

1. REPORT NUMBER  CA13-2157-B	2. GOVERNMENT ASSOCIATION NUMBER	3. RECIPIENT'S CATALOG NUMBER
4. TITLE AND SUBTITLE Advanced Traffic Signal Control Algorithms	5. REPORT DATE September 2013	
	6. PERFORMING ORGANIZATION CODE	
7. AUTHOR  Andreas Winckler	8. PERFORMING ORGANIZATION REPORT NO.  UCB-ITS-PRR-2013-XX-B	
	9. PERFORMING ORGANIZATION NAME AND ADDRESS California PATH 1357 S.46th Street, Building. 452 Richmond, CA 94804-4648	
12. SPONSORING AGENCY AND ADDRESS California Department of Transportation Division of Research, Innovation and System Information 1227 O Street Sacramento CA 95814	10. WORK UNIT NUMBER	
	11. CONTRACT OR GRANT NUMBER  65A0376 DTFH61-10-H-00002	
13. TYPE OF REPORT AND PERIOD COVERED Final Report		14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES  
 This work was performed as part of the California PATH program of the University of California, in cooperation with the State of California, Transportation Agency, Department of Transportation, and the United States Department of Transportation, Federal Highway Administration.

16. ABSTRACT  
 This research is aimed at reducing fuel consumption in situations where the car is facing multiple traffic signals in a row on its route. Recent studies show that it is possible to reduce the fuel consumption by 12% by adjusting the driving strategy using Signal Phase and Timing (SPaT) information. However, these results are based on simulations. These results are further corroborated by the results reported in(Asadi, 2011) (Mahler, 2012). The main goal of this project is to build a prototype system that shows that it is possible to reduce the fuel efficiency in the above mentioned situations. Therefore an in-vehicle system computes a speed recommendation - based on current SPaT information - and provides it to the driver on a graphical interface. Based on this recommendation the driver should be able to adjust his/her driving speed resulting in improved fuel consumption. In the first field test, the position data of the vehicle is sent to a second system called Adaptive Priority for Individual Vehicle (APIV). APIV is an operational strategy that adapts signal timing to facilitate the movement of individual vehicles through signalized intersections. While the main focus of the speed recommendation system is on reducing fuel consumption, the prime focus of APIV is on reducing the number of stops at red traffic and reduce fuel consumption. In addition APIV helps in reducing the number of stops at red traffic and associated intersection delays which leads to reduced travel time. Comprehensive field tests using a BMW vehicle showed that significant fuel savings are possible using the APIV.

17. KEY WORDS Traffic signals, adaptive control, connected vehicles, mathematical models	18. DISTRIBUTION STATEMENT This document can be distributed without any restrictions	
19. SECURITY CLASSIFICATION (of this report)  Unclassified	20. NUMBER OF PAGES  54	21. COST OF REPORT CHARGED

## **DISCLAIMER STATEMENT**

This document is disseminated in the interest of information exchange. The contents of this report reflect the views of the authors who are responsible for the facts and accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California or the Federal Highway Administration. This publication does not constitute a standard, specification or regulation. This report does not constitute an endorsement by the Department of any product described herein.

For individuals with sensory disabilities, this document is available in Braille, large print, audiocassette, or compact disk. To obtain a copy of this document in one of these alternate formats, please contact: the Division of Research and Innovation, MS-83, California Department of Transportation, P.O. Box 942873, Sacramento, CA 94273-0001.



# **APPENDIX A**

## **Exploratory Advanced Research Project**

### **“Advanced Traffic Signal Control”**

#### **BMW Final Report**

*Date: July 31, 2012*

*Authors: Andreas Weber, Andreas Winckler*

# Table of Contents

1	Motivation and Problem Description .....	3
2	Test System Setup .....	4
2.1	System Overview.....	4
2.2	Communication.....	6
2.2.1	Communication mechanism .....	6
2.2.2	Time synchronization .....	6
2.2.3	Messages.....	7
2.2.3.1	MAP message.....	7
2.2.3.2	SPaT Message.....	7
2.3	Vehicle setup.....	10
2.3.1	Research Vehicle .....	10
2.3.2	Modules .....	12
2.3.2.1	CarApp.....	12
2.3.2.2	Programmable instrument cluster.....	14
2.4	Cloud Server .....	14
2.4.1	Server Setup .....	14
2.4.2	Database Tables .....	15
2.4.3	Modules .....	15
2.4.3.1	Server Forwarder .....	16
2.4.3.2	Data Base Writer .....	16
2.4.3.3	Data Base Archiver .....	16
2.4.3.4	ServerSide .....	17
3	Fixed time traffic signals .....	19
3.1	Velocity Planning Algorithm.....	19
3.2	Adaptive Priority for Individual Vehicle .....	20
3.3	Richmond Field Station test field .....	20
3.3.1	Setup .....	20
3.3.2	Scenarios .....	21
3.3.3	Results.....	22
4	Actuated and coordinated traffic signals.....	28
4.1	New speed advisory algorithm .....	28
4.1.1	Basic idea .....	28
4.1.2	Step 1: Select best arrival time .....	29
4.1.3	Step 2: Calculate speed trajectory .....	38
4.1.4	Step 3: Get speed recommendation from trajectory .....	42
4.1.5	Shortcomings & ideas for improvements .....	42
4.2	El Camino Real.....	43
5	Conclusions .....	48
6	Appendix: MAP and SPaT Messages .....	49
7	Bibliography .....	52

# 1 Motivation and Problem Description

When a car stops at a red traffic signal, it uses approximately 0.02 liters of fuel when it pulls away. This corresponds to roughly 5 grams of CO<sub>2</sub>. In urban traffic, which in the United States is regulated by roughly 311,000 traffic signal lights (National Transportation Operation Coalition, 2012), the 254 million cars here emit 80 million tons of CO<sub>2</sub> or approximately 20 percent of their total emissions while accelerating from a traffic signal.

In 2010, urban Americans had a total of 4.8 billion hours traffic delay which resulted in an extra 1.9 billion gallons of fuel use for a congestion cost of \$101 billion. In 2010, on average an urban American spent 34.4 hours in traffic delay (Schrank, 2011). A big portion of the idle time is the time spent in front of traffic signals. (US DOT Federal Highway Administration). Effective advanced traffic signal control methods such as traffic-actuated signals and signal synchronization have been deployed at many traffic intersections and help to save precious time and expensive fuel every day. However, such measures are very costly to implement and maintain. The total spending on signal operation and maintenance programs only, was around 425 million dollars among 219 districts that responded to the NTOC survey (National Transportation Operation Coalition, 2012). Even with these measures in place, people often cruise at full speed towards a green traffic signal and have to come to a sudden stop when the light turns red. Additionally, safety is also compromised at intersections when a driver is faced with the critical decision to proceed or stop when a light turns yellow. This lack of information about the future state of the traffic signal increases fuel consumption, trip time, emissions and engine and brake wear (Asadi, 2011). In an ideal situation, if the future of a light timing and phasing is known, the speed could be adjusted for a timely arrival at green. Summed up, the problem statements are

- too many standstills in front of a red light
- as well as unnecessary braking and acceleration events
- resulting in higher fuel cost and travel time for the driver
- and more emissions to the environment.

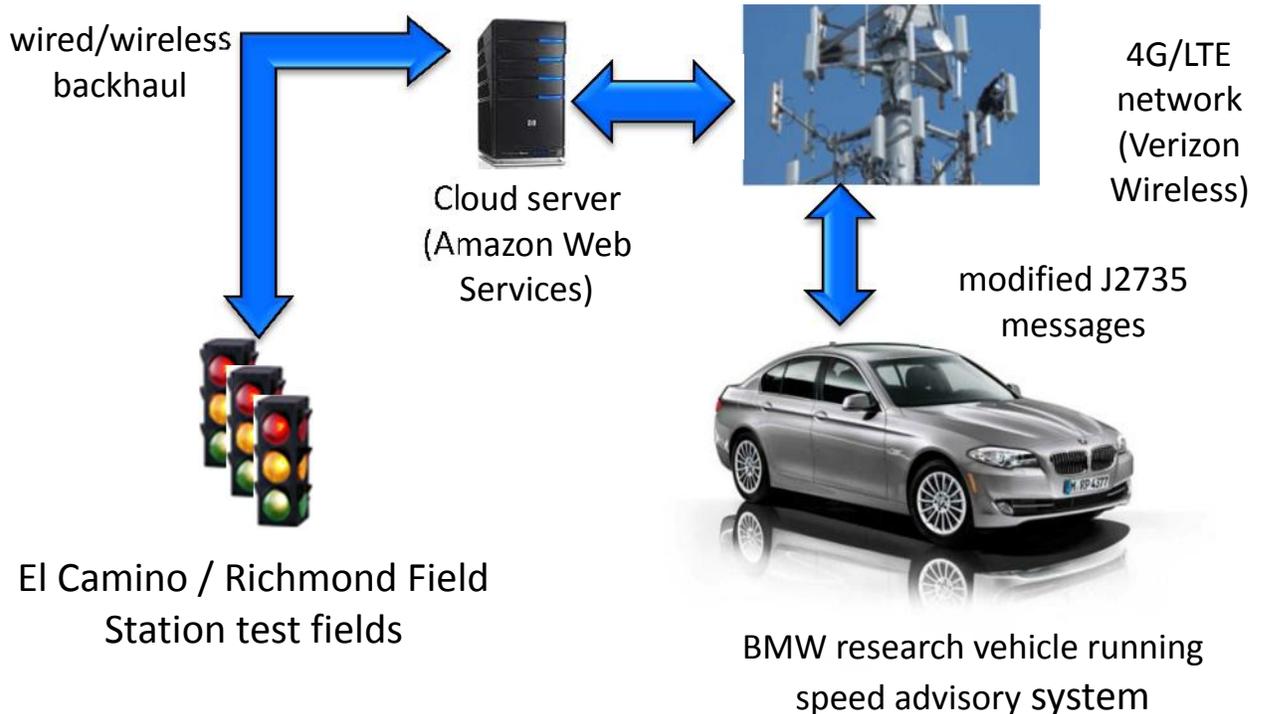
This research is aimed at reducing fuel consumption in situations where the car is facing multiple traffic signals in a row on its route, in order to solve the problems stated. Recent studies (Barth, Mandava, Boriboonsomsin, & Xia, 2011) showed that it is possible to reduce the fuel consumption by 12% by adjusting the driving strategy using Signal Phase and Timing (SPaT) information. However, these results are based on simulations. These results are further corroborated by the results reported in (Asadi, 2011) (Mahler, 2012). The main goal of this project is to build a prototypic system that shows that it is possible to reduce the fuel efficiency in the above mentioned situations. Therefore an in car system computes a speed recommendation - based on current SPaT information - and provides it to the driver on a graphical interface. Based on this recommendation the driver should be able to adjust his/her driving speed resulting in improved fuel consumption. In the first field test, the position data of the

vehicle is sent to a second system called *Adaptive Priority for Individual Vehicle (APIV)*. APIV is an operational strategy that adapts signal timing to facilitate the movement of individual vehicles through signalized intersections. Whereas the main focus of the speed recommendation system is on reducing fuel consumption, the main focus of APIV is on reducing the number of stops at red traffic and associated intersection delays which leads to reduced travel time. So, a combination of both may be a good solution for the mentioned problem statements.

## 2 Test System Setup

### 2.1 System Overview

The chapter provides information about the architecture of the prototypic system which allows data communication between cars and traffic signals to experience a faster, safer, and more fuel efficient ride. Figure 1 shows a general overview of the system. The SPaT information is transmitted between the traffic signals and a cloud server over a 3G network. The communication between the server and the research vehicle is based on a 4G/LTE connection.



**Figure 1: System overview**

The prototype consists of several modules that interact with each other to guarantee both flexibility and performance. Figure 1 shows an overview of these modules divided into three spatially separated systems.

- **Traffic signals**  
Predicting next traffic signals phases and transmitting the information to database servers.
- **Cloud server**  
Storing of all relevant static (position and geometry) and dynamic (timing and phases) data of fixed-time and actuated traffic signals. Processing, encoding, and communication modules.  
Archiving timing data for later statistical analysis.
- **Car**  
Gathering of GPS position of the car and all relevant intersection data from a cloud server.  
Processing and providing of the data to local clients including the HMI.

The IP communication between the car and the data base server is based on a 4G/LTE connection. Also the data between fixed-time/actuated traffic signals and the server is transmitted wirelessly over a 3G network or in the case of Richmond Field Station over a landline internet connection. Both servers (data base and PATH) are stationary and so possess a normal internet connection.

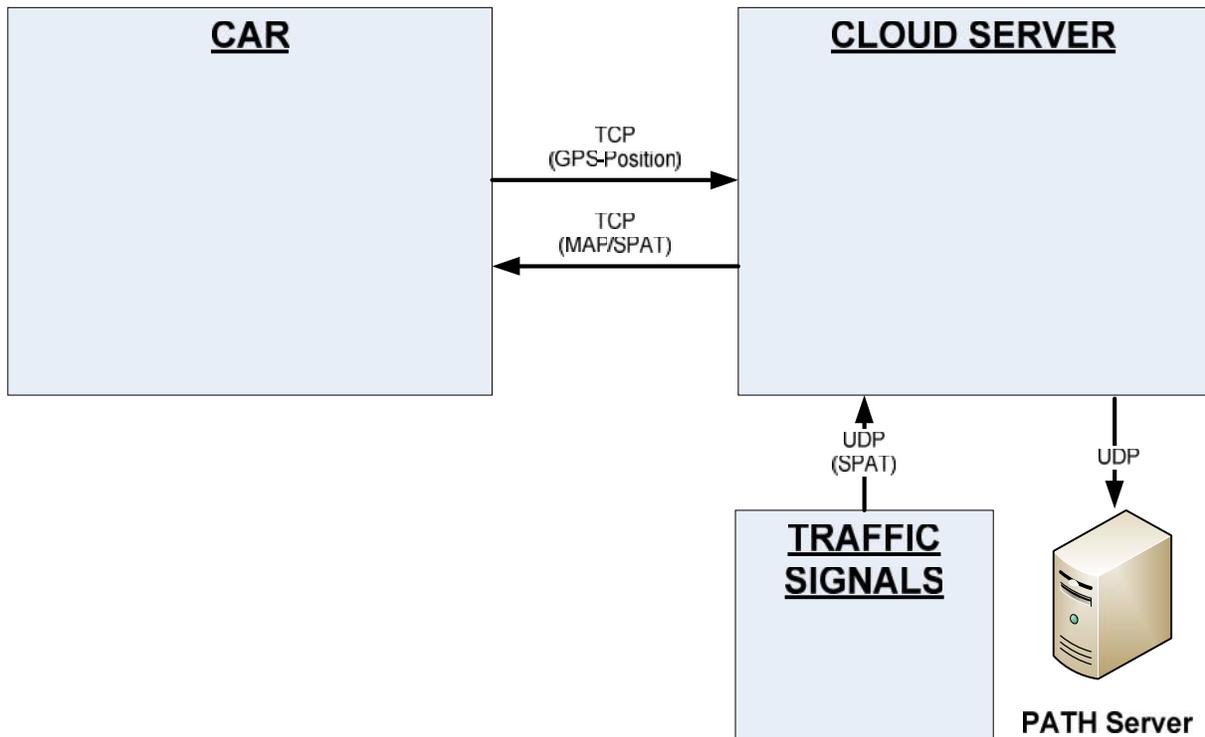
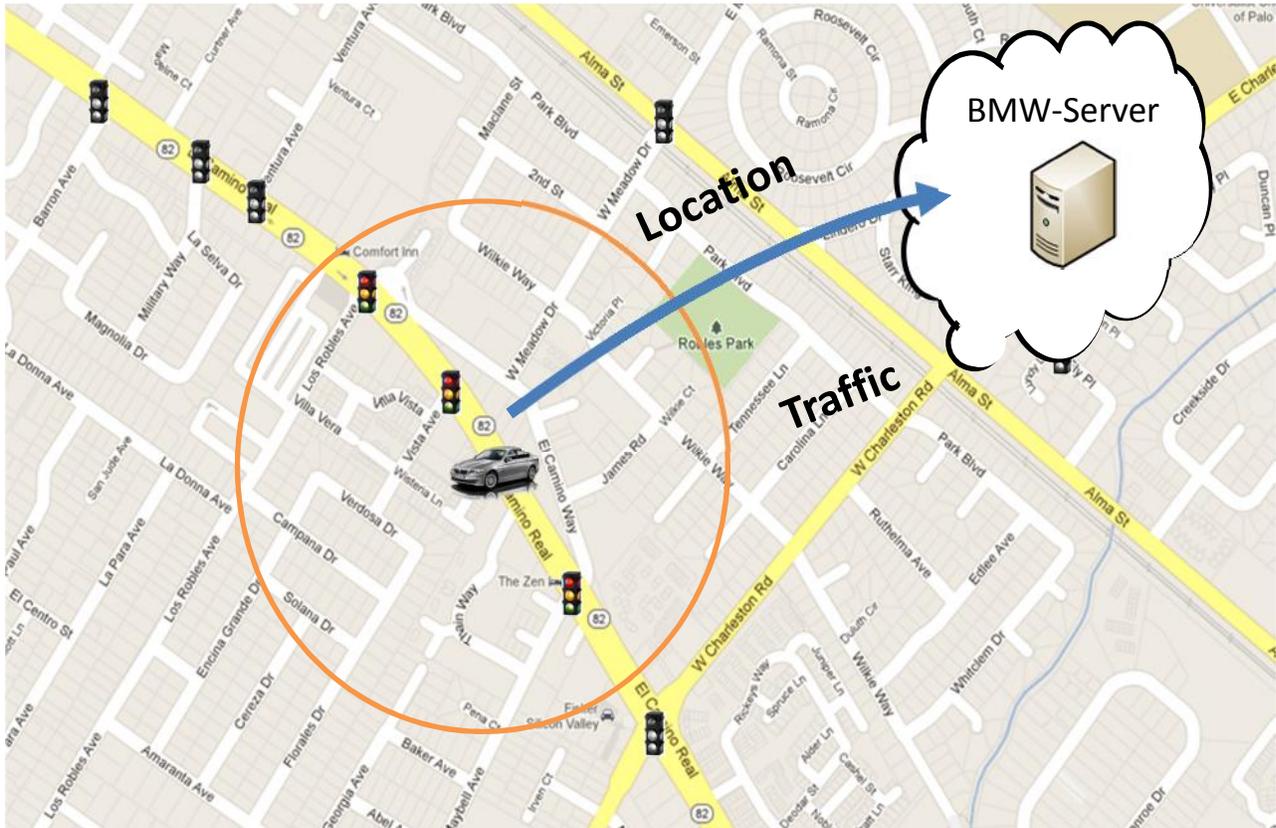


Figure 2: System overview

Figure 2 shows a more detailed view of the system. The different software modules on the car as well as on the server side are explained in detail in chapter 2.3.2 and 2.4.3.4 respectively.

## 2.2 Communication

### 2.2.1 Communication mechanism



**Figure 3: Communication mechanism**

In a real world scenario, it is not feasible to send a client updates for all traffic signals in every time step. For that reason, the client sends its position periodically to the server. That allows the server to keep track of the position of its clients (i.e. equipped cars). Knowing the position of the client, the server only sends SPaT and MAP messages for traffic signals that are close to the client's position. So in the situation shown in Figure 3, the server sends only information about the traffic signals that are located in the orange circle. The information about the current location, as well as the information about traffic signals is communicated at 1Hz. A more detailed description of the update mechanism is described in chapter 2.4.3.4.

