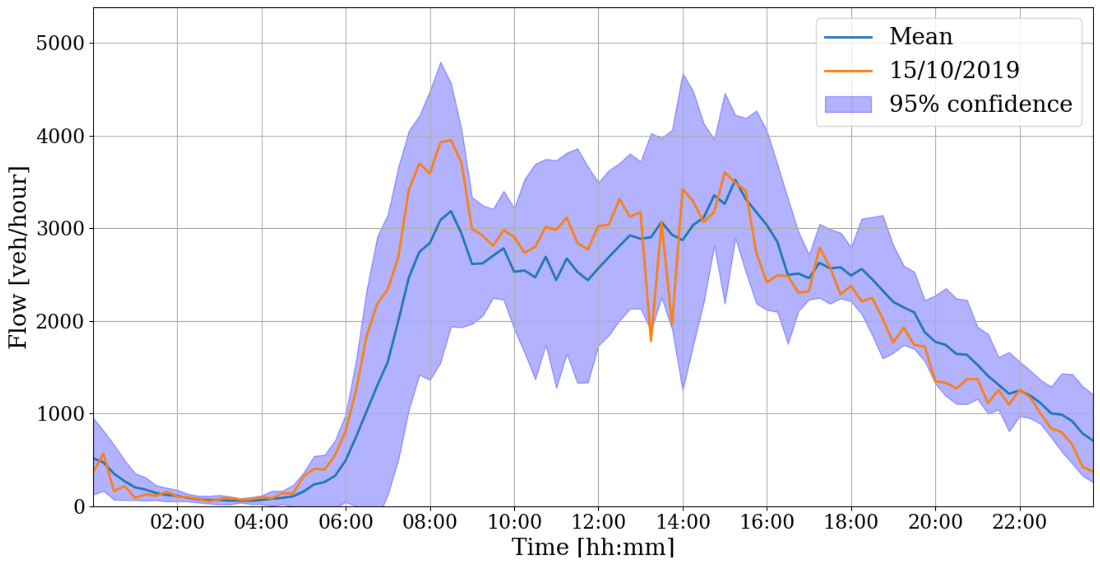


Figure 6-3: Traffic flow at stopline detectors of intersection 201234 for 15/10/2019 and mean and 95% confidence interval of the workdays three weeks thereafter in local time (UTC+2).

The traffic demand sets the input flow for the simulation scenario. This can be done via estimation of Origin-Destination (OD)-matrices for larger networks with scattered detector data or via traffic states obtained from detector measurements [38]. Since the network is small with no traffic sinks or sources present and detector data is made available by the province of North-Holland, real-world upstream detections will be used as input for the simulation. In this case, vehicles will be generated at the speed limit of 80 km/h at the location of the stoplines of the neighbouring intersection's signal groups leading to intersection 201234. The generated vehicles are given a destination, which is determined using historical detector data at the controlled intersection 201234.[1] For example, when a vehicle is generated at intersection 201231, the future arrival detections from the real-world data set are evaluated to see if that vehicle arrives at signal group 7 or 8. The vehicle is then given a right turn or a straight ahead destination, respectively. If the destination cannot be determined, then the destination is based on a random choice function with a probability equal to the turning percentage, which is calculated as a moving average during the simulation.

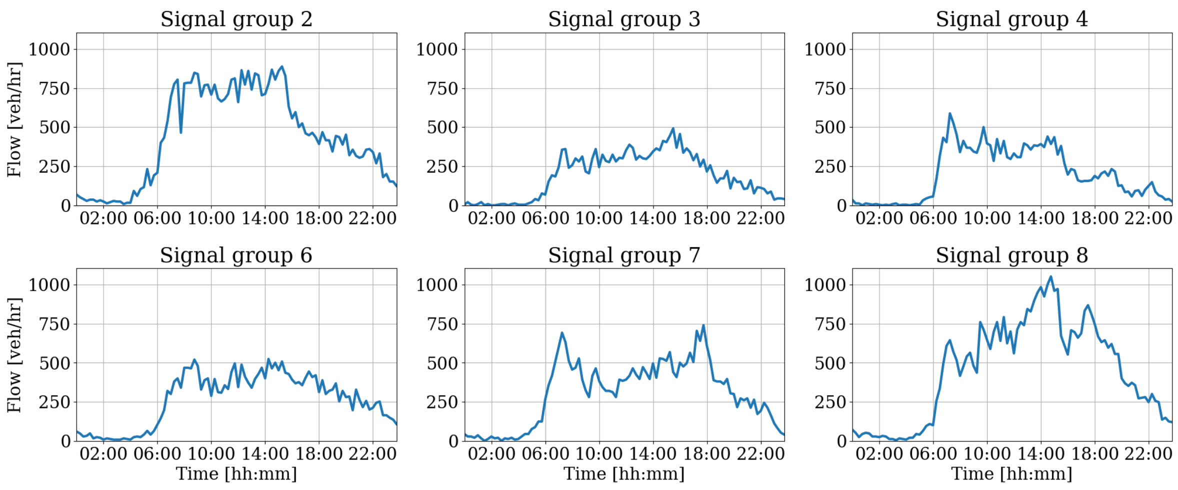


Figure 6-4: Traffic demand input of intersection 201234 on 15/10/2019 in local time (UTC+2).

Simulation Control System

A simplified schematic overview of the control system used for simulation is shown in Figure 6-5. In this scheme, all blocks are separate collections of programs that receive and send messages via a cloud messaging broker, named RabbitMQ, and CVNIP communication. Both communication tools are explained in Appendix B. The dashed arrows indicate these communicated messages and are also labelled by a qualitative description of the important variables in the message sent between two blocks.
As mentioned, the input detections are played back in real-time from a historical data set.

[1]It should be noted that in the used simulation program, called Aimsun, the implementation of the traffic demand input from real-world detections is not done via traffic states, but via OD-matrices, where every entry from origin to destination is initialised with 1 vehicle input. Vehicles are put into the simulation at the correct time via an external Aismun API. However, using the OD-matrices, the destination of each vehicle must also be known. Therefore, during vehicle generation, an origin and a destination must be assigned to each vehicle.

Based upon these detections, vehicles are generated in the simulation at the location of stoplines of the neighbouring intersections. The raw detections of these vehicles in the simulation network are communicated to the intelligent Traffic Light Controller (iTLC). The raw departure, arrival, and queue detections are aggregated per control time step, meaning these detections are gathered by the aggregation program and released at the time of the next control time step. Using this new detector information, the input matrix of the Long Short-Term Memory (LSTM) neural network is updated and future arrivals $n^a(k)$ for $k \in [0, N_\mathrm{p}]$ are predicted. At the same time, the number of vehicles in the queues is estimated using a rule-based approach. The arrivals, queues and the actual current signal, received form the iTLC, are then all set as initial conditions of the optimisation problem. Once these initial conditions are set, the optimisation process is executed. When the optimisation is finished, the optimised inputs of the first time step $u(1)$ are sent to a finite state machine that handles the switching of the signals and checks the safety of these desired signals. Once the signals are released at the correct time by the finite state machine, these desired signals are sent to the iTLC.

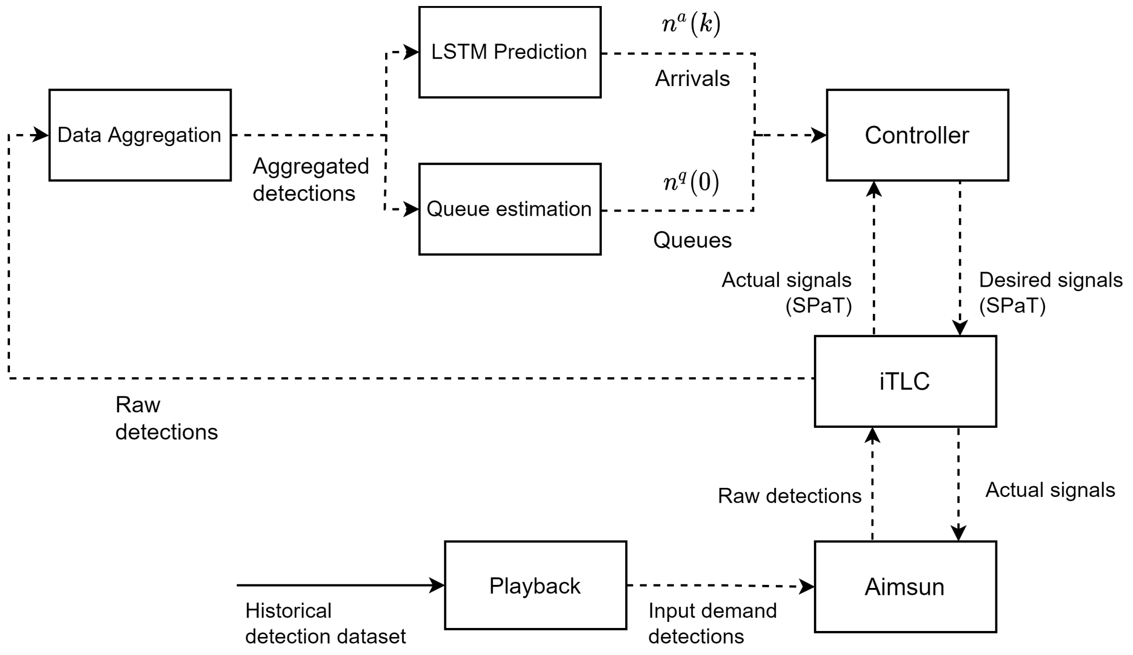


Figure 6-5: Simulation control system.

The iTLC consists of the Traffic Light Controller (TLC) hardware rack shown in Appendix C in Figure C-1 and also a virtual machine, called mobility data broker, which receives, reformulates, and sends all the detection and Signal Phase and Timing (SPaT) messages. The traffic light control hardware rack contains a real on-street traffic light control interface, which has a safeguard mechanism that checks and monitors the signals and the state of the active control software, i.e., the control method that is *in control* of the intersection. If an unsafe signal is sent to the TLC, it responds by sending an error message and throwing the current active control software *out of control*. This means that the TLC returns to controlling the intersection using the backup actuated control method.

The control system is run in real-time, meaning that each 6 hour simulation takes the same amount to run. Communication via RabbitMQ takes on average 10 milliseconds and the implementation of a desired signal sent to the TLC takes between 200 and 300 milliseconds. The simulation control system is developed by colleagues at Yunex Traffic and previous interns. However, to switch the control method, many other programs needed to be changed or reconfigured. For a more detailed view of the control system, refer to Appendix B and for more information about the queue estimation and data aggregation, refer to the thesis of T. Glastra [62]

6-1-2 Controller Parameters

In this subsection, the conflict matrix, clearance times, and other parameters of the intersection are set. Thereafter, the control time step, prediction horizon, and model paratmers of the controller are also determined.

Intersection Parameters

The intersection parameters corresponding to intersection 201234 are provided by the province of North-Holland. Firstly, the minimum clearance time $t^c_{r,s}$ for every ordered pair of conflicting signal groups $(r, s) \in \mathcal{C}_o$ is set to the values summarised in the conflict matrix of Figure 6-6. The maximum wait time is set to 120 seconds and the minimum red and green times are set to 2 seconds and 4 seconds, respectively. Lastly, for signal groups 2 and 8, the amber time is equal to 4 seconds, while for signal groups 3, 4, 6, and 7 the amber time is 3 seconds.

s	2	3	4	6	7	8
2		-	-	32	-	-
3	-		-	25	53	39
4	-	-		-	-	14
6	32	29	-		-	16
7	-	7	-	-		-
8	-	5	5	1	-	

Figure 6-6: Conflict matrix of intersection 201234, including clearance times in deciseconds.

Control Time Step and Prediction Horizon

The control parameters that need to be set are the control time step k_c, the prediction horizon of the MLD model N_p and the weights c_i of the objective function. The objective function weights are all set to one and thus every queue is weighted equally. The control time step must be small enough to grant fast responsiveness to changing situations and mismatches between the prediction model and the real world. At the same time, the control time step must be large enough for the overall computation time that is needed for communication, optimisation, and implementation of the signals. In general, the prediction horizon should be long enough to capture the effects of the optimised signal timings on the traffic network.

Specifically, it is recommended that the value of the prediction horizon should be equal to the time it takes for a vehicle to cross the network [38]. Additionally, the control time step and the prediction horizon are linked. By decreasing the control time step or increasing the prediction horizon, the size of the to be optimised model grows, as does the computation time of the optimisation algorithm.
In this case, the distance that a vehicle needs to travel to cross the network is equal to the distance from the stopline of the preceding intersection to the point it leaves the simulation just after the controlled intersection. Additionally, the LSTM network predicts on the basis of detections at the neighbouring intersection from 100 seconds ago up until the current time step. Therefore, it can only predict with acceptable accuracy for a horizon that is equal to the average travel time of a vehicle on the link connecting the preceding intersection and the controlled intersection. Considering the above, the prediction horizon is set to the free-flow travel time of the longest link plus the average delay of the intersection,

$$N_{\mathrm{p}} = \frac{TT_{\mathrm{link}}^{\mathrm{free}} + D_{\mathrm{average}}}{T_{\mathrm{c}}}, \tag{6-1}$$

where $TT_{\mathrm{link}}^{\mathrm{free}}$ is the link free-flow travel time based on the speed limit of 80 km/h and D_{average} is the average vehicle delay, obtained from previous experiments by T. Glastra [62]. In the simulation network of Figure 6-1, the longest link is the link between intersections 201295 and 201234. The resulting prediction horizon is 40 seconds, determined from a free-flow travel time equal to 26,1 seconds and average delay equal to 15 seconds per vehicle [62]. This should be long enough enough to incorporate all the accurate predictions made by the LSTM and the affects of the signal timings on traffic network. The prediction horizon is long enough for Green Light Optimised Speed Advice (GLOSA) implementation, since GLOSA has an effect on the stop time and traffic flow from a distance of 300 meters [77].

The shorter the control time step, the faster the controller can react to mismatches between the model and the real world. More importantly, the operation of scheduling traffic signals becomes more efficient as the signals can switch sooner to green and red when necessary, saving time that would otherwise be lost with a larger control time step. Therefore, the control time step of 5 and 2 seconds is chosen. A control time step of 5 seconds is frequently used in literature and allows for enough room for the iterations needed in the coordination of the Distributed Model Predictive Controller (DiMPC) [88]. A control of two seconds is mainly chosen to investigate the effect of a larger model on the computation time. The effect of a smaller control time step on the delay performance was already investigated by Katwijk [57].

Lastly, it should be noted that in a formal controller design process, the control time step and the prediction horizon should be tuned for several values to make a trade-off between computation time, prediction horizon, and responsiveness of the controller. However, as mentioned above, the maximum prediction horizon is chosen and in this case the computation time is not large enough to be of importance in this trade-off. More importantly, when tuning these parameters, a lot of simulations need to be made, i.e., for every combination of the control time step and prediction horizon. Since the simulation system runs in real-time and simulation scenarios can only be simulated during the day, formal tuning is not feasible within the project time frame.

Saturation Flow Rate

The only parameter that needs to be set in the queue model is the discharge saturation flow rate μ_s^{sat}, which is the maximum flow rate that can be achieved by a saturated stable moving queue of vehicles passing over the stopline. The fact that only one parameter needs to be set is beneficial for the deployment of the traffic controller, because road owners already know the value of this well-known parameter. If not, an estimation via stopline detector data can be carried out.
Parameters in a model can be set by using optimal parameter estimation methods, where the optimal parameters minimise the difference between the measured departures and the departures predicted by the model [38]. In literature, the saturation flow rate for calculating departures on a small time step scale (seconds) is mostly set to a standard value obtained from a table, a simple formula from the Highway Capacity Manual (HCM) [89] or is assumed to be known [13, 59]. For larger time step scales, optimal parameter estimation is used. Therefore, since detector and signal data are available, the discharge saturation flow rate is set to be equal to the average discharge saturation headway rate corresponding to the fifth position in the queue [13]. Then, the discharge saturation flow rate is calculated using the conversion relation between headway and flow rate in Table 2-1. In Figure 6-7, a box plot is shown of the discharge headway per queue position for the two stopline detectors of signal group 6 at intersection 201234. This is based on data gathered from workdays three weeks after 15/10/2019. The headways are calculated from detection data where only detections during green signals are taken into account and where there were at least 5 vehicles standing in the queue of the signal group. The discharge headway of the other signal groups of intersection 201234 is presented in Appendix C. The determined average saturation headway per lane h^{sat} and its corresponding discharge saturation flow rate μ_s^{sat} for all singal groups are shown in Table 6-1.

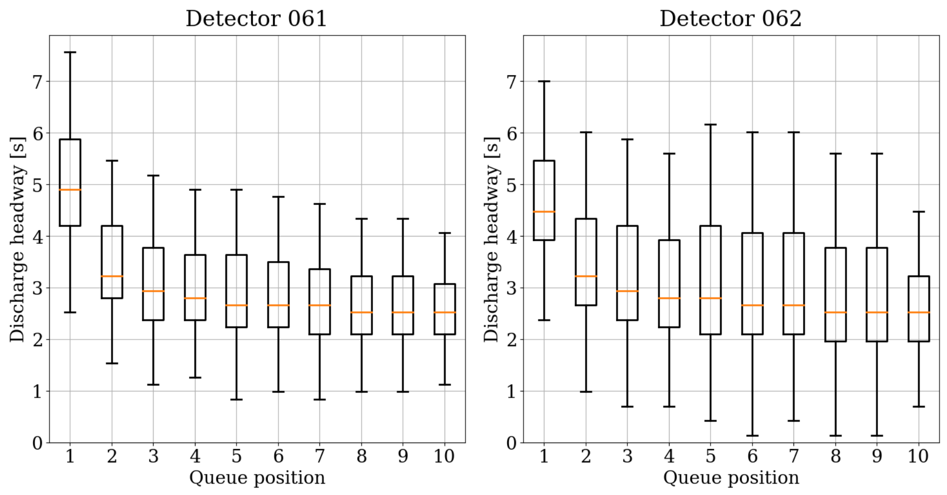


Figure 6-7: Discharge headways at stopline detectors of signal group 6. Based on data from the workdays three weeks after 15/10/2019.

Table 6-1: Average discharge saturation headway h^{sat} per lane and discharge saturation flow rate μ_s^{sat} per signal group s of intersection 201234.

Signal group	h^{sat} [s]	μ_s^{sat} [veh/h]
2	2.12	3396
3	2.63	2738
4	2.37	1518
6	2.75	2618
7	2.7	2667
8	2.61	2759

Turning Percentage

In the definition of the arrival prediction model in Equation 4-3, the turning percentage is used to assign arrivals to a signal group. These turning percentages p_s^{turn} are dynamically set every 15 minutes to the average turning percentage during the three weeks after the chosen simulation date. The turning percentages can be seen in Figure 6-8 where the mean over this period is shown together with the 95% confidence and turning percentages during the simulation date. From this figure, it can be seen that the turning percentages vary greatly from midnight until around 06:00. From then on, the turning percentages are fairly close to each other between different days.

