Model-predictive control of the lane configuration at signalized intersections

F.J.C. Anema

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Model-predictive control of the lane configuration at signalized intersections

Master of Science Thesis

For the degree of Master of Science in Mechanical Engineering at Delft University of Technology

F.J.C. Anema

September 15, 2015

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

The work in this thesis was supported by TrafficQuest and TNO. Their cooperation is hereby gratefully acknowledged.

Abstract

The topic of this study is the control of signalized intersections, specifically the control of *switching lanes*, which are approach lanes of which the traffic assignment can be changed. When the demand at an intersection varies considerably, being able to change the lane configuration can reduce delay, especially in saturated conditions. Currently five intersections in the Netherlands already have a single switching lane that switches at two fixed times per day. However, using multiple switching lanes and being able to switch when needed may further reduce delay.

Previous work on this topic was fairly limited in scope, did not use predictive control and did not integrate the control of traffic signals and switching lanes. In this study two model-predictive controllers for integrated traffic signal and switching lane control of a single intersection were designed. The first aims to minimize delay while the second aims to minimize the queue length. The main differences are that the first features a more elaborate prediction model and uses longer time step durations, while the second emphasizes faster measurements and computation and uses shorter time step durations.

Both controllers were evaluated using PTV Vissim, a microscopic traffic simulation program. Fictional data consisting of simple signals as well as real-world data were used. In the fictional data tests, compared to the static situation, switching reduced delay for both controllers. In the real-world data tests no significant difference between static and switching was found. A state-of-the-art signal controller used for comparison performed equal or better in all cases.

Considerable insight has been gained concerning the working of a dynamic lane configuration. The potential for delay reduction by using a predictive controller has been shown, but more research is needed to make the controllers suitable for handling real-world situations. The most important recommendation for future work is to improve the signal control part of both controllers.

Table of Contents

List of Figures

List of Tables

Acknowledgements

First, I want to sincerely thank my daily supervisor dr. ir. Ronald van Katwijk for his energetic support during the past year. I really enjoyed our talks, they were always helpful and sparked new insights, and so I cannot imagine having done this research without your help. Your door was always open and you were always enthusiastic, which I deeply appreciate.

I also want to express gratitude to my advisor prof. dr. ir. Bart De Schutter. Your guidance and support have been great, our meetings always motivated me. I also earnestly appreciate your help in proofreading this thesis.

My sincere thanks also goes to dr. ir. Henk Taale and Anahita Jamshidnejad for taking the time in their busy schedules to be a member of the graduation committee.

I am also grateful for the support of my friends and family, some of whom even helped proofread this thesis. My parents deserve a special expression of thankfulness, because they have always supported me.

I would like to conclude these acknowledgments with stating that I hope that this thesis shows I now know what the numbers mean.

Delft, University of Technology F.J.C. Anema
September 15, 2015

Chapter 1

Introduction

Futuristic visions from the 1950's predicted that by now we would fly in cars through our cities, without congestion or delay. This prediction has turned out to be too optimistic as we still drive our cars. We still have red lights to wait at. Although the study of intelligent vehicles is an exciting and upcoming field, promising to drastically change the way we travel in the urban environment, the reality on the road is still somewhat similar to 60 years ago, and therefore in need of intelligent solutions.

In a dense urban environment a significant part of the delay experienced by drivers is caused by intersections. That is why for several decades a lot of research has focused on improving the throughput of intersections. The part of the research that focuses on control has one thing in common: it sticks to the degrees of freedom traffic signals have. Control methods for signalized intersections currently have two degrees of freedom: the order in which the signals get green and the duration each signal is green.

This thesis aims to extend these two degrees of freedom with a third: the lane configuration. This term describes the way the flows of traffic with different directions are assigned to each approach lane. The lane configuration does not have to be fixed, but can instead by dynamic. There are already a few intersections where dynamic lane use is implemented, though small-scale, switching at fixed times, and un-evaluated.

As cars become more connected, flexible use of infrastructure becomes more feasible, and dynamic lane use has interesting but until now neglected promise. This thesis will add knowledge to this unexplored research field and for the first time add the full possibilities of lane configuration control to the already powerful capabilities of traffic signal control.

This thesis will explore this new approach of combining lane and signal control, with the aim of reducing delay for urban intersections. This is a small step towards travel time reduction, less noise and emissions, and in general more livable cities.

1-1 Motivation

The motivation for this study is to further reduce the delay experienced by drivers at signalized intersections. The assumption is that when demand for a certain direction is significantly higher than for other directions, adding more lane capacity to that direction is beneficial in terms of delay reduction. To motivate this assumption three short examples will be given. These will show that using a dynamic lane configuration where the combination of lanes and signals can be controlled can result in a reduction of delay.

In these examples in each situation a and situation b the same flow is used, indicated by the number of vehicles in the image. Time losses are neglected so that the *phase order*, which is the order in which the signals get green, is not of influence and that the optimal signal timing can be calculated easily. This makes sure that the two current ways of intersection control, phase order and signal timing, are already optimal and it thus comes down to what lane control can achieve.

In the examples the practical concept of a *switching lane* is introduced. This term is used to describe a lane on the approach of an intersection that can switch its assignment. It can for example be used by left-turning or right-turning traffic, depending on which movement has more demand.

In the figures the green time of each movement is indicated, which is calculated assuming a *saturation flow* of $s = 1800\,\mathrm{veh\,h^{-1}}$. In these examples the smallest possible time in which all traffic can be serviced is called the *cycle time* T_C. Since the demand in each example a and b is the same, a lower cycle time is an indication of a more efficient process with less delay. This is an intuitive truth that can be verified using Webster's famous simplified delay formula [1]. By comparing the cycle times the effect of a switching lane can be assessed.

Example 1 Consider the image in Fig. 1-1. In situation 1a, for traffic originating from the south, there are two approach lanes for going right and one for going left. It takes 10 s to fully drain the queue of traffic going left. Traffic on the other two legs is light and takes 2 s each. With a cycle time of $T_\mathrm{C} = 12\,\mathrm{s}$ all traffic can be serviced, first 10 s for the traffic from the south followed by 2 s for the traffic from the east and west.

Suppose that the middle lane of the south approach can switch its assignment, that it is a *switching lane*. In situation 1b the assignment of the middle lane on the south leg is switched from right to left. Now traffic going left takes 6 s to drain, resulting in a lower cycle time of $T_\mathrm{C} = 8\,\mathrm{s}$. As stated before a lower cycle time while processing the same demand means traffic will experience less delay. This example shows a situation where, next to an optimal phase order and signal timing, lane switching can reduce delay even further.

Example 2 A suspicion following from example 1 could be that the lane configuration in situation 1b is just more optimal than the one in situation 1a for any demand. This is

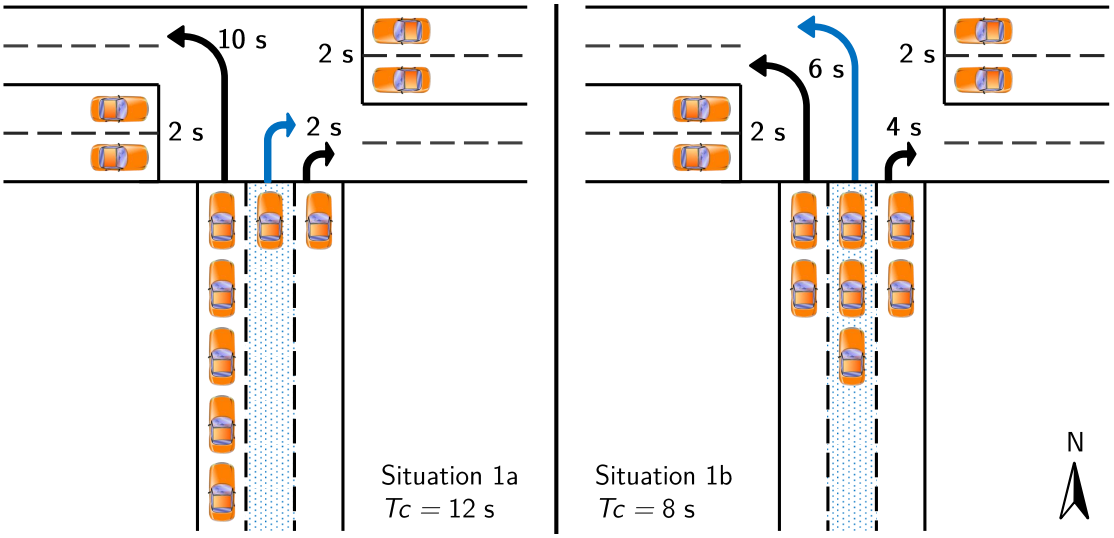


Figure 1-1: First example of the benefit of a switching lane

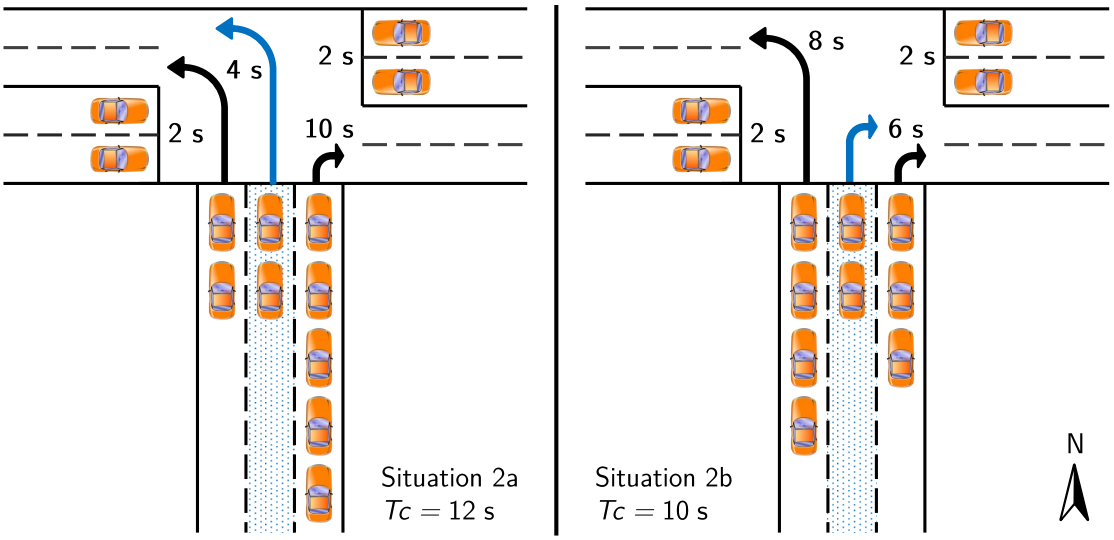


Figure 1-2: Second example of the benefit of a switching lane

not true, as will be shown in the second example in Fig. 1-2. Situation 2a has the same lane configuration as in situation 1b, but demand has changed such that much more traffic wants to turn right instead of left. Switching the assignment of the middle lane back to the one in situation 1a reduces the cycle time by 2 s and thus also reduces delay. This example, in combination with the first example, shows that with changing demand it can be beneficial to change the assignment of a switching lane back and forth.

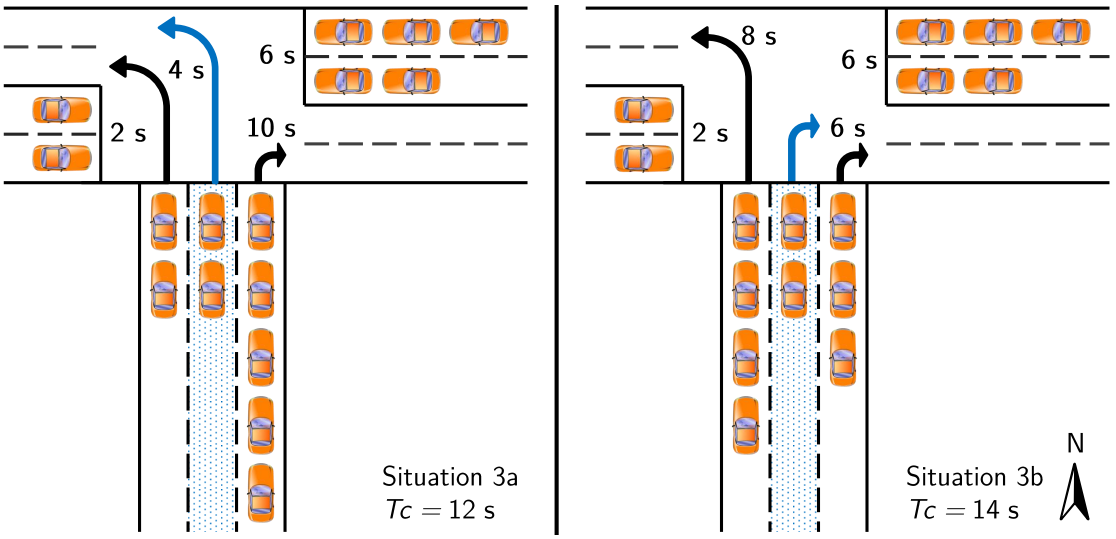


Figure 1-3: Third example, where applying a switch seems beneficial but is in fact not

Example 3 Example 1 and example 2 could be interpreted as proof that the optimal lane configuration can be found by equalizing the flows over the available lanes, or by equalizing the green times of the movements. The third example in Fig. 1-3 will show this is not true. In this example the traffic originating from the south is the same as in example 2, but the amount of traffic from the east is increased. The cycle time now increases when the lane is switched in situation b, resulting in more delay, even though compared to situation a the flow is distributed more equally over the lanes and the green times are more close to their mean. This shows that switching a lane to equalize the flows or green times not necessarily leads to less delay.

In example 2 and 3 the amount of traffic on the south leg is the same, but switching was only beneficial in example 2, because in example 3 the demand on another leg changed. This shows that the amount of traffic on conflicting movements has an impact on the effect of a switching lane. It is not sufficient to only consider the movements that use a switching lane, instead the whole intersection has to be taken into account when determining if switching is beneficial or not.

Conclusions These simple examples give a couple of important insights:

- Switching lanes can decrease delay.
- With optimal phase order and signal timings, a switching lane can reduce delay even further.
- Using a switching lane to equalize the flow on each lane does not necessarily lead to less delay.
- Using a switching lane to equalize green times does not necessarily lead to less delay.
- The whole intersection needs to be taken into account when determining the optimal lane configuration.

1-2 Goal

This study has two main goals. The first is to develop two controllers for a dynamic lane configuration system. The second is to evaluate these controllers and produce insights on how a dynamic lane configuration works.

There currently is almost no literature available on the control of lane configurations and so there is no precedent on how such a controller should be build. With two controllers instead of one, each with a different approach, results can be compared, which will provide more insights on how both control methods interact with the dynamic lane configuration.

Given the exploratory nature of this study, the questions that this study hopes to answer are fairly broad:

- What type of controller is suitable for handling a dynamic lane configuration system?
- What is the impact of a switching lane on the whole intersection?
- Should the controller both consider signal and lane control in an integrated fashion?
- Can the controller work online, operating while the intersection is being used?
- What is the effect of using prediction?
- Does applying the controller result in an optimal lane configuration?

1-3 Scope

To clarify the scope of this research some of the aspects that will not be considered are listed here.

- The research will be limited to isolated intersections, ignoring networking effects. Urban traffic networks are a very interesting field of study but would broaden the scope too much.
- The number of available lanes will be considered fixed. This means that each approach and each exit has a fixed number of lanes. Instead what *can* change dynamically is which movement is allowed on which approach lane.
- A dynamic lane configuration will have consequences for the behavior of drivers. Driver acceptance is an aspect that should be studied but it will be left as future work.
- Some information on the practical implementation of a lane switching mechanism will be given, but it will not be discussed in depth. Furthermore the monetary costs of such a system will not be considered.
- Demand prediction is needed for the methods in this study, but how that should be done is not part of this study. It is an extensive research field of its own and it would be too much to include it here. Instead a referral to relevant research will be given.
- A fixed phase order will be used. A test with two intersections and a range of demands and lane configurations showed that for each intersection a single optimal phase order could be selected. In light of this the phase order is fixed to reduce complexity. However, it is possible to include the determination of the phase order in the methods developed in this thesis, and it could certainly be included in future research.
- Pedestrians and bicycles will not be included. This way the effects of the lane switching mechanism can be seen more clearly.
- Only passenger cars will be considered. The effect of trucks, public transport, and other non-standard vehicles is neglected. In reality they have a great impact on the working of an intersection and they should therefore be included in future research.

1-4 Outline

Theoretical background In Chapter 2 the knowledge on dynamic lane use that forms the basis for this study will be presented. In the Netherlands there are five intersections

where a switching lane mechanism has already been implemented. Two of these practical cases will be discussed. What follows is a literature study on static lane configuration design methods, one of which was used as a basis for the work in this study, and on dynamic lane configurations, of which very few studies exist.

Design In Chapter 3 the two controllers that have been developed in this study will be introduced. Their description will be preceded by a part on the development considerations.

Method Chapter 4 is about the test setup with which the two controllers have been evaluated. The chapter describes the intersections and demand data that were selected, and the way microsimulation was used to perform the experiments.

Results In Chapter 5 the results from the evaluation with fictional demand data will be presented and analyzed. Chapter 6 shows and analyzes the results from the evaluation with real-world demand data. Chapter 7 features a comparison of the results of the two controllers with a state-of-the-art signal controller.

Discussion, conclusions and recommendations Chapter 8 will feature a discussion of the presented work. Then in Chapter 9 this thesis will be concluded with a small summary, general conclusions based on the questions posed in this introduction, and finally recommendations for further research.

Chapter 2

Existing knowledge on lane configuration optimization

In this chapter the existing knowledge on the optimization of lane configurations will be presented. It will start with two practical cases of dynamic lane use at signalized intersections in Section 2-1. Next a brief review of the literature on lane configuration optimization will be presented. An important part of the papers on lane configuration optimization considers the static case in which an optimal static configuration is the goal. Literature on this static case will be discussed first in Section 2-2. Afterwards in Section 2-3 the way these models were used for online control of the lane configuration will be described.

2-1 Dynamic lane use at intersections in practice

The dynamic use of lanes is not very common at intersections. At almost all intersections in the Netherlands the movements allowed on each lane are fixed. There are however five examples of intersections where on a single lane the type of movement can be switched. This is done mostly to accommodate a tidal-like rush hour peak and therefore the lane is switched at fixed times. Of these lane-switching intersections two cases will be discussed: the first is located in Utrecht at the A12 and the Europalaan, and the second is located in Deventer at the N344 and the Zweedsestraat. The three intersections that will not be discussed are located in Bunnik at the A12 and N229, in Beverwijk at the A22 and N197, and in Nieuwegein at the N408 an the Martinbaan.

2-1-1 Intersection A12–Europalaan, Utrecht

A large intersection in Utrecht links the on- and off-ramps of the A12 freeway with the Europalaan, an important urban artery. See Fig. 2-1 for an overview. The municipal traffic office analyzed the flows on the north and westbound directions originating from the south. They compared the demands during morning and evening rush hours and found a tidal-like change. That is why they decided to try to use a lane in a flexible way, switching its assignment twice a day. DHV, an engineering consultancy, was approached to help write the signal software. No microsimulation was done before implementation, but the system was tested in a numerical way using COCON, a traffic signal design program.

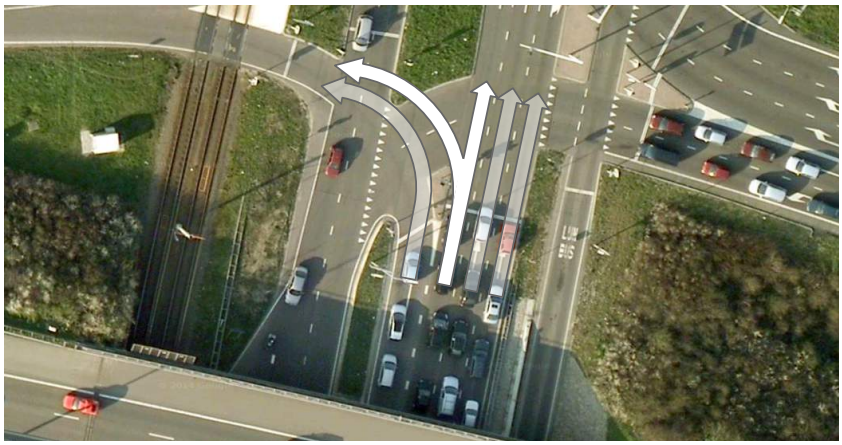


Figure 2-1: Aerial overview of the old situation at the intersection A12–Europalaan in Utrecht. The white arrow indicates the lane that is now in use as a switching lane.

Figure 2-2: Dynamic message signs indicate the lane use at the intersection A12–Europalaan in Utrecht

The switching lane is implemented using electronic as well as mechanical variable message signs and two signal heads, one for each movement, of which only one can be active at the same time. In October 2014 electronic message signs where also added to the other lanes to make the situation more clear to drivers. There is no pavement marking on the switching lane. Part of the hardware that is implemented at the intersection is shown in Fig. 2-2.

No formal analysis was performed after implementation. Marcel de Lange, traffic engineer at the municipality of Utrecht, did however state in a personal interview that the use of the switching lane is perceived as beneficial. During the evening peak the left-going lane is no longer over-saturated and spillback is reduced. Sometimes drivers do go straight ahead when the switching lane is in use for left-turning traffic. Because of conflicting movements this is a dangerous situation, although no incidents are known to Marcel.

2-1-2 Intersection Holterweg–Zweedsestraat, Deventer

The intersection Holterweg–Zweedsestraat in Deventer is part of a larger junction linking two urban highways and a larger urban road. An overview of the intersection is shown in Fig. 2-3. A clear tidal-like flow pattern was seen in the demand for left-turns and straight-ahead movements on the westbound direction. DHV developed the necessary software to turn the middle one of three lanes at that direction into a switching lane. No microsimulation was performed before implementation, but the system was numerically analyzed in COCON.

Figure 2-3: Aerial overview of the old situation at the intersection Holterweg–Zweedsestraat in Deventer. The white arrow indicates the switching lane that is now in use.

The implementation features double signal heads of which only one can be in use at a time. This is shown in Fig. 2-4. Switching happens twice a day at fixed times. First the lane is closed by electronic message signs. When the detector loops in the road confirm that the lane is empty the operating mode can be switched.

There has been no official evaluation after implementation. Klaas-Jan op den Kelder, a traffic engineer at DHV, stated in a personal interview that the system works as intended as the intersection has an improved throughput. In the beginning road users did not understand the switching lane and stayed on the far left lane, but gradually the switching lane is used more often. It still happens that people drive straight ahead when the lane is in use for left-turns, but fortunately there are no conflicting movements with which a collision could occur. According to Klaas-Jan the implementation of the switching lane has not reduced safety at the intersection.

Figure 2-4: Double signal head for the switching lane at the intersection Holterweg–Zweedsestraat in Deventer

2-1-3 Conclusions on dynamic lane use at intersections in practice

In practice only five intersections in the Netherlands have a dynamic use of lanes. In all cases only one approach lane is used as a switching lane. The designation of that lane switches at fixed times. In a sample of two of the intersections it was found that no written evaluation is available.

A single switching lane can be implemented in practice. For an overview of possible hardware the reader is directed to [2, Ch. 7]. For anything more than a single switching lane, such as multiple switching lanes, or switching lanes extending over multiple intersections, there is no precedent.

There is no information available on why there are not more cases of dynamic lane use at intersections in practice. There are several possible factors: the costs of such a system, perhaps also a lack of intersections which have a demand profile that would

clearly benefit from a switching lane, the lack of information on this topic available to traffic engineers, and the expected difficulties with road user acceptance.

Based on the current state of dynamic lane use at intersections what still needs to be investigated is the use of multiple switching lanes at a single intersection, the effect of a switching lane over multiple intersections, and how these measures should be implemented in practice.

2-2 Literature on optimal static lane configuration design

In a Dutch guide for urban traffic methodologies the design of an intersection is described in a certain number of steps [3, p. 1101]. First flows are estimated, then a preliminary design is made. The feasibility of the design is analyzed by calculating the exit capacities and cycle times for the configuration. If needed the design is edited, else the traffic engineer can continue with determining the signal plan. This methodology shows that the design of the lane configuration and the signal plan are separate steps in the procedure. The intersection layout is considered an input for the signal planning. In practice the lane configuration is based on the flow data and the experience of the traffic engineer.

What if instead an optimization algorithm is used that automatically designs an optimal intersection layout? Some research has been done in this field, first only considering the lanes and signal phases, later also integrating signal timing, and the number of lanes. The methods of these papers and their merits will be discussed in this section.

2-2-1 Lam et al. (1997)

Category	Optimization includes	Optimization criteria	Problem type	Solving method
Static	Lanes and signal phase structure	Flow ratio	MILP	Three stages. Generic MILP solver.

Lam et al. were one of the first to integrate the design of the intersection and signal plan in one optimization scheme [4].

Commonly for intersections variables are based on movements or links instead of lanes. In contrast, this paper models an intersection by defining variables for each lane. Other papers that will be presented afterwards follow this idea, that is sometimes called the *lane-based method*.

The model uses integer variables to indicate the movement and lane use patterns. Traffic belonging to a single movement is assumed to distribute evenly over the available lanes, a concept called *flow equalization*. Cyclists are not considered but pedestrians are.

The optimization is done in three separate stages, which was necessary in order to reduce the computation time to an acceptable range. All three problems are of a Mixed Integer Linear Programming (MILP) kind and can be solved with any applicable optimization algorithm. In the paper the IMSL Numerical Library is named as an example of a possible solution method.

First only vehicular traffic is considered. As objective function the sum of the flow ratios of all lanes is used and as such signal timings do not have to be calculated. This step results in a lane configuration and a phase structure. Secondly the pedestrian movements are allocated to the previously determined phases. This is done by maximizing the sum of the binary right-of-way variables of the pedestrians. If not all pedestrian movements can be allocated an all-red phase will be added in which all pedestrian movements are placed. Third the sequence of the phases is reordered. This is done by maximizing a reward function, where a reward is given if the ordering allows movements to continue in consecutive phases.

The evaluation of the method is based on data collected at six intersections in Shenzhen, China. The existing and the integrated designs are compared using TRANSYT-7F, a mesoscopic traffic simulation and signal optimization program. The integrated design performs much better than the existing design, reducing the average delay by almost 73 % and the number of stops by 25 %.

