The development of a cooperative heavy-duty vehicle for the GCDC 2011: Team Scoop

Jonas Mårtensson*, Assad Alam, Sagar Behere, Muhammad Altamash Ahmed Khan, Joakim Kjellberg
Kuo-Yun Liang, Henrik Pettersson, and Dennis Sundman

***Abstract*—The first edition of the Grand Cooperative Driving Challenge (GCDC) was held in the Netherlands in May 2011. Nine international teams were competing in urban and highway platooning scenarios with prototype vehicles using cooperative adaptive cruise control. Team Scoop, a collaboration between KTH Royal Institute of Technology in Stockholm and Scania CV AB in Södertälje, participated in the GCDC with a Scania R-series tractor unit. This paper describes the development and design of team Scoop's prototype system for the GCDC. In particular we present considerations regarding system architecture, state estimation and sensor fusion, design and implementation of control algorithms and implementation issues regarding the wireless communication. The purpose of the paper is to give a broad overview of the different components that are needed to develop a cooperative driving system; from architectural design, workflow and functional requirements descriptions to the specific implementation of algorithms for state estimation and control. The approach is more pragmatic than scientific; it collects a number of existing technologies and gives an implementation oriented view of a cooperative vehicle. The main conclusion is that it is possible, with a modest effort, to design and implement a system that can function well in cooperation with other vehicles in realistic traffic scenarios.**

I. Introduction

The transportation system faces big challenges. The demand for transportation is steadily increasing, while the impact on the environment needs to be significantly reduced and road congestion better controlled. Fortunately, the rapid development in information and communication technology (ICT) presents an excellent opportunity to tackle these problems through novel integrated intelligent transportation system (ITS) solutions. Transportation is responsible for the main part of the increase in oil consumption during the last three decades and the growth is expected to continue. In 2006, road transport accounted for 26% of the total energy consumption and for 93% of the total transport-related CO_2 emissions [1]. Hence, the entire transport sector, and particularly road freight transport by trucks and lorries, has been targeted as a main policy area where further environmental and overall efficiency improvements are critical for a sustainable future of European transport. To ensure sustainability and global acceptance of commercial transportation, new systems which reduce the dependence on oil and minimize emission of greenhouse gases need to be developed. The European Commission's goal for 2030 is to reduce European road transport greenhouse gas emissions to around 80% of the 2008 level [2]. In the same period, the freight transport is expected to increase by 75% [3].

Advanced ITS technologies will play a key role in addressing the transportation challenges outlined above. An example is given by the so called green corridors, which will require the development of a new type of intelligent and cooperative vehicles and supporting ICT infrastructure and systems. The vehicles in these corridors need to communicate with each other (V2V) and with the infrastructure (V2I). These technologies open up opportunities for concepts like platooning, or convoy driving, which could reduce the environmental impact significantly as well as relieve traffic congestion.

A. Platooning and cooperative driving systems

Vehicle platooning is the concept of having several vehicles drive safely together with a short intermediate distance, often (but not necessarily) making use of vehicle-to-vehicle communication. Platooning has the potential of contributing to the solution of several challenging transportation problems; relieving congestions, reducing the emission of greenhouse gases and making the road transport more energy efficient. Platooning with short intermediate distance between the vehicles means that the vehicles can be more densely packed and, hence, the transportation capacity of the road network will increase. Another benefit of platooning is that the aerodynamical forces acting on the vehicles decrease, which leads to substantial reductions of fuel consumption and greenhouse gas emissions. This is in particular true for platoons of heavy duty vehicles for which the fuel saving potential is about 5-15% [4], [5]. Although these effects are visible already at a distance of 30 meters, the big savings will come when the vehicles can drive closer than 10 meters apart on the highway. This of course requires accurate (semi-)autonomous[1] control and safe and robust electronic systems, since the reaction times of human drivers are insufficient.

Many people have studied platooning and cooperative driving systems before; both the performance and stability of the cooperating vehicles themselves, and the effect it may have on the rest of the traffic [6], [7], [8], [9], [10], [11], [12],

*Corresponding author.
J. Mårtensson, D. Sundman and M. A. A. Khan are with the ACCESS Linnaeus Center, School of Electrical Engineering, KTH Royal Institute of Technology, Stockholm, Sweden.
S. Behere is with the Department of Machine Design, School of Industrial Engineering and Management, KTH Royal Institute of Technology, Stockholm, Sweden.
A. Alam, K.-Y. Liang, J. Kjellberg and H. Pettersson are with Research and Development, Scania CV AB, Södertälje, Sweden.

[1]Semi-autonomous means that longitudinal control is autonomous, while lateral control is manual.

[13], [14]. Cooperation among vehicles equipped with wireless communication capabilities is a very important research topic now. The European Union has been hosting a number of research projects in this are, for example CVIS[2] that develops communication technology, SAFESPOT[3] and COOPERS[4] that focus on road safety, and SARTRE[5] on platooning. There is also significant publication activity in the area. Of particular interest are two recent special issues on wireless vehicular communications [15] and emergent cooperative technologies [16].

B. The Grand Cooperative Driving Challenge

The first Grand Cooperative Driving Challenge (GCDC) was organized by TNO of the Netherlands in Helmond in May 2011. The competition background, scenarios and judgement methodology are presented in [17]. In short, the GCDC is a competition in platooning (in the sense of Cooperative Adaptive Cruise Control), where the teams are supposed to develop a prototype system for a vehicle that semi-autonomously drives in two pre-defined scenarios; one urban scenario and one highway scenario. The urban part consists of an automatic launch from a traffic light, arranged so that a rear platoon catches up on and joins a front platoon. The highway part is driven at a higher speed and a lead vehicle is injecting disturbances (accelerations/decelerations) into the platoon. The goal is to maintain a specified intermediate distance to the vehicle ahead and to make sure that the disturbances are not amplified from one vehicle to the next. The rules and technological requirements mandated by the GCDC organization can be found in [17].

C. Outline and scope

This paper describes the development of a cooperative driving system for the Grand Cooperative Driving Challenge (GCDC) 2011. The implementation is made by team Scoop, which is a collaboration between KTH Royal Institute of Technology and Scania CV AB. A large part of the implementation and design was a direct result of eight master thesis projects [18], [19], [20], [21], [22]. The purpose of the paper is to give a broad overview of the different components that are needed for participating in the GCDC. The approach is pragmatic; several existing techniques for communication, state estimation and control are combined and experimentally validated in real-world scenarios together with other vehicles with different implementations. Solutions to many practical problems are presented; such as dealing with traffic light information, forming and splitting of platoons and degradation strategies when communication fails.

The following four sections are devoted to certain aspects of our implementation; system architecture in Section II, wireless communications in Section III, state estimation and sensor fusion in Section IV, and control algorithms in Section V. Finally, a brief summary and some general conclusions are given in Section VI.

[2]http://www.cvisproject.org/
[3]http://www.safespot-eu.org/
[4]http://www.coopers-ip.eu/
[5]http://www.sartre-project.eu

II. Architecture

This section describes the architecture of the Scoop system. The Scoop system refers to the hardware and software that is used to realize the cooperative, adaptive cruise control functionality. The system is installed as a set of distinct, additional components in a factory standard vehicle. These components comprise sensors, hardware processors and communication media between the hardware processors. The interaction between the Scoop system and the rest of the vehicle takes place via the vehicle CAN bus. Specifically, the Scoop system reads relevant signals from vehicle sensors and subsystems connected to the CAN bus and writes appropriate actuation signals to the CAN bus. These actuation signals are intended for consumption by the various factory standard motion control subsystems within the vehicle.

The Scoop system architecture emphasizes clean design, separation of concerns and encapsulation of functionality into components.

A. Engineering requirements

Some key engineering requirements were formulated before the design was started. The requirements were based mostly on engineering experience, rather than a specific development methodology or theory. The requirements were that the architecture should

1) aid separation of functions, both in their development and implementation. A function is a self-contained unit of functionality that does not depend on other functions (at the same hierarchical level), except for receiving its input. A function may be composed of a hierarchy of constituent functions.
2) allow dynamic changes to the way its constituent components interact, in order to give rise to different system behaviors.
3) make it easy to swap in and out different algorithms for achieving individual system functions.
4) contain diagnostic and self-monitoring services so that the health of constituent parts can be actively monitored and faults, if any, can be easily isolated and detected.
5) make it possible to interact with the system as it is running. It should be possible to examine and modify parameter values in a running system and observe their effects.
6) be implementable using existing and proven tools, software frameworks, libraries and services.
7) be minimally intrusive/invasive in the existing vehicle architecture.

B. Solution description

This section describes the Scoop architecture in more detail. The description begins with statements of some preliminary decisions and their motivation. Next, the functional decomposition is described. This is followed by a system level view, implementation details and an explanation of how the architecture leads to intended system behavior.

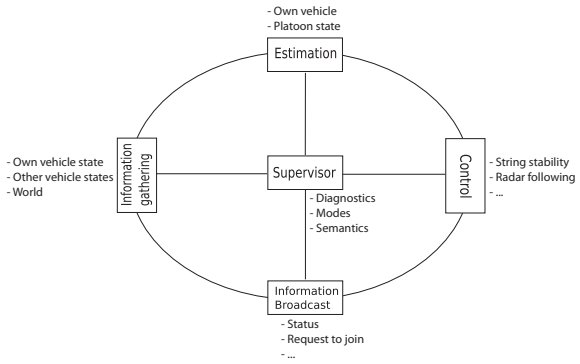


Figure 1: Top level functions

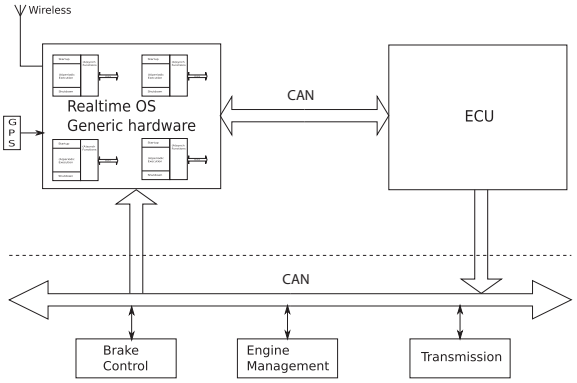


Figure 2: System view

1) Preliminary decisions: The preliminary decisions are summarized as

1) The architectural pattern is based on simultaneous execution of multiple components.
2) The computation/control is split over two distinct types of hardware. One is a generic computer with a GNU/Linux software environment, while the other is a standard automotive electronic control unit (ECU). The generic computer handles data communication over the wireless interface, GPS and reads data from the vehicle CAN bus. It is responsible for sensor data filtering, fusion and state estimation. The ECU realizes platooning and vehicle control functions.
3) In the generic computer, standard operating system services (a.k.a *daemons*) are utilized for communication with the GPS devices and system time synchronization. This decision is inline with the engineering requirement of reusing existing libraries and services. Specifically, the readily available *gpsd* daemon[23] is used as an abstraction for GPS devices and the standard *ntpd* daemon[24] is used to automatically synchronize system clock with the GPS clock.
4) The Orocos[25], [26] software component framework is used for instantiating and executing components in the generic computer. This framework is chosen because it is mature, open source, cross-platform, supports real-time execution of components and due to reasons of prior experience with it and its support community.
5) A datalogger component is provided for logging inter-component dataflows, as well as for providing developers with services to log additional data and human machine interface (HMI) messages. This component does not interfere in the core system operation and it is possible to run the system with this component disabled.
6) The HMI part of the system is completely independent in design and execution of the core system. It utilizes information from the datalogger component and is capable of running on a remote computer.

