Master Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Control Engineering

Modeling and optimization for traffic signal preemption for emergency vehicles using V2X communication

Lukáš Pospíchal

Supervisor: doc. Ing. Zdeněk Hurák, Ph.D.
Field of study: Cybernetics and Robotics
May 2021

Acknowledgements

I wish to thank primarily my supervisor doc. Hurák, for this great opportunity to immerse myself into such an interesting and rich topic. Secondly I would like to thank all my family and friends for their emotional support, and in case of some even financial. Namely I would like to thank in no order of no preference, thus alphabetically, my lovely mother Eva, her solid husband Jiří, my wise father Jiří and my amazing girlfriend Klára, who suffered from my studies the most.

Thank you all.

Declaration

I declare that this work is all my own work and I have cited all sources I have used in the bibliography.
Prague, 21. May 2021

Abstract

With the introduction of modern approaches to traffic data collection come new possibilities for making traffic safer and less congested. It is viable to use the data to make our cities more spacious and productive through clever algorithms.

In large cities, emergency vehicles have a significant impact on traffic. Furthermore, emergency vehicles have the highest rate of accidents, including fatal ones, thus making the inevitable responses of emergency vehicles safer, swifter while having as little impact on the surrounding urban traffic is of the utmost importance.

Traffic signal preemption is a common method to give preference to emergency vehicles at intersections, enabling vehicles that would be otherwise in the way of the emergency vehicle to clear the intersection before it reaches them. Even if proven by time, the methods currently in use are technologically outdated and have many drawbacks. The aim of this thesis is to propose a new, modern approach to traffic signal preemption for emergency vehicles.

This thesis proposes a new time-optimal traffic signal preemption algorithm designed to mitigate the drawbacks of existing preemption techniques using real-time traffic data collection and V2X communication technology. Even though the algorithm has been verified in traffic simulations, the current infrastructure is insufficient for real traffic deployment.

Keywords: traffic signal preemption, microscopic traffic simulation, time-optimal control

Supervisor: doc. Ing. Zdeněk Hurák, Ph.D.
Katedra řídicí techniky FEL ČVUT,
Karlovo náměstí 13/E,
121 35 Praha 2,
Czech Republic

Abstrakt

S příchodem moderních přístupů ke sběru dat z dopravy se otevírají dveře jeho využití ke zvyšování bezpečnosti dopravy a snižování její hustoty. S využitím chytrých algoritmů se nabízí možnost využít data z dopravy k tomu, aby naše města byla více prostorná a produktivní.

Ve velkých městech, mají vozidla ISZ velký vliv na dopravu. Navíc mají najvyšší pravděpodobnost nehod, včetně těch smrtených, a tak zvýšení bezpečnosti a zrychlení jejich výjezů a zároveň omezení jejich vlivu na okolní dopravu je velmi důležité.

Prioritní průjezd vozidel je běžnou metodou preference jistých vozidel na světelných křižovatkách, která umožňuje, aby vozidla stojící na světelné křižovatce ji opustila ještě před tím, než k ní dojede vozidlo IZS. Přestože existující přístupy jsou prověřený časem, jsou také velmi technologicky zastaralé a mají spustu nedostatků. Cílem této práce je návrh moderní metody prioritního průjezdu vozidel IZS na světelných křižovatkách.

Tato práce popisuje návrh časově optimálního algoritmu pro prioritní průjezd vozidel IZS, který potlačuje nedostatky metod existujících za pomoci sběru dopravnách dat v reálném čase a V2X kominikační technologie. Přestože tato metoda byla simulačně otestována, na její nasazení do reálného provozu zatím existující infrastruktura nestačí.

Klíčová slova: prioritní průjezd vozidel, mikroskopická simulace dopravy, časově optimální řízení

Překlad názvu: Modelování a optimalizace pro zajištění prioritního průjezdu vozidel IZS světelně signalizovanou křižovatkou s využitím komunikace V2X

Contents

Figures

Tables

Chapter 1
Introduction

1.1 Motivation

In most urban environments, it is common to witness an *emergency vehicle* (EV), e.g. an ambulance, a police car or a fire engine with its emergency light signal and siren active, rushing to its assigned accident site. It often includes going through a *traffic light controlled junction* (TLJ). The typical behaviour of vehicles in the EV's path is to slow down, approach the side of the road and let the EV pass. Other vehicles give right-of-way to the EV. The EV can break many standard traffic rules. In Czechia and other EU countries, EVs can run red lights, overtake on the right, and overtake on the left, where such manoeuvres are forbidden for regular traffic[1].

While the safety of participants in traffic is of the utmost importance, minimising the response time of EVs also very important. Furthermore, the effect EVs have on traffic is also often investigated since it is not uncommon to have many active EVs within a city, each having the potential to restrict traffic in a given direction temporarily. Thus, the possibility of resulting traffic jams exists.

A common approach in tackling the issues mentioned above is to employ *traffic signal preemption* (TSP). A scheme will give a green signal to any traffic ahead of an EV while restricting any traffic that could collide with an EV or slow it down. Doing so allows the EV a free passage through the TLJ. There are various methods to tackle this problem using various communication, computation, and conceptual strategies[2]. Most of the existing methods will be generalised in this thesis and differentiated by selected characteristics. Moreover, some characteristics (e.g. methods of communication) I will omit entirely.

It should be noted that TSP is not the only strategy commonly used to make EV responses safer and more efficient. There exists another widespread group of methods collectively labelled as *emergency vehicle route optimisation*. It is not the focus of this thesis, however.

1.2 Existing methods

1.2.1 No traffic signal preemption

As stated before, when EVs approach TLJs without any TSP capability, EV drivers often need to make use of their special permissions to run red lights or, in case there are vehicles already halting at the red light by the time the EV approaches the TLJ, they need to overtake through the opposite direction lane. Such behaviour is hazardous, however, as EVs have a high chance of accidents[3], and most of those happen at intersections, specifically, as a result of running red lights[1]. Thus more sophisticated methods are needed.

1.2.2 Distance-based (polygon) traffic signal preemption

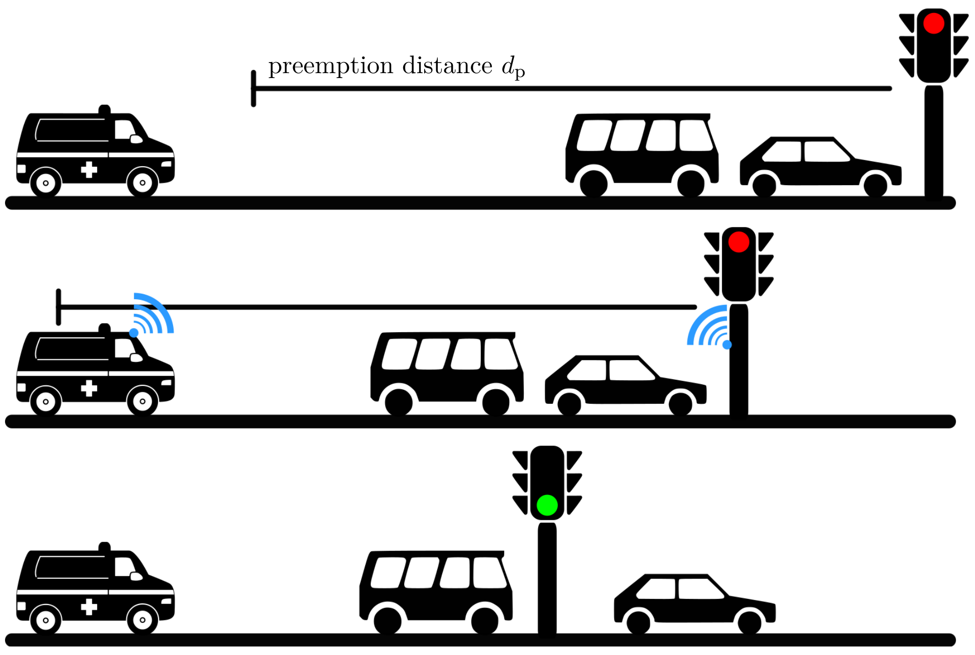


Figure 1.1: Distance-based preemption

The most widespread variation of TSP in Czechia is one, where for each incoming road of a TLJ, there is a distance d_p within which an EV detection triggers a preemption signal sequence in the respective direction of traffic.

This approach is often called a polygon preemption, as the distances (generally a few hundreds of meters) in all incoming directions form a polygon around the intersection. This static polygon is manually designed and optimised for each intersection by an experienced traffic engineer.

In the nomenclature of Humagain et al., this is labeled as a passive defined-area-based green light preemption.

This approach can lead to unnecessarily long green phases for the EV and red phases for other vehicles, leading to the formation of jams. This is the main argument against this approach and, together with the problematic polygon design, is the main reason for its limited use.

1.2.3 Adaptive polygon traffic signal preemption

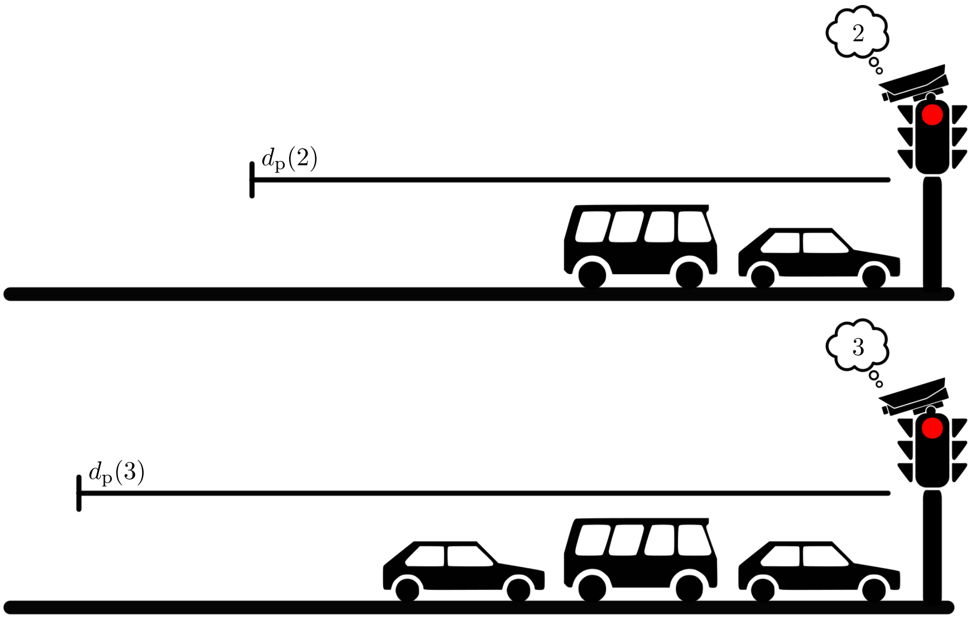


Figure 1.2: Adaptive polygon preemption

Another more advanced approach aims to mitigate the disadvantages of the distance-based method by changing the polygon shape using real-world traffic information. Such data can take the form of historical data aggregate, induction loop feedback, traffic density estimates, etc. For more details on the individual types of active TSP, see the summary by Humagain et al.[2].

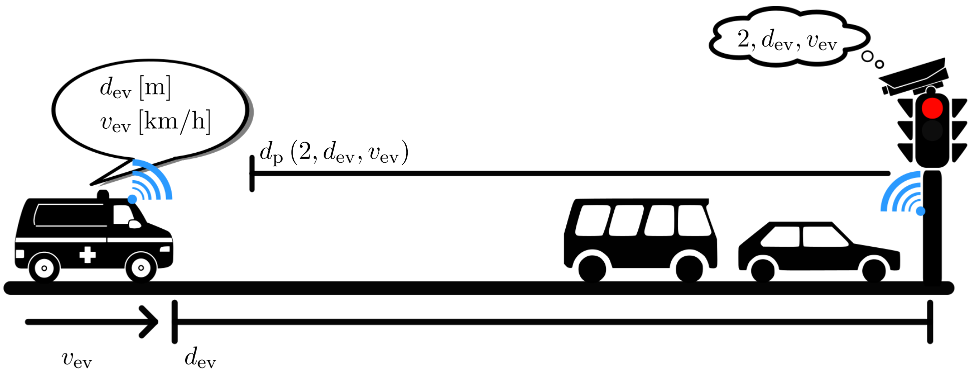


Figure 1.3: Queue discharge preemption

An especially relevant method, queue discharge-based traffic signal preemption, has been developed by Obrusník et al.[4]. It uses the estimate of the number of vehicles halting due to red light at the TLJ ahead of the EV as well as information on position and velocity of the EV given to the signal plan controller via *vehicle to infrastructure* (V2I) communication device to alter the size of the polygon in the direction of the EV, see Figure 1.3.

This particular method is significantly relevant to this thesis due to its development in partnership with a Czech manufacturer of the necessary infrastructure[1], aiming for simple implementation on existing hardware and subsequent testing in real traffic.

1.3 Objectives

This thesis introduces a new simulation-driven method of time-optimal traffic signal preemption for emergency vehicle prioritisation at TLJs. This method takes advantage of integrating modern technologies in traffic data collection, which are currently being introduced.