Analysis

The biggest drawback of this method is the splitting of the problem in three serial steps. Because pedestrians are considered separately often an exclusive pedestrian phase is added by the algorithm during the second step. In the case study for each intersection a separate all-pedestrian phase was added by the method. For pedestrians this can be beneficial for wide intersections with low traffic volume [5], but it is in general not applied in the Netherlands. There are however a few intersections where an exclusive cyclist phase is used, but overall this strategy is not advised [6, p. 290]. Reasons are the increase in waiting time for motorized traffic, the inability to include pedestrians in this all-green phase because of conflicts between cyclists and pedestrians, the decrease in safety because of conflicting cyclist streams moving at the same time, and the risk of red light negation by vehicles when there are only a few cyclists.

A limitation of the model is that signal timing is not included in the optimization. Only the movements per phase and the phase order are considered. Including the cycle time and phase timing in the optimization may provide overall better results.

The choice for flow ratio as objective function is reasonable since in a way it maximizes capacity. But it does not explicitly minimize the factors most visible to road users: delay and the number of stops. Including these as objective variables may reduce robustness but increase user experience.

2-2-2 Wong & Wong (2003): linear cost function

Category	Optimization includes	Optimization criteria	Problem type	Solving method
Static	Lanes, signal timing	Capacity, cycle time	MILP	Branch-and-bound + LP

In the spirit of the paper by Lam et al. a lane-based optimization method was presented in [7]. Compared with its predecessor this paper unifies the optimization of traffic and pedestrian movements. It also considers cycle time minimization instead of only capacity maximization. Another difference is that signal timing calculations are included.

The model used is somewhat similar. Again traffic of each movement is assumed to divide equally over the available lanes. For each lane three binary variables describe to which legs traffic may travel. Signal timing is included, the model finds the start and duration of green for each movement based on the conflicts and their clearance times.

As objective criteria capacity maximization and cycle length minimization are used, but not simultaneously. Delay minimization is omitted since it would make the problem non-linear. Capacity is defined as a demand matrix multiplier, assuming that flows would increase in proportion to the demand matrix, while keeping a maximum acceptable degree of saturation. The cycle time is minimized directly and has a lower and an upper constraint. Both problems are of the MILP kind and can be solved with any suitable method, such as branch-and-bound combined with a Linear Programming (LP) solver.

The method is demonstrated by calculating the optimal lane configuration and signal plan for an isolated intersection with hypothetical traffic demand. Computing time was on average 20 s on a Pentium III 600 MHz computer. The demonstration of the method is based on fictional demand data and furthermore no comparison with other methods is made.

Analysis

The framework presented in the paper is more useful than the one by Lam et al. discussed in Section 2-2-1, because it does its optimization in one stage and also incorporates signal timing. It is flexible, many specifications can be incorporated in the constraints. It is also computationally attractive since it can be solved in reasonable time.

The assumption of flow equalization, the concept that drivers will divide equally over the available lanes in their direction, makes the model easier to handle but less accurate. In practice drivers may prefer one of the available lanes making the queues there longer. This preference can depend on local circumstances such as a lane merging just after the intersection.

The choice of objective criteria is limited. Capacity maximization is useful for increasing the robustness against future demand increases but does not necessarily provide drivers with a shorter travel time. For area traffic control it is useful to know the lower bound on the cycle length, but in general its minimization does not provide better intersection performance. Delay, number of stops, travel time, or queue lengths would be more interesting objective criteria.

The paper does not make clear how good the solutions of the method are in terms of for example delay compared to other methods, such as manual designs. This makes it difficult to assess the quality of this method.

2-2-3 Wong & Wong (2003): non-linear cost function

Category	Optimization includes	Optimization criteria	Problem type	Solving method
Static	Lanes, signal timing	Delay (Webster's two-term)	Non-convex MINLP	Cutting plane, line search using common demand multiplier and cycle length

In a paper [8] by the same authors the problem is solved using delay as objective function. Webster's two-term delay expression is used, as was done in [9], where the resulting convex objective function was linearized by a piecewise affine approximation. Here this is not possible because the objective function is non-convex and the solution space is more complex. Instead the study tests a cutting plane algorithm for pseudo-convex problems and a heuristic line search algorithm.

For the line search algorithm first a range of cycle times is chosen of which the smallest is determined by cycle time minimization as described in their earlier paper presented in Section 2-2-2. Then for each cycle time, using capacity maximization as optimization criterion, the optimal lane configuration and signal structure and timing is found. Using SIGSIGN, a traffic signal design program, the optimal signal timing is again calculated and the overall delay is found for each of the configurations. The lane configuration with the lowest delay is chosen as the solution.

The method is tested using a standard four-leg isolated intersection with fictional demand. First a lane configuration is devised in the way a traffic engineer would. Then both non-linear solvers are tried. Compared with the manual design the cutting plane algorithm reduced delay with 9.1 %. It needed 156 iterations with a total runtime of 68 h. The line search method reduced delay with 9.4 % and needed 40 min to find this solution.

Analysis

It is interesting that they tried to solve the non-convex MINLP problem directly using the cutting plane algorithm, since these types of problems are hard to solve. It is unfortunate that their computing time became so large.

Their heuristic line search method was much faster than the cutting plane algorithm and delivered a better result as well. Out of a range of solutions the one with the smallest delay is selected, so the quality of the solution depends on the step size in the initial range of cycle times. The method does not optimize based on delay directly, but through a number of steps. In the last step the commercial software package SIGSIGN is used to find the delay. Not much can be found about this software, but it can probably also be done using other methods as well.

2-2-4 Wong, Wong, and Tong (2006): multi-period demand

Category	Feature	Optimization includes	Optimization criteria	Problem type	Solving method
Static	Multiple demand matrices	Lanes, signal timing	Capacity, cycle length, delay	MILP, non-convex MINLP	Linear: not specified. Non-linear: line search

In [10] the same model as in the previous papers is extended to incorporate multiple demand matrices, where previously it used only one demand matrix as input.

The model is mostly similar to the previous work, but now most constraints must be satisfied for multiple demand matrices. As optimization criterion the same non-linear delay as before is used. This non-linear problem is solved by using the heuristic line search method that was explained before. It finds several lane configurations by maximizing a common demand multiplier for a range of cycle times and then selects the configuration with the lowest delay.

In the paper the method is demonstrated with a numerical analysis. No comparison with other methods is made. Three demand matrices are used and therefore the number of variables and constraints is tripled, and since it is a mixed-integer problem the search space grows exponentially. On a Pentium IV 1.8 GHz computer it took 14 min to find the solution to the linear capacity maximization subproblem. The non-linear heuristic method needed 6 h and 20 min to find a solution.

Analysis

What the effect is of using multiple demand matrices in the optimization instead of just one is not presented in this paper. The authors only showed that their method functions.

The same remarks on the heuristic line search method as before can be made. In addition to those remarks, the selection of the configuration with the lowest delay is less straightforward, since the multiple periods add a dimension to the selection problem. The delay of each period has a different order of magnitude, which the authors solved by creating a weighted sum of the delays for the periods of a single cycle time. The size of the weights should be decided by the user and has a big influence on the outcome of this method.

2-2-5 Wong & Heydecker (2011): entry and exit lanes

Category	Feature	Optimization includes	Optimization criteria	Problem type	Solving method
Static	Also optimize split between approach and exit lanes	Lanes, signal timing	Capacity	MILP	CPLEX

The same main author of the previous papers, together with Heydecker, presents in [11] more research on the same topic. Difference with previous research is that the *split* between approach and exit lanes is added to the optimization. This means that the method also determines which part of the available lanes is used as approach and which part is used as exit lanes. Constraints are added to the original model such that a feasible design is achieved. Most notably the number of exit lanes at each leg should be high enough to accommodate the movements using them.

Again a common demand multiplier is used as objective function in order to maximize reserve capacity, resulting in a MILP problem. The method is demonstrated with a numerical test. With 7 lanes per leg the algorithm needs 35 min to find a solution on a Pentium Duo 2.0 computer. Seven manual designs of the same test case were made, these all had a 1 to 14 % lower capacity reserve than the solution found by the algorithm.

Analysis

The extension presented in this paper is a good contribution to the study of lane configuration optimization. The idea to include the split between entry and exit lanes has proven to be potentially beneficial, but increases the problem size and computation time.

Missing from this paper are pedestrian phases, which were included in Wong's previous research. The non-linear delay case is also not considered, probably because of high computation time.

2-2-6 Conclusions on the optimal design of intersections

The base work by Wong & Wong provides a general model of an isolated intersection that can be used to optimize the lane configuration and signal timings. The extensions presented in further papers show the flexibility of the model. There are however some restrictions as well:

- The models presented are only valid for isolated intersections with demand data available.
- An assumption that limits the models accuracy is the flow equalization principle: in reality when more lanes are available to a movement, queues may in general not have the same length.
- Computation time increases rapidly when a non-linear objective function is used or when approach and exit lanes are added to the optimization.
- For minimizing delay a heuristic solving method is used that is not attractive because of its multiple steps and multiple computer programs, and its dependency on the choice of step size in the initial cycle time range.

The models discussed here could form a basis for a dynamic lane configuration model. Since for online use computation time is more critical, only the linear case with fixed approach and exit lanes seems feasible. But as computational power has increased over time, solving the optimization problems with a non-convex cost function may already be a possibility.

2-3 Literature on dynamic lane configurations

The papers discussed in the previous section provide a traffic engineer with a way to design an optimal intersection based on a certain demand dataset. A logical step is to apply the same techniques to an intersection where the assignment of lanes can change in real time: a dynamic lane configuration. This section will discuss papers that present research on this topic.

2-3-1 Hausknecht (2011)

Category	Feature
Multi-static	Compare two lane configurations, no optimization

In [12] lane reversal on an intersection is briefly discussed. For a single intersection as a test case the demand for a certain direction is increased significantly. Lane reversal is implemented by adding an extra lane to the direction with the increased demand, reducing the number of lanes available to the neighboring direction. The lane reversed situation is compared with the unchanged lane configuration by using microsimulation. The throughput was found to increase by 6 % when the lane configuration was changed.

Analysis

This small study provides no contribution to the modeling or control of a dynamic lane configuration, but gives an indication of how big gains can be when lanes are used in a dynamic way.

2-3-2 Zhang & Wu (2012)

Category	Feature	Optimization includes	Optimization criteria	Problem type	Solving method
Multi-static	Find lane configuration for multiple demand matrices	Lanes, signal timing	Flow ratio	MINLP	Exhaustive search

Based on the Wong & Wong model, which was presented in Section 2-2-2, Zhang & Wu demonstrated the possible benefits of a dynamic lane configuration system [13]. As optimization criterion they used the flow ratio. They minimized the largest flow ratio, as a result equalizing the flows on the lanes. The optimization is done for each leg of the intersection separately. The minimum is found by calculating the objective function for all feasible lane configurations.

The method was tested by doing a numerical analysis, looping over a range of fictional demand data. The delay was found to decrease with an increase in spatial variation of the demand. Compared to a fixed-time signal controller the average delay was reduced with 35 %.

Analysis

Their analysis is very limited and only functions as a numerical proof of concept. It is not based on simulation and only considers the static case. This method can be seen as redesigning the intersection and recalculating a fixed timing strategy for each demand variation. No algorithm for switching the lane configuration is presented. The implications of the system being dynamic are not researched.

In the paper some difficulties with online implementation are identified, such as safety, getting information to the road users, and the real-time estimation of traffic demand. Apart from that no discussion is provided.

The choice for using the flow ratio as optimization criterion is questionable. It results in the flows being equal on each lane, and as was shown in the examples in the introduction this does not always lead to an optimal configuration, as it could even increase delay.

2-3-3 Zhao, Ma, Zhang, & Yang (2013)

Category	Feature	Optimization includes	Optimization criteria	Problem type	Solving method
Dynamic	Define switching criteria	Lanes	Flow ratio	MINLP	Exhaustive search

A much better approach is presented in [14]. It also uses the Wong & Wong model that was discussed in Section 2-2-2.

First of all they conclude that while in the static case signal timing and lane configuration can be optimized in an integrated approach, this is not advisable for real-time dynamic control. They name computational complexity and differing horizons as reasons for this separation.

In the paper two steps are introduced. First an optimal lane configuration is found by integer non-linear programming. Second the decision whether to implement the found configuration is made by evaluating a binary-type threshold.

The authors state that the lane assignment scheme of an arm is related only to that of the opposite arm and not the entire intersection, and as such they make groups of arms. Then for each group the lane optimization algorithm is performed separately.

The optimization criterion used is the critical flow ratio, defined as the highest ratio in a signal phase of actual flow divided by saturation flow. The problem is solved by first finding all valid lane configurations possible. Then the objective function is minimized for each configuration and the one with the lowest result is selected.

The lane configuration is switched if three criteria have been met, which are a 10 % predicted drop in delay, a 15 min window between the last switch, and the last two optimization cycles resulting in the same solution.

The method was tested by microsimulation using PTV Vissim. Three different fictional demand profiles where used, assuming that the future demand values are known by the controller. A traffic-actuated signal controller was implemented. Applying the dynamic lane algorithm on their test case reduced the average delay by 14.7 %.

Analysis

In the introduction of the paper the authors explain why they did not combine lane assignment and signal timing. Then they use an exhaustive search solving approach that finds a solution within 1 s, so it seems computational complexity is not an issue. The argument that the horizons of signal and lane control should have a different scale is more convincing.

The arm grouping is an interesting approach for reducing the search space. This is valid if only lanes are considered and signals are not included. What the effect of this simplification is on the result still needs to be investigated. The assumption in this thesis is that signal control cannot be ignored when also applying lane control.

Using the flow ratio as optimization criterion was done because other criteria such as delay or cycle time would need signal timing values, which are not included. Again, optimizing the flow ratio does not necessarily lead to an optimal configuration in terms of delay. It is unclear why the optimization problem is non-linear, while in [4] the use of flow ratios resulted in a linear system.

The logic control rules that decide whether to implement a found configuration are a good approach for their test case with three demand profiles. If they also work well with a faster changing demand remains a question. Furthermore their approach is not predictive but reactive, which limits the optimality of the placement of the switching moments. Also, it is unclear how they changed the lane configurations mid-simulation in Vissim.

Their evaluation is very limited, they tested only one case with a fictional demand profile. With this they showed an extreme situation in which their method works. How good the results are with other, more realistic demand profiles remains untested.

2-3-4 Conclusion on dynamic lane configuration

Little literature on dynamic lane configuration is currently available. The most recent study is the only one that truly studies dynamic behavior. It combines a model-based lane configuration optimization with a rule-based controller. It finds an optimal lane configuration by minimizing the flow ratio and implements the configuration if certain conditions are met. Some critical notes on this study include their use of an exhaustive search solving method, their exclusion of signal timings, their use of the flow ratio as optimization criterion, and the lack of a test using real-world data.

To the best of the author's knowledge no other papers on the topic of dynamic lane configuration are available in literature.

2-4 Conclusions on the existing knowledge on lane configuration optimization

In this chapter existing knowledge on static and dynamic lane configurations was presented. In practice only a few instances of dynamic lane use at intersections exist. At these the function of only a single lane is switched and it is done at fixed times between the morning and evening rush hour.

One model can be found in literature for the optimal design of the lane configuration of intersections. Its most important feature is that it results in a Mixed Integer Linear Programming problem, with variables on an individual lane basis. The optimization criteria encountered in literature include flow ratio, capacity, cycle length, and delay.

Little research on the topic of dynamic lane control at intersections is available. The study that best investigated the dynamic behavior of a dynamic lane configuration was based on the model for optimal lane configuration design. Possible improvements on that study are: including prediction in the control scheme, including signal timing, and using delay as optimization criterion instead of the flow ratio.

Chapter 3

Switching lane controller design

In this chapter the two controllers for a dynamic lane configuration system that have been developed as part of this research will be presented. Section 3-1 will give some background on the origin of the two controllers. Then in Section 3-2 the first controller, called the delay-based method, will be presented. The second controller, which is called the queue length-based method, will be presented next in Section 3-3.

3-1 Development considerations

This study started out as an adaption of the static lane configuration optimization model by Wong & Wong which was presented in Section 2-2-2. Their model results in a Mixed-Integer Linear Programming (MILP) problem that finds the lane configuration, signal order, and signal timing for a static situation. The goal when starting this study was to extend this static description into a dynamic, online method. However, in the two developed controllers not much remains from the Wong & Wong model. In this section the considerations in the development will be discussed. This forms a link between the literature from the previous chapter and the controllers that will be presented in the next sections.

3-1-1 Optimization criterion

Criteria available in static model

In the static model by Wong & Wong two optimization criteria are available: cycle time and capacity. Both result in a linear problem.

Cycle time Minimizing the cycle time would result in a fast signal timing, i.e. the green times are just long enough to be able to process the demand. This results in a degree of saturation of 1 for the critical signal groups, which means the signal timings are not robust for non-uniform vehicle arrivals. This can be solved by setting a constraint on the degree of saturation. Then the question becomes how large this value should be. A rule of thumb is to set it to 0.9 [6, p. 155].

Capacity The capacity of an intersection can be maximized by applying a multiplier on the demand values. The larger the multiplier, the more traffic the intersection can handle. Maximizing this multiplier results in a robust configuration. The problem with this approach is that it will always result in the largest allowed cycle time, so when demands are low the delay will be very high.

First criterion for a dynamic method

An intersection design that will not change over time, i.e. a static configuration, should be robust for fluctuations and possible future increases in demand. That is why capacity maximization is a sensible choice when designing a fixed configuration. For a dynamic configuration robustness is less important, since the whole idea of having a changing configuration is to be able to deal with changes in demand. That is why for a dynamic system the more fundamental goal of an intersection can become more prominent: to process each arriving vehicle as fast as possible. This corresponds to the concept of delay minimization.

Cycle time minimization cannot be directly used to minimize delay, because as Webster [1] demonstrated a minimum cycle time does not result in the least delay. Instead the optimal cycle time, which incorporates the randomness of vehicle arrivals, must be found. In signal control delay models are used to calculate the optimal cycle time, these can also be used to calculate and minimize the delay. Therefore the first controller will use a direct formulation of delay as optimization criterion.

Second criterion for a dynamic method

Delay is not an ideal criterion since delay models tend to be fairly complex. They are non-linear functions, most of them non-convex. Only the most simple delay model is convex in two parameters [9]. Using a complicated delay model will have a large impact on computation time.

As an alternative to delay, queue length, or the amount of vehicles waiting to be serviced, is proposed as a second criterion. To keep things as elementary as possible a *vertical queue* is used, which amounts to a simple input-output model. As a result the second controller can have a linear cost function, which can be solved faster than the non-convex cost function of the first controller.

3-1-2 Extending the static model over time

The model by Wong & Wong is static, as it finds a configuration for a single period of time. The duration of this period is undefined and is actually determined by the time span of the selection of the demand. A very straightforward way to extend the model over time is to calculate a configuration for each step in a range of time steps, that way getting a configuration that changes over time. This was done in the research by Zhao et al. presented in Section 2-3-3. A downside of this approach is that the effect a control input has on the following time steps is not incorporated. This matters because there is a cost associated with the switching of lanes.

Costs of switching lanes

At the few switching lanes that exist in the Netherlands switching procedures are defined that make sure the transition of a switching lane from one movement to the other happens in a safe way. The basic idea is that the switching lane should be empty before vehicles from the new movement are allowed on the lane. In this study that behavior is modeled by demanding a 120 s period between uses by different movements, during which the switching lane is not available.

This switching period comes with a cost. The lane cannot be used by both movements during those 2 min, so to still be able to handle all traffic the green times of those movements must be prolonged, which means that vehicles of conflicting movements will have to wait longer.

So when comparing switching with not switching, first the delay is higher for 2 min, followed by a longer period of lower delay. This means that the control input must be based not only on the direct effects but also on the longer term effects. In short, prediction is needed.

MPC

Model Predictive Control (MPC) is a class of control strategies of which the main feature is that it finds an optimal control input based on predictions of future system states. The basic components of an MPC scheme are a prediction model to predict the system state, a cost function to determine the optimal control input, possibly constraints on the optimization vector, and a receding horizon setup.

The receding horizon principle is a way to advance over time. The optimal control input for each time step in the prediction horizon is found, but only the first entry is applied. Then the time is advanced with one step and the process is repeated.

One of the biggest advantages of the MPC methodology compared to other control techniques is that it is able to deal with constraints on the controller signals. Some of the disadvantages of MPC are its dependency on the quality of the prediction model and

its computational complexity, which means it is harder to set up and is less fast than regular feedback control techniques.

The interested reader can find more information on the MPC methodology in the papers referred to here. MPC was initiated in the late 1970's as described in this survey [15]. It has since matured into its "glorious present", a term coined in a more recent survey [16]. An overview of research on the stability of constrained MPC is provided in [17]. MPC can also be used with discrete variables, which then falls under the hybrid systems category. In [18] an important type of hybrid systems is presented, while also discussing an MPC approach for controlling this type of system.

MPC is used in this study because it provides a methodology to find switching moments based on prediction, which is assumed to lead to better results than reactively changing the lane configuration.

3-1-3 From lane-based to movement-based

The model by Wong & Wong is lane-based, which means that each lane has its own set of variables. The variables include the movements allowed on each lane, the signal orderings, the flow on each lane, the start of green and the duration of green for each lane and separately also for each movement. The movements allowed on each lane are defined by binary lane use variables, and there are many of them. Even if a lane will only be used for right-turning traffic, the variables for straight-ahead and left-turning traffic still need to be used.

When extending the model over multiple time steps the number of variables increase even further and efficiency becomes more important. To reduce complexity and speed up calculations the lane-based approach was abandoned and instead variables per movement were adopted. The Wong & Wong model was largely discarded, instead the lane configuration is modeled by a few binary variables and the saturation flows.

What does remain from the Wong & Wong model is an adaptation of the way it calculates the start and duration of green for each movement. This is described for the first controller in the part on the constraints in the lower-level task in Section 3-2-3. For the second controller it is included in the part on the signal timings in Section 3-3-7.

3-2 Delay-based method

In this section the first controller that was developed will be presented. It is called the delay-based method and it is an MPC approach. Using a non-convex delay model it finds the lane configuration and signal timing of an intersection that results in minimal delay for users.

3-2-1 General idea

The main idea of this method is to split the calculation of the signal timings and the determination of the lane configuration. The result is an optimization problem with two levels. The lower-level uses a delay model to find the signal timings that result in the lowest delay for each time step in the horizon. The upper-level uses this delay value to select the lane configuration with the lowest delay. This bi-level optimization is performed for each time step, resulting in an optimal configuration over time.

Why a bi-level approach?

There are ways to solve an optimization problem with a mixture of integer and continuous variables directly. So why not solve a single optimization problem that finds the lane configuration as well as the signal timing? There are three main reasons:

1. Since an initial queue delay is needed the cost function is not only non-linear but also non-convex. Solvers that can handle a mixed-integer problem with non-convex cost function were not readily available.

2. Splitting the problem means a solver specialized in the type and the size of the subproblem can be used.

3. With separate problems also different prediction horizon lengths can be used. This can work similar to how a control horizon works in more regular MPC descriptions. Computational complexity can be reduced and it furthermore makes sense because of the delay between the lane switch and its effects on the intersection.

Horizons

To further specify how the two different horizons work consider the example in Fig. 3-1. In this example at time step k a prediction is done for the next five time steps. This value corresponds with the lower-level prediction horizon N_{v}. The prediction contains differing signal timings for each time step. This is not the case for the lane configuration. The upper-level horizon N_{c}, from here on called the control horizon, dictates in this example that for the last two time steps the same lane configuration as in k+2 has to be used.

The effect of this control horizon is that the computation becomes faster, because the number of lane configuration combinations is reduced. Furthermore the effect of a lane switch takes time to become fully visible, and as such it makes sense to predict more time steps after the last lane configuration control input in the horizon.

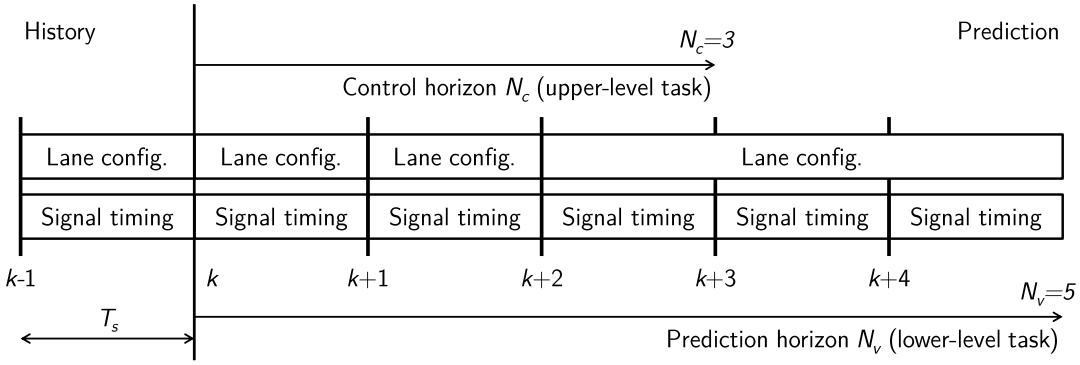


Figure 3-1: Visualization of both horizon values, with $N_\mathrm{c} = 3$ and $N_\mathrm{v} = 5$, where k is the current time and T_s is the time step duration

Assumptions

An important assumption is that the demand is known for future time steps. In a simulation setting this can be achieved by using a self-created or a historical data set. Another possibility is to predict the demand online. This can be done with for example a simple extrapolation or a time series model. Demand prediction is a whole field of study in itself and the interested reader is referred to a survey of the field in [19].

Another assumption, or restriction, is that the signal grouping and phase order is fixed. A signal structure must be determined externally so that it can be used as input for the method. Though inclusion of the optimization of the phase order is possible, it was left out in order to make the problem less complex, and as such be better able to see the workings and effects of the method. A short analysis of possible phase orderings for different lane configurations indicated that the orderings used in this study perform well for a multitude of lane configurations.

3-2-2 Upper-level optimization

The upper-level task finds the optimal lane configuration based on the delay values it gets from the lower-level task.

Variables and problem

The only variables used in this task are binary variables indicating how each of the lanes capable of switching is being used over time. Each switching lane has one variable for each time step in the horizon. Suppose there are four switching lanes and the control horizon is two steps, then this would result in eight binary variables.