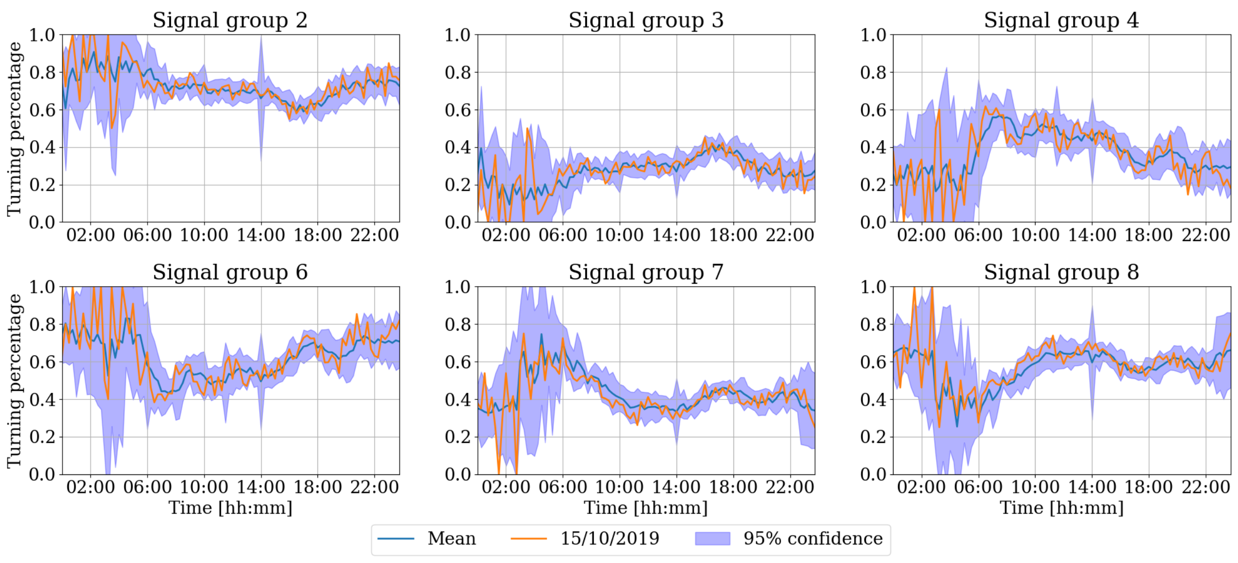


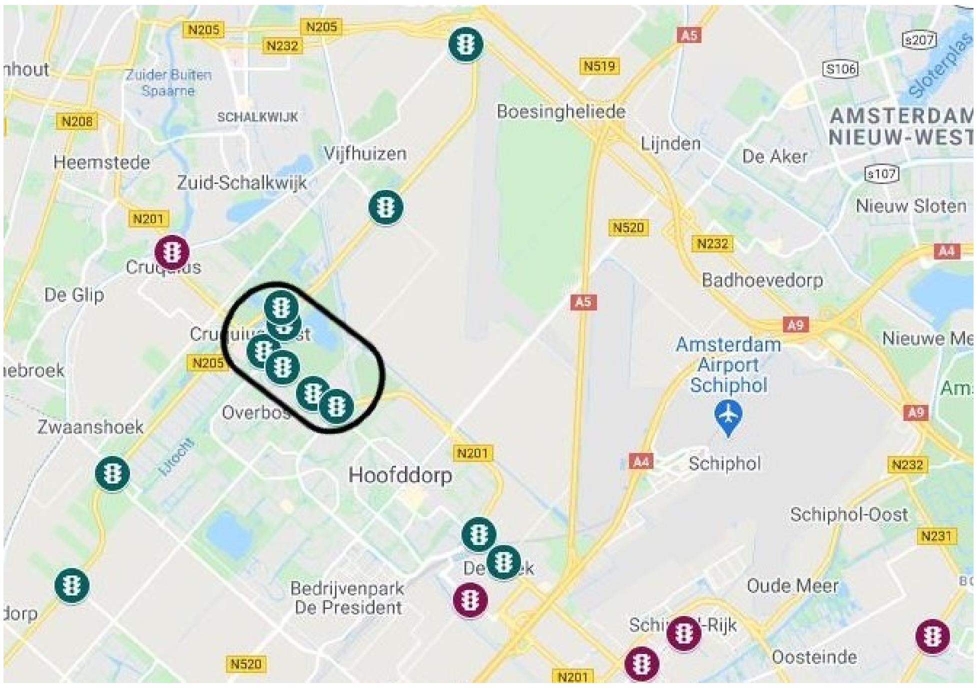
Figure 6-8: Turning percentage at stopline of intersection 201234 on 15/10/2019 and mean and 95% confidence interval of the workdays three weeks thereafter in local time (UTC+2).

6-1-3 Overview of Simulated Scenarios and Control Parameters

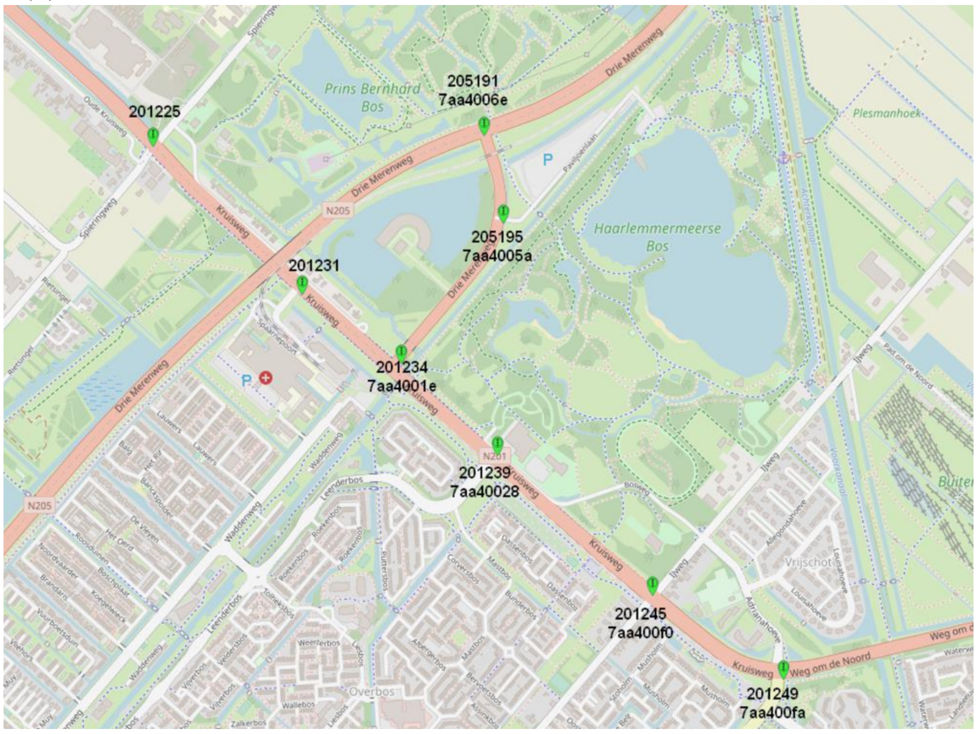
In Table 6-2, all the simulated scenarios are shown with their configured time step and prediction horizon, which is expressed in the number of time steps.
The first scenario of the arterial intersection network is simulated with the actuated controller, formally called CCOL, which is the standard traffic controller in the Netherlands. The control method is explained in section 3-1-1 and in greater detail in the appendix of the thesis by N. Helmy [63]. DIRECTOR is Yunex's current method developed by J.C. Van Senden [64] and based upon the self-organising controller from literature [73, 74]. DIRECTOR is briefly explained in section 3-4-2 and in Appendix A. The Decentralized Model Predictive Controller (DeMPC) is the control method developed in chapter 4. Since not all combinations of control methods, prediction methods and time steps could be simulated due to time constraints, it is opted to simulated the DeMPC with LSTM and linear arrival prediction methods and with a 5 second control time step. Next to that, to give an insight into the solution time of the optimisation problem, the DeMPC was also simulated with a 2 second control time step and LSTM prediction, because the LSTM prediction method yielded vastly better performance in the prediction method comparison.
All the LSTM models were trained on the detection and signal data from workdays three weeks after 21/10/2019 with a 20-80 test and training data set split. The simulation date and time of all scenarios is 15/10/2019 from 4 am to 10 am. The used data sets for training and testing of the prediction model and for generating the simulation input demand do not contain any detector faults. Five replications or iterations were run per simulation scenario, each having a different random seed for car/driver parameters, to achieve representative results. As mentioned, the control system and simulation were run in real-time, thus each replication of a scenario took 6 hours to run. The simulations were performed using Aimsun simulation software [87]. The Mixed Logical Dynamical (MLD) model was implemented in Python using the optimisation modelling language Pyomo [90] and the Mixed-Integer Linear Programming (MILP) optimisation problem was solved using CPLEX, where the branch-and-bound solution algorithm is used in combination with dynamic heuristic tree searching strategies [91]. The simulation control system (simulation, optimisation, controller, aggregation and prediction programs) were run on a laptop with CPU Inter(R) Core(TM) i5-8365u CPU @ 1.60 GHz with 4 cores and 8 logical processors.

Table 6-2: Simulation scenarios for arterial intersection 201234.

Scenario name	Prediction method	k_c	N_p
Actuated	-	0.1	-
DIRECTOR	LSTM	5	8
DeMPC	LSTM	5	8
DeMPC	Linear	5	8
DeMPC	LSTM	2	20



(a) Location of arterial network on N201 in North-Holland, The Netherlands. Source [21].



(b) Arterial network and intersection identification numbers. Source [78]

Figure 6-9: Location of arterial network and its intersections.

6-2 Arterial Network

In this section, the simulation network will be extended to an arterial network with multiple intersections. In subsection 6-2-1, the simulated arterial network is set up. This entails the modelling of the traffic demand and the implementation into the simulation control system. Thereafter, in subsection 6-2-2, the controller parameters are discussed. In the end, an overview of the proposed simulations is given in subsection 6-2-3.

6-2-1 Simulation Setup and Implementation

The design of the simulation experiments requires the modelling of the arterial network, determining the simulation scenario and traffic input demand, along with extending the control system.

Simulated Arterial Network

The network is the arterial where intersection 201234 is situated. It is located in Hoofddorp, which lies South-West of Amsterdam, close to Schiphol Airport. The exact location and intersection identification numbers within the arterial network can be seen in Figure 6-9. The simulated network in Figure 6-1 already contained most of the intersections in the arterial, except intersection 201225. This intersection was added using overlay DWG mapping files provided by the province of North-Holland. The detectors of all intersections were also placed in the simulation model together with their corresponding identification number. In addition, the signal group configuration, intersection IDs and adaptible signal plans of all intersections were also configured. The resulting simulation network can be seen in Figure 6-11 and a schematic view of the intersections and their signal groups is presented in Figure 6-12.

Traffic Scenario and Traffic Demand

The same date and time scenario is chosen as in the arterial intersection simulation described in section 6-1-1. Similarly to the arterial intersection, the input detections of the arterial network are also played back from a historical data set via an external Aimsun API program. The generated vehicles are put into the simulation at the arrival detectors of the approaches at the edge of the network. All vehicles are generated at the speed limit of the specific approach. However, for some approaches at the edge of the network, no arrival detectors are available. In Figure 6-12, the approaches for which no arrival detectors are available can be seen by the absence of a white triangle. In these cases, the input flow is determined by using the detections of the stopline. Specifically, the flow rate at the stopline is aggregated per 15 minutes and set as the input traffic flow in the traffic state of the approach. Vehicles are then generated at the start of the modelled approach.
Therefore, instead of OD-matrices in the arterial intersection simulation model, the input traffic demand is modelled using traffic states in the arterial network simulation. A traffic state is composed of a set of flows at an input approach of the network and a set of turn percentages at every intersection. Traffic states are most often used for small area intersection design or arterial networks where there is comprehensive data available and the emphasis is

laid upon flow optimisation and detailed road design, with no traveller reaction to the traffic conditions [87]. With traffic states, individual vehicles do not have a destination, they choose each turn-based on the turning percentages at the next intersection and traverse the network until they find an exit section.
The turning percentages at each intersection are entered directly into the simulation model via these traffic states. However, these turning percentages vary across the simulation time. Therefore, the turning percentages are analysed at each intersection and aggregated per 15 minutes. These aggregated turning percentages at the stopline detectors, shown in Figure 6-10, are then configured as the turning proportions of the traffic states. The stopline detectors are chosen to determine the turning percentages, because vehicles sometimes decide closer to the intersection in which direction they want to go.

The white triangles in Figure 6-12 indicate that the LSTM model is trained for the input approach. For intersections 201234 and 201249.1, these triangles are accompanied by an intersection number. This means that the approach has a preceding intersection from which the detections can be used as an input to the LSTM in order to predict the future arrivals. Regarding the other input approaches, without preceding intersections, the LSTM is trained only on the time, detection and signal data from the input approach. This is done to predict arrivals for longer-term trends based on the time of day and day of week data, since traffic flow is highly cyclical from day to day.

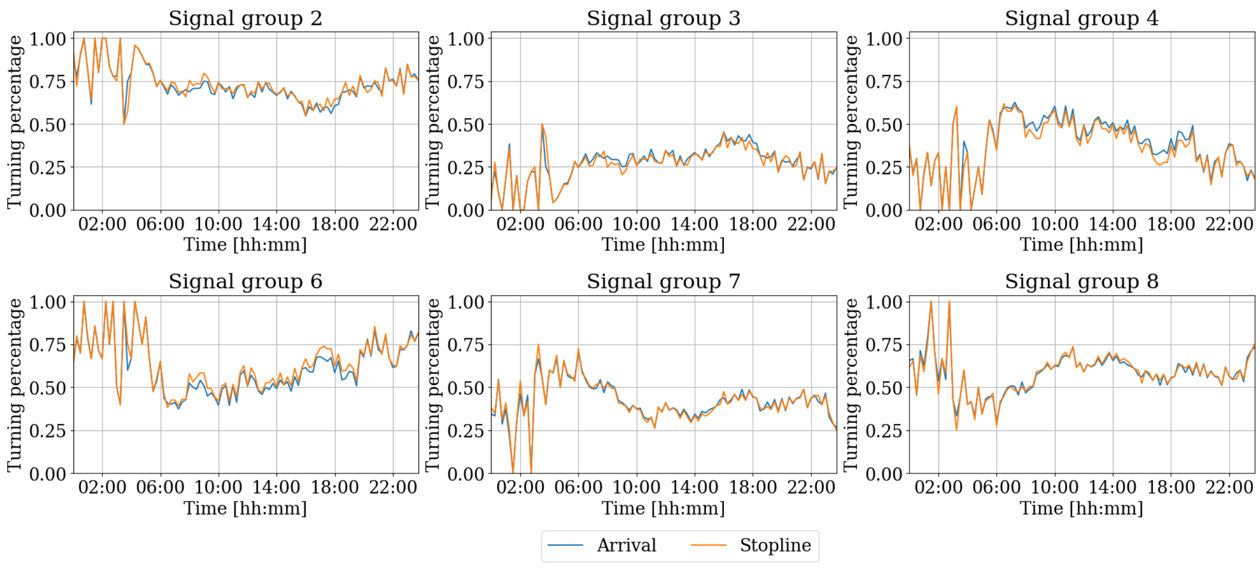


Figure 6-10: Turning percentages at arrival and stopline detectors of intersection 201234 on 15/09/2019.

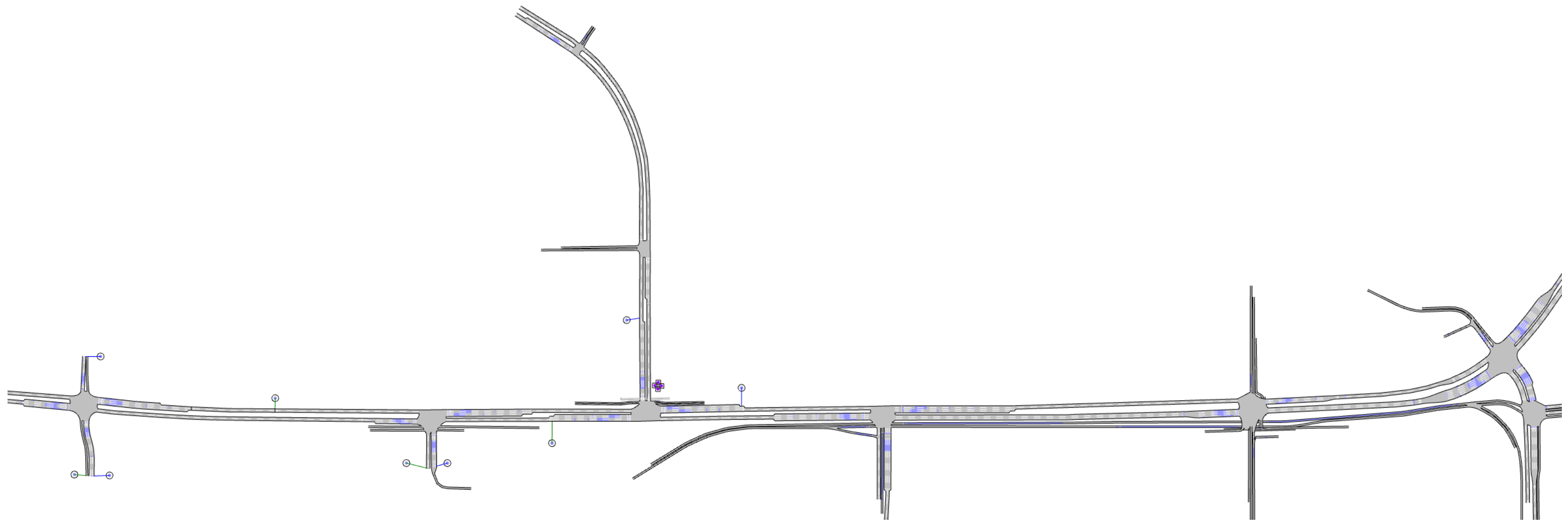


Figure 6-11: Aimsun simulation network of arterial network N201 in North-Holland.

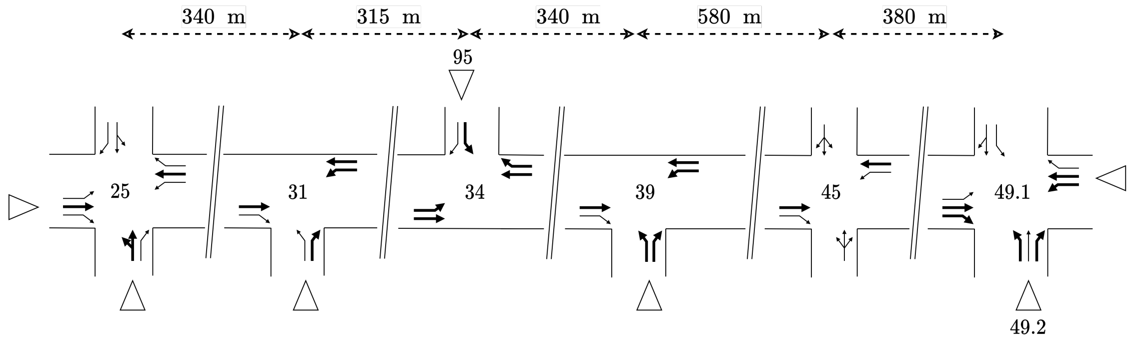


Figure 6-12: Schematic view of intersections and their last two ID numbers in arterial network. The white triangles with intersection ID numbers beside them indicate that at those approaches the arrivals are predicted by the LSTM model based on detections from the preceding intersection. The white triangles without any ID numbers indicate that the LSTM model is trained only on the detections from that approach (see section 6-2-1). The bold arrows indicate signal groups that have two or more lanes, while the other arrows indicate the signal groups with one lane.