### 2.2.2 Time synchronization

The messages that are exchanged between traffic signals, server and car contain information about the timing of traffic signals. To be able to interpret this timing information in a correct way, it is important to

do a time synchronization between the systems. Therefore all computer systems (traffic signal PC, BMW-server, car PC) constantly synchronize their clocks over an internet connection. Using a timestamp (that indicates the creation time) in the SPaT message allows to recalculate the timing of signals without distortions due to the time needed for transmission of the message.

### 2.2.3 Messages

The communication between traffic signals, car, and server is based on the SAE J2735 protocol definition. In particular, MAP and SPaT message types are used to provide traffic signal information to the car. The following sections describe simplified versions of the MAP/SPaT messages. The real implementation of the MAP and SPaT messages can be found in Appendix (6).

#### 2.2.3.1 MAP message

This message is used for the communication between the BMW-Server and the car. It consists of static data about the traffic signal, e.g. its GPS position, available lanes and the direction of those lanes. Figure 4 depicts the structure of the MAP message. To identify the intersection that is described in the MAP message the id field contains a unique identification of the intersection. The field position contains the geographic coordinates of the center of the intersection. Phases describe the different approaching and exiting directions that have own controller timings. Typically four-legged intersections have 8 phases: 4 for through movements and 4 for left turns. The MAP message contains for each phase an id, the speed limit for this phase and the position/direction of the phase relative to the center of the intersection.

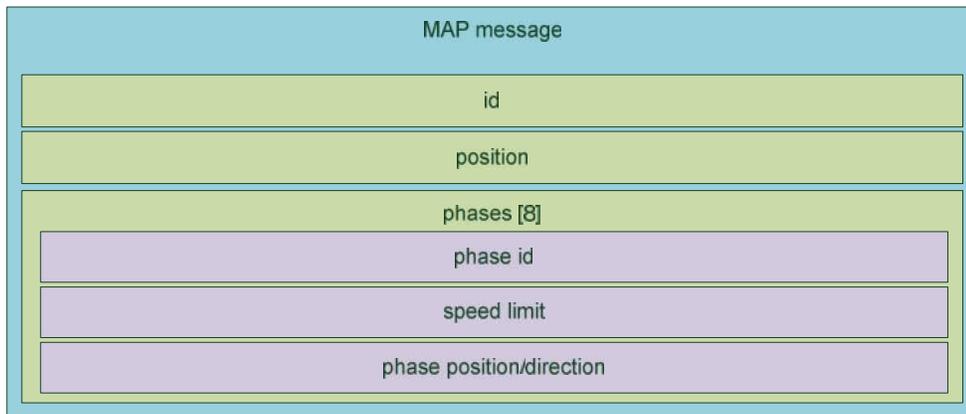


Figure 4: Structure of MAP message

#### 2.2.3.2 SPaT Message

The SPaT message (SPaT = Signal Phase and Timing) contains dynamic information about the traffic signal, such as current and future phase timings or the current status of the signal. Due to different information requirements different SPaT messages were used for fixed-time traffic signals and actuated coordinated traffic signals. The SPaT message for actuated coordinated traffic signals contains the same information as the message for fixed-time traffic signals, but is extended by some additional fields.

### 2.2.3.2.1 Fixed time traffic signals



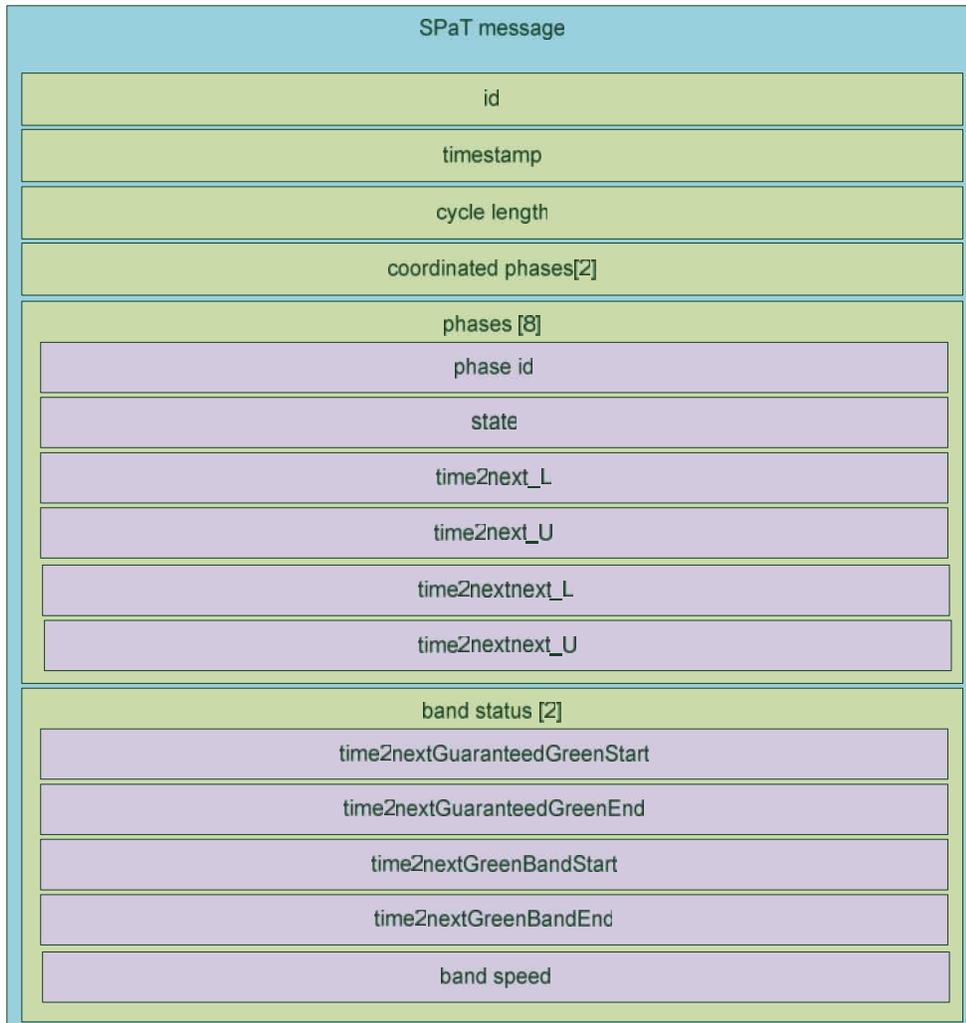
**Figure 5: Structure of SPaT message for fixed time traffic signals**

Like the MAP message, the SPaT message contains a unique id to identify the intersection. A timestamp field contains the time when the message was sent by the traffic signals. The timestamp makes it easy to decide, if the message is up to date and can be processed or an old one that is out of date and not useful for current computations anymore. Like the MAP message, the SPaT message contains further information for every phase. The field state contains the current status of the traffic signal for this phase (green, yellow or red). time2next\_L/time2next\_U contains the earliest /latest point of time when the current state is going to change (i.e., green to yellow, yellow to red, or red to green). time2nextnext\_L/time2nextnext\_U contains the earliest/latest point of time when the next state change will occur (i.e., green to red, yellow to green, or red to yellow). For fixed time traffic signals, it is valid that time2next\_L = time2next\_U and time2nextnext\_L = time2nextnext\_U. For actuated traffic signals, the earliest and latest points of time for state change are usually different to address the uncertainties in phase durations that are influenced by pedestrian/vehicle calls and vehicle actuations.

### 2.2.3.2.2 Actuated and coordinated traffic signals

The SPaT message for actuated coordinated traffic signals extends the SPaT message for fixed-time traffic signals with information about the guaranteed green window at the targeted intersection and the green band with adjacent arterial signals. The guaranteed green specifies an interval where the signal status is green in every cycle no matter what the status of the actuation is. The green band is a time interval that allows the driver to hit the green of several subsequent lights – given that he/she drives with band speed. This additional information is not given for the side streets and left-turn movements (they don't operate coordinated and don't have guaranteed green times) but for the thru movements of

the main arterial. The SPaT message contains the start/end of the next guaranteed green as well as of the next green band.



**Figure 6: Structure of SPaT message for actuated traffic signals**

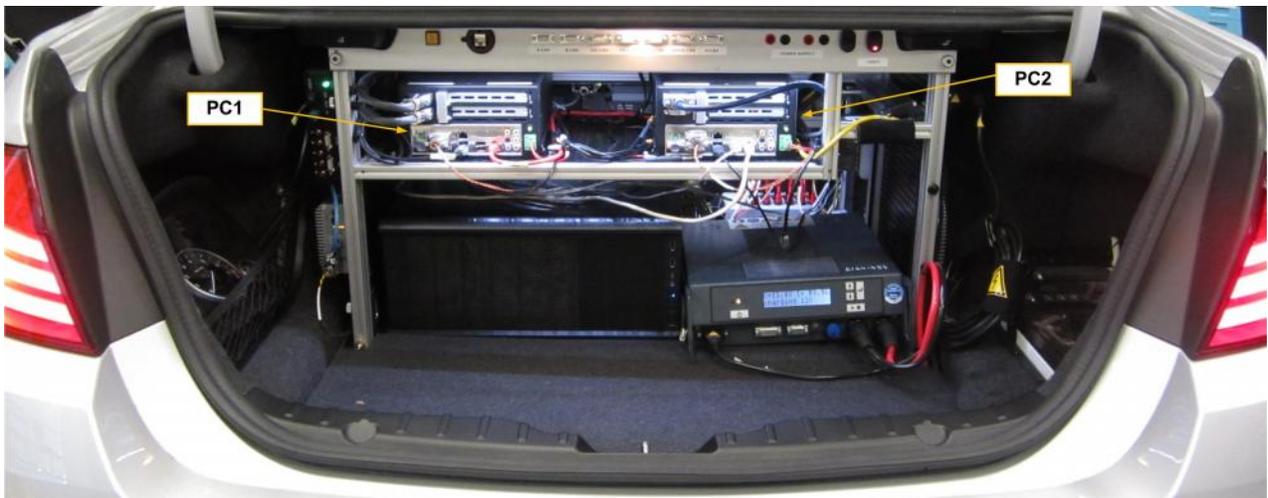
## 2.3 Vehicle setup

### 2.3.1 Research Vehicle



**Figure 7: Development Vehicle**

A BMW 5 series (F10) is used as the research vehicle. To facilitate an efficient communication between the car and the BMW server, the car is equipped with a 4G LTE Stick, which is connected to a router. In the trunk of the vehicle two PCs are installed (see Figure 8).



**Figure 8: Trunk of development vehicle**

Both PCs are connected to the router over a local network and so have internet access. One PC (SIM-PC1) is connected to the Programmable Instrument Cluster. By means of this PC speed recommendations can be displayed directly in the instrument cluster. The other PC (SIM-PC2) is used to collect the car data (GPS position, velocity, fuel consumption, etc.), to receive traffic signal information and to calculate a speed recommendation based on this data. This PC is also connected to the dashboard display for debugging purposes. See Figure 9 for an overview of the installed hardware of the development vehicle.

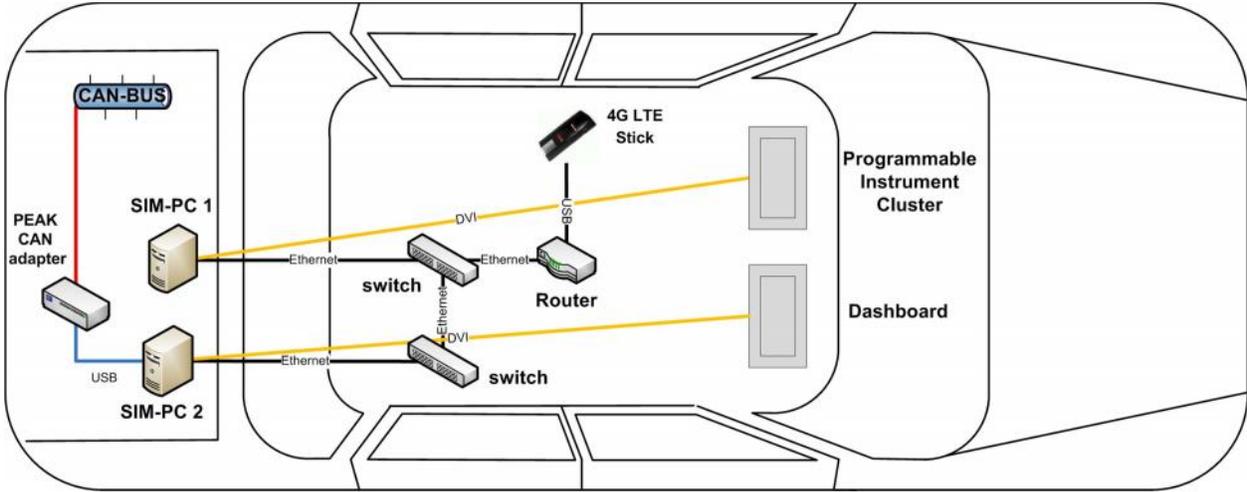


Figure 9: Development vehicle - hardware setup

## 2.3.2 Modules

This chapter describes software modules running in the prototype vehicle.

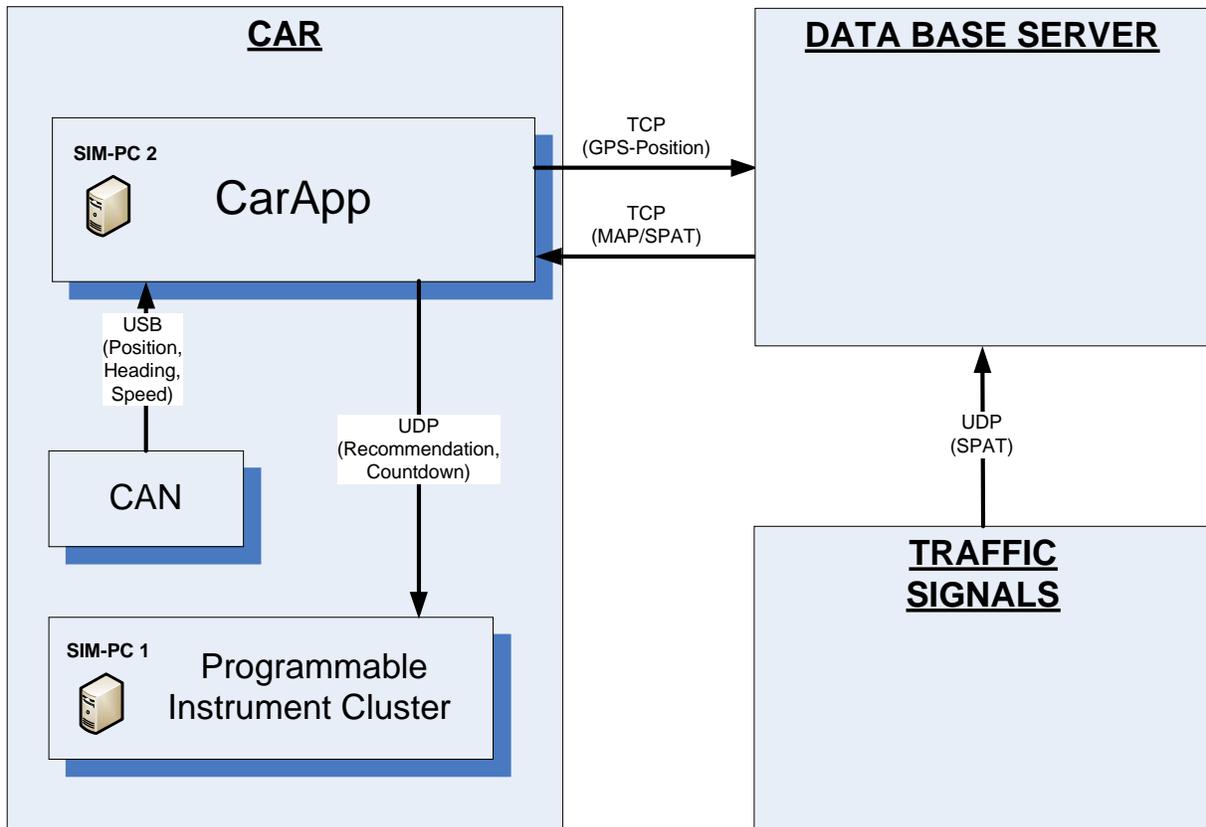
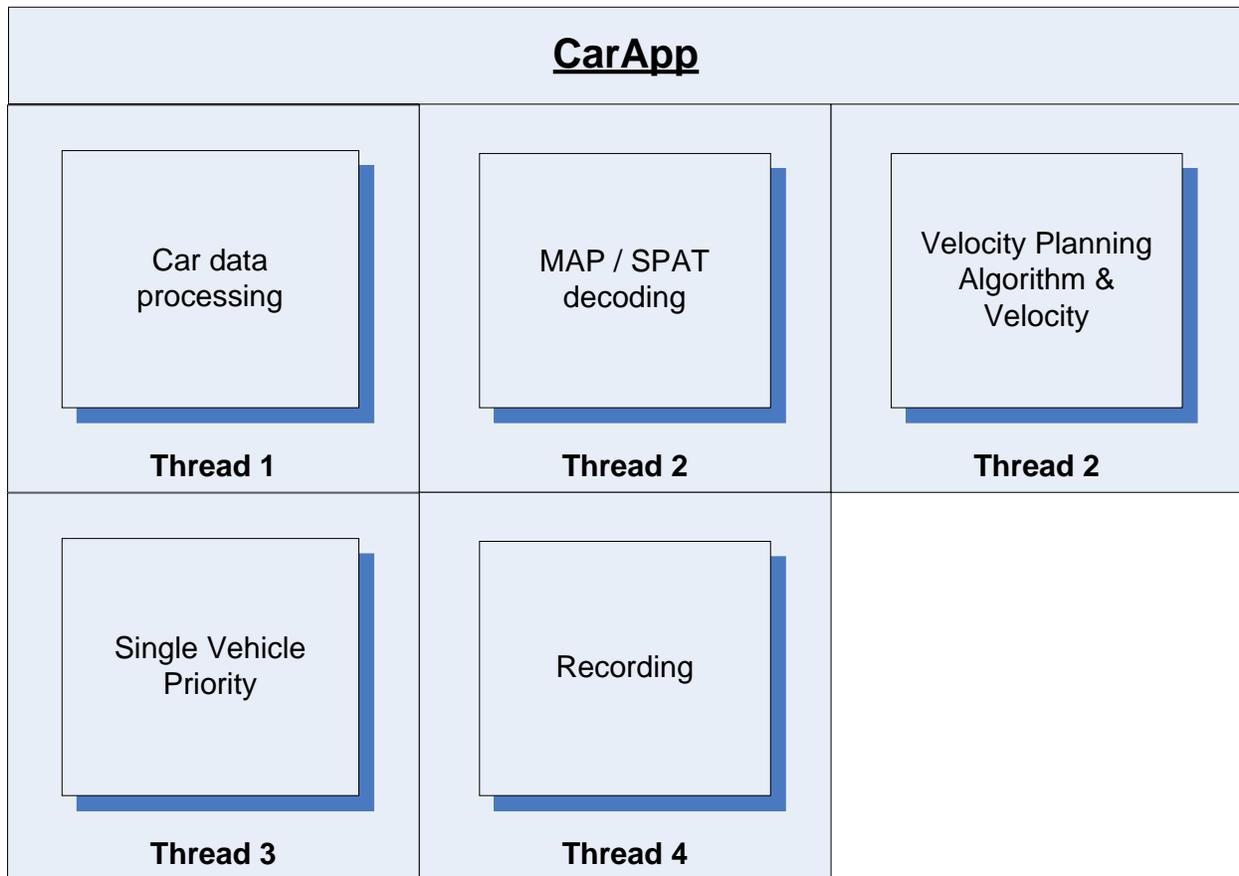


Figure 10: Modules in the vehicle

### 2.3.2.1 CarApp

This is the main module of the prototype which runs in the car. Its tasks are:

- Acquiring GPS data from the car over the gpsd interface
- Processing and sending the current GPS position to the server
- Receiving data from the server (traffic signal information)
- Determining the next upcoming intersections that is being approached
- With the heading angle, determining the direction from which the intersection is approached
- Calculating the distance to this intersection
- Out of all collected traffic signal data, picking the data about the next intersection and build a UDP packet which can be sent to client modules over local broadcast



**Figure 11: Threads in the CarApp**

- **Car data processing**

Decodes and stores data available from CAN-BUS:

- GPS position, Heading
- Speed, Acceleration
- Fuel consumption
- Travelled distance

Position updates are sent to the server.