2) Top level functions: The system is decomposed into five toplevel functions, as shown in Figure 1.

The **information gathering** function has the responsibility of gathering data about the host vehicle and the environment. The environment consists of other vehicles in the vicinity and road objects like speed signs, traffic lights, lane markings etc. This information can be gathered by multiple means. For example, information about the host vehicle can be gathered by reading the vehicle's CAN bus and via a GPS device. Information about other vehicles can be gathered from wireless broadcasts or local sensors like radar. Information about road objects can be gathered from wireless broadcasts or local sensors like cameras. The information gathering component is thus a conceptual function that provides all the raw data necessary for operation, regardless of the nature and source of the data. The raw data provided by the information gathering function is filtered and fused by the **estimation** function. The output of the estimation function is a set of host vehicle, platoon and environment states.

The **control** function is responsible for providing vehicle speed, acceleration and other motion parameters to the rest of the system. It is also responsible for decisions regarding joining and leaving platoons.

A vehicle participating in a cooperative driving scenario must broadcast certain information about itself. This information is collected from various parts within the system and broadcast (generally over wireless media) at different frequencies[6]. The broadcasts may also contain control requests, for example, requests to join existing platoons. The **information broadcast** function is the single place from which information goes out of the vehicle.

The **supervisor** function is responsible for the overall working of the system. It performs mode management, diagnostic monitoring and coordinates the information flow within the system. Conceptually, it is above all other functions in the hierarchy. The supervisor is also responsible for graceful system degradation in case of problems.

3) System view: This section presents a high-level logical view of the architecture as shown in Figure 2.

The ECU represents the control function. The reason to show it separately is to emphasize the fact that in function, implementation and operation, it is separate from the rest of the system. It reads a certain set of data as input and makes certain actuation requests as output.

The brake control, engine management and transmission are the 'actuators' influenced by the ECU. These are standard

[6]Frequencies here means at different intervals, not the radio frequency of transmission.

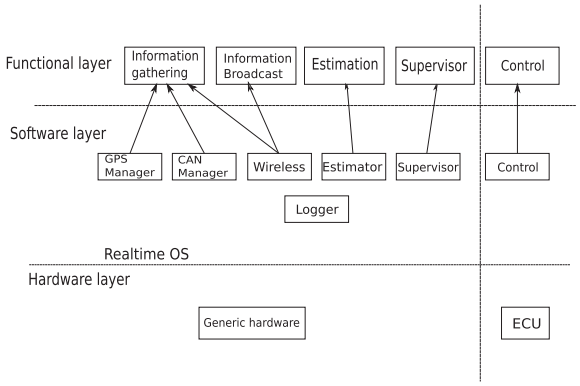


Figure 3: Implementation view

vehicle functions provided by the manufacturer for the vehicle's motion control. The ECU uses them to realize desired vehicle trajectories. Finally, the rest of the functions are grouped together for execution on a generic computer. This is because their implementation is as individual components in a component based software framework.

The generic computer and ECU are connected to the vehicle's CAN bus, on which the 'actuators' are also present. There is a dedicated CAN bus connecting the generic computer and the ECU to handle the high bandwith communication between them.

The GPS device as well as wireless routers are connected to the generic computer.

4) Implementation: The implementation view of the architecture is shown in Figure 3. This view shows the functions, software components and hardware and also how they are mapped to each other. The functional layer shows the top level functions that the architecture should realize. It is a purely abstract layer, intended as a guideline for the implementation of the other layers.

The next layer is the software layer, which shows the developed software components and their contribution to the top level functions. A top level function may be realized using more than one software component. For example, the Information Gathering function is realized using a combination of the GPS manager, CAN manager and wireless components. A software component may contribute to the realization of more than one top level function. For example, the wireless component contributes to the realization of the Information Gathering as well as the Information Broadcast functions. There also exist software components that do not directly contribute to the realization of top level functions. An example is the logger component. The logger is useful for debugging the system and exists purely at the software level. The software components are executed by a realtime operating system, in this case GNU/Linux, running the Xenomai realtime framework [27] for the Linux kernel.

The software implementation of the control function is similar in principle to other components in the software layer. However, the control function differs in its implementation. The control component is designed and implemented in Simulink and the software code it executes is autogenerated from within Simulink. Thus, there is no traditional hand-coding of the control function.

The lowermost layer is the hardware layer, which executes the contents of the software layer. This hardware layer is partitioned into two parts. The first part is the ECU, which is a physically distinct piece of hardware that executes the control software. The second part is a generic computer running the GNU/Linux operating system.

5) Emergence of system behavior: A big portion of the architecture consists of a set of connected components within the generic computer, which exchange data. All the components in the generic computer are periodic. In every execution period, a component wakes up, does a specific task and goes back to sleep. However, before going back to sleep, a component services requests that may have been made by other components while it was asleep. Thus, overall system behavior emerges from the interaction among the components (data flows and service requests). The different components behave as follows, every time they wake up

i) The GPS, CAN and wireless components read data from their respective information sources and send it to the estimator. The CAN component reads data from the ECU as well as the vehicle CAN bus.
ii) The estimator uses its input data to update state vectors for the host vehicle, surrounding vehicles and road objects. This information is then sent to the CAN component.
iii) The CAN component forwards the estimator information to the control component in the ECU.
iv) The control uses the information to enter/stay in an appropriate control strategy and influences the vehicle actuators. Information about the ECU actions is sent back to the CAN component.
v) The supervisor component is responsible for system initialization, monitoring the status of other components, rerouting data-flows in case of component malfunctions and system level error management.
vi) The wireless component also periodically reads data from the estimator and broadcasts it over the wireless interface.
vii) All components send logging data to the Logger component which periodically serializes it through a TCP socket to a separate computer. This separate computer logs the data to disk and also extracts relevant data and presents on a graphical interface.

Now let us see how this process works in a typical scenario. Assume that the vehicle is standing at a red light, and should start moving when it turns green. The following sequence of events takes place

i) The GPS reports current vehicle coordinates to the estimator
ii) Information about the position and state of the traffic light is obtained by the wireless component and sent to the estimator
iii) The estimator calculates the distance to the traffic light, the color and the vehicle position and sends this information to the control via the CAN component
iv) The control component continuously monitors this information and keeps the brakes engaged while the traffic light directly ahead is red and the vehicle is within a

threshold distance to the light

v) When the control component receives information that the traffic light has turned green, it disengages the brakes and starts accelerating the vehicle forward

C. Evaluation

One criterion for our evaluation of the architecture is how much it could stay in the background, avoiding intrusion in the function developers mindspace. A good architecture should enable developers to easily get the job done, without thinking too much about architectural limitations or using 'quick-and-dirty' hacks to bypass the limitations. Other criteria include extensibility, robustness and safety. Extensibility means the ability to easily add more functionality without having to redesign the fundamental structure. Robustness implies lower sensitivity to problems in the operating environment as well as data being processed. Safety means that at no point should the architecture result in an operating condition that is hazardous to the vehicle or its environment.

Our architecture scores very well on the above measures. It clearly identifies what solution patterns are possible, in terms of available data streams, their routing among the components and the ways in which components can utilize services offered by other components. In doing so, it leads and guides the thinking of the developers, rather than constraining their ideas. At the same time, the breadth and depth of these patterns is such that it serves to show possibilities, rather than constrain the ideas of a developer. The components within the architecture are fairly independent, and there exists loose coupling between them. At no point does any component require a detailed knowledge of the inner working of another component. This independence of the components also helps to isolate faults and limit their propagation, thus making the system more robust. The supervisor component can be used to quickly "re-wire" the connections between components, leading to flexible system behaviors.

A few minor issues were identified which occurred more due to lack of sufficient internal communication among the team, rather than architectural restrictions. However, none of these issues are especially severe and fixing them does not require fundamental changes to the architecture. At no point did the system enter unsafe states. The emergency and manual overrides always worked, but weren't actually needed during the competition. We believe this architecture is a substantially sound basis for further development of vehicular ITS systems.

III. Communication

The wireless communication in our system is designed using the required 802.11p physical layer, the CALM-fast protocol and ASN1 package coding. CALM stands for communication access for land mobiles, fast refer to non-IP based communication and ASN1 stands for abstract syntax notation version one. In this section we will describe our implementation of the communication module. For more details regarding the communication protocols, see [19].

A. Communication Protocols

The protocol for the wireless communication is decided by GCDC: a CALM-fast stack on top of the 802.11p physical layer, with ASN1 encoding. In the following we will discuss our solution to on how to implement the 802.11p physical layer and the CALM-fast stack. We will also give some comments on the ASN1 encoding/decoding.

1) CALM daemon and libcalmfast: The CALM-fast protocol is not a standard and in order to make the network card for 802.11g/h work with 802.11p, some modifications to the driver was necessary. In addition to these modifications, a calm library and a background process (*daemon*) called the calmd (for calm daemon) is used. This software was provided by GCDC organization provided a 32-bit kernel. However, many teams used a 64-bit kernel and team Annieway hosted a fork of the software. Most teams who were using it contributed in terms of bug-fixes, including us.

2) ASN1 encoder/decoder: For the ASN1 encoding and decoding, we use the open source software asn1c [28]. Because our system is 64-bit it is necessary to be careful with two things. The first thing is to make sure you use version 0.9.23 (only possible to get via GIT) and the second thing is to replace all ambigious types (`unsigned long`, etc).