The problem to be solved is finding which combination of binary variables results in the least delay. In order to do this first all possible combinations of variables are determined.

Since each variable has only two possible values the number of combinations is 2^{nN_c}, where n is here the number of switching lanes and N_c is the control horizon. This number can be reduced by removing infeasible combinations, such as when certain switching lanes are not allowed to be switched at the same time.

Solver

The resulting problem is a pure integer problem. There are three strategies for solving this kind of problem. The first is enumeration, which constitutes of trying each possible combination. The second is branch-and-bound, which is a more systematic way than enumeration, eliminating certain subsets of combinations during optimization. The third is using heuristics such as genetic algorithms or simulated annealing.

Enumeration is easy to implement and will result in finding the most optimal solution, but it is only a possibility when the number of variables is low, since the problem size grows exponentially with the number of variables. Because of these reasons enumeration will be used to solve the problem when the number of variables is sufficiently small.

The benefit of using heuristics is that even with a very large number of variables an acceptable result can still be obtained in reasonable time. It all depends on the settings; in general the longer the search, the more optimal the result will be. If the search is very short, the result will be similar to a random sample. For solving the upper-level optimization with larger number of variables the Genetic Algorithm provided in Matlab's Global Optimization toolbox will be used.

Branch-and-bound can greatly reduce the number of function calls compared to enumeration and it would therefore be a good choice to solve this problem. It can be tuned to be more or less aggressive in its search. It has not been applied on this problem for practical reasons. Matlab does not supply a good branch-and-bound solver and since the other two solving strategies were already functioning well no further effort was applied to implementing a solver of this type.

Genetic Algorithm

The `ga` function supplied in Matlab's Global Optimization toolbox is a method for solving non-linear derivative-free problems. It is well suited to deal with bit strings, where each variable is a binary value. The cost function can consist of basically any kind of function or algorithm. The function mimics biological evolution processes. It creates a population consisting of possible solutions, calculates the cost of each individual, than creates a new generation, and repeats the process until stop criteria have been met.

Creating the next population is done with two methods: crossover and mutation. Crossover is the combination of two individuals to form a new individual, where the parents are selected based on their cost. Mutation is a random change in an individual. Crossover results in the population stabilizing towards its most fit individual. Mutation makes

sure that more parts of the search space are explored. The emphasis in tuning genetic algorithms lies on the trade-off between crossover and mutation.

The initial population is constructed as follows: each possible lane configuration for the first time step is determined. For example for a system with four switching lanes this are $2^4 = 16$ combinations. Then each combination is repeated in the following time steps in the control horizon. With this initial population at least all possible switching actions in the first time step are considered.

More emphasis is added on mutation by reducing Matlab's default crossover fraction to 0.6. Furthermore the default mutation rate, which is the chance an entry of an individual selected for mutation will change, is increased to 15 %. With these two changes the solver does not converge to a, most likely, local minimum but keeps exploring the search space, until there has not been a change in the cost of the best individual for 40 generations.

A downside of using a genetic algorithm is that when the same individual appears in multiple generations, still with each appearance the cost function is called. This can be averted by using a history of cost values in a lookup table. Unfortunately when using a genetic algorithm combined with parallel computing this is not possible. For this study the parallel computing increased speed more than a lookup table, so the latter was not used.

Pseudo-code

As a summary of the workings of the upper-level task, consider the pseudo-code presented in Algorithm 1.

Algorithm 1 Upper-level optimization task

```
function UPPER_LEVEL(Q_b,0, Q)
    n_b ← nN_c
    if n_b ≤ 8 then
        C_b ← 2^n_b combinations
        for i ← 1 : n_b do
            LOWER_LEVEL(C_b(i))
    else
        C_b ← 2^n combinations
        GENETIC ALGORITHM(C_b)
```

where:

$$\begin{aligned} Q_{\mathrm{b},0} &= \text{queue remainder of previous time step} \\ Q &= \text{demand} \\ n &= \text{number of switching lanes} \\ N_{\mathrm{c}} &= \text{control horizon} \end{aligned}$$

3-2-3 Lower-level optimization

The lower-level task receives a sequence of lane configurations and calculates signal timings for each time step in the prediction horizon N_{v}. The optimization criterion that is used is delay.

From binary variables to a non-integer problem

The lower-level task translates the binary lane-use variables it receives from the upper-level to saturation flow values. The saturation flow of a movement in a time step is calculated by multiplying the lane capacity K with the number of lanes the movement has in that time step. This is shown in Eq. (3-1). The result is a 2d-matrix with saturation flows for each movement in each time step in the horizon. The lane capacity K is a parameter that can be tuned to fit the prediction model.

$$s_{i,k} = \lambda_{i,k} K \tag{3-1}$$

where:

$$\begin{aligned} s_{i,k} &= \text{saturation flow of movement } i \text{ at time step } k \ (\text{veh}\,\text{h}^{-1}) \\ \lambda_{i,k} &= \text{number of lanes of movement } i \text{ at time step } k \\ K &= \text{lane capacity } (\text{veh}\,\text{h}^{-1}) \end{aligned}$$

Modeling switching behavior

When a lane switches its function, it will be closed for $T_{\mathrm{sw}} = 2$ minutes before opening with its new direction. This behavior is modeled in the lower-level task.

In addition to the saturation flow matrix based on the binary lane-use variables, a second saturation flow matrix is made that represents the situation when movements have less lanes available due to intermediate lane closure. The time-weighted average of these two matrices is used further:

$$s = \frac{(T_{\mathrm{s}} - T_{\mathrm{sw}})\, s_{\mathrm{as}} + T_{\mathrm{sw}} s_{\mathrm{sw}}}{T_{\mathrm{s}}} \tag{3-2}$$

where:

$$\begin{aligned} T_{\mathrm{s}} &= \text{duration of a time step} \\ T_{\mathrm{sw}} &= \text{duration of the switching period} \\ s_{\mathrm{as}} &= \text{saturation flow after switching} \\ s_{\mathrm{sw}} &= \text{saturation flow during switching} \end{aligned}$$

The reason for this modeling choice is that this way the signal timing does not have to calculated twice if a switching action occurs. With approximately half of the problem

consisting of time steps in which switching occurs this decision reduces the problem size with 33 %.

The consequence of this modeling approach is that the time step duration must be at least as large as the duration of the switching period, so $T_s \geq T_{sw}$. If $T_s = T_{sw}$ this procedure results in $s = s_{sw}$. For longer time steps this procedure becomes less accurate because the average saturation flow is used during and after switching, which leads to green times that are too short during switching and too long after switching.

Variables

There are three types of variables for which the lower-level task must find a value. The first is the reciprocal of the cycle time $\zeta = {}^1/_{T_C}$, the second is the start of green θ, a fraction of the cycle time, and the third is the duration of green ϕ, also a fraction of the cycle time.

There are as many θ and ϕ variables as movements on the intersection, and there is a set of the three types of variables for each time step in the horizon. For an intersection with 12 movements and a prediction horizon of $N_v = 4$ this results in an optimization vector with a length of $4(12 + 12 + 1) = 100$.

All three types of variables are bounded. ζ is bounded by the inverse of the minimum and maximum cycle time and θ and ϕ are bounded by 0 and 1.

Cost function

The cost function calculates the intersection delay given the cycle time and start and duration of green in the optimization vector. External inputs to the cost function are the demand, the saturation flow matrix, and the initial queue following from the previous time step.

The delay model that is being used as cost function consists of two parts. First is the combined uniform and overflow queue delay model by Akçelik as described in [6, p. 355]. Second is the initial queue delay model as written in [20, p. 16-F]. These delay models result in an average delay per vehicle, expressed in seconds per vehicle.

Cost function — Akçelik

The delay model by Akçelik [21] consists of two parts. The first, D_1, represents the delay assuming uniform arrivals. It is an adaptation of the delay model by Webster [1]. Note that D_1 cannot be used without the second part, D_2, which is the delay due to the randomness of arrivals. It represents the delay experienced by vehicles that arrive during green but cannot be serviced and have to wait until the next green phase. This is called the overflow queue delay. Both D_1 and D_2 are defined as follows:

$$
\begin{aligned}
x &= \frac{Q}{su}, \qquad y = \frac{Q}{s}, \qquad z = x - 1 \\
x_0 &= 0.67 + \frac{su}{600\zeta} \\
D_1 &= \frac{0.5\,(1-u)^2}{\zeta\,(1-y)} \\
D_2^* &= 0.25 T_\mathrm{s} \left(z + \sqrt{z^2 + \frac{12\,(x - x_0)}{suT_\mathrm{s}}} \right) \\
D_2 &= \begin{cases} D_2^* & \text{when } D_2^* > 0 \\ 0 & \text{when } D_2^* \leq 0 \end{cases}
\end{aligned}
\tag{3-3}
$$

where:

u = green fraction, equal to ϕ

x = degree of saturation

x_0 = degree of saturation above which overflow queues start appearing

These equations can be solved for each movement and for each time step. Each value of D_1 and D_2 is not dependent on other time steps.

Cost function — initial queue delay

If a movement is over-saturated a queue will start building. In the next time step this queue needs to be dissolved first before the traffic of this next time step can be serviced. The vehicles in the initial queue as well as the new arrivals both experience more delay. An initial queue delay model is a way to describe this behavior. This is essential in a receding horizon setup, since without it queues due to over-saturation would just disappear between time steps.

Note that initial queues are different from overflow queues. The first are solely used to transfer delay from one time step to the next, the latter describe delay during a single time step. They both concern a different part of saturated behavior and so they complement each other.

The formula to calculate the initial queue delay D_3 is presented here:

$$
\begin{aligned}
D_3 &= \frac{0.5Q_{\mathrm{b}}\left(1+u_{\mathrm{b}}\right)t_{\mathrm{b}}}{usT_{\mathrm{s}}} \\
t_b &= \begin{cases} 0 & \text{when } Q_{\mathrm{b}} = 0 \\ \min\left(T_{\mathrm{s}}, \dfrac{Q_{\mathrm{b}}}{us\left(1-\min\left(1,x\right)\right)}\right) & \text{when } Q_{\mathrm{b}} > 0 \end{cases} \\
u_b &= \begin{cases} 0 & \text{when } t_{\mathrm{b}} < T_{\mathrm{s}} \\ 1 - \dfrac{usT_{\mathrm{s}}\left(1-\min\left(1,x\right)\right)}{Q_{\mathrm{b}}} & \text{when } t_{\mathrm{b}} \geq T_{\mathrm{s}} \end{cases}
\end{aligned}
\tag{3-4}
$$

where:

$$
\begin{aligned}
Q_{\mathrm{b}} &= \text{initial queue (veh)} \\
t_{\mathrm{b}} &= \text{time before initial queue is solved (s)} \\
u_{\mathrm{b}} &= \text{model parameter}
\end{aligned}
$$

It depends mainly on the initial queue, which is the number of vehicles left in queue at the end of the previous time step due to over-saturation. This value can be calculated with Algorithm 2. If a movement is over-saturated then $z > 0$ and the initial queue will grow during a time step. If not, then $z < 0$ and the initial queue will decrease. Indexing starts at 2, since the initial queue of the first time step in the horizon is a result from the previous time step.

Algorithm 2 Calculate initial queue

for $i \leftarrow 2 : (N_{\mathrm{v}} + 1)$ **do**
$Q_{\mathrm{b}}(i{+}1) = \max\left(0,\ Q_{\mathrm{b}}(i) + u(i)s(i)z(i)T_{\mathrm{s}}\right)$

Cost function — combined delay

The three delay values are combined and the weighted average over all movements and time steps in the horizon is taken. The weights that are used are the demand values. This combined delay, together with an extra term, can be calculated as follows:

$$
D = \frac{\sum\left(Q\left(D_1 + D_2 + D_3\right)\right)}{\sum\left(Q\right)} + 0.05Q_b(N_{\mathrm{v}}{+}1) \tag{3-5}
$$

Multiplying the demand and the average delay per vehicle for each movement and time step results in the average seconds of delay per hour experienced at each movement during each time step. The sum of these values is the total average seconds of delay per hour for the whole intersection and prediction horizon. When this is divided by the

sum of all demands the result is the average seconds of delay per vehicle for the whole intersection and prediction horizon.

The queue that is left at the end of the prediction horizon is added as an extra term, multiplied with a scaling parameter. This parameter must be small, but large enough to make sure that the last time step in the horizon cannot be over-saturated without consequence.

MPC implementation

To visualize how the switching behavior is modeled and how the delay is summed, consider the schematic representation in Fig. 3-2. From top to bottom the figure first shows how the delay is calculated, then how the initial queue is transfered to the next time step, then the signal timings of each time step, and finally how the lane configuration is switched.

At time k the middle lane is closed for a duration of T_{sw}. After that the lane is opened again. Both of these lane configurations happen in the same time step, with the same signal timing.

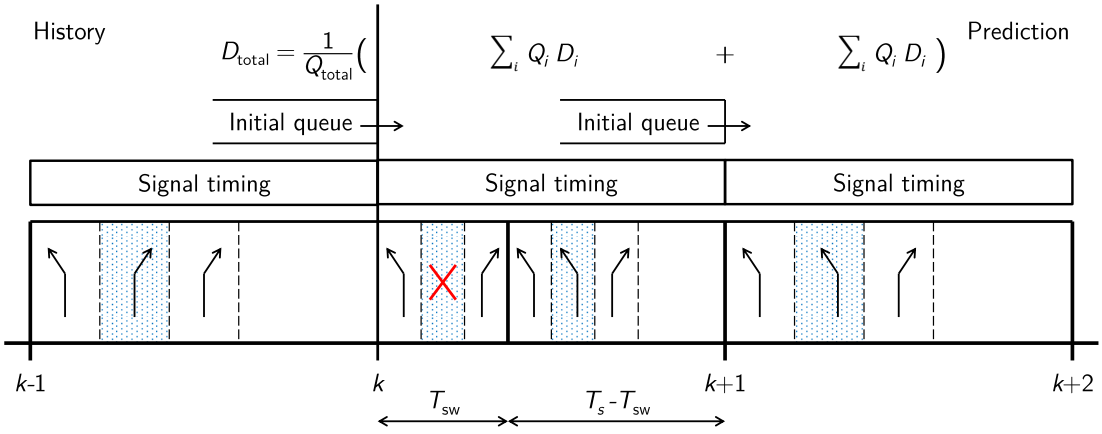


Figure 3-2: Schematic representation of the MPC implementation, with $N_{\text{c}} = 2$ and $T_{\text{s}} > T_{\text{sw}}$

Constraints

The cost function calculates delay, but it does not say what the restrictions on ζ, θ and ϕ are. This traffic signal behavior is modeled using two linear constraints, both originate from the Wong & Wong model presented in Section 2-2-2.

Constraint — duration of green

For safety reasons it is common to set a minimum on the green time. For this study a minimum green time of $g_{\min} = 5\,\mathrm{s}$ is used. This constraint must be satisfied for each ϕ:

$$-\phi + g_{\min}\zeta \leq 0 \tag{3-6}$$

Constraint — signal order and duration

The second constraint guarantees that the signals are in the right order, that conflicting movements do not get green at the same time, and that a clearance time between conflicting movements is obeyed.

The constraint is not repeated for each variable, such as the previous constraint, but for each pair of conflicting movements. That is why first a list of all conflicting pairs needs to be made. For each pair the clearance time has to be calculated, which can be done for example with a method described in [22].

The clearance time of each pair is increased with a constant g_e, which describes how much of the yellow time is used, minus the time loss at the start of green. For this study it is assumed 2 seconds of yellow time are used, and 1 second is lost at the start of green, resulting in $g_e = 1$. The combination of clearance time and g_e is stored in ω_{lm}, which is the minimum time between the end of green of movement l and the start of green of movement m.

Next also the order of each conflict pair has to be determined. In the Wong & Wong model the determination of this ordering was included in the optimization, here a fixed ordering is used to reduce the complexity. The ordering is captured in Ω_{lm}, which is 0 if movement l precedes movement m, and is 1 if the opposite is true.

The constraint is defined as follows:

$$\theta_l + \phi_l + \omega_{lm}\zeta - \theta_m \leq \Omega_{lm} \tag{3-7}$$

Suppose m follows l, so $\Omega_{lm} = 0$. Then the start of green θ_m must be larger than the start of green θ_l plus the duration of green ϕ_l plus the augmented clearance time ω_{lm} as a fraction of the cycle time.

Solver

The resulting problem has a non-convex cost function, continuous variables, and linear inequality constraints. The cost function is continuous but not differentiable in every point. D_2 for example is set to 0 when $D_2^* < 0$, creating a bend at $D_2 = 0$ where it cannot be differentiated. So strictly speaking the problem has a non-smooth cost function.

Matlab's `fmincon` function can solve many classes of constrained optimization problems. Though it is not meant for non-smooth functions, it is able to solve some cases. For this specific problem the Interior Point, Sequential Quadratic Programming (SQP), and Active-Set algorithms are candidates.

The three solver algorithms were evaluated in a short comparison of which the results are displayed in Table 3-1. For this test the signal timings for 12 different lane configurations with a prediction horizon of 4 steps had to be calculated. The SQP and Active-Set algorithms were able to solve all 12 problems, while the Interior-Point algorithm could not find a feasible point in 4 out of the 12 problems. Using SQP took the least amount of time and gave the best results.

In this test the starting point results in over-saturation and thus an initial queue building. It was observed that the SQP algorithm successfully traversed the bend in the initial queue delay formulation toward a solution without initial queue. This indicates that the algorithm is able to solve this non-smooth problem.

Table 3-1: Comparison of `fmincon` solver algorithms

Algorithm	Calculation time (s)	Success rate (%)	Avg. delay (s veh^{-1})	Lowest delay (s veh^{-1})
SQP	45.69	100	26.17	17.86
Active-Set	429.20	100	26.29	17.87
Interior-Point	55.43	67	28.10	20.14

3-2-4 Summary of delay-based method

In this section the first of the two controllers developed for this thesis was presented. It constitutes of a method for finding the lane configuration and signal timing, based on delay, and using a receding horizon.

The upper-level optimization task uses binary variables to describe the use of the switching lanes. It receives delay values from the lower-level task that it uses to evaluate the possible lane configurations. If there are 8 options or less each option is evaluated, if there are more an genetic algorithm is used.

The lower-level task finds the cycle time, start of green, and duration of green for each movement and each time step in the prediction horizon. It uses a delay model as cost function, consisting of uniform delay, overflow queue delay and initial queue delay. The non-convex problem with linear inequality constraints is solved by Matlab's `fmincon` solver using the SQP algorithm.

3-3 Queue length-based method

The second controller developed in this study, the queue length-based method, is an MPC approach for finding the lane configuration and signal timing of an intersection

that results in minimal queue lengths.

The main assumption is that if the queue lengths are minimized this will also result in a low delay. In this study the queue length is seen as the number of vehicles that have yet to be serviced, and if this number is kept as low as possible, then vehicles will have to wait as little as possible.

This approach results in a quadratic and thus convex model, in contrast with the non-convex model in the delay-based method, such that the optimization problems based on this model can be solved efficiently.

The method consists of several parts. First in Section 3-3-1 the general idea of the MPC approach will be explained. Then in Section 3-3-2 the prediction model will be described, followed by the cost function in Section 3-3-3 and the constraints in Section 3-3-4. Next in Section 3-3-6 the solving method is discussed. Section 3-3-7 concludes this section with the way the results from the MPC scheme are translated into signal timings.

3-3-1 MPC implementation

The general idea of the MPC implementation is to predict the queue length L over N_{v} steps, based on the queue length at time k and the control inputs for each step in the horizon. A history of the lane configurations is kept such that the controller knows if a lanes has been closed long enough so that it may be opened again.

In Fig. 3-3 the MPC approach is drawn schematically. It shows how at time step k the queue length is predicted for three time steps ahead, based on three signal timings and three lane configurations. Furthermore because of the choice of the time step duration the middle lane must be closed for two time steps before it may be opened again. Lastly notice how the time steps in the prediction have a fixed duration of T_{s}, but that the optimization is called after the previously found cycle time T_{C} has passed.

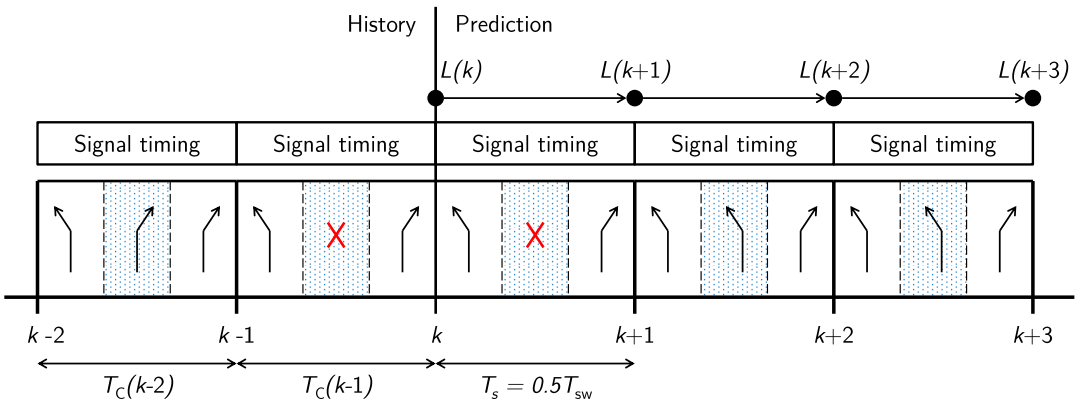


Figure 3-3: Schematic of the MPC implementation, with $N_{\mathrm{v}} = 3$ and $T_{\mathrm{s}} = 0.5T_{\mathrm{sw}}$

3-3-2 Prediction model

The basis for the prediction model is a very simple input-output formula:

$$L(k+1) = L(k) + T_s Q(k) - T_s s u(k) \tag{3-8}$$

where:

$L(k)$ = queue length (veh) at time step k
T_s = time step duration (s)
$Q(k)$ = demand (veh/s) in time step k
s = saturation flow (veh/s)
$u(k)$ = green ratio (green time / cycle time) in time step k

In words, the queue length in each time step is equal to the queue length in the previous time step, plus the amount of vehicles arriving at the intersection in that time step, minus the number of vehicles that passed the green light in that time step.

This type of model corresponds to the store-and-forward approaches, a class of models for traffic control of which a general description can be found in [23]. Note that this model uses a vertical queue, since queue dynamics are not incorporated, only the input-output behavior.

The novel idea here is to add a saturation flow and green ratio term for each switching lane and use binary variables to switch the green ratio on or off. For example for a movement with two regular lanes and one switching lane the queue length can be described as follows, where the subscript r indicates a regular lane and s indicates a switching lane:

$$L(k+1) = L(k) + T_s Q(k) - 2T_s s u_r(k) - T_s s u_s(k) \tag{3-9}$$

Now this description is transformed into a formulation such that all N_i movements are included:

$$x(k+1) = Ax(k) + B_1 w(k) + B_2 v(k) \tag{3-10}$$

where:
x = system state, contains queue length and history of N_h sets of binary variables $\bar{b}$
w = collection of known signals, which are the demand values
v = control input: green ratios for both regular and switching lanes, and binary lane-use variables

The signals x, w and v are defined as follows:

$$x(k) = \begin{bmatrix} L_1(k) \\ \vdots \\ L_{N_\mathrm{i}}(k) \\ \bar{b}(k-N_\mathrm{h}) \\ \vdots \\ \bar{b}(k-1) \end{bmatrix} \qquad w(k) = \begin{bmatrix} Q_1(k) \\ \vdots \\ Q_{N_\mathrm{i}}(k) \end{bmatrix} \qquad v(k) = \begin{bmatrix} u_{\mathrm{r},1}(k) \\ \vdots \\ u_{\mathrm{r},N_\mathrm{i}}(k) \\ u_{\mathrm{s},1}(k) \\ \vdots \\ u_{\mathrm{s},N_\mathrm{i}}(k) \\ b_1(k) \\ \vdots \\ b_{N_\mathrm{i}}(k) \end{bmatrix} \tag{3-11}$$

The system matrices are constructed as follows:

$$\begin{aligned} A &= \begin{bmatrix} I_{N_\mathrm{i}} & 0 & 0 \\ 0 & 0 & I_{N_\mathrm{i}(N_\mathrm{h}-1)} \\ 0 & 0 & 0 \end{bmatrix} \quad B_1 = T_\mathrm{s} I_{N_\mathrm{i}} \\ B_2 &= -T_\mathrm{s} \begin{bmatrix} s_{\mathrm{r},1} & & & s_{\mathrm{s},1} & & 0 & \\ & \ddots & & & \ddots & & \ddots \\ & & s_{\mathrm{r},N_\mathrm{i}} & & s_{\mathrm{s},N_\mathrm{i}} & & 0 \end{bmatrix} \end{aligned} \tag{3-12}$$

3-3-3 Cost function and prediction

Now the cost function J can be defined. It is based on the predicted states that are made using the prediction model:

$$J(k) = \sum_{j=1}^{N_\mathrm{v}} \hat{x}^T(k+j|k) W \hat{x}(k+j|k) \tag{3-13}$$

where:

N_v = prediction horizon length
$\hat{x}(k+j|k)$ = prediction of state x for time step $k+j$, evaluated at time step k
W = weighting matrix

The predictions of future states are made as is illustrated below:

$$\begin{aligned} \hat{x}(k+1|k) &= Ax(k) + B_1 w(k) + B_2 v(k) \\ \hat{x}(k+2|k) &= A^2 x(k) + AB_1 w(k) + AB_2 v(k) + B_1 w(k+1) + B_2 v(k+2) \\ &\vdots \end{aligned} \tag{3-14}$$

Now the signals of each time step are stacked in vectors. These variables, that contain information for the whole prediction horizon, are denoted with a tilde, as shown here:

$$\tilde{x}(k) = \begin{bmatrix} \hat{x}(k+1|k) \\ \hat{x}(k+2|k) \\ \vdots \\ \hat{x}(k+N_\mathrm{v}|k) \end{bmatrix} \quad \tilde{w}(k) = \begin{bmatrix} w(k) \\ w(k+1) \\ \vdots \\ w(k+N_\mathrm{v}-1) \end{bmatrix} \quad \tilde{v}(k) = \begin{bmatrix} v(k|k) \\ v(k+1|k) \\ \vdots \\ v(k+N_\mathrm{v}-1|k) \end{bmatrix} \tag{3-15}$$

The state predictions $\tilde{x}(k)$ can be expressed in terms of $x(k)$, $\tilde{w}(k)$ and $\tilde{v}(k)$:

$$\begin{aligned} \tilde{x}(k) &= \tilde{A}x(k) + \tilde{B}_1\tilde{w}(k) + \tilde{B}_2\tilde{v}(k) \\ &= \begin{bmatrix} \tilde{A} & \tilde{B}_1 & \tilde{B}_2 \end{bmatrix} \tilde{z}(k) \end{aligned} \tag{3-16}$$

The construction of the matrices $\tilde{A}$, $\tilde{B}_1$ and $\tilde{B}_2$ and the definition of the optimization vector $\tilde{z}$ are shown below:

$$\tilde{A} = \begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^{N_\mathrm{v}} \end{bmatrix} \quad \tilde{B}_1 = \begin{bmatrix} B_1 & & & & \\ AB_1 & B_1 & & & \\ A^2B_1 & AB_1 & B_1 & & \\ \vdots & \vdots & \vdots & \ddots & \\ A^{N_\mathrm{v}-1}B_1 & A^{N_\mathrm{v}-2}B_1 & A^{N_\mathrm{v}-3}B_1 & \dots & B_1 \end{bmatrix}$$

$$\tilde{B}_2 = \begin{bmatrix} B_2 & & & & \\ AB_2 & B_2 & & & \\ A^2B_2 & AB_2 & B_2 & & \\ \vdots & \vdots & \vdots & \ddots & \\ A^{N_\mathrm{v}-1}B_2 & A^{N_\mathrm{v}-2}B_2 & A^{N_\mathrm{v}-3}B_2 & \dots & B_2 \end{bmatrix} \quad \tilde{z} = \begin{bmatrix} x(k) \\ \tilde{w}(k) \\ \tilde{v}(k) \end{bmatrix} \tag{3-17}$$

Now the cost function can be written as a quadratic function. Note that this could also be a linear function, which is computationally less complex than a quadratic one. Still, in this study a quadratic formulation was used. More on this is included in the discussion in Section 8-2.

$$\begin{aligned} J(k) &= \tilde{z}^T \begin{bmatrix} \tilde{A} & \tilde{B}_1 & \tilde{B}_2 \end{bmatrix}^T \tilde{W} \begin{bmatrix} \tilde{A} & \tilde{B}_1 & \tilde{B}_2 \end{bmatrix} \tilde{z} \\ &= \tilde{z}^T H \tilde{z} \end{aligned} \tag{3-18}$$

The weight matrix is chosen such that only the queue lengths are penalized:

$$\tilde{W} = \begin{bmatrix} W & & \\ & \ddots & \\ & & W \end{bmatrix} \qquad W = \begin{bmatrix} I_{N_\mathrm{i}} \\ 0 \end{bmatrix} \tag{3-19}$$

3-3-4 Constraints

There are six constraints that have to be satisfied. Five of them are linear inequality constraints and one of them is a linear equality constraint.