- **MAP / SPaT decoding**

Receives, decodes and stores MAP and SPaT message received from the server

- **Velocity Planning Algorithm & Visualization**

Determines the next upcoming intersection and the direction of approach to this intersection. Based on the current car and intersection information a speed recommendation is calculated.

See for chapter 3.1 for the fixed-time algorithm and chapter 4.1 for the new algorithm that can handle actuated coordinated traffic signals. The speed recommendation is forwarded to the other PC that displays the speed recommendation in the programmable instrument cluster.

- **Single Vehicle Priority**  
Sends Single Vehicle Priority requests to a server that is connected to a traffic signal. See chapter 3.2.
- **Recording**  
Records car as well as intersection information and writes it into a csv file. System evaluations are based on the recorded data.

### 2.3.2.2 Programmable instrument cluster

This module realizes a **graphical user interface** to present the data sent by CarApp to the driver in an intuitive way. The speed recommendation is visualized directly in the speedometer as a color coded area. Additionally a countdown that displays the remaining timing in the current signal status is shown in the center of the speedometer.

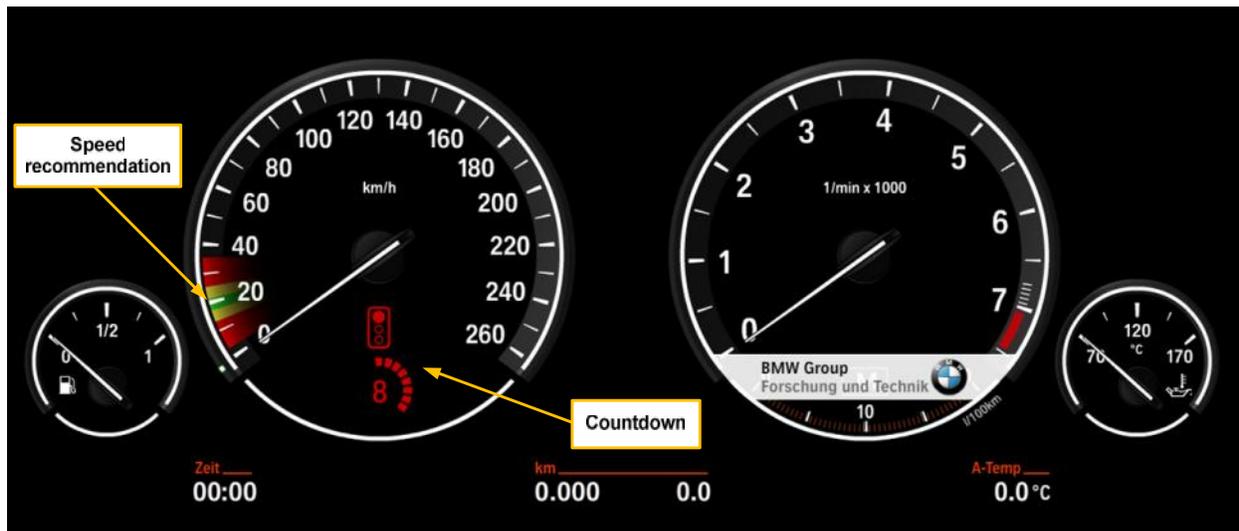


Figure 12: Speed recommendation displayed in programmable instrument cluster

## 2.4 Cloud Server

### 2.4.1 Server Setup

All relevant data about fixed-time and actuated traffic signals are stored in a MySQL database running on a Linux server.

To develop applications for our demands we figured that we have to use a simple socket based packet communication. For smallest possible response times the server running the applications has to be located near the traffic infrastructure.

To achieve this goal the decision was taken to rent a Virtual Private Server, which gives us root access to a machine with a large bandwidth, performance, and UPS. This gives us full root access to an OS of our choice, full user control, the right to install the applications we want, the option to develop any application we want, and a full bandwidth high speed Internet connection.

### 2.4.2 Database Tables

On the server, all information about the traffic signals and information about current clients (cars) are stored in a MySQL database. The most important tables of this database are explained in the following paragraph.

- **cars**

This table contains for each client (car) that is currently connected to the server the following information

- IP                      IP address of one client (car)
- Timestamp            Time of the last update received for this client
- Lat                     Latitude value of the current client position
- Lon                     Longitude value of the current client position

- **MAP**

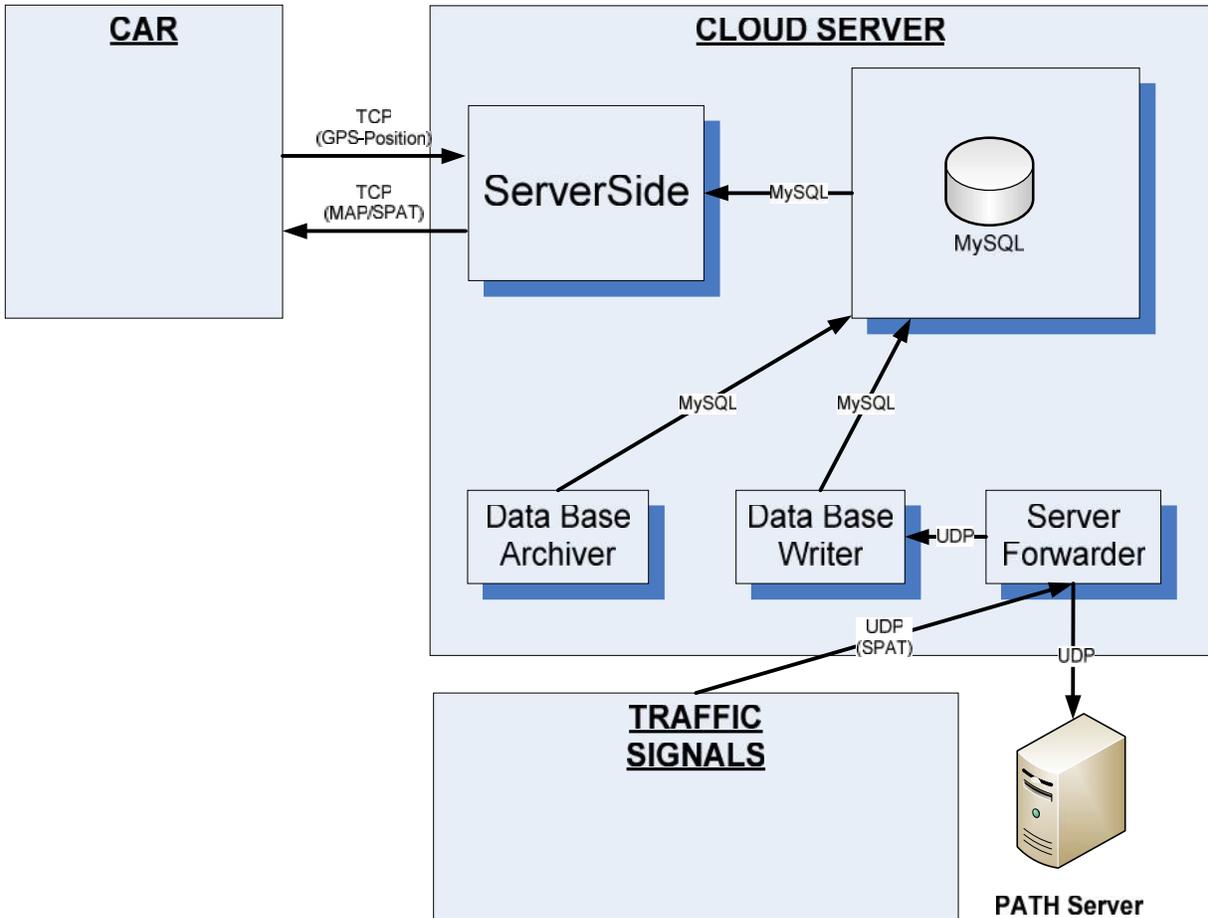
Static information about the traffic signals is stored in this table. It contains information about the position of the signalized intersection as well as information about the different approach and exiting directions that are possible at the intersections.

- **SPAT**

The dynamic information about the traffic signals is stored in this table. It's updated frequently as new timing information is received by the server on a second-by-second basis.

### 2.4.3 Modules

This chapter describes software modules responsible for the transmission and storage of traffic signal information on a data base server.



**Figure 13: Modules on the server**

### **2.4.3.1 Server Forwarder**

This module receives the timing information sent by the traffic signals. The information is sent as SPaT (Signal Phase and Timing) message via an UDP connection. All received messages are forwarded the module “Data Base Writer” as well as to the PATH server. The latter forwarding allows the PATH research group further statistical analysis on the SPaT messages.

### **2.4.3.2 Data Base Writer**

The Data Base Writer module is responsible to save the SPaT messages received from the Server Forwarder into the MySQL database.

### **2.4.3.3 Data Base Archiver**

The Data Base Archiver module was developed to keep the size of the tables in the MySQL small to speed up the query process. For that reason, this module is executed once a day to archive all the data of the previous day.

#### 2.4.3.4 ServerSide

This module is the main module on the server. It handles connections to the cars and sends the current traffic signal data of nearby intersections to them. This module is divided into two different sub modules: ServerReceiver and ServerSender.

##### 2.4.3.4.1 ServerReceiver

This sub module is responsible to receive data from cars (clients).

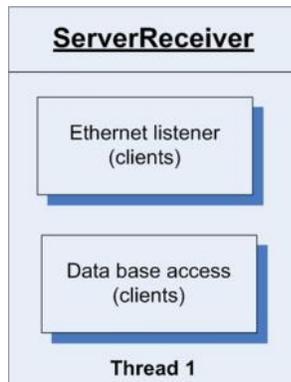


Figure 14: ServerSide sub module: Server Receiver

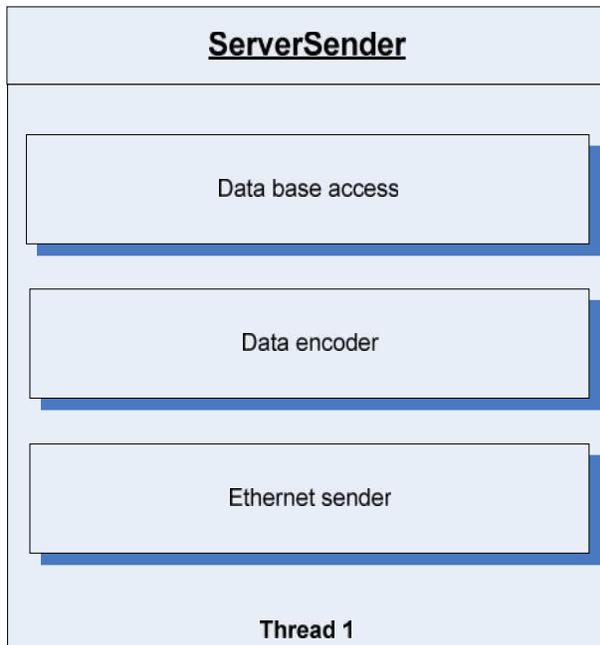
- **Ethernet listener**  
Receives real time data from clients (at the moment these is the current GPS position of the cars).
- **Data base access**  
Stores client information into the data base.

The car frequently transmits its GPS position to the server (every second, configurable). With those UDP packets, the cars IP address and its position become known and are stored into a table of the 'trafficlights' database table **cars**. Therefore, a time stamp is generated as well.

Clients are uniquely identified by their IP address. If the **cars** table already holds an entry for an IP address which has to be added, this entry is just updated. Otherwise, a new entry will be created.

##### 2.4.3.4.2 ServerSender

The *ServerSender* sub module is responsible to gather data from the database wrap them into UDP packets and send them to a car (client).



**Figure 15: ServerSide sub module: Server Sender**

- **Data base access**  
Relevant geographical and timing information is fetched from the data base server. Only intersections in a defined area around the client are taken into consideration.
- **Data encoder**  
Intersection data about traffic signals is analyzed and encoded into MAP/SPaT messages.
- **Ethernet sender**  
Sends generated MAP/SPaT messages to the client.

To be able to send data to cars – later referred as clients – the IP address of those clients has to be known. Furthermore, since it is undesirable to send data about all known intersections to every single client, a small filtering algorithm is implemented as well. The car frequently transmits its GPS position to the server. By this UDP packet, the cars IP address and of course its position becomes known and is stored into the table 'cars' of the 'traffilight' database. The server is now able to send only relevant information about traffic signals, which are close to the cars position, to the cars IP address.

Also, the 'cars' table contains a time stamp, which shows, when the information has been stored into the database. So, if the information is outdated, the ServerSender module goes to sleep, still continuously checking the database for new clients. Also, outdated databases entries are deleted immediately to prevent sending data to dead clients. Additionally, this automatically allows a car to change its IP address, since it will be handled as a new client then.

If valid client information – i.e. a database entry with a non-outdated time stamp – is available, the ServerSender module determines all relevant intersections that surround the client. Then, data about

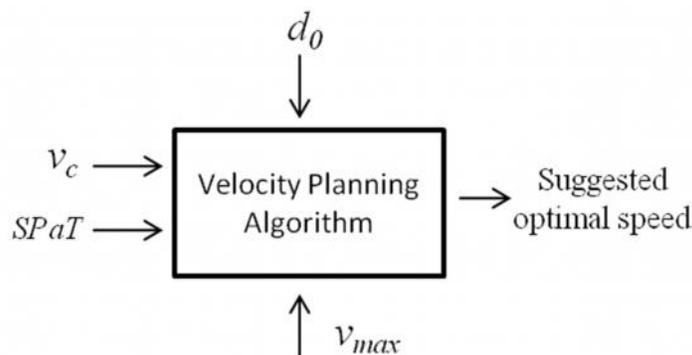
those intersections is acquired from the database and both SPaT and MAP messages are formed and sent with this data. Since there are two different types of traffic signals – fixed and actuated – the algorithms for each type differ.

### 3 Fixed time traffic signals

This chapter presents the algorithm used to calculate the speed recommendation and the mechanism for adapting the timing in the APIV system. The algorithm was developed by the University of California Riverside (UCR).

#### 3.1 Velocity Planning Algorithm

To calculate the speed recommendation a Velocity Planning Algorithm is used.



**Figure 16: Velocity Planning Algorithm**

It uses several input parameters:

$d_0$  the distance from the vehicle to the intersection

$v_{max}$  the maximum speed

$v_c$  the current vehicle velocity

**SPaT** information about signal phase and timing

There are different possibilities for determining the  $v_{max}$ . One is to just use the local street speed limit. Another possibility is to also take into account the velocity of the proceeding vehicle and the safe headway distance or time. For simplicity the current system only uses the local speed limit as maximum speed.

Based on the input parameters the Velocity Planning Algorithm calculates a speed trajectory and from this trajectory a recommended optimal speed is determined. Details about computing an energy

efficient acceleration and deceleration trajectory as well as more details about the algorithm in general can be found in (Xia, et al., 2012). A similar algorithm can be found in (Asadi, 2011).

## 3.2 Adaptive Priority for Individual Vehicle

APIV aims to adjust signal timing to provide additional green time (when applicable) to favorite individual vehicles approaching to a signalized intersection. APIV strategies could reduce the chance that a vehicle has to stop for otherwise a red signal and reduce the waiting time when a stop is inevitable.

Two priority strategies were implemented: 1) extends the green phase for the individual vehicle until it clears the intersection (GE = Green Extension), or 2) advances the start-of-green on the vehicle approach (EG = Early Green).

More details about the algorithm can be found in (Zhou).

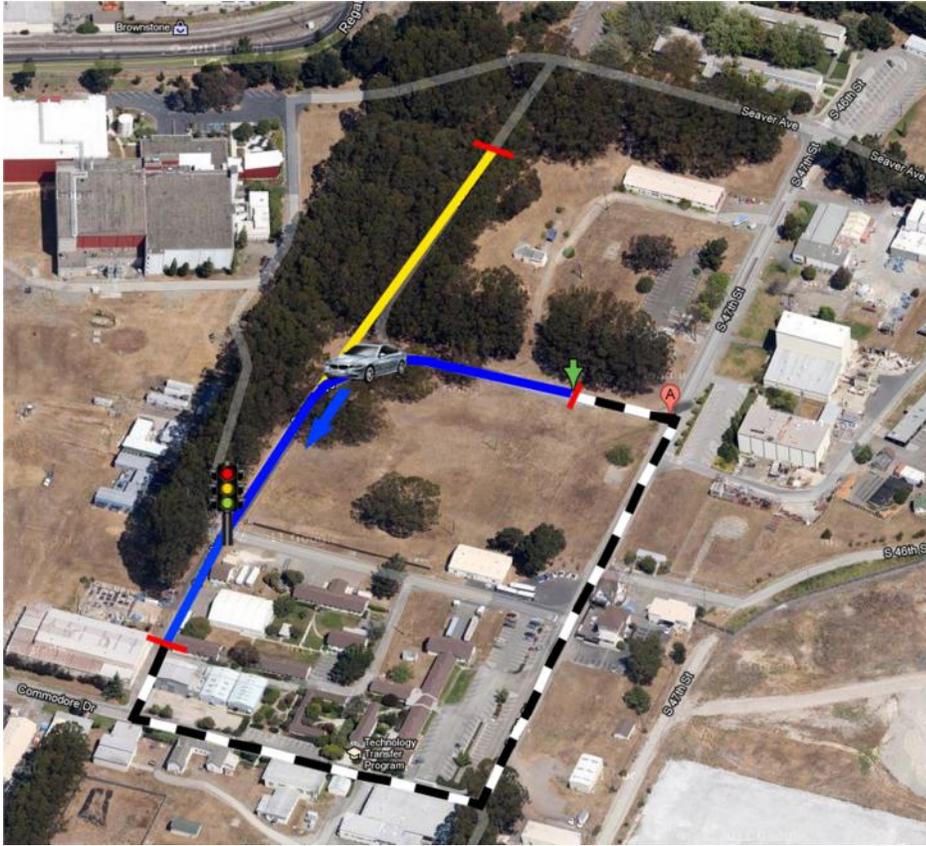
## 3.3 Richmond Field Station test field

The test site “Richmond Field Station” provides a controllable test environment. The absence of cross-traffic or other vehicles on the test track results in the opportunity to test different scenarios under exactly the same conditions. All tests carried out at Richmond Field Station can be repeated easily.