B. Program flow

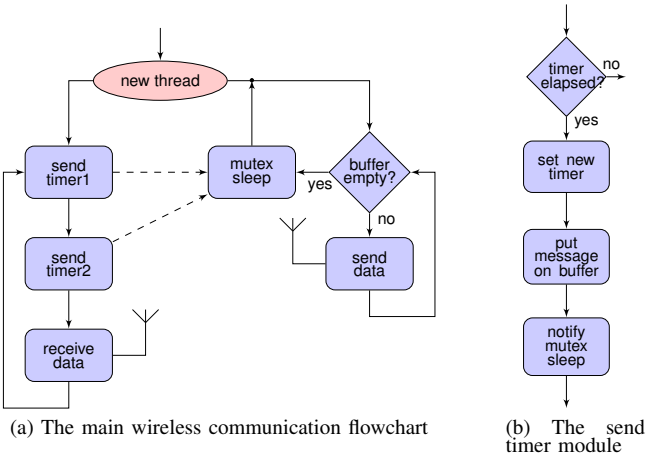
The main challenge in constructing the wireless communication component of our system is the requirement of robustness and computational efficiency. The program needs to be robust since internal control relies heavily on what we receive from other vehicles and we can expect that other vehicles rely heavily on what we are transmitting. The program needs to be computationally efficient since it resides on the same platform as the more complex (and crucial) state estimations and CAN-message communication components.

A limiting factor in the communication libraries is that the proposed sending function call `cf_publish` is a blocking call. We can not allow the communication component to block the system, so we create two threads. One thread then takes care of the blocking sending function and the other takes care of the remaining, non-blocking, tasks. When dealing with multi-threaded programs it is important to beware of dead-locks. A dead-lock can occur when two, or more, threads simultaneously wait for the other to finish some task. To avoid dead-locks, mutexes (mutual exclusion) are used. It is very important that any code involving mutexes is free of bugs, since a problem with them may lock the whole system. To simplify thread and mutex implementation we are using the Boost libraries [29].

A simplified program flow of the communication component can be seen in Figure 4a. On the left-hand side of the "new thread"-cloud of the diagram, the parent thread is shown and on the right-hand side of the cloud the child thread is shown. The parent keeps adding information to the outgoing buffer, which is sent over the wireless by the child. The dashed arrows from the timers to the mutex sleep symbolizes a notification from the parent to the child thread that there is work to do. Besides the timers, there are other components in the parent

thread that can put information to the transmit buffer, but these are not depicted here. The parent also receives data and forwards it to other components in the system, i.e. the Estimator and CAN components.

In Figure 4b, the send timer flow is depicted. In this figure, the timer is restarted, a message is put on the buffer and the notification to the mutex sleep is sent. There are two send timer, because there are two types of messages that are sent repeatedly at different timer-intervals.



(a) The main wireless communication flowchart (b) The send timer module

Figure 4: Communication program flow

Besides handling the wireless communication, the wireless component takes care of some simple transformation from external to internal units. For example, externally, *km/h* is the unit for speed but internally the unit is *m/s*. This conversion is done in the receive data and send data clouds in Figure 4a. These clouds also take care of the encoding/decoding from ASN1 and incorporates the appropriate functionalities from the calm libraries (see section III-A1).

C. Physical Set-Up

From our system's perspective, the calmd works as an abstraction for the wireless interface. Our wireless component sends/receives data to the calmd via TCP/IP sockets. Therefore, it is irrelevant where the calmd is executing. In case one system should fail, we have two equivalent physical options for the wireless. The first option is to let the calmd run on the same machine as the rest of the system, with an off-the-shelf network card and modified network stack. The second option is to let the calmd run on a wireless router provided by TNO (ALIX). Switching between these two options requires the change in the configuration containing IP-address information of where the calmd is running.

D. Evaluation/Conclusions

It turned out that we had some trouble with our own network hardware. The problem appeared as communication losses spanning a few seconds up to several minutes at seemingly random occasions. The reason for this is not identified, but we suspect that it may depend on a faulty PCI to miniPCI converter used for the network card. We solved the problem by switching to the equivalent system setup where the calmd runs on the ALIX board (see previous section).

At three occations during the preparation week, our system mysteriously crashed due to some problem in the wireless code. At the last crash, we had the forseight to make sure a coredump with crash information was generated. It turned out that one of the asn1c function methods (`uper_decode`) for decoding a message coming from other vehicles may return success when in fact the actual message it returns is empty. When we then later try to release the memory allocated by this (empty) message, the program crashes. We solved this by a redundant message sanity check.

With the two problems fixed, we experienced very good wireless performance. Thanks to the truck, our antenna was mounted at a relative high position providing good coverage and thanks to the simple design, communication delays were down to order of milliseconds.

IV. State estimation

In this section we look more into the state estimation module, which estimates a number of states based on the available information, and the mathematical models that are involved. The information sources for the state estimation are the vehicles' onboard sensors and also the data transmitted from the other vehicles in the platoon. Kalman filters [30] are employed as the state estimator. For a more detailed description of the estimator, see [20].

A. Background

The states to be estimated are the variables that are necessary for the controller to control the vehicle. They mainly include the relative distances of the platoon vehicles with respect to the own (host) vehicle, their absolute velocities and their absolute accelerations [18]. Additionally, GCDC has specified mandatory information that are exchanged between the vehicles via wireless, which also has to be estimated. There are four types of objects that need to be tracked: the host vehicle, the other vehicles, speed signs, and traffic lights. For the host vehicle, a Kalman filter based on the classical bicycle model (see e.g. [31]) is considered. For the other vehicles, a simpler Kalman filter is used and for the speed signs and traffic lights, only a few states are estimated.

B. Available information

Four types of sensors are used for gathering the measurement data in this study. These are the GPS, wheel speed sensor, acceleration sensors and the gyro / steering sensors. All of these sensors except the GPS are embedded internally in the truck and can be accessed via the controller area network (CAN) bus system. In addition to these raw sensors, we have also access to a radar and information gathered from the infrastructure and other vehicles over the wireless. Details of the sensors are given below.

1) GPS: The GCDC organization provided a real time kinematic (RTK) GPS transmitter and to guarantee that we could meet the position accuracy we use the Trimble SPS 852 GPS receiver that can achieve an order of decimeter accuracy using the RTK signal. From the GPS receiver, we are mainly interested in the position (longitude & latitude), heading and ground speed information. The GPS is configured to deliver data at 10 Hz rate.

2) Wheel speed sensor: The ground speed data from the GPS is fairly accurate, but as the GPS signal reception can sometimes fail, a redundant source of ground speed has to be available. Therefore, the speed from the wheel speed sensor (WSS) was used as the secondary vehicle road speed source. Also, the variation in the wheel radius can result in less accurate speed estimates. This primarily occurs due to changes in the tyre pressure over the passage of time. The resulting error incurred in the speed from the WSS is therefore time varying in nature. This error is modeled by a time varying factor which is estimated with the help of GPS road speed serving as the primary reference.

3) Acceleration sensors: Two types of acceleration signals are available from the embedded sensors in the truck. One is the longitudinal acceleration, which is obtained by differentiating the vehicle speed of the truck. Apart from the longitudinal acceleration, the lateral acceleration signal is also measured. The lateral signal source is an accelerometer present in the truck that directly measures the body y-axis acceleration. This signal also contains an inherent bias, which may be time varying in nature. It can be caused by non-horizontal mounting of the sensor or by driving on a non-horizontal road. It could also represents some degree of imprecision in the instrument manufacturing.

4) Gyro and steering sensor: The vehicles built-in gyroscope is used to measure the yaw-rate of the truck. A bias signal that is associated with the measurements is also modeled and estimated. Apart from the yaw-rate, steering wheel angle measurements are also used. In the current set-up, there is no sensor providing the angle directly from the front wheels. The wheel angle is derived from the angle of the servo for the steering wheel.

5) Radar: The TRW AC20 millimeter wave radar is used, which is a standard radar present in many Scania trucks and it works in the 76-77 GHz band. It has an advantage over other types of sensors, such as optical or infrared sensors, in that it performs reliably during day, night and in most weather conditions. It is used to get the distance of the vehicle in front, its relative speed and absolute acceleration.

6) Wireless: The wireless communication is the source of information from the other vehicles in the platoon. Each vehicle transmits its state information in a pre-decided message format.

C. Requirements from GCDC

The GCDC committee specified the accuracy requirements for the dynamic states that each competing vehicle has to deliver to the other vehicles over the wireless channels. These are listed below

a) Position accuracy: The requirement for the position accuracy ϵ_p is given by, $\epsilon_p \leq 1\,\mathrm{m}$.

b) Speed accuracy: Only longitudinal speed estimates are required to be transmitted. The criterion for the speed accuracy ϵ_s is given by, $\epsilon_s \leq 0.5\,\mathrm{m/s}$.

c) Acceleration accuracy: The criterion for the longitudinal acceleration accuracy ϵ_a is given by, $\epsilon_a \leq 0.2\,\mathrm{m/s^2}$.

D. Vehicle Modeling

For the host vehicle, the choice of which states to be estimated is based on multiple things: required information by the controller, requirements imposed by GCDC and information availability from the various sensors, see section IV-B. The states estimated for the own vehicle are: position in east-north-up (ENU) reference frame $[p_e, p_n, p_u]$, velocity in the body frame $[v_x, v_y]$, vertical velocity v_d, acceleration in the body frame $[a_x, a_y]$, heading angle ψ, side-slip angle β and yaw-rate ω. In addition, some scaling and bias parameters are tracked: speed scale factor η, lateral accelerometer bias b_{ay} and yaw-rate gyro bias b_ω. For efficiently keeping track of all these parameters, estimation is divided in three sub-models described in the following sections.

1) Speed Estimation Sub-Model: This model deals with estimation of speed and acceleration dynamics: v_x, v_d, a_x and η, and are defined by the following set of equations

$$\begin{aligned} \dot{v}_x &= a_x \\ \dot{v}_d &= 0 \\ \dot{a}_x &= 0 \\ \dot{\eta} &= 0. \end{aligned} \tag{1}$$

In the above model, the vertical velocity v_d and longitudinal acceleration a_x, together with the speed scale factor η are modeled as random walk i.e. their derivatives are set to zero. This is because their dynamics could not have been modeled based on the available information. The measurement equations for this model are:

$$\begin{aligned} v_x^{can} &= (1+\eta)\, v_x + e_{v,can} \\ v_G^{gps} &= v_x + e_{v,gps} \\ v_d^{gps} &= v_d + e_{vd} \\ a_x^{can} &= a_x + e_{ax}. \end{aligned} \tag{2}$$

Here, v_G^{gps} is the GPS measured ground speed. Here, the measurement noise vector is given by $e^{(spd)}$ is given by $e^{(spd)} = [e_{v,can}, e_{v,gps}, e_{vd}, e_{ax}]^T$.