Simulation Control System

In Figure 6-13, the distributed control system for simulation is shown, where state and signal variables are send and received to and from the neighbouring intersections. These variables of each intersection are published on one RabbitMQ exchange with the intersection IDs as keys. For a more detailed view of the control system, refer to Appendix B.
To control multiple intersections, different instances with intersection-specific configuration of data aggregation, LSTM prediction and queue estimation, and controller programs are run on a virtual machine. Since these programs were only tested on one simulated intersection before, the prediction program was not robust to different intersection and controller configurations. This was adapted such the control system can now deal with all intersection configurations and the programs within can run with different time steps and prediction horizons. For example, the prediction horizon of the LSTM prediction program can differ from the controller's prediction horizon.

The XML configuration files of all intersections were provided by the province of North-Holland and a program was written to convert these XML configuration files to the Json configuration files used as input of the data aggregation, LSTM prediction, and controller programs. This would further increase the ease of deployment of the controller. The province of North-Holland also provided two extra data sets for intersections 201245 and 201249 containing detection and signal data in V-Log protocol [19]. These files were preprocessed for the input data of the LSTM neural network, which is trained for each intersection in the arterial network. On the simulation side, the configurations of the Playback and Aimsun programs were also reconfigured to input the correct detections for the input approaches of the network. The Aimsun block in Figure 6-13, consist of the Aimsun simulation program and also the Aimsun API program. This API program was also adapted to handle the communication of detections and SPaT messages of multiple intersections. Lastly, since only one hardware rack of the TLC is available, the TLC was simulated using an already developed program within Yunex, called TLCSim. This simulated TLC was adapted and reconfigured for simulation with multiple intersections.

6-2-2 Controller Parameters

The intersection parameters were set in the same way as in section 6-1-2. Similarly, the saturation flow rates of the model are also determined by the approach defined in section 6-1-2. The control time step of the controller is set at 5 seconds. This yields a trade-off between the iterations needed for the distributed coordination algorithm and the responsiveness of the controller. The prediction horizon is set at double the amount decided in section 6-1-2, which is 80 seconds. Since the neighbouring intersection controllers are predictive, the variables, i.e., departures, queues, signals, and arrivals, can be used to predict the arrivals further into the future at the controlled intersection. Therefore, the prediction horizon is set to two times the predicting horizon of the decentralised controller deployed at one intersection of the arterial.

6-2-3 Proposed Simulations

In Table 6-3, the proposed simulation scenarios are presented with their configured time step and prediction horizon, which is expressed in number of time steps. It is not possible

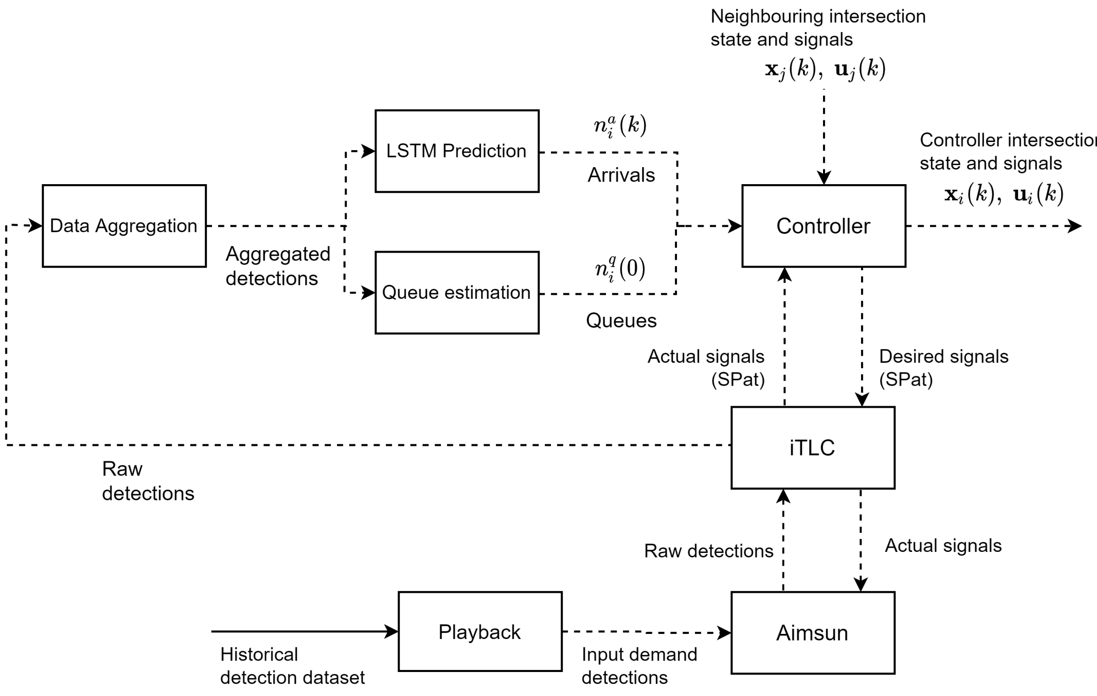


Figure 6-13: Distributed control system for simulation.

to simulate the actuated controller, named CCOL, for multiple intersections, because only one hardware rack (depicted in Figure C-1) is available. The proposed simulations do entail Yunex's current control method DIRECTOR, the newly developed DeMPC as well as DiMPC, developed in chapter 5. All the LSTM models were trained on the detection and signal data from workdays three weeks after 21/09/2019 with a 20-80 test and training data set split. The simulation date and time of all scenarios is 15/10/2019 from 4 am to 10 am.

Unfortunately, the simulation of the arterial network could not be carried out. This is caused by the implementation of the simulation environment. Specifically, the mobility data broker program, depicted in Figure B-1, could not be adapted within the project time frame to receive and send data to multiple intersections in combination with the simulated TLC. The controller also could not be tested offline, since all programs run in real-time, consuming and publishing in real-time from and to specific communication channels.

Table 6-3: Simulation scenarios for arterial network

Scenario name	Prediction method	k_c	N_p
DIRECTOR	LSTM	5	16
DeMPC	LSTM	5	16
DiMPC	LSTM	5	8

Chapter 7

Results

This chapter covers the results of the prediction models and the simulation experiments proposed in subsection 6-1-3. In section 7-1, the prediction accuracy of the arrival prediction models is shown. Thereafter, the focus is laid upon the results from the simulation scenarios in section 7-2.

7-1 Arrival Prediction Models

First, the used performance metric will be explained in subsection 7-1-1. Then, the performance of the prediction models will be evaluated for each approach and signal group over the prediction horizon. The considered models for this evaluation are the Long Short-Term Memory (LSTM) model with 2 and 5 second time steps and the linear delay prediction model with a 5 second time step. The evaluated models all have the same prediction length discussed in subsection 6-1-2.

7-1-1 Performance Indicator

The performance of the arrival prediction models is evaluated using the Normalised Root Mean Square Error (NRMSE) defined as

$$\text{RMSE} = \sqrt{\frac{\sum_{k=1}^{n} \left(y(k) - \hat{y}(k)\right)^2}{n}} \tag{7-1}$$

$$\text{NRMSE} = \frac{\text{RMSE}}{\sigma}, \tag{7-2}$$

where y represents the test data, $\hat{y}$ the output of the prediction model, n the number of data points in the test data set and σ the variance of the test data set. The NRMSE is chosen, because test data sets differ per approach to the intersection and each data set has its own variance. Using the NRMSE, different data sets can be accurately compared. As mentioned, the models are trained for three weeks (only weekdays) after 20/09/2015 with a 80-20 train-test split.

7-1-2 LSTM Model with Different Time Steps

In Figure 7-1, the NRMSE is shown for all three approaching links of intersection 201234. Approach 0 contains signal groups 2 and 3, while approach 1 contains signal groups 4 and 6, and, approach 2 includes signal groups 7 and 8. The configuration of the signal groups is illustrated in Figure 6-2. It can be seen that the prediction model of approach 1 shows a significantly higher NRMSE than the other two approaches. Recall, approach 1 connects intersection 201234 to its furthest away neighbour, namely 20195 with a distance of 580 meters, compared to around 320 meters for the other two approaches. Secondly, the performance of approaches 0 and 2 starts to deteriorate from time step 3, from a NRMSE of 0.93 and 0.91 at time step 2 to around 1.21 and 1.1 at time step 7, respectively. The performance of approach 1 starts to deteriorate from time step 5, with a NRMSE around 1.5 at time step 4 and 1.58 at time step 7, respectively. Analysing the NRMSE per signal group, in Figure 7-2, it can

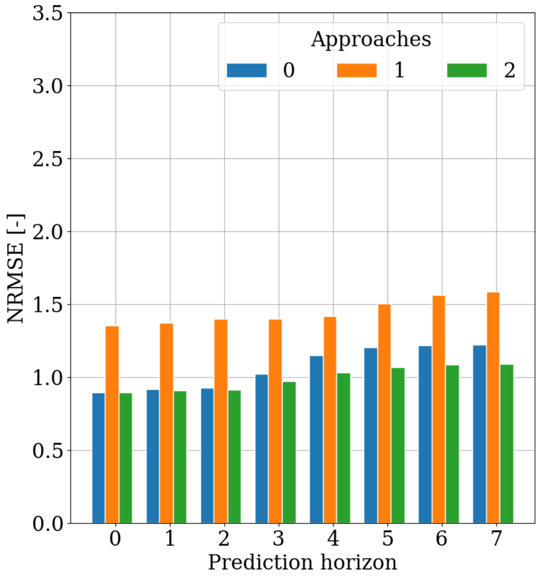


Figure 7-1: NRMSE per approach of LSTM prediction model with a time step of 5 seconds and a prediction horizon of 8 time steps.

be seen that straight-ahead moving signal groups with the highest demand, namely signal groups 2 and 8, perform best. This is followed by left-turn signal group 6 and right-turn signal group 7. Signal groups 3 and 4 have the lowest performance. With regards to the prediction horizon, the same trend can be seen as in Figure 7-1.
Looking closer at the predictions and actual arrival detections during simulation, in Figure 7-3, it can be seen that the LSTM predicts the arrival trends quite well, however, it suffers from noise when no arrivals are detected. This effect is less present in higher traffic demand shown in Figure 7-4.

Inspecting the LSTM prediction model with a time step of 2 seconds in Figure 7-5, it can be seen that the same phenomenon occurs as with the 5 seconds prediction model, namely that approach 1 performs the worst and approach 0 has a slightly inferior accuracy than approach 2. The performance gap between approaches 0 and 2 increases over the prediction horizon. With regards to the prediction horizon, the NRMSE starts to rise from time step 7 for approach 0, and around 8 for approach 2, and, in the case of approach 1 only around

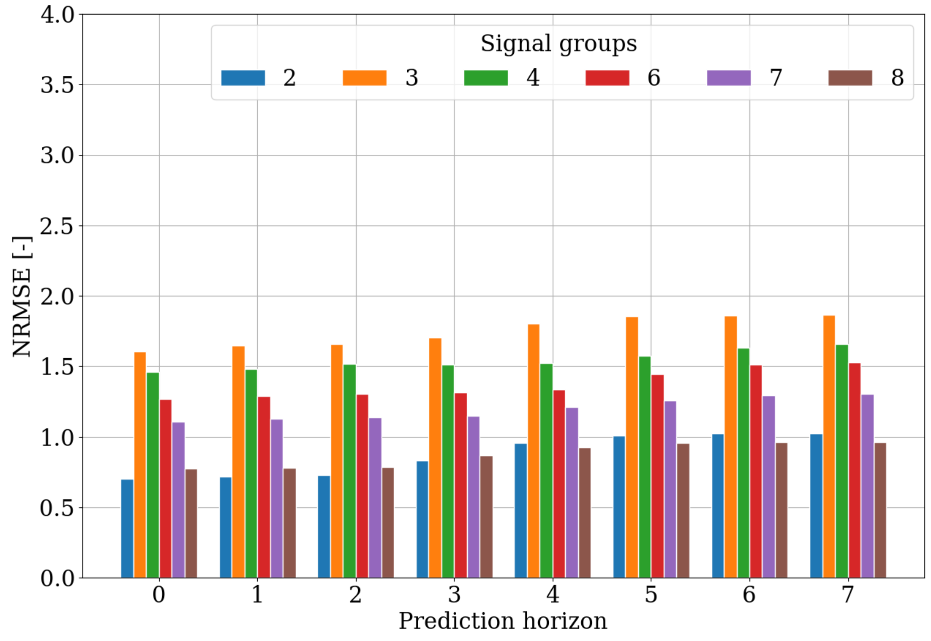


Figure 7-2: NRMSE per signal group of LSTM prediction model with a time step of 5 seconds and a prediction horizon of 8 time steps.

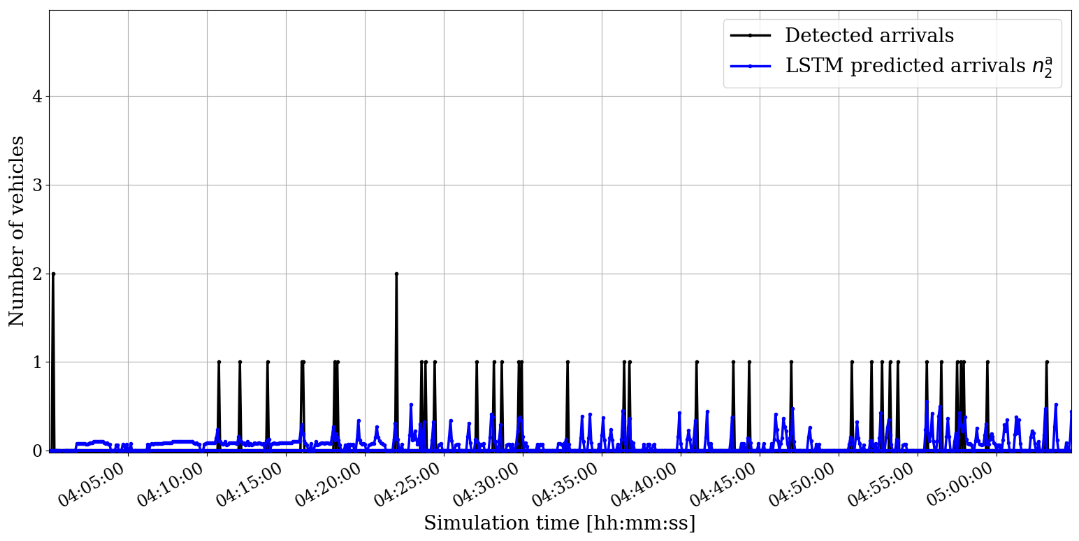


Figure 7-3: Predicted arrivals by LSTM and detected arrivals during simulation in low traffic demand.

time step 12. Comparing the overall performance to the 5 second LSTM model, the 2 second time step model's accuracy is considerably lower. In terms of signal groups, again, the same pattern can be seen as the 5 second time step LSTM model, meaning that signal groups 2 and 8 have the highest accuracy, followed by signal groups 6 and 7. The lowest accuracy is shown by signal groups 3 and 4.

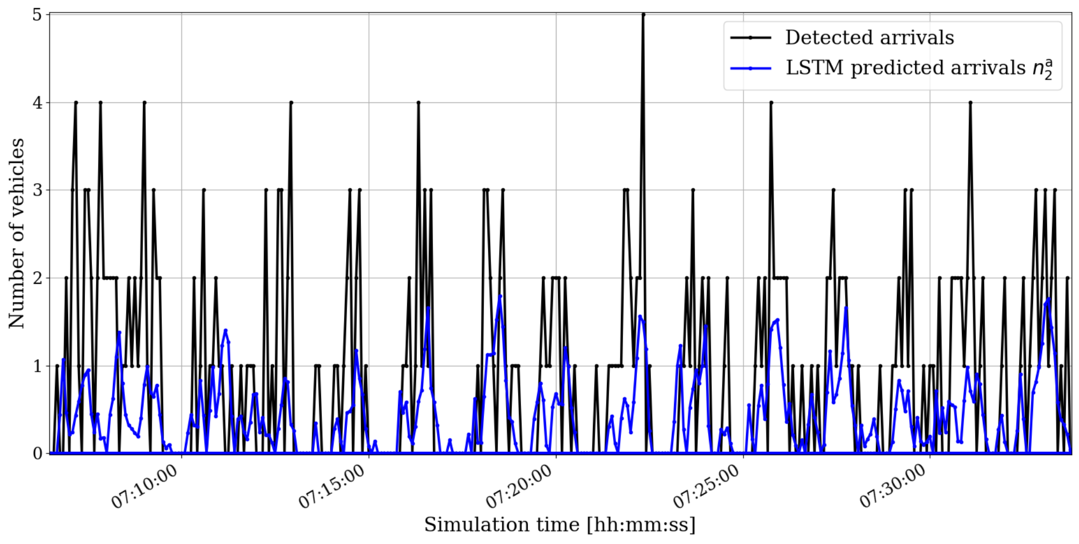


Figure 7-4: Predicted arrivals by LSTM and detected arrivals during simulation in high traffic demand.

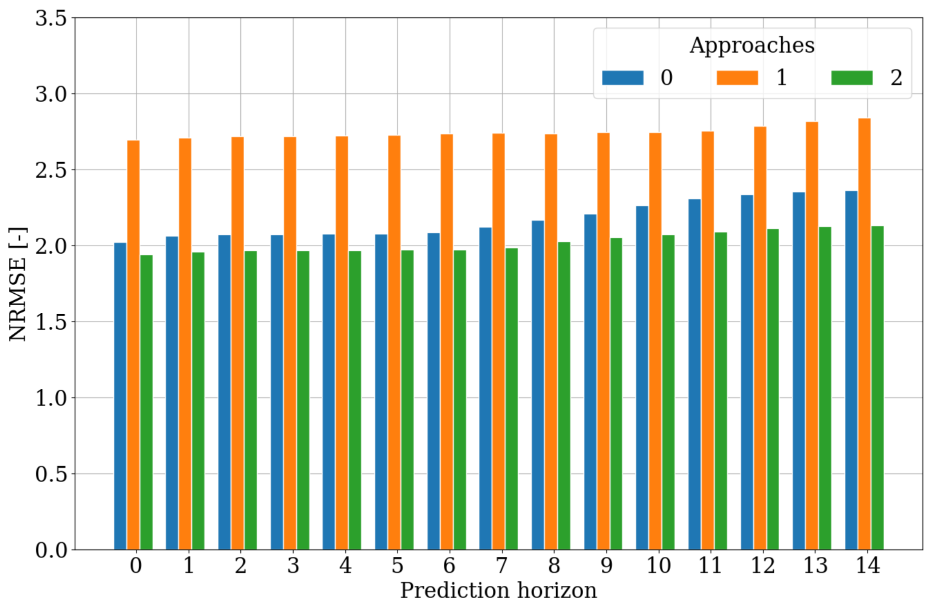


Figure 7-5: NRMSE per approach of LSTM prediction model with a time step of 2 seconds and a prediction horizon of 15 time steps.