### 3.3.1 Setup

The map of the test intersection at Richmond Field Station is shown in Figure 17. Blue line shows the test segment used in this experiment. The ideal selection of the test segment is the straight yellow line. However, the driveway marked by the yellow line is not mapped in the digital map of the vehicle’s embedded navigation system. As the test setup uses map matched locations only, the yellow driveway was unusable for testing. Therefore, the blue line was chosen as the test segment. The total distance of the entire test “loop” is 307 meters. In each run, we started from the green arrow, passed through the intersection, and then exited the test segment at the red bar downstream, as shown in Figure 17. The eco-approach technology equipped vehicle would travel around the test loop, at a nominal speed of around 41 kph (25 mph), corresponding to the speed limit in the area.

The traffic signal controller was set up with fixed-cycle signal timing. The signal cycle length was set to be 60 seconds, with 30 seconds of green, 3 seconds of yellow and 27 seconds of red on the test segment approach. Using APIV the maximum allowable priority interval is set to 9 seconds. So the green signal phase can be extended for up to 9 seconds or the start of the green phase can be advanced for up to 9 seconds. The actual priority interval would depend on the needs of the vehicle when approaching the intersection, ranged from 0 to 9 seconds.



**Figure 17: Test track at Richmond Field Station**

### 3.3.2 Scenarios

In order to compare the eco-approach technology with regular driving in terms of fuel consumption, two testing scenarios were carried out. The first scenario is called “Uninformed Driver” and functions as base scenario. In this scenario the driver drives without any speed recommendations or knowledge of the time left in current signal phase. To have a realistic comparison, the driver drives in a reasonably efficient way as much as possible in the uninformed scenario. The second scenario is called “Informed Driver”, where the driver is provided with the recommend speed in the programmable instrument cluster at 1 Hz while driving through the intersection. A total of 292 runs were made for the informed driving scenario and 270 for the uninformed driving scenario.

To measure the impact of the APIV on fuel consumption and travel time two more scenarios were carried out. In the third scenario called “Uninformed Driver & APIV” the driver again drives uninformed but this time the APIV is used to adapt the signal timing. In the last scenario called “Informed Driver & APIV” signal timing is adapted by APIV and the driver is provided with a speed recommendation. A total of 108 runs were made for each of the APIV scenarios.

For each run, the test vehicle entered the test segment randomly in time without knowing the current signal information. Fuel consumption was recorded on a 1 Hz basis, with an accuracy of  $10^{-6}$  liter.

### 3.3.3 Results

Second-by-second fuel consumption were measured during the real-world experiments for each run from the point the test vehicle entered the test segment till it exited at the other end, as the blue line shows in Figure 17. Note that by driving through the blue curve while maintaining a speed, an additional shear force is generated at the wheels that alter the balance of forces and has to be compensated for by the engine. Thus, the engine would consume more fuel. However, this was the case for all four scenarios and thus, the effect on the relative differences between the two scenarios are negligible.

	Uninformed Driver	Informed Driver	Uninformed Driver & APIV	Informed Driver & APIV
<b>Fuel (l/100km)</b>	10.23	8.84	8.28	7.33
<b>Improvement</b>	Base Scenario	- 13.59 %	- 19.06 %	- 28.35 %

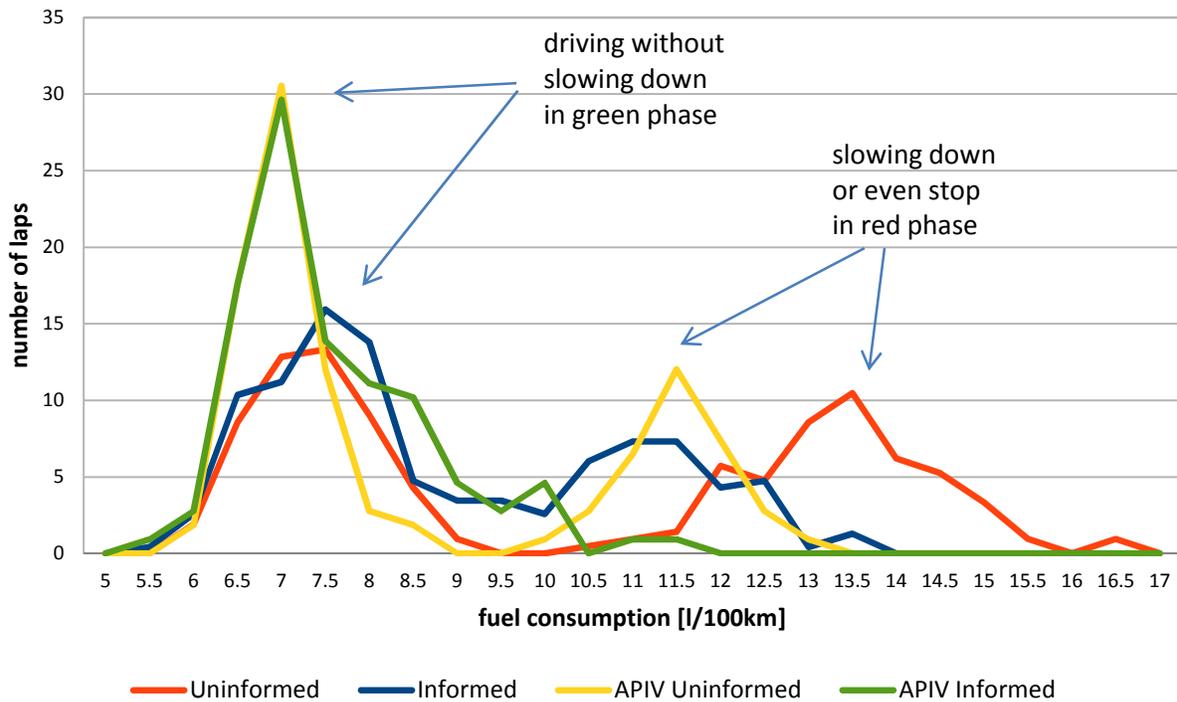
**Table 1: Fuel consumption and savings**

After averaging the fuel consumption results over all the runs for all scenarios, we found that the informed driving gained a 13.59 % of fuel savings compared to the uninformed driving as illustrated in Table 1. A 2<sup>nd</sup> driver was also recorded to cross check the results, showing similar fuel savings compared with the 1st driver. The results shown in Table 1 are an average of both drivers.

The uninformed driving with APIV gained 19.06% of fuel savings compared to the base scenario. The combination of both, APIV and informed driving gained even better results. For this scenario fuel savings of 28.35% were observed.

Further exploring the data of fuel consumption provides more insight on where the fuel was saved. A frequency distribution of fuel economy for each test loop is shown in Figure 18. In this figure, an initial peak can be seen in the range of 5 to 9 l/100km, corresponding to the scenario where the driver didn't need to slow down since the vehicle can pass the intersection while the signal is green. This is approximately the same for both the informed and uninformed driving cases. For the APIV cases this peak is considerably higher. This corresponds to a higher proportion of green drives and results from the fact that there was no cross-traffic and so APIV was able to give priority green (up to 9 seconds) in many cases.

A second set of peaks is observed in the range of 10 to 15 l/100km, corresponding to the case when slowing down is inevitable due to a red light. In this scenario, the informed driver is able to coast down and consume less fuel than the uninformed driver. So the second peak is flattened (less full stops) and moved to the left (less waiting time). For the APIV Uninformed Scenario the shape of the second peak is almost the same as in the base scenario, but it moved to the left. That indicates that the behavior of the driver stayed the same, but on average the waiting time after a full stop is reduced. In the last scenario that combines APIV and Informed Driving the second peak is dwindling small since the driver doesn't have to do a full stop in almost all cases.



**Figure 18: Frequency distribution of fuel consumption over test loops**

Figure 19 gives more insights about how “Informed Driving” influences the driving behavior. It shows the average speed (in m/s) during the test lap. The intersection is located at around 75% of the whole trip. It is shown in the figure that the informed driver tends to slow down earlier when approaching the intersection, and has a higher average speed to pass the intersection, compared to the uninformed driver. Additionally, the curve is much smoother than in the uninformed scenario. The APIV Uninformed scenario is similar to the base scenario again. The shape of the curve stays almost the same but moved upward. That’s because with APIV, the driver sees a green signal more often, but in case of a red signal he has the same inefficient driving behavior. Combining both APIV and Informed Driving is beneficial as this scenario results in a smooth energy saving speed curve at a high speed level. Figure 20 and Figure 21 are based on the same data, but the drives have been split up into either “case green” drives or “case red” drives. Figure 20 shows only “case green” drives, where the driver hadn’t had to decelerate. Here the driver drives with speed limit in all scenarios. The different percentage of green drives between APIV and no APIV drives isn’t visible anymore. Figure 21 shows only drives where the driver had to decelerate because of a red signal – “case red” drives. As there are no “green drives” included in this graph, the influence on the driving behavior is visible more clearly. Now the speed profiles for the uninformed driver and the uninformed driver with APIV are almost the same and show the same inefficient behavior of the driver. In contrast, the speed profiles for the informed driving are much smoother and more efficient.

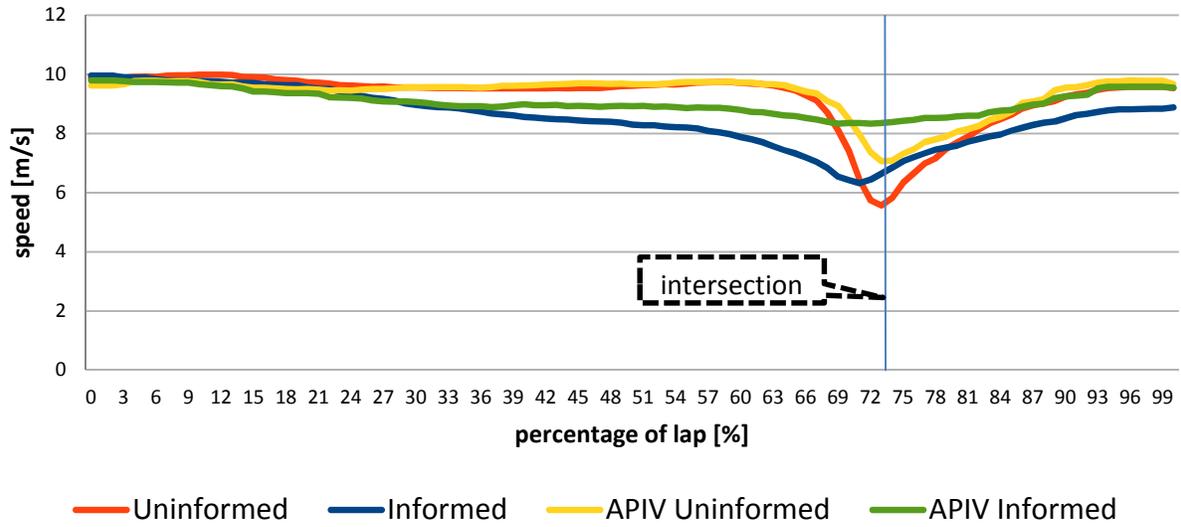


Figure 19: Average speed during test lap

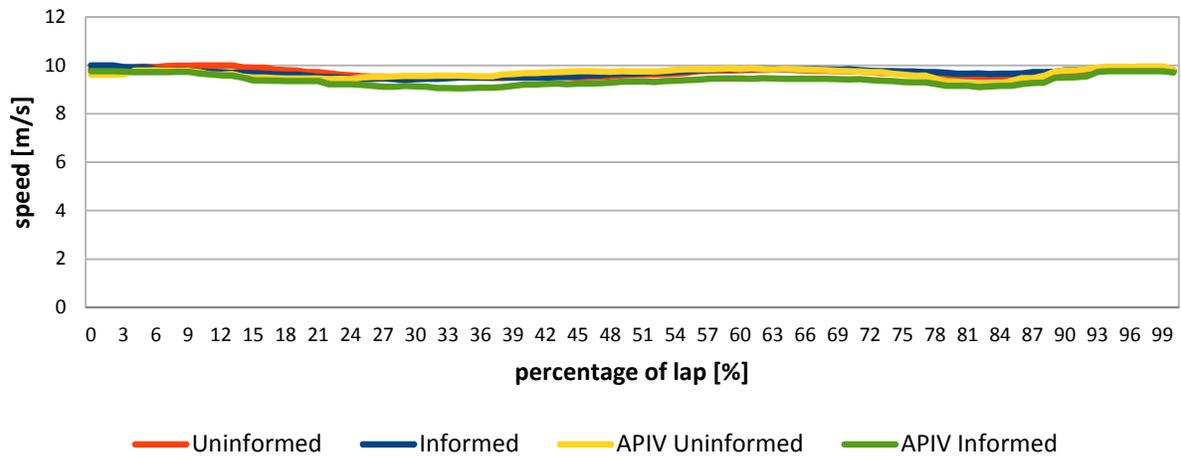


Figure 20: Average speed during test lap - case green

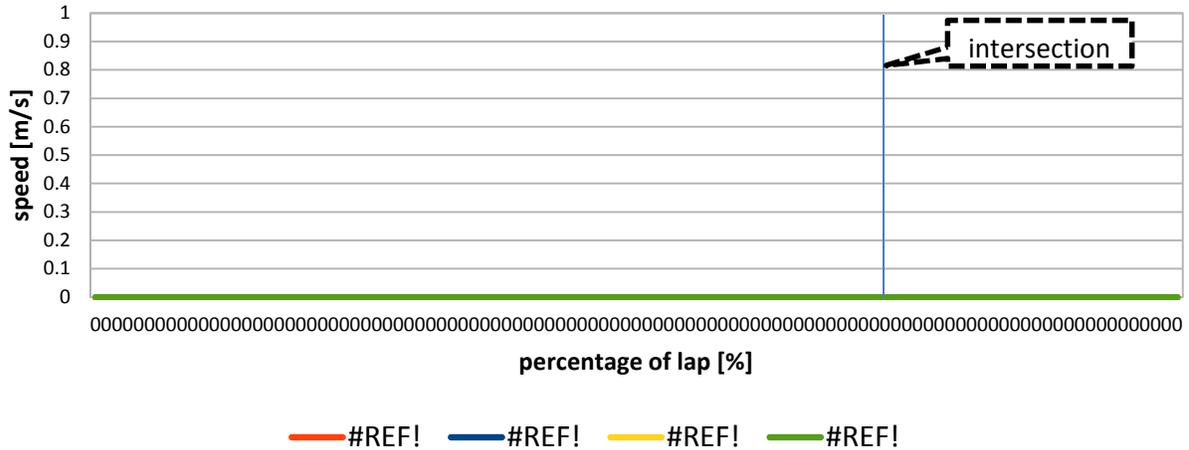


Figure 21: Average speed during test lap - case red

Figure 22 shows the cumulative average fuel consumption during a trip. From this figure, we can find out where during a trip the fuel was saved by comparing cumulative fuel consumption of informed driver and uninformed driver. It's noted that when vehicle is far before intersection, an informed driver uses a little less fuel than an uninformed driver since the informed driver tends to decelerate earlier. As the vehicle is getting close to the intersection, if the current signal is red, the informed driver has more chance to avoid a complete stop and cruises through the intersection at a constant speed. In contrast, the uninformed driver has to stop in most cases. Therefore, an uninformed driver shows higher fuel consumption in this range. After the vehicle passed the intersection, informed driver generally had higher speed than uninformed driver since informed driver tried to avoid full-stop at the intersection and kept a constant speed instead. Thus, informed driver doesn't need as much fuel to accelerate back to speed limit as uninformed driver need. For the APIV cases the curves are lowered as the driver doesn't have to stop so often because of an advanced/extended green phase.

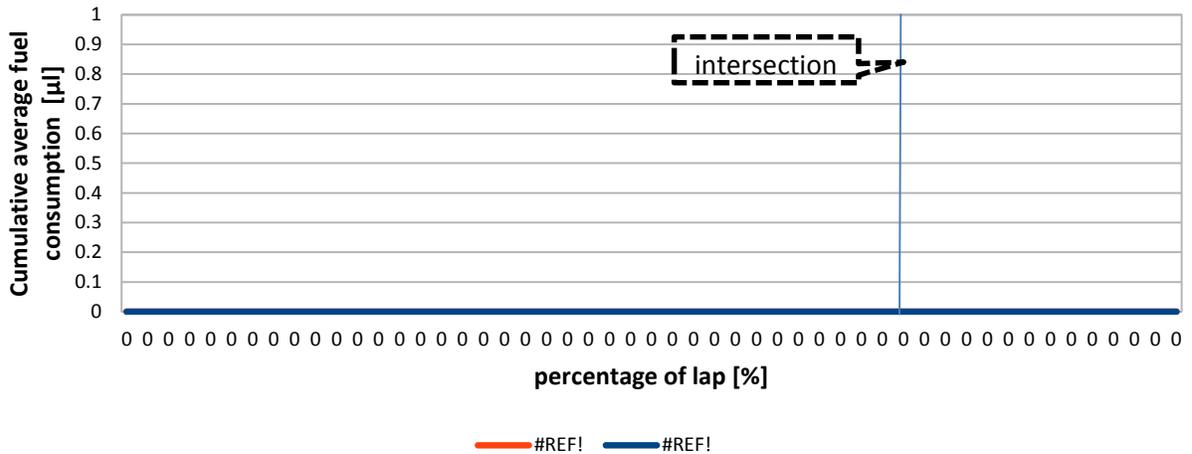
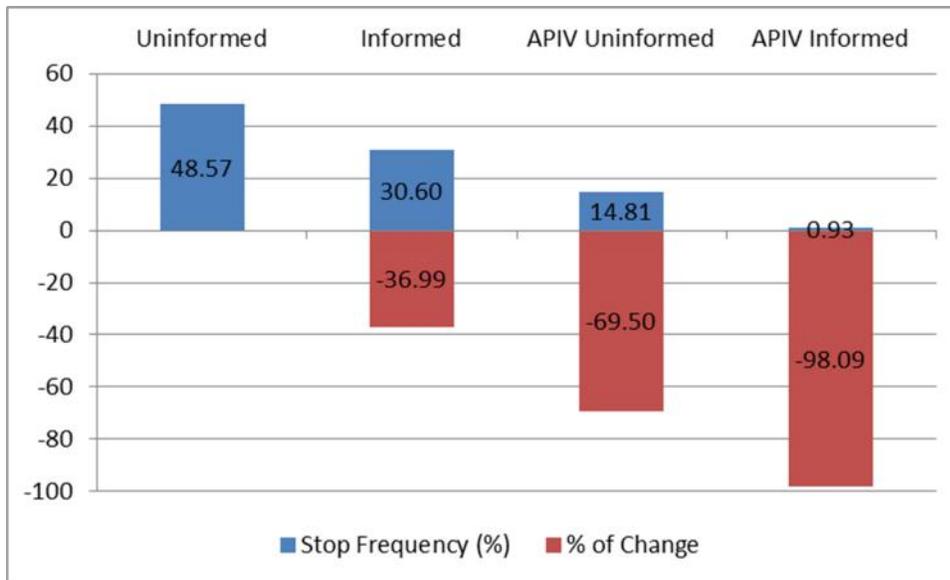
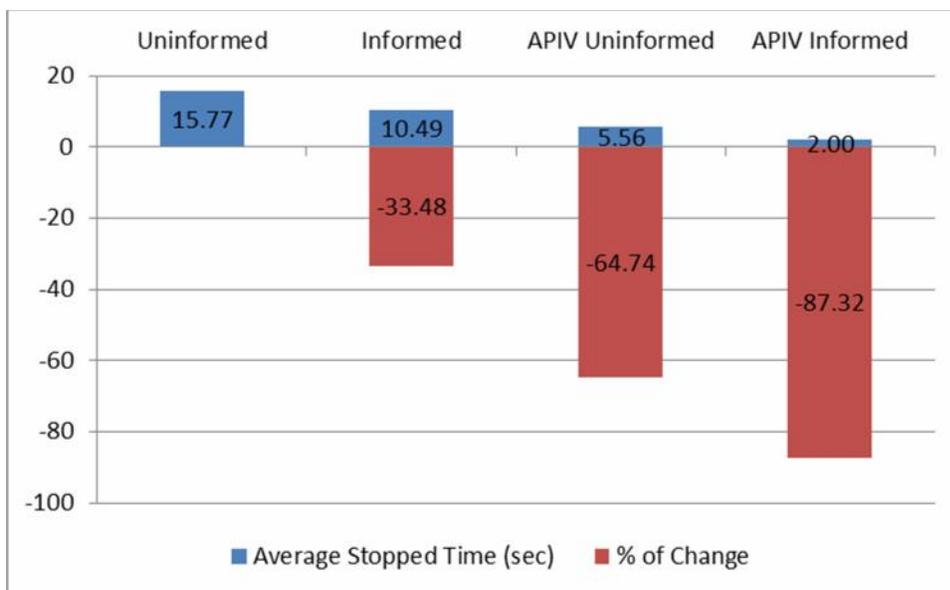