Constraint 1: positive queue length

The first constraint prevents the queue lengths from becoming negative. Since the queue length will be minimized it would become minus infinite without this constraint. It has to be satisfied for each movement and in each time step in the prediction horizon:

$$-L_i(k+j) \leq 0, \qquad \text{for } i = 1, \ldots, N_\mathrm{i};\ j = 1, \ldots, N_\mathrm{v} \tag{3-20}$$

Constraint 2: switching lane off

The second constraint makes sure that when the binary variable of a movement is 0 its corresponding switching green ratio is also 0. So if $b = 0$ then $u_\mathrm{s} = 0$. Its implementation is shown below:

$$u_{\mathrm{s},i}(k+j) - b_i(k+j) \leq 0, \qquad \text{for } i = 1, \ldots, N_\mathrm{i};\ j = 0, \ldots, N_\mathrm{v}-1 \tag{3-21}$$

Constraint 3: switching lane on

When the binary variable of a movement is 1 its corresponding switching green ratio should be equal to the regular green ratio. Both green ratios belong to the same movement and thus get the same green time. So if $b = 1$ then $u_\mathrm{s} = u_\mathrm{r}$. This constraint is constructed as two inequalities:

$$\begin{aligned} u_{\mathrm{r},i}(k+j) - u_{\mathrm{s},i}(k+j) + b_i(k+j) &\leq 1 \\ -u_{\mathrm{r},i}(k+j) + u_{\mathrm{s},i}(k+j) + b_i(k+j) &\leq 1 \end{aligned} \qquad \text{for } i = 1, \ldots, N_\mathrm{i};\ j = 0, \ldots, N_\mathrm{v}-1 \tag{3-22}$$

Constraint 4: sum of green ratios

This constraint is concerned with the conflicts between movements. The sum of all green ratios in a *conflict group* cannot exceed the maximum green ratio for that group, in order to reserve parts of the cycle for clearance times and add robustness by reducing the degree of saturation. This maximum green ratio is calculated using a formulation for the practical minimum cycle time:

$$T_{\mathrm{C}} = \frac{T_{\mathrm{v},\psi}}{1 - \dfrac{Y_\psi}{\rho}} \tag{3-23}$$

where:

$$\begin{aligned} T_{\mathrm{v},\psi} &= \text{lost time of conflict group } \psi \\ Y_\psi &= \text{sum of flow ratios in the conflict group } \psi \\ \rho &= \text{maximum degree of saturation} \end{aligned}$$

The sum of flow ratios can be described as follows:

$$Y_\psi = \sum_{i \in \psi} (u_i x_i) \tag{3-24}$$

where:

$$x_i = \text{degree of saturation of movement } i$$

In the prediction model it is assumed that during u_i the full saturation flow can flow. As a consequence $x_i = 1$ and $Y = \sum u_i$ in this description. Now the lost time can be transformed into the constraint definition:

$$\sum_{i \in \psi} u_i(k+j) \leq \rho \left(1 - \frac{T_{\mathrm{v}}}{T_{\mathrm{C,max}}}\right) \qquad \text{for } \psi \in \Psi;\ j = 0, \ldots, N_{\mathrm{v}}-1 \tag{3-25}$$

As value for T_{C} the maximum allowable cycle time is used. A commonly used value is $T_{\mathrm{C,max}} = 120\,\mathrm{s}$. This constraint has to be satisfied in each time step for all conflict groups in Ψ.

Constraint 5: switching time

The next constraint deals with the time a switching lane is closed between functions. This constraint works over multiple time steps.

First consider the following example. Suppose movement 01 and movement 02 share a switching lane, the switching time is $T_{\text{sw}} = 120\,\text{s}$, and the time step duration is $T_{\text{s}} = 60\,\text{s}$. The constraint can then be implemented using the binary variables, as shown here:

$$\begin{aligned} 3b_1(k) + b_2(k) + b_2(k{+}1) + b_2(k{+}2) &\leq 3 \\ 3b_2(k) + b_1(k) + b_1(k{+}1) + b_1(k{+}2) &\leq 3 \end{aligned} \tag{3-26}$$

If $b_1(k) = 1$ then b_2 can become 1 at time step $k + 3$ the earliest. These equations are then repeated for each switching lane and for each time step in the horizon.

In addition to applying these equations to the prediction horizon, the system should also have a memory in order to know whether a switching lane has been closed sufficiently long such that it can be opened during the current time step. This memory should store the binary variables up to the switching time. As shown in Eq. (3-11) the state x does not only contain the queue lengths L, but also the past binary variables. The number of historic steps is $N_{\text{h}} = T_{\text{sw}}/T_{\text{s}}$. Note that the consequence of this modeling choice is that this fraction must produce an integer result and that the time step duration T_{s} should be smaller than T_{sw}. This can be written as $nT_{\text{s}} = T_{\text{sw}}$ with $n \in \mathbb{Z}_{>0}$.

The constraint can be defined starting from the oldest binary variables in the history up to the last time step in the prediction horizon. The constraint is defined as follows:

$$Mb_l(k{+}j) + \sum_{\epsilon=0}^{\min(N_{\text{h}},N_{\text{v}}-1-j)} b_m(k{+}j{+}\epsilon) \leq M \qquad \begin{array}{c} \text{for } j = -N_{\text{h}}, \ldots, N_{\text{v}}{-}1; \\ l, m \in \pi;\ \pi \in \Pi \end{array} \tag{3-27}$$

where:

$M = N_{\text{h}} + 1$ (or another sufficiently large number)

$\Pi =$ set containing all switching lanes

$\pi =$ index of Π, contains all movements dependent on switching lane π

l and $m =$ certain combination of two movements dependent on switching lane π

It has to be satisfied for each switching lane. For example if Π contains 4 switching lanes, each switching lane π depends on 2 movements, and a history of $N_{\text{h}} = 2$ and a prediction horizon of $N_{\text{v}} = 4$ is used this results in 48 constraints.

Note that a switching lane can depend on more than two movements. For example with a switching lane dependent on three movements the constraint has to be satisfied for each possible combination of two of the three movements, resulting in 6 constraints per time step.

Constraint 6: control horizon

The last constraint fixes the control input for the time steps beyond the control horizon. This reduces the computational load and can in certain cases make the control signal calmer. It is a linear equality constraint:

$$-v(k+N_{\mathrm{c}}-1)+v(k+j)=0 \qquad \text{for } j=N_{\mathrm{c}},\ldots,N_{\mathrm{v}}-1 \tag{3-28}$$

Another option is to not constrain the whole control input vector v but only the binary variables b. This would still lessen computational complexity, and compared to constraining the whole control input it would prevent infeasible results due to insufficient control input freedom.

3-3-5 Constraints in horizon

Constraint 1 and constraint 5 depend on values contained in the system state x. That means they cannot use the variables in the optimization vector $\tilde{z}$ directly but need the predictions of x.

Consider the following formulation, where C_1, C_2 and C_3 define the constraints for a single time step:

$$\gamma(k)=C_1x(k)+C_2w(k)+C_3v(k)\leq\Gamma(k) \tag{3-29}$$

Similar as in Section 3-3-3 this expression can be stacked for each time step in the prediction horizon to form the collection $\tilde{\gamma}$:

$$\tilde{\gamma}(k)=\tilde{C}_1x(k)+\tilde{C}_2\tilde{w}(k)+\tilde{C}_3\tilde{v}(k)\leq\tilde{\Gamma}(k) \tag{3-30}$$

The vector $\tilde{\gamma}$ is formed with the predictions of γ, while $\tilde{\Gamma}$ is known on forehand:

$$\tilde{\gamma}(k)=\begin{bmatrix}\hat{\gamma}(k|k)\\ \hat{\gamma}(k+1|k)\\ \vdots\\ \hat{\gamma}(k+N_{\mathrm{v}}-1|k)\end{bmatrix} \qquad \tilde{\Gamma}(k)=\begin{bmatrix}\Gamma(k)\\ \Gamma(k+1)\\ \vdots\\ \Gamma(k+N_{\mathrm{v}}-1)\end{bmatrix} \tag{3-31}$$

The $\tilde{C}$-matrices are constructed as follows:

$$\tilde{C}_1 = \begin{bmatrix} C_1 \\ C_1A \\ C_1A^2 \\ \vdots \\ C_1A^{N_\mathrm{v}-1} \end{bmatrix} \quad \tilde{C}_2 = \begin{bmatrix} C_2 & & & \\ C_1B_1 & C_2 & & \\ C_1B_1^2 & C_1B_1 & C_2 & \\ \vdots & \vdots & \vdots & \ddots \\ C_1B_1^{N_\mathrm{v}-1} & C_1B_1^{N_\mathrm{v}-2} & C_1B_1^{N_\mathrm{v}-3} & \dots \quad C_2 \end{bmatrix} \quad \tilde{C}_3 = \begin{bmatrix} C_3 & & & \\ C_1B_2 & C_3 & & \\ C_1B_2^2 & C_1B_2 & C_3 & \\ \vdots & \vdots & \vdots & \ddots \\ C_1B_2^{N_\mathrm{v}-1} & C_1B_2^{N_\mathrm{v}-2} & C_1B_2^{N_\mathrm{v}-3} & \dots \quad C_3 \end{bmatrix} \tag{3-32}$$

Now these linear equality constraints can be evaluated using the optimization vector $\tilde{z}$:

$$\begin{bmatrix} \tilde{C}_1 & \tilde{C}_2 & \tilde{C}_3 \end{bmatrix} \tilde{z} \leq \tilde{\Gamma} \tag{3-33}$$

3-3-6 Solver

The problem is a Mixed-Integer Quadratic Programming (MIQP) problem with linear constraints. It can be solved with an open-source solver such as SCIP [24], or commercial solvers such as Gurobi or CPLEX. For larger problems, such as intersections with four legs, SCIP proved insufficient as it was often unable to find a feasible solution. For the simulations in this study the TOMLAB implementation of CPLEX was used, which performed satisfactory in all cases.

3-3-7 Secondary optimization problem: signal timings

The green ratios need to be translated into signal timings, i.e. a cycle time and start and durations of green. This is done by a secondary optimization task.

The green ratios that follow from the primary optimization already have a maximum degree of saturation taken into account. Therefore they can be used directly as durations of green. They only need to be altered if they result in a green time that is smaller than the minimum green time. If the resulting durations of green are placed in the predetermined phase order and the right clearance times are added in between, the cycle time can be determined by taking the duration of the critical conflict group.

This process can be done using a similar optimization problem as in the lower-level task in the delay-based method, in Section 3-2-3. Here the optimization vector also consists of the start of green θ, the duration of green ϕ, and the reciprocal of the cycle time $\zeta = 1/T_\mathrm{C}$.

The variables θ and ϕ are again bounded by 0 and 1. Furthermore the two constraints presented in Section 3-2-3 are used here as well. The first is a constraint on the minimum green time (Eq. (3-6)) and the second implements the clearance time and signal order (Eq. (3-7)). A new addition is a constraint that demands that the duration of green is as large as the green ratio from the primary optimization result:

$$-\phi_i \leq -u_i \qquad \text{for } i = 1, \ldots, N_\text{i} \tag{3-34}$$

The cost function of this secondary optimization problem minimizes the cycle time:

$$f = \begin{bmatrix} 0 & \ldots & 0 & -1 \end{bmatrix} \tag{3-35}$$

This secondary optimization problem is a Linear Programming (LP) problem with linear inequality constraints and can be efficiently solved with a solver such as Matlab's `linprog` function.

3-3-8 Summary of the queue length-based method

In this section the second method that was developed has been presented. It is an MPC approach for finding the lane configuration and signal timing that will minimize queue lengths.

The main optimization problem consists of a prediction model to predict the queue length development, a cost function that minimizes this queue length, and linear constraints that model, among other things, the lane switching behavior. The resulting problem can be solved with an MIQP solver. The output contains optimal lane configurations and green ratios for the prediction horizon, of which the first time step gets implemented. In a separate LP optimization problem the green ratios get translated into a cycle time and green times.

3-4 Summary of switching lane controller design

In this chapter the two controllers for online dynamic lane configuration optimization that have been developed as part of this study have been presented. Both are Model Predictive Control approaches that find the lane configuration and signal timing.

The first is called the delay-based method and finds the lane configuration and signal timing that minimizes delay. The delay model includes terms for uniform delay, overflow queue delay, and initial queue delay. The method uses a bi-level optimization problem, of which the upper-level is an integer problem and the lower-level is a non-linear problem with linear constraints. The upper-level task is solved by enumeration for small problems

and a genetic algorithm for bigger problems. The lower-level task is solved with an SQP solver.

The second method is called the queue length-based method and finds the lane configuration and signal timings that minimize the number of vehicles waiting at the intersection. It uses a simple input-output model to predict future states. The problem is written as a Mixed-Integer Quadratic Programming (MIQP) problem with linear constraints. After solving this problem the green fractions get translated into a cycle time and green times by a separate Linear Programming (LP) problem.

The biggest differences between the two methods are shown in Table 3-2.

Table 3-2: Differences between the two methods

	Delay-based	**Queue length-based**
Optimization criterion	Delay	Queue length
Input information	Demand	Demand and queue length
Time step duration	$T_s \geq T_{sw}$	$nT_s = T_{sw}$ with $n \in \mathbb{Z}_{>0}$
Optimization problem	Bi-level: integer, and non-convex with linear constraints	MIQP, followed by LP

Chapter 4

Evaluation method

The tests that will be used to evaluate the developed controllers will be presented in this chapter.

First in Section 4-1 the two real intersections that were used will be discussed. Next in Section 4-2 the tests, in particular the demand data that was used as input, will be presented. Following is Section 4-3 on PTV Vissim, the software that was used to perform simulations. The section details on how it was used and how the switching of lanes was implemented. In Section 4-4 the data visualization method that was developed for this thesis will be presented. The chapter will be concluded with Section 4-5, which presents the method used to analyze statistical significance.

4-1 Intersection selection

In this section the two intersections that were selected for this study will be presented. For both intersections the demand profile, their configuration and parameters will be discussed.

4-1-1 Selection method

The goal of the selection is to find two intersections, of which one is simple and the second is complex. The first was mainly used during development of the controllers, while the second became more prominent during the evaluations.

Two factors are important when selecting a test case. First is that the intersection should be large enough, it should have enough approach and exit lanes to be able to place a switching lane. Second is that the demand profile should be varying considerably, since

the more demand changes over time and between movements, the more useful a dynamic lane configuration can be.

Demand data and maps of intersections located in the municipality of The Hague were available for this research. Their layout and demand profiles have been inspected and out of the candidates two intersections were selected as test case. These two were chosen mainly because their demand profile changed significantly during the day and between movements.

4-1-2 K359

Intersection K359 is located at the A12 entry/exit number 5 between Nootdorp and Leidschenveen. It has three legs with each three approach lanes and two exit lanes, and so each movement can have either one or two approach lanes. Thus the middle lane of each leg can function as a switching lane. A schematic representation of the intersection with the switching lanes and the possible movements is shown in Fig. 4-1. The actual layout of the intersection is included in Appendix A-1.

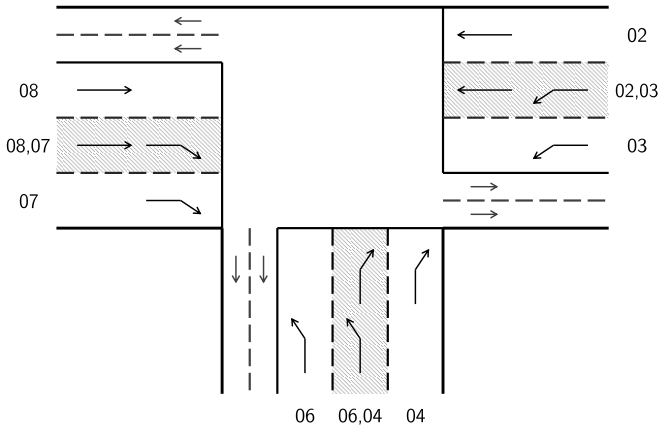


Figure 4-1: Schematic representation of K359 with switching lanes

The default, static layout of the intersection is presented in Table 4-1.

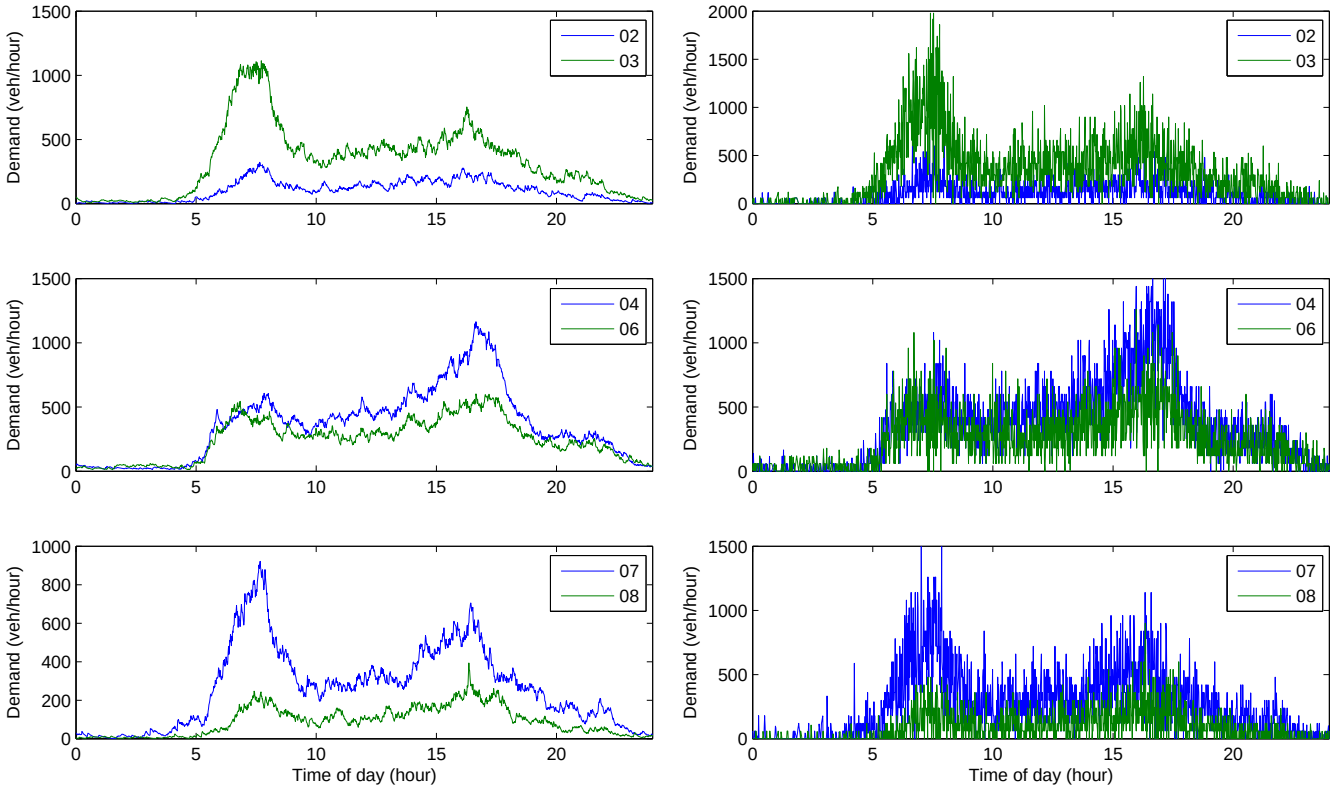
Table 4-1: Default lane configuration of K359

Movement	02	03	04	06	07	08
No. of lanes	1	2	1	2	2	1

Demand data For K359 demand data from 1 March up to 4 June 2014 was available. By visual inspection it was determined that most workdays have a similar demand profile. Tuesday 8 April 2014 was selected as test case because it clearly contains the exemplary

demand profile and because its mean demand is above average, which makes for a more interesting case than a day with low demand.

The demand data of 8 April 2014 is plotted in Fig. 4-2. The graph on the right uses the original data, the graph on the left is made with data after it was smoothened with a low-pass filter, of which the details will be discussed later. On each leg there is a moment in time, for example the morning rush hour, where a big difference in the demands for the two movements on the leg occurs. To illustrate this consider the peak in the demand for movement 07 in the morning. Or for example movements 04 and 06, that have the same demand during the morning but a very different demand in the evening. Such differences on multiple legs could warrant the use of multiple switching lanes.



(a) With low-pass filter **(b)** Without filtering

Figure 4-2: Demand data of K359 on 8 April 2014

Clearance times The clearance time of each conflict on K359 has been determined using the conflict zone approach method from [22], which is also included in [6, p. 57]. The publication also contains standard values for the parameters, which have been used in this study, as shown in Table 4-2. The clearance times are included in Table 4-3.

Table 4-2: Parameters for determining clearance time

Discharge speed straight-ahead	$12\,\mathrm{m\,s^{-1}}$
Discharge speed turning traffic	$8\,\mathrm{m\,s^{-1}}$
Vehicle length	12 m
Reaction time	0 s
Sum of accelerations	$3\,\mathrm{m\,s^{-2}}$
Entry speed	$14\,\mathrm{m\,s^{-1}}$
Critical entry distance	33 m

Table 4-3: Clearance times (s) for K359

		Stopping movement					
		02	03	04	06	07	08
Starting mov.	02	-		3.7			
	03		-		2.1	1.0	0.4
	04			-			1.8
	06	0.0	2.1		-		0.5
	07		3.5			-	
	08		1.3	0.7	0.8		-

Phase order The signal structure that is being used at the real-world intersection was not available for this study, instead the phase order for K359 has been determined using VRIGen, a tool developed at the TU Delft [25]. This tool provides an automated way to perform the steps described in [6]. Given the intersection layout and demand it lists the phase orderings and orders them based on their *lost time* and potential flexibility.

The optimal phase ordering was determined for several lane configurations and using the demand data of the morning and of the afternoon of 8 April 2014. For all of these tests one phase ordering had the lowest lost time, it is shown in Table 4-4. This phase order has been used further in this study.

Table 4-4: Phase order of K359

Phase	Movement					
a	02	03	04			
b	02				07	08
c			04	06	07	

4-1-3 K302

Intersection K302 is located in the city center of The Hague, close to the Central Station. It is a large intersection, it has four legs and each leg has between three and six approach lanes.

Each leg has two exit lanes, which means that each movement can have a maximum of two approach lanes. Most movements on K302 already have that number of approach lanes, and so to be able to implement switching lanes the size of the intersection is reduced in this study. Still this intersection was selected, because its demand profile is varying considerably. A schematic representation of the layout that will be used in this study is shown in Fig. 4-3. The actual layout of the intersection is included in Appendix A-1.