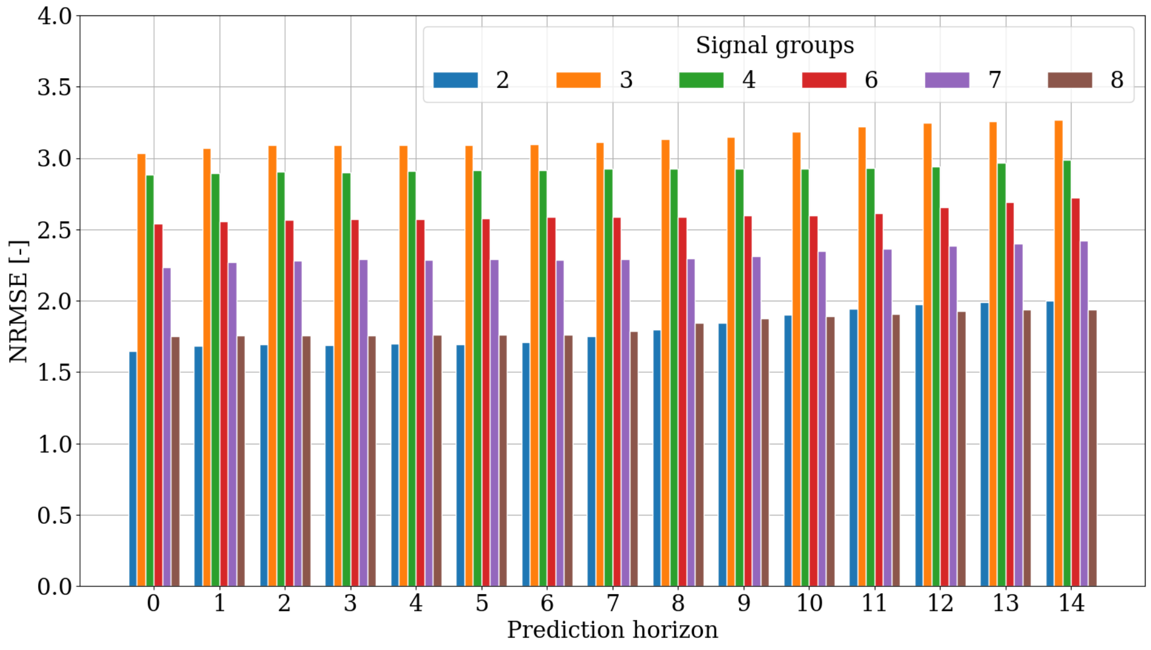


Figure 7-6: NRMSE per signal group of LSTM prediction model with a time step of 2 seconds and a prediction horizon of the first 15 time steps.

7-1-3 Linear Model

In general, the performance of the linear prediction model is less accurate for all approaches and prediction horizons, compared to the LSTM. As can be seen in Figure 7-7, the NRMSE is around one point mark higher than the LSTM model for all signal groups and prediction horizons. However, the performance does not deteriorate over the prediction horizon. Examining the signal groups individually, in Figure 7-8, turning movements perform the worst and straight-ahead signal groups 2 and 8 perform best.

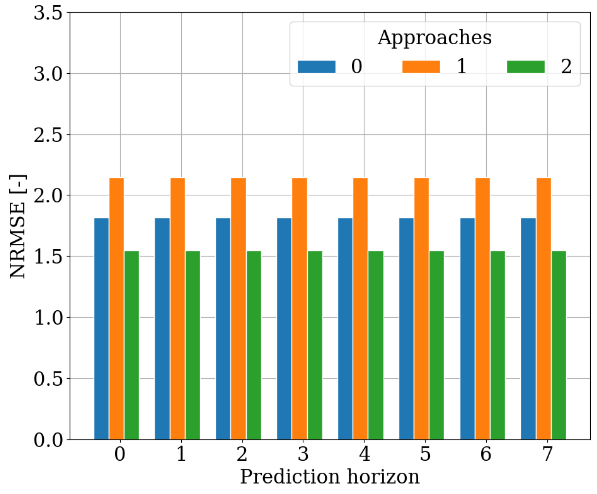


Figure 7-7: NRMSE per approach of linear delay prediction model with a time step of 5 seconds and a prediction horizon of 8 time steps.

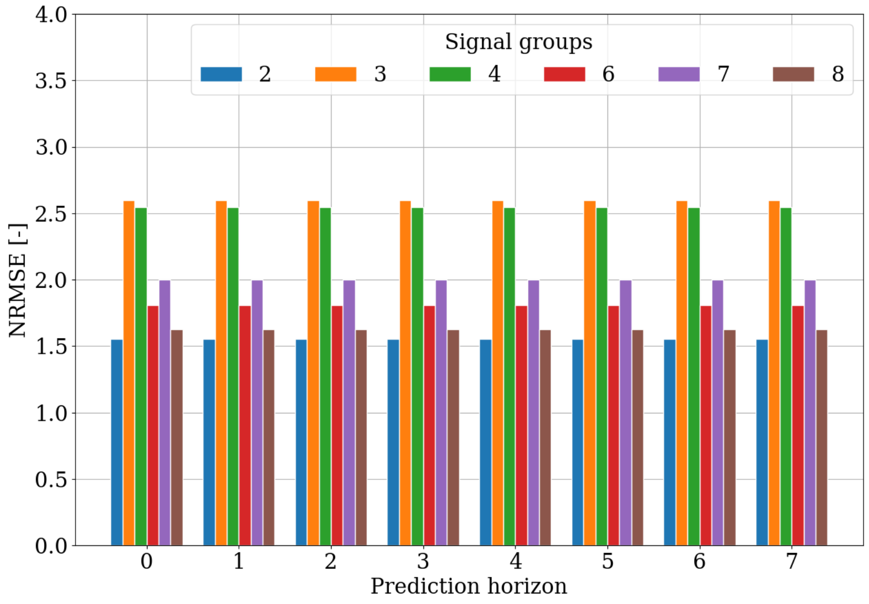


Figure 7-8: NRMSE per signal group of linear delay prediction model with a time step of 5 seconds and a prediction horizon of 8 time steps.

7-2 Simulation Experiments

The results of the simulation runs will be covered in comparative sections. In subsection 7-2-1, the actuated CCOL controller will be compared to Yunex's current DIRECTOR method and the newly developed Decentralized Model Predictive Controller (DeMPC). Next, in subsection 7-2-2, the solution time of the optimisation algorithm used in the DeMPC will be compared between 2 and 5 seconds time step Mixed Logical Dynamical (MLD) models. Thereafter, the results of the DeMPC using linear and LSTM prediction models will be covered in subsection 7-2-3. Lastly, in subsection 7-2-5, an overview of all simulation results will be presented. All simulation results are aggregated per 15 minutes.

7-2-1 Comparison of Actuated, DIRECTOR and Decentralised MPC Control Methods

In Figure 7-9, the mean delay time per vehicle over all simulation runs is depicted. DIRECTOR has a considerably higher delay time than the actuated control method and the DeMPC, both in low and high traffic demand. The delay time of the actuated controller is slightly lower than the DeMPC during the whole duration of the simulation. Regarding the mean Highway

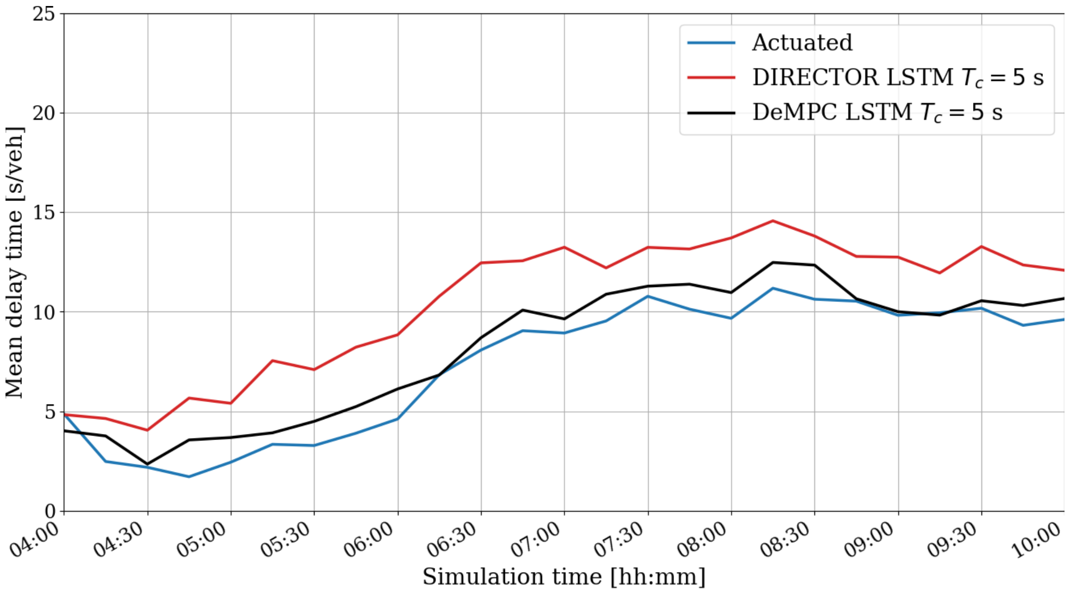


Figure 7-9: Mean delay time per vehicle of DeMPC, actuated and DIRECTOR methods

Capacity Manual (HCM) number of stops per vehicle during the simulation, shown in Figure 7-10, all methods have around 1,6 stops in high traffic demand. In low traffic demand, the actuated control method has the lowest number of stops, followed by the DeMPC and lastly DIRECTOR. In very low traffic demand, between 4:00 and 4:30, the gap between the number of stops of DeMPC and actuated controller is closer than between 4:30 and 6:00. In Figure 7-11, the mean queue length at the intersection displayed. The number of queued vehicle is highly similar between the actuated and the DeMPC during all traffic conditions. In very low traffic demand, from 4:00 to 5:00, the queue lengths at the intersection controlled by DIRECTOR are similar to the other two methods. From 5:00, the queue lengths for

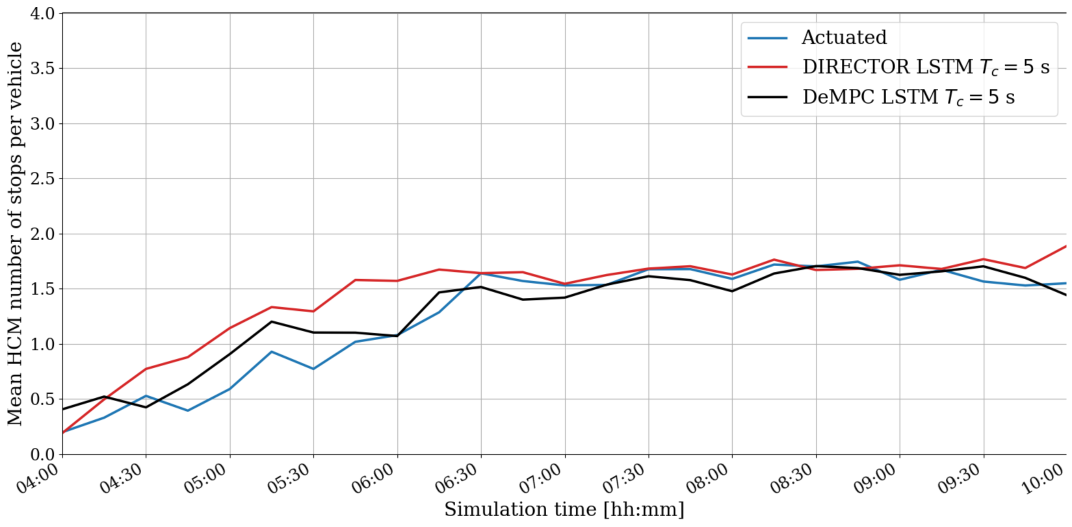


Figure 7-10: Mean number of stops per vehicle of DeMPC, actuated and DIRECTOR methods.

DIRECTOR start to raise slightly and from 06:30, there is a considerable difference between the DIRECTOR method and the other two methods in terms of mean queue length at the intersection.

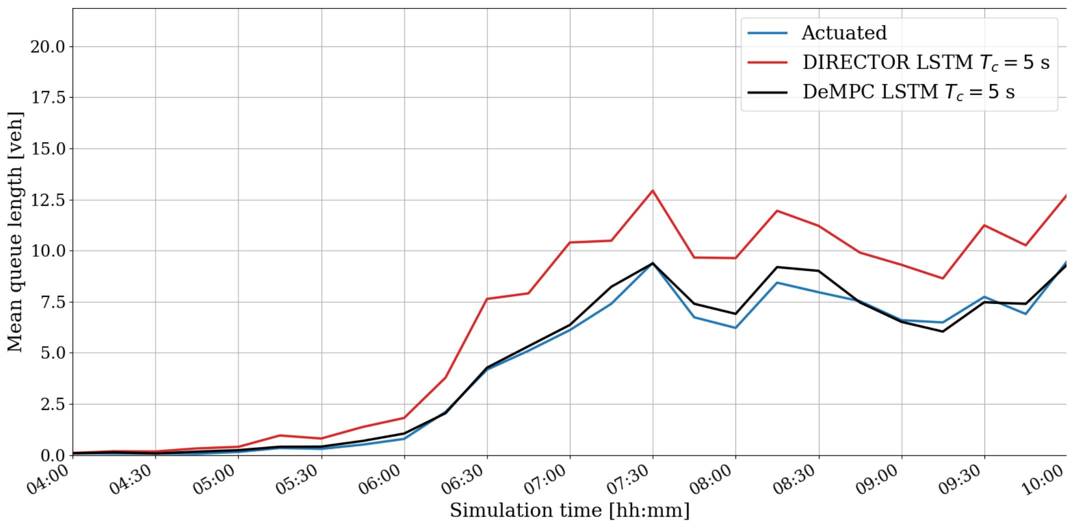


Figure 7-11: Mean intersection queue length of DeMPC, actuated and DIRECTOR methods.

7-2-2 Solution Time

The mean solver solution time over one simulation run is show in Figure 7-12. The solver solution time of the DeMPC with time step of 2 seconds is around double the time of the DeMPC with 5 second control time step. The optimisation problem of the 2 second model is on average 2,5 times larger than the optimisation problem with a 5 second control time step. In Figure 7-13, it can be seen that the solution time is comparable in very low traffic demand

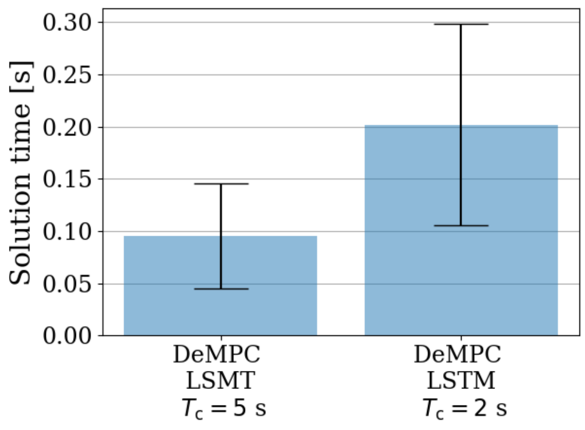


Figure 7-12: Mean and standard deviation of solver solution time between 5 and 2 second control time steps.

between 4:15 and 5:00. As the traffic demand increases, the solution time of the larger control time step optimisation problem increases to around 0.12 seconds, from 0.08 seconds in very low traffic demand. The same phenomenon happens to the smaller time step optimisation model, only the solution time increases to a greater extend, from around 0.1 seconds in very low traffic conditions to around 0.25 seconds in high traffic demand.

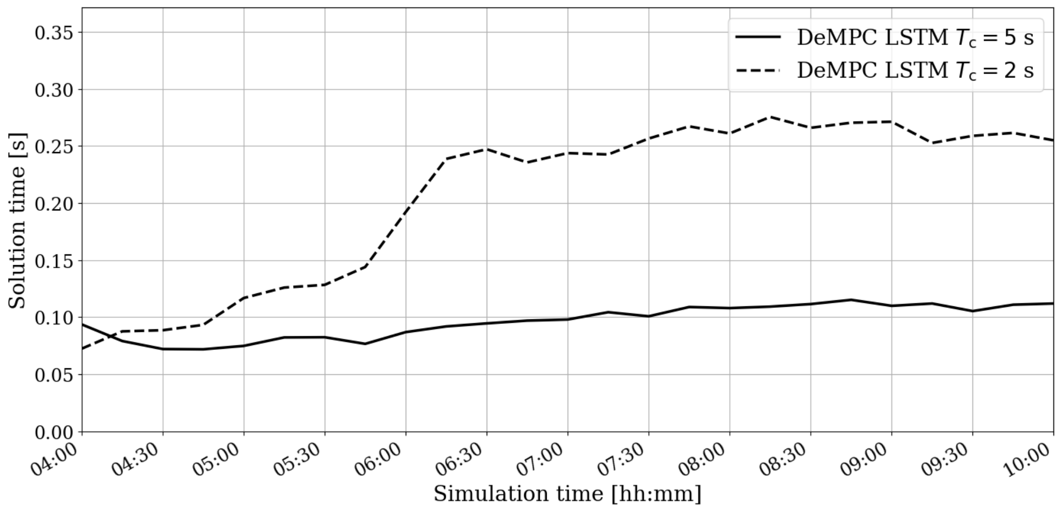


Figure 7-13: Solver solution time during simulation.

7-2-3 DeMPC with Different Arrival Prediction Models

The DeMPC with a 5 second time step is tested with a linear arrival prediction model and an LSTM neural network model. In Figure 7-14, the model comparison can be seen in terms of mean delay time per vehicle over all simulation runs. The mean delay is similar in low to medium traffic demands, from 04:00 to 6:30. In higher traffic demands, the DeMPC method with the arrivals predicted by the LSTM model considerably outperforms the linear prediction method.
Considering the HCM mean number of stops per vehicle over the simulation time, depicted in Figure 7-15, the linear prediction method results in slightly less number of stops per vehicle than the LSTM prediction method during all traffic conditions.
The mean number of queued vehicles at the intersections follows the same trend as the delay time in Figure 7-14. The two arrival prediction models coupled to the DeMPC result in similar queue lengths, shown in Figure 7-16, in low traffic demand conditions from 04:00 until 06:00. After 06:00, the queue lengths of the controller with the linear prediction method moves away from the controller with the LSTM prediction method.

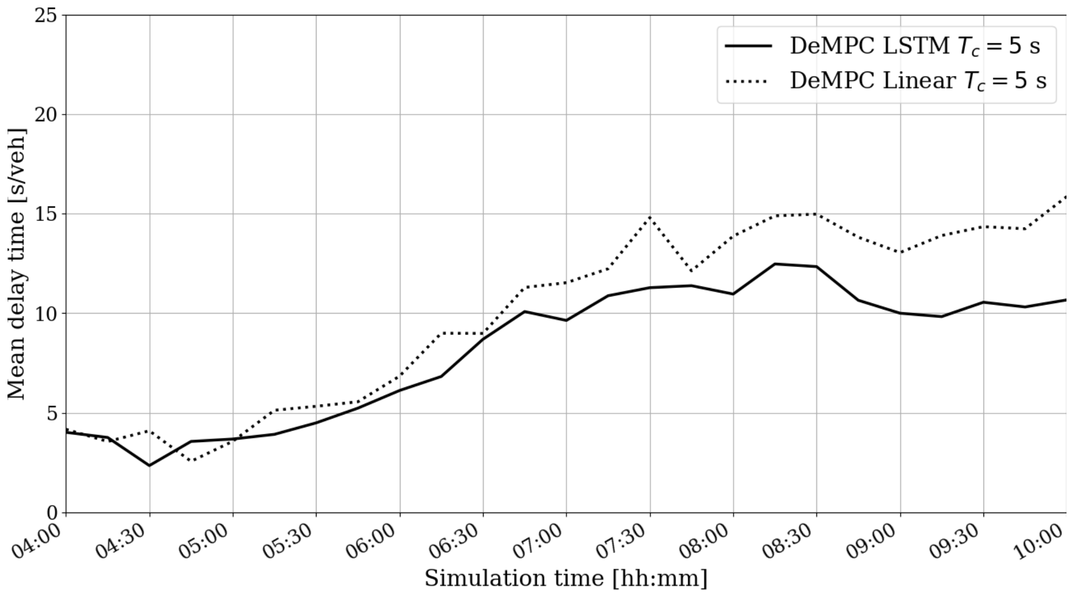


Figure 7-14: Mean delay time per vehicle of DeMPC with 5 second control time step with LSTM and linear arrival prediction methods.