Figure 22: Cumulative averaged fuel consumption

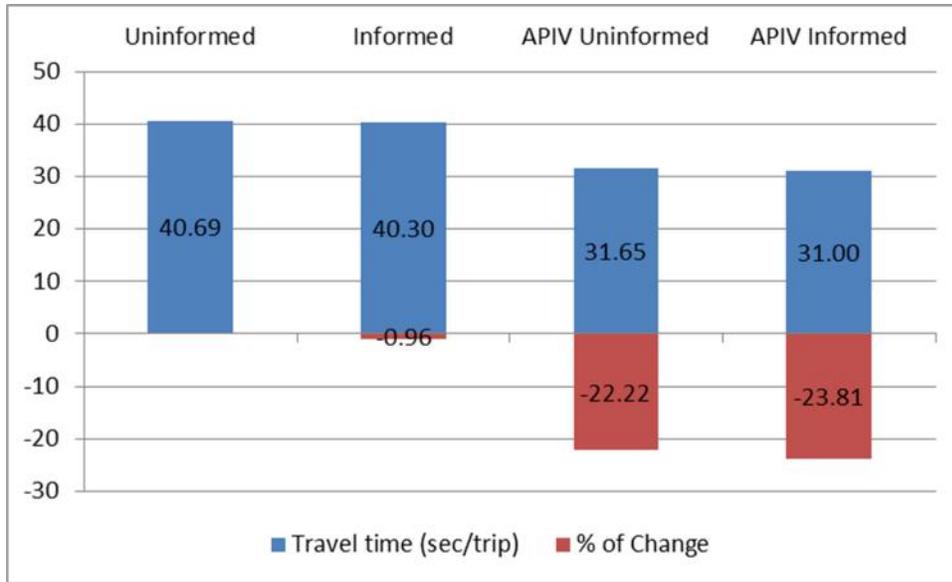


**Figure 23: Stop frequency**



**Figure 24: Average stopped time**

The impact on the stop frequency as well as on the average stopped time is illustrated in Figure 23 and Figure 24. Both, the stop frequency and the average stopped time can be reduced significantly in the informed driving and in the APIV uninformed driving scenarios. For the informed driving this is a result of the early slow down (cruising) that leads to a later arrival at the intersection. In case of using APIV the improvements are the result of an early or extended green. The combination of informed driving and APIV shows remarkable results for the stop frequency and the stopped time. Combining both assistance systems allows the driver to pass the test intersection at green without stopping almost every time.



**Figure 25: Travel time**

Figure 25 shows how informed driving and APIV affect the travel time. For the informed driving scenario there is almost no reduction of travel time measurable. As the driver entered the test lap always at speed limit, it's not possible to get into a green through acceleration. So the informed driver cannot cross the intersection earlier than an uninformed driver. A little improvement in travel time occurs for the case that the informed driver doesn't have to stop at the intersection where an uninformed driver has to stop. Then the reaction to the signal change and the acceleration back to speed limit result in a little longer travel time for the uninformed driver. For the APIV scenarios the travel time is reduced by about 22%. This reduction is a result of the early green or green extension given by the APIV logic.

Table 2 summarizes the results related to stops, travel time and fuel consumption. For all performance measures the best results are obtained by the combination of informed driving and APIV.

	Uninformed	Informed	APIV Uninformed	APIV Informed
Number of Laps	210	232	108	108
Stop Frequency (%)	48.57	30.60	14.81	0.93
% of Change	-	-36.99%	-69.50%	-98.09%
Mean Stopped Time (sec)	15.77	10.49	5.56	2.00

% of Change	-	-33.48%	-64.74%	-87.32%
Travel Time (sec/trip)	40.69	40.30	31.65	31.00
% of Change	-	-0.96%	-22.22%	-23.81%
Fuel (l/100km)	10.2	8.8	8.3	7.3
% of Change	-	-13.59%	-19.06%	-28.35%

Table 2: Summary: Stops, Travel time, Fuel consumption

## 4 Actuated and coordinated traffic signals

### 4.1 New speed advisory algorithm

#### 4.1.1 Basic idea

The new algorithm divides the problem into three separate problems. These three problems result in different steps of the algorithm. In each step the results of the previous step are used to solve current problem (see Figure 26).

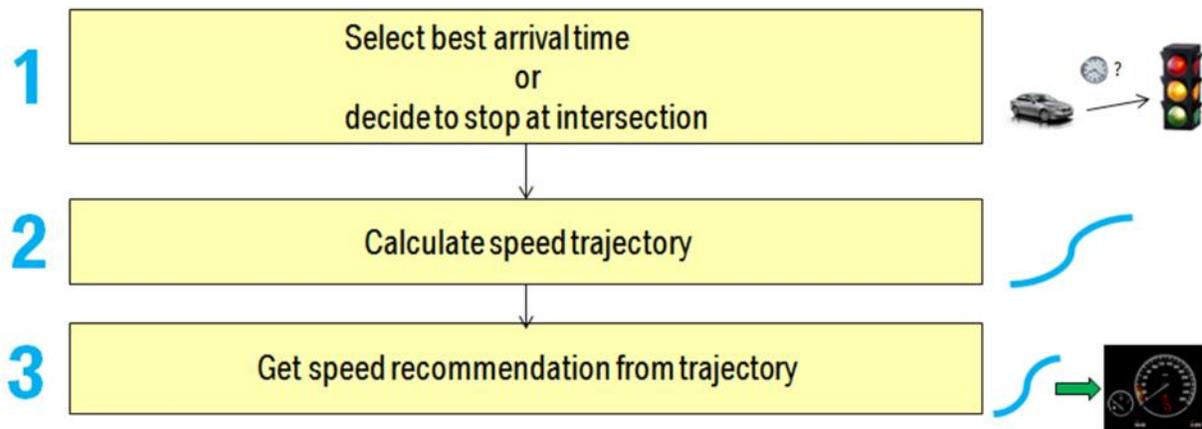


Figure 26: Algorithm overview

The first problem is to determine if it is possible to arrive at the next intersection at green under given constraints: The driver must not travel above the speed limit and the acceleration must not exceed a feasible maximum acceleration/deceleration threshold. Additionally the driver has to travel at least with

an appropriate minimum speed. For the case that it is possible to arrive at the next intersection at green, the time when it best to arrive at the intersection is calculated in this step.

The second problem is to determine a speed trajectory that allows the driver to arrive at exactly that point of time that was calculated in the first step.

Given a speed trajectory, the last problem consists of presenting the driver a speed recommendation that allows to the driver to follow the speed trajectory that was planned in the second step.

Figure 27 gives an overview of how the different steps are connected. Each step is described in detail

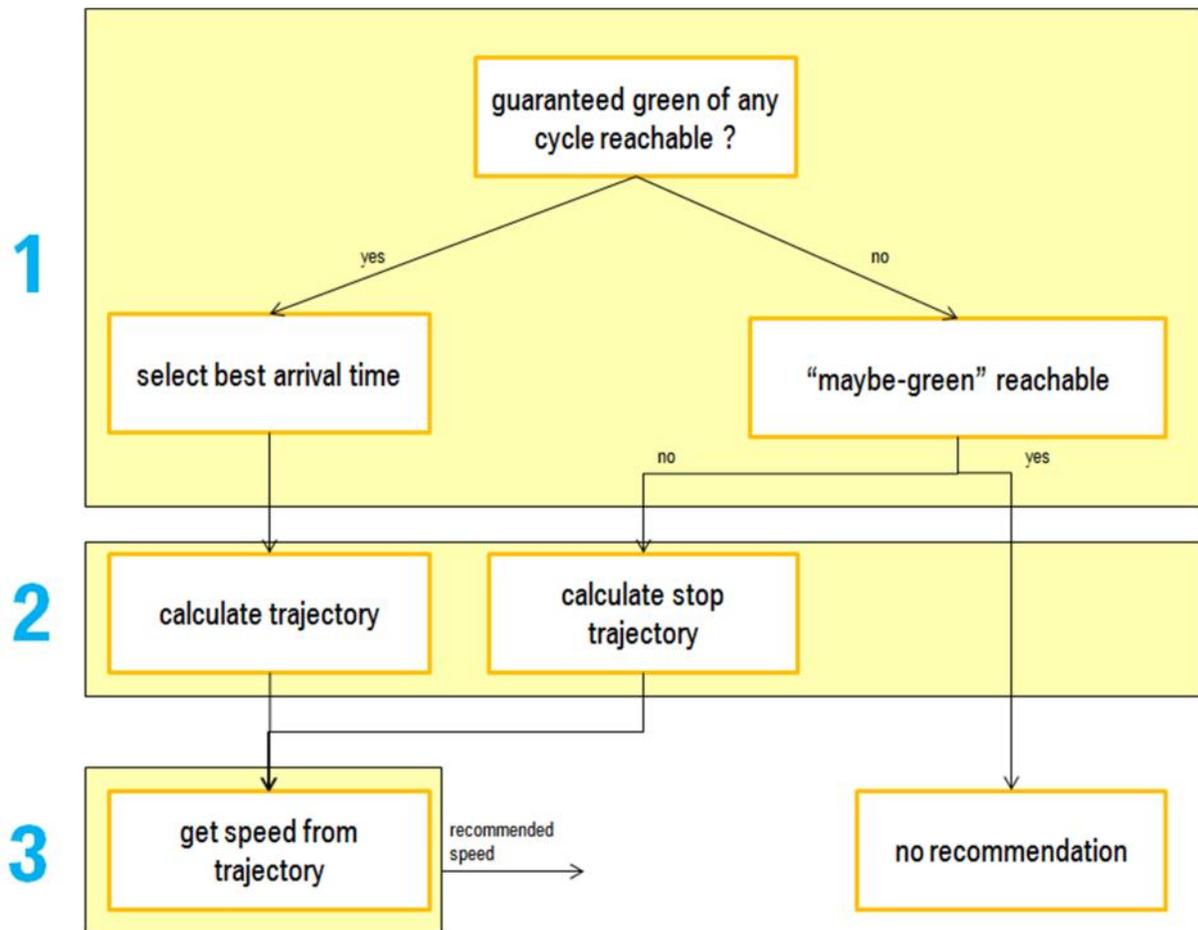


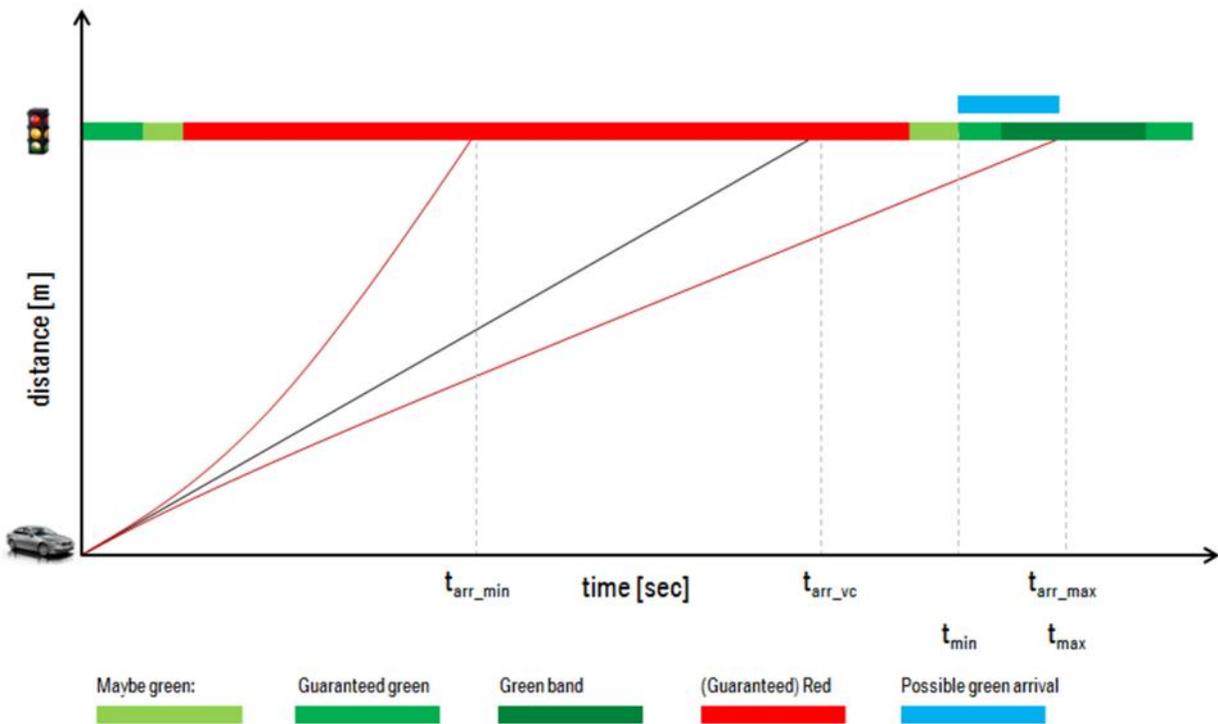
Figure 27: Detailed algorithm overview

#### 4.1.2 Step 1: Select best arrival time

This is the most complex part of the algorithm. That's why it's split up again in different sub steps.

First we determine a time interval within which it is possible for the vehicle to arrive at the next intersection and on the other hand the status of the traffic signal is green at the same time. So after the

first sub step, the result is a time interval where it makes sense to arrive at the intersection. If there is no such interval the driver will have to stop at the intersection. To calculate this interval we first calculate some auxiliary values: The earliest arrival time  $t_{arr\_min}$  and the latest arrival time  $t_{arr\_max}$ . These times describe the earliest and the latest possible moment of an arrival at the intersection, given constraints like maximum/minimum speed, maximum acceleration/deceleration. The interval  $[t_{arr\_min}, t_{arr\_max}]$  is now intersected with all guaranteed green intervals of the next traffic signal. The remaining part  $[t_{min}, t_{max}]$  of  $[t_{arr\_min}, t_{arr\_max}]$  is the desired interval (arrival possible and green status). Since we want to give a reliable recommendation, we don't use the maybe green interval, but the guaranteed green interval. Figure 28 depicts one possible situation and the according result. The red lines mark the speed trajectories that were used to determine the earliest and latest arrival time. The black line marks the trajectory for the case that the speed of the driver stays on the current level. The part of the guaranteed green that is possible to arrive is our desired interval (marked blue).



**Figure 28: Determining the desired possible green arrival interval – normal case**

If several guaranteed green intervals intersect with  $[t_{arr\_min}, t_{arr\_max}]$ , the first guaranteed green that intersects with  $[t_{arr\_min}, t_{arr\_max}]$  is selected (see Figure 29). This assures that the driver can arrive at the destination as early as possible.

For the current cycle, it can occur something that is called “Early green”. It describes a situation where the main arterial signal starts green earlier than the guaranteed green due to less service requests on non-coordinated phases (i.e. somewhere in the maybe green interval just before the start of guaranteed green). As soon as an “Early green” is observed, the remaining time of the maybe green before the start

of guaranteed green is now guaranteed to be green, too. So the desired arrival zone can be extended like shown in Figure 30.

Figure 31 and Figure 32 show examples, where it is not possible to reach any guaranteed green. In these cases, it's not possible to give a speed recommendation with that the driver can arrive at the intersection at green for sure. But for some cases - like in Figure 31 – it is possible to arrive in the maybe green interval. Then, the algorithm gives no recommendation as there is a chance to arrive at green, but it is not sure. If it's even not possible to arrive at maybe green, it's inevitable to stop at the intersection (see Figure 32). This means in step 2 a stop trajectory is calculated for case.