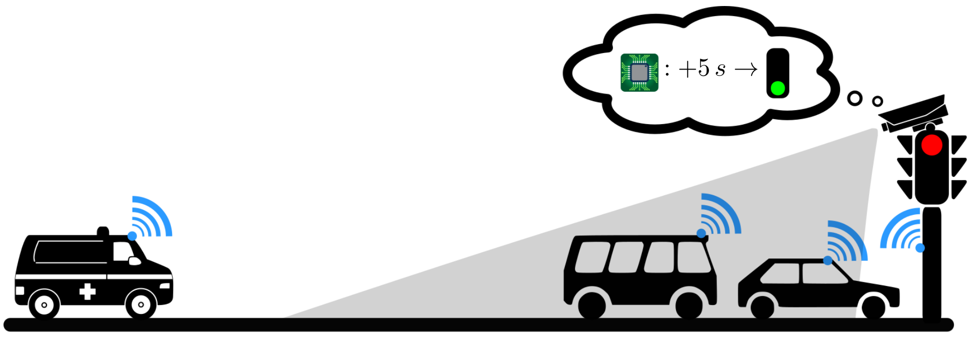


Figure 1.4: Time-optimal traffic signal preemption

The method, visualised in Figure 1.4, comes in the form of an algorithm or procedure, which can compute the optimal time to start a traffic signal preemption sequence when an EV approaches the intersection using real-time data inputs and traffic simulations. Thanks to its design, it minimises the impact EV has on surrounding traffic while still retaining the desired properties of standard methods, namely minimal delay of the EV and its maximum safety.

In order to ensure the best results possible, an appropriate car-following model needs to be selected. The model is necessary to simulate the passage of EVs forward in time accurately. This selection is made through the use of model-fitting on data from real-world traffic.

An essential part of the algorithm is the cost function used to discriminate among possible preemption times. The selection of cost functions in traffic engineering is generally rarely an exact science, and thus, it is crucial to explain the reasoning behind the cost function used in the algorithm.

Lastly, the proposed method will be validated in SUMO, a popular traffic simulator where it will be compared to typical methods of preemption. Additionally, the interpretation of the results will take place, followed by a summary and overall conclusions.

[1]`https://www.herman.cz/`

1.4 Outline

This thesis divides into chapters based on their respective focus.

Chapter 1 includes the introduction to the problem, a brief overview of existing methods, most notably the method by Obrusník et al., which is the precursor to this work. Then, the thesis states the objectives of this work and gives an outline of this thesis.

Chapter 2 investigates how accurate car-following models can be in simulating traffic at intersections. The best model considered is selected for use throughout the rest of the thesis.

Chapter 3 follows the conception of a time-optimal traffic light preemption algorithm using real-time traffic data. It discusses its viability and offers possible solutions to some of its insufficiencies.

Chapter 4 describes the validation of the proposed algorithm in SUMO, a traffic simulator, and discusses the results.

Chapter 5 offers a summary of this thesis, includes interpretation of overall results and offers suggestions for future research.

Chapter 2

Car-following models

A car-following model is a system of differential equations describing the behaviour of a vehicle in one-lane traffic. The engineering community has developed various models, and their main idea is to mimic a driver following another vehicle. The driver usually wants to go as fast as possible while not taking unnecessary risks and following a desired distance from the leading vehicle.

Car-following models are generally feedback-controlled systems. Where a human driver would observe their surroundings and, based upon those observations, would either press the gas paddle or the brake paddle and, therefore, change the acceleration of the vehicle accordingly, in a car-following model, this role falls upon the acceleration function, whose inputs are state variables and various external inputs.

There exists a large number of car-following models with varying degree of complexity. The more complex a model is, generally, the more aspects of driving behaviour it can model. There always exists a trade-off between accuracy and complexity when it comes to car-following models. Therefore, we would like to use a model using enough parameters to be accurate but not too many to be too challenging to fit measured trajectories of actual vehicles.

2.1 Car-following models at intersections

Out of the commonly used and acceptably simple models for traffic simulations, i.e. Krauss[5], IDM[6] and Wiedemann[7], IDM has already been determined to be the most accurate for simulating vehicles passing an intersection without stopping at a red light[8].

Thus, we need to ask, can it also be used to simulate queue discharge effectively? This efficiency is, unfortunately, entirely subjective and depends on a specific goal. For this reason, the model is evaluated in terms of the estimated time it takes the last vehicle in queue to reach a pre-defined speed threshold.

Let us assume an EV approaches a TLJ, where several vehicles halt due to a red light. The least disruptive scheme for a preemptive algorithm would be to start the preemption signal at such a time that the EV smoothly joins the departing vehicles, passes the intersection, and then the regular traffic plan resumes. There are several caveats to this approach. We do not want to initialise the preemption too soon since it would lead to an unnecessary traffic halt in intersecting directions. We also do not wish to start the queue discharge too late, as that would delay the EV.

For these reasons, the most critical metric for evaluating the performance of a car-following model, as it relates to queue discharge simulation, is the error in time the last vehicle in the queue reaches the desired speed. If this error is sufficiently small, then we can optimise over the preemption initialisation time.

2.1.1 Intelligent Driver Model (IDM)

IDM is one of the most straightforward collision-free time-continuous models. Nevertheless, its intuitive design and acceptable accuracy have made it very a popular model for traffic simulations. Its best properties include smooth transitions between driving modes and easily understandable physical interpretation of its parameters. Its acceleration function is

$$\dot{v} = a\left[1 - \left(\frac{v}{v_0}\right)^{\delta} - \left(\frac{s^*\left(v, \Delta v\right)}{s}\right)^2\right], \tag{2.1}$$

with desired distance

$$s^*\left(v, \Delta v\right) = s_0 + \max\left(0, vT + \frac{v\Delta v}{2\sqrt{ab}}\right) \tag{2.2}$$

and desired speed v_0, bumper-to-bumper distance to the leading vehicle s_0, speed relative to the leading vehicle Δv, time headway T, comfortable acceleration and deceleration a and b respectively. There is also an acceleration exponent δ. It is, however, usually set to $\delta = 4$. See *Traffic Flow Dynamics*[6] for more detailed information on IDM.

Please note that there is a strong case for restricting the backwards motion of vehicles for any car-following model used for queue discharge simulation. If IDM parameters are imprecise so that during the first iteration of the simulation $s^*/s > 1$ and $v \approx 0$, the vehicle will start reversing, shortening the bumper-to-bumper distance behind it and eventually causing all vehicles behind it to reverse as well. This behaviour is far from desired, and thus one can disable backwards motion in the model using

$$\frac{s^*}{s} > 1 \wedge v \leq 0 \implies v := 0,\ \dot{v} := \max\left(0, \dot{v}\right) \tag{2.3}$$

after each step of the simulation. Additionally, restriction of reverse motion has real-world justification as well. If a human driver stops unusually close to a leading vehicle while stopping at a red light, they do not usually start reversing to correct the spacing. They wait for the leading vehicle to accelerate, and once the leading vehicle is sufficiently far away, they start to accelerate as well.

Lastly, as seen in Figure 2.1, vehicles simulated using IDM never reach the speed of their leading vehicles, always cautiously moving slightly slower than

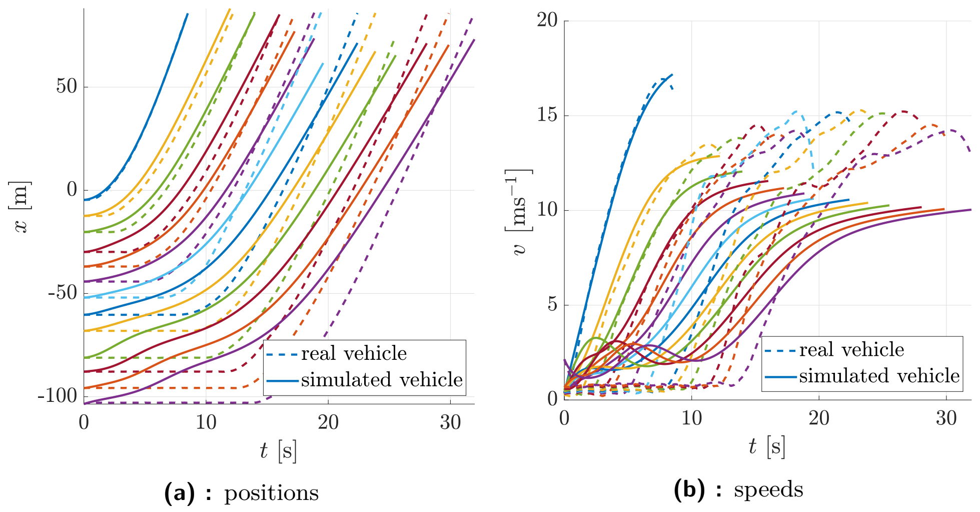


Figure 2.1: Queue discharge using IDM

their leaders. Luckily, there are a plethora of extensions to IDM to make it more accurate.

2.1.2 Improved Intelligent Driver Model (IIDM)

IIDM is given in *Traffic Flow Dynamics*[6] as an example of methods used to improve the accuracy of car-following models without adding new parameters. Thankfully, IIDM enables all vehicles in a queue to reach saturation speed. Which is, of course, achieved by altering the acceleration function. The improved acceleration function is defined as

$$\dot{v} = \left\{ \begin{array}{ll} v \le v_0 : & \left\{ \begin{array}{ll} z \ge 1 : & a\left(1 - z^2\right) \\ z < 1 : & a_{\text{free}}\left(1 - z^{\frac{2a}{a_{\text{free}}}}\right) \end{array} \right\| \\ v > v_0 : & \left\{ \begin{array}{ll} z \ge 1 : & a_{\text{free}} + a\left(1 - z^2\right) \\ z < 1 : & a_{\text{free}} \end{array} \right\| \end{array} \right\| , \tag{2.4}$$

where

$$a_{\text{free}} = \left\{ \begin{array}{ll} v \le v_0 : & a\left[1 - \left(\frac{v}{v_0}\right)^{\delta}\right] \\ v > v_0 : & -b\left[1 - \left(\frac{v_0}{v}\right)^{\frac{a\delta}{b}}\right] \end{array} \right\| , \text{ and } z = \frac{s^*}{s}. \tag{2.5}$$

The parameters $v_0, s_0, T, a, b, s, \delta$ and the desired distance s^* are defined identically and serve the same purpose as in the case of IDM. Again, more details can be gathered in *Traffic Flow Dynamics*.

Please note that in case $v = v_0$ and $z < 1$, the acceleration function is undefined. While in the real world, this condition is an unachievable edge case, for the purposes of numerical simulation, it is necessary to define a special case

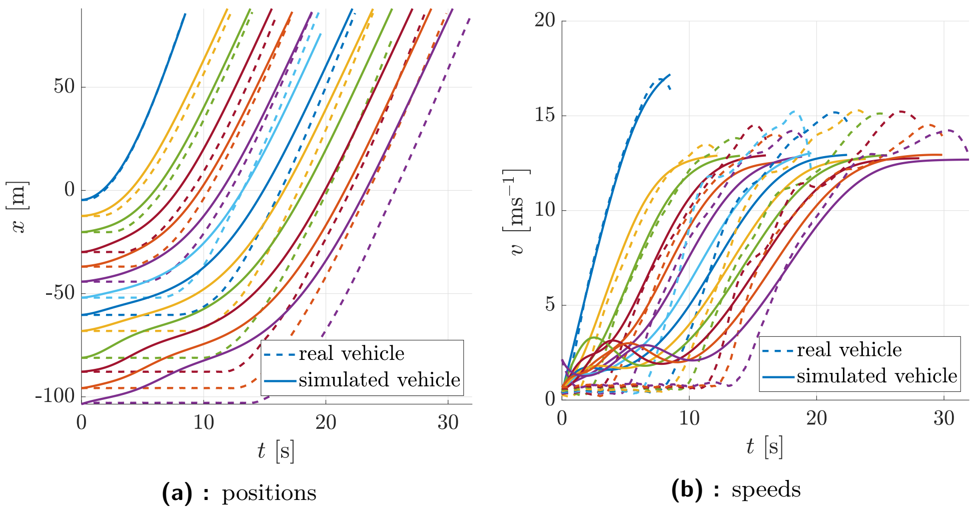
$$\dot{v} = \lim_{a_{\text{free}} \to 0^+} a_{\text{free}} \left(1 - z^{\frac{2a}{a_{\text{free}}}}\right) = 0, \quad v = v_0 \text{ and } z < 1, \tag{2.6}$$

for the model, especially given the fact that computers evaluate expressions inside-out. Without this addition to the definition, any vehicle further from the leading vehicle than s_0, which reaches its desired speed v_0, would crash the simulation. Even with this addition, it is still possible to achieve floating-point errors since

$$\lim_{a_{\text{free}} \to 0^+} z^{\frac{2a}{a_{\text{free}}}} = \infty, \quad v \leq v_0 \text{ and } z < 1. \tag{2.7}$$

Thus special care should be taken to handle these special cases.