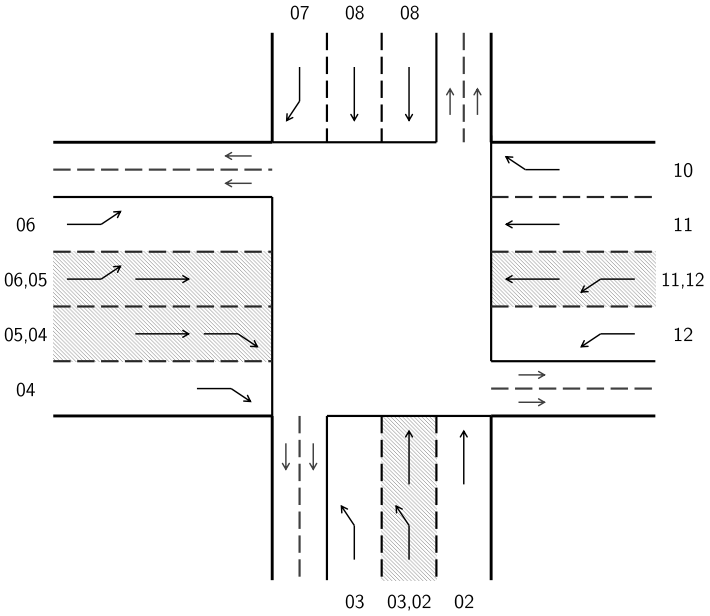


Figure 4-3: Schematic representation of K302 with switching lanes

The movements 01 and 02 are combined, they share an approach lane and therefore their signal settings are also equal. The same goes for movements 08 and 09. Since the settings of these combined movements should always be the same, the number of variables in the models can be reduced without consequence by fusing them. As such movements 01 and 09 will be removed, and their demand will be added to movements 02 and 08, respectively.

The default, static layout of the intersection is presented in Table 4-5.

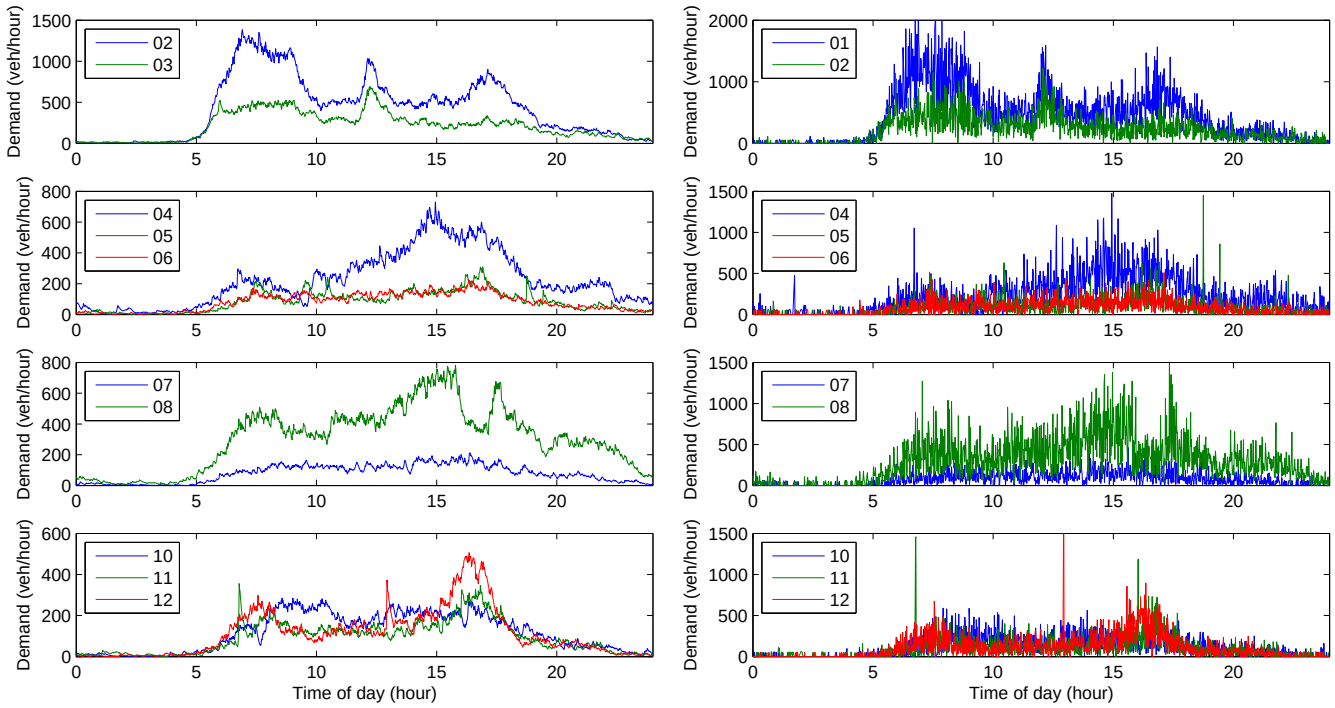
Table 4-5: Default lane configuration of K302

Movement	02	03	04	05	06	07	08	10	11	12
No. of lanes	2	1	1	2	1	1	2	1	1	2

Demand data Demand data from 6 March up to 4 June 2014 was available for this intersection. To select a day for further study two criteria have been used. The first looks

at the differences in the difference between left-turning and straight-ahead movements for each day. A big value could indicate that demand for both movements changes during the day, making it interesting to be able to switch lanes. The second criterion is the overall sum of the demand for each day, since a day with little traffic is not an interesting case. Based on this method the day that was selected for further study is Tuesday 27 May 2014. It should be noted that this is an average day and as such many other days would also have been suitable.

The demand data of this day is plotted in Fig. 4-4, again after and before filtering. Interesting aspects of this demand profile are the difference between movements 02 and 03 during and after the morning rush hour, the increase for movement 04 in the afternoon, and the demand of movements 10, 11, and 12 being almost equal throughout the day.



(a) With filtering **(b)** Without filtering

Figure 4-4: Demand data of K302 on 27 May 2014

Clearance times The clearance times of the conflicts on K302 were calculated using the same method and parameters as for K359. The result is shown in Table 4-6. Since movement 09 was assimilated by 08, the conflicts and clearance times of 08 also include those of 09. Note that the intermediate traffic signals of the original intersection were not modeled, which leads to high clearance times.

Table 4-6: Clearance times (s) for K302

Starting movement \ Stopping movement	02	03	04	05	06	07	08	10	11	12
02	-			4.1	3.8		5.1	0.0	0.0	0.0
03		-		3.3	3.1	0.0	4.6		0.0	0.5
04			-				2.7			5.4
05	0.0	0.0		-			1.6			3.7
06	0.1	2.7			-		0.9	0.0	0.0	
07		9.1				-			4.0	
08	0.0	1.0	0.0	0.0	0.6		-		3.0	1.7
10	3.2				6.4			-		
11	2.7	4.6			5.0	0.0	0.0		-	
12	1.5	2.7	0.0	0.0			1.1			-

Phase order Again the actual signal structure being used at the real-world intersection was not available, instead the phase order of K302 was found using VRIGen. Five different settings were tested using the demand from 27 May 2014. The tests included the morning rush hour demand with the regular lane configuration, the demand at 15:00 with the regular lane configuration and with a lane switch between movement 04 and 05, and the demand at 16:00 with the regular lane configuration and with a lane switch between movement 07 and 08. For all these tests one phase order was found to be optimal, it is presented in Table 4-7.

Table 4-7: Phase order of K302

Phase	Movement									
a								10	11	12
b						07	08	10		
c			04	05	06	07				
d	02	03	04							

4-2 Experiments

In this section the different cases that are used to evaluate the controllers will be presented. Two types can be distinguished, first are fictional demand cases and second are real-world demand cases.

4-2-1 Fictional demand data

The fictional demand cases will use simple signals that can be simulated in a short time, in order to be able to tune the controllers and find out whether they work as intended.

Two types of test signals will be used: a step input and a sinusoid input. For these tests only the switching of the lane associated with the movements receiving a demand increase will be considered.

Signal construction The signals are constructed as follows. Each movement will have a continuous base demand, which is a value in vehicles per hour multiplied with the number of lanes assigned to that movement. The demand of a single movement will increase for a certain period of time and with a certain amplitude. This increase can be in the form of a step or a sine. An example of both the step and sinusoid signal for one movement are shown in Fig. 4-5.

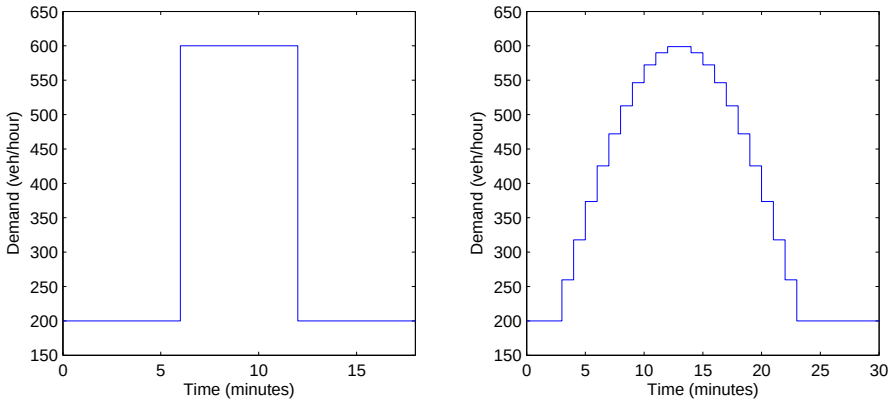


Figure 4-5: Example of step and sinusoid demand increase

K359 For K359 the evaluations with fictional demand data will concern the allocation of the switching lane between 07 and 08. Since the intersection is symmetric the other switching lanes would have most likely produced similar results. Two different base demand values will be used: 400 veh/lane/h and 450 veh/lane/h. The first results in an under-saturated situation, the second results in a saturated situation. The demand increase will be three times the base value. Furthermore the demands of movements 07 and 08 will be taken equal, instead of a per lane value. The result is that at the peak the demand of movement 07 is three times higher than the demand of movement 08. This difference corresponds with the differences between the movements on each leg seen in the demand data, which are around three to four times the lower value.

K302 For K302 the switching lane of movements 02 and 03 was selected for the evaluation with fictional data, since in the real-world demand profile the demand difference between these movements varies over time. The movement that will receive an increase in demand is 03. The two values used as base demand will be 200 veh/lane/h and 250 veh/lane/h, which is less then for K359 because this intersection is bigger and has

more phases. Again these values result in an under-saturated and over-saturated situation, respectively. The amplitude of the demand increase will be three times the base value, which is the same as for K359, but since the base demand is multiplied with the initial number of lanes of each movement, at the peak the demand of movement 02 will effectively be 1.5 times higher than the demand of movement 03. The reason for this is that movement 03 has more conflicts than movement 02, and so a small difference already warrants a lane switch.

Time step duration For the delay-based method the shortest time step possible is $T_s = 120\,s$, which will be used for these experiments, since a high resolution will most likely produce the best results for these short experiments. The queue length-based method can have lower time step durations, but they should not be too low because then the number of steps in the prediction horizon needs to become too large. For these experiments $T_s = 60\,s$ is used, a value that, combined with a prediction horizon of $N_v = 8$ results in a prediction horizon timespan that is just large enough to contain the entire demand increase.

4-2-2 Real-world demand data

Using real-world demand data gives the opportunity to simulate how the controllers would work in a situation that resembles reality. The same demand data that was presented in Section 4-1 will be used for evaluation. For K359 this is 8 April 2014 and for K302 this is 27 May 2014. The time interval that will be used is from 05:00 to 23:00. Before and after these times there is almost no traffic.

Filtered and unfiltered data As discussed earlier in reality some form of demand prediction is needed. These demand forecasts are used by the controllers as input for their predictions. These forecasts will not be perfect and will most likely not have a very high level of detail. To simulate this behavior the controllers were fed with a filtered version of the demand data, the same as has been shown in Fig. 4-2a and Fig. 4-4a. At the same time the vehicles were placed in the simulation using the original unfiltered data, thus adding noise to the simulation. A low-pass filter was used that can be described by the following discrete transfer function:

$$\frac{0.1}{1 - 0.9z^{-1}} \tag{4-1}$$

Time step duration For the delay-based method three time step durations T_s will be evaluated: 2, 5 and 15 min. With a prediction horizon of 4 steps these values result in prediction horizons of 8, 20 and 60 min. The queue length-based method for the real-world demand evaluations again uses a time step duration of $T_s = 60\,s$. A higher

value of 120 s, which is also the maximum, would also be a possibility, but 60 s better corresponds with the cycle times that commonly occur. A lower value is not advisable, since then the prediction horizon time span would become too short.

4-3 Experiment setup in PTV Vissim

To test the performance of the different methods the microscopic traffic simulation software PTV Vissim is used.

4-3-1 About PTV Vissim

Vissim is a microsimulation program owned by the PTV Group. Microsimulation means that each vehicle placed in the simulation is treated as an agent and has its own way of driving. In each time step all the vehicles are moved through the network according to the models describing their behavior. The accuracy of the simulation depends largely on these models. The heart of Vissim is the car-following-model developed by Wiedemann in 1974. This model, extended with other models and stochastic distributions together try to replicate individual driving behavior. An excellent overview of the research behind Vissim can be found in [26, Ch. 2].

For this study the standard urban motorized traffic parameter set was used. Only the standard car vehicle type was used. No custom calibration was performed because that would have taken up a too large part of the scope of this study.

Vissim has a feature that makes it possible to control the software from external sources through the COM port. This feature is very extensive, many model and simulation parameters can be accessed and also changed. For this study Matlab was used to access Vissim through COM.

4-3-2 Lane switching in Vissim

Vissim has no preprogrammed way to implement switching lanes at intersections and there is no information on possible work-arounds available. Therefore part of this study was to test and evaluate ways to implement this kind of infrastructure in Vissim. For a more in-depth overview of the methods that were tried the reader is referred to Appendix A-2.

The combination of Vehicle Input and Routes was used to build a switching lane mechanism. The rate with which a Vehicle Input places vehicles in the simulation can be changed through COM during simulation. Also the way a Routing Decision point distributes the vehicles over the destinations can be changed through COM during simulation. The consequence of this decision is that each lane must be modeled by a link.

An example of the way a switching lane is implemented in Vissim is shown in Fig. 4-6. It features an approach on which two movements, green and blue, share three lanes. The middle lane is a switching lane, which is modeled by two links stacked on top of each other. Routing decisions placed on the links with the vehicle inputs determine with a simple 1 or 0 if vehicles are allowed on the switching lane or not.

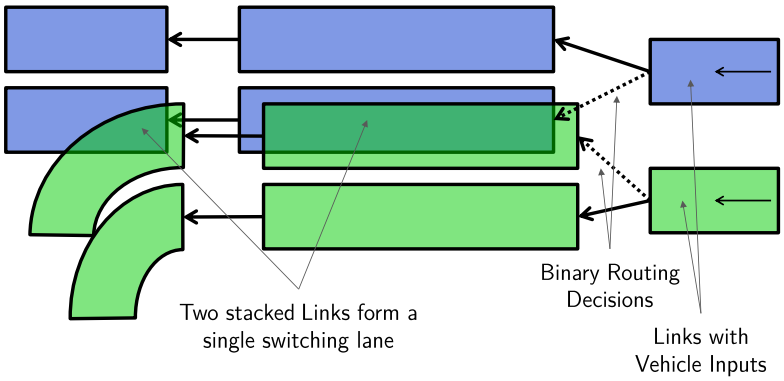


Figure 4-6: Example of the modeling in Vissim of an approach with three lanes of which one can switch

For an overview of the implementation of the intersections K359 and K302 in Vissim the reader is referred to Appendix A-2.

4-3-3 Simulation measurements

Vissim has many ways to evaluate the performance of the model. Two of those options will be used to analyze the outcome of the simulations.

Measurement sampling time The sample time that will be used in the measurements is 60 s. This is a reasonable value since the typical cycle time lies between 40 s to 120 s. If needed the measurements can be resampled afterwards to longer sampling times. A shorter sampling time would add an unnecessary level of detail, since the dynamics during a cycle are not assessed in this study.

Performance measures The primary performance measure in this study is the total delay, which is defined as the average delay per vehicle in a time interval multiplied with the number of vehicles that contributed to that average delay. Delay is one of the most common performance measures in traffic signal analysis and is equivalent with travel time and total time spent, two other commonly used terms.

Another common performance criterion in traffic signal studies is the number of stops. It will therefore be used as a secondary performance criterion when comparing the two controllers.

4-4 Result visualization

Since there is no precedent on how to present data on dynamic lane configurations, a custom way of displaying this kind of data was developed for this thesis. The goal is to show how the lane configuration changes over time, combined with showing a metric or some other signal, such as delay. The reason they should be combined is that in this way the possible influence of the changing lane configuration on the signal can be identified.

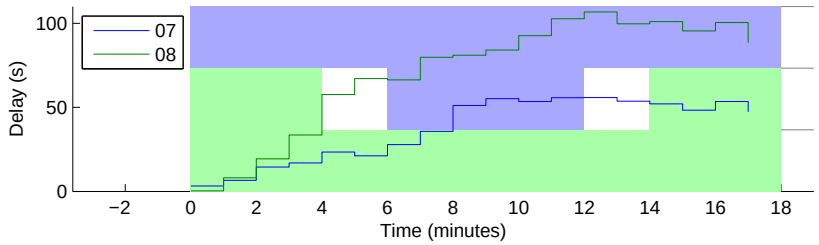


Figure 4-7: Example of how a performance metric is presented in combination with the lane configuration

A simple example of a graph showing the lane configuration and delay for a single leg of an intersection is shown in Fig. 4-7. The delay of movement 07 over time is presented by the blue line, the delay of movement 08 by the green.

The leg consists of three approach lanes and as such the graph is divided in three horizontal bars. The horizontal borders of these bars are indicated by the gray lines at the far right of the axis. The color that the bar has at each moment in time shows for what direction the lane is used. White means the lane is closed for both movements.

It is important to notice that the colored bands are always from top to bottom in order of the movements. Furthermore note that there is no correlation between the height of the lanes and the vertical axis, the height of the lanes has no meaning for the metric.

4-5 Statistical analysis method

When comparing the results of Vissim simulations a statistical test must be used to determine the certainty of the differences between the results.

In this study each experiment in Vissim is simulated ten times, each time with a different random seed. When Vissim results are presented in the next chapters, they will always be the mean of these ten results.

When comparing two experiments, the difference between the two means will be shown. The certainty of this difference will be determined using Student's t-test, a statistical hypothesis test most commonly taught in textbooks for this type of analyses [27, ch. 27-28]. Since for each experiment the same set of random seeds and the same simulation

model are used, the paired version of the test is used. Furthermore since there are no assumptions on one mean being larger than the other the test is two-tailed.

The t-test delivers a p-value, which is a measure for the probability that the difference between the two means is based on chance. In this study a difference will be called *significant* if the p-value is smaller than 5 %, a value which is often encountered in literature [27, ch. 26].

4-6 Summary of the test setup

In this chapter the methods with which the two controllers will be evaluated were presented.

Two intersections will be used, K359, which has three legs, and K302, which has four legs. The number and the location of the switching lanes on each intersection was presented. For both intersections clearance times, turning radii, and a phase order were determined. A single day with an interesting demand profile was selected for each intersection.

Two types of test signals were constructed. In both each movement has a base demand, and the demand of a single movement is increased for a portion of the experiment time. The first type features a step, a sudden increase and decrease in the demand. The second type has a gradual increase and decrease in the form of a sinusoid.

PTV Vissim, a microsimulation software package, will be used to simulate the intersections. The mechanics of the switching lanes was implemented by using Vehicle Inputs and Routing Decisions, essentially placing the control over the lanes in the hands of the user through the COM interface and Matlab. In Vissim the delay and the number of stops will be measured, sampling at each 60 s.

A way of displaying the lane configuration in graphs was explained. Colored horizontal bars, displayed behind the regular graph, indicate how each lane is used over time.

To test the statistical significance of the difference between two mean values resulting from the Vissim simulation, the Student's two-tailed paired t-test will be used. With a p-value of less than 5 % a difference is considered to be significant.

Chapter 5

Evaluation with fictional data

In this chapter the two controllers that were developed will be evaluated using fictional data. The goal of this chapter is to assess how the methods work, which will be done by varying the controller parameters and comparing the results. Most importantly, by comparing results created with a static and with a dynamic lane configuration the situations in which switching lanes are beneficial can be identified.

The four fictional demand cases that were presented in Section 4-2 will be used. In Section 5-1 the first controller, the delay-based method, will be evaluated for intersections K359 and K302. In Section 5-2 the second controller, the queue length-based method, will be evaluated using only intersection K302. Since the results of K359 and K302 are quite similar for the following experiments K359 was no longer used.

5-1 Delay-based method

The evaluation of the delay-based method with fictional data consists of four parts.

First in Section 5-1-1 the prediction model of the delay-based method will be calibrated by choosing a value for the lane capacity. Apart from calibration this parameter can also be used to increase the robustness of the controller. Therefore the value that results in the best controller performance in terms of delay will be selected.

Secondly in Section 5-1-2 the influence of switching will be evaluated by comparing the results of the static case with a case in which switching took place. Afterwards in Section 5-1-3 a single experiment will be used to provide a more in-depth look at the difference between static and switching.

Lastly in Section 5-1-4 the influence of the control horizon will be investigated. The goal is to find out if the control horizon can be reduced, and thus the computation time lowered, without diminishing the performance.

For all experiments a time step duration of $T_s = 120\,s$, which is the smallest possible value, and a prediction horizon of $N_v = 4$ was used. This combination results in a horizon of 8 min, which is enough to contain all or a large part of the demand increase.

5-1-1 Calibrating the prediction model

The goal of the calibration is to select a lane capacity value for which the controller has the best results, i.e. the lowest delay. This is not necessarily the actual simulation lane capacity. By selecting a lower value the green times will be higher than needed and thus robustness is increased.

The controller was evaluated for a range of prediction model lane capacity values: 1500, 1600, 1700, 1800, 1900, 2000 and 2100 veh h^{-1}. In these experiments the default, static lane configuration was used. The results are shown in Table 5-1, where a dark gray cell indicates the lowest value of a row and lighter gray cells indicate which values in a row did not differ significantly from the lowest value in the same row.

Table 5-1: Total delay (1×10^4 s) for different prediction model lane capacities

Inter-section	Signal type	Base demand	Lane capacity (veh/h) 1500	1600	1700	1800	1900	2000	2100
K359	step	400	1.182	1.162	1.146	1.151	1.126	1.182	1.079
		450	1.535	1.493	1.440	1.469	1.402	1.455	1.504
	sine	400	1.974	1.968	1.986	1.945	1.974	1.912	1.902
		450	2.472	2.489	2.514	2.506	2.452	2.522	2.636
K302	step	200	2.127	1.945	1.804	1.731	1.760	1.693	1.728
		250	3.397	3.339	3.480	2.911	3.012	2.929	2.929
	sine	200	3.597	3.111	3.063	3.053	2.984	2.988	2.993
		250	5.908	5.542	5.173	5.032	4.935	5.445	5.004

In 4 of the 8 experiments the lane capacity resulting in the least delay was $K = 1900$. In two cases the best result was achieved with $K = 2100$. Given the values that did not differ significantly, which can be seen as a confidence interval around the lowest value in each row, the best lane capacity value could range from $K = 1800$ to $K = 2100$. The actual lane capacity in Vissim was measured to be approximately 2080 veh h^{-1}. Given these results a prediction model lane capacity of $K = 2000$ was selected for further use, since it lies in the range of possible values and is lower than the actual Vissim value, thus improving robustness.

5-1-2 Influence of switching

The effects of lane switching were investigated by comparing the static case with a situation in which switching took place. A full control horizon of $N_c = 4$ was used.

The results are shown in Table 5-2. The last column in this table contains the p-values in percentages, where a value above 5 % indicates the difference is not statistically significant.

Table 5-2: Comparing static and switching

Inter-section	Signal type	Base demand	Total delay (1×10^4 s) Static	Switching	Diff. (%)	p (%)
K359	step	400	1.182	1.093	-7.59	0
		450	1.455	1.388	-4.62	10
	sine	400	1.926	1.795	-6.79	0
		450	2.508	2.300	-8.31	1
K302	step	200	1.693	1.622	-4.16	1
		250	2.929	2.511	-14.28	0
	sine	200	2.988	2.770	-7.30	0
		250	5.445	4.335	-20.39	0

In 7 out of the 8 cases switching caused a significant delay reduction. In the remaining case the result was inconclusive. The delay reduction was higher for the experiments with a high base demand.

These results can be interpreted as a clear indication switching is beneficial. Furthermore with higher demand the benefits of switching increases.

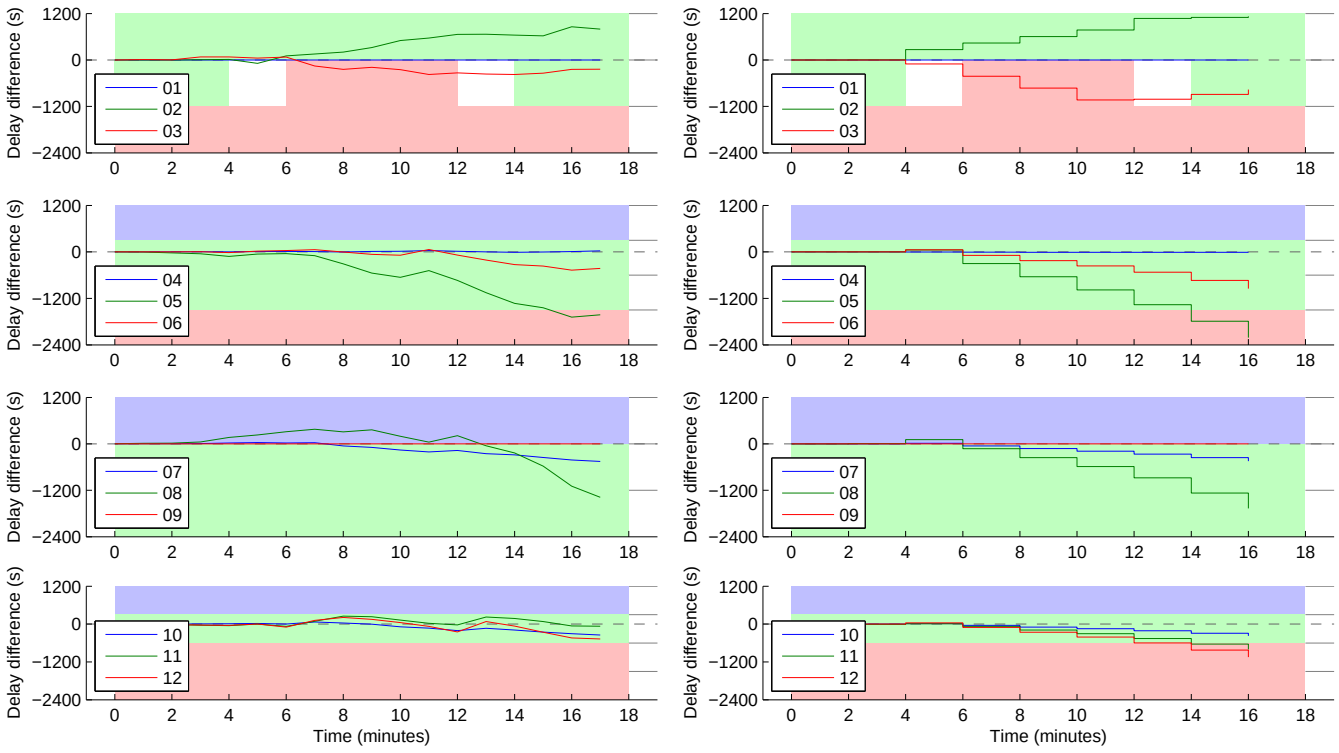
5-1-3 In-depth look at switching

Next a more in-depth analysis based on a single experiment is presented. The results of the step increase with a base demand of $d = 250$ were used. Two aspects are compared at the same time: the difference between static and switching and the difference between simulation results and prediction model.