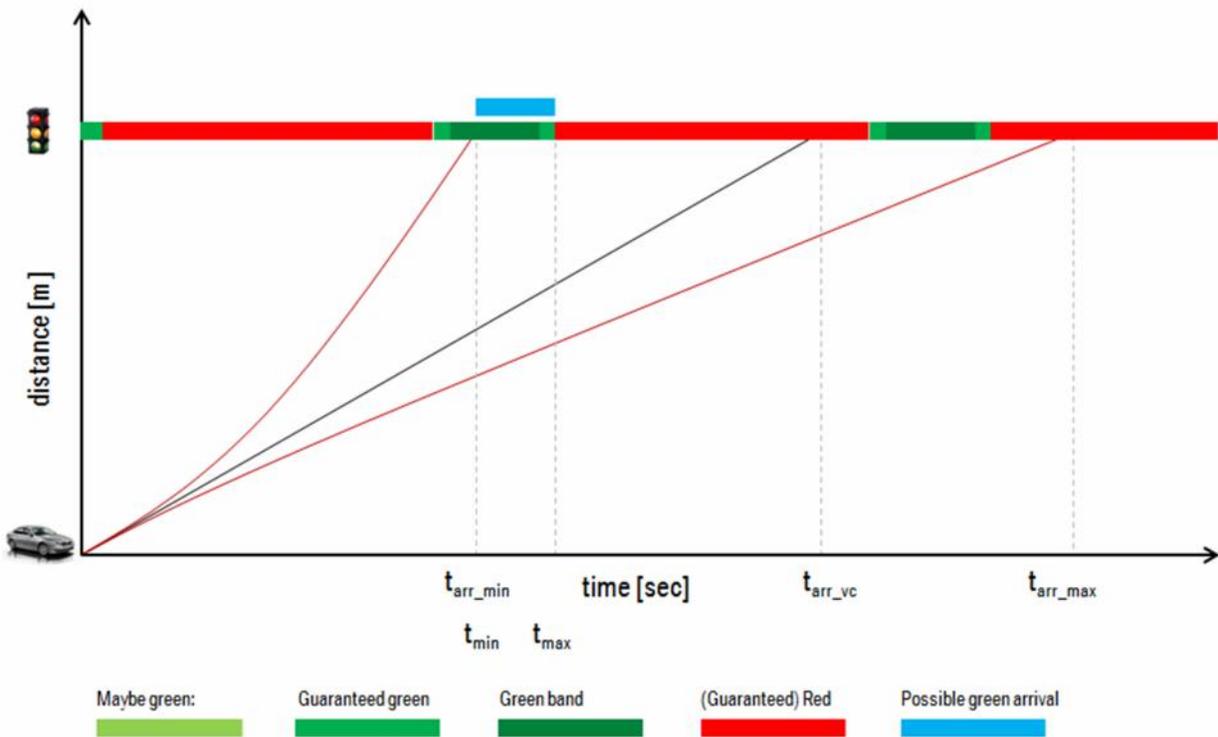


Figure 29: Several guaranteed greens reachable

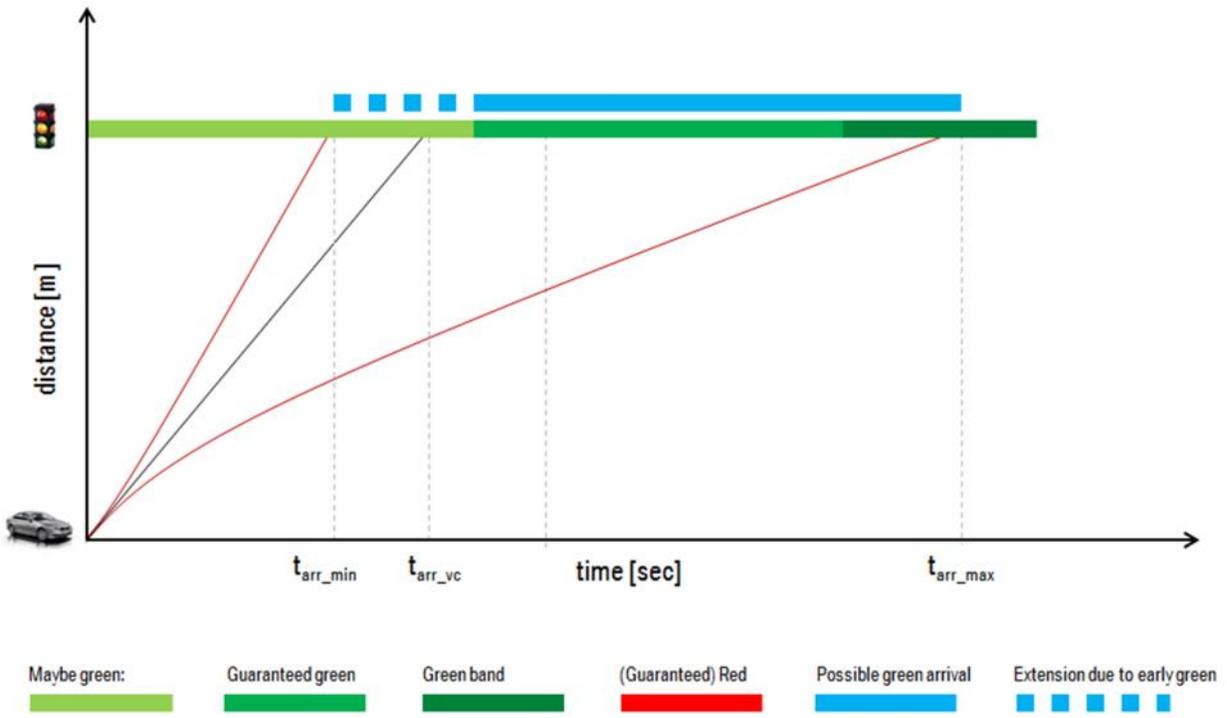


Figure 30: Early green: Include the maybe green

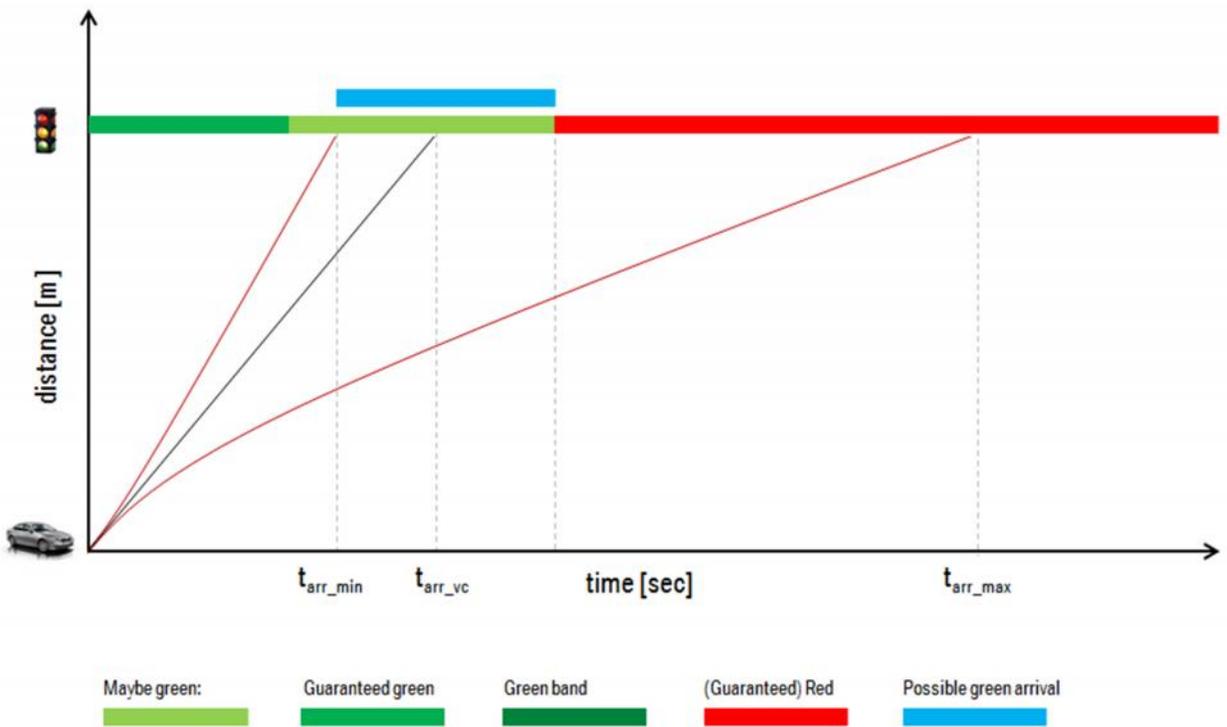
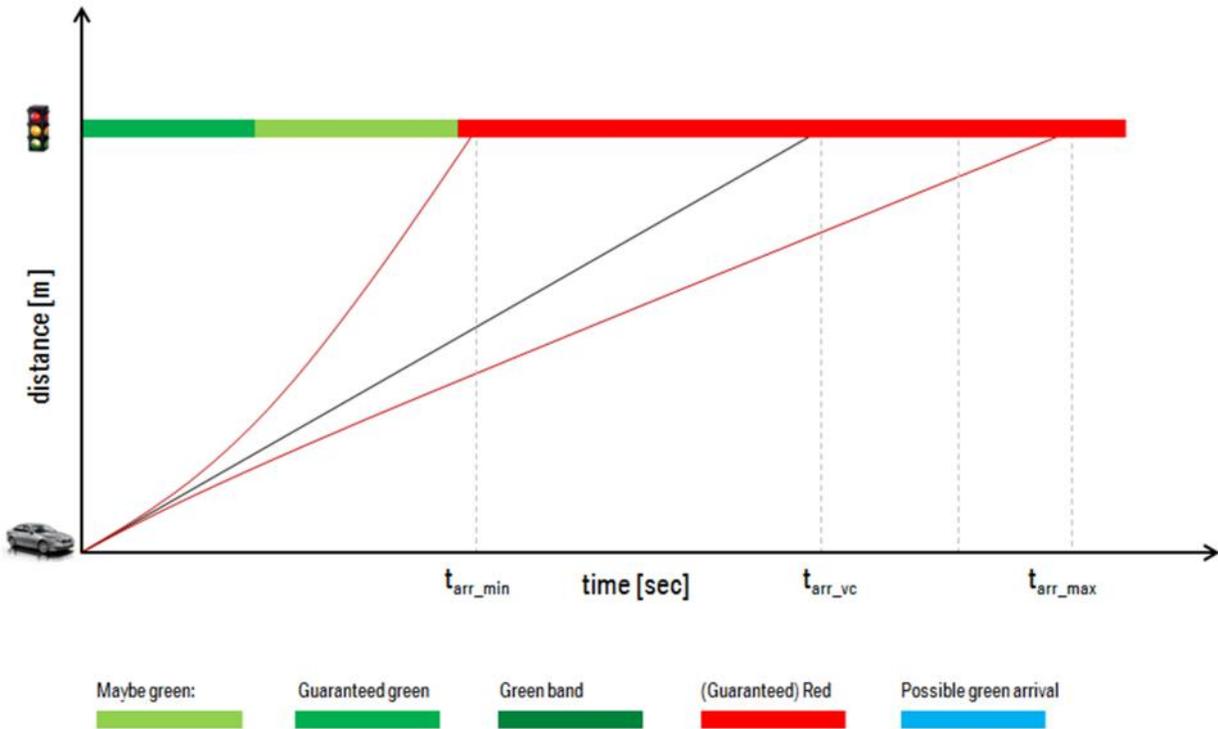


Figure 31: No guaranteed green reachable, but maybe green → No recommendation



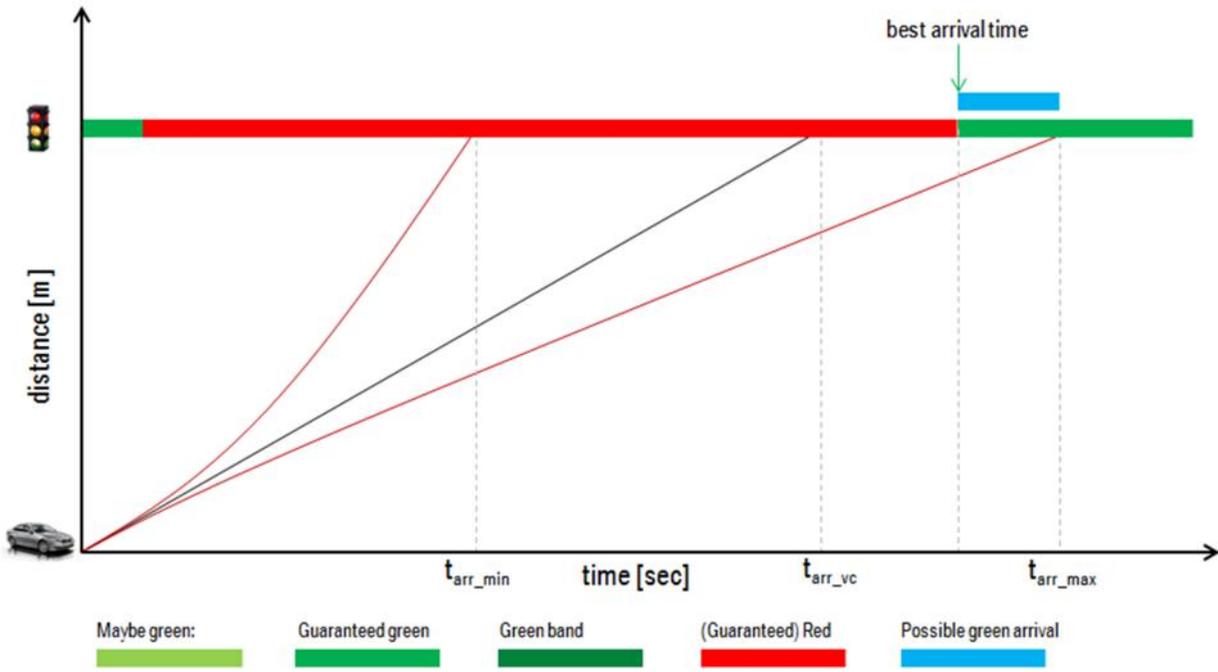
**Figure 32: No green reachable: stopping inevitable**

After having found an interval (marked blue in the previous images) where it makes sense to arrive at the next intersection the next sub step is to determine where it is best to arrive in this interval.

The best arrival time depends on the current velocity of the car as well as on the position of the green band relative to the desired arrival interval. To find the best arrival time we use 3 basic strategies. The strategy depends on the arrival without any recommendation (i.e. just maintaining the current speed):

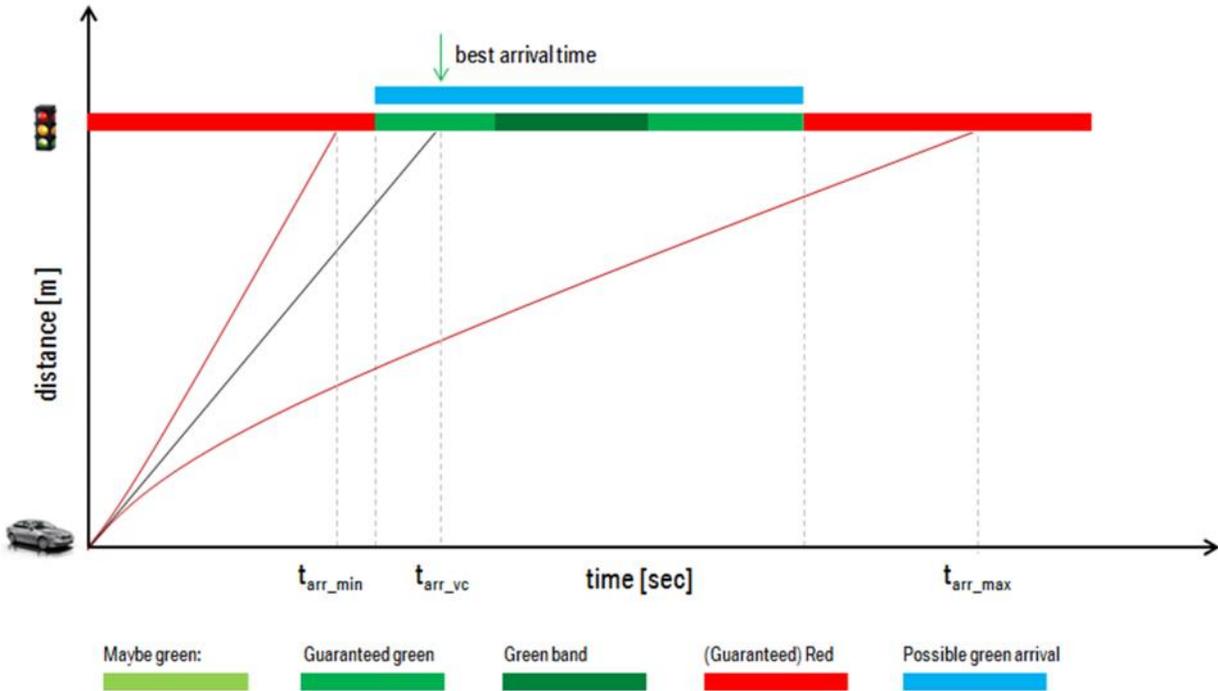
- Arrival before arrival interval:  
→ decelerate to start of arrival interval
- Arrival in arrival interval, before a green band:  
→ maintain speed
- Arrival in or after green band:  
→ Try to get to the start of a green band

The first strategy is somehow obvious. If the driver would continue driving at the current speed, he would arrive at red. Using the strategy he just arrives in the earliest moment where the signal is guaranteed green (see Figure 33).



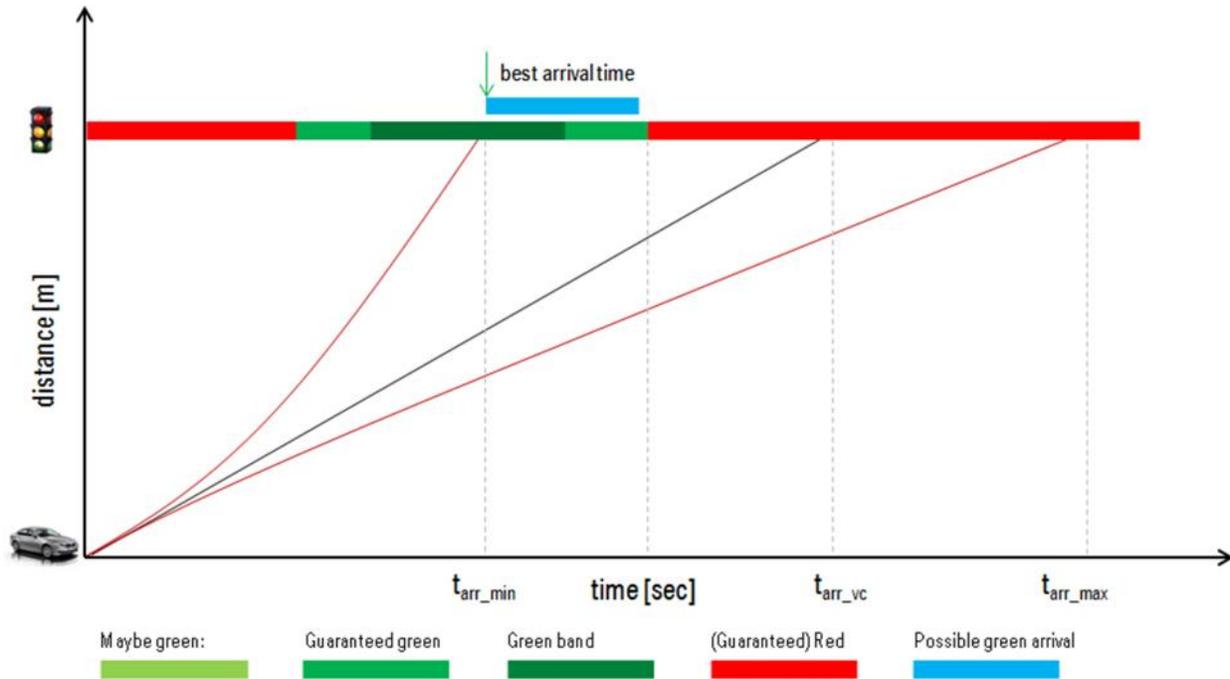
**Figure 33: Arrival before arrival interval → decelerate to start of arrival interval**

Maintaining speed in the second strategy (see Figure 34) is reasonable, as acceleration would make the driver get away from the green band and deceleration would confuse the driver as he is recommended to slow down although he may already see a green light. If it's really necessary to slow down to get into the guaranteed green of the next further downstream traffic signal, that can also be done after passing the upcoming traffic signal.



**Figure 34: Arrival in arrival interval, before green band → maintain speed**

The last strategy tries to let the driver arrive as closely as possible to the start of the green band (see Figure 35). The start of the green band is optimal as it offers a maximal safety margin for unexpected delays (e.g. car in front). In cases that it's not possible to get to the start of green band due to speed limit and acceleration constraints, the best arrival time is selected as the earliest possible arrival time.



**Figure 35: Arrival in or after green band -> try to get to start of green band**

To formalize these basic strategies in a way that is easy to implement them in an algorithm, many different cases of how arrival time  $t_{arr\_vc}$  is positioned relative to possible green arrival interval and the green band are analyzed (see Figure 36). Based on these cases a decision tree was created (see Figure 37). Using this decision tree the best arrival time can easily be determined using the arrival time, the start/end of the arrival interval  $t_{min}/t_{max}$  and the start time of the green band  $t_{band\_st}$ .

