Delft University of Technology

Delft Center for Systems and Control

Technical report 12-008

Hierarchical, intelligent and automatic controls*

B. De Schutter, J. Ploeg, L.D. Baskar, G. Naus, and H. Nijmeijer

If you want to cite this report, please use the following reference instead:

B. De Schutter, J. Ploeg, L.D. Baskar, G. Naus, and H. Nijmeijer, "Hierarchical, intelligent and automatic controls," Chapter 5 in *Handbook of Intelligent Vehicles* (A. Eskandarian, ed.), London, UK: Springer, ISBN 978-0-85729-084-7, pp. 81–116, 2012.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: `https://www.dcsc.tudelft.nl`

*This report can also be downloaded via `https://pub.deschutter.info/abs/12_008.html`

Hierarchical, Intelligent and Automatic Controls

Bart De Schutter*, Jeroen Ploeg#, Lakshmi Dhevi Baskar*, Gerrit Naus†, Henk Nijmeijer†

* Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands, email: b.deschutter@tudelft.nl, lakshmibaskar@gmail.com
# TNO Technical Sciences, Automotive, Helmond, The Netherlands, email: jeroen.ploeg@tno.nl
† Department of Mechanical Engineering, Dynamics and Control section, Eindhoven University of Technology, Eindhoven, The Netherlands, email: g.j.l.naus@tue.nl, h.nijmeijer@tue.nl

Abstract

We present a survey on traffic management and control frameworks for Intelligent Vehicle Highway Systems (IVHS). First, we give a short overview of the main currently used traffic control methods that can be applied in IVHS. Next, various traffic management architectures for IVHS such as PATH, Dolphin, Auto21 CDS, etc. are briefly discussed and a comparison of the various frameworks is presented. Subsequently, we focus on control of vehicles inside a platoon, and we present a detailed discussion on the notion of string stability. Next, we consider higher-level control of platoons of vehicles. Finally, we present an outlook on open problems and topics for future research.

1 Introduction

Intelligent Vehicle Highway Systems (IVHS) (Fenton, 1994; Sussman, 1993; Bishop, 2005) incorporate intelligence in both the roadway infrastructure and in the vehicles with the intention of reducing congestion and environmental impact, and of improving performance of the traffic network, by exploiting the distributed nature of the system and by making use of cooperation and coordination between the various vehicles and the various elements of the roadside infrastructure. IVHS comprise traffic management systems, driver information systems, and vehicle control systems. In particular, vehicle control systems are aimed at developing an automated vehicle-highway system that shifts the driver tasks from the driver to the vehicle (Varaiya, 1993). These driver tasks include activities such as steering, braking, and making control decisions about speeds and safe headways. Automated Highway Systems (AHS), go one step further than IVHS and involve complete automation of the driving task, with the vehicles being organized in platoons. For better (network-wide) coordination of traffic activities, AHS can also distribute the

intelligence between the vehicles and the roadside infrastructure. In this chapter, we will focus on AHS and on the relations and interactions between the vehicles in the AHS inside platoons as well as with the roadside infrastructure. In particular, we will consider the control aspects of these systems.

The currently implemented traffic control and management systems are mainly using intelligence in the roadside infrastructure for controlling and managing the traffic system, However, such a system does not make use of the significant benefits offered by the intelligence — including the additional control, sensing, and communication capabilities — provided by (autonomous) Intelligent Vehicles (IVs). An interesting functionality that is allowed by full automation is to arrange the vehicles in closely spaced groups called "platoons" (Broucke and Varaiya, 1997). To avoid collisions, intra-platoon spacing (i.e., vehicle spacing within a platoon) is kept very small and the inter-platoon spacing is kept larger (Li and Ioannou, 2004; Varaiya, 1993). In the literature, many control frameworks, mainly intended to study inter-vehicle communication technologies and to control the platoon maneuvers in cooperation with the roadside infrastructure, have been developed and investigated (see, e.g., (Hedrick et al., 1994; Rao and Varaiya, 1994; Hsu et al., 1993; Tsugawa et al., 2001)).

In this chapter we discuss various control methods and frameworks for platoons of IVs. After presenting a brief overview of the most frequent control methods that can be used for IVs, some general hierarchical frameworks for control of platoons of IVs are presented. Next, we consider control of vehicles inside platoons, with special attention to string stability. Afterwards, control methods for the higher levels of the control hierarchy are presented. We conclude with an outlook and open issues.