Consider the graphs in Fig. 5-1, which both display a cumulative delay difference. The total cumulative delay of the static case was subtracted from the total cumulative delay of the switching case. This for example means that a negative difference indicates switching resulted in less delay than the static case. Furthermore in Fig. 5-1a the delay results from the simulation were used while in Fig. 5-1b the delay as found by the prediction model was used.

The shape of the graphs of the simulation and model correspond with each other. The biggest discrepancy is the predicted delay decrease for movement 03 that was not as large in the simulation, which is an indication of a model-simulation mismatch.

When looking only at the results of movements 02 and 03 it could be concluded the switching was not beneficial, since it resulted in a net delay increase for those two movements. However, switching resulted in an overall decrease in delay when the whole



(a) Simulation **(b)** Prediction model

Figure 5-1: Difference in cumulative delay between static and switching; for step with $d = 250$

intersection is considered. This shows that the delay-based method takes the whole intersection into account.

The lane configuration of the switching case is displayed in the figure. What can be clearly seen is that the controller switches the lane prior to the demand increase at 6 min. This indicates the MPC approach works as intended, using prediction to decide to close the switching lane before the demand increases.

5-1-4 Influence of control horizon

For the previous results the control horizon was as large as the prediction horizon. A lower control horizon reduces the computation time, but may lead to worse results. In this part the effect of reducing the control horizon is investigated by comparing two control horizon values: $N_c = 2$ and $N_c = 4$.

Table 5-3: Comparing control horizon values

Inter-section	Signal type	Base demand	Total delay (1×10^4 s) $N_c = 2$	$N_c = 4$	Diff. (%)	p (%)
K359	step	400	1.153	1.093	-5.20	1
		450	1.434	1.388	-3.20	22
	sine	400	1.790	1.795	0.29	59
		450	2.286	2.300	0.62	63
K302	step	200	1.599	1.622	1.49	24
		250	2.441	2.511	2.84	11
	sine	200	2.899	2.770	-4.48	9
		250	4.390	4.335	-1.25	1

In Table 5-3 the results for the two control horizon values are shown. In 6 out of the 8 experiments the results are inconclusive. In two cases a shorter control horizon resulted in more delay. These results suggest that it is better to use a full control horizon.

5-1-5 Conclusions on evaluating the delay-based method with fictional data

In 7 out of 8 cases switching decreased the total delay compared to a static configuration. In the other case the result was inconclusive.

The in-depth analysis shows the MPC approach works as intended, closing the switching lane before the demand increases. Furthermore the controller could quit correctly predict the delays, except for the movement receiving the extra lane.

This controller takes the whole intersection into account and as such switching does not necessarily benefit only the movements on its own leg, but instead can give the other movements more room in the cycle, reducing overall delay.

5-2 Queue length-based method

The evaluation of the queue length-based method with fictional data consists of four parts.

First in Section 5-2-1 two parameters in the prediction model will be calibrated to improve performance. Three values for the lane capacity (1800, 1850 and 1900 veh h^{-1}) and four values for the maximum degree of saturation (0.5, 0.6, 0.7 and 0.8) will be evaluated.

Secondly in Section 5-2-2 the influence of the prediction horizon will be investigated by comparing the results of four different prediction horizon values: 1, 4, 8 and 12.

The working of the controller will be analyzed by studying two graphs. In Section 5-2-3 the queue length of a sine signal experiment will be used to show how the controller reacts when it switches a lane. In Section 5-2-4 the static and the switching cases of a step experiment will be compared to investigate the influence of switching on the results.

In all experiments in this section a time step duration of $T_s = 60$ s was used.

5-2-1 Calibrating the prediction model

Table 5-4: Total delay (1×10^4 s) for different prediction model parameters

Signal type	Base demand	Lane capacity	Max. degree of saturation 0.5	0.6	0.7	0.8
step	200	1800	1.883	1.906	1.953	2.188
		1850	1.883	1.829	1.894	2.075
		1900	1.896	1.898	1.844	2.083
step	250	1800	3.112	2.881	3.044	3.435
		1850	3.016	2.860	2.895	3.234
		1900	3.015	2.937	2.989	3.249
sine	200	1800	3.364	3.250	3.228	3.587
		1850	3.355	3.242	3.326	3.513
		1900	3.365	3.233	3.212	3.478
sine	250	1800	5.979	5.158	5.005	5.413
		1850	5.878	5.123	5.015	5.343
		1900	5.626	5.156	5.002	5.418

The queue length-based method was evaluated for a range of prediction model lane capacities and maximum degrees of saturation. In these experiments a fixed prediction horizon of 8 steps was used. The results are shown in Table 5-4. A dark gray cell indicates the lowest value of a block and lighter gray cells indicate which values in a block did not differ significantly from the lowest value in the same block.

In the step experiments $K = 1850$ and $\rho = 0.6$ resulted in the lowest delay. In the sine experiments $K = 1900$ and $\rho = 0.7$ resulted in the lowest delay. Given the confidence interval on these minima the optimal parameter values could lie between 1800 and 1900 and between 0.6 and 0.7.

5-2-2 Influence of prediction horizon

Next the effect of altering the prediction horizon is investigated. Four different values have been tested: 1, 4, 8 and 12. In the experiments the prediction model parameters resulting in the lowest delay for each signal type where used. The results are shown in Table 5-5.

Table 5-5: Total delay (1×10^4 s) for different prediction horizons

Signal type	Base demand	Prediction horizon 1	4	8	12
step	200	1.982	1.880	1.894	1.893
	250	2.907	2.942	2.895	2.984
sine	200	3.697	3.276	3.250	3.382
	250	5.491	5.056	5.158	5.978

The lowest delay was achieved in half of the cases with a prediction horizon of $N_v = 4$, in the other half with $N_v = 8$. In all cases the difference between $N_v = 4$ and $N_v = 8$ was not significant. Both $N_v = 1$ and $N_v = 12$ had a significantly higher delay than the lowest value in 3 out of 4 cases.

Counter-intuitively the highest prediction horizon does not result in the lowest delay. This is most likely due to a mismatch between prediction model and simulation. With a longer horizon the controller will base its decisions on increasingly worse predictions.

For each result of $N_v = 1$ no switching occurred, which is as expected because the controller cannot switch lanes in one time step. Given this information it could be concluded that for most of these experiments switching resulted in less delay than the static case.

5-2-3 Analysis of state evolution

The system state, which is the queue length, of a single experiment will be studied more in-depth to see how the controller behaves when it applies a lane switch. The single best result of the sine experiments with a high base demand and a prediction horizon of $N_v = 8$ is used. The graph is shown in Fig. 5-2.

During the first switch the queue length of movement 03, which is receiving the switching lane, goes up. It is as if the controller, anticipating the extra lane, does not grant movement 03 extra time during switching, leading to queue length increases. Checking

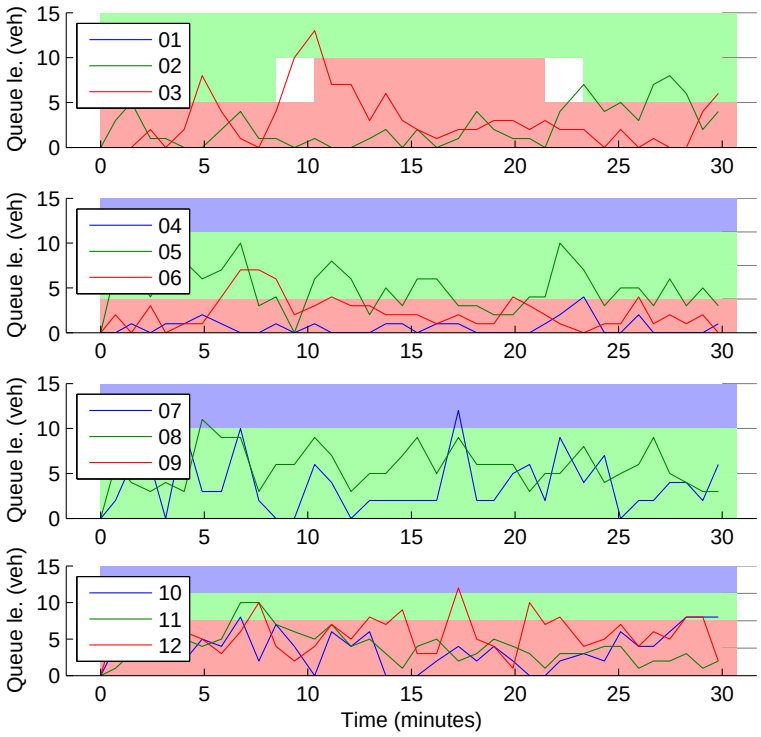


Figure 5-2: Queue length, single simulation with sine signal, $d = 250$, $N_{\mathrm{v}} = 8$, $K = 1900$, $\rho = 0.7$

the green fractions of movement 03 confirms this suspicion. During the switching period the green time drops from 9 s in the first time step of switching to 3.6 s in the second, while the green times of other movements and the cycle time stay the same.

The same thing seems to happen to movement 02 when the lane switches again at $t = 22$ min. During switching the green time of 02 drops from 6.3 s in the first step to 0.3 s in the second. Meanwhile the green times of all the other movements doubles or even triples and the cycle time is increased from 43 s to 69 s.

This example shows how the controller is not able to handle all dynamics on the intersection well. A possible explanation is that the sudden change by the binary variable acts as a step input, and the effect seen in the green fractions is overshoot.

5-2-4 Comparing static and switching

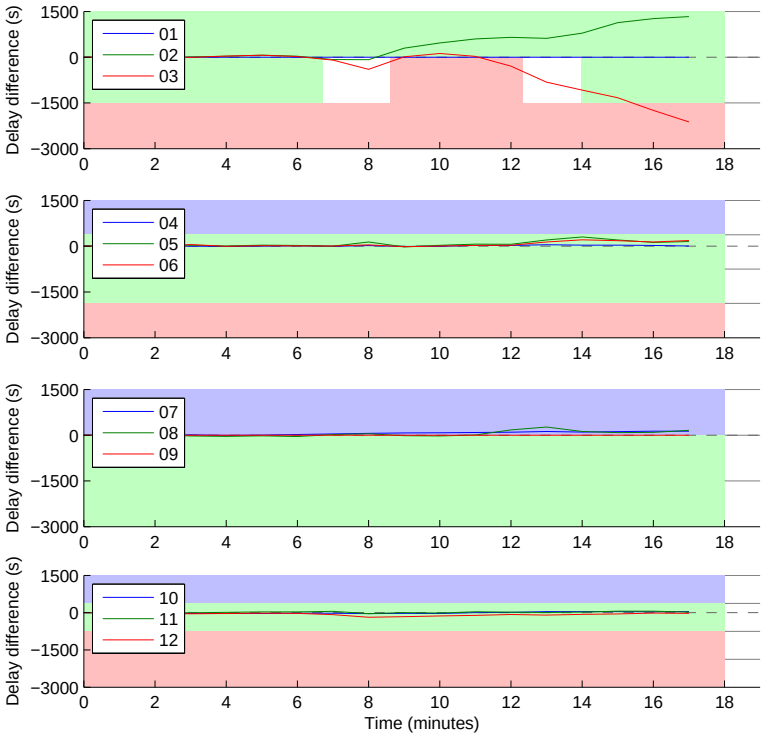


Figure 5-3: Difference in cumulative delay between $N_v = 8$ and $N_v = 1$; for step, $d = 250$, $K = 1850$, $\rho = 0.7$, showing the lane configuration of $N_v = 8$

Next the effect of switching will be studied more in-depth by comparing the results of a step experiment with $N_v = 8$ and with $N_v = 1$, where the last resulted in a static lane configuration. Consider the graph in Fig. 5-3, where the cumulative delay of $N_v = 1$ was subtracted from the cumulative delay of $N_v = 8$.

First notice the moment of switching, which is after the delay increase at $t = 6\,\text{min}$. This shows the controller does not correctly anticipates the effect of the sudden demand increase.

There is almost no difference between both delays for the movements 04 to 12, compared to the big difference for the movements 02 and 03. This shows the controller does not spread the effects of a switching lane to the movements on the other legs.

When the switching occurs at around $t = 7\,\text{min}$ the delay of $N_\text{v} = 8$ first goes up compared to $N_\text{v} = 1$, but not long after that the benefit of the switch starts showing and the delay of $N_\text{v} = 8$ goes down fast. The second switching moment around $t = 13\,\text{min}$ halts the discrepancy. In the end the delay decrease for movement 03 is much larger than the delay increase for movement 02, resulting in a net positive effect of switching.

In the other experiments, of which the graphs are not shown here, the same form was found. When the delay of the switching case was lower than the static case, this was because the delay of movement 02 did not go up as much as in the graph in Fig. 5-3.

5-2-5 Conclusions on evaluating the queue length-based method with fictional data

The prediction horizons $N_\text{v} = 4$ and $N_\text{v} = 8$ were found to perform equally well. A horizon of $N_\text{v} = 1$ resulted in a static lane configuration and led to higher delays. A horizon of $N_\text{v} = 12$ also resulted in higher delays, most likely due to a model-simulation mismatch.

The in-depth analysis provided three insights. First is that the green fractions of the movements involved with a switching lane dropped drastically during a switching period, while others went up, something that is assumed to be overshoot due to the step-like nature of switching. Second is that the time of switching was unlogical considering the timing of the demand increases, which can be attributed to a model-simulation mismatch. Third is that the switching only affected the movements involved with the switching lane.

5-3 Summary of evaluation with fictional data

In this chapter the two controllers that have been developed were evaluated using fictional data.

The delay-based method was found to behave as expected, switching at the right moment and thus showing the effectiveness of the MPC approach. Switching reduced total delay significantly in almost all test cases. The delay reduction by switching did not necessarily happen to the movements using the switching lane, instead other movements could benefit. This shows that the controller takes the whole intersection, including signal control, into account when deciding at what moments to switch. A discrepancy between

prediction model and simulation was identified for a single experiment, however it did not negatively influence the results.

The queue length-based method performed better when it was able to switch than when it was not. There were however some peculiarities in its behavior. One is that during the switching period, when the switching lane is closed, the green fractions of the movements using the switching lane dropped significantly, while others went up, an effect that most plausibly can be explained as overshoot due to the binary nature of switching. Second is that the switching did not occur at the moment when demand increased, but a bit later, which shows the prediction model did not correctly anticipate the queue length increases. Third is that unlike in the delay-based method the switching of a lane only affected the delay of the movements associated with that switching lane.

Chapter 6

Evaluation with real-world data

In this chapter the two control methods will be evaluated using real-world data.

In the previous chapter fictional data was used, and because the input signals were simple and well-known they were great for studying the controller behavior. In this chapter the goal is to investigate the controller behavior in a realistic situation. Further details were discussed in Section 4-2-2.

First in Section 6-1 the delay-based method will be evaluated, then in Section 6-2 the queue length-based method will be evaluated. For both intersection K302 was used. For brevity K359 is not included, since a few tests indicated the same insights were found as with K302.

6-1 Delay-based method with real-world data

In this section the delay-based method will be evaluated using real-world demand data. It will consist of three parts. First in Section 6-1-1 the effect of different time step durations will be presented. Next in Section 6-1-2 the results of both static and switching cases will be compared. Section 6-1-3 will further discuss this difference, using a single experiment for a more in-depth analysis.

In Section 5-1-1 a prediction model lane capacity of $K = 2000$ was selected for further use, it was used in this section as well. Furthermore a prediction horizon of $N_\mathrm{v} = 4$ and, for the switching cases, a control horizon of $N_\mathrm{c} = 4$ was used. A shorter prediction horizon of $N_\mathrm{v} = 1$ was briefly tested but did not yield satisfactory results. A longer prediction horizon of $N_\mathrm{v} > 4$ was not yet tested because it would result in significantly longer computation times. Instead the time step duration is varied to see the effect of the prediction horizon time span.

6-1-1 Influence of time step duration

First the influence of the time step duration on the results will be investigated.

The choice of the time step duration is a trade-off. A short time step duration means the signal timings can react quickly to fluctuations in demand, but the prediction horizon timespan will be short, for a fixed number of steps in the prediction horizon. A long time step duration results in a long prediction horizon timespan, but leads to less resolution in the signal timings, and, because of the way the delay-based method is modeled, will result in inaccurate signal timings in time steps where switching occurs.

In Table 6-1 the comparison of three time step durations for both the static and the switching case are shown. Clearly a higher time step duration results in more delay. It is unlikely that this effect is due to the inaccurate signal timings during time steps in which switching occurs, because the amount of these time steps is small compared to the total number of time steps, and as such the effect should be hard to notice. A more likely conclusion is that having high-resolution signal control is more important than having a longer prediction horizon timespan.

Table 6-1: Comparing time step duration values

	T_s (min)	Delay (1×10^5 s)	Diff. (%)	p (%)
static	2	7.817	1.7	0
	5	7.952	2.4	0
	15	8.139		
switching	2	7.789	2.3	0
	5	7.971	3.0	0
	15	8.212		

6-1-2 Comparing static and switching

Next the effect of switching is investigated by comparing the results of the static and the switching cases.

The results, for each time step duration, are presented in Table 6-2. For the two smaller time step durations the result is inconclusive as there is no significant difference between static and switching. For $T_s = 15$ min being able to switch led to a 0.9 % higher delay than the static case.

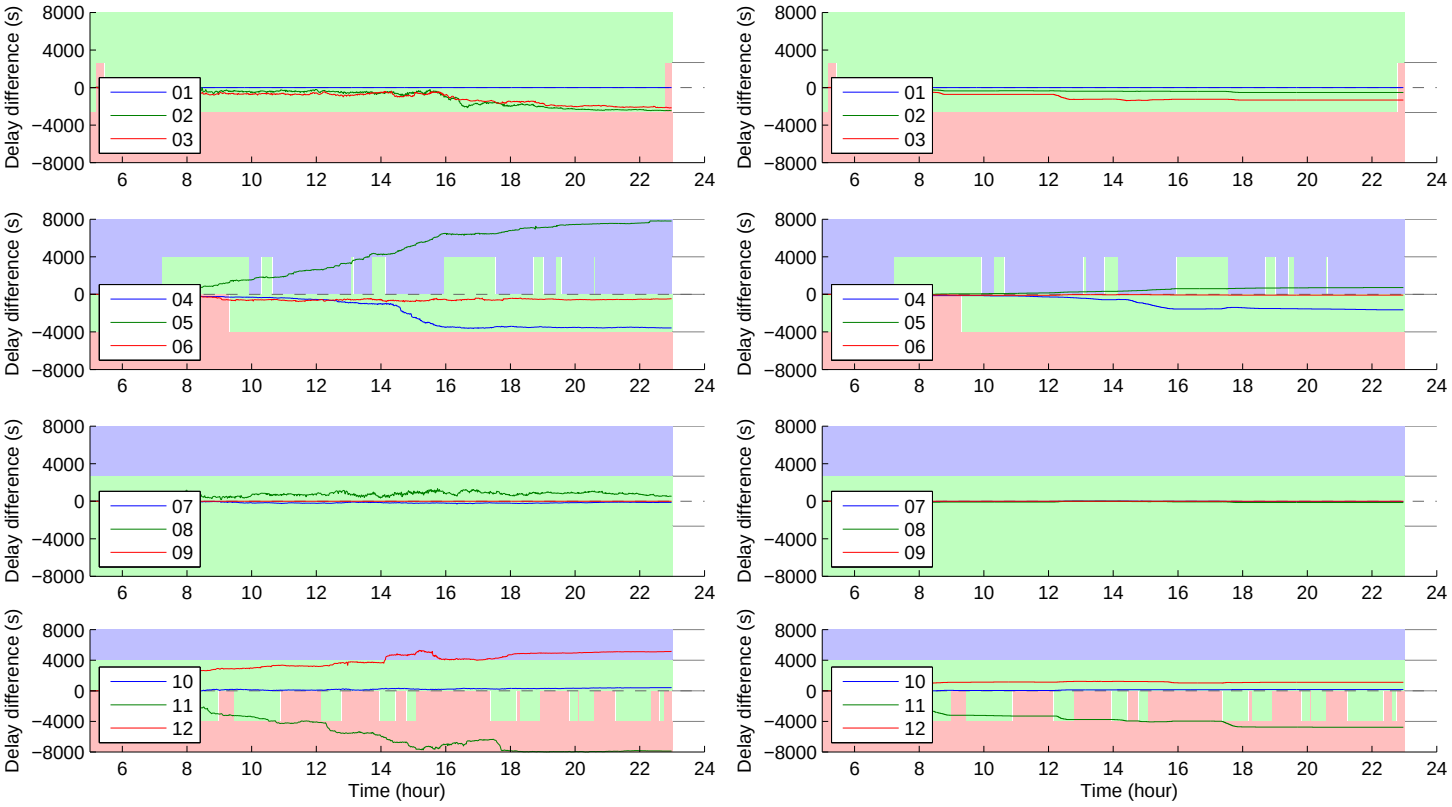
The conclusion that can be drawn is that the switching moments found by the delay-based method neither improve nor degrade the performance of the intersection.

Table 6-2: Comparing static and switching

T_s (min)	Delay (1×10^5 s) static	switch	Diff. (%)	p (%)
2	7.817	7.789	-0.4	12
5	7.952	7.971	0.2	8
15	8.139	8.212	0.9	1

6-1-3 In-depth look

To find out why switching did not decrease delay, the results using a time step duration of $T_s = 2\,\text{min}$, the value that resulted in the least delay, will be studied more in-depth.



(a) Simulation **(b)** Model

Figure 6-1: Difference in cumulative delay between static and switching; for $T_s = 2\,\text{min}$

Consider the graphs in Fig. 6-1, which show the difference in cumulative delay between static and switching. The delay of the static case was subtracted from the switching case, so for example a negative value means that switching led to less delay. In Fig. 6-1a the results from the simulation and in Fig. 6-1b the results from the prediction model

are shown.

The delay increase of movements 05 and 12 due to switching were much higher in simulation than was predicted by the controller. The delay decrease of movements 04 and 11 was also higher than predicted, but the difference is not as severe as with the delay increase. As a result the switching did not provide the overall delay decrease that was predicted by the controller. This indicates a model-simulation mismatch that greatly influences the controller performance.

6-1-4 Conclusions on evaluating the delay-based method with real-world data

High resolution signal timings lead to good performance, and as such the time step duration should be small.

The controller cannot sufficiently predict the delay increase of a movement which loses a lane when switching is applied. This mismatch between prediction and simulation can be due to a structural problem or wrongly calibrated parameters in the prediction model.

Overall, applying switching did not improve or worsen the results compared to the static case.

6-2 Queue length-based method with real-world data

In this section the performance of the second controller, the queue length-based method, will be evaluated using real-world data.

First in Section 6-2-1 the results of different prediction horizons will be compared, which will show the influence of the prediction horizon and the effect of switching. Next in Section 6-2-2 the results of the static and switching cases of a single prediction horizon value will be compared and studied more in-depth.

For these experiments a time step duration of $T_s = 1\,\text{min}$ was used, just as with the fictional data sets. Using the results from Section 5-2-1 a prediction model lane capacity of $K = 1850$ was selected. Furthermore as maximum degree of saturation both 0.6 and 0.7 were tested with real-world data, and 0.6 was found to result in less delay and was therefore used further.

6-2-1 Influence of prediction horizon

First the effect of different prediction horizon values will be investigated. The delay for each value and the differences between the values are shown in Table 6-3.

Using $N_v = 1$ resulted in no lanes being switched. Though the delay of this static case is lower than $N_v = 4$, in which switching did occur, the difference is not significant. The delay of $N_v = 8$ is significantly higher than that of $N_v = 4$.

Also considering the results of the evaluations with fictional data, these results can be explained by two insights. First is that a longer horizon will lead to higher delays, most likely because the decision to switch is based on increasingly worse predictions. Second is that being able to switch does not improve or degrade the performance compared to the static situation.

Table 6-3: Comparing prediction horizon values

N_v	Delay (1×10^5 s)	Diff. (%)	p (%)
1	6.752		
		0.4	20
4	6.778		
		0.6	1
8	6.816		

6-2-2 In-depth look

To study why there is no difference between the switching and non-switching case a comparison of the results of $N_v = 1$ and $N_v = 4$ will be studied more in-depth. Consider the graph in Fig. 6-2, which shows the difference in cumulated delay. Again a negative result indicates switching led to less delay and vice-versa.

First thing to notice is that there are many points were small switching actions occur, switching a lane for a period as small as a single time step. Though the effect of this chattering gets lost in the bigger picture, it most likely has a negative impact.

On the first leg one 'block' can be recognized at 12:00. This is a switching action that is beneficial. The delay of movement 03 drops sharply, while at the same time the delay of 02 only rises slightly.

Another 'block' that is beneficial is the one on the second leg between 14:00 and 16:00. The delay of movement 04 falls strongly while the delay of movement 05 rises less severe. It is not until around 17:00 that the delay of 05 increases, probably because of the chattering at that period.

Similar as in the evaluations based on fictional data the actions of a switching lane only seem to affect the movements associated with it.

The graph of the difference between $N_v = 8$ and $N_v = 1$, not shown here, is more or less similar to this one, with the only difference that the delay increase of movements 02, 03 and 05 is a bit higher.