Additionally, arguments and solutions for restricting backward motion are identical to those of IDM.



(a) : positions **(b) :** speeds

Figure 2.2: Queue discharge using IIDM

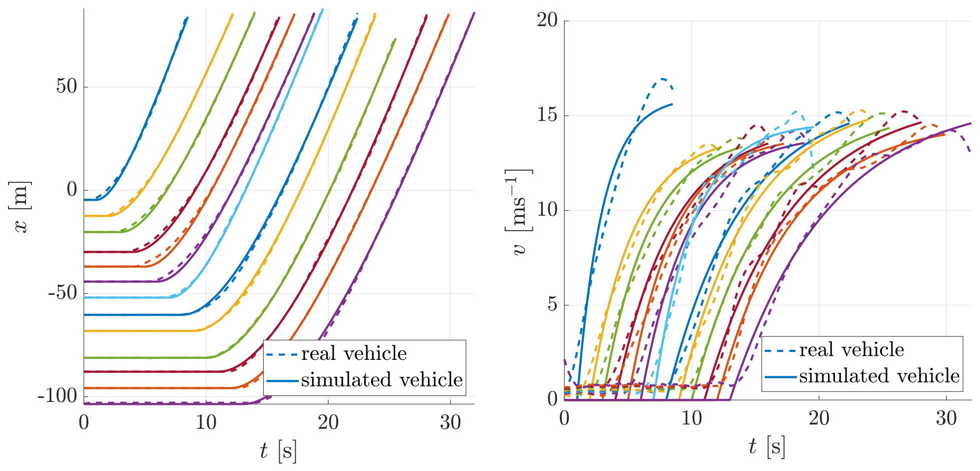
As seen in Figure 2.2, vehicles simulated using IIDM can match the speed of their leading vehicles. Furthermore, the error in the velocity profile of the last vehicle in the queue is smallerr.

2.1.3 Linear Queue Discharge Model (LQDM)

While IIDM looks very promising, it is worth investigating whether a much simpler model could be used to simulate the queue discharge since IIDM is quite difficult to simulate numerically correctly. Its acceleration function is

$$\dot{v} = \begin{cases} t < t_{\text{start}} + T_{\text{delay}} : 0 \\ t \geq t_{\text{start}} + T_{\text{delay}} : a\left(1 - \dfrac{v}{v_0}\right) \end{cases} \Bigg|, \tag{2.8}$$

where t_{start} is the time a leading vehicle starts accelerating, T_{delay} is a time a vehicle waits, after its leader starts accelerating, before starting accelerating itself, a is maximum acceleration and v_0 is desired speed.



(a) : positions

(b) : speeds

Figure 2.3: Queue discharge using LQDM

While this model does not meet the definition of a car-following model, it can be pretty useful to simulate queue discharge after switching to green light, as seen in Figure 2.3. It is, however, completely useless for other traffic scenarios at intersections since it requires all vehicles at $t = 0$ to be almost stationary. It is of little usage for simulations, where vehicles have non-zero initial velocities. Also, estimating the values of the parameters ahead of the queue discharge for individual vehicles might be very challenging.

2.2 Parameter fitting

2.2.1 Sequential fitting

All parameters of the models simulated in 2.1 have been fitted using `lsqcurvefit` function in MATLAB[9]. Parameters $\mathbf{p}_{\text{model}}$ of all vehicles in a given queue have been fitted sequentially, i.e. first vehicle was fitted first and then the second and so on. The function was used to minimise *mean square error*(MSE) of simulated positions and speeds of a given vehicle.

$$\mathbf{p}^*_{\text{model}} = \operatorname{argmin} \text{MSE}\left(\begin{bmatrix}\mathbf{x}_{\text{sim}}\left(\mathbf{p}_{\text{model}}\right)\\ \mathbf{v}_{\text{sim}}\left(\mathbf{p}_{\text{model}}\right)\end{bmatrix}, \begin{bmatrix}\mathbf{x}_{\text{real}}\\ \mathbf{v}_{\text{real}}\end{bmatrix}\right), \tag{2.9}$$

where * denotes optimality.

While IDM and IIDM had better performance when all vehicles were fitted simultaneously, it is more realistic to assume that the parameters of a leading vehicle cannot be affected by the trajectory of the following vehicle. The goal of fitting the parameters this way was to show how accurate these models can be if all vehicles pass perfect estimates of their parameters to the simulation before the queue discharge, i.e., how accurate the models can be given perfect conditions.

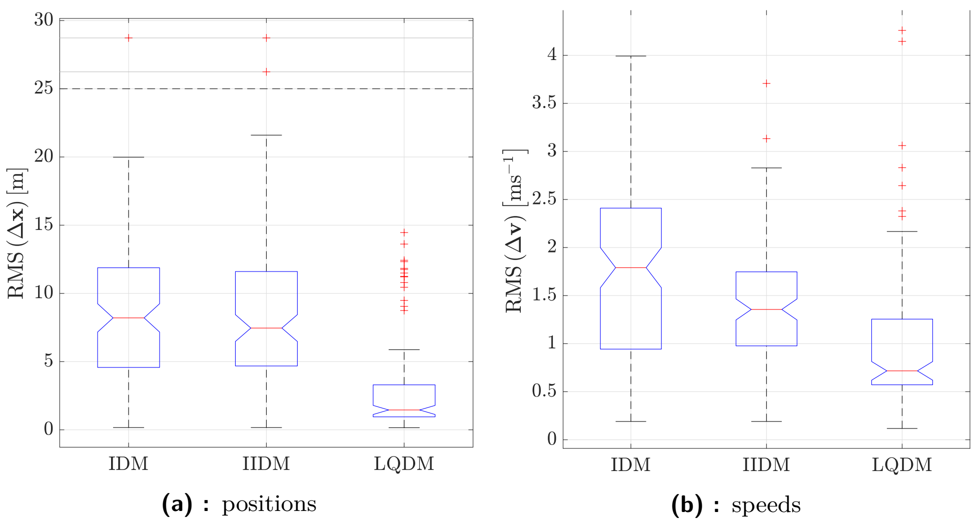


Figure 2.4: Boxchart of root mean square error of simulated vehicles

When using this parameter fitting method, optimal parameters were found for each model for 112 vehicles in 35 different queues from different lanes of two real-world TLJs. As seen in Figure 2.4, the smallest median fitting *root mean square*(RMS) error for both positions and velocities has been achieved by LQDM. Furthermore, IIDM has been found to generate more accurate velocity profiles than IDM, which is the main idea behind IIDM.

As explained in 2, the most important metric is the velocity profile of the last vehicle in a queue. The boxchart of RMS errors of velocity profiles of the last vehicles in their respective queues is shown in Figure 2.5. The error in time it takes a vehicle to reach saturation speed was not evaluated, as stated in 2, since it is not guaranteed that a real vehicle reaches the saturation speed, and thus, such metric would be meaningless. It is, however, strongly related to the accuracy of the velocity profile. While it is not guaranteed that minimising one minimises the other, minimising RMS error of the velocity profiles keeps the time difference of reaching saturation speed sufficiently small.

The LQDM fares best when it comes to velocity profiles of the last vehicles of queues, IIDM shows better performance over IDM, in this regard.

2.2.2 Parameter averaging

While it is important to see how well a model can be used for a queue discharge simulation, it can be argued that it is currently infeasible to expect vehicles to pass their model parameters estimates to the TLJ controller using *vehicle to everything* V2X communication. Therefore, it makes sense to ask, how the models fare if model parameters of vehicles are estimated by the infrastructure. A possible scenario for such a system would use a computer vision system installed at the TLJ to gather data for individual lanes and vehicle types, and from those, it could assign an estimate of model parameters

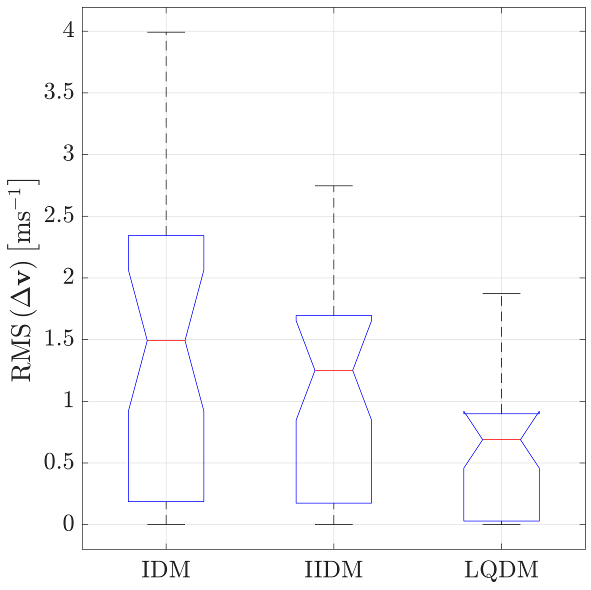
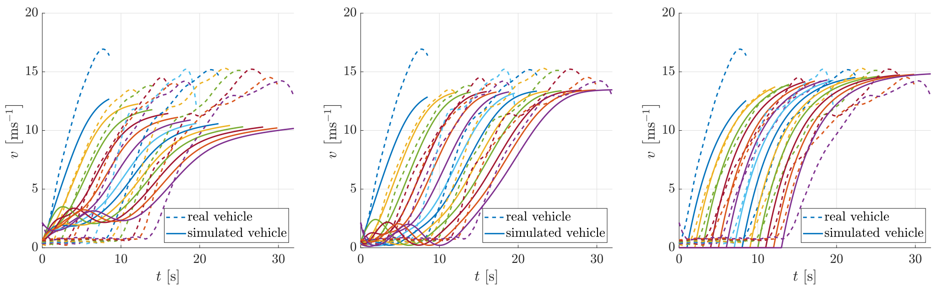


Figure 2.5: Boxchart of root mean square error of velocities of last vehicles in their respective queues.

as soon as a vehicle enters the detection range.

Given the fact that the available dataset is too limited to consider vehicle types, the same evaluation from 2.2.1 was repeated, only this time the parameters of all vehicles passing the intersection using the same lane were averaged, thus emulating an estimate from historical data. Velocity profiles of queue discharge simulations of the same scenario as in 2, now using averaged parameters, can be seen in Figure 2.6.



(a) : IDM **(b) :** IIDM **(c) :** LQDM

Figure 2.6: Velocity profiles of a queue with averaged parameters

It is clear that the velocity profiles in Figure 2.6 perform worse, when the parameters are averaged, then using the optimal parameters for every vehicle. However, the fit is skewed due to the fact, that my dataset does not include information about the length of individual vehicles, which is the only absolutely static car-following model parameter, which is not dependent on the driver and would be easily accessible in the aforementioned hypothetical scenario with computer vision systems estimating the parameters.

In the previous case, it did not matter that I did not have access to vehicle

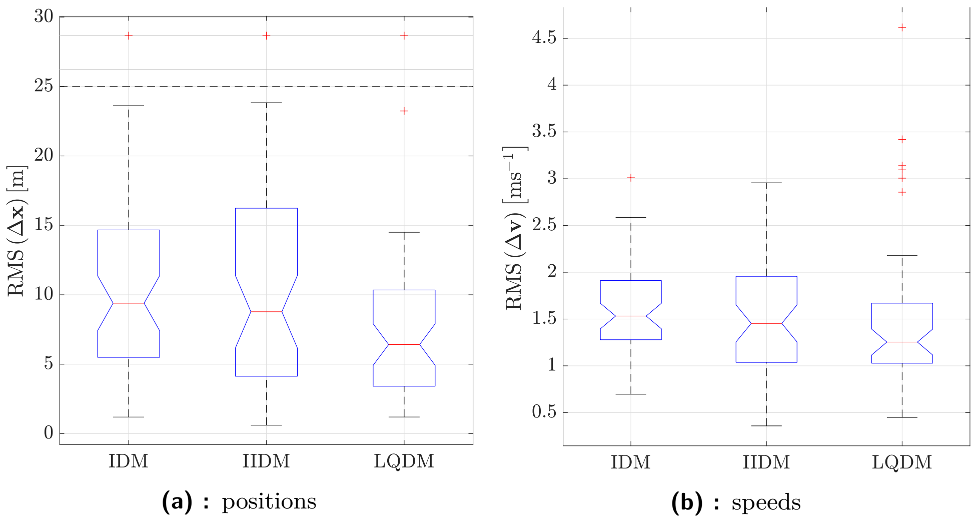


Figure 2.7: Boxchart of root mean square error of simulated vehicles with averaged parameters

lengths, since the $l + s_0 = const$ for any optimal parameter fit, where l is the length of the leading vehicle and s_0 is the desired bumper-to-bumper distance of the leading vehicle. However, with suboptimal parameters a new issue arises. If a vehicle stops too close behind a too short a vehicle, it may lead to the computed bumper-to-bumper distance $s < 0$. Which can be interpreted as a collision and breaks the simulation.

To eliminate the possibility of having vehicles virtually colliding in the first step of the simulation, I adjusted $l = \min(l, x_{\text{leader}} - x - s_0)$. This is a completely naive solution to the problem. Surely, more nuanced solution can be found. I do not believe it is necessary, however, since the unavailability of vehicle lengths is not a very realistic condition, given the assumption of having a modern position and velocity detection device at the TLJ or V2X communication capability.