2 Control design methods

In the literature different control methodologies have been presented for controlling and managing traffic networks (Daganzo, 1997; Kachroo and Özbay, 1999; Papageorgiou, 1983). In this section we focus in particular on methods that also apply for platoons of IVs such as

- static feedback control,
- optimal control and model predictive control,
- artificial intelligence (AI) techniques.

The control methods we discuss below operate in discrete time. This means that at each sample time instant $t = kT$ where T is the sampling interval and the integer k is the discrete-time sample step, measurements of the traffic are performed and fed to the traffic controller. The controller then uses this information to determine the control signal to be applied during the next sampling interval.

2.1 Static feedback control

Dynamical systems can be controlled in two ways: using open-loop control and using closed-loop control. In an open-loop system, the control input does not depend on the output of the system, whereas in a closed-loop system, the control action is a function of the output of the system. Feedback or closed-loop control systems are particularly suited for applications that involve uncertainties or modeling errors.

In "static[1]" feedback control methods, the controller gets measurements from the system and determines control actions based on the current state of the system in such a way that the performance of the system is improved. The main examples of static feedback controllers are state feedback controllers and PID controllers (Åström and Wittenmark, 1997).
The general form of a state feedback controller is $\mathbf{u}(k) = \mathbf{L}\mathbf{x}(k)$, where $\mathbf{x}(k)$ is the state vector at time step k, $\mathbf{u}(k)$ the (control) input to be applied at time step k, and $\mathbf{L}$ is the feedback gain matrix. This feedback gain can be computed using, e.g., pole placement.
PID controllers are typically defined in continuous time and for single-input single-output systems, and they are of the form

$$u(t) = K_{\mathrm{p}}\left(e(t) + \frac{1}{T_{\mathrm{i}}}\int_0^t e(\tau)d\tau + T_{\mathrm{d}}\frac{d}{dt}e(t)\right)$$

where $e(t)$ denotes the error signal between the measured output and the set-point value at time t, while the parameters K_{p}, T_{i}, and T_{d} denote respectively the proportional gain, the integral time constant, and the derivative time constant. To determine these parameters several tuning rules exist, such as the Ziegler-Nichols rules.

Static feedback strategies in general do not handle any external constraints. This is a major drawback of this control scheme.

2.2 Optimal control and model predictive control

Now we discuss two dynamic control methods that apply optimization algorithms to determine optimal control actions based on real-time measurements: optimal control and model predictive control.

2.2.1 Optimal control

Optimal control determines a sequence of admissible control actions that optimize a performance function by considering future demands and by satisfying the constraints (Kirk, 1970; Sussmann and Willems, 1997). A general discrete-time optimal control problem contains the following elements:

- dynamical system model equations,
- an initial state $\mathbf{x}_0$,

[1] By "static" we mean here that the control parameters of the feedback controller are taken to be fixed.

- an initial time t_0,
- constraints,
- measurements,
- a performance index J.

More specifically, consider a multi-input multi-output dynamical system expressed by the following equation:

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k), \mathbf{d}(k))$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector, $\mathbf{u} \in \mathbb{R}^m$ the vector of manipulatable control inputs, $\mathbf{f}$ is a continuously differentiable function, and $\mathbf{d}$ is the disturbance vector.

For a given time horizon K, the optimal control problem consists in determining a sequence of control inputs $\mathbf{u}(0), \mathbf{u}(1), \ldots, \mathbf{u}(K-1)$ in such a way that the performance index J takes on the minimum possible value subject to the initial conditions, system dynamics, and constraints, i.e.,

Minimize

$$J = \vartheta[\mathbf{x}(K)] + \sum_{k=0}^{K-1} \varphi\big(\mathbf{x}(k+1), \mathbf{u}(k), \mathbf{d}(k)\big)$$

subject to

$$\begin{aligned}
&\mathbf{x}(0) = \mathbf{x}_0 \\
&\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k), \mathbf{d}(k)) && \text{for } k = 0, \ldots, K-1, \\
&\mathbf{u}_{\min}(k) \leq \mathbf{u}(k) \leq \mathbf{u}_{\max}(k) && \text{for } k = 0, \ldots, K-1, \\
&\mathbf{c}(\mathbf{x}(k), \mathbf{u}(k), k) \leq \mathbf{0} && \text{for } k = 0, \ldots, K-1,
\end{aligned}$$

where ϑ and φ are twice differentiable, nonlinear functions and are called the terminal cost and Lagrangian respectively, $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$ are bounds for the control variables, $\mathbf{c}$ expresses path constraints imposed on the state $\mathbf{x}$ and the control trajectories $\mathbf{u}$ over the period $[t_0, t_0 + KT]$. The disturbance vector $\mathbf{d}$ is assumed to be known over the period $[t_0, t_0 + KT]$. There are two basic approaches to solve the above optimal control problem: calculus of variations (Hestenes, 1966; Gelfand and Fomin, 1991) and dynamic programming (Bellman, 1957).