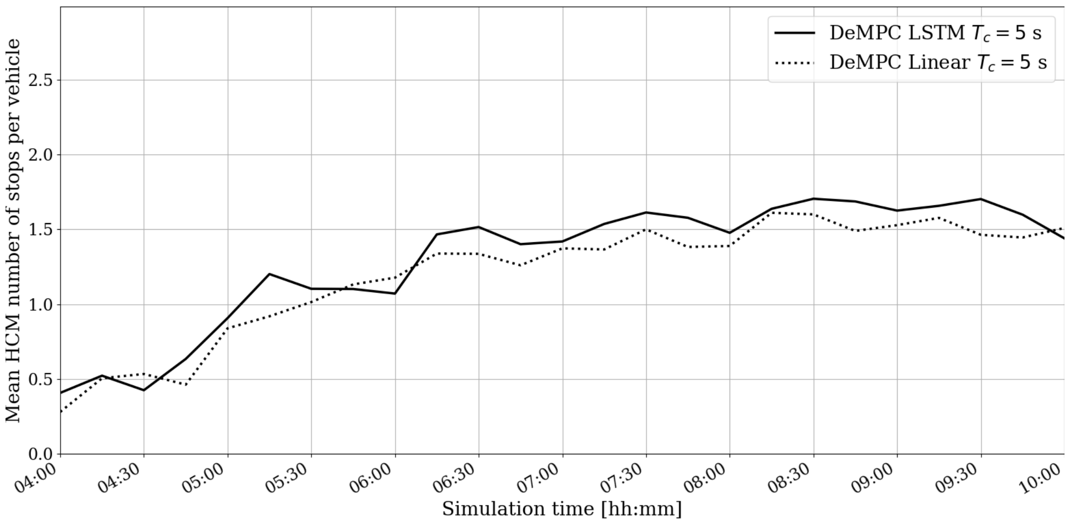


Figure 7-15: Mean number of stops per vehicle for the DeMPC with LSTM and linear arrival prediction methods.

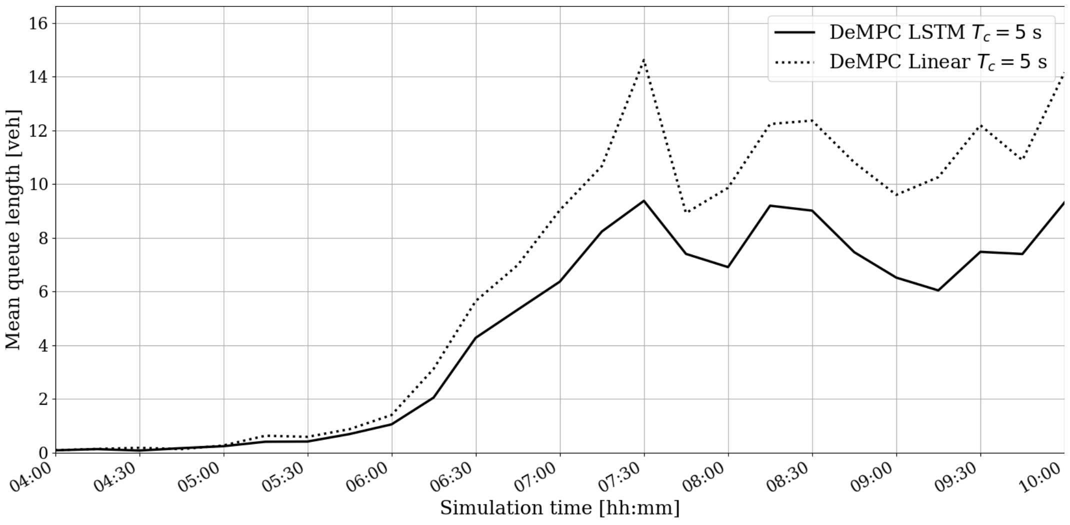


Figure 7-16: Mean intersection queue length for the DeMPC method with LSTM and linear arrival prediction methods.

7-2-4 DeMPC with Different Time Steps

The DeMPC was also simulated with two different time steps. In Figure 7-17, this comparison can be seen in terms of mean delay time per vehicle over all simulation runs. The mean delay is similar in low to medium traffic demand, from 04:00 to 6:30. In higher traffic demand, the DeMPC with a 5 second time step outperforms the same controller with a 2 second time step. For the 2 second time step, the mean HCM number of stops per vehicle over the simulation time, depicted in Figure 7-18, is slightly less than for the 5 second time step in the lower traffic demand from 04:00 to 06:00. In higher traffic demand, the number of stops per vehicle is similar between the two time steps.
The mean number of queued vehicles at the intersection follows the same trend as the delay time in Figure 7-14. The two time steps have similar queue lengths in low traffic demand conditions from 04:00 until 06:00. After 06:00, a gap in the queue length starts to arise between the two time step controllers.

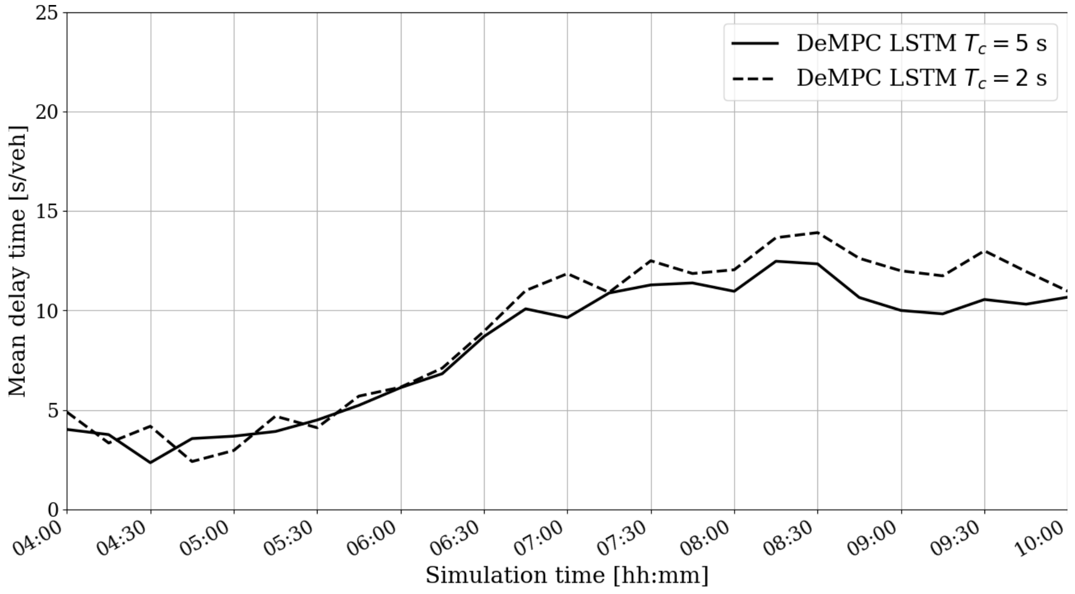


Figure 7-17: Mean delay time per vehicle for the DeMPC with 2 and 5 seconds control time steps.

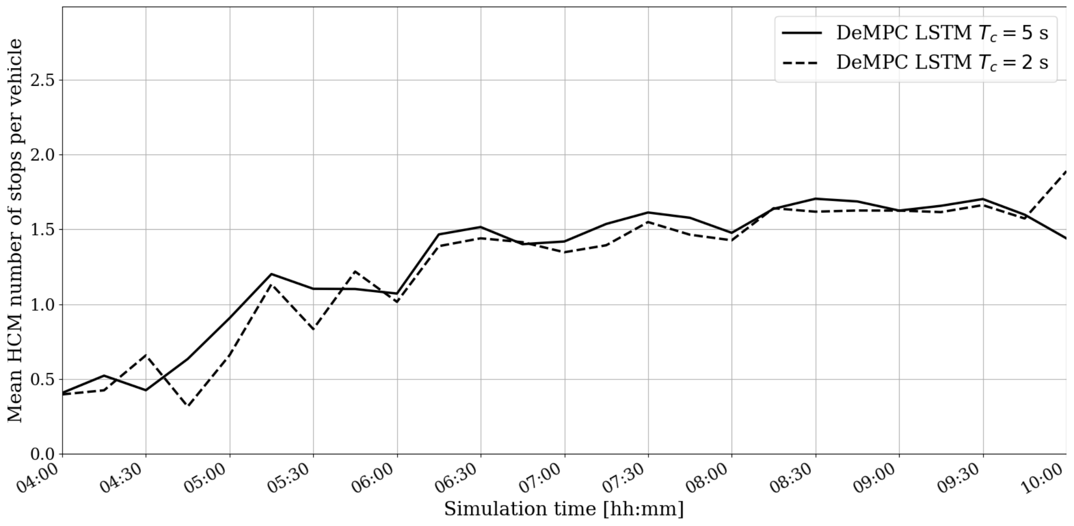


Figure 7-18: Mean number of stops per vehicle for the DeMPC with 2 and 5 seconds control time steps.

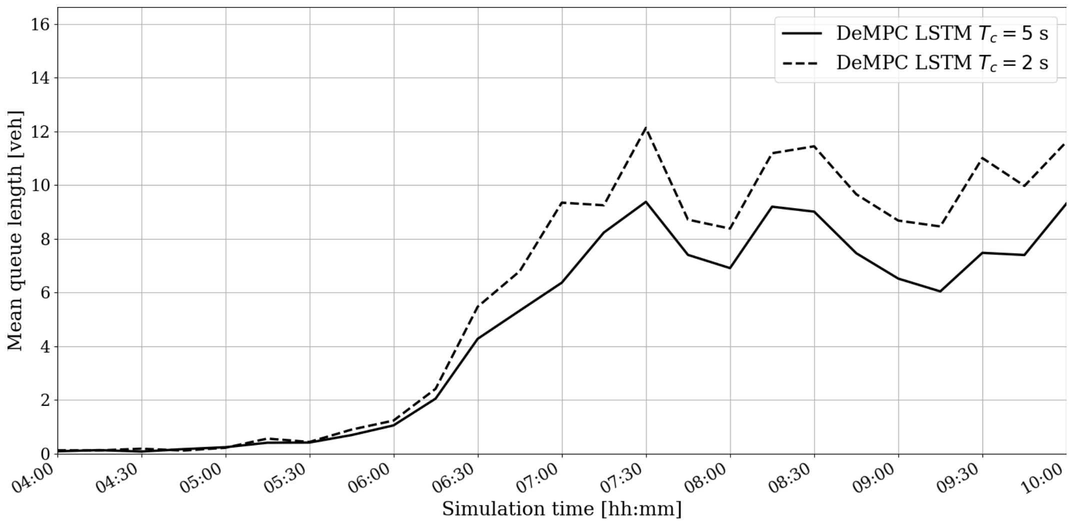


Figure 7-19: Mean intersection queue length for the DeMPC method with linear and LSTM prediction arrival models.

7-2-5 Overview

In Table 7-1 and Figure 7-20, all simulation results are presented for all performance indicators. These are presented as the mean and standard deviation over all replication runs. The number of stops indicate the total number of stops during the simulation. This is chosen instead of the average number of stops per vehicle as it is more indicative. Since the actuated controller and DIRECTOR do not use optimisation, the solver solution time is not shown for these methods. In terms of mean delay time per vehicle, the actuated methods performs best at 12.86 seconds per vehicle followed by the DeMPC with LSTM prediction and 5 second control time step at 13.95 seconds per vehicles. The DeMPC with linear prediction and 5 second control time step has an average delay of 17.44 second per vehicle, close to the 17.25 of DIRECTOR controller. All in all, the newly developed controller performs 24% better than DIRECTOR and only 8% behind the actuated controller in terms of mean delay time per vehicle. The DeMPC with linear arrival prediction and 5 second control time step performs comparable to the DIRECTOR method with LSTM prediction and a similar time step. The DeMPC with a 2 second time step has a 12% worse mean delay time and 13% lower total number of stops compared to the simulations with a 2 second time step.

Considering the total number of stops, the DeMPC with a time step of 5 seconds and LSTM arrival prediction outperforms the actuated controller by around 10% and DIRECTOR by 26%. The DeMPC with linear prediction and 5 seconds time step still achieves a 17% lower total number of stops than the DIRECTOR. As mentioned, the solution time doubles, from 0.095 seconds to 0.21, on average by increasing the model size by 2.5 times. The mean number of queued vehicles at the intersection is lowest in the actuated scenario. However, this is followed by the DeMPC LSTM with 5 seconds time step, which is only 5.3% off the actuated scenario in terms of the mean queue, but this is still 45% lower than DIRECTOR.

Table 7-1: Overview of simulation results.

Scenario	Delay time [s/veh]		Number of stops		Solution time [s]		Queue [veh]	
	Mean	± STD	Mean	± STD	Mean	±STD	Mean	±STD
Actuated	12.86	0.70	5861	1165	-	-	3.59	0.86
DIRECTOR LSTM $T_c = 5$	17.25	0.81	6689	156	-	-	5.5	0.85
DeMPC LSTM $T_c = 5$	13.95	0.46	5309	806	0.095	0.06	3.78	0.97
DeMPC Linear $T_c = 5$	17.44	0.35	5732	79	0.094	0.05	5.64	0.16
DeMPC LSTM $T_c = 2$	15.6	0.80	6025	616	0.21	0.09	4.94	0.47

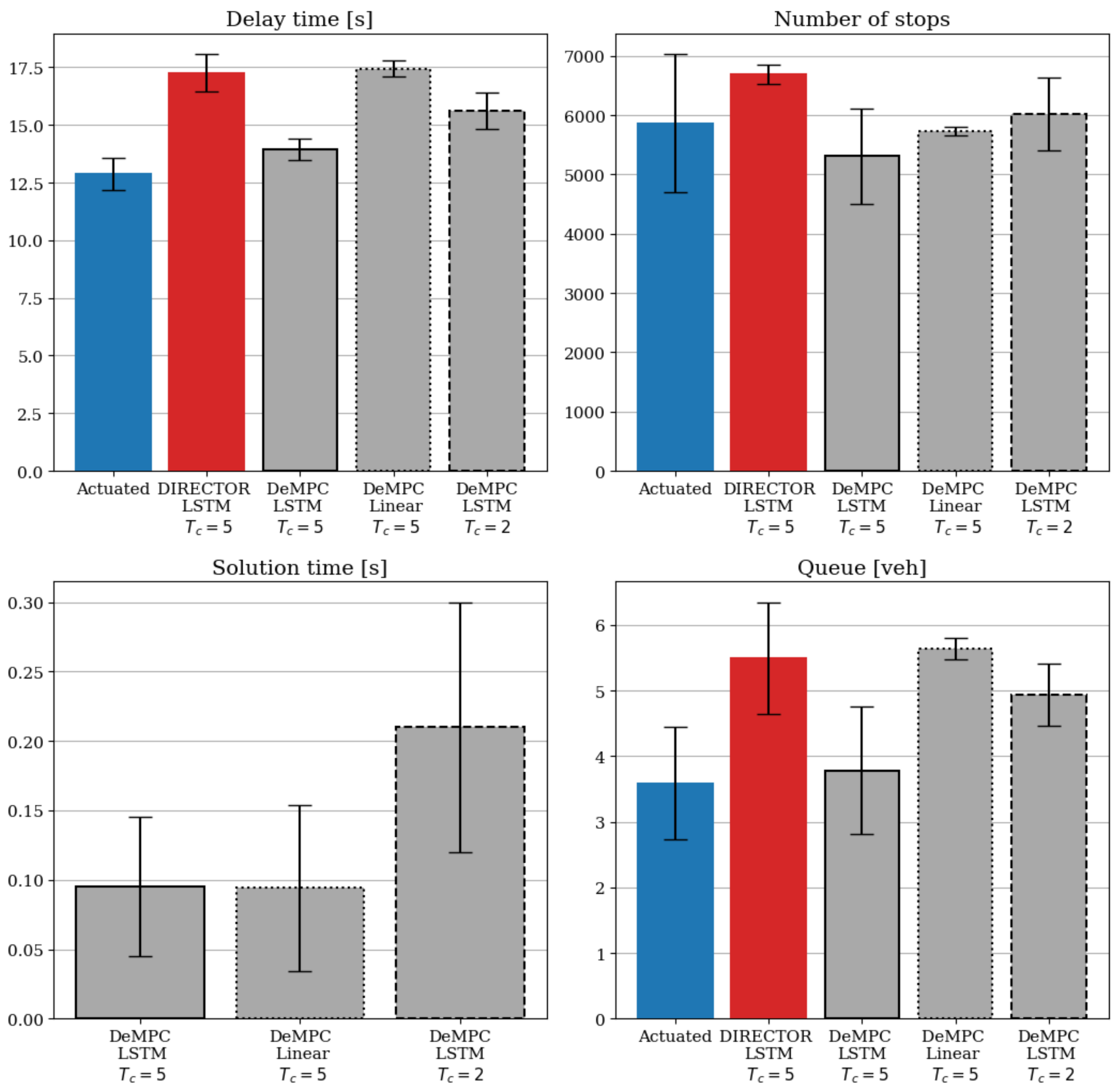


Figure 7-20: Overview simulation results.

Chapter 8

Discussion

The discussion of the results presented in the previous chapter will be divided in terms of the arrival prediction models in section 8-1 and the results obtained from the performed simulation experiments in section 8-2.

8-1 Prediction Models

Within each of the evaluated prediction models, approach 1 exhibits the most inferior Normalised Root Mean Square Error (NRMSE), because the departures originate from the furthest neighbour of intersection 201234, namely 20195 with 580 meters, while the other two intersection are located around 320 meters from intersection 201234. The platoon dispersion over longer links is thus more difficult to predict due to different driver and car characteristics. Regarding the prediction accuracy per approach over the horizon. For the LSTM prediction models, the accuracy deteriorates for all approaches from the time step that corresponds to the free-flow travel time of the link. This is due to the fact that the most up-to-date information the Long Short-Term Memory (LSTM) has, is the latest implemented signals and detections from the neighbouring intersections. Concerning the accuracy per signal group, signal groups 2 and 8 perform best. These are the straight-ahead signal groups with the highest traffic demand. Arrivals for signal groups that have turning movements are the most difficult to predict, due to the extra uncertainty in the turning percentage and added increased behaviour variety in braking for a turn during green [13].
Predicting arrivals with high accuracy in a 2 second time step is more difficult, since the prediction accuracy must increase further within a smaller precision of 2 seconds time step. The accuracy of the linear model was tested to see if it would obtain an acceptable accuracy for use in distributed optimisation techniques that requires decomposition of the link equation between two intersections. The linear prediction model performs poorly, since it does not take into account platoon dispersion and variability in the driving behaviour. However, the accuracy does not deteriorate over the prediction horizon, because the linear delay is based on a constant free-flow travel time.