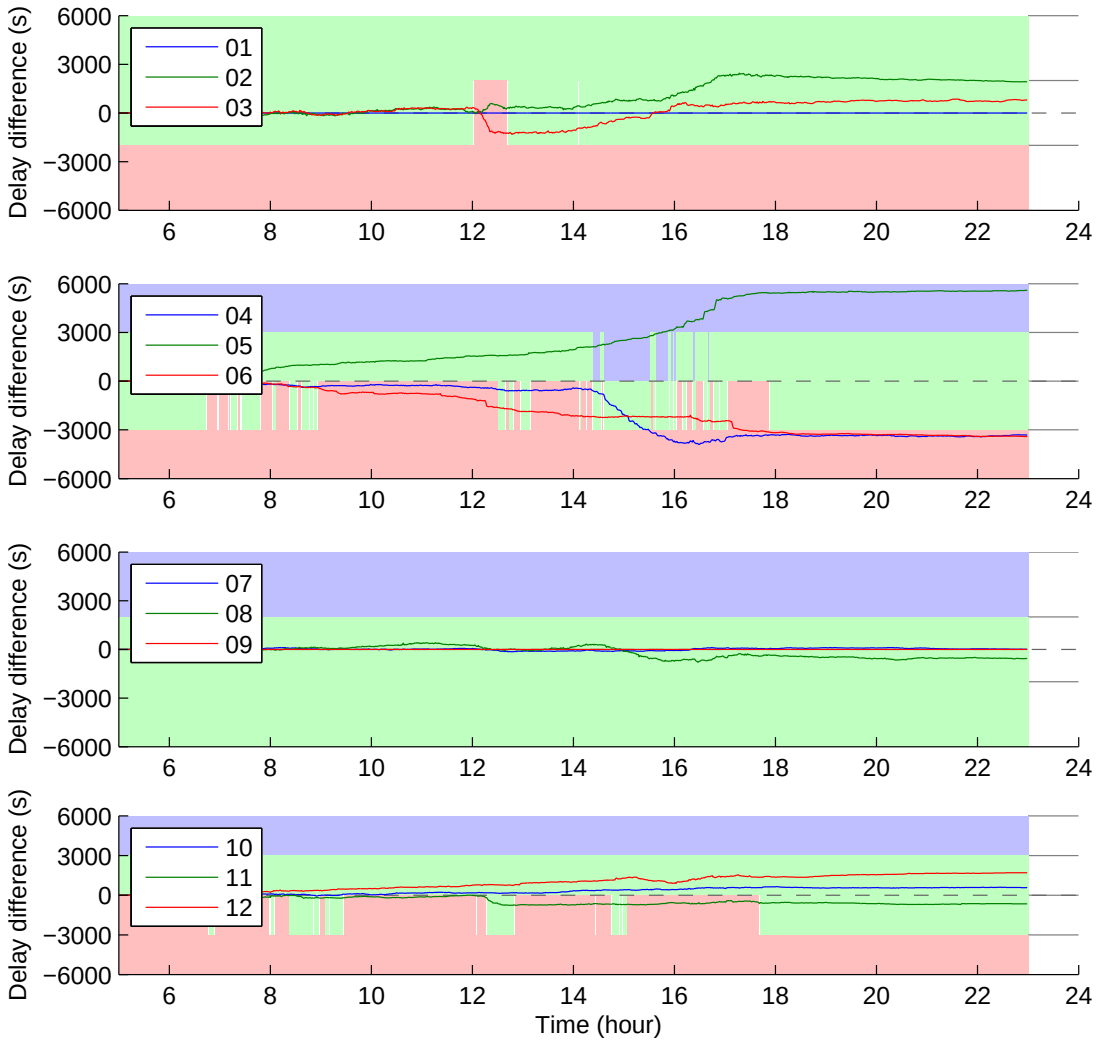


Figure 6-2: Difference in cumulative delay between static and switching

6-2-3 Conclusions on the queue length-based method with real-world data

A longer prediction horizon seems to lead to a higher delay. This can possibly be explained by a model-simulation mismatch, due to which the decision to switch is based on increasingly worse predictions.

Applying a prediction horizon that resulted in switching did not decrease delay compared to a prediction horizon that resulted in a static lane configuration. Though some switching actions that decreased the overall delay can be identified, the net result is a very small difference between static and switching.

At some points chattering occurred, where a switching lane was opened and closed a couple of times in a short time span. This is not entirely unexpected, since there is no constraint on this behavior in the controller.

6-3 Conclusions on evaluation with real-world data

For both the delay-based method and the queue length-based method switching did not noticeably improve or degrade the performance compared to not-switching. This could mean that the demand profiles of the intersections that were selected are such that there is no benefit in using switching lanes. Were that the case, than the controllers should have found a static lane configuration even when switching was allowed, which they did not. So it is more likely that the controllers were both unable to find the right moments to switch.

Chapter 7

Controller comparison

In this chapter the two controllers developed in this study will be compared with each other. This will be done by using a benchmark, which is a state-of-the-art signal controller that is assumed to perform consistently for each test case. The differences between each controller and the benchmark can be used to compare the two controllers with each other. Moreover, these differences also show the performance of each controller compared to the state-of-the-art signal controller.

In Section 7-1 first the comparison method will be introduced. This will include a short description of the benchmark and an overview of the parameters selected for this comparison. Then in Section 7-2 the results of the comparison will be presented and discussed.

7-1 Comparison method

7-1-1 Juno

The benchmark that is used to compare the controllers is called Juno (Junction Network Optimizer) and is a state-of-the-art signal control method [28]. Juno is assumed to perform consistently, giving the opportunity to compare the two controller developed for this thesis. Juno uses an MPC approach to find the optimal signal order and timing. It is a detailed method, taking each vehicle individually into account. In this study it was applied on the same simulation as the other two controllers.

7-1-2 Simulation knowledge of each method

The delay-based method uses demand data, which for the real-world data case is smoothened with a low-pass filter. The queue length-based method also uses this demand data, and furthermore measures the queue length during simulation. Juno does not need demand data, but measures individual vehicles when they pass detection loops. In the simulations done for this study Juno is assumed to measure the location and speed of each vehicle perfectly at the stop line-, extension- and entry detector loops.

7-1-3 Result selection

Only the results of intersection K302 are considered. The methods are compared for each of the five cases that have been used so far. For each case the best result of the delay-based method and the best result of the queue length-based method are selected to be compared with each other and with the result from Juno.

The results that were selected are shown in Table 7-1. Note that for the fictional data experiments of the queue length-based method only the results of $N_v = 8$ where considered, since not every parameter combination was tried, and $N_v = 8$ seems to perform well in these tests. For the real-world data test of the delay-based method strictly speaking not the lowest value was selected, since this was the static case of $N_c = 1$. Instead $N_c = 4$ was used, which did not differ significantly from $N_c = 1$.

Table 7-1: Result selection for comparison

	Delay-based				Queue length-based			
Case	K	N_c	N_v	T_s	K	ρ	N_v	T_s
27 May 2014	2000	4	4	2	1850	0.6	4	1
Step $d = 200$	1900	2	4	2	1850	0.6	8	1
Step $d = 250$	1900	4	4	2	1850	0.6	8	1
Sine $d = 200$	2000	4	4	2	1900	0.7	8	1
Sine $d = 250$	1900	2	4	2	1900	0.7	8	1

7-2 Comparison results

The comparison using delay as performance criterion is shown in Table 7-2, the comparison using the number of stops is shown in Table 7-3. Most important aspect of these tables are the differences between the two controllers and Juno.

Juno performs better than the two controllers developed in this study, which is not unexpected since Juno is very detailed, taking each individual vehicle into account, whereas the two other controllers operate on a less detailed level. Even so, for the cases

Table 7-2: Comparison of three controllers, using delay as performance measure

	Juno	Delay-based			Queue length-based		
	Delay 1×10^4 s	Delay 1×10^4 s	Diff. (%)	p (%)	Delay 1×10^4 s	Diff. (%)	p (%)
27 May 2014	56.26	77.89	38	0	67.78	21	0
Step $d = 200$	1.418	1.588	12	0	1.829	29	0
Step $d = 250$	2.241	2.437	9	6	2.860	28	0
Sine $d = 200$	2.608	2.770	6	3	3.212	23	0
Sine $d = 250$	4.089	4.187	2	40	5.002	22	0

Table 7-3: Comparison of three controllers, using the number of stops as performance measure

	Juno	Delay-based			Queue length-based		
	No. of stops	No. of stops	Diff. (%)	p (%)	No. of stops	Diff. (%)	p (%)
27 May 2014	21950	29430	34	0	26550	21	0
Step $d = 200$	558	625	12	0	703	26	0
Step $d = 250$	795	846	6	8	1016	28	0
Sine $d = 200$	1004	1090	9	0	1166	16	0
Sine $d = 250$	1442	1461	1	51	1659	15	0

with a high base demand the difference between the delay-based method and Juno was not significant, indicating they performed similarly.

The differences in delay and the differences in number of stops seems consistent with each other. The order of magnitude is similar for each case and if the difference is not significant, it is so for both the delay and the number of stops. This indicates that measuring the number of stops is not necessary if already the delay is being measured.

When comparing the fictional and real-world cases it is interesting to see that while the difference between the queue length-based method and Juno is consistent, between 21 and 29 % in each case, the difference between the delay-based method and Juno is not so consistent. Comparing those for the fictional cases gives a difference of 1 to 10 %, while they differ with more than 25% in the real-world case. This indicates that the delay-based method performs relatively bad in the real-world case. This is most likely due to the fact that this control method does not use actual feedback from the simulation, and thus model-simulation mismatches do not get solved during simulation. The fictional demand experiments are relatively short and so in those this effect is probably less noticeable.

7-3 Conclusions on controller comparison

Juno is a more sophisticated signal controller compared to the signal control used in the two controllers. Even so, the delay-based method performed similarly in the fictional data evaluations with high base demand. This shows that when the intersection becomes saturated lane switching is very beneficial.

The delay-based method performed much worse on the real-world case in comparison with the fictional cases. This shows the controller has issues with robustness during a long simulation.

Chapter 8

Discussion

In this section the results and conclusions will be discussed. First in Section 8-1 the design and the evaluation of the delay-based method will be discussed. Then in Section 8-2 similar topics but now for the queue length-based method will be addressed. Finally in Section 8-3 some general discussion points, including the comparison of the controllers, will be included.

8-1 Delay-based method

Global optimization Since the delay model that is being used is non-convex, the question was raised if the solutions found by the SQP solver are global minima. To investigate this a multi-start technique was used, which in all cases found the same solution as the single-start. Though it cannot be proven easily, the suspicion is that the delay-model is monotonically decreasing. It can be shown that the delay formula by Akçelik is strictly quasi-convex, whether that is also the case for the combination of Akçelik's model and the initial queue delay model has not been investigated. In any case, the multi-start tests showed a single solution is found for each starting point, which was interpreted as a proof the solver could find the global minimum from a single starting point.

Solver feasibility Feasibility is not a problem in the optimization. There is an upper constraint on the cycle time, but this does not result in infeasibility for high demands. Instead when the intersection becomes over-saturated the delay goes up, but a solution can still be found.

Switching period modeling The way the switching period is implemented in the controller model has some downsides. The saturation flow during the period the switching lane is closed is averaged with the saturation flow after the lane is opened, where both factors are weighted based on the time of the lane closure and the time step duration. This leads to green times that are too short during the lane closure and too long afterwards. The reason for this modeling choice is that calculating signal timings for each switching period separately would increase the size of the optimization problem with approximately 50 %. Fortunately the impact of this decision is expected to have been small, since it would have only been a problem in the real-world data evaluations, in which only very few time steps contained a switching action.

Feedforward For the short and predictable fictional data tests the delay-based method performed well, but for the long real-world data experiments its results deteriorated. This is most likely due to the fact that the controller has limited information on how well it is performing. The only information the controller uses is a demand forecast, which is based on for example measurements on the approach lanes and on data from neighbor intersections. The controller does not know if its green times are long enough, this is essentially a feedforward process.

Model-simulation mismatch The discrepancy between the delay of the simulation and the prediction model was between 26 and 36 % in the final real-world experiments. If this figure could be reduced the performance of the controller would almost certainly improve. The mismatch could be due to a deficiency in the model, or a wrong calibration, or both. It would be interesting to use a mechanism that updates the prediction model based on the controller performance, thus adding feedback to the now feedforward system.

Short control horizon Using a control horizon of $N_c = 1$ is not advisable, because then the controller is not able to make the distinction between applying a lane switch now or later, since it has only one option. With $N_c = 2$ or larger this choice does get considered and the controller can let the switching happen at an optimal time. Therefore it is recommended to use a control horizon of $N_c \geq 2$.

Time step duration A major downside of the delay-based method is its trade-off in choosing a time step duration. For signal control a short time step is beneficial, since then signals can be updated quickly to handle the randomness of vehicle arrivals. A long one is beneficial for prediction, because then the controller can plan the lane switches better by looking further ahead. This could of course also be achieved with a prediction horizon with more steps, but that would increase the computation time significantly.

8-2 Queue length-based method

During switching In the fictional data tests the green fractions of the movements involved with a lane switch were found to drop drastically during the switching period. At the same time, in some cases, green fractions of other movements rose. The most likely reason for this behavior is that the sudden changes result in overshoot. The effect of the binary variables is similar to a step response and the system needs a few time steps to adjust. The simulation may behave very differently to the sudden change than the prediction model anticipates, leading to a mismatch between simulation and prediction that results in the unwanted oscillations in the control input.

Model-simulation mismatch It is practically unavoidable that there is a mismatch between prediction model and simulation or plant. It is expected to be the cause of the delayed reaction of the controller to the step increase and also part of the explanation for the overshoot in the switching period. A better calibration of the prediction model could prove beneficial.

Larger prediction horizons Contra-intuitively the controller in general performed worse for larger prediction horizons. This could be because of the model-simulation mismatch. The switching decision for the next step is based on predictions that are, with a larger prediction horizon, increasingly incorrect. This could result in a short horizon having a better performance, since the prediction is more accurate.

Measurements The queue length needs to be measured in order for this controller to function. In Vissim it is easy to detect all vehicles on a link and also know their destination, in reality these measurements are not so straight-forward. The interested reader is referred to [29] for a state-of-the-art solution for queue length estimation.

Quadratic cost function The cost function of the main optimization problem consists of the predictions of the queue lengths. This cost function was written as a quadratic function, which in retrospect would not have been necessary, as a linear combination of the queue length prediction would also have worked. That would have made solving the optimization problem easier. It is assumed that using a quadratic combination of predicted queue lengths instead of a linear one did not negatively impact the results, since the effect of both would have been similar.

Continuous queue lengths The queue lengths are modeled using continuous variables, while in reality and also in the simulation they are discrete. For large queue lengths this difference is negligible, but for small queue lengths this further increases the simulation-model mismatch. Continuous queue lengths were used because, as is also explained in [30], it reduces the complexity of the optimization problem.

Queue length constraint An important constraint in this method is the linear equality constraints that demands the queue length to be equal to or larger than zero. This is necessary since the queue length is minimized. The constraint has some side-effects. One is that when demand is low, the controller may close a switching lane for both movements. This effect was largely avoided by adding the binary lane use variables as minor optimization criterion. Another additional fix could be, when the controller returns a control input that wants to close a lane, to look at the intention. If the lane is not closed to be opened later for the other movement, it is an unnecessary closure and thus should not be implemented. The history of binary variables should also reflect this choice. This approach could perhaps solve this problem, but a more elegant solution would be welcome.

Maximal green times In the secondary optimization problem that translates the green fractions into signal timings, currently no incentive is given to maximize the green times. They are constrained to be at least as large so as to correspond with the green fractions found in the MPC-scheme, but if they can be larger than that without interfering with other movements there is no incentive for them to increase. A simple fix would be to add the green times as a minor optimization criterion. Since the green fractions found by the MPC optimization should be enough to handle the available traffic, it is uncertain how big the improvement with this fix would be.

Stability The system matrix of the prediction model that was used in itself is marginally stable. Without a control input a queue will dissolve nor grow. Still, stability can be an issue when a model-plant mismatch or a time delay is not handled well. There were no cases where the queue length state became unstable, so an upper bound on this state was not necessary. There was however some chattering in the real-world data tests. These oscillations of the binary variables should be prevented in future implementations of the queue length-based method.

Control horizon In the experiments for this thesis the control horizon constraint was not used. It was found that it did not improve the computation time drastically and was therefore left out to reduce the number of experiments. Future studies could include a control horizon in the experiments. It would be interesting to see what the effects on the stability are.

Solver feasibility Feasibility of the optimization problems was not an issue. Indeed the CPLEX solver almost never exited its routine with an error about infeasibility. The second optimization problem, the LP problem that finds the green times, in the real-world data tests did sometimes exit because it could not find a feasible solution. These occurrences were incidental, so it was solved by using typical signal timing values for that time step instead.

8-3 General discussion

Filtering of demand data For the real-world data experiments the choice was made to feed an unfiltered version to Vissim, and a filtered, more smooth, version to the controllers. This way the vehicles arrived more randomly, just as they had done in reality. At the same time the controllers received a signal more like a demand prediction they would get in reality. In hindsight it would have been better to give both controller and simulation the same signal in order to see how well the controllers worked. This was tested shortly during the writing of this thesis, with the real-world data and the delay-based method, and it was found that although overall delay was lower the same differences between static and switching occurred.

Juno The results of Juno were mainly intended to compare the two controllers with each other. Nevertheless, it must be noted that Juno performed as good or better than the two controllers, which is not strange, since Juno is a very detailed state-of-the-art signal controller. This shows that a switching lane controller should include a good signal controller as well. As will also be proposed in the recommendations in Section 9-3, combining Juno with one or both of the controllers would be an interesting study.

Delay-based v.s. queue length-based The two controllers developed in this thesis both have their positive and negative aspects. The delay-based method has a high modeling power, as was seen in the fictional data tests. The downsides are its computational complexity and its robustness issues during longer tests. The queue length-based method on the other hand is less computational complex and is more robust, since it does contain a feedback loop. Its downside is that, as was seen in the fictional data tests, it is not so good at placing the switching moments. In that sense both controllers can be seen as each others opposites.

No clear conclusion on which is better, or which has more potential can be given. Both controllers can be improved, as will be discussed in the following chapter.

Chapter 9

Conclusions and recommendations

This chapter consists of three parts. First in Section 9-1 a short summary of this study will be provided. Then in Section 9-2 follow the conclusions, which are formulated based on the questions that were posed in the introduction. Section 9-3 concludes the chapter and also this thesis with recommendations for further studies.

9-1 Summary

In this thesis two controllers were designed that integrate signal timing and switching lane control of a single intersection. This study has been mostly exploratory, generating insights on how a dynamic lane configuration works and how it can be controlled.

In literature a single paper on dynamic lane configuration control was found, which was based on an optimization model for static lane configuration design. The method presented in the paper did not use prediction, did not include signal timing, and minimized the flow ratio, which does not necessarily lead to the least delay. The two controllers developed in this study aim to improve on this work.

Both controllers use a Model Predictive Control approach. The first minimizes delay using a three-term model, which results in a Mixed-Integer Non-Linear Programming problem. The second minimizes queue length using a simple store-and-forward model, which results in a Mixed-Integer Quadratic Programming problem. The biggest differences between the controllers are that the first is more oriented on its prediction model and uses longer time step durations, while the second is more oriented on faster measurements and faster computation and uses shorter time step durations.

Both controllers were evaluated using PTV Vissim, a microscopic traffic simulation program. Fictional data with a simple step and sine increase as well as real-world data

were used. In the fictional data tests both controllers showed that switching resulted in less delay than a static situation. In the real-world data test no significant difference between static and switching was found. The state-of-the-art signal controller used as benchmark performed equal or better in all cases.

The most important contribution of this thesis is that for the first time signal control and switching lane control have been integrated. With the two control methods developed in this work, situations in which the use of switching lanes provides benefits in terms of delay were identified. The two controllers provide a basis for further research on dynamic lane configurations.

9-2 Conclusions

Based on the questions that were posed in Chapter 1 and which are repeated here, some general conclusions can be formulated.

What type of controller is suitable for handling a dynamic lane configuration system?

A predictive controller. As shown in Section 5-1-3 the delay-based method planned a switching procedure *before* a sharp demand increase instead of after, which is an example of why a predictive controller is most suited for a dynamic lane configuration.

What is the impact of a switching lane on the whole intersection?

In the work by Zhao et al. presented in Section 2-3-3 a switching lane is assumed to only influence its own leg and the opposite leg. In this study the delay-based method showed that a single switching lane could positively affect the delay of all movements on the intersection.

Should the controller consider both signal and lane control in an integrated fashion?

Yes, the controller should consider signal timings when determining the lane configuration. The delay-based method showed that this enabled the controller to divide the delay benefit over the intersection. However, the comparison with Juno also showed the importance of including good signal control. Therefore a separate high resolution signal controller together with a lane controller will most likely yield better results than integrated signal and lane control.

Can the controller work online, operating while the intersection is being used?

Yes, both of the controllers are fast enough to work in real-time. However, this does depend on the choice of time step duration and prediction horizon.

What is the effect of using prediction?

The fictional demand data tests showed the usefulness of the prediction of the delay-based method. On the other hand the queue length-based method performed worse for longer prediction horizons, most likely because with a longer horizon its predictions get worse. Both insights are as expected, since the delay-based method emphasizes its prediction model, while the queue length-based method is designed to be fast and more responsive to measurements. In conclusion, the predictions are necessary to plan the switching moments, but to what extent depends on the controller design.

Does applying the controller result in an optimal lane configuration?

Yes and no. Yes, because the fictional demand data tests showed that both controllers find a dynamic lane configuration that reduces delay compared to the static situation. The delay-based method is best in finding the optimal switching moments, since it placed those at the expected moments in time. No, because the real-world data tests showed that switching did not improve or degrade the results compared to the static situation, which can thus be said to be optimal. The controllers should have found a static configuration, but they did not, indicating that they are not yet suitable for a real-world implementation.

9-3 Recommendations

Little research was available on the topic of dynamic lane configurations. Therefore this study was mainly exploratory, finding insights on how such a system works. With the findings of this study many interesting prospects for future work can be defined.

9-3-1 Short extensions on this study

Previous work A first extension on this study should be to use the data from the paper by Zhao et al. presented in Section 2-3-3 and compare the results with the results presented in that paper. This would show how much is gained by the improvements this study made on the work by Zhao et al. Their method did not use prediction and did not integrate signal control, so a comparison would show the importance of both.

Prediction model improvements For both the delay-based method as the queue length-based method the prediction models could not perfectly predict the simulation behavior. Efforts to improve these models, by more careful calibration, or making adjustments to their design, will further improve the controller performance.

Queue length-based method: small improvements Another extension could focus on making small improvements on the queue length-based method. In the discussion two simple additions were proposed: firstly to use the maximum green time instead of the needed green time, and secondly to take the intention of the controller into account when the control input suggests a lane switch. Two other aspects that should be investigated are the effect of using a control horizon, and the causes of and a solution for the overshoot behavior when a lane is switched. Lastly an attempt could be made to find the longest horizon that still yields acceptable computation times, which could be accompanied by making the optimization problem more compact and efficient.

Phase order In this study a fixed phase order was used. But as also discussed in the introduction, the determination of the phase order could very well be included in the control methods developed in this thesis. Tests should show how beneficial this addition is, and how much computational complexity is added by it.

Other road users This study only considered passenger cars. A future study could repeat the evaluations performed in this work, but using more realistic traffic compositions, including vans, trucks and buses. Furthermore pedestrians and bicyclists were not included as well, and their role in the control of switching lanes could be investigated as well.

9-3-2 Extensions on this study

Delay-based method: add feedback An important insight of this study is that the delay-based method is less robust for longer simulation durations. This was associated with the controller being essentially feedforward, since its control input does not influence the measurements. A future study could add a feedback loop to the design, for example in the form of a mechanism that alters the prediction models parameters based on its performance.

Better signal control The two controllers were designed mainly for switching lane control and as such their signal control parts are not state-of-the-art. An obvious improvement to both controllers would be to include a better, more detailed signal control part. For the delay-based method this could be done by running two instances simultaneously, where the first finds the lane configurations using a long time step duration, and the second updates the signal timings after each cycle. For the queue length-based method this could be achieved by releasing the signals-per-cycle approach and instead find the signal states directly.

Alternatively, an external state-of-the-art signal controller, such as for example Juno, could be combined with a switching lane control approach such as the delay-based method. This could potentially combine the best aspects of both.

Intersection identification An important aspect of dynamic lane configurations is selecting which intersections are suitable for implementing such a system. Future work could develop a systematic way to asses whether an intersection could benefit from using switching lanes.

Hardware for implementation There is currently no guide on how to implement a dynamic lane configuration in practice. Given the intersections with a switching lane presented in Chapter 2 and the hardware solutions listed in [2], a plan for a real-world implementation of a dynamic lane configuration system can be made.

Road user acceptance In this study human behavior has not been considered, even though it is suspected that it has a big influence on the performance of a switching lane. A future study could research the behavior of road users when using a switching lane.

9-3-3 Propositions on flexible use of infrastructure

Fully dynamic intersections Future work could include the development of a way to plan the configuration of the entire intersection, including the split between approach and exit lanes. This would not only be a challenge to model and control, but also to design a practical implementation. This type of research could however very well connect with the field of intelligent vehicles and should also be a part of the now following proposition.

Network configuration This study considered a single, isolated intersection, while currently a lot of research investigates the control of urban road networks. A study into the application of dynamic lane configurations on a corridor or even a network of intersections would be a significant contribution to the field of flexible infrastructure.

9-3-4 Work for the future

Connected cars The field of traffic control is expected to change drastically over the coming decades with the advent of connected cars and connected infrastructure. This thesis fits into this story in two ways. First is the demand prediction. With enhanced data sources the quality of demand predictions will improve, which will benefit the control of switching lanes. Second is that information can be communicated directly to connected cars, so that to implement a switching lane less hardware is needed at the intersection.

Self-driving cars When fully autonomous self-driving cars are omnipresent, the intersection as we know it today will no longer exist. But until that time the world faces a long transition period. Methods for the use of intersections by solely self-driving cars already

exist [31], but what remains a question is what should happen during the transition period. Future work could try to combine the design of a dynamic lane configuration, one that can be implemented right now, with a plan for how the same system can transform over the years to accommodate the increasing number of self-driving cars. One could for example imagine that as their number grows self-driving cars will use a separate approach lane and get a separate part of the cycle. This research could extend over multiple topics, including network management, flexible use of infrastructure throughout the network, driver behavior and acceptance, and field tests and evaluations.