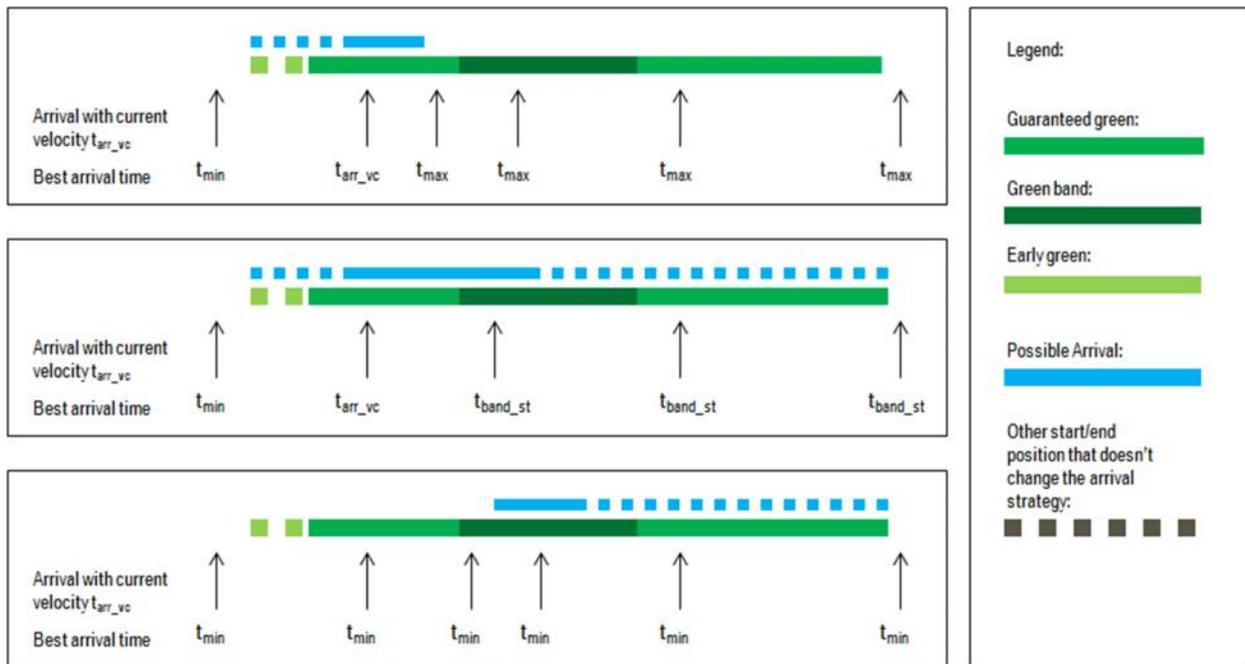


Figure 36: Best arrival time for different arrival scenarios

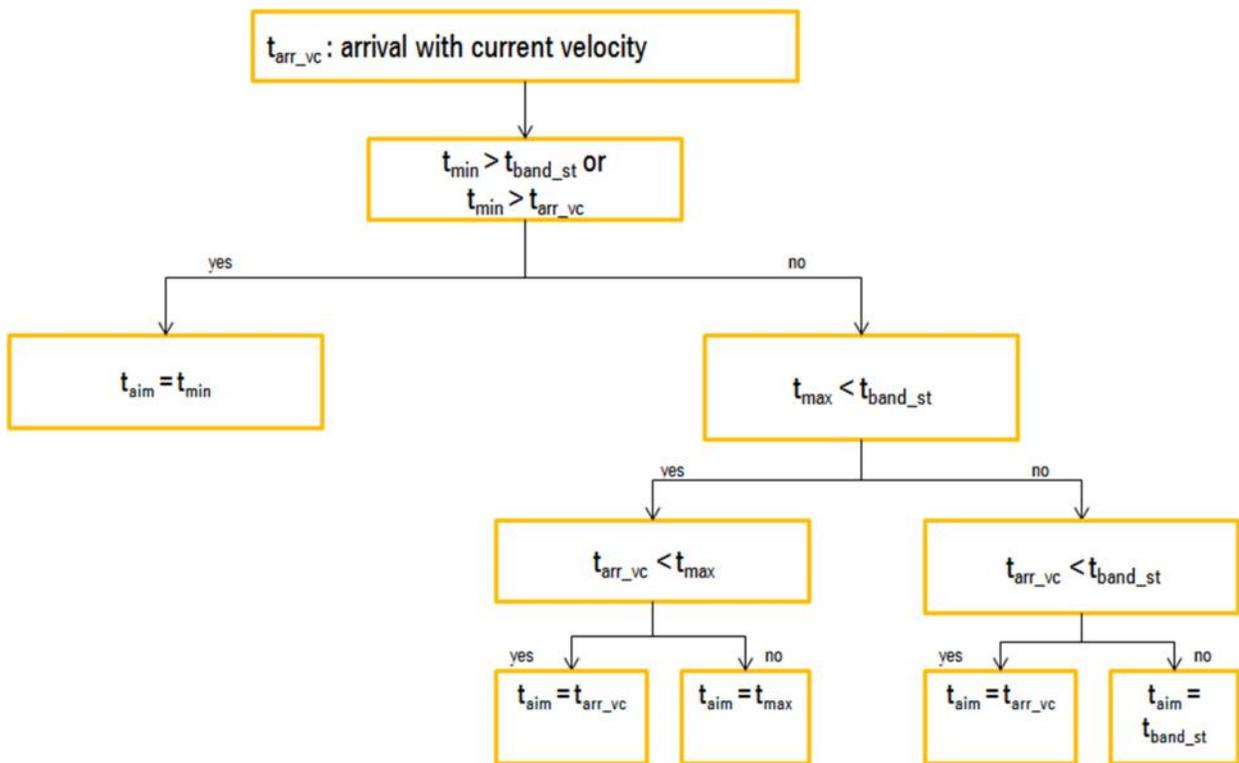


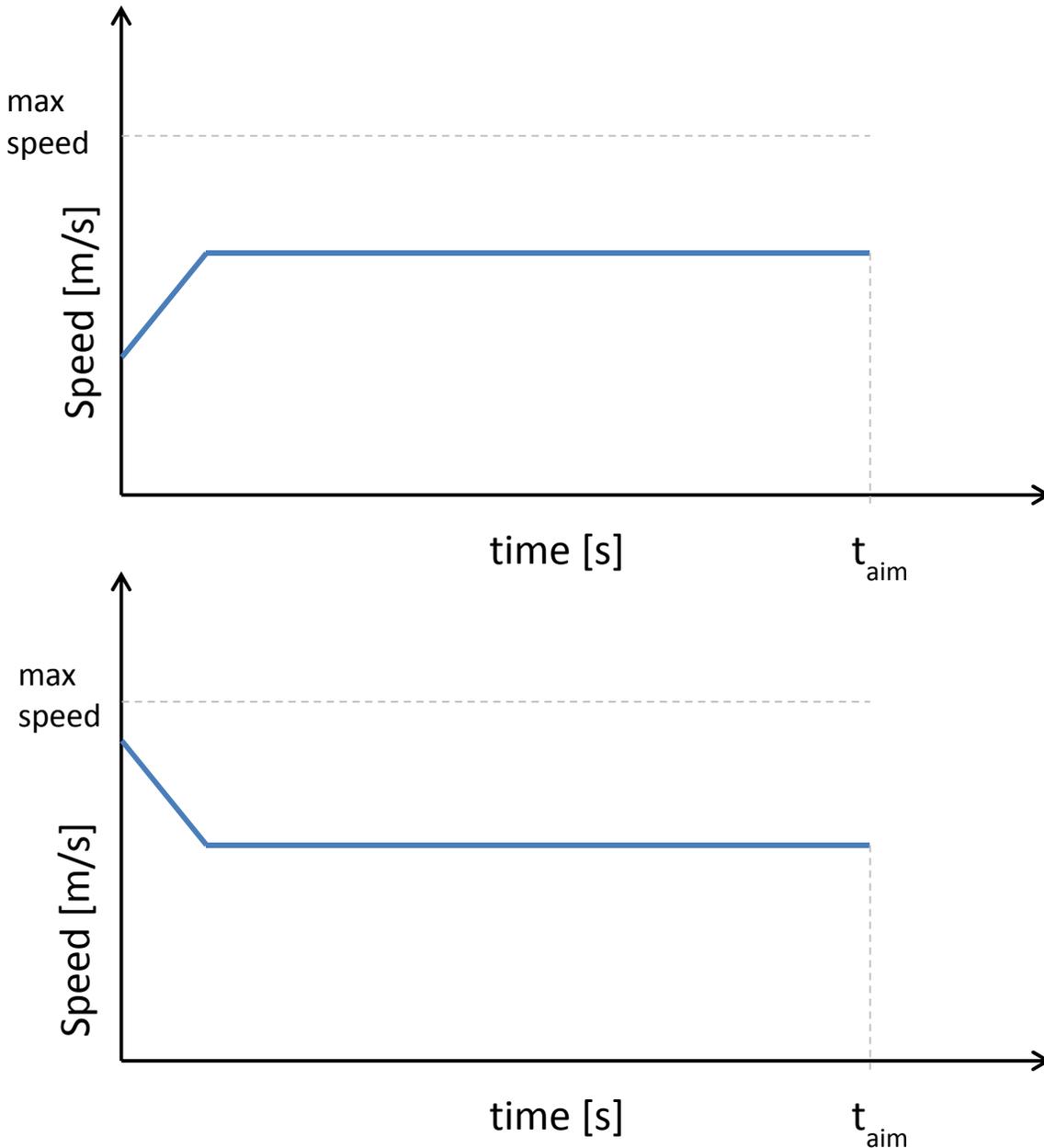
Figure 37: Decision tree used for determining the best arrival time

To sum up, there can be three different outcomes after the first step:

- a) The optimal time to arrive at the next intersection
- b) The decision to give no recommendation
- c) The decision to calculate a stop trajectory in the next step

#### **4.1.3 Step 2: Calculate speed trajectory**

After the first step, either the optimal time to arrive at the next intersection is known or it's known that it is inevitable to stop. Additionally there is the case that no recommendation is given, but then there is no need for calculating a speed trajectory. So the next step is to calculate an according speed trajectory. For simplicity, the speed trajectories use a constant acceleration/deceleration. For future developments a speed trajectory that reflects the engine and gear map of the car is desirable.

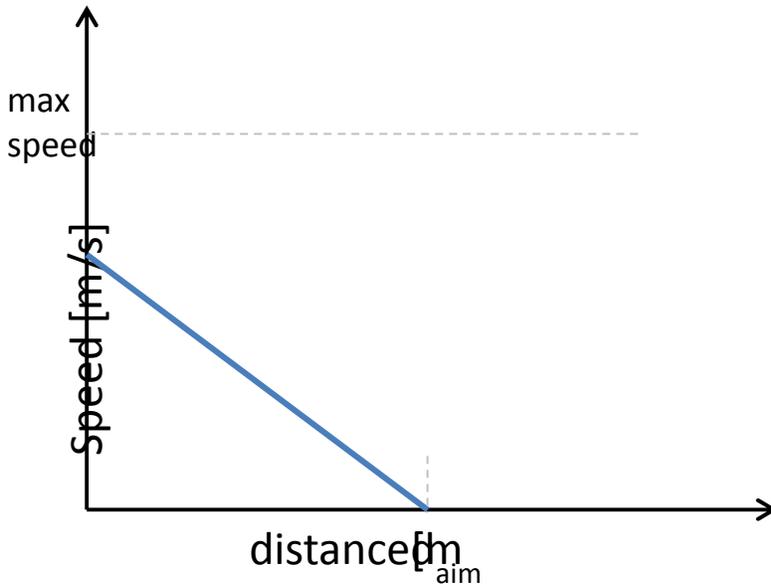


**Figure 38: Constant acceleration/deceleration speed profiles**

Figure 38 shows example profiles that use a constant acceleration/deceleration approach to reach the intersection at the desire time  $t_{aim}$ . Both profiles assume a constant acceleration/deceleration at a predefined feasible maximum acceleration/deceleration.

For the stop trajectory case, no arrival time is specified by the first the algorithm. This leaves some freedom of how to design the stop trajectory. As coasting down without using the brake is more fuel efficient than just braking, the stop trajectories are designed in a way that they (if possible) end with an coast down without braking. For simplicity, it was assumed that deceleration of a car that coasts down  $a_{coast}$  is constant.

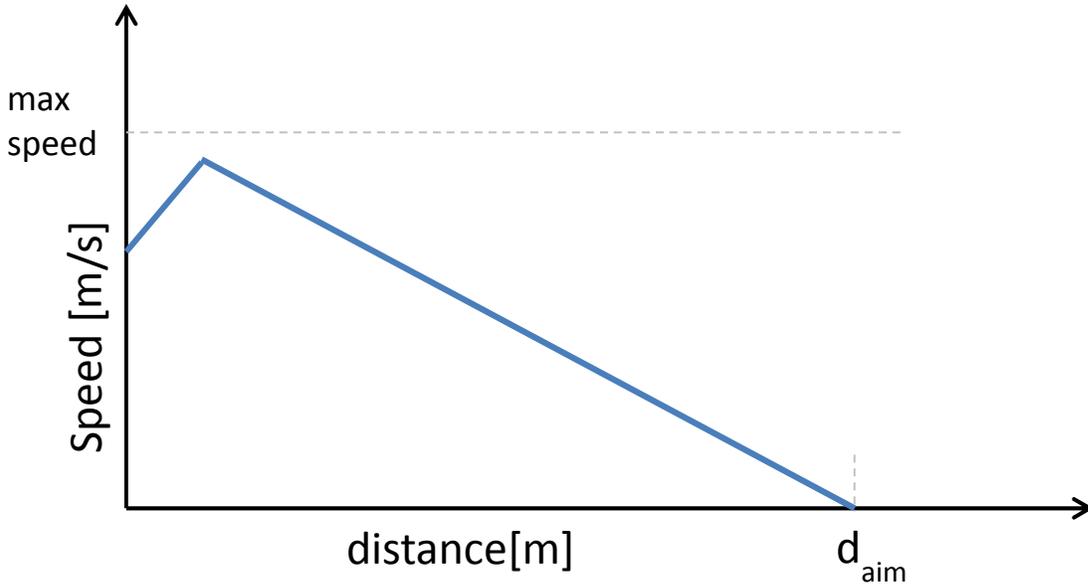
If the car is already so close to the intersection, that the needed deceleration is larger than  $a_{\text{coast}}$ , the speed profile consists of just one constant deceleration that starts immediately (Figure 39).



**Figure 39: Stop profile: immediate deceleration**

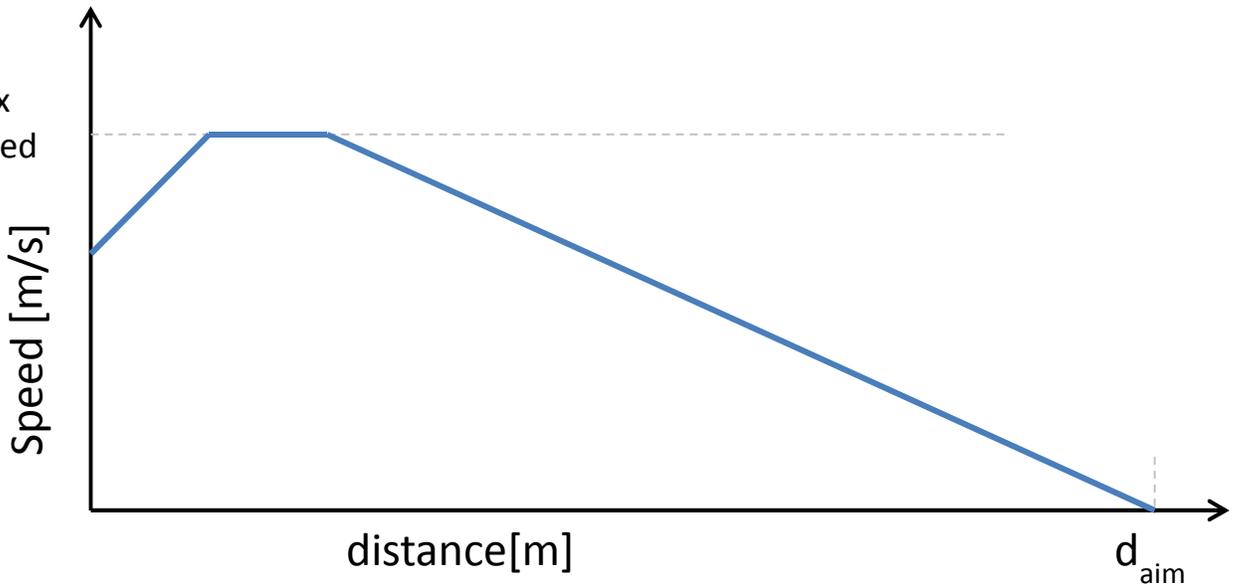
If the driver arrived before the intersection, when he would coast down immediately, the speed profile is adapted in a way that the deceleration doesn't start immediately.

One solution would be to just drive with the current speed until it's possible to coast down with  $a_{\text{coast}}$  and stop exactly at the intersection. This solution is easy but for slow initial speeds this solution doesn't produce the desired result. The driver would then have to drive at slow speeds for a long time.



**Figure 40: Stop profile: acceleration followed by deceleration**

Therefore the stop profile was designed in a way that the driver accelerates at the beginning, as shown in Figure 40. The duration of the acceleration selected so as after the acceleration the driver can decelerate with  $a_{\text{coast}}$  to stop exactly at the intersection. The speed after the acceleration is limited to a maximum speed (speed limit). The stop speed profile can therefore also look like in Figure 41. The acceleration is followed by driving at constant speed and later deceleration with  $a_{\text{coast}}$  to stop at the intersection.



**Figure 41: Stop profile: Acceleration to max speed followed by deceleration**

#### 4.1.4 Step 3: Get speed recommendation from trajectory

After the first two steps, it is known when it is best to arrive at the next intersection. It's also known what speed profile allows the driver to arrive at the next intersection at exactly that time. The last step consists of presenting a speed recommendation to the driver in a way that he can follow the desired speed profile.

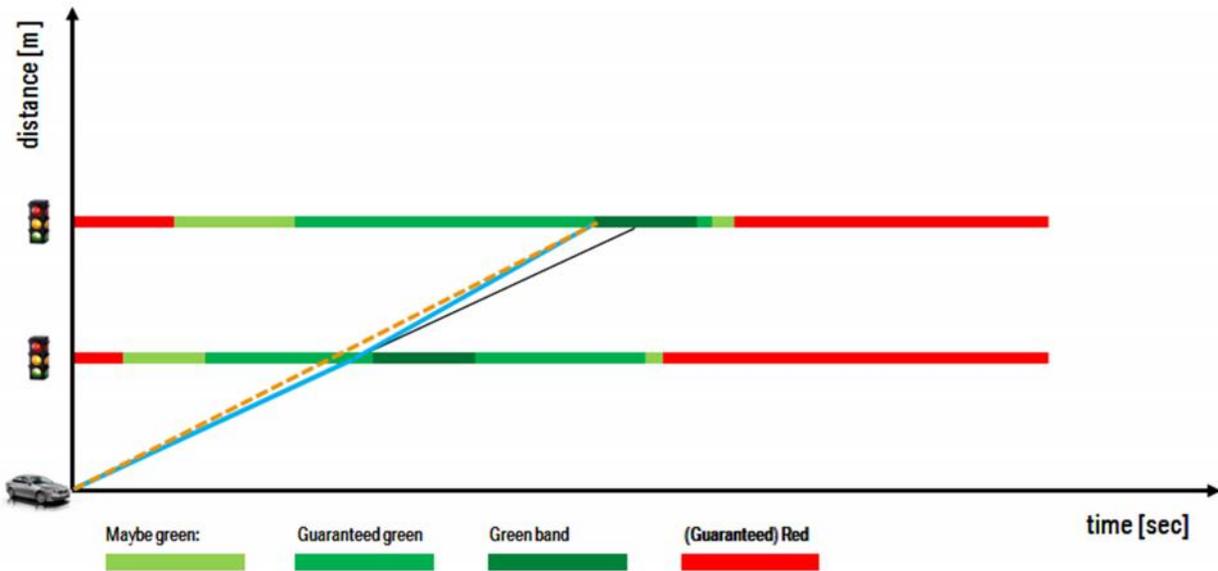
For this project a simple constant forward looking mechanism was implemented to retrieve the displayed recommendation. The speed recommendation that is displayed to the driver is the speed that the driver should have in a predefined time in the future. At the moment we use one second as the forward looking interval.