8-2 Simulation Experiments

The discussion of the simulation experiments will be carried out per performance indicator, namely, delay time, number of stops and solution time. The discussion will be divided per traffic condition, since this has the most influence on the behaviour of the different controllers.

8-2-1 Delay Time

In low traffic conditions, the actuated controller outperforms both the Decentralized Model Predictive Controller (DeMPC) and DIRECTOR. In these low traffic conditions, the actuated controller can react fast to detections due to its short controller time step. Next to that, the signals always stay red when there are no vehicles arriving. In this way, the actuated control method can react even quicker to vehicle detections as it does not have to handle the clearance and amber times for the previously scheduled signal groups if a conflicting vehicle arrives. The signal timings of the DeMPC are dependent on the predictions made by the LSTM neural network, queue estimation and the developed queue model. The predictions made by the LSTM suffer from noise, meaning that when there are no arrivals, the LSTM predicts low arrival values of around 0.05 to 0.15. This is due to the fact that the LSTM is trained using the root-mean-square error loss function and the output is allowed to be real valued. The predicted noise allows for better root-mean-square error loss function, since this function penalises larger errors quadratically more than smaller errors. This comes to the forefront when the number of arrivals is small, i.e., one or zero arrivals in one time step. All this causes the signals of the DeMPC to give green to signal groups with no arrivals in low traffic demand. When a vehicle does arrive from a conflicting direction, the controller needs to switch the green signal and take into account the clearance times. This results in vehicles waiting or stopping in front of a red signal in low traffic conditions. With a 5 second control time step, the reaction time and efficiency of the implementing the signals is slower then the actuated method. The effect of the noise in the LSTM is further exemplified by comparing it to the linear prediction method. Even though the accuracy of the linear prediction method is poor, the delay performance of the DeMPC with linear prediction method is still comparable to the controller with LSTM predictions in low traffic conditions.

It was expected from previous research [57], that the DeMPC with a smaller time step would outperform one with a larger time step, because of the faster reaction time between mismatches between the model and the traffic system and more efficient control of the signals due to the smaller discretisation of time. However, this is not the case for the performed simulations. The 2 second controller has longer delay time per vehicle as the 5 second controller in low traffic conditions. From a detailed observation of the logging file of the control system, it was observed that the function evaluation of the learnt prediction models took more time than the control time step. Note that one prediction model is trained per approach and per time step in the prediction horizon. The predictions are obtained by looping serially through all prediction models. Since the smaller time step has a larger number of time bins this process takes between 2 and sometimes 5 seconds to complete. Therefore, the controller sometimes receives information about the arrivals and initial conditions from 2 time steps ago, because the results of the queue estimation and prediction models are communicated in one message to the controller once all prediction functions are evaluated. Therefore, in low traffic conditions, the delay performance between the DeMPC with a two second time step is similar to 5 second time step, because the miscommunication is offset by the better efficiency of implementing the signals.

The assumption that the vertical queue is placed at the arrival detector and that vehicles arriving are allowed to depart in the undersaturated condition has its consequences. The DeMPC method switches to green too early and switches to red too soon in low traffic demand, resulting in vehicles waiting in front of the red signal. Then, in the next time step, the controller needs to correct by giving green again for the same signal group.
The DIRECTOR control method also suffers from the above during low traffic flow. In addition, it schedules the signal group with the maximum delay at the end of the prediction horizon. This method is thus even more reliant on the predictions made by the LSTM model and does not perform an optimisation but a maximisation procedure, while DeMPC takes all the possible combinations of signal timings into account as well as the affect of the control signals on the next time steps is incorporated into the optimisation.

In medium to saturated traffic demand, the effect of noise in the predictions of the LSTM is less pronounced as more than zero or one vehicle is predicted per time step. The impact of the modelling assumptions is also less noticeable, since the the tail of the queue in higher traffic demand is closer to the location of the modelled vertical queue at the arrival detectors. Additionally, the impact of a larger control time step of the DeMPC, compared to the actuated controller, on the delay performance is less pronounced, because longer signal timings are scheduled during higher traffic conditions, compared to short green times for only one or two vehicle in lower demand conditions. Therefore, in medium to saturated traffic demand conditions, the performance of the DeMPC with 5 second time step is more comparable to the actuated controller, although it has slightly better delay performance. The DIRECTOR method performs worse than both DeMPC and actuated methods. Concerning the DeMPC with a shorter time step, the efficiency benefit of implementing the traffic signals disappears and the controller with the 2 seconds control time step performs worse in terms of delay in medium to high traffic demand. From observation, it can be seen that the DeMPC sometimes schedules signal groups with less demand first, such that afterwards the signal group with a larger demand can be scheduled for a longer time. It then schedules the higher demand signal groups together with predicted arrivals. This is not the case for DIRECTOR method with its greedy maximisation procedure, it can not take the longer term effect into account. Concerning the comparison of the linear and LSTM prediction modes, the DeMPC with linear prediction has a higher delay time in high traffic demand due to the fact that the noise in the arrivals is less pronounced than in low traffic demand.

Since the DeMPC minimises the queue length over the whole prediction horizon, the queue length is more comparable to the actuated method then would be expected from the difference in delay performance.

8-2-2 Number of Stops

In low traffic conditions, i.e., from 04:00 to 06:30, the number of stops of the DeMPC is significantly more than the actuated controller. Normally, the DeMPC should perform at least similar to the actuated method, since DeMPC has more information about future arrivals [57]. However, due to the noise in the predicted arrivals, the larger control time step of 5 seconds and modelling assumptions, this is not the case. In higher traffic conditions, the number of stops is very similar between the three methods, since vehicles have to stop in the queue anyway. The difference here is terminating the green time not too soon, so that vehicles do not have to stop twice. In these high conditions, the number of stops of the DeMPC is slightly less compared to the actuated method. It can be observed that the DeMPC waits to schedule a signal group until new arrivals are coming and keeps serving green longer, since the LSTM arrival predictions are more stochastic and predicted over multiple time steps, instead of one time step. These features reduce the number of stops, however it deteriorates the delay performance. The number of stops in high traffic conditions for DeMPC method is better than the actuated method, while the mean delay is lower for the actuated method in these conditions.

8-2-3 Solution Time

The solution time of the optimisation algorithm depends on the difficulty of finding the minimal queues over the prediction horizon. In lower traffic conditions this is fairly easy to find, since only the signal with an arriving vehicle needs to be green on the predicted time step of the arrival. Here, the optimal solution is found more easily as less branches in the branch-and-bound optimisation technique need to be explored. In high traffic demand, a more fine-grained weight-off needs to be made and more branches need to be explored. This is even more true for the problem with two second time step, containing a model that is 2.5 times larger than the 5 second optimisation problem. All thing considered, the problem of a too large solution time is not present here as a 0.2 seconds solution time still leaves enough room for the iterations of the coordination algorithm.

Chapter 9

Conclusions and Recommendations

In this thesis, a decentralised and distributed model predictive traffic signal controllers are developed to reduce delays in urban arterial networks. A Mixed Logical Dynamical (MLD) intersection model is formulated which includes operating constraints on the signal timings to make the controller on-street applicable for urban intersections. A distributed coordination algorithm is also developed to coordinate the signals of intersections in an urban traffic network. The project is undertaken in collaboration with Delft University of Technology (TU Delft) and Yunex Traffic. In the following, conclusions from the preformed simulation experiments are drawn and recommendations are formulated for future work.

The developed Decentralized Model Predictive Controller (DeMPC) with, an MLD intersection model with a 5 second time step and vehicle arrivals predicted by a Long Short-Term Memory (LSTM) netural network outperforms Yunex's DIRECTOR controller by 24% in terms of mean delay time per vehicle and by 26% in terms of total number of stops. Compared to the Dutch actuated controller, the DeMPC achieves just 8% more mean delay time per vehicle, while producing 10% less stops. Road owners that want to implement in-vehicle information services such as Green Light Optimised Speed Advice (GLOSA) are willing to trade-off the slightly higher delay time for the added predictability that the DeMPC offers.

Several assumptions were made in the modelling of the traffic system. The first places the vertical queue model at the arrival detector and the second models that arriving vehicles can depart in the same time step as their arrival, taking into account the saturation capacity is not met. These two assumptions deteriorate the delay performance most in low traffic demand, but decrease the total number of stops. The DeMPC minimises the queue length over the prediction horizon and is therefore more effective in reducing the queue length than the delay at the intersection. This is exemplified by a mean intersection queue of 5.1% above the mean queue of the actuated controller. This gap is larger than the delay performance gap.

The DeMPC with a linear arrival prediction method, based upon a constant link delay, was also simulated to investigate if it would obtain an acceptable accuracy for use in distributed optimisation techniques that require decomposition of the link equation between two intersections. It is concluded that this is not achieved as the delay time per vehicle is comparable to Yunex's greedy controller. This means that for modelling a traffic network to be used in a traffic signal controller, the link dynamics should be nonlinear to achieve acceptable performance. The accuracy of the LSTM arrival prediction method depends on the length of the link and its corresponding free-flow travel time. Straight-ahead signal groups with the highest demand have the highest accuracy and signal groups with lower demands and turning movements have the lowest Normalised Root Mean Square Error (NRMSE) accuracy.

To investigate the solution time, the DeMPC was simulated with a 5 and 2 second time step while keeping the prediction horizon for both methods at 40 seconds. This yielded a 2.5 fold increase in the MLD model size, while the solution time only increased by a factor of 2 to 0.21 seconds on average during a simulation of 6 hours. The solution time of the optimisation algorithm also depends on the traffic demand. During high traffic demand, the solution time of the 2 second model is around 0.25 seconds larger than in low traffic demand. For the 5 second model, this difference is 0.04 seconds. It is thus concluded that the solution time of the DeMPC is small enough for the controller to be extended to larger intersection models containing more traffic signals. The mean delay, total number of stops and mean queue length of the performed simulations with the 2 second time step are not included in this conclusion, because these results are not indicative, meaning the performance of the controller was influenced by the implementation of the LSTM predictions (see subsection 8-2-1).

For future work, it is first recommended that the proposed simulations of the DeMPC and Distributed Model Predictive Controller (DiMPC) for the arterial network should be carried out. Next, the DeMPC should be tested in combination with GLOSA. This firstly entails investigating the predictability of the signal timings and tuning the number of time steps that the signals are fixed ahead. Currently, the first time step of the optimised signals is implemented into the system at each control time step.
Next, it is recommended that the mismatch between the used models and the traffic system should be reduced. For the LSTM prediction models, this entails reducing the noise during low traffic volumes caused by training the model using the Root Mean Square Error (RMSE). Additionally, for the implementation of the LSTM, the function evaluations of the models should be done in parallel through multi-threading and not by looping through the models in a serial manner. For the intersection model, the arrivals should be added to the queue at the tail of the queue and not at the arrival loop. The mismatch can be further reduced by using a horizontal queue model that models the queue, arrival and departure dynamics more closely. Then, the variables used in the horizontal queue model can be utilised in the objective function to approximate the delay at intersection more accurately than approximating the delay by integrating the queue length over the prediction horizon. Furthermore, a formal tuning of the DeMPC should be performed to find the ideal trade-off between the control time step and the prediction horizon.

It is also recommended that the current intersection model should be tested in a MPC controller with multi-modal signals for pedestrians, cyclists, and public transportation. This can be easily done by adding departure and queue equations for new signals. For road owners, it is also desirable to choose the objective that fits their own preferences. Therefore, multi-objective optimisation, which combines multiple objectives into one objective function, is also desirable. In addition, the weights of the current objective function are all set to a value of one, meaning that each signal group is weighted the same amount. An investigation into the effect of varying these weights could also be performed. In this way, road owners can express their preferences for specific signal groups or modalities.

Appendix A

Traffic Signal Controller Yunex

The current traffic signal controller is based on the self-organising controller proposed by Lämmer et al. in [73, 74, 92, 93, 94], but adapted from the ground-up in the thesis of J.C van Senden's [64]. The controller is called DIRECTOR, which stands for Data-driven Intersection and Road Environment Controller for Traffic Optimisation in Real-time. The controller is implemented using the Python programming language and tested using VISSIM and AIMSUM traffic simulators.

The controller's inputs are the predicted vehicle arrival times at the upstream detector (see section A) and the measurements from the upstream, queue and stop line detectors. The control objective is to minimise the average travel time delay per vehicle. Therefore, every ten seconds, the controller gives a green signal to the phase group that has the most cumulative predicted travel time delay. The cumulative travel time delay is the sum of travel time delays of all vehicles within a phase group. This is calculated by summing the number of queued vehicles for each lane over a 10 second time interval. The cumulative predicted travel time delay is described by the last rule of Equation A-1, where the phase group (PG) with the maximum delay (D_{od}) over two time bins, zero and one, is selected. The term $\sigma_{PG}(t_o)$ denotes the switching penalty, which accounts for the added delay caused by a signal switch (due to clearance time and amber time). In essence, the controller can be said to be "greedy" in its determination of which signal to schedule as only the phase with the maximum predicted delay is chosen. The controller thus tends to allocate the green signal to the phase that has a longer queue length.

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

Network stability is guaranteed by bounding the queues at all times. When the tail of a queue spills backwards to permanently cover the upstream detector, the phase group corresponding to that queue is added to the FIFO (First-In-First-Out) set Ω. This is an ordered set, according to a stable, cyclic, fixed-time schedule. Whenever this set is not empty, the next phase group to be serviced is the next entry of Ω. This also holds for FIFO set Ψ, where a phase group is added when it has exceeded the maximum time without service. This is necessary to ensure that there is no negation of red lights by drivers. Phase groups from Ψ are preferred over other scheduling decisions to ensure safety. It should be noted that for the phase sequence to be stable (i.e., no oscillation of phases), when operating in the back-up fixed-time schedule mode, the maximum time without service is at least as large as the cycle time ($t^{max} \geq t^{cycle}$). Otherwise, the phase sequence will not be able to complete a full cycle.

Prediction method

The prediction method comprises of a machine learning technique, called Long Short-Term Memory (LSTM) network developed by N. Helmy [63] and further improved upon by J.C. Van Senden [64] and T. Glastra [62]. This technique uses historical detector and signal state data to learn the predicted vehicle arrival times. The input features are the following:

- The day of the week and time of day
- Preceding intersection's departures, queues, arrivals (measurements of stop line, queue and upstream detectors, respectively) and signal states.
- Controlled intersection's departures, queues and arrivals and predicted signal states.

These inputs can be seen in Figure 2-3, which depicts the preceding intersection on the left side and the controlled intersection on the right side. The outputs of the network are the vehicle arrivals at the location of the upstream detectors with turning percentages, predicted queue presence, and predicted signal states. Only the predicted vehicle arrivals are used as inputs to the controller. For vehicle detections closer or more downstream to the intersection, the controller relies on the actual measurement from the queue and stop-line detectors. This can be seen in the Figure A-1 below, where T stands for the sampling time of 10 seconds. This method can predict arrivals with reasonable accuracy up to 50 seconds horizon. It should be noted that this is based on signal states that are not known in advance. When the neighbouring intersection is operated using a predictive controller, the signal timing can be known in advance and the prediction horizon could be extended.

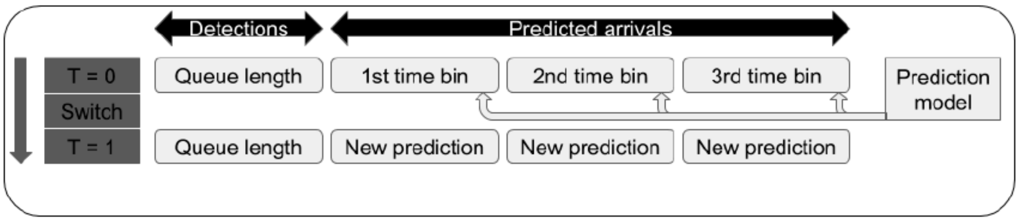


Figure A-1: Vehicle predictions and detections.Source [62].

Appendix B

Control System

In this appendix, the simulation control system is shown in detail. In Figure B-1, a detailed control system is depicted. In Figure B-2, an overview of all configuration files is presented together with the needed variables to be configured. The mobility data broker is a protocol converter that receives messages, reformulates messages into another protocol, for example, from Json format to binary Contactgroep Verkeersregeltechnici Nederland Internet Protocol (CVNIP) format. This is needed since the Traffic Light Controller (TLC) communicates using binary CVNIP messages and the rest of the communication is done via RabbitMQ using Json format. CVNIP is thus a standardised interface for communicating with TLCs made in the C programming language. RabbitMQ is an open source message broker [95] that enables application components to be decoupled and run separately while sending information between them.