2) Lateral Dynamics Estimation Sub-Model: This model deals with estimation of the lateral dynamics: $\beta, \omega, \psi, b_{ay}, b_\omega$ and v_y. These states are defined by the following set of equations, as mentioned in [32], [33], [34]:

$$\begin{aligned} \dot{\beta} &= \tfrac{-C_F - C_R}{m v_x}\beta + \left(\tfrac{-aC_F + bC_R}{m v_x^2} - 1\right)\omega + \tfrac{C_F}{m v_x}\delta, \\ \dot{\omega} &= \tfrac{-aC_F + bC_R}{I_z}\beta + \left(\tfrac{a^2 C_F + b^2 C_R}{I_z v_x}\right)\omega \\ &\qquad + \left(\tfrac{-aC_F + bC_R}{I_z v_x}\right) v_y + \tfrac{aC_F}{I_z}\delta, \\ \dot{v}_y &= \left(-v_x + \tfrac{-aC_F + bC_R}{m v_x}\right)\omega + \left(\tfrac{-C_F - C_R}{m v_x}\right) v_y + \tfrac{C_F}{m}\delta, \\ \dot{\psi} &= \psi, \qquad \dot{b}_{ay} = 0, \qquad \dot{b}_\omega = 0. \end{aligned} \tag{3}$$

In (3), m is the vehicle mass, a and b are the distances between the center of gravity and the front and back wheels, C_F and C_R are the cornering stiffness of the tires and I_z is the vehicle moment of inertia about the z-axis. These parameters need to be estimated in some way, but that problem is not treated here. Longitudinal velocity v_x and the wheel steering angle δ appears as the control input. $\dot{\psi}$ is set to ψ as mentioned in [32] and the derivative of two biases b_{ay} and b_ω are set to zero as they modeled as random walk. The measurement equation for this sub-model is given by

$$\begin{aligned} \psi^{gps} &= \psi + e_\psi \\ \omega^{can} &= \omega + e_\omega \\ a_y^{can} &= \tfrac{-aC_F - bC_R}{mv_x}\omega + \tfrac{-C_F - C_R}{mv_x}v_y + b_{ay} + \tfrac{C_F}{m}\delta + e_{ay}. \end{aligned} \tag{4}$$

In (4), subscript on the measurement variables indicate the source of the measurement. The measurement noise vector $e^{(lat)}$ is given by $e^{(lat)} = [e_\psi, e_\omega, e_{ay}]^T$.

3) Navigation Sub-Model: This model estimates the positions $[p_e, p_n, p_u]$ in ENU frame by the following set of equations

$$\begin{aligned} \dot{p}_e &= v_x \sin\psi \sqrt{1 - \tfrac{v_d^2}{v_x^2+v_y^2}} + v_y \cos\psi \\ \dot{p}_n &= v_x \cos\psi \sqrt{1 - \tfrac{v_d^2}{v_x^2+v_y^2}} - v_y \sin\psi \\ \dot{p}_u &= \tfrac{-v_x v_d}{\sqrt{v_x^2+v_y^2}}. \end{aligned} \tag{5}$$

The position measurements from the GPS receiver are defined in geodetic frame of reference i.e. in terms of longitude, latitude and altitude above the mean sea level. They are converted into the position in ENU frame using equation mentioned in [35]. The measurement equations for the model, in ENU frame, are given by

$$\begin{aligned} p_e^{gps} &= p_e + e_e \\ p_n^{gps} &= p_n + e_n \\ p_u^{gps} &= p_u + e_u, \end{aligned} \tag{6}$$

where the noise is given by the noise vector $e_k = [e_e, e_n, e_u]^T$.

4) Implementation: Three interconnected filters are created using the sub-models. This filter structure is depicted in Figure 5. First is the speed estimation filter that feeds its estimates to the lateral dynamics model. Finally the navigation filter is run based on the outputs from the former two filters. The speed estimation filter is implemented as an extended Kalman filter, while the other two filters are regular Kalman filters.

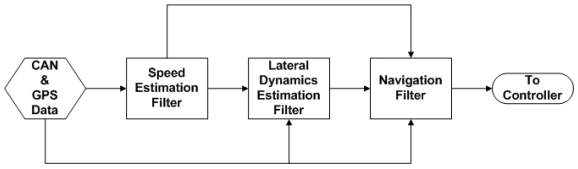


Figure 5: Implementation structure of the Kalman filter

E. Platoon Vehicle Modeling

When multiple vehicles travels one after another they can choose to form a platoon. In our definition of a platoon, each vehicle that wants to participate in the platoon will have to join the platoon by changing their platoon ID. The platoon ID of a vehicle is broadcasted over the wireless so other vehicles know which platoon they are part of. The vehicle at the absolute front of the platoon is termed as the platoon leader. Each vehicle tries to keep the coherence in the platoon by controlling its speed i.e. maintaining a constant distance to the vehicle ahead. The vehicle ahead will be carefully estimated regardless of wether it is part of our platoon or not. In our implementation, the controller bases many of its actions on properties of the platoon members in front of us and of the vehicle ahead. The estimation module is responsible for providing a set of control states, defined in [18], of the platoon vehicles to the controller. The control states are not necessarily the same states as the states tracked by the estimator, but they are rather a set of parameters associated with each individual vehicle in relation to the host vehicle influencing its behavior. The states in our system are:

- The relative base distance of the other platoon vehicle relative to the host vehicle along the axis of motion. This variable is positive for the vehicles ahead and negative for the ones behind.
- The absolute speed of the other platoon vehicles.
- The absolute acceleration of the other platoon vehicles.

1) Vehicle Model: For our system, we are only interested in tracking the states of vehicles in front of us. All vehicles ahead of us are represented by six state variables: east and north position coordinate in the ENU frame p_e^i and p_n^i, longitudinal velocity v_x^i, longitudinal acceleration a_x^i, heading (or yaw-angle) ψ^i and yaw-rate ω^i, where the index i denotes the position in the platoon. The process model equations in this case are

$$\begin{aligned} \dot{p}_e^i &= v_x^i \sin\psi^i, & \dot{p}_n^i &= v_x^i \cos\psi^i \\ \dot{v}_x^i &= a_x^i, & \dot{a}_x^i &= 0 \\ \dot{\psi}^i &= \omega^i, & \dot{\omega} &= 0 \end{aligned} \tag{7}$$

The measurement equations are

$$\begin{aligned} p_e^{i,w} &= p_e^i + e_e^i, & p_n^{i,w} &= p_n^i + e_n^i \\ v_x^{i,w} &= v_x^i + e_{vx}^i, & a_x^{i,w} &= a_x^i + e_{ax}^i \\ \psi^{i,w} &= \psi^i + e_\psi^i, & \omega^{i,w} &= \omega^i + e_\omega^i \end{aligned} \tag{8}$$

The measurement noise vector e_k is given by $e_k = [e_e^i, e_n^i, e_{vx}^i, e_{ax}^i, e_\psi^i, e_\omega^i]^T$, where the superscript w denotes that the source of this information is the wireless communication. The noise covariance matrices are used as tuning parameters in the Kalman filters. They have been assigned a diagonal structure, which corresponds to that all noise sources are uncorrelated.

F. Object Management

There are three types of objects that need to be tracked in the estimator: vehicles, speed signs and traffic lights. Besides tracking and estimating a number of parameters for each object, the estimator also need to order, label and decide which of them are important or not in order to provide the correct information to the controller. In this section we will describe

how these objects are managed by starting with describing the relative base distance (see next section). All objects, except the vehicle ahead, are forgotten by the estimator if they have not been transmitting any information over the last 3 seconds.

1) Relative Base Distance: The relative base distance b is a distance derived from the host vehicle to any object. It is relative because it is signed: it is negative if the object is behind the host vehicle and positive if it is in front. We refer to it as a base distance because it is the position of the object projected on the line in which the host vehicle is traveling. The line bearing φ is the angle to the object relative the heading of our vehicle. See Figure 6 for an example of relative base distance and line bearing.

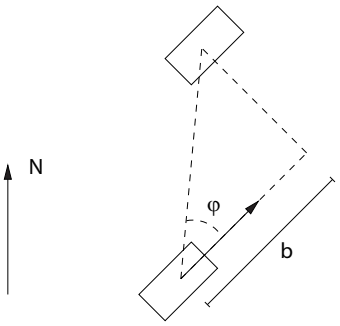


Figure 6: Relative base distance b. Line bearing φ.

2) Vehicle ahead: The Vehicle Ahead is defined as the vehicle that is immediately in front of the host vehicle. To find out which vehicle is placed immediately in front of us, we sort a list of all known vehicles and pick the vehicle with smallest positive relative base distance as the vehicle ahead. In order not to identify vehicles in adjacent lanes we also sort out the vehicles that have line bearing within $\pm$ 10 degrees and that have the same heading as our own vehicle (within some tolerance). The vehicle ahead has a critical importance in the control of host vehicle since we want to stay as close as possible to it, which is why it also deserves special treatment in the estimation process. The idea here is that the accuracy and reliability of the information of the vehicle ahead can be significantly improved by fusing its estimated data with the data from the radar. Data from the radar gives the following information about the vehicle ahead: the relative distance, relative speed compared to the host vehicle and absolute acceleration. Let X_{ctrl} be the vector of the control states containing the relative base distance, absolute speed and absolute acceleration estimates from the EKF, and X_{rdr} be the vector containing the same variable measured by the radar. These two vectors have covariances matrices P_{ctrl} and P_{rdr} respectively. They can be fused together by using the formula given in [36, p. 29-30],

$$X_{fus} = (P_{ctrl}^{-1} + P_{rdr}^{-1})^{-1}(P_{ctrl}^{-1}X_{ctrl} + P_{rdr}^{-1}X_{rdr}), \qquad (9)$$

where X_{fus} is the fused vector having covariance matrix $(P_{ctrl}^{-1} + P_{rdr}^{-1})^{-1}$. The covariance matrix P_{ctrl} is obtained by taking the differences of the position and velocity state covariances for the own vehicle and the vehicle in front, and using the term for the longitudinal acceleration in the state covariance matrix of the speed estimation filter. The covariance P_{rdr} is obtained from the measurement inaccuracies, given in the radar data sheet.

The implementation of the process of estimating vehicles is shown in Figure 7.

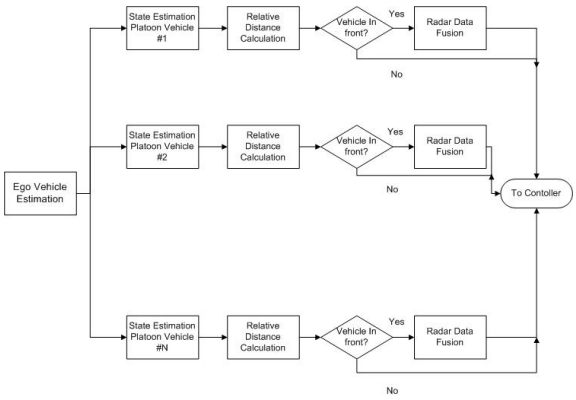


Figure 7: Separate estimation model representation

3) Platoon Members: Of all the members of our platoon, the controller is only interested in platoon member vehicles in front of the host vehicle. To find the platoon members in front of the host vehicle, the vehicles in front that have the same platoon ID as the host vehicle are picked out, sorted by relative base distance, and passed on to the controller.