Given the aforementioned limitations the absolute values shown in Figure 2.7, need to be read with scepticism. On the other hand, the relative performance of the investigated methods did not change, only the median errors of velocities of IDM and LQDM have moved closer to the error of IIDM, whose median RMS error has increased very little.

Curiously enough, as shown in Figure 2.8, the RMS erros of the last vehicles in their respective queues have decreased for all the models investigated. This result, again, has to be taken with a grain of salt, because I suspect the result is skewed by the lack of vehicle length information.

2.3 Final thoughts on car-following model selection

Given the simulations I ran using IDM, I do not find it to be particularly appealing for usage in queue discharge simulations. It does not simulate sufficiently accurate velocity profiles for the vehicles at the end of a queue.

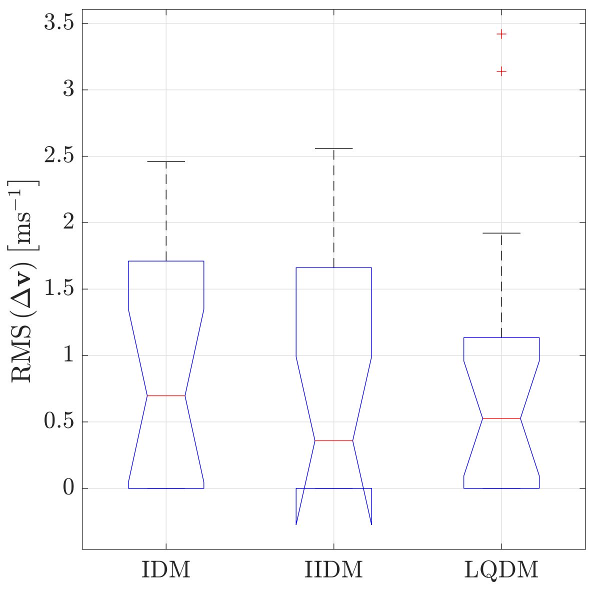


Figure 2.8: Boxchart of root mean square error of velocities of last vehicles in their respective queues

And since our main goal is to eventually reach the end of a queue at a time when all the cars have reached sufficiently high velocities, IDM should not be used. IIDM, on the other hand does produce sufficiently accurate results and unlike LQDM, whose results were in many cases even better, it can be used to simulate all one lane traffic before a TLJ, whereas LQDM can only be used to simulate the queue discharge with no initial velocities.

For these reasons I will use IIDM in the rest of the thesis for simulations of queue discharge and preemption.

Finally, I want to point out how surprisingly simple and accurate LQDM turned out to be. Even though, it might not be perfect for the use case of this thesis, it was by far the most accurate in simulating the positions of vehicles in any scenario I simulated.

Chapter 3

Time optimal traffic signal preemption algorithm

While queue discharge-based traffic signal preemption has been shown to be effective, as stated by Obrusník et al.[4] in their conclusions, it would be preferable to formulate the problem of finding the optimal preemption time using clearly defined optimisation techniques. This is the focus of this chapter.

3.1 Assumptions

Let us start with assuming unrealistically ideal conditions, specifically pertaining to the data available to the traffic signal controller. Later, we will discuss how various limitations of these assumptions can be tackled in terms of formulation of the algorithm.

Therefore, let us assume an emergency vehicle sends a sign-in signal (This can be thought of as a signal telling the controller to start the preemption time computation) to the traffic light controller at time $t = 0\,\mathrm{s}$. We can simulate its approach to, and crossing of the intersection given these conditions:

1. Knowledge of the velocities and positions of vehicles in the platoon ahead of the EV and the EV itself.
2. Knowledge of the model parameters for all such vehicles and the EV for a selected car-following model.
3. Knowledge of the default traffic light signal plan for the duration of approach and passage of the emergency vehicle.
4. Knowledge of the constraints of the traffic light signal plan (minimum signal lengths etc.).

3.2 Simulation

Using the assumptions in 3.1 and a correct selection of a car-following model, we can simulate the approach of the EV to the end of the vehicle platoon ahead. Figure 3.1 shows such simulation for the case of late preemption. We can see how the EV stops at a red light, while this is very unrealistic, since the EV would most surely overtake the vehicle ahead and cross the intersection through a red light in the real world, we do not need to model such behaviour, because both of these behaviours are undesirable from the point of view of our problem. Since the preemtion eliminates the necessity to run red lights by the EV, by disallowing overtaking and penalizing waiting, our algorithm will deem late preemption times very costly.

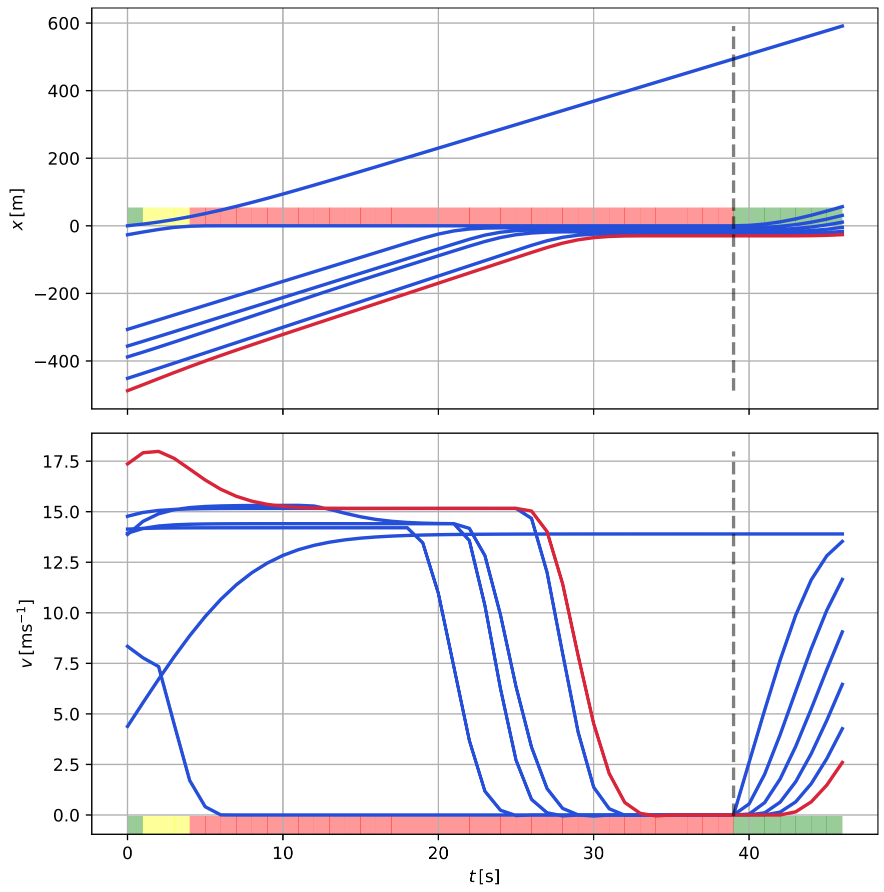


Figure 3.1: Simulation of an approach of an emergency vehicle using IIDM with a signal plan with late preemption.

On the other hand, early preemption times, example of which is shown in Figure 3.2, should also be avoided. In the example simulation, we can see that the preemption had been initiated more than 30s before the EV entered the intersection. Surely, some of this time would have been better used, if traffic in intersecting directions was allowed and the preemption started when the EV was much closer to the intersection.

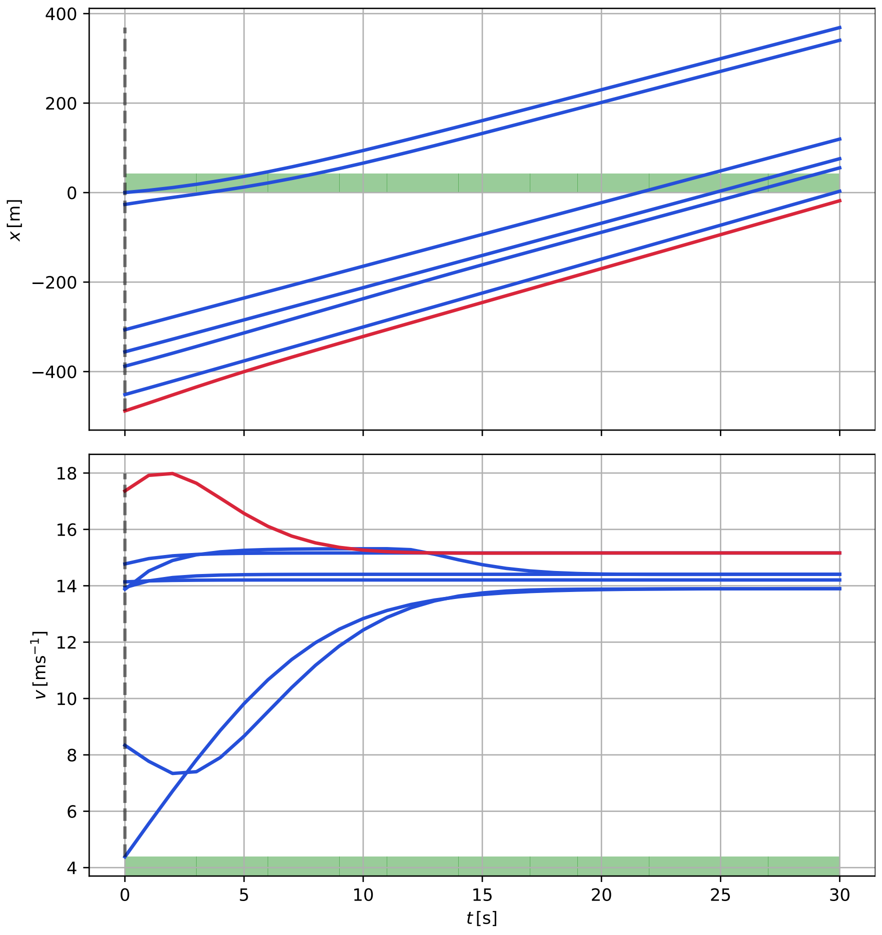


Figure 3.2: Simulation of an approach of an emergency vehicle using IIDM with a preemption time $t_p = 0$ (dotted line).

3.3 Cost function

Understandably, both of the two extreme cases for preemption times in 3.2 are undesirable, and the optimal time needs to exist somewhere in between the two extremes. Another observation, however, can help with the initial idea behind a way to design a preemption time cost function.

Suppose, the requested preemption time $t_\text{p} = 0$, this will result in a traffic light signal plan where the EV receives a green signal as soon as condition 4 allows it. Given this, no other traffic light signal plan can cause less deceleration. Let us define

$$V^*_\text{min} = V^0_\text{min} = \min\left(\mathbf{v}_\text{emergency}(t, t_\text{p} = 0)\right), \tag{3.1}$$

which is the highest achievable minimum velocity of the emergency vehicle for the duration of the simulation, i.e. $\nexists t_\text{p} \geq 0 : V^{t_\text{p}}_\text{min} > V^*_\text{min}$, and

$$t^\text{max}_\text{p} = \frac{v_\text{des,emergency}}{d_\text{0,emergency}}, ^1 \tag{3.2}$$

$^1 v_\text{des,emergency}$ denotes the desired speed of the EV and $d_\text{0,emergency}$ its initial distance the upper limit of the interval upon which it is useful to search for

$$t_{\mathrm{p}}^{*} = \max t_{\mathrm{p}} : V_{\min}^{t_{\mathrm{p}}} = V_{\min}^{*},\, t_{\mathrm{p}} \in \left[0, t_{\mathrm{p}}^{\max}\right], \tag{3.3}$$

the latest preemption request signal time, which will not delay the emergency vehicle compared to immediate preemption at earliest possible time.

Finally, let us define cost function

$$C(t_{\mathrm{p}}) = C_1 + C_2 = (1-\gamma)\left(1 - \frac{t_{\mathrm{p}}}{t_{\mathrm{p}}^{\max}}\right) + \gamma\left(1 - \frac{V_{\min}^{t_{\mathrm{p}}}}{V_{\min}^{*}}\right),\; 0 < \gamma \ll 1, \tag{3.4}$$

where the parameter γ introduces a small slope to the regions where C_2 is constant, this enables reliable localization of t_{p}^{*}, without the effect of floating point errors in the evaluation of C_2. We can see an example of a cost function $C(t)$ in Figure 3.3.