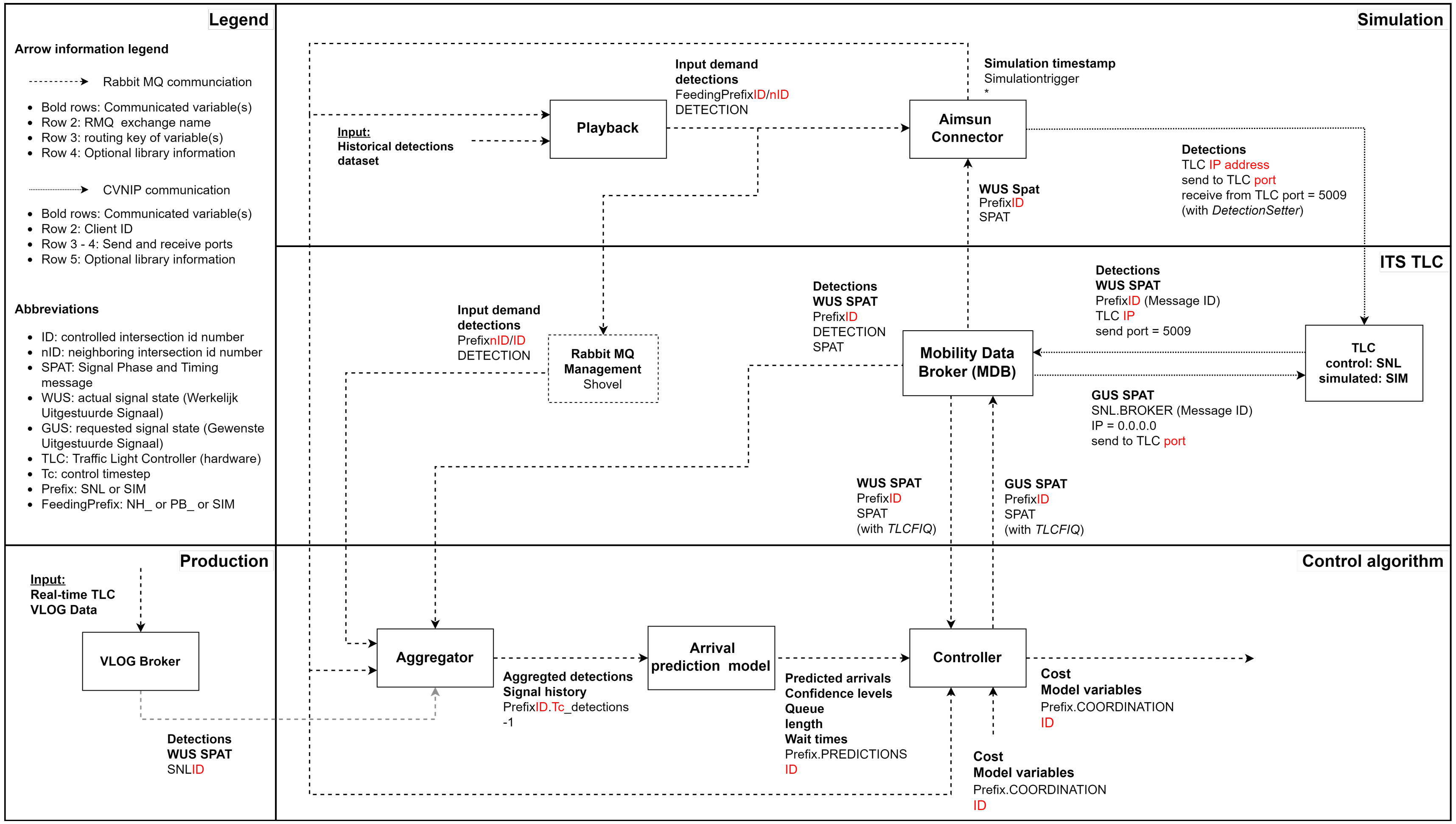
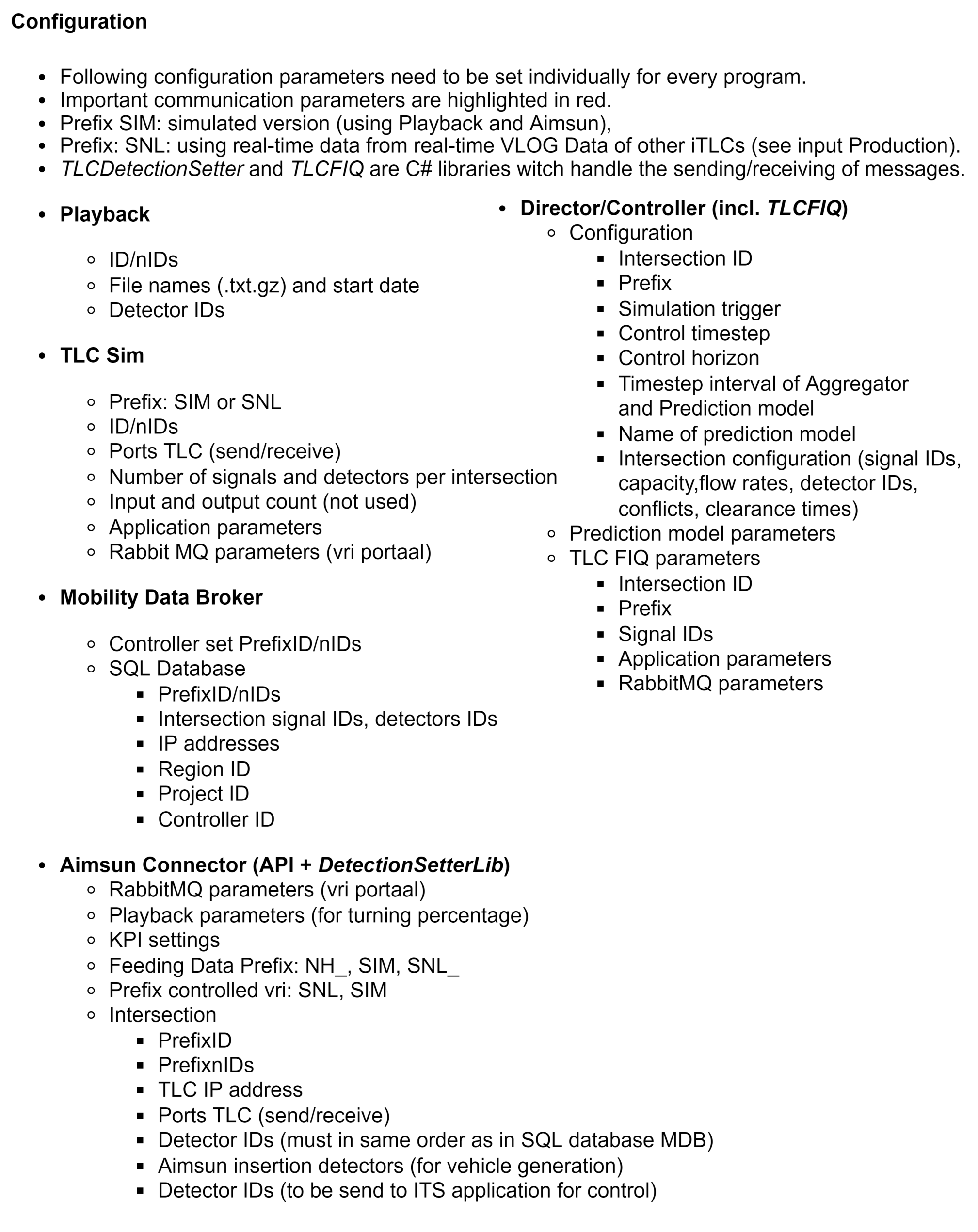

Legend
Arrow information legend
Rabbit MQ communciation
Bold rows: Communicated variable(s)
Row 2: RMQ exchange name
Row 3: routing key of variable(s)
Row 4: Optional library information
CVNIP communication
Bold rows: Communicated variable(s)
Row 2: Client ID
Row 3 - 4: Send and receive ports
Row 5: Optional library information
Abbreviations
ID: controlled intersection id number
nID: neighboring intersection id number
SPAT: Signal Phase and Timing message
WUS: actual signal state (Werkelijk Uitgestuurde Signaal)
GUS: requested signal state (Gewenste Uitgestuurde Signaal)
TLC: Traffic Light Controller (hardware)
Tc: control timestep
Prefix: SNL or SIM
FeedingPrefix: NH_ or PB_ or SIM
Simulation
Input: Historical detections dataset
Playback
Input demand detections FeedingPrefixID/nID DETECTION
Aimsun Connector
Simulation timestamp Simulationtrigger *
Detections TLC IP address send to TLC port receive from TLC port = 5009 (with DetectionSetter)
WUS Spat PrefixID SPAT
ITS TLC
Input demand detections PrefixnID/ID DETECTION
Rabbit MQ Management Shovel
Detections WUS SPAT PrefixID DETECTION SPAT
Mobility Data Broker (MDB)
Detections WUS SPAT PrefixID (Message ID) TLC IP send port = 5009
TLC control: SNL simulated: SIM
GUS SPAT SNL.BROKER (Message ID) IP = 0.0.0.0 send to TLC port
WUS SPAT PrefixID SPAT (with TLCFIQ)
GUS SPAT PrefixID SPAT (with TLCFIQ)
Production
Input: Real-time TLC VLOG Data
VLOG Broker
Detections WUS SPAT SNLID
Control algorithm
Aggregator
Aggregted detections Signal history PrefixID.Tc_detections -1
Arrival prediction model
Predicted arrivals Confidence levels Queue length Wait times Prefix.PREDICTIONS ID
Controller
Cost Model variables Prefix.COORDINATION ID
Cost Model variables Prefix.COORDINATION ID

Figure B-1: Detailed control system.



Configuration

- Following configuration parameters need to be set individually for every program.
- Important communication parameters are highlighted in red.
- Prefix SIM: simulated version (using Playback and Aimsun),
- Prefix: SNL: using real-time data from real-time VLOG Data of other iTLCs (see input Production).
- *TLCDetectionSetter* and *TLCFIQ* are C# libraries witch handle the sending/receiving of messages.

- **Playback**
  - ID/nIDs
  - File names (.txt.gz) and start date
  - Detector IDs
- **TLC Sim**
  - Prefix: SIM or SNL
  - ID/nIDs
  - Ports TLC (send/receive)
  - Number of signals and detectors per intersection
  - Input and output count (not used)
  - Application parameters
  - Rabbit MQ parameters (vri portaal)
- **Mobility Data Broker**
  - Controller set PrefixID/nIDs
  - SQL Database
    - PrefixID/nIDs
    - Intersection signal IDs, detectors IDs
    - IP addresses
    - Region ID
    - Project ID
    - Controller ID
- **Aimsun Connector (API + *DetectionSetterLib*)**
  - RabbitMQ parameters (vri portaal)
  - Playback parameters (for turning percentage)
  - KPI settings
  - Feeding Data Prefix: NH_, SIM, SNL_
  - Prefix controlled vri: SNL, SIM
  - Intersection
    - PrefixID
    - PrefixnIDs
    - TLC IP address
    - Ports TLC (send/receive)
    - Detector IDs (must in same order as in SQL database MDB)
    - Aimsun insertion detectors (for vehicle generation)
    - Detector IDs (to be send to ITS application for control)
- **Director/Controller (incl. *TLCFIQ*)**
  - Configuration
    - Intersection ID
    - Prefix
    - Simulation trigger
    - Control timestep
    - Control horizon
    - Timestep interval of Aggregator and Prediction model
    - Name of prediction model
    - Intersection configuration (signal IDs, capacity,flow rates, detector IDs, conflicts, clearance times)
  - Prediction model parameters
  - TLC FIQ parameters
    - Intersection ID
    - Prefix
    - Signal IDs
    - Application parameters
    - RabbitMQ parameters

Figure B-2: Detailed configuration of control system

Appendix C

Additional Figures

In this appendix, additional figures are placed. These contain the Traffic Light Controller (TLC) hardware rack and saturation headway per queue position.

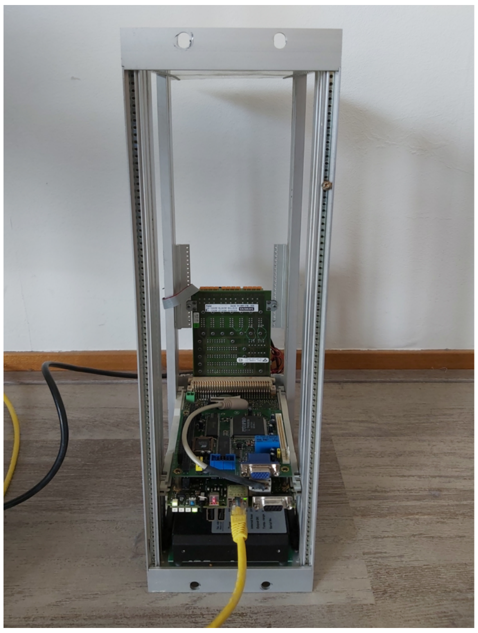


Figure C-1: TLC hardware rack.

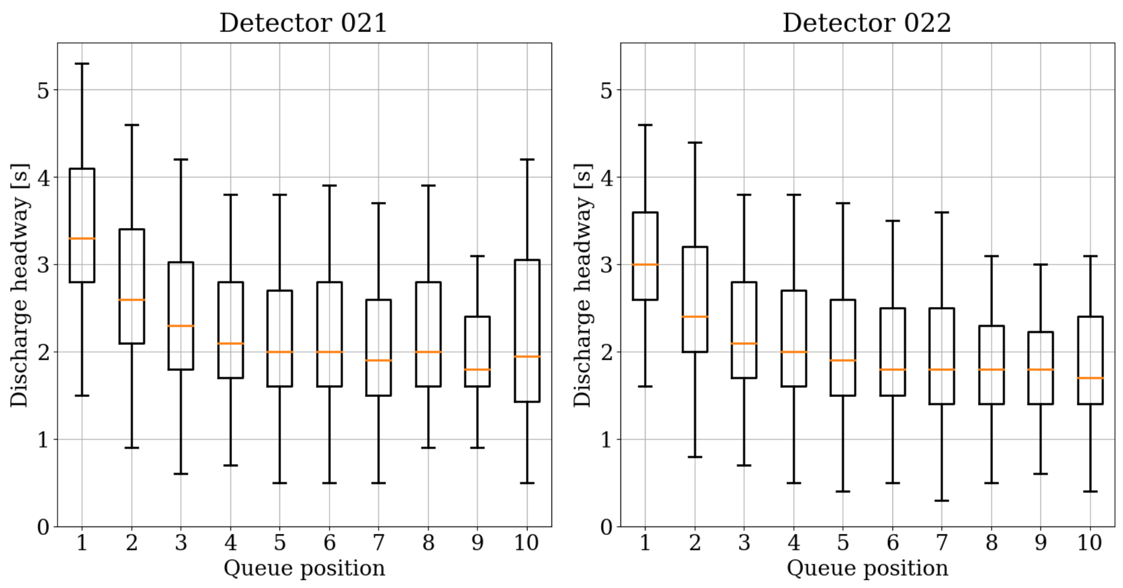


Figure C-2: Discharge headways at stopline detectors of signal group 2, filtered on queue presence and during green. The data is averaged over the 5 workdays the week before 15/10/2019.

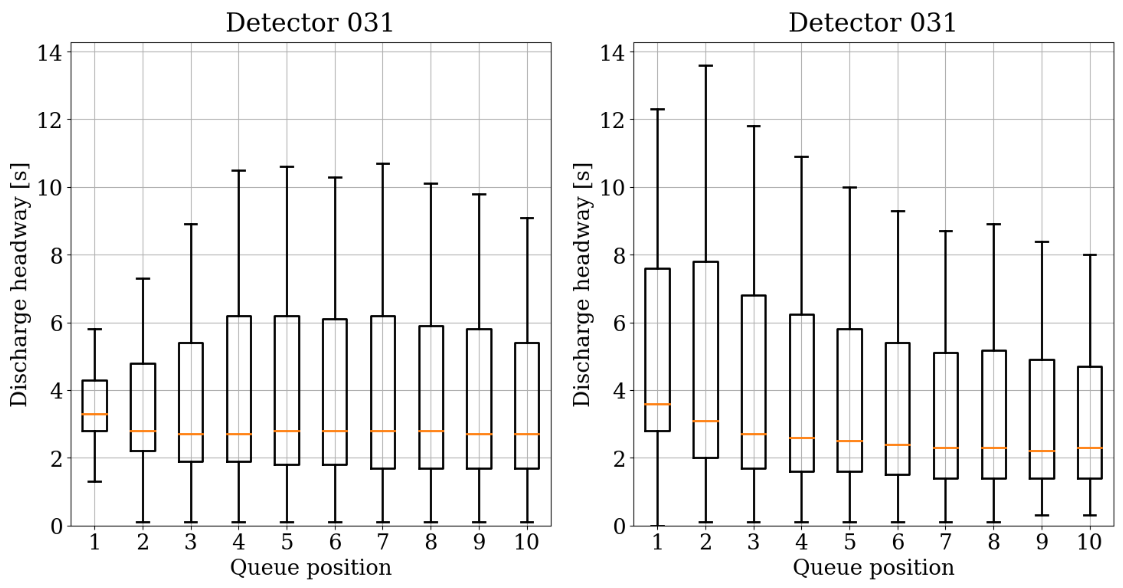


Figure C-3: Discharge headways at stopline detectors of signal group 3, filtered on queue presence and during green. The data is averaged over the 5 workdays the week before 15/10/2019.

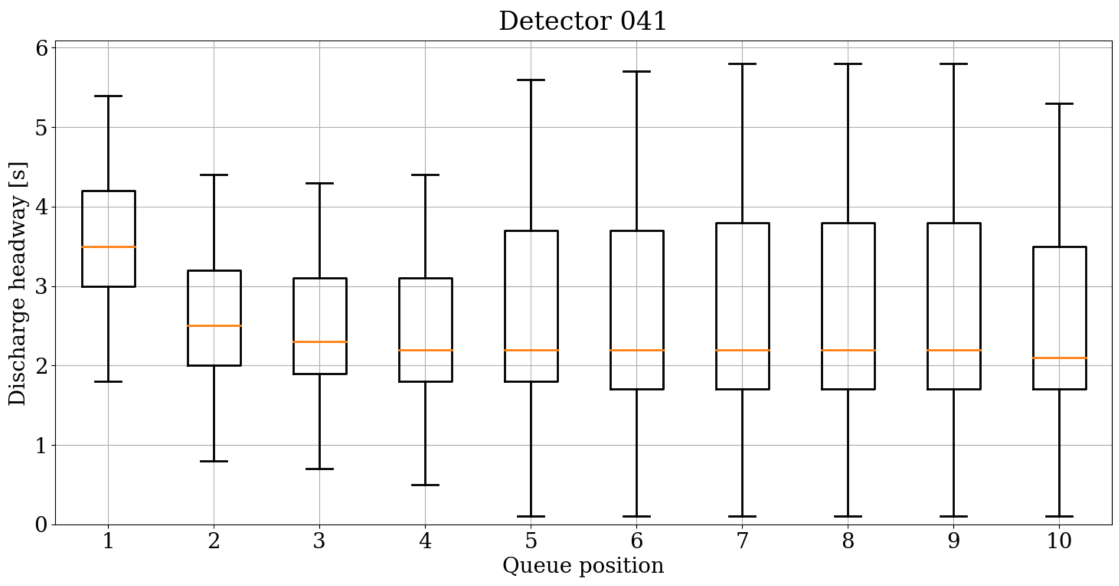


Figure C-4: Discharge headways at stopline detectors of signal group 4, filtered on queue presence and during green. The data is averaged over the 5 workdays the week before 15/10/2019.

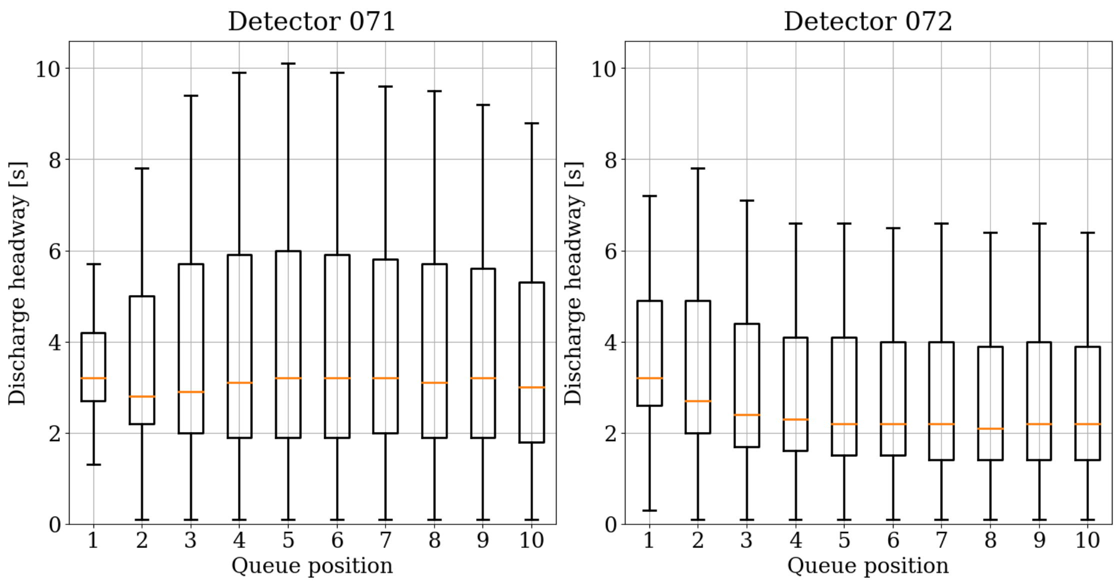


Figure C-5: Discharge headways at stopline detectors of signal group 7, filtered on queue presence and during green. The data is averaged over the 5 workdays the week before 15/10/2019.