4) Speed Signs: The speed sign information is also sent to the controller. For these, the current speed and the relative base distance to a speed change along with the speed is forwarded to the controller. No other state estimations are done here.

5) Traffic Lights: The traffic lights send information about when the change of color is going to be made and to which color, e.g. "in 2 seconds I will turn green". This information is processed in the estimator and handed to the controller. In case of many traffic lights, only the closest traffic light ahead of the host vehicle is considered.

G. Summary and conclusions

State estimation can be characterized in two distinct stages. In the first stage, the states required to be sent to the other vehicles are estimated together with some other additional states. This is implemented as a cascade of three Kalman filter, as shown in Figure 5. The lateral dynamics of the vehicle, e.g. side slip and lateral velocity, are estimated via the bicycle model. The filters are fed with data from the vehicle's CAN bus and the RTK-GPS. The platoon vehicle estimation is the second stage of the state estimation process. This stage is implemented in form a parallel Kalman filter bank, with each vehicle being estimated by one of the filters, individually, see Figure 7. They are fed by the data from the wireless interface and the estimated states of the host vehicle. The state model equations for each vehicle are similar to the ones of a simple two dimensional kinematic model. Each time a new vehicle joins the platoon, a new filter has to be initiated for that vehicle. Likewise, when a vehicle leaves the platoon, its filtering operation has to be aborted. Each filter has a state vector and a state error covariance matrix that has to be stored

and kept track of. For the vehicle ahead, additional protection is provided by incorporating the data from the onboard radar. The performance of the state estimation clearly depends on the quality of the available data, for example the outage duration of the GPS measurements and of the wireless communication (from the other vehicles) and the accuracy of the measurements in general. The accuracy of the process models of course also influence the state estimation. For example, as no information about the lateral dynamics of the platoon vehicles is available, the filters track the vehicles' motion in a less precise manner when their lateral motion is significant, e.g., when undergoing sudden turns.

V. Control

With the aid of vehicle to vehicle (V2V) communication, vehicle characteristics and events such as harsh braking can be transmitted within the range of the wireless channels. This local information can be utilized as inputs for an automated control strategy to govern the vehicles at close intermediate spacing, far less than what is feasible for a human driver due to their relatively high reaction time. As intermediate spacing is reduced, the control must be increasingly stringent due to safety concerns. Driver comfort considerations are also important and may put a limit on the minimal distance between vehicles [37].

In this section we introduce the control objectives and requirements that are emphasized in the control design. A control system architecture is presented to decompose the control problem into manageable subsystems. A system model is described, which is used as a basis for the control design. Finally, a robustness analysis is presented along with experimental results.

A. Control objectives and requirements

To ensure the performance for a single vehicle within a platoon, the controller must handle several different scenarios. Information regarding traffic light states is facilitated through wireless transceivers, which are mounted on road side units. The controller must be able to deduce whether the vehicle will be able to pass the traffic light before it turns red or if it must come to a halt. Another performance criterion is that a (sub-)platoon must be able to merge with another platoon if that scenario arises or split from the platoon if deemed necessary. Hence, a single vehicle should adapt its velocity with respect to several situations. Furthermore, due to safety concerns the vehicles are not allowed to exceed the given road speed limit. The driver must also be allowed to override the system if necessary. Therefore, the system must shut off if the throttle or brake pedal is pressed or if an emergency button is activated.

In addition to performance criteria, several constraints are set upon the system. Only longitudinal control is mandated. The driver should at all times be active and control the lateral movement of the vehicle. Vehicles are not allowed to enter or exit the platoon from the side. Finally, maximum acceleration and deceleration constraints are set upon the system due to vehicle heterogeneity.

Table I: Table of the specified signals required to provide the vehicle specific information span.

Signal:	Description:
Position	Absolute value as given by the GPS.
Position accuracy	An estimate provided in meters within the 95 % confidence interval.
Speed	Derived from the GPS or the onboard tachometer.
Longitudinal acceleration	Absolute value.
Heading	To distinguish the platoon from oncoming traffic.

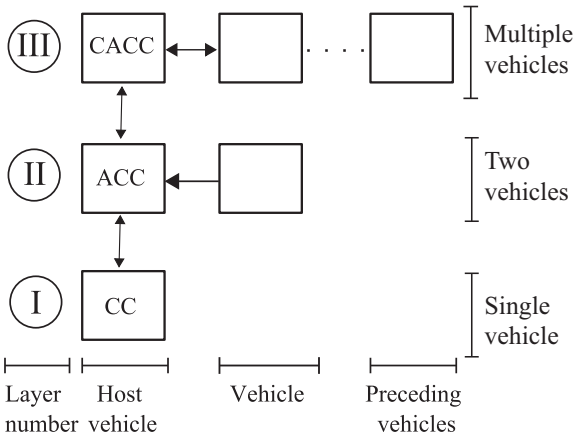


Figure 8: Control system architecture.

A standard has not yet been set regarding what information should be available through V2V communication. However, the signals listed in Table I are considered here and is seen as mandated to monitor the local vehicle behavior. A maximum information span of ten vehicles is taken into account. In the event of a communication failure, the controller must still produce a robust control strategy. Therefore, the system always keeps track of the vehicle ahead, regardless of which platoon that vehicle belongs to.

B. Control architecture

The need for a carefully developed control architecture has been studied, for example in [38] and in [5]. The control system relies on the underlying information span and its structure will vary if the range of inputs changes. To divide the problem under consideration into manageable subsystems for control design, a three layer control system is designed as depicted in Figure 8. Each layer is assumed to have access to the information from the road side units at all time. The differentiating factor between each layer is the availability of platoon information.

Starting from the bottom, layer I includes control challenges for a single vehicle when no surrounding traffic is taken into consideration. The controller can simply use the onboard cruise controller (CC) to maintain a given reference velocity. However, an amendment to the conventional CC is the ability to handle traffic light information.

In layer II, information regarding an immediate preceding vehicle is introduced through radar functionality. The control

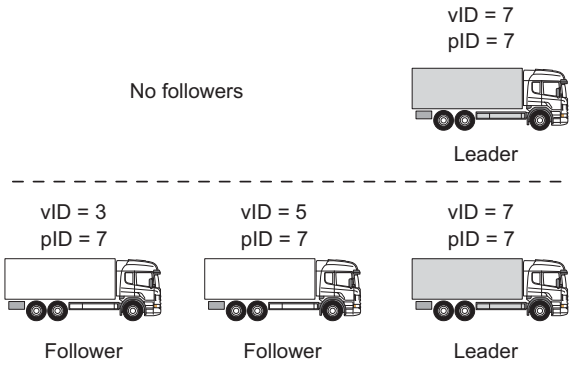


Figure 9: Every vehicle has a unique vehicle ID (vID), while its platoon ID (pID) is always set to the leader's vehicle ID.

objective in this layer is to maintain a given relative distance and velocity to a vehicle ahead. The set of controllers that we are referring to in this layer is for example the conventional adaptive cruise controller (ACC).

Lastly, in layer III interaction between the vehicles in the platoon is introduced. The controller challenge and objective in this layer is to form a cooperative adaptive cruise control (CACC) strategy. The controller must be able to handle the varying range of vehicle information and the platoon logic. In the platoon logic every vehicle has a platoon ID and a unique vehicle ID, which are broadcasted periodically. When taking the role of a platoon leader, the platoon ID shall be set to the own vehicle ID, as shown in the top part of Figure 9. Similarly, when acting as a follower, the platoon ID shall be set to the leader's vehicle ID, which is illustrated in the lower part of Figure 9.

As we move up in the architecture layers, the controller complexity increases. However, the controllers in each layer can be designed independently and incorporate different modes depending on the underlying control input span. The presented architecture allows for a graceful degradation. If the communication node of the host vehicle fails, the control degrades to layer II, where the radar functionality still enables an automated control for platooning. Furthermore, if the radar system fails, the driver is instructed to take full control of the system.

C. System Model

The state equation of a single vehicle in a platoon is [39]

$$\begin{aligned} m_t\dot{v} = F_{engine} - F_{brake} - F_{airdrag}(v) \\ - F_{roll}(\alpha) - F_{gravity}(\alpha) \end{aligned} \tag{10}$$

where v is the vehicle velocity and m_t denotes the accelerated vehicle mass. The modeled forces in (10) that act on a vehicle in motion are internal forces consisting of the engine and the brake force and the external forces being the air drag, roll friction and gravity. For simplicity we assume that the road slope α is constant and we lump together the effects of roll friction and gravity into one constant k_f. The air drag is modeled as $F_{airdrag} = k_d v^2$, where k_d is the vehicle specific air drag coefficient. We combine the engine and brake forces and let the control signal be $u = F_{engine} - F_{brake}$. Now the model (10) is written as

$$m_t\dot{v} = -k_d v^2 - k_f + u \tag{11}$$

By introducing new variables $\tilde{v} = v - v_o$ and $\tilde{u} = u - u_o$ that are the deviations from an equilibrium point $\{v_o, u_o\}$ of (11), the linearized model is then given by

$$\dot{\tilde{v}} = a\tilde{v} + b\tilde{u} \tag{12}$$

where $a = -2k_d v_o/m_t$ and $b = 1/m_t$.