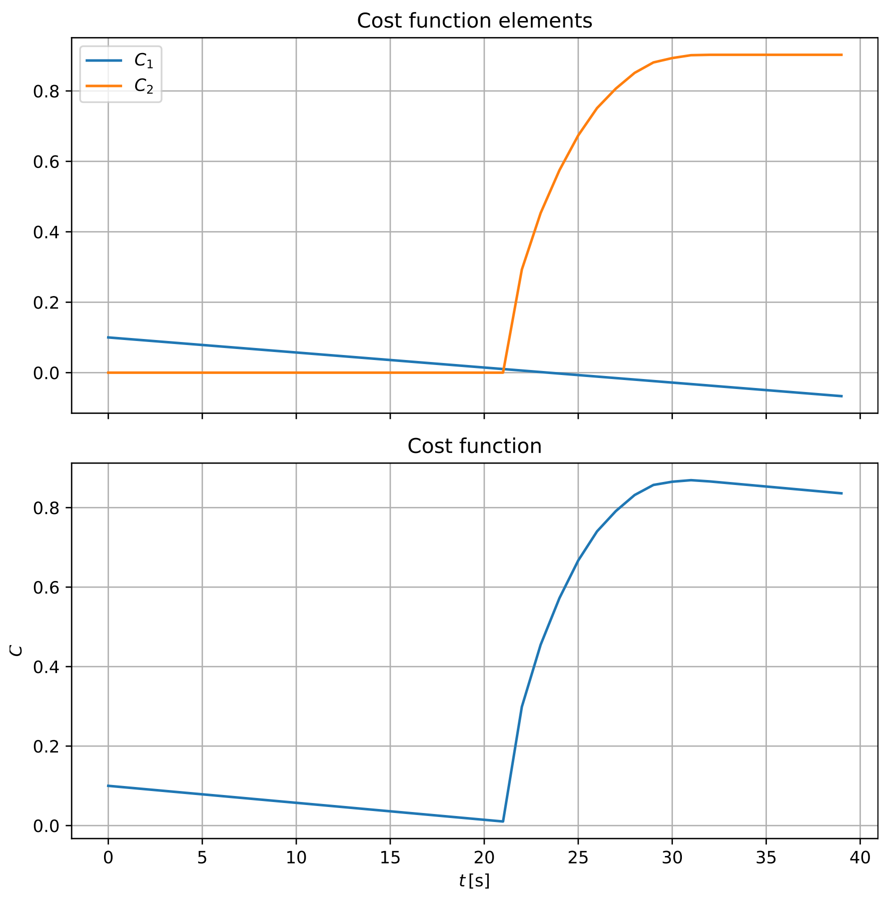


Figure 3.3: Cost function computed for the scenario in Figure 3.1.

to the intersection in the context of the simulation

3.4 Time optimal preemption algorithm

Let us define the algorithm for the optimal preemption time t_p^* computation. Let us assume it has access to all the necessary resources available. The algorithm used to find t_p^* can be seen in Algorithm 1.

Algorithm 1: Optimal preemption time t_p^*

```
$P(t_\mathrm{p}) \leftarrow$ signal plan with preemption in effect at time $t = t_\mathrm{p}$
$v(P(t_\mathrm{p})) \leftarrow$ simulated velocity profile of the EV if preemption
  sequence starts at $t_\mathrm{p}$
$v_\mathrm{des} \leftarrow$ desired speed of the EV
$d_0 \leftarrow$ initial distance of the EV
$t_\mathrm{left} \leftarrow$ initial time
$t_\mathrm{right} = t_\mathrm{p}^\mathrm{max} = t_\mathrm{left} + \frac{v_\mathrm{des}}{d_0}$ max preemption time
$\epsilon_\mathrm{min} \leftarrow$ termination difference
$V_\mathrm{min}^* = \min(v(P(0))$
$C(t_\mathrm{p}) = C(t_\mathrm{p}), v(P(t_\mathrm{p})), t_\mathrm{p}^\mathrm{max}, V_\mathrm{min}^*) \leftarrow$ preemption time cost function
while $|t_\mathrm{right} - t_\mathrm{left}| \geq \epsilon_\mathrm{min}$ do
    if $C(t_\mathrm{left}) < C(t_\mathrm{right})$ then
        $t_\mathrm{right} = \frac{t_\mathrm{right} + t_\mathrm{left}}{2}$
    else
        $t_\mathrm{step} = \frac{t_\mathrm{right} - t_\mathrm{left}}{2}$
        $t_\mathrm{left} = t_\mathrm{right}$
        $t_\mathrm{right} = \max\left(t_\mathrm{right} + t_\mathrm{step}, t_\mathrm{p}^\mathrm{max}\right)$
Result: $t_\mathrm{p}^* = t_\mathrm{left}$
```

The optimal preemption time t_p^* is then given to the traffic light signal plan controller to apply the signal plan $P(t_\mathrm{p})$. A simulation of an approach of an EV using the optimal preemption time can be seen in Figure **??**.

3.4.1 Notes on the algorithm output

It could be argued, it might be mode convenient to interpret the output in terms of preemption distance. That is, reshaping the polygon within which an EV would trigger the preemption. This, however, leads to the risk of rendering the preemption impossible to trigger at the right time. If the EV enters this computed region too late, it might be too late to trigger preemption in time. Thus it is better to trigger the preemption based upon time duration since EV sign-in. That is the advantage of having access to the default signal plan and its limitations. We can incorporate them into the simulation.

It should be noted that the algorithm itself does not alter the signal plan, it is only selecting one of the possible signal plans, the signal plan controller feels comfortable with. This makes the method as a whole less intrusive to the overall signal plan design. Where traffic engineers might feel uncomfortable allowing a new algorithm to control the signal plans of an intersection, allowing

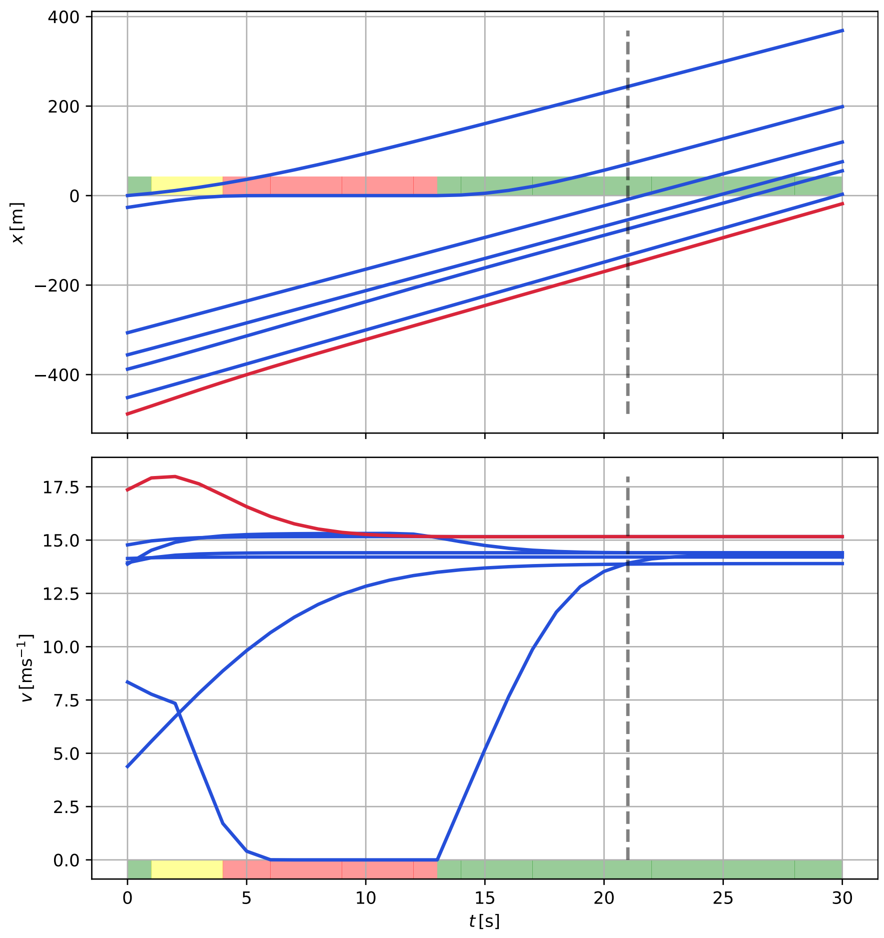


Figure 3.4: Simulation of an approach of an emergency vehicle using IIDM with a preemption time t_p^*.

an algorithm to choose from a set of signal plans with predefined constraints should be well within their comfort zone.

3.4.2 Real-world adjustments

Let us discuss the adjustments we can make to the algorithm in order for it to be applicable to real-world traffic. Specifically, we need to address the unrealistically ideal assumption upon which the algorithm stands. It is not my intention to invent new traffic infrastructure technology. I aim to give examples of possible approaches, which would make the method introducible to the real world traffic.

Firstly, the assumption 1 might seem too unrealistic at first. However, there exist commercial solutions for traffic data collection using computer vision (example being DataFromSky[2]) and there is also the future potential of using V2X communication, for the purposes of each vehicle reporting the data to the infrastructure.

[2]url: `https://datafromsky.com/`

Secondly, same concepts could be used to satisfy assumption 2. However to find car-following model parameters from the limited time a vehicle spends in the range of the AI-powered detectors might be infeasible, thus I suggest using estimation over vehicle types through fitting to aggregated historical data. In the context of the V2X data collection route, each vehicle could be equipped with its own parameter estimator and reporting its output to the infrastructure.

Furthermore, assumptions 3 and 4 do not need any new technology introduced to be met.

In case of the use of computer vision in data gathering, we need to address the possibility of the detectors not having the range required to detect all vehicles in front of the EV at the time of sign-in. A naive solution would be to start the computation of the preemption after the EV enters the visual range of the detectors. Alternatively, having good estimates on the current traffic flow density of all the vehicle types would allow filling the area between the EV and the end of the visual range of the detectors with “virtual” vehicles and only then starting the algorithm.

Finally, there is no reason not to use the algorithm repeatedly (e.g. every second) until t_{p}^* with the intention to get more accurate result with each evaluation. This approach is more computationally intensive but the nature of the algorithm allows for strong parallelisation. Additionally, the fact that the method as a whole is very technologically demanding, making use of other computationally intensive technologies, makes the case for there being sufficient computation power available. For the purposes of this thesis I call this approach the *recomputation* method.

Chapter 4

Validation in SUMO

4.1 Scenario

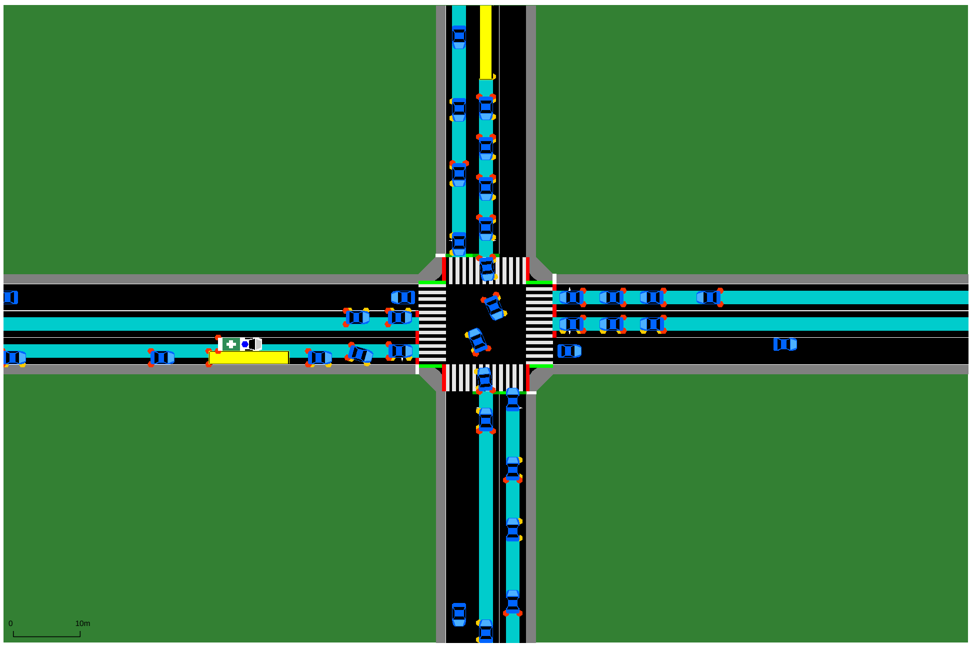


Figure 4.1: Intersection scenario simulated in SUMO.

For the purposes of validating the preemption method proposed in Chapter 3, a scenario was created using the traffic simulator SUMO[10]. A snapshot taken from its graphical user interface running the scenario can be seen in Figure 4.1. In the snapshot, we can see blue passenger vehicles, yellow buses and an emergency vehicle approaching the junction from the west. Take note of the fact, that vehicles in front of the EV are in the process of forming a virtual middle lane for the EV to pass through. The light blue stripes represent lane area detectors, which are used to gather information about vehicles approaching the intersection.

The map consists of the four roads, each stretching 1 km away from the central intersection. Vehicles are randomly injected at the end of each road based on desired traffic flow density. Each of the incoming roads consist of two lanes, the left of which is dedicated for left turning vehicles and the right lane is dedicated for vehicles continuing straight and to the right. All the outgoing lanes consist of a single lane each.