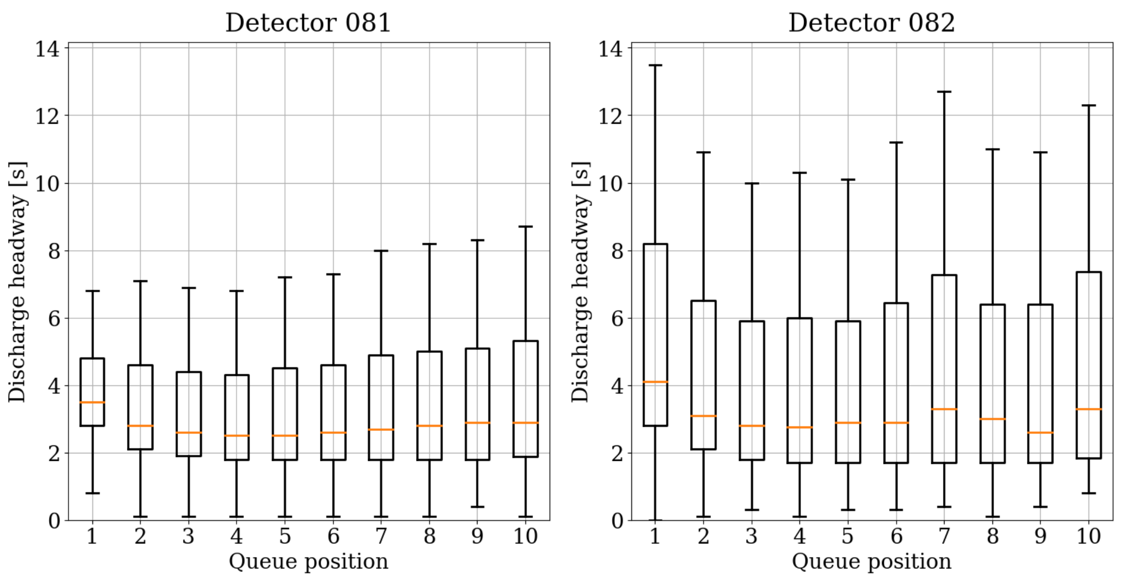


Figure C-6: Discharge headways at stopline detectors of signal group 8, filtered on queue presence and during green. The data is averaged over the 5 workdays the week before 15/10/2019.

Bibliography

[1] USchools, “Business concepts: Intersection,” 2016. [Online; accessed 28/08/2020].

[2] G. S. van de Weg, M. Keyvan-Ekbatani, A. Hegyi, and S. P. Hoogendoorn, “Linear mpc-based urban traffic control using the link transmission model,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–16, 2019.

[3] M. Barth and K. Boriboonsomsin, “Real-world carbon dioxide impacts of traffic congestion,” *Transportation Research Record*, vol. 2058, no. 1, pp. 163–171, 2008.

[4] E. E. A. (EEA), “Greenhouse gas emissions from transport in europe.” (accessed:23.11.2020).

[5] P. Christidis, “Measuring road congestion,” *JRC Scientific and Policy Reports*, 2012.

[6] ANWB, “17 procent meer files op de nederlandse wegen.” [Online; accessed 27.04.2020].

[7] cbs, “Aantal wegvoertuigen blijft stijgen.” [Online; accessed 03/03/2020].

[8] B. De Schutter, H. Hellendoorn, A. Hegyi, M. Berg, and S. Zegeye, *Model-based Control of Intelligent Traffic Networks*, pp. 277–310. 11 2010.

[9] G. S. van de Weg, H. L. Vu, A. Hegyi, and S. P. Hoogendoorn, “A hierarchical control framework for coordination of intersection signal timings in all traffic regimes,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1815–1827, 2019.

[10] D. M. of Infrastructure and Environment, “Partnership talking traffic.” [Online; accessed 03/03/2020].

[11] D. Krajzewicz, M. Heinrich, M. Milano, P. Bellavista, T. Stützle, J. Härri, T. Spyropoulos, R. Blokpoel, S. Hausberger, and M. Fellendorf, “Colombo: Investigating the potential of v2x for traffic management purposes assuming low penetration rates,” 2013.

[12] A. Stevanovic, J. Stevanovic, and C. Kergaye, “Comparative evaluation of benefits from traffic signal retiming and green light optimized speed advisory systems,” 06 2013.

[13] V. Knoop, "Traffic flow modelling & control." November 2019.

[14] J. Rawlings and D. Mayne, *Model Predictive Control: Theory and Design.* 01 2009.

[15] B. Ye, W. Wu, K. Ruan, L. Li, T. Chen, H. Gao, and Y. Chen, "A survey of model predictive control methods for traffic signal control," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 3, pp. 623–640, 2019.

[16] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Yibing Wang, "Review of road traffic control strategies," *Proceedings of the IEEE*, vol. 91, no. 12, pp. 2043–2067, 2003.

[17] R. Negenborn, Z. Lukszo, and H. Hellendoorn, *Intelligent Infrastructures.* Intelligent Systems, Control and Automation: Science and Engineering, Springer, 2010.

[18] L. Klein, M. Mills, and D. Gibson, *Traffic Detector Handbook: Third Edition - Volume I.* Federal Highway Administration, Georgetown Pike, 2006.

[19] Vialis, *V-Log protocol en definities. Versie 2.7.0.* 07 2020.

[20] subWG NL profile, "Spat data, dutch profile version 2.1.a," tech. rep., Talking Traffic, 2019.

[21] V. P. Siemens, "Aantal active vri's," 2022. [Online; accessed 13/05/2020].

[22] B. S. Kerner, C. Demir, R. G. Herrtwich, S. L. Klenov, H. Rehborn, M. Aleksic, and A. Haug, "Traffic state detection with floating car data in road networks," in *Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005.*, pp. 44–49, 2005.

[23] M. P. Hunter, S. K. Wu, H. K. Kim, and W. Suh, "A probe-vehicle-based evaluation of adaptive traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 704–713, 2012.

[24] K. Aboudolas, M. Papageorgiou, A. Kouvelas, and E. Kosmatopoulos, "A rolling-horizon quadratic-programming approach to the signal control problem in large-scale congested urban road networks," *Transportation Research Part C: Emerging Technologies*, vol. 18, pp. 680–694, 10 2010.

[25] D. Ni, *Traffic Signal Coordination*, pp. 291–326. Cham: Springer International Publishing, 2020.

[26] D. Roberson, "Transyt: A traffic network study tool," *Road Research Laboratory Report*, 1969.

[27] E. S. Prassas and R. P. Roess, *Fundamental Concepts for Interrupted Flow*, pp. 17–36. Cham: Springer International Publishing, 2020.

[28] F. Luyanda, D. Gettman, L. Head, S. Shelby, D. Bullock, and P. Mirchandani, "Acs-lite algorithmic architecture: Applying adaptive control system technology to closed-loop traffic signal control systems," *Transportation Research Record*, vol. 1856, pp. 175–184, 01 2003.

[29] R. B. Allsop, “Sigcap: A computer program for assessing the traffic capacity of signal-controlled road junctions,” *Traffic Engineering Control*, vol. 17, pp. 338–341, 1976.

[30] D. Ni, *Traffic Signal Coordination*, pp. 291–326. Cham: Springer International Publishing, 2020.

[31] D. Hale, *Traffic Network Study Tool, TRANSYT-7F.* RRL report, McTrans Center, University of Florida, Gamesville, Florida, 2006.

[32] D. I. Robertson and R. D. Bretherton, “Optimizing networks of traffic signals in real time-the scoot method,” *IEEE Transactions on Vehicular Technology*, vol. 40, no. 1, pp. 11–15, 1991.

[33] P. Lowrie, “The sidney co-ordinated adaptive traffic system: Principles, methodology, and algorithms,” *IEE Conference on Road Traffic Signaling*, vol. 207, pp. 67–70, 1982.

[34] A. G. Sims and K. W. Dobinson, “The sydney coordinated adaptive traffic (scat) system philosophy and benefits,” *IEEE Transactions on Vehicular Technology*, vol. 29, no. 2, pp. 130–137, 1980.

[35] V. Mauro and C. D. Taranto], “Utopia,” *IFAC Proceedings Volumes*, vol. 23, no. 2, pp. 245 – 252, 1990. IFAC/IFIP/IFORS Symposium on Control, Computers, Communications in Transportation, Paris, France, 19-21 September.

[36] C. Diakaki, V. Dinopoulou, K. Aboudolas, M. Papageorgiou, E. Ben-Shabat, E. Seider, and A. Leibov, “Extensions and new applications of the traffic-responsive urban control strategy: Coordinated signal control for urban networks,” *Transportation Research Record*, vol. 1856, pp. 202–211, 01 2003.

[37] C. Diakaki, M. Papageorgiou, and K. Aboudolas, “A multivariable regulator approach to traffic-responsive network-wide signal control,” *Control Engineering Practice*, vol. 10, no. 2, pp. 183 – 195, 2002.

[38] M. Burger, M. van den Berg, A. Hegyi, B. De Schutter, and J. Hellendoorn, “Considerations for model-based traffic control,” *Transportation Research. Part C: Emerging Technologies*, vol. 35, no. October, pp. 1–19, 2013. Harvest.

[39] S. Hoogendoorn and P. Bovy, “State-of-the-art of vehicular traffic flow modeling,” *J. Syst. Cont. Eng.*, vol. 215, pp. 283–303, 06 2001.

[40] F. Kessels, J. Lint, C. Vuik, and S. Hoogendoorn, “Genealogy of traffic flow models,” *EURO Journal on Transportation and Logistics*, vol. 4, pp. 1–29, 01 2014.

[41] R. R. Negenborn and J. M. Maestre, “Distributed model predictive control: An overview and roadmap of future research opportunities,” *IEEE Control Systems Magazine*, vol. 34, no. 4, pp. 87–97, 2014.

[42] S. Grammatico, “Lecture notes model predictive control,” 2019. [Online; accessed 03/03/2020].

[43] K. Aboudolas, M. Papageorgiou, and E. Kosmatopoulos, “Store-and-forward based methods for the signal control problem in large-scale congested urban road networks,” *Transportation Research Part C: Emerging Technologies*, vol. 17, no. 2, pp. 163 – 174, 2009. Selected papers from the Sixth Triennial Symposium on Transportation Analysis (TRISTAN VI).

[44] S. Lin, B. De Schutter, Y. Xi, and J. Hellendoorn, “A simplified macroscopic urban traffic network model for model-based predictive control,” *IFAC Proceedings Volumes*, vol. 42, no. 15, pp. 286 – 291, 2009. 12th IFAC Symposium on Control in Transportation Systems.

[45] C. F. Daganzo, “The cell transmission model, part ii: Network traffic,” *Transportation Research Part B: Methodological*, vol. 29, no. 2, pp. 79 – 93, 1995.

[46] G. C. Gazis and R. B. Potts, “The over-saturated intersection,” *Proceedings of the Second International Symposium on the Theory of Traffic Flow*, pp. 221–237, 1963.

[47] M. J. Lighthill and J. B. Whitham, “On kinematic waves. i. flow movements in long rivers. ii. a theory of traffic flow on long crowded roads.,” *Proceedings Royal Society*, vol. A229, pp. 281–345, 1955.

[48] A. Muralidharan and R. Horowitz, “Computationally efficient model predictive control of freeway networks,” *Transportation Research Part C: Emerging Technologies*, vol. 58, 04 2015.

[49] H. K. Lo, “A novel traffic signal control formulation,” *Transportation Research Part A: Policy and Practice*, vol. 33, no. 6, pp. 433 – 448, 1999.

[50] B. D. Schutter and W. Heemels, *Lecture notes "Modeling and Control of Hybrid Systems"*. 12 2015.

[51] D. Liu, W. Yu, S. Baldi, J. Cao, and W. Huang, “A switching-based adaptive dynamic programming method to optimal traffic signaling,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 11, pp. 4160–4170, 2020.

[52] K. M. Ng, M. B. I. Reaz, and M. A. M. Ali, “A review on the applications of petri nets in modeling, analysis, and control of urban traffic,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 858–870, 2013.

[53] R. Negenborn, B. De Schutter, and J. Hellendoorn, “Multi-agent model predictive control for transportation networks: Serial versus parallel schemes,” *Engineering Applications of Artificial Intelligence*, vol. 21, no. 3, pp. 353–366, 2008.

[54] P. R. Mendes, J. M. Maestre, C. Bordons, and J. E. Normey-Rico, “A practical approach for hybrid distributed mpc,” *Journal of Process Control*, vol. 55, pp. 30–41, 2017.

[55] M. Mehrabipour and A. Hajbabaie, “A cell-based distributed-coordinated approach for network-level signal timing optimization,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 32, no. 7, pp. 599–616, 2017.

[56] T. van den Boom and B. D. Schutter, *Lecture notes "Optimization in Systems and Control"*. 09 2018.

[57] R. T. van Katwijk, “Multi-agent look-ahead traffic-adaptive control.” Januari 2008.

[58] Y. Liu, J. Guo, and Y. Wang, “Vertical and horizontal queue models for oversaturated signal intersections with quasi-real-time reconstruction of deterministic and shockwave queueing profiles using limited mobile sensing data,” *Journal of Advanced Transportation*, 2018.

[59] S. Chen and D. J. Sun, “An improved adaptive signal control method for isolated signalized intersection based on dynamic programming,” *IEEE Intelligent Transportation Systems Magazine*, vol. 8, no. 4, pp. 4–14, 2016.

[60] J. Mathew, H. Thomas, A. Sharma, L. Devi, and L. Rilett, “Studying platoon dispersion characteristics under heterogeneous traffic in india,” *Procedia - Social and Behavioral Sciences*, vol. 104, pp. 422–429, 2013. 2nd Conference of Transportation Research Group of India (2nd CTRG).

[61] Z. Yao, “High-resolution traffic flow prediction model based on deep learning,” vol. 1, 01 2019.

[62] G. Thom, “Enabling glosa for on-street operating traffic light controllers.” April 2020.

[63] H. Noorhan, “An intelligent traffic flow progression model for predictive control applications.” December 2017.

[64] J. C. van Senden, “Enabling advanced driver assistance systems with predictive signalized intersection control using lstm networks.” May 2018.

[65] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[66] N. H. Gartner, F. J. Pooran, and C. M. Andrews, “Implementation of the opac adaptive control strategy in a traffic signal network,” in *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No.01TH8585)*, pp. 195–200, 2001.

[67] J. Henry, J. Farges, and J. Tuffal, “The prodyn real time traffic algorithm,” in *Control in Transportation Systems* (D. Klamt and R. Lauber, eds.), pp. 305 – 310, Pergamon, 1984.

[68] S. Sen and L. Head, “Controlled optimization of phases at an intersection,” *Transportation Science*, vol. 31, pp. 5–17, 02 1997.

[69] NEMA, “Nema standards publication ts 2-v02.06 traffic controller assemblies with ntcip requirements.,” *National Electrical Manufacturers Association*, 2003.

[70] J. Haddad, B. De Schutter, D. Mahalel, I. Ioslovich, and P.-O. Gutman, “Optimal steady-state control for isolated traffic intersections,” *Automatic Control, IEEE Transactions on*, vol. 55, pp. 2612 – 2617, 12 2010.

[71] B. De Schutter and B. De Moor, “Optimal traffic light control for a single intersection,” *European Journal of Control*, vol. 4, no. 3, pp. 260 – 276, 1998.

[72] B. De Schutter, “Optimizing acyclic traffic signal switching sequences through an extended linear complementarity problem formulation,” *European Journal of Operational Research*, vol. 139, no. 2, pp. 400 – 415, 2002. EURO XVI: O.R. for Innovation and Quality of Life.

[73] S. Lämmer, “Selbst-gesteuerte lichtsignalanlagen im praxistest,” *Strabenverkehrstechnik*, vol. 60, 2016.

[74] S. Lämmer, R. Donner, and D. Helbing, “Anticipative control of switched queueing systems,” *Physics of Condensed Matter*, vol. 63, pp. 341–347, 06 2008.

[75] S. G. Shelby, “Single-intersection evaluation of real-time adaptive traffic signal control algorithms,” *Transportation Research Record*, vol. 1867, no. 1, pp. 183–192, 2004.

[76] S. R. Sunkari, R. Engelbrecht, and K. N. Balke, “Evaluation of advance coordination features in traffic signal controllers,” 2004.

[77] K. V. Katsaros, R. Kernchen, M. Dianati, and D. Rieck, “Performance study of a green light optimized speed advisory (glosa) application using an integrated cooperative its simulation platform,” *2011 7th International Wireless Communications and Mobile Computing Conference*, pp. 918–923, 2011.

[78] *Internal communication within Yunex/Siemens Mobility*, 2022.

[79] M. Berg, A. Hegyi, B. Schutter, and H. Hellendoorn, “Integrated traffic control for mixed urban and freeway networks: A model predictive control approach,” *European Journal of Transport and Infrastructure Research*, vol. 7, no. 3, 2007.

[80] S. Göttlich, M. Herty, and U. Ziegler, “Modeling and optimizing traffic light settings in road networks,” *Computers Operations Research*, vol. 55, pp. 36–51, 2015.

[81] D. Zhao, Y. Dai, and Z. Zhang, “Computational intelligence in urban traffic signal control: A survey,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 485–494, 2012.

[82] S. Lin, “Efficient model predictive control for large-scale urban traffic networks,” 2011.

[83] A. Bemporad and M. Morari, “Control of systems integrating logic, dynamics, and constraints,” *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[84] B. D. Schutter and W. Heemels, *Model Building in Mathematical Programming, 5th Edition.* 03 H. Paul Williams.

[85] I. Shatnawi, P. Yi, and I. Khliefat, “Automated intersection delay estimation using the input–output principle and turning movement data,” *International Journal of Transportation Science and Technology*, vol. 7, no. 2, pp. 137–150, 2018.

[86] B. Kouvaritakis, M. Cannon, M. Müller, M. Reble, and F. Allgöwer, “Cooperative control of dynamically decoupled systems via distributed model predictive control,” *International Journal of Robust and Nonlinear Control*, vol. 22, 08 2012.

[87] Aimsun, *Aimsun Next 8.4 User’s Manual.* Barcelona, Spain, aimsun next 8.4.4 ed., Accessed on: Month, Day, Year 2021. [In software].

[88] N. H. Gartner and C. Stamatiadis, *Traffic Networks, Optimization and Control of Urban Traffic Networks*, pp. 9470–9500. New York, NY: Springer New York, 2009.

[89] T. R. Board, *Highway Capacity Manual 6th Edition: A Guide for Multimodal Mobility Analysis.* Washington, DC: The National Academies Press, 2016.

[90] W. E. Hart, J.-P. Watson, and D. L. Woodruff, "Pyomo: modeling and solving mathematical programs in python," *Mathematical Programming Computation*, vol. 3, no. 3, pp. 219–260, 2011.

[91] I. I. Cplex, "V12. 1: User's manual for cplex," *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.

[92] D. Helbing, J. Siegmeier, and S. Lämmer, "Self-organized network flows," *Networks and Heterogeneous Media*, vol. 2, 03 2007.

[93] S. Lämmer and D. Helbing, "Self-stabilizing decentralized signal control of realistic, saturated network traffic," *Technical Report*, 01 2011.

[94] S. Lämmer and M. Treiber, "Self-healing networks - gridlock prevention with capacity regulating traffic lights," in *2012 IEEE Sixth International Conference on Self-Adaptive and Self-Organizing Systems Workshops*, pp. 61–65, 2012.

[95] A. Wood, *Rabbit MQ: For Starters.* North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2016.