9-3-5 Final words

Although futuristic visions of how we will drive through our cities in fully automated cars without congestion or delay are attractive and do help with creating funds and public awareness for this topic, the fact remains that methods that can make our cities more livable right now are still very welcome. Dynamic lane configurations at intersections could be such a solution. This thesis provides a step towards implementation of switching lanes in practice, and although more research is clearly needed, it is still believed that the development and evaluation of flexible infrastructure systems can make our drives shorter, cleaner, and more comfortable.

Appendix A

A-1 Intersection layouts

Actual layouts of intersections K359 and K302 are displayed in Fig. A-1 and Fig. A-2.

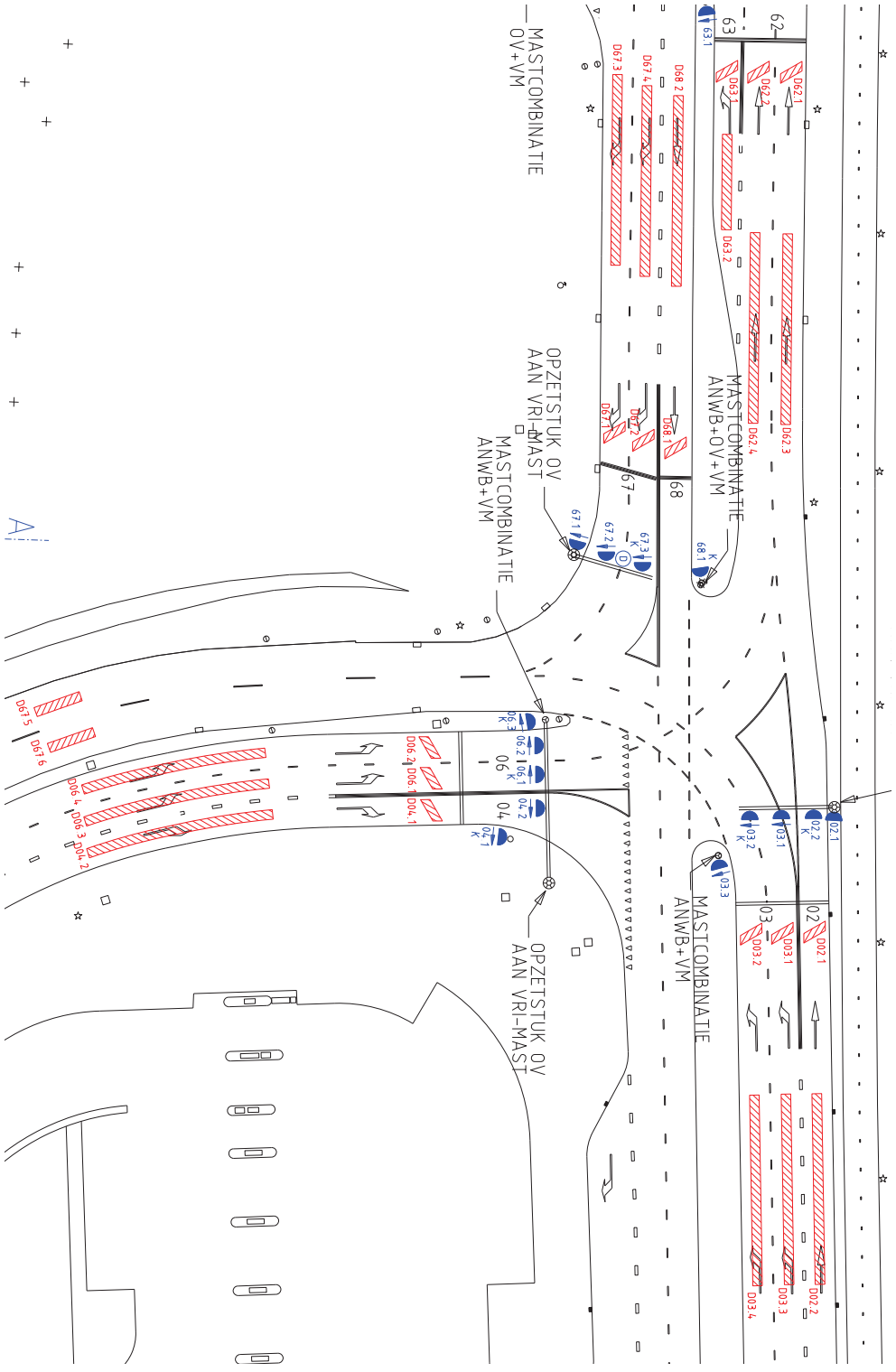


Figure A-1: Layout of intersection K359 (source: municipality of The Hague)

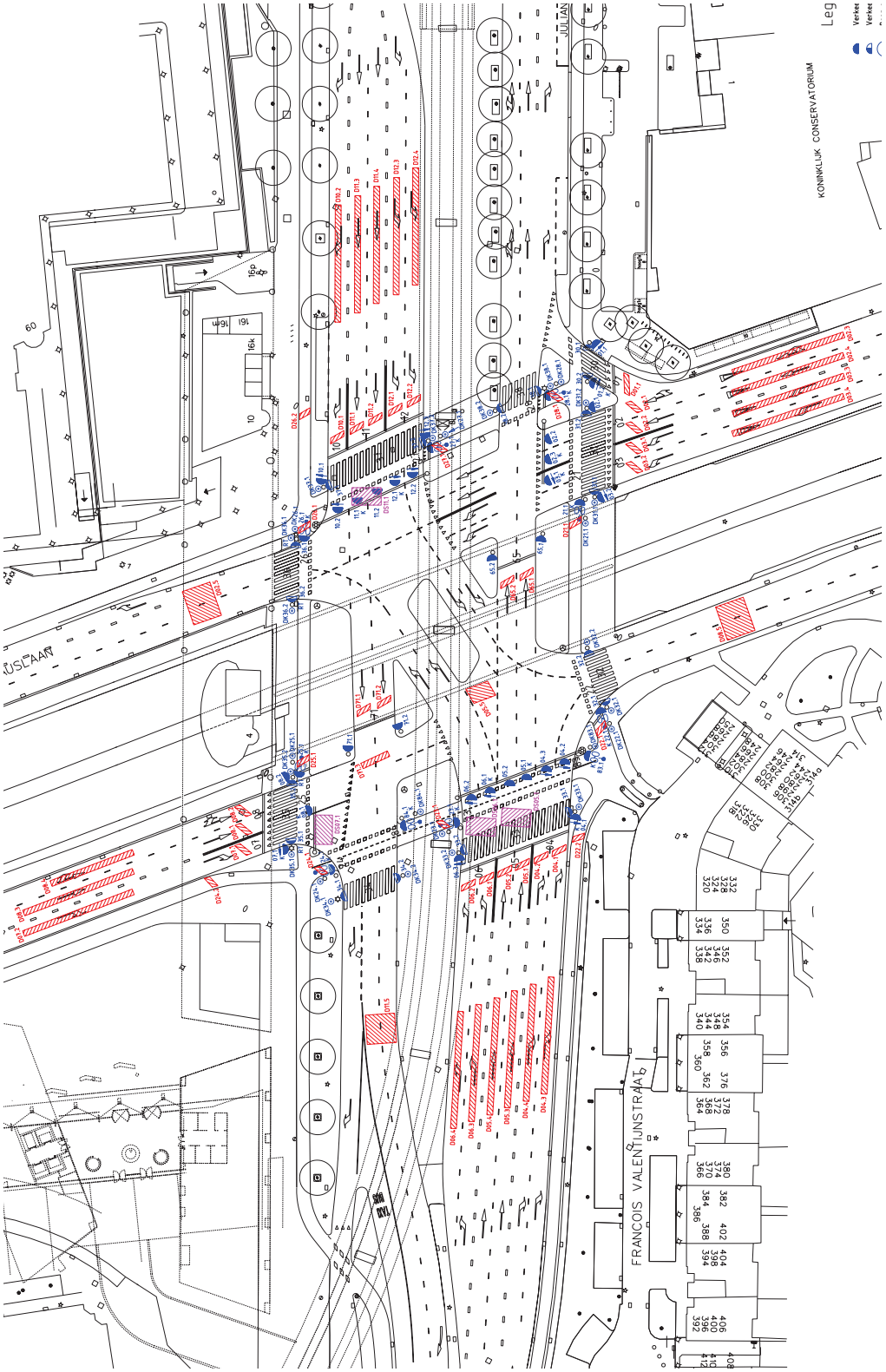


Figure A-2: Layout of intersection K302 (source: municipality of The Hague)

A-2 On Vissim

First some terminology used in Vissim:

- Link: a stretch of road that consists of one or more lanes
- Connector: similar to a link, but connects two links
- Vehicle Input: a simple way to enter vehicles in the simulation. Vehicles are placed at a point on a link but do not have a destination.
- Route: a simple way to give vehicles a destination. At a point on a link vehicles are given a destination according to a certain split.
- Dynamic Assignment: a more advanced way to place vehicles in the simulation and guide them to their destination. Vehicles are placed at a point on a link and then try one of the paths through which they can reach their destination. When the destination is reached the costs of that certain path are stored. Paths with lower costs will be chosen more often in future time steps.

A-2-1 Switching lane design

The options for a switching lane implementation that were investigated will be presented here. The general idea is to be able to open or close the switching lane for certain movements during simulation while not influencing the fidelity of the results.

For finding a way to implement lane switching in Vissim the following points were taken into consideration:

- Ideally it must be possible for a lane, or a link, to be temporarily closed, as if a red cross was applied to an electronic message sign above the road.
- This must be possible to do during simulation.
- Ideally the approach of a leg would consist of a single link, such that vehicles can still switch lanes.
- If that is not possible, the approach of each movement should consist of a single link.

Lane Closure Vissim has a built in feature with which a single lane or multiple lanes of a link can be closed. This is meant to simulate the closing of a lane on the freeway. A vehicle enters the link on the closed lane, sees that the lane is closed and will try to change lanes. If all lanes on a link are closed, the vehicle will still enter the link and will not change lanes. This method can not be used to close a link.

Link/Connector Closure Each link and connector in Vissim has the option to be closed and this works just as intended: once closed vehicles will not be able to enter the link. This option can also be accessed through COM. Unfortunately this feature cannot be changed during simulation, making it unfit to be used to simulate switching lanes.

Costs Costs can be attributed to each link, such that links with high will be avoided by vehicles. A possible way to close a link would be to increase the costs of that link drastically. Unfortunately just as with Link Closure these values cannot be changed during simulation.

Snapshot Vissim has a snapshot feature that can be triggered through COM. What it does is make a snapshot of the state of the simulation. It can then be stopped and closed; after opening again the snapshot can be loaded such that the simulation continues where it left of. With this method Link Closure can be used to open and close links during simulation. Downside is that evaluation data and signal control states are not saved in the snapshot and that the whole process adds time to the runtime.

Dynamic Assignment Since Dynamic Assignment is a way to dynamically place and guide vehicles it seems reasonable to assume it is possible to change this guidance during simulation. Unfortunately it seems not possible to force Vissim to update its path choice through COM.

Vehicle Input + Routes The rate with which a Vehicle Input places vehicles in the simulation can be changed through COM during simulation. Also the way a Routing Decision point distributes the vehicles over the destinations can be changed through COM during simulation. With this a switching lane mechanism can be build in Vissim. Downside is that the routing choices are placed in the hands of the user entirely, whereas with Dynamic Assignment the simulation would decide routes based on costs, resulting in a more natural route choice. Furthermore routes work on the link-level, such that each approach lane must be modeled by a separate link in Vissim.

A-2-2 Intersections in Vissim

K359

The implementation of intersection K359 in Vissim can be seen in Fig. A-3. Each movement has a link with one lane on which the Vehicle Input is placed. That input link is connected with two links, one for each possible approach lane. A Routing Decision placed on the input link distributes the vehicles over the two links. Note that the middle lane is modeled by two links placed on top of each other. It is the task of the control

algorithm to make sure the lane cannot be used by vehicles from both movements at the same time.

Reduced Speed Areas have been applied to the curved connectors to simulate the reduced saturation rate of left- and right-turning traffic.

K302

K302 was implemented in Vissim as shown in Fig. A-4. This is according to the schematic representation in Fig. 4-3. Movement 02 and 03 originate from the south-south-east, the other movements follow clockwise. Each movement has its own vehicle input on a link with a single lane. That link is connected with one ore more long links, which are the possible approach lanes for that movement. If there are multiple a routing decision on the input link distributes the vehicles over the approach links.

Again the curved connectors have Reduced Speed Areas in order to simulate the reduced saturation rate for left- and right-turning traffic.

Note that the links for movements 01 and 09 are still present in the model, even though they are not used.

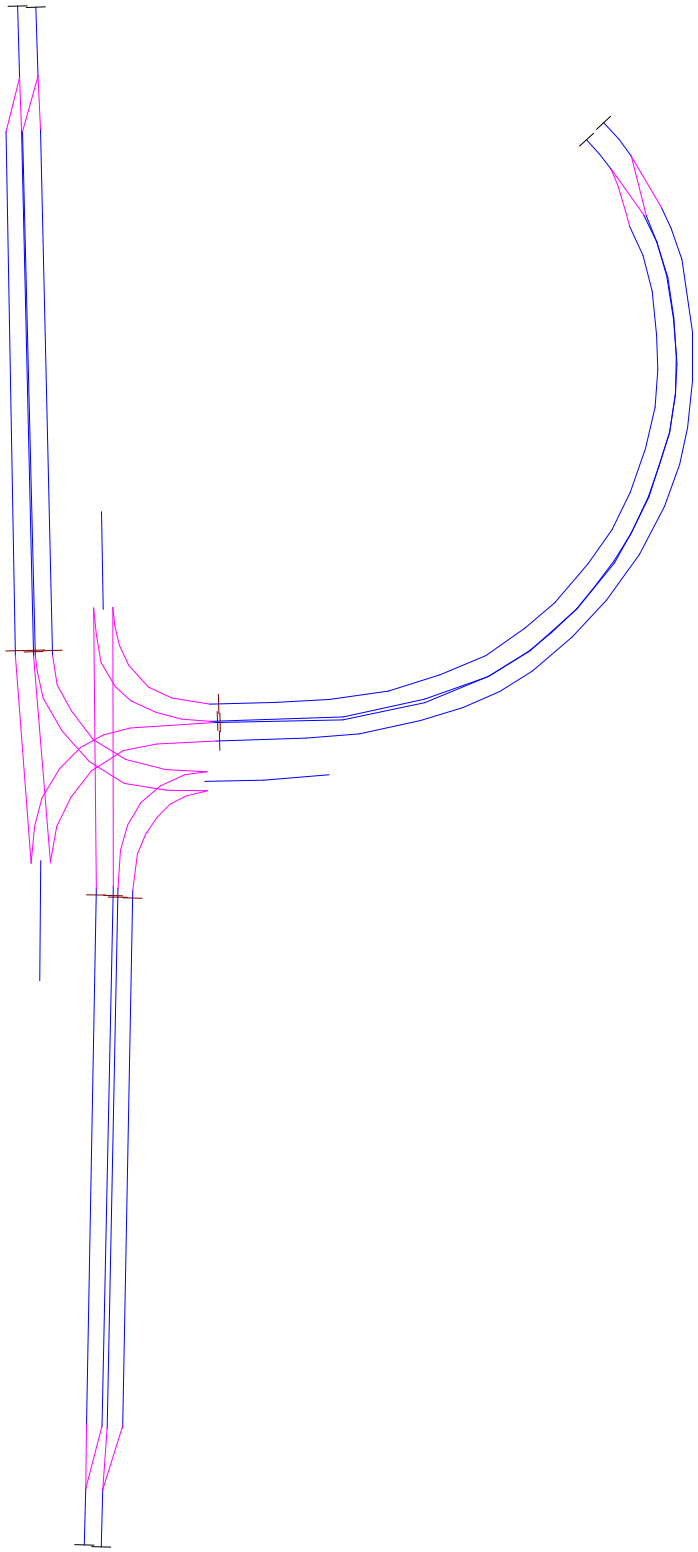


Figure A-3: Implementation of K359 in Vissim

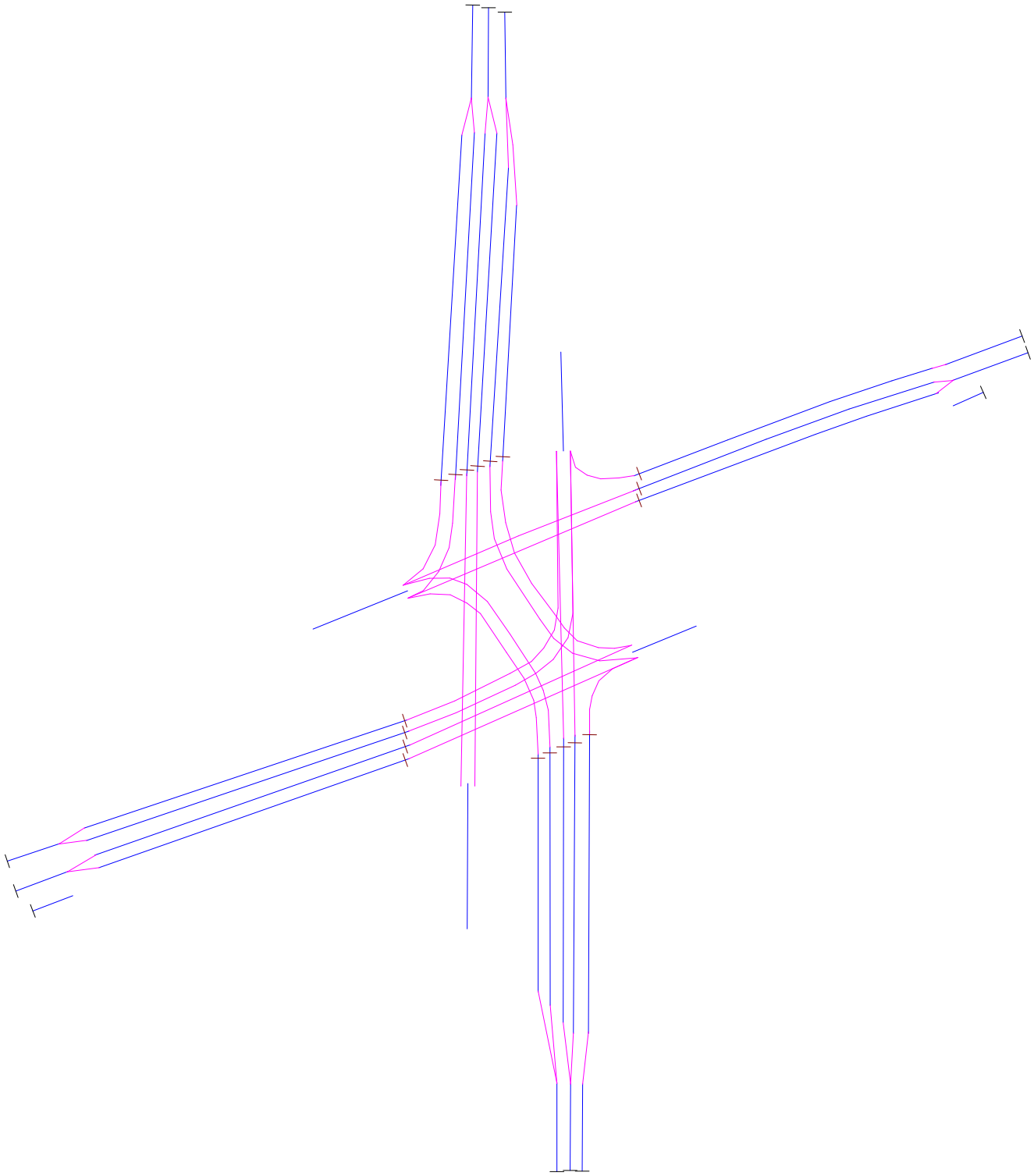


Figure A-4: Implementation of K302 in Vissim

Bibliography

[1] F. Webster, "Traffic signal settings," *Road Research Technical Paper No. 39*, 1958.

[2] B. Keppers and R. Mesotten, *Possibilities for flexible use of network capacity on intersections*. PhD thesis, K.U.Leuven, 2006.

[3] DTV Consultants, Grontmij B.V., and Royal HaskoningDHV, *ASVV 2012 - Aanbevelingen voor verkeersvoorzieningen binnen de bebouwde kom*. CROW, 2012.

[4] W. H. K. Lam, A. C. K. Poon, and G. K. S. Mung, "Integrated model for lane-use and signal-phase designs," *Journal of Transportation Engineering*, vol. 123, pp. 114–122, Mar. 1997.

[5] Z. Tian, T. Urbanik, R. Engelbrecht, and K. Balke, "Pedestrian timing alternatives and impacts on coordinated signal systems under split-phasing operations," *Transportation Research Record*, vol. 1748, pp. 46–54, Jan. 2001.

[6] A. Wilson, *Handboek Verkeerslichtenregelingen*. CROW, 2006.

[7] C. Wong and S. Wong, "Lane-based optimization of signal timings for isolated junctions," *Transportation Research Part B: Methodological*, vol. 37, pp. 63–84, 2003.

[8] C. Wong and S. Wong, "A lane-based optimization method for minimizing delay at isolated signal-controlled junctions," *Journal of Mathematical Modelling and Algorithms*, vol. 2, no. 4, pp. 379–406, 2003.

[9] G. Improta and G. Cantarella, "Control system design for an individual signalized junction," *Transportation Research Part B: Methodological*, vol. 18, pp. 147–167, Apr. 1984.

[10] C. Wong, S. Wong, and C. Tong, "A lane-based optimization method for the multi-period analysis of isolated signal-controlled junctions," *Transportmetrica*, vol. 2, pp. 53–85, Jan. 2006.

[11] C. Wong and B. Heydecker, "Optimal allocation of turns to lanes at an isolated signal-controlled junction," *Transportation Research Part B: Methodological*, vol. 45, pp. 667–681, May 2011.

[12] M. Hausknecht, T.-C. Au, P. Stone, D. Fajardo, and T. Waller, "Dynamic lane reversal in traffic management," in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, vol. 2, pp. 1929–1934, 2011.

[13] L. Zhang and G. Wu, "Dynamic lane grouping at isolated intersections," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2311, pp. 152–166, Dec. 2012.

[14] J. Zhao, W. Ma, H. M. Zhang, and X. Yang, "Two-step optimization model for dynamic lane assignment at isolated signalized intersections," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2355, pp. 39–48, Dec. 2013.

[15] C. Garcia, D. Prett, and M. Morari, "Model predictive control: theory and practice — a survey," *Automatica*, vol. 25, no. 3, 1989.

[16] M. Morari and J. H. Lee, "Model predictive control: past, present and future," *Computers & Chemical Engineering*, vol. 23, pp. 667–682, May 1999.

[17] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control : Stability and optimality," *Automatica*, vol. 36, 2000.

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

[19] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we are going," *Transportation Research Part C: Emerging Technologies*, vol. 43, pp. 3–19, June 2014.

[20] Transportation Research Board, "Highway Capacity Manual," 2000.

[21] R. Akcelik, "Highway Capacity Manual delay formula for signalized intersections.," *ITE Journal (Institute of Transportation Engineers)*, vol. 58, no. 3, pp. 23–27, 1988.

[22] T. H. Muller, T. Dijker, and P. Furth, "Red clearance intervals: theory and practice," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1867, pp. 132–143, Jan. 2004.

[23] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, "Review of road traffic control strategies," *Proceedings of the IEEE*, vol. 91, no. 12, 2003.

[24] T. Achterberg, "SCIP: Solving constraint integer programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, 2009.

[25] A. Salomons, "Optimising cycle times of controlled intersections with VRIGen," in *Colloqium vervoersplanologisch speurwerk*, (Santpoort), 2008.

[26] J. Barcelo, *Fundamentals of traffic simulation*, vol. 145. Springer-Verlag New York, 2010.

[27] F.M. Dekking, C. Kraaikamp, H.P. Lopuhaä, and L.E. Meester, *A modern introduction to probability and statistics*. Springer-Verlag London, 1 ed., 2005.

[28] R. van Katwijk, *Multi-agent look-ahead traffic-adaptive control*. Doctoral thesis, Delft University of Technology, 2008.

[29] S. Lee, S. Wong, and Y. Li, "Real-time estimation of lane-based queue lengths at isolated signalized junctions," *Transportation Research Part C: Emerging Technologies*, vol. 56, pp. 1–17, 2015.

[30] B. De Schutter and B. De Moor, "Optimal traffic light control for a single intersection," *European Journal of Control*, 1998.

[31] K. Dresner and P. Stone, "A multi-agent approach to autonomous intersection management," *Journal of Artificial Intelligence Research*, vol. 31, no. 1, pp. 591–656, 2008.

Glossary

Clearance time

A property of each conflict is its *clearance time*, the minimum time the signals must be red between two consecutive conflicting movements in order to keep them from colliding.

Conflict group

A conflict group is a set of conflicting movements. This includes sets of two and more, as long as each movement in the group is conflicting with all other movements in the group.

Cycle time

Movements are given green in fixed combinations and in a fixed order. One complete succession of this repeating sequence is called a *cycle*. The duration of a cycle is called the *cycle time*. A longer cycle time will have as effect that the lost time will be lower relative to the effective green time. A shorter cycle time will have more lost time, but will decrease the maximum waiting time. An optimal cycle time is a trade-off between reducing lost time and not letting drivers wait for too long.

Leg

A leg is a part of an intersection consisting of approach and exit lanes. Most commonly intersections have three or four legs.

Lost time

The time during which vehicles discharge from the queue with saturation flow is called the *effective green time*. It comprises of the green time, minus the time lost due to start-up, and with addition of the part of the yellow time that is used. When summing the effective green times the time of the cycle that is left is the *lost time*. It contains the total start-up lost time, the total time of yellow that is not used, and the total time lost clearing the intersection.

Movement
The part of traffic on a leg of an intersection that has the same destination leg.

Saturation flow
The amount of vehicles that can pass the stop line per second during green is limited. The size of that limit is captured in the *saturation flow*, which is the average flow achieved by a saturated, stable moving queue of vehicles.

Vertical queue
A vertical queue assumes the queue is building in a single point, such that each vehicle can drive up to the stop line before being added to the queue. Imagine the vehicles are stacked vertically at the stop line. This is of course a simplification but it is widely used because it makes calculations much easier.