All vehicles in the platoon will be assumed to have linear dynamics according to (12) and a sub-index will be used on all variables and parameters to associate them with a specific vehicle in the platoon. (The vehicles will of course not have identical dynamics, but the information about individual dynamics is not available, why this assumption is made.) The lead vehicle has index 1 and the host vehicle has index i. To establish a simplified system model as a premise for control design, it is assumed that all other vehicles control their velocities based only on the own velocity and the velocity of the vehicle directly ahead. In particular, we consider the control $\tilde{u}_k = \beta_k\tilde{v}_k + \gamma_k\tilde{v}_{k-1}$ for $k = 1, ..., i-1$, where $\tilde{v}_o \equiv 0$ (since the lead vehicle does not have a vehicle in front of it). The model of the entire platoon is

$$\begin{bmatrix} \dot{\tilde{v}}_1 \\ \dot{\tilde{v}}_2 \\ \vdots \\ \dot{\tilde{v}}_{i-1} \\ \dot{\tilde{v}}_i \\ \dot{d}_{i,i-1} \end{bmatrix} = \underbrace{\begin{bmatrix} \theta_1 & 0 & \dots & 0 & 0 & 0 \\ \eta_2 & \theta_2 & \dots & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \eta_{i-1} & \theta_{i-1} & 0 & 0 \\ 0 & 0 & \dots & 0 & a_i & 0 \\ 0 & 0 & \dots & 1 & -1 & 0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \tilde{v}_1 \\ \tilde{v}_2 \\ \vdots \\ \tilde{v}_{i-1} \\ \tilde{v}_i \\ d_{i,i-1} \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ b_i \\ 0 \end{bmatrix}}_{B} \tilde{u}_i \tag{13}$$

where $\theta_k = a_k + b_k\beta_k$ and $\eta_k = b_k\gamma_k$. The model also includes the state variable $d_{i,i-1}$, which is the relative distance between vehicle i and $i-1$. In the following analysis it will be assumed that all other vehicles have identical dynamics and identical controllers. It will also be assumed that the controllers are designed so that all vehicles have the same velocity in steady state. This means that $\theta_k = \theta$ and $\eta_k = -\theta$ for all k.

D. Control Design

The controller structure varies depending on whether the vehicle is alone or in a platoon. When the vehicle is not in a platoon, it operates through the CC with the given road speed as the reference and acts on traffic lights. Information regarding traffic light state is received through V2I communication. The received information contains the traffic light position, current status, next and second next light, time to next and second next light.

To ensure that the vehicle does not cross the traffic light while red, the controller in layer I, Figure 8, checks if the traffic light is green and calculates if the vehicle can pass with its current speed as given by

$$t_{red} > \frac{d_t - d_{offset}}{v} \tag{14}$$

where d_t is the distance to the traffic light with a stopping offset d_{offset}, v is the vehicle speed and t_{red} is the time when the traffic light turns red. If (14) is not satisfied, the controller deems that the vehicle has to stop at the traffic light. Then an appropriate deceleration trajectory is calculated based upon the distance to the traffic light and the current speed required to stop at the desired distance from the traffic light. Starting from initial velocity v_i with constant acceleration a gives at time t the velocity $v(t)$ and the travelled distance $d(t)$ according to

$$v(t) = v_i + at, \quad d(t) = v_i t + \frac{at^2}{2} \tag{15}$$

Squaring the first equation gives

$$\begin{aligned} v^2(t) &= (v_i + at)^2 \\ &= v_i^2 + 2v_i at + a^2 t^2 \\ &= v_i^2 + 2ad(t) \end{aligned} \tag{16}$$

In this case $v(t) = 0$, $v_i = v$ and $d(t) = d_t - d_{offset}$. Thus, the acceleration needed to stop is given as

$$a = -\frac{v^2}{2 \cdot (d_t - d_{offset})} \tag{17}$$

Hence, the control task in this layer is to maintain a given velocity reference through the onboard CC or to adapt the velocity with respect to the upcoming traffic light state. In this layer, the vehicle is assumed to act as a platoon leader.

In layer II, Figure 8, the onboard ACC is utilized to govern the vehicle and mainly serves as a precautionary control strategy in case of wireless communication failure. However, the control objectives in layer III mandate a more advanced strategy. Here, the controller task is to maintain an appropriate velocity with respect to all the preceding vehicles in the local platoon and a given intermediate spacing with respect to the immediate vehicle ahead.

When the vehicle is a follower in the platoon, a linear quadratic regulator (LQR) control is utilized, where the controller reacts on deviations from the given relative distance to the vehicle ahead as well as for relative speed of all the preceding vehicles. The controller is derived analytically by minimizing the cost function

$$\begin{aligned} J^* = \min_u \int_0^\infty x^T Q x + u^T R u \\ \text{s.t.} \quad \dot{x} = Ax + Bu \end{aligned} \tag{18}$$

where $Q \geq 0$, $R > 0$ are weight matrices and A, B are system matrices. These matrices are of varying size based upon the number of preceding vehicles in the platoon. u denotes the control input and x is the state vector that entails the speed of the host vehicle, the relative distance to the closest preceding vehicle and the velocity of preceding vehicles in the platoon. For general LQR-design the weighting factors need to be specified and adjusted based upon the results of the specified design goals [40]. The lead vehicle's objective is to follow a given reference velocity. However, the follower vehicles in the platoon have an additional objective of maintaining the set intermediate distance. The desired relative distance is set to vary depending on the vehicle velocity. It is determined by setting a time gap τ s, which gives the desired headway as $d_{ref} = \tau v_i + d_{min}$ for the host vehicle, where d_{min} is the minimum allowed intermediate spacing. Thus, considering the platoon objectives, the cost function can be set up as

$$\begin{aligned} J(u_i^*) = &\min_{u_i} \int_{t_0}^{t_f} w_i^\tau (d_{(i-1)i} - d_{ref})^2 + \sum_{j \in \mathcal{N}} w_j^{\Delta v} (v_j - v_i)^2 \\ &+ w_i^d d_{(i-1)i}^2 + w_i^v v_i^2 + w_i^{u_i} u_i^2 dt \\ = &\min_{u_i} \int_{t_0}^{t_f} x^T Q x + R_i u_i^2 + w_i^\tau d_{min} dt \end{aligned} \tag{19}$$

where $\mathcal{N}$ denotes the set of preceding vehicles and

$$Q = \begin{bmatrix} Q_1 & Q_2 \\ Q_3 & Q_4 \end{bmatrix}, \quad R = w_i^{u_i}, \tag{20}$$

are cost matrices. The submatrices, $Q_1, \ldots, Q_4$, varies with the size of the number of preceding vehicles and are given as

$$\begin{aligned} Q_1 &= \begin{bmatrix} w_1^{\Delta v} & 0 & \ldots & -w_1^{\Delta v} \\ 0 & \ddots & 0 & -w_2^{\Delta v} \\ \vdots & 0 & w_{i-1}^{\Delta v} & \vdots \\ -w_1^{\Delta v} & -w_2^{\Delta v} & \ldots & \tau^2 w_i^\tau + w_i^v + \sum_{j \in \mathcal{N}} w_j^{\Delta v} \end{bmatrix}, \\ Q_2 &= \begin{bmatrix} 0 & \ldots & 0 & -\tau w_i^\tau \end{bmatrix}^T, \\ Q_3 &= Q_2^T \\ Q_4 &= \begin{bmatrix} w_i^d + w_i^\tau \end{bmatrix}, \end{aligned} \tag{21}$$

In accordance with the objective for a vehicle traveling in a platoon, w_i^τ in (19) determines the importance of not deviating from the desired time gap and $w_j^{\Delta v}$ creates a cost for deviating from the velocity of the preceding vehicles. The following terms, $w_i^d, w_i^v, w_i^{T_i}$, put a cost on the deviation from the linearized states and the control input. Since the main objective is to maintain a set intermediate distance, w_i^τ must be set larger than the remaining weights. However, there is an inherent constraint on the weight parameter, $w_i^{u_i}$, on the control input such that it remains within the physical constraints set within the system.

After setting the weighting parameters, the optimal control input, u^*, can then be derived analytically by solving a Riccati equation

$$\begin{aligned} u_i^* &= -Lx = -(l_1 v_1 + l_2 v_2 + \cdots + l_i v_i + l_d d_{i-1,i}), \\ L &= R^{-1} B^T P, \\ 0 &= PBR^{-1}B^T P - A^T P - PA - Q. \end{aligned} \tag{22}$$

The controller gains, $l_1, l_2, \ldots, l_i$, are derived for a range up to nine preceding vehicles (ten including the host vehicle) and a gain scheduling control is formed based upon several different equilibrium points and platoon lengths.

E. Robustness and string stability

In a platoon, string stability is an important robustness criterion [7], [8], [13], [41]. See also [17, Section A.2] for a nice overview of how the string stability concept is used in the GCDC. String stability is a measure on how good the vehicles in a platoon are at subduing disturbances introduced by preceding vehicles. A system in which a change of velocity of the lead vehicle can be amplified and propagated throughout the platoon is not considered to be string stable. This disturbance amplification could eventually lead to vehicle collisions, hence it is of most importance to have a string stable controller.

First, let the relation between two subsequent vehicles be described as

$$V_k(s) = G_k(s)V_{k-1}(s) \tag{23}$$

where $G_k(s)$ is the transfer function from $k-1$:th to k:th vehicle and $V_k(s) := \mathcal{L}(v_k(t))$ is the Laplace transform of the time domain velocity. The string stability criterion can then be described as [41]:

$$\|G_k(s)\|_\infty \leq 1 \tag{24}$$

where $k \in [2, N]$, where the platoon consists of N vehicles and where $\|G_k(s)\|_\infty = \max_\omega |G_k(j\omega)|$. This states that a disturbance in the velocity of a preceding vehicle cannot be amplified. The above analysis is valid for the case when the velocity of one vehicle only depends on the velocity of the next vehicle ahead. If all vehicles fulfill (24) then any disturbance injected in the platoon will be attenuated when it propagates from vehicle to vehicle. The first $i-1$ vehicles in the model (12) can be analyzed in this way. The transfer function from one vehicle to the next is given by $G_k(s) = \eta_k/(s-\theta_k)$, which fulfills (24) if and only if $\theta_k < 0$ and $|\eta_k| \leq |\theta_k|$.

The controller (22) that we propose for our vehicle, however, depends on the velocity of all preceding vehicles in the platoon and the dynamics of our vehicle can be written as

$$V_i(s) = \sum_{k=1}^{i-1} H_k(s)V_k(s) \tag{25}$$

for some transfer functions H_k. It would be tempting to think that string stability was ensured if $\|H_k\|_\infty \leq 1$ for all k, but we also have to take into account that there are couplings between the other vehicles and that the disturbance in one vehicle will propagate to the other vehicles. Instead, we analyze the situation based on the entire platoon model (12) by applying the control (22) and adding a velocity disturbance δv_k at vehicle number k. The closed loop system from δv_k to v_i is $G(s) = C_{cl}(sI - A_{cl})^{-1}B_{cl}$, where $A_{cl} = A - BL$, $B_{cl} = \begin{bmatrix} 0 & \cdots & \theta_k & \eta_{k+1} & \cdots & 0 & -b_i l_k & 0 \end{bmatrix}^T$ and C is the row vector that extracts the state for the host vehicle velocity, $\tilde{v}_i$. In our case, we obtained $\|G(s)\|_\infty \leq 1$ for all k for all proposed LQR controllers.

It should be remarked that the analysis on string stability above does not guarantee stability of the real-world system. Additional simulations and tests are required to obtain sufficient confidence in the string stability in practice. In the nest section some in-vehicle evaluations are described.