The main drawback of optimal control is that the method is essentially an open-loop control approach and thus suffers from disturbances and model mismatch errors. Next, we will discuss model predictive control, which uses feedback and a receding horizon approach to overcome some of the drawbacks of optimal control.

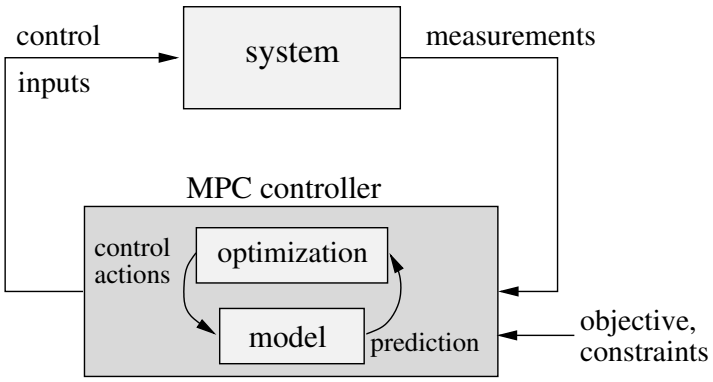

Figure 1: Schematic view of the MPC approach

2.2.2 Model predictive control

Model Predictive Control (MPC) (Maciejowski, 2002; Rawlings and Mayne, 2009) has originated in the process industry and it has already been successfully implemented in many industrial applications. MPC is a feedback control approach that can handle constrained, complex dynamical systems. The main difference between optimal control and MPC is the rolling horizon approach used in MPC (this essentially means that the optimal control is performed repeatedly but over a limited horizon). On the one hand, this results in a suboptimal performance compared to optimal control. However, on the other hand, the rolling horizon approach introduces a feedback mechanism, which allows to reduce the effects of possible disturbances and of model mismatch errors.

The underlying concept of the MPC controller (see Figure 1) is based on on-line optimization and uses an explicit prediction model to obtain the optimal values for the control measures subject to system dynamics and constraints. More specifically, at each control step k the state of the traffic system is measured or estimated, and an optimization is performed over the prediction period $[kT,(k+N_{\mathrm{p}})T]$ to determine the optimal control inputs, where N_{p} is the prediction horizon. Only the first value of the resulting control signal (the control signal for time step k) is then applied to the traffic system. At the next control step $k+1$ this procedure is repeated.

To reduce the computational complexity and to improve stability often a control horizon N_{c} ($< N_{\mathrm{p}}$) is introduced in MPC, and after the control horizon has been passed the control signal is taken to be constant.

There are two loops in MPC: the rolling horizon loop and the optimization loop inside the controller. The loop inside the controller of Figure 1 is executed as many times as is needed to find (sufficiently) optimal control signals at control step k, for the given prediction horizon N_{p}, control horizon N_{c}, currently measured traffic state, and expected demands. The loop connecting the controller and the traffic system is performed once for each control step k and provides the state feedback to the controller. This feedback is necessary to correct for the ever present prediction errors, and to provide disturbance rejection (i.e., compensation for unexpected traffic demand variations). The advantage of this rolling horizon approach is that it results in an on-line adaptive control scheme that also allows us to take (slow) changes in the system or in the system

parameters into account by regularly updating the model of the system.

MPC for linear systems subject to a quadratic objective function and linear constraints can be solved using quadratic programming. Other types of MPC problems in general require global or multi-start local optimization methods such as sequential quadratic programming, pattern search, or simulated annealing (Pardalos and Resende, 2002).

Just as optimal control MPC can take into account constraints on the inputs and outputs, and it can also deal with multi-input multi-output systems. MPC has an advantage over optimal control due to receding horizon approach. This feedback mechanism of MPC makes the controlled system more robust to uncertainties and disturbances. Nevertheless, MPC still has some of the drawbacks of optimal control such as computational complexity, the need of an explicit model for prediction purposes, and the fact that the external inputs and disturbances need to be known fairly accurately in advance for the entire prediction horizon.