In SUMO, EVs are simulated as vehicles of a special *emergency* vehicle class with a *bluelight* device. Such vehicles are able to either overtake other vehicles through virtual middle lanes, or through the opposite direction lane, depending on the lane change model used. Having both functionalities simultaneously is not currently supported in SUMO. Of the two, I opted to use the functionality with the formation of virtual middle lanes, since it more closely resembles desired behaviour of other vehicles. Under the current functionality, vehicles in range of the bluelight device move to the right edge of their lane unless they occupy the leftmost lane, in which case, they move to the left.

The vehicles in range of the bluelight device currently do not change their speed in reaction to the presence of the EV nor have the ability to give right of way to an EV running a red light when their traffic light signals green. Thus an EV crossing a TLJ with no method of preemption can have little to no effect on the surrounding traffic. One option to handle this would be not to use a blue light device at all and make the EV wait like all the other vehicles. This approach seems even more unrealistic though, and as such I implemented the former one.

4.2 Implementation of traffic signal control in SUMO using TraCI

SUMO itself has very limited options for the implementation of preemption algorithms. Traffic lights can be programmed to dynamically change signal plan phase durations within a predefined range based upon information from induction loops and the occupation of lane area detectors. For more sophisticated functionality one must use TraCI, a communication interface enabling on-line communication with the SUMO simulation.

A TraCI instance is initialised within a Python script, it connects to a SUMO simulation and allows for precise control of the behaviour of objects in the simulation. There is a special class object defined by TraCI, namely `stepListener`, which automatically runs a given code every simulation step.Therefore, I created a child class of `stepListener` called `Scheduler`, which functions as a central node of the traffic signal control for the intersection. `Scheduler` facilitates communication among various other classes, each systematically dedicated to a single subtask of traffic signal control. The interaction among these individual parts can be best explained using the schematic in Figure 4.2

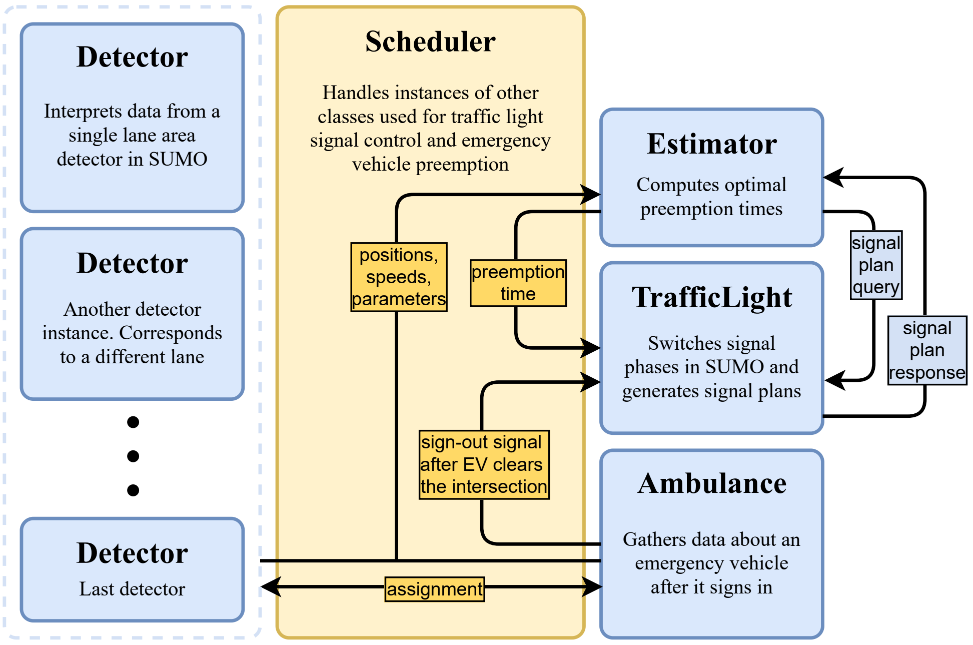

Detector
Interprets data from a single lane area detector in SUMO
Detector
Another detector instance. Corresponds to a different lane
Detector
Last detector
Scheduler
Handles instances of other classes used for traffic light signal control and emergency vehicle preemption
positions, speeds, parameters
preemption time
sign-out signal after EV clears the intersection
assignment
Estimator
Computes optimal preemption times
TrafficLight
Switches signal phases in SUMO and generates signal plans
Ambulance
Gathers data about an emergency vehicle after it signs in
signal plan query
signal plan response

Figure 4.2: Schematic of the functionality of `Scheduler` and other classes.

4.2.1 Functionality of `detector`

Each of the eight lane area detectors in the simulation has an `detector` instance assigned by `scheduler`. Since it is also a child of the `stepListener` class, it is designed to keep tack of the vehicles in the range of the detector, identify them, count them and request their positions and speeds from SUMO through TraCI.

4.2.2 Functionality of `Ambulance`

Whenever `Scheduler` detects an EV an instance of `Ambulance` is assigned to it. As a child of `stepListener` it requests position and speed of the EV from SUMO through TraCI every simulation step.

4.2.3 Functionality of `TrafficLight`

The primary function of `TrafficLight` is to increment the traffic signal plan on the intersection. Even though, this is normally handled natively by SUMO, the limitations on the traffic light signal plan (minimum phase lengths, insertion of yellow signal phase before preemption) made the automatic signal phase incrementation by SUMO very inconvenient. It required many requests on the current state of the signal plan to be made each simulation step, their subsequent interpretation and transformation to a more usable format, and then in the overwhelming majority of simulation steps, it was evaluated that no action is needed to be issued back to SUMO trough TraCI. For these reasons I have decided to keep track of the signal plan internally within the class and increment the signal plan when necessary through TraCI.

The next function of the class is to ensure preemption is active at the requested time. Assume that at $t = 0\,\mathrm{s}$ the desired premption time t_p^* is received. This means, the green phase of the preemption sequence must start at the latest possible time $t_\mathrm{switch} \leq t_\mathrm{p}^*$. Given minimal signal lengths l_min and the requirement for transition phases (yellow signal before red) of a fixed length l_trans we can determine the earliest time t_switch using the simple Algorithm 2. The preemption sequence needs to be initiated at time t if we compute that $t_\mathrm{switch}(t + \Delta t) > t_\mathrm{p}^*$. This requires `TrafficLight` to be able to compute the signal plan into the immediate future.

Algorithm 2: Earliest guaranteed preemption time

```
$p \leftarrow$ default signal plan
$t \leftarrow$ initial time
$\Delta t \leftarrow$ time increment
$t_\mathrm{last} \leftarrow$ last signal phase increment time
$t_\mathrm{trans}$ = transition phase length
$found$=false
while not found do
    $phase = p(t)$
    if not $phase.isTranisition$ then
        if $t_\mathrm{last} + phase.minDuration \leq t$ then
            $found$ =true
    $t := t + \Delta t$
Result: $t_\mathrm{switch} = t + t_\mathrm{trans}$
```

Lastly, `TrafficLight` has the capability to compute the signal plan including the preemption sequence. This is simply done using the default plan up to $\max(t) : t_\mathrm{switch}(t) \leq t_\mathrm{p}^*$ and then adding a transition state of length l_trans and green state downstream of the EV and a red state everywhere else for the rest of the desired signal sequence. Note, that a transition state does not necessarily mean a yellow signal. A transition phase state is determined according to the Table 4.1

		to	
		green	red
from	green	green	yellow
	red	red	red

Table 4.1: Transition phase state table

4.2.4 Functionality of `Estimator`

For the purposes of the validation of the proposed algorithm, I have implemented various methods of preemption corresponding to the following modes of operation of `Estimator`.

1. No preemption mode

In this mode of operation, `TrafficLight` only facilitates default traffic signal plan.

2. Polygon mode

   Preemption is initialised at the earliest time possible after an EV sign-in. Which means t_{p}^{*} is always zero.

3. Single computation mode

   In the simulation step, when the `Scheduler` detects an EV sign-in, the proposed algorithm is used to compute the optimal preemption time t_{p}^{*}. In order to do this `Estimator` requests signal plans with various preemption times t_{p} from `TrafficLight`. Since any supplied signal plan is feasible and optimal for a given t_{p} (meaning the preemption sequence starts at the latest possible time), `Estimator` has a guarantee of a feasible plan for simulations.

4. Recomputation mode

   The optimal preemption time t_{p}^{*} is computed repeatedly for as long as `TrafficLight` can guarantee, that preemption will be in effect at the previously computed time, i.e., $t < t_{\mathrm{switch}}\left(t_{\mathrm{last}}\right)$. The frequency of computation can be set to a desired value.

5. Short range mode

   In case the sign-in distance is greater than the range of the lane area detectors, there can be virtual vehicles inserted between the EV and the end of the range of the detectors. These make use of the traffic flow density for each vehicle type used to inject vehicles into the simulation. It takes the time to reach the detector range

   $$t_{\mathrm{EV2DET}} = \frac{d_{\mathrm{EV}} - l_{\mathrm{DET}}}{v_{\mathrm{EV}}}, \tag{4.1}$$

   where d_{EV} is the distance of the EV to the detector, l_{DET} is the range of the detector and v_{EV} is the speed of the EV.

   Then, given the probability p_{type} that a vehicle of a given type enters the simulation within a given second (which is known due to the fact, that this information is used for injecting vehicles into the scenario), we can compute the estimated number of vehicles of the given type $n_{\mathrm{type}} = \mathrm{ceil}\left(p_{\mathrm{type}} * t_{\mathrm{ev2det}}\right)$ between the end of the detection range and the EV, where ceil() equates to rounding up. This rounding is done to minimise the chance of an optimistic estimate, which could lead to unnecessarily delaying the EV. This is only due to a subjective preference of delaying surrounding traffic over delaying the EV.

   Lastly while alternating between vehicle types vehicles are placed between the end of detector range and the EV, until there are n_{type} vehicles of each vehicle type. Then they are equidistantly spread over this region and are given initial velocities taken from linear interpolation of velocities of the EV and the furthest vehicle in detector range. The optimization part of the proposed algorithm then includes these virtual vehicles.

It should be noted, that this estimation procedure is quite elementary and there exist plethora of more sophisticated techniques. I used this method mainly because the traffic flow density estimation is not the focus of this thesis and also it seemed excessive to do otherwise, given the fact, I already had the required values available.

4.3 Simulation experiments

To validate the proposed method I have run the scenario using various settings. The values I gathered from each were average halt times of the surrounding traffic and the EVs themselves as well. To be considered affected by the passage of the EV a vehicle had to fulfil the condition of there being such a time instance when the vehicle is on its way to the intersection and simultaneously the EV has signed-in but not yet cleared the intersection. This allows me to fairly compare different preemption methods, even though for some methods, vehicles which are not affected by the EV fulfil this condition.

Each test consists of 30 simulations where in each the injection time of the EV is incrmented by 3 s thus covering the whole 90 s of the default signal plan length. The earliest injection of an EV happens at $t = 300\,\mathrm{s}$, to allow the simulation to approach steady state. The set of 30 simulations is repeated total of 30 times using a different seed for the pseudorandom number generator uses to plan vehicle injection times. Thus getting a different distribution of vehicles for each set. This totals to 900 individual simulations for each tested setting. In all of the tests the EV enters from the west and leaves to the east.

For the purpose of the tests I defined the used traffic flow densities for the vehicle types as shown in Table 4.2

	vehicles per hour		
	LIGHT	MODERATE	HEAVY
passenger	1500	2250	3000
bus	60	75	90

Table 4.2: Definition of used traffic flow densities.

4.3.1 Preemption trigger distance for polygon method

To be able to compare the proposed method to the more traditional static polygon approach, we need to find the optimal polygon size for all the used traffic flow densities. In Table 4.3, we can see how the size of the polygon in the direction of the approach of the EV (trigger distance) combined with different traffic flow densities affects median minimum velocity to which an EV is forced to slow down, this is analogous to V^0_{min} from equation 3.1. In the table, the optimal values for individual traffic flow densities are highlighted.

In the rest of the thesis, whenever a polygon method with a given traffic flow density is used, the corresponding optimal trigger distance will be used.

	veh/hour	LIGHT	MODERATE	HEAVY
trigger distance	100 m	6.68	4.97	0.20
	200 m	13.41	9.07	2.59
	300 m	13.92	12.98	3.77
	400 m	13.90	13.50	5.59
	500 m	13.90	13.58	8.95
	600 m	13.90	13.59	11.01
	700 m	13.90	13.59	13.33
	800 m	13.90	13.59	13.33

Table 4.3: Median values of the slowest speeds $\left[\mathrm{ms}^{-1}\right]$ achieved by EVs.

4.3.2 Performance comparison between preemption methods

To compare the performance of individual methods I have used the MODERATE traffic flow density setting and a sign-in distance of 400 m. Figure 4.3 shows a box chart containing the halt times of an average vehicle influenced by the passage of the EV.