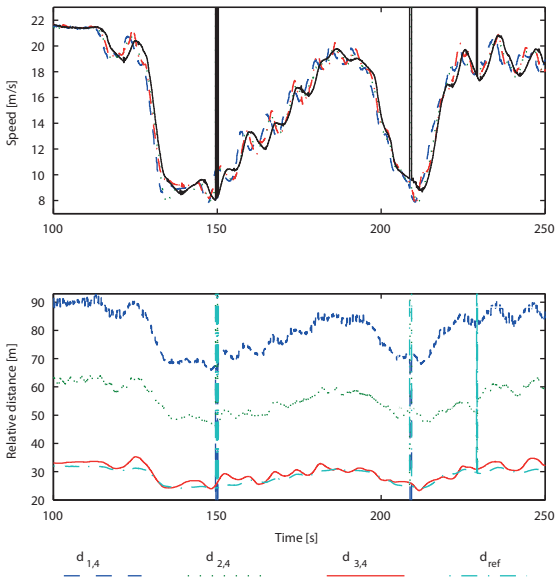


Figure 10: Trajectories for a vehicle platoon consisting of four vehicles. The top plot contains the velocity trajectories for each vehicle. The host vehicle (index 4, solid black line) is the fourth vehicle in the platoon. The lead vehicle's velocity trajectory (dashed blue line) has subindex 1, the second vehicle (dotted green) has subindex 2, and the third vehicle (dashed-dotted red) has subindex 3. The bottom shows the relative distance between the host vehicle and the preceding vehicles, with the subindices assigned accordingly.

F. Evaluation

When studying the behavior of vehicles in a finite platoon, the velocity should not deviate significantly from the lead vehicles velocity trajectory. Several race heats were conducted in which the proposed controller performance was thoroughly evaluated. The races consist of several different challenging tasks and a varying number of preceding heterogeneous vehicles. Data collected from one of the heats are presented in Figure 10.

The top plot presented in Figure 10 shows the velocity trajectories of the preceding vehicles and the host vehicle. The bottom plot shows the relative distance between the host vehicle and three preceding vehicles. The presented measurements in the plots was received through V2V communication and then filtered through the on-board state estimator. The first scenario displayed between the 100 s - 140 s time span shows the results from evaluating a common harsh braking scenario. In this scenario the lead vehicle initiates a semi harsh braking and then accelerates. This is then followed by a much harder braking; a scenario common in traffic jams. The results show that the velocities of all the follower vehicles in the platoon do not deviate significantly compared to the lead vehicles velocity. The intermediate spacing is also maintained and the relative distance reference to the vehicle in front of the host vehicle is tracked fairly accurately. The decrease in relative distance during a deceleration is due to the time varying distance reference. Note that the vertical lines seen in Figure 10 at the 150 s and 210 s time marker occurs due to information loss in

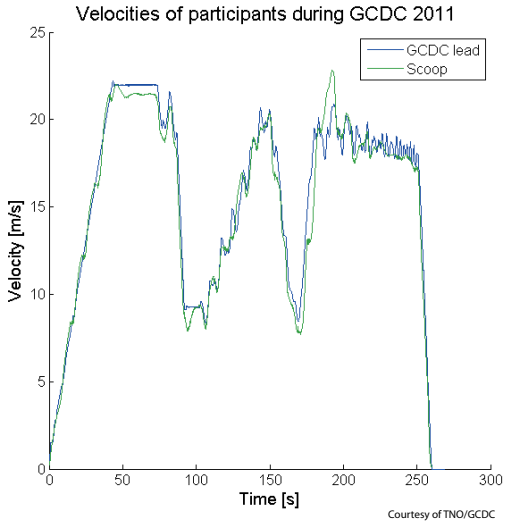


Figure 11: GCDC heat when the Scoop vehicle is following directly behind the GCDC lead vehicle.

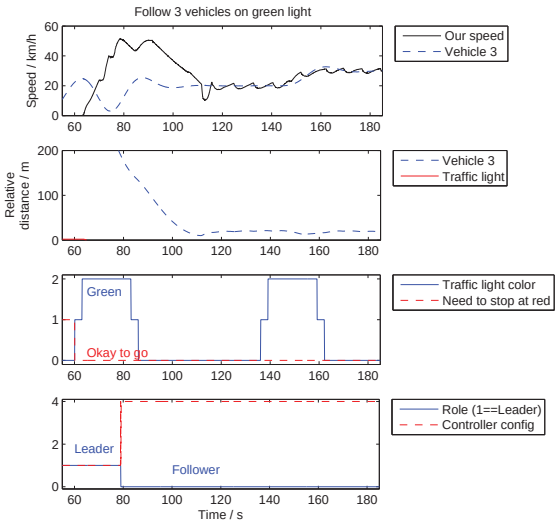


Figure 12: Evaluation of the ability to perform automatic starts, obey traffic light and join platoons. The test is performed in the real vehicle but the other vehicles are virtual simulation objects.

the wireless package transmission.

The entire duration of another heat is depicted in Figure 11. In that run the Scoop vehicle was the first vehicle, directly following the GCDC lead vehicle. Two things are clearly visible: Fast oscillations are attenuated, (see for example between 100 s and 150 s and after 200 s), which is a desired stable behavior. The frequent and quite slow gear changes can also be seen clearly, e.g. during the initial acceleration phase and also at 180 s. This is an inherit limitation of a heavy-duty vehicle that never can be as agile as a passenger car.

The next evaluation that is presented here is an example of the automatic start from traffic light and the catching up and joining of a platoon. The test is performed in the real vehicle but the other vehicles and the traffic light are represented by virtual simulation objects. The results are shown in Figure 12 (note that only the vehicle directly ahead is shown, not the two leading vehicles).

When going from a green light there might be vehicles ahead. In this test case, the truck drives on green light and catches up with a platoon of three vehicles driving at around 20 km/h. When the vehicle ahead is 190 m away, the platooning role goes from 'leader' to 'follower' and the controller configuration goes from 1 (leader) to 4 (fourth in line), causing the truck to follow the three vehicles. Once the truck catches up with the platoon ahead, the distance to the closest vehicle varies between 13 and 21 m, even when the platoon leader increases the speed from 20 to 30 km/h. As a result, the truck tries to keep up with the leader and accelerates even before the vehicle directly ahead of us does. Note that these test were performed with an earlier version of the vehicle tracking controller.

Finally, it will be shown how the system behaves when a red light appears that forces the vehicle to brake and split from the platoon. This is shown in Figure 13. Stopping at a red light should result in two things: coming to a halt and splitting the platoon. When the vehicle approaches the traffic light, the system is requested to decelerate. When the vehicle

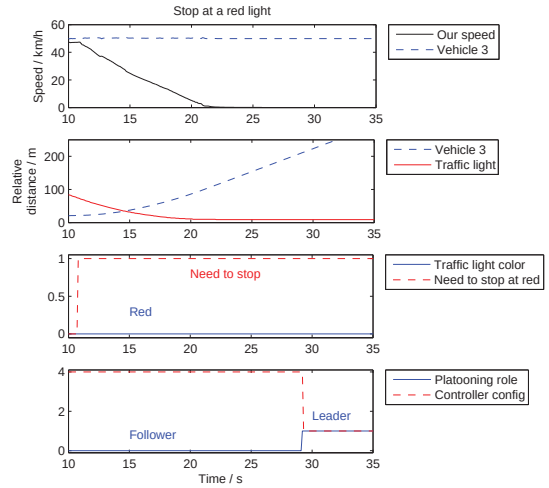


Figure 13: Evaluation of the ability to stop when the traffic light turns red and to split from the platoon. The test is performed in the real vehicle but the other vehicles are virtual simulation objects.

ahead reaches a relative distance of 210 m (at t = 29 s), the platooning logic makes the decision to become platoon leader and the controller configuration changes from 4 to 1 (i.e from being the fourth vehicle to being the first).

VI. Summary and Conclusions

Team Scoop has developed a prototype system for cooperative adaptive cruise control of a heavy-duty vehicle and successfully participated in the first edition of the Grand Cooperative Driving Challenge. Within the project we have

1) Designed and implemented an architecture for the hardware and software of the system that interfaces with the vehicle and runs all algorithms

2) Implemented and analyzed the communication and interaction protocols that enable the V2V and V2I communication
3) Designed and implemented models and algorithms for state estimation and sensor fusion of the necessary variables of the host vehicle and of the fellow platoon vehicles
4) Designed, implemented and analyzed algorithms for maintaining the distance to the vehicle(s) ahead, for automatic start and stop functionality, for automatic starts and stops at traffic lights, and for joining and splitting of platoons.

A. Future work

Many aspects of cooperative driving systems have been studied in the development of our vehicle for the GCDC 2011, but there are still many interesting issues that deserve more attention in the future. Some suggestions are:

- *Improved vehicle tracking.* The LQR approach that was used here could definitely be developed further, for example by adding integral action on the relative distance to the vehicle ahead. The finite-time horizon version could be employed which would mean solving Riccati equations online, or we could use time-varying models. Another interesting option would be to use MPC (model predictive control) to incorporate constraints in the optimization and possibly also using nonlinear models.
- *Improved state estimation.* Digital maps and georeferencing techniques could be used to improve the tracking of the own vehicle and all other traffic objects. This should for example help in determining which lane other vehicles are traveling in.
- *Non-communicating vehicles.* In a real traffic scenario there will be non-equipped vehicles that could interfere and that has to be handled in some way. If such a vehicle enters a platoon, the automated vehicles must be able to detect that and back off. The current implementation relies on the radar for this, but perhaps it can be improved by using camera systems.
- *Truly cooperative control.* In GCDC 2011, all vehicles operated individually, based on information provided by the other vehicles and the infrastructure. For more advanced tasks there is the need of having negotiation protocols that enable the vehicles to agree on which actions are to be performed.

B. Conclusion

During the GCDC competition we have shown, together with many of the other teams, that cooperative adaptive cruise control is manageable and feasible for implementation in realistic traffic situations. Different vehicle types, with different and individually developed prototype systems, operate together in what has been an excellent showcase for the potential of cooperative mobility. We believe that our developed system will be useful in future research projects and educational activities at KTH and Scania and that it will serve as a platform for our future developments of cooperative vehicular ITS systems. In particular, we hope that it will be the starting point for the development of a vehicle for the next Grand Cooperative Driving Challenge.

ACKNOWLEDGEMENTS

This project was funded by the Swedish Transport Adminstration within the IVSS programme under contract *TRV 2010/55777, CoAct*. Additional financial support was provided from the KTH Transport Platform, the ACCESS Linnaeus Centre and the Innovative Centre for Embedded Systems (ICES), all at KTH Royal Institute of Technology, Stockholm, Sweden, and from Scania CV AB, Södertälje, Sweden. The work has been carried out in association with SAFER - Vehicle and Traffic Safety Centre at Chalmers University of Technology, Göteborg, Sweden.