#### 4.1.5 Shortcomings & ideas for improvements

During our very first test drives on the test site we experienced some shortcomings of the new algorithm and its displayed speed recommendations. This chapter describes them together with ideas for improvements.

The algorithm was designed to make it possible to pass a corridor of green lights. However we only use the information about one traffic signal at a time. For that reason, it can happen that the speed recommendation jumps in the moment when the driver passes the intersection. For some cases this behavior is desired. For example if the next traffic signal is green and the next but one traffic signal is red and we don't have a chance to arrive there at green. In that case first the recommendation would be a derived normal speed trajectory and in the moment where the driver passes the traffic signal the recommendation would be derived from a stop trajectory and therefore the speed recommendation will most probably change.

During one of the test drives we discovered that sometimes the change of the recommendation when passing an intersection is not desired. Figure 42 shows an example. The blue line shows the speed recommendation of the algorithm. As long as the driver is before the first intersection, the speed recommendation is to maintain speed as the driver arrives in guaranteed green before green band. In the moment when the driver passes the first intersection, the algorithm recognizes, that with current speed the driver would arrive in the green band and would therefore recommends the driver to accelerate to get to the start of the green band. For the driver this change to higher speed appears confusing, especially if the driver is able to see the second green light before he/she arrives at the first intersection.



**Figure 42: Speed recommendation changes at intersection**

A better trajectory would therefore possibly be the dashed orange one. With a slight speed up, the driver arrives exactly at the beginning of the green band of the second intersection - without confusing speed changes at the first intersection. For this solution it would be necessary to take information about not only the next but the next two or even more intersections into account as shown in (Asadi, 2011).

Another problem that was observed during the first test is that it is very difficult to follow a speed recommendation that is changing continuously. With some training it is possible to follow it when there is no other traffic, but it's impossible to follow a speed recommendation that is changing continuously when there is other traffic around.

One solution for the normal (no stop) trajectories would be to not display the speed that the driver should have in the next immediate seconds but instead the end speed that the driver should have when passing the intersection. This speed would only change, if the driver doesn't follow the trajectory.

## 4.2 El Camino Real

In order to test the system in a real environment a second test site "El Camino Real" was chosen. In contrast to Richmond Field Station cross-traffic exists and other vehicles are present on the test track on this test site.

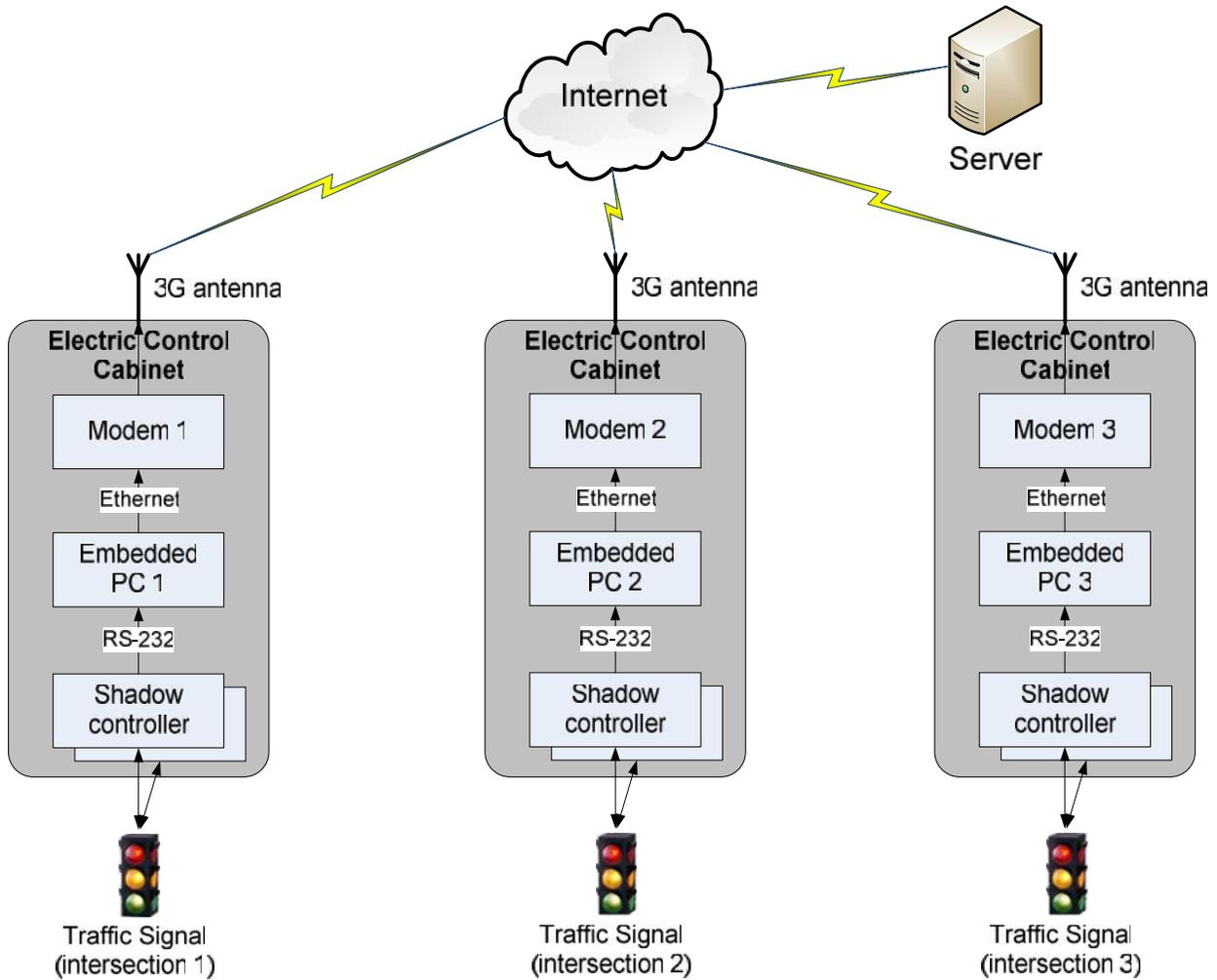
For further extended testing with real traffic signals three intersections on El Camino Real in Palo Alto were chosen (Figure 43):

- El Camino Real / Venture Ave
- El Camino Real / Los Robles Ave
- El Camino Real / Maybell Ave



**Figure 43: Equipped traffic signals on El Camino Real**

Figure 44 shows a schematic overview of the communication structure for the test site with three intersections. However, using the traffic signal data for Cruise control applications (which implicate computations of the optimal driving speed) requires data from a greater number of intersections. As the software is prepared to deal with a theoretically unlimited number of intersections, the only expansion would be to equip the electrical control cabinets of adjacent intersections.



**Figure 44: Test site overview**

Each of the electric control cabinets (ECC) for these intersections has been equipped with a shadow controller, an embedded PC and a 3G modem. The shadow controller mirrors the current state of the traffic signal controller and allows the embedded PC to collect the data without influencing the traffic signal functionality. The processed prediction data is transmitted to the server over a 3G network.

See implementation of the hardware in the control cabinets at the intersections below.

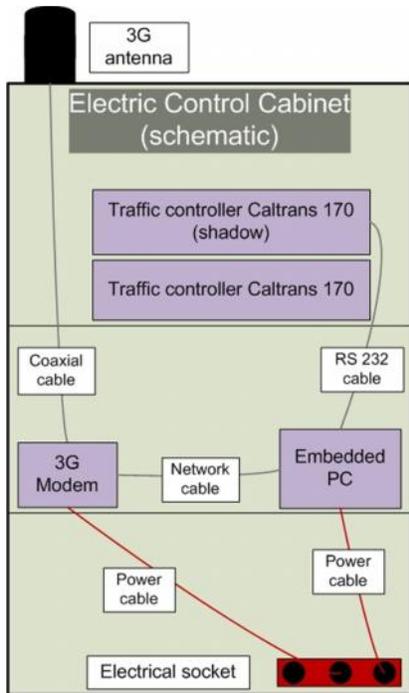


Figure 45: General schematic of an equipped ECC

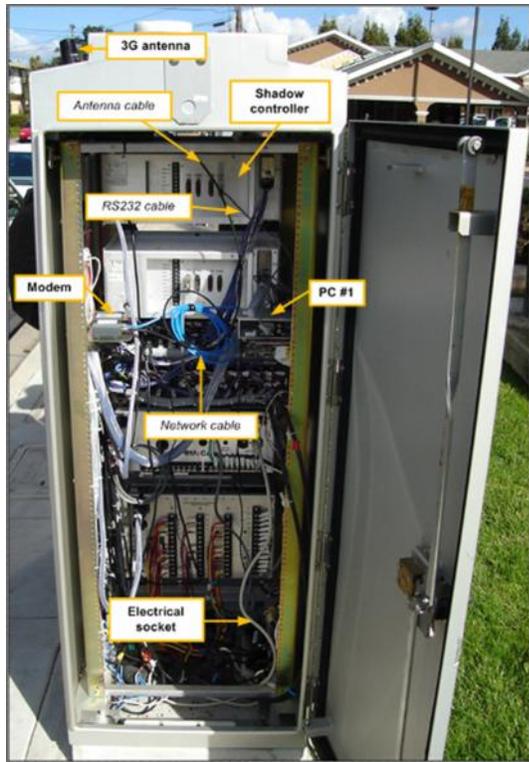


Figure 46: ECC on El Camino Real / Ventura Avenue

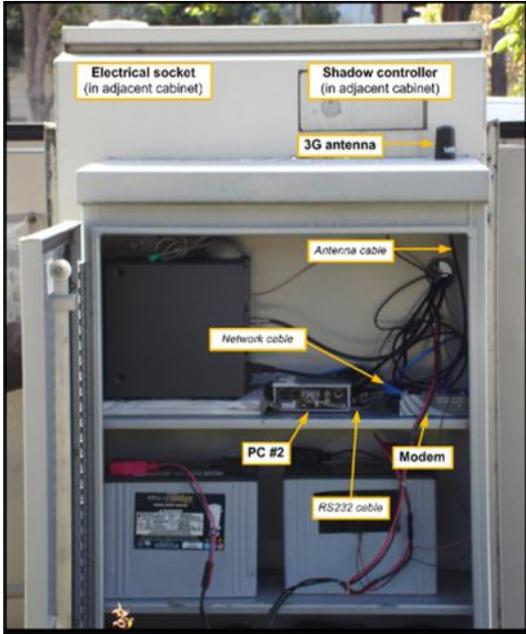


Figure 47: ECC on El Camino Real / Los Robles

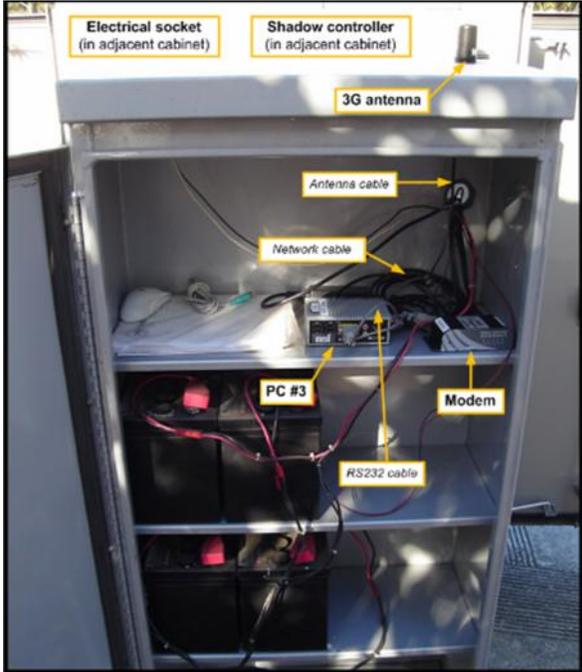


Figure 48: ECC on El Camino Real / Maybell

## 5 Conclusions

A significant amount of the fuel consumption in urban traffic is due to stopping at and accelerating from traffic signals. Beside the waste of fuel, waiting and reaccelerating at a signalized intersection also means additional CO<sub>2</sub> emission.

Recently several studies (Barth, Mandava, Boriboonsomsin, & Xia, 2011) (Asadi, 2011),(Mahler, 2012) have shown in theory, based on simulations, that it is possible to reduce the fuel consumption when approaching traffic signals by adjusting the speed of a vehicle based on knowledge about future signal state changes.

In this project, a prototypic test and demonstration system was developed with the objective of investigating the practical benefits of such eco approach technologies. The developed system gives the driver a speed recommendation when he/she approaches a traffic signal. Using the speed recommendation the driver should be able to adjust his/her speed in a way that is more fuel-efficient. In addition to the speed recommendation a second concept called APIV( Adaptive Priority for Individual Vehicle) was investigated alone as well as in combination with the speed advisory system. In this concept, the timing of a traffic signal is manipulated using information about approaching vehicles.

The developed system consists of a research vehicle, a cloud server and several traffic signals equipped with wireless communication devices. The vehicle was equipped with PCs for the computation of the speed recommendation as well as the capability to display the speed recommendation directly in a programmable instrument cluster. The connection to the cloud server is maintained via a 4G connection. The traffic signals send information about their future timings via a 3G connection to a cloud server, which forwards the appropriate information to the development car.

To test the system for the first step, a controllable environment was chosen: A fixed time traffic signal with no cross traffic or other traffic present at Richmond Field Station. Compared to the uninformed driving scenario the speed advisory system achieved significant fuel savings: using the speed advisory system, fuel savings of 13.6% were observed. For the APIV approach without the speed advisory fuel savings of 19.1% were achieved. Even better, the combination of both approaches showed fuel savings of 28.4%. These results show that each individual system is beneficial for saving fuel on their own, and much higher fuel savings can be achieved by combining the two systems.

Further analysis indicates that fuel savings for the informed driving scenario are mainly from the drivers' early slowing down and cruising through the intersection without having to fully stop at the intersection. In contrast the fuel savings when using APIV are mainly contributed by an increased proportion of driving through green.

To provide a real world environment for further testing, a second test field on El Camino Real was established. Instead of only one fixed time traffic signal this test field consists of 3 traffic signals that are actuated and coordinated. As the initial algorithm was not able to meet the new requirements that come along with an actuated coordinated traffic signal, a new speed advisory algorithm was developed.

Due to limited time and resources, it was not possible to do any meaningful measurements on the El Camino Real test field. Nevertheless an important observation was made during some very first test drives. Driving within real traffic, it is very difficult to follow a speed recommendation that is continuously changing over time. Whereas during testing on the first test site, it was – with some training – possible to follow a continuously changing speed recommendation, but this is impossible when other traffic is present. In conclusion this clarifies that beside the speed advisory algorithm itself, the method of displaying the speed recommendation is crucial for the success of a similar systems in the future.

## 6 Appendix: MAP and SPaT messages

MAP message:

```

typedef struct MapData
{
    DSRCmsgID_t          msgID;          // MessageID for MAP is 7
    MsgCounng            msgCnt;        // Set to 0
    struct intersection   *intersections;
    {
        // Only one intersection per MapData in current version
        struct Intersection list[1]
        {
            IntersectionID_t id;          // Chosen intersection ID
            LaneWidth_t      *laneWidth; // Defined to 500 cm
            struct Position3D *refPoint
            {
                Latitude_t lat; // Latitude of intersec centre
                Longitude_t Long; // Longitude of intersec centre
            }
            struct approaches approaches // Up to 8 approaches
            {
                // In most cases a=4 (for N,S,E,W)
                struct ApproachObject list[a]
                {
                    struct Approach *approach
                    {
                        // Unique approach ID
                        ApproachNumber_t *id;
                        struct drivingLanes *drivingLanes;
                        {
                            // b is the number of lanes per approach
                            struct VehicleReferenceLane list[b]
                            {
                                // Driving phases (1-8) of intersection
                                LaneNumber_t laneNumber;
                                // Additional attributes - not used now
                                VehicleLaneAttributes_t laneAttributes;
                                // Speed limit of current lance
                                LaneSpeedLimit_t laneSpeedLimit;
                                struct NodeList *keepOutList;
                                {
                                    // c is number of points to describe one lane
                                    struct Offsets list[c]
                                    {

```



```

long long send_Timestamp;           // unix time in milliseconds that
                                     // this message is packed
unsigned char intersection_ID;      // local controller address
unsigned char plan_num;             // current control plan
unsigned int cycle_lenght;          // cycle lenght in 10th of second
unsigned char coordinated_phases[2]; // directional main-street phase
                                     // ID, e.g.(2,6) 2 for southbound
                                     // and 6 for northbound

struct Phase_Status
{
    unsigned char phase_id;          //ranged from 1 to 8
    char light_color;               //either G,Y,or R
    // time (in 10th of second) that this phase has been in current
    // state
    unsigned int time_in_current_state;
    // lower bound (in 10th of second) for time-to-change to the next
    // state
    unsigned int time2next_L;
    // upper bound (in 10th of second) for time-to-change to the next
    // state
    unsigned int time2next_U;
    // lower bound (in 10th of second) for time-to-change to the next-
    // next state
    unsigned int time2nextnext_L;
    // upper bound (in 10th of second) for time-to-change to the next-
    // next state
    unsigned int time2nextnext_U;
} phase_status[8];

struct Band_Status
{
    // time (in 10th of second) from now to the start of the next
    // guaranteed green on this phase
    unsigned int time2nextGuaranteedGreenStart;
    // time (in 10th of second) from now to the end of the next
    // guaranteed green on this phase
    unsigned int time2nextGuaranteedGreenEndt;
    //time (in 10th of second) from now to the start of the next green
    // band on this direction
    unsigned int time2nextGreenBandStart;
    // time (in 10th of second) from now to the end of the next green
    // band on this direction
    unsigned int time2nextGreenBandEnd;
    // progression speed (in m/s) for the green band
    unsigned int bandSpeed;
} band_status[2];

```

## 7 Bibliography

- Asadi, B. V. (2011). Predictive Cruise Control for Improving Fuel Economy and Reducing Trip Time. *IEEE Transactions on Control Systems Technology*, 19, 707-714.
- Barth, M., Mandava, S., Boriboonsomsin, K., & Xia, H. (2011). Dynamic ECO-driving for arterial corridors. *Integrated and Sustainable Transportation System (FISTS)*.
- Mahler, G. V. (2012). Reducing Idling at Red Lights Based on Probabilistic Prediction of Traffic Signal Timings. *American Control Conference*. Montreal: IEEE.
- National Transportation Operation Coalition. (2012). *2012 National Traffic Signal Report Card*. Retrieved July 30, 2012, from <http://www.ite.org/REPORTCARD>
- Schrank, D. L. (2011). *2011 Urban Mobility Report*. Retrieved 2012, from Texas Transportation Institute: <http://tti.tamu.edu/documents/mobility-report-2011.pdf>
- US DOT Federal Highway Administration. (n.d.). *Focus on Congestion Relief*. Retrieved July 30, 2012, from <http://www.fhwa.dot.gov/congestion/toolbox/service.htm>
- Xia, H., Boriboonsomsin, K., Schweizer, F., Winckler, A., Zhou, K., Zhang, W.-B., et al. (2012). Field Operational Testing of ECO-Approach Technology at a Fixed-Time Signalized Intersection. *15th International IEEE Annual Conference on Intelligent Transportation Systems*.
- Zhou, K. Adaptive Priority for Individual Vehicles.