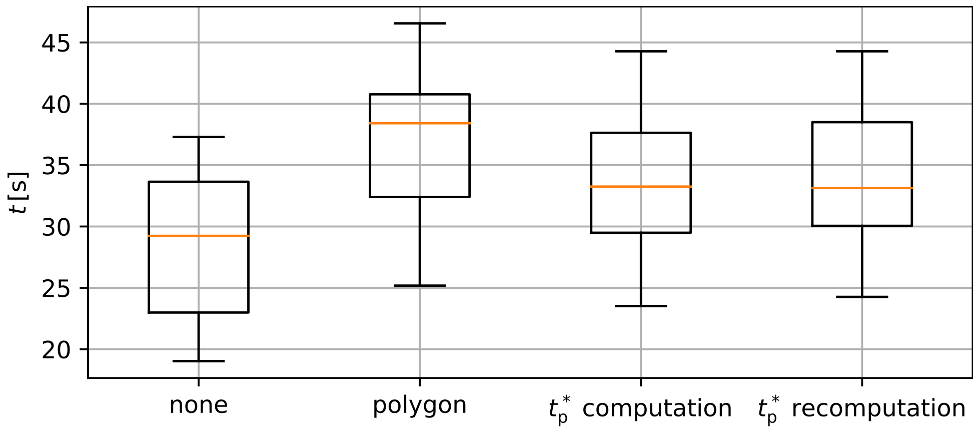


Figure 4.3: Average halt times of affected vehicles using different modes.

At first it may seem, that using no preemption has definitely the least impact on the surrounding traffic. This result is, however, the least accurate of all the methods, since SUMO does not currently support realistic behaviour of vehicles in proximity to the EV as stated at the beginning of Chapter 4. The other methods minimize the interaction between the EV and other vehicles and thus, this shortcoming of SUMO has a smaller impact on the result in those cases. The median values from Figure 4.3 can be seen in Table 4.4.

While the impact of the EV on surrounding traffic can be inaccurately simulated by SUMO, the opposite is not the case. Vehicles take time to form

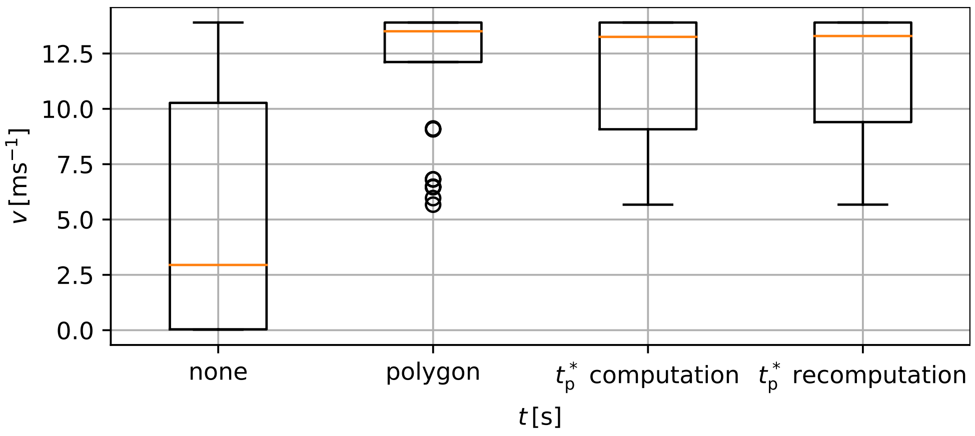


Figure 4.4: Minimum EV velocities $V_{\min}$.

method	none	polygon	t_p^* computation	t_p^* recomputation
time [s]	29.24	23.41	33.25	33.13

Table 4.4: median halt time of an average vehicle for different preemption methods

a virtual middle lane and do not break the rules of traffic to move out of the way. Thus, measuring the lowest speed $V_{\min}$, to which the EV is forced to slow down, can be used to represent this influence.

Furthermore, since the main goal of premption is to shorten the response times of EVs and make their passage safer, even if SUMO could simulate the impact the EV has on an intersection more accurately, the impact an intersection without preemption has on the EV cannot be justified, as seen in Figure 4.4. We can see that without preemption the median slowdown is very significant, while for all the investigated methods of preemption these values are comparable.

4.3.3 Influence of sign-in distance

For this series of tests, I investigate the impact of the distance from which an EV sends its sign-in signal to initiate the given preemption method. I compare the proposed method with the static polygon approach. For the polygon the preemption distance is set to its optimal one as per Table 4.2. The traffic flow density setting used is MODERATE. The investigated distances are 200, 500 and 800 m. The results can be seen in Figures 4.5 and 4.6.

From the results it can be seen, that the proposed method should should have a sign-in distance at least identical to the polygon-based method. Lower distances do not give the algorithm enough flexibility to allow for optimal preemption. As for longer distances, the single computation variation of the proposed algorithm does lose its accuracy with the rising sign-in distance, Therefore, if it were to be implemented in real world the sign in distance would need to be adjusted to best perform on a given intersection.

The recomputation version of the algorithm does perform the best of all

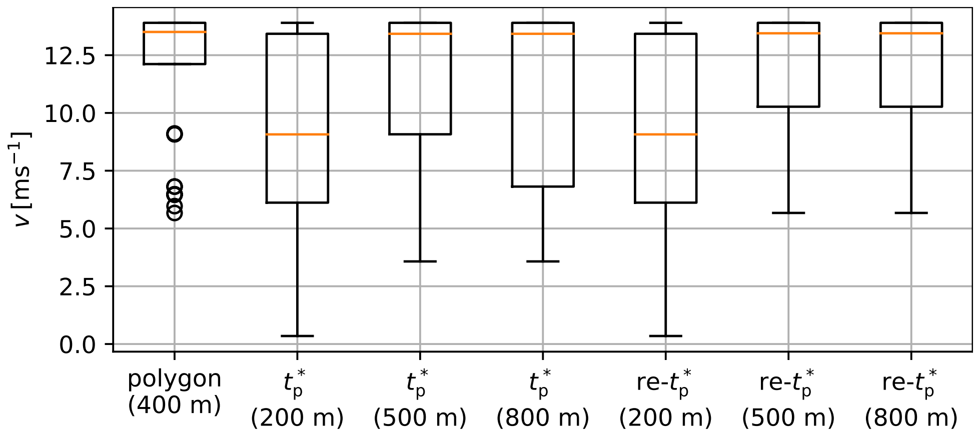


Figure 4.5: Effect of diffrent sign-in distances and preemption methods on the minimum EV speed

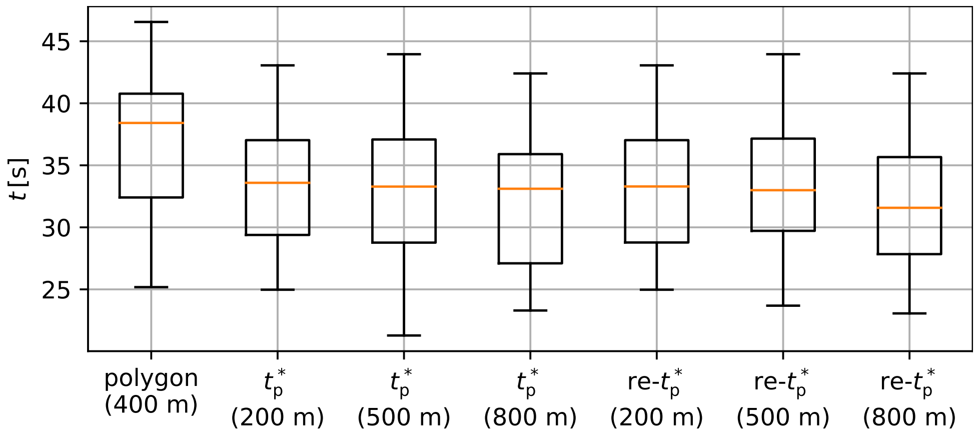


Figure 4.6: Effect of diffrent sign-in distances and preemption methods on the average halt times

the investigated options and with higher sign-in distances it has more time to reevaluate the optimal time of preemption. Thus, if it were to be implemented in the real world, the sign-in distance should be set to the longest possible.

4.3.4 Influence of traffic flow density

This series of tests shows how the single computation and heavy computation handle changing traffic flow density and if there is any need for manual adjustment of the methods. The sign-in distance in all tests is set to 800 m. As seen in Figures 4.7 and 4.8, there is little to no difference between the average halt times, thus the recomputation only helps with lowering passage times of the EVs when higher traffic flow densities are encountered.

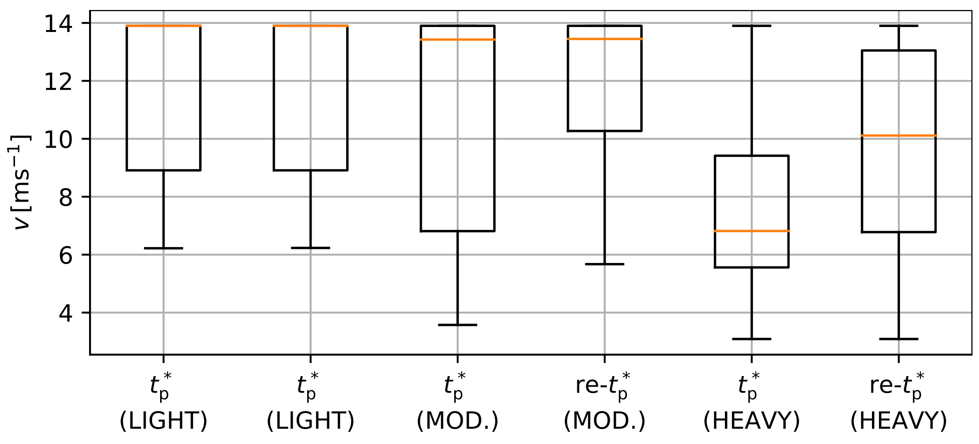


Figure 4.7: Comparison of the effect diffrent traffic flow densities have on the minimum EV speed

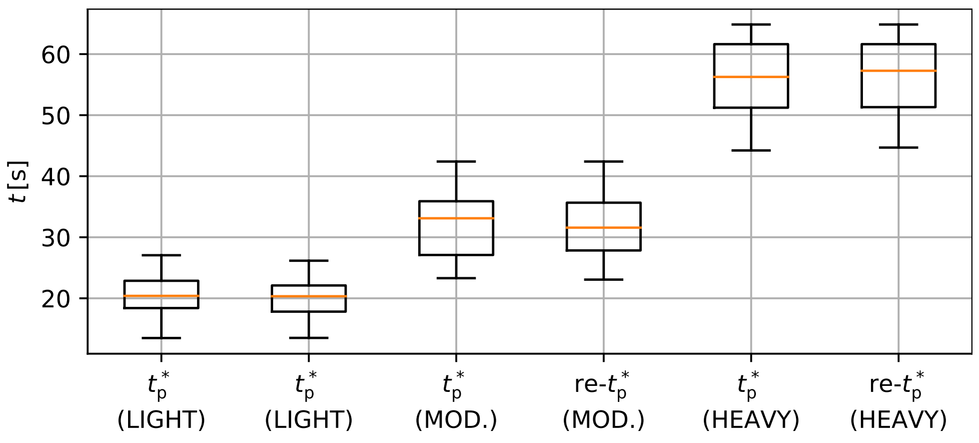


Figure 4.8: Comparison of the effect diffrent traffic flow densities have on the average vehicle halt time

4.3.5 Viability of short detector ranges

In this last series of test I aim to show whether the proposed method can be used in scenarios where the detector range is limited, but the sign-in distance is greater. This could be interpreted as a situation where the data is collected using computer vision with cameras placed around the intersection, but not too far from it. This is the implementation most likely to be implemented using current technologies and thus could be the first testable version of the algorithm.

All tests have a sign-in distance of 400 m, the traffic flow density is set to MODERATE and the range of the lane area detectors is limited to 100 m, thus the algorithm needs to make virtual vehicles to fill the unobserved area ahead of the EV. The results, labeled SHORT, are compared to the previous results, labeled LONG, where the detector range was unlimited and the static polygon generated results.

The Figures 4.9 and 4.10 show there is no decrement in the minimum speed

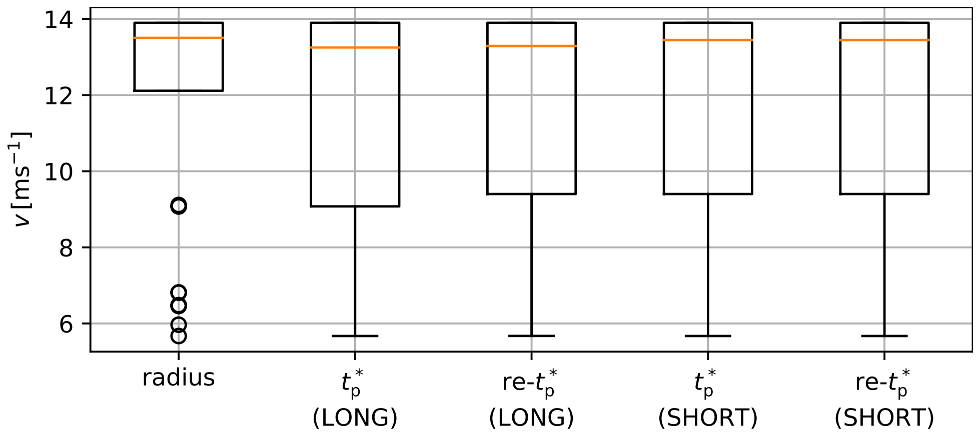


Figure 4.9: Short detector range effect on the minimum EV speed.