The authors would like to thank the other members of team Scoop; Elin Stålklinga, Mattias Björk, Liliana Garcia Alonso, Farzad Khaksari and Rickard Lyberger. Team Scoop would also like to acknowledge the two other Swedish GCDC teams from Chalmers University of Technology and Halmstad University, who also were funded by the CoAct project and with which we conducted valuable validation tests before the competition.

Finally, team Scoop wishes to thank the following people for their help, guidance or support; Tony Sandberg, Martin Törngren, Karl Henrik Johansson, Magnus Jansson, John-Olof Nilsson, Fredrik Asplund, Dave Zachariah and Meng Guo.

REFERENCES

[1] European Commission, *Panorama of transport*. Eurostat - Statistical Books, 2009.
[2] ——, "Roadmap to a single european transport area - towards a competitive and resource efficient transport system," White Paper COM(2011) 411 final, 2011.
[3] World Business Council for Sustainable Development, "Mobility 2030: Meeting the challenges to sustainability," 2004.
[4] F. Browand, J. McArthur, and C. Radovich, "Fuel saving achived in the field test of two tandem trucks," California PATH Research Report, Tech. Rep. UCB-ITS-PRR-2004-20, 2004.
[5] A. Alam, *Fuel-Efficient Distributed Control for Heavy Duty Vehicle Platooning*. Stockholm, Sweden: Licentiate thesis TRITA-EE 2011:059, Royal Institute of Technology, 2011.
[6] P. Ioannou and C. Chien, "Autonomous intelligent cruise control," *Vehicular Technology, IEEE Transactions on*, vol. 42, no. 4, pp. 657 –672, nov 1993.
[7] D. Swaroop and J. Hedrick, "String stability of interconnected systems," *Automatic Control, IEEE Transactions on*, vol. 41, no. 3, pp. 349 –357, mar 1996.
[8] D. Swaroop and J. K. Hedrick, "Constant spacing strategies for platooning in automated highway systems," *Journal of Dynamic Systems, Measurement, and Control*, vol. 121, no. 3, pp. 462–470, 1999. [Online]. Available: http://link.aip.org/link/?JDS/121/462/1
[9] B. De Schutter, T. Bellemans, S. Logghe, J. Stada, B. De Moor, and B. Immers, "Advanced traffic control on highways," *Journal A*, vol. 40, no. 4, pp. 42–51, Dec. 1999.
[10] A. Bose and P. A. Ioannou, "Analysis of traffic flow with mixed manual and semiautomated vehicles," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 4, pp. 173–188, 2003.
[11] B. Van Arem, C. J. G. van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 7, pp. 429–436, 2006.
[12] D. Kim, S. Moon, J. Park, H. J. Kim, and K. Yi, "Design of an adaptive cruise control / collision avoidance with lane change support for vehicle autonomous driving," *Strategy*, pp. 2938–2943, 2009.

[13] G. Naus, R. Vugts, J. Ploeg, M. van de Molengraft, and M. Steinbuch, "String-stable cacc design and experimental validation: A frequency-domain approach," *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 9, pp. 4268 –4279, nov. 2010.

[14] G. Naus and J. Ploeg, "Cooperative adaptive cruise control , design and experiments," *Focus*, vol. 1, no. 1, pp. 6145–6150, 2010.

[15] S. Ukkusuri, Y. Wang, and T. Chigan, Eds., *Special Issue on Exploiting Wireless Communication Technologies in Vehicular Transportation Networks*, ser. IEEE Transactions on Intelligent Transportation Systems, vol. 12, no. 3, 2011.

[16] M. . Sotelo, J. W. C. van Lint, U. Nunes, L. B. Vlacic, and M. Chowdhury, Eds., *Special Issue on Emergent Cooperative Technologies in Intelligent Transportation Systems*, ser. IEEE Transactions on Intelligent Transportation Systems, vol. 13, no. 1, 2012.

[17] GCDC Organisation. (2011) Rules & technology document, version 2.0. TNO. Helmond, the Netherlands. [Online]. Available: http://www.gcdc.net/

[18] J. Kjellberg, "Implementing control algorithms for platooning based on V2V communication," Master's thesis, KTH Royal Institute of Technology, Stockholm, Sweden, Apr 2011.

[19] M. Khaksari, "Analysis of communication architecture of GCDC 2011," Master's thesis, BTH Blekinge Institute of Technology, Stockholm, Sweden, Apr 2011.

[20] M. A. A. Khan, "Specifications and strategies for state estimation of vehicle and platoon," Master's thesis, KTH Royal Institute of Technology, Stockholm, Sweden, Aug 2011.

[21] K.-Y. Liang, "Linear quadratic control for heavy duty vehicle platooning," Master's thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2011.

[22] L. G. Alonso, "Formal requirements specification, cooperative driving system GCDC 2011, use case," Master's thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2011.

[23] [Online]. Available: http://gpsd.berlios.de/

[24] [Online]. Available: http://en.wikipedia.org/wiki/Ntpd

[25] [Online]. Available: http://www.orocos.org

[26] P. Soetens, "A software framework for real-time and distributed robot and machine control," Ph.D. dissertation, Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium, May 2006, http://www.mech.kuleuven.be/dept/resources/docs/soetens.pdf.

[27] [Online]. Available: http://www.xenomai.org

[28] L. Walkin. (2010) Asn1c encoder/decoder software. [Online]. Available: http://lionet.info/asn1c/blog/

[29] O. S. Community. (2011) boost c++ libraries. [Online]. Available: http://www.boost.org/

[30] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[31] T. D. Gillespie, *Fundamentals of Vehicle Dynamics*. Warrendale, PA, USA: SAE, 1992.

[32] J. F. King TinLeung, "Road vehicle state estimation using low-cost gps/ins," *Mechanical Systems and Signal Processing*, vol. 25, no. 6, pp. 1988–2004, 2010.

[33] M. B. David and P. Bradford, "Cascaded kalman filters for accurate estimation of multiple biases, dead-reckoning navigation, and full state feedback control of ground vehicles," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 2, pp. 199–208, 2007.

[34] J. R. David M. Bevly and J. C. Gerdes, "Integrating ins measurement with gps measurments for continuous estimation of vehicle sideslip, roll and tire cornering stiffness," *IEEE Transaction on Intelligent Transport System*, 2006.

[35] J. A. Farrell, *Professional Aided Navigation GPS with High Rate Sensors*. New York: McGraw-Hill, 2008.

[36] F. Gustafsson, *Statistical Sensor Fusion*. Lund: Studentlitteratur, 2010.

[37] A. Alam, A. Gattami, K. H. Johansson, and C. J. Tomlin, "Establishing safety for heavy duty vehicle platooning: A game theoretical approach," in *18th IFAC World Congress*, Milan, Italy, August 2011.

[38] P. Varaiya, "Smart cars on smart roads: Problem of control," *IEEE Transactions on Automatic Control*, vol. 38, no. 2, February 1993.

[39] P. Sahlholm and K. H. Johansson, "Road grade estimation for look-ahead vehicle control using multiple measurement runs," *Control Engineering Practice*, vol. 18, no. 11, pp. 1328 – 1341, 2010.

[40] A. Alam, A. Gattami, and K. H. Johansson, "Suboptimal decentralized controller design for chain structures: Applications to vehicle formations," in *IEEE 50th Annual Conference on Decision and Control and European Control Conference*, Dec. 2011.

[41] Y. Yamamura and Y. Seto, "A study of string-stable ACC using vehicle-to-vehicle communication," in *2006 SAE World Congress*, ser. 2006-01-0348, Apr. 2006.

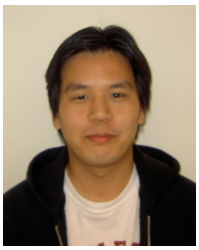
Jonas Mårtensson received his M.Sc. degree in Vehicle Engineering and his Ph.D. degree in Automatic Control from KTH Royal Institute of Technology, Stockholm, Sweden, in 2002 and 2007, respectively. Since 2007 he is a researcher with the School of Electrical Engineering at KTH. His research interests include system identification, experiment design, intelligent transportation systems and modeling and control of internal combustion engines.

Assad Alam received his M.Sc. degree Electrical Engineering from KTH Royal Institute of Technology, Stockholm, Sweden in 2008. Since 2009 he is an industrial PhD student with KTH and Scania CV AB in Södertälje, Sweden. His main research topic is fuel efficient control of heavy-duty vehicle platooning.

Sagar Behere received a Bachelor in Mechanical Engineering from the university of Pune, India in 2003 and a Master in Systems, Control and Robotics from KTH Royal Institute of Technology in Stockholm, Sweden in 2010. Since 2010, he is a doctoral researcher in Mechatronics at KTH. His research interests include real-time programming, autonomous systems, architectures for embedded systems and related applications in the automotive and aerospace domain.

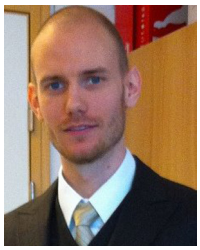
Muhammad Altamash Ahmed Khan is a Master student at the department of signal processing at KTH. He received his B.E.(Electrical Engineering) from National University of Sciences and Technology (NUST), Pakistan in 2006. He is currently working as a research engineer at the department of Automatic control KTH.

Joakim Kjellberg received the M.Sc. degree in Electrical Engineering from KTH Royal Institute of Technology, Stockholm, Sweden, in 2011. His Master's thesis in implementing control algorithms for platooning was conducted at Scania CV AB, where he is presently working as a development engineer with testing of embedded software.



Kuo-Yun Liang received his M.Sc. degree in Electrical Engineering from KTH Royal Institute of Technology, Stockholm, Sweden, in 2011. Since 2011 he is an industrial PhD student in Research and Developement at Scania CV AB in collaboration with the Automatic Control department at KTH. His research is based on heavy duty vehicle platooning.

Henrik Pettersson received his M.Sc and his Ph.D at the Division of Fluid and Mechanical Engineering Systems, Department of Mechanical Engineering at Linköping universitet, Linköping, Sweden. Since 2002 he is working at Scania CV AB with power train control. In resent year the main focus has been on fuel saving functionality and applications based on V2x communication, such as platooning.



Dennis Sundman is a PhD student at the Communication Theory Lab at KTH - Royal Institute of Technology, Sweden, which he joined shortly after receiving his MSc degree in 2008. His research interest is towards compressed sensing in sensor networks where he focus on models, applications and solution algorithms assuming correlated data.