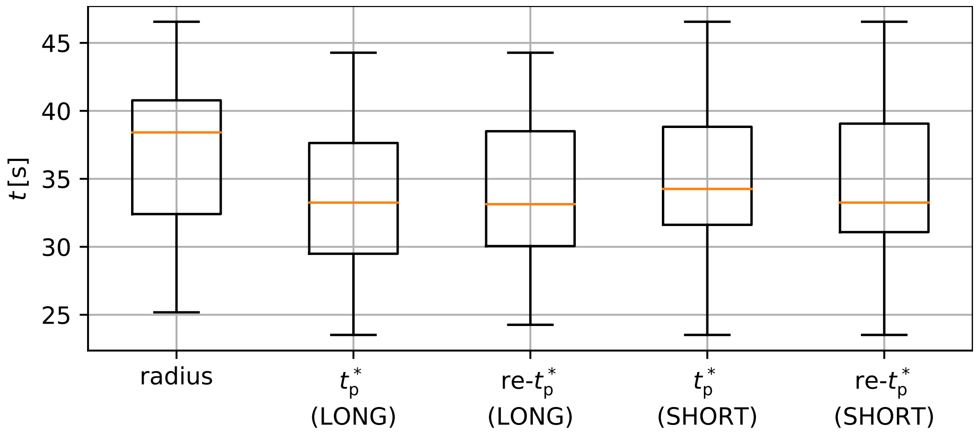


Figure 4.10: Short detector range effect on the average vehicle halt time.

EVs reach form the LONG variant and the impact on surrounding traffic due to the passage of EVs does not increase significantly and still performs better than the static polygon approach. It should be noted, however, that the polygon happens to have the preemption distance fine-tuned to the traffic flow density, and therefore, the result is likely closer to an adaptive polygon implementation.

Given the fact, that the recomputation version of the algorithm has been shown to perform best, when large sign-in distances are used. As a last experiment I decided to repeat the previous tests with different values. The results shown in Figures 4.11 and 4.12 show that the proposed method is in this scenario comparable to a dynamic (400 m) polygon preemption method, the static (800 m) polygon method fares significantly worse. The results show that the median halt time of an average vehicle is still below that of any of the polygon methods tested. The variance is quite large, however. I believe this could be mitigated through better estimates on the number of vehicles between the end of the detector range and the EV This estimate could be entirely eliminated through the use of V2X communication. On the other hand, it is quite difficult to argue for the communication not being used to

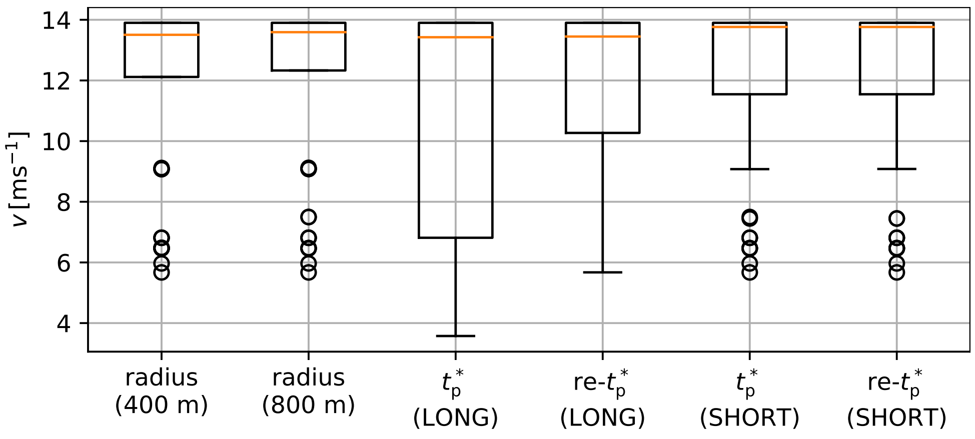


Figure 4.11: Short detector range effect on the minimum EV speed with sign-in at 800 m.

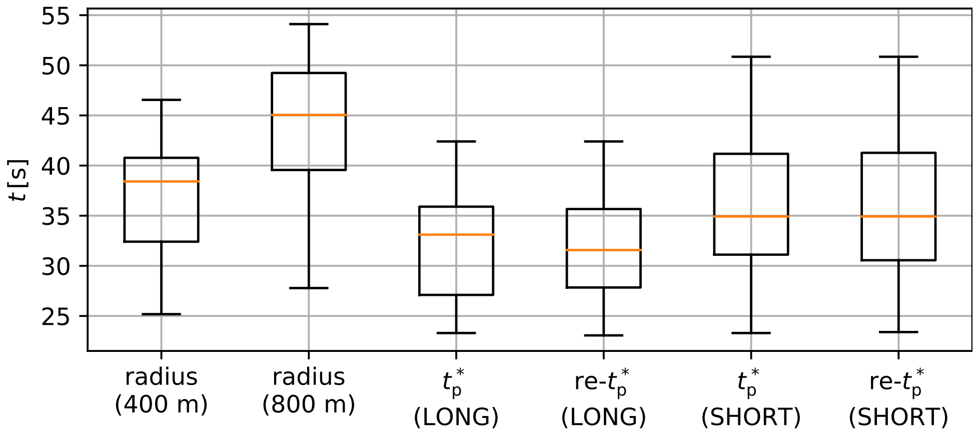


Figure 4.12: Short detector range effect on the average vehicle halt time with sign-in at 800 m.

give precise location and speed information.

Chapter 5
Conclusion

In conclusion, this thesis focused on the development of new traffic light preemption techniques for emergency vehicle prioritization at traffic light junctions.

In Chapter 1, I have introduced the motivations behind traffic light preemption, its advantages and its shortcomings. I have give a summary of existing approaches and discussed their individual strengths and weaknesses. Furthermore, I have given a short outline of this thesis.

In Chapter 2, I have investigated car-following models, explained how they work and when they are useful. Next, I have done an analysis of the accuracy of several car-following models at intersections, namely IDM and IIDM, I also proposed a linear queue discharge model, which has shown to be surprisingly accurate in simulating the positions of vehicles in a queue at a traffic light.

When it comes to IDM and IIDM, I showed that the commonly used IDM is unnecessarily inaccurate in modelling velocities of vehicles going through an intersection, and that its improved version is much better at this task. I selected IIDM as the model for simulation of vehicle platoons going through a traffic light controlled intersection for the remainder of my work.

In Chapter 3, I proposed a new time optimal traffic signal preemption algorithm which utilizes the potential of modern traffic data gathering technology. The method is built upon several assumptions, namely, knowing accurate positions and speeds of vehicles, having good estimates on the car-following model parameters of these vehicles, and having complete information about the traffic signal phase plan, including alternative plans containing preemption sequences, for the duration of passage of the emergency vehicle.

Furthermore, I have discussed how the aforementioned assumptions can be satisfied in the real world and offered an adaptation of the method for situations, where the range of vehicle localization devices is limited.

In Chapter 4, I used the traffic simulator SUMO to validate the developed method. I have explained, how the simulation scenario was created and how the method was implemented through a communication interface TraCI, using Python programming language. Next I designed multiple tests to show the performance of the method compared to the most widespread method of traffic signal preemption, namely, lane area triggered preemption.

The results show that the developed method surpasses the traditional approaches in every practical case except for complexity. Specifically, it ensures the emergency vehicle passes the intersection with minimum delay and maximum safety, while also minimizes the potential impact on surrounding traffic. The method delays the preemption sequence initialization to the latest possible instance, which still enables the emergency vehicle minimum delay.

The alternative method for cases where detection devices have limited range, has also been verified, It, understandably, performs worse than the unlimited method, but I haven't been able to show it performing worse than the commonly used approach. The overall performance was more significantly influenced by various specific traffic conditions present in the scenario.

Lastly, I want to suggest a direction for future research. Given the fact, that the alternative, limited, method is technologically more accessible, I suggest to find a more advanced method for filling the dead zone between the emergency vehicle and the end of the range of vehicle detectors with virtual vehicles for the purposes of simulation.

On a similar note, the method could be, potentially, enriched with a lane changing model. As it stands, the method does not simulate overtaking, something emergency vehicles do quite often. Should the method have the capability to accurately simulate the sometimes aggressive manoeuvres emergency vehicles make, I suspect the resulting method could have major impact on city traffic. The implementation, in that case, should not be done as crudely as I demonstrated. And instead, should be implemented directly into a popular traffic simulator, such as SUMO, which in turn would be used to simulate real traffic for the purpose of evaluating the cost function of the algorithm. Contributing to the SUMO project is something I wish I had time to do simultaneously with the work described in this thesis.

Bibliography

[1] Laura Bieker. “Traffic safety evaluations for Emgergency Vehicles”. In: *Young Researchers Seminar.* June 2015. URL: https://elib.dlr.de/97829/.

[2] Subash Humagain et al. “A systematic review of route optimisation and pre-emption methods for emergency vehicles”. In: *Transport Reviews* 40 (July 2019), pp. 1–19. DOI: 10.1080/01441647.2019.1649319.

[3] C. Shawn Burke, Eduardo Salas, and J. Peter Kincaid. “Emergency Vehicles that Become Accident Statistics: Understanding and Limiting Accidents Involving Emergency Vehicles”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 45.4 (2001), pp. 508–512. DOI: 10.1177/154193120104500451. eprint: https://doi.org/10.1177/154193120104500451. URL: https://doi.org/10.1177/154193120104500451.

[4] Vít Obrusník, Ivo Herman, and Zdeněk Hurák. “Queue discharge-based emergency vehicle traffic signal preemption”. In: *Proceedings of the 21st IFAC World Congress.* Vol. 21. Berlin, Germany: IFAC, July 2020. URL: https://www.ifac2020.org/.

[5] S. Krauss, P. Wagner, and C. Gawron. “Metastable states in a microscopic model of traffic flow”. In: *Phys. Rev. E* 55 (5 May 1997), pp. 5597–5602. DOI: 10.1103/PhysRevE.55.5597. URL: https://link.aps.org/doi/10.1103/PhysRevE.55.5597.

[6] Martin Treiber. *Traffic Flow Dynamics.* Jan. 2013. ISBN: 978-3-642-32459-8. DOI: 10.1007/978-3-642-32460-4.

[7] Martin Fellendorf. “VISSIM: A Microscopic Simulation Tool to Evaluate Actuated Signal Control including Bus Priority”. In: Oct. 1994.

[8] Laura Bieker-Walz et al. “Evaluation of car-following-models at controlled intersections”. In: *European Simulation and Modelling Conference.* Vol. 31. Oct. 2017, pp. 247–251. URL: https://elib.dlr.de/115720/.

[9] MATLAB. *9.9.0.1524771 (R2020b).* Natick, Massachusetts: The MathWorks Inc., 2020. URL: https://www.mathworks.com/products/matlab.html.

[10] Pablo Alvarez Lopez et al. “Microscopic Traffic Simulation using SUMO”. In: *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, Nov. 2018, pp. 2575–2582. URL: `https://elib.dlr.de/127994/`.

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Pospíchal Lukáš**	Personal ID number:	**465813**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Control Engineering**		
Study program:	**Cybernetics and Robotics**		
Branch of study:	**Cybernetics and Robotics**		

II. Master’s thesis details

Master’s thesis title in English:

Modeling and optimization for traffic signal preemption for emergency vehicles using V2X communication

Master’s thesis title in Czech:

Modelování a optimalizace pro zajištění prioritního průjezdu vozidel IZS světelně signalizovanou křižovatkou s využitím komunikace V2X

Guidelines:

1. Prepare a thorough survey of the state of the art in the domain. If possible, include not only scholarly papers but also some technical information about currently deployed solutions, at least in Czechia.
2. Choose a suitable type of a mathematical model of the "discharge" of a queue of vehicles setting into motion upon the traffic light turning green. Validate your model using empirical data.
3. Design a feedback control algorithm for a preemption of a traffic light signal plan that uses the V2I communication with the incoming emergency vehicle, the estimate of the number of cars in the queue and the knowledge of the current signal plan.
4. Verify you solution(s) using numerical simulation with some domain specific simulator such as SUMO.

Bibliography / sources:

[1] Vít Obrusník, Ivo Herman, Zdeněk Hurák. Queue discharge-based emergency vehicle traffic signal preemption. 21th IFAC World Congress (https://www.ifac2020.org/), Berlin, July 2020.
[2] Martin Treiber and Arne Kesting. Traffic Flow Dynamics: Data, Models and Simulation. Springer-Verlag, Berlin Heidelberg, 2013.

Name and workplace of master’s thesis supervisor:

doc. Ing. Zdeněk Hurák, Ph.D., Department of Control Engineering, FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **09.02.2021** Deadline for master's thesis submission: **21.05.2021**

Assignment valid until: **30.09.2022**

doc. Ing. Zdeněk Hurák, Ph.D.	prof. Ing. Michael Šebek, DrSc.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

.
Date of assignment receipt Student’s signature