2.3 Artificial intelligence techniques

Artificial Intelligence (AI) techniques aim at imitating aspects of human intelligence and thinking while solving a problem by introducing human intelligence (to perceive a situation, to reason about it, and to act accordingly) into computer programs (Chen et al., 2008). AI techniques are mainly used in decision support systems, and the most important ones, in particular in the context of traffic control, are (Ritchie, 1990; Nguyen and Walker, 1999; Sutton and Barto, 1998; Weiss, 1999):

- case-based reasoning,
- fuzzy logic,
- rule-based systems,
- artificial neural networks,
- multi-agent systems.

Case-based reasoning, as the name suggests, solves a problem using the knowledge that was gained from previously experienced similar situations (cases) (Aamodt and Plaza, 1994; Ritchie, 1990). In this way, this technique learns the way a new problem is solved, tests the proposed solution using simulation methods, and stores the new solution in a database. A disadvantage of this approach is that it might not be clear what should be done for a case that is not yet present in the case base. However, new cases could be added on-line to deal with this problem.

Fuzzy logic systems, like humans, can handle situations where the available information about the system is vague or imprecise (Klir and Yuan, 1995; Nguyen and Walker, 1999). To deal with such situations, fuzzy sets are used to qualify the variables of the system in a non-quantitative way. Fuzzy sets are characterized using membership functions (e.g., Gaussian, triangle, or normal) that take a value between 0 and 1, and that indicate to what degree a given element belongs to the set (e.g., a speed could be 70 % "high" and 30 % "medium"). The membership degrees can then be used to combine various rules and to derive conclusions. This process

consists of three parts: fuzzification, inference, and defuzzification. Fuzzification involves the transformation of a value of a variable into a fuzzy value, by linking it a given fuzzy set and determining a value for degree of membership. Inference uses a set of rules based on expert opinions and system knowledge and combines them using fuzzy set operators such as complement, intersection, and union of sets. Defuzzification converts the fuzzy output of the inference step in to a crisp value using techniques such as maximum, mean-of-maxima, and centroid defuzzification. One of main difficulties of a fuzzy system can be the selection of appropriate membership functions for the input and output variables. Moreover, fuzzy systems are often combined with other AI techniques for their complete deployment.

Rule-based systems solve a problem using "if-then" rules (Hayes-Roth, 1985; Russell and Norvig, 2003). These rules are constructed using expert knowledge and stored in an inference engine. The inference engine has an internal memory that stores rules and information about the problem, a pattern matcher, and a rule applier. The pattern matcher searches through the memory to decide which rules are suitable for the problem, and next the rule applier chooses the rule to apply. These systems are suited to solve problems where experts can make confident decisions. However, this system works only with already created rules and in its basic implementation it does not involve learning.

Artificial neural networks try to mimic the way in which the human brain processes information (Yao, 1999; Hammerstrom, 1993). These systems are useful in solving nonlinear problems where the rules or the algorithm to find solutions are difficult to derive. The basic processing unit of a neural network is called neuron or node. Each node fires a new signal when it receives a sufficiently high input signal from the other connected nodes. These nodes are organized in layers (an input layer, an output layer, and a number of hidden layers) and are interconnected by links or synapses, each associated with weights. A disadvantage is that artificial neural networks are non-informative models, and do not provide an explanation for the outcomes or for any failure that may occur in the process.

An agent is an entity that can perceive its environment through sensors and act upon its environment through actuators in a such way that the performance criteria are met (Ferber, 1999; Weiss, 1999). Multi-agent systems consist of a network of agents that are interacting among themselves to achieve specified goals. A high-level agent communication language is used by the agents for communication and negotiation purposes. Multi-agent systems can be applied to model complex systems, but their dynamic nature and the interactions between agents may give rise to conflicting goals or resource allocation problems.

3 Hierarchical control framework

Now we discuss several control architectures that have been developed for linking the roadside infrastructure and automated platoons. In particular, we consider the PATH, Dolphin, Auto21 CDS, CVIS, SafeSpot, and PReVENT frameworks. We will in particular expand on the PATH framework as it will allow us to capture the control of platoons and collections of platoons later on in the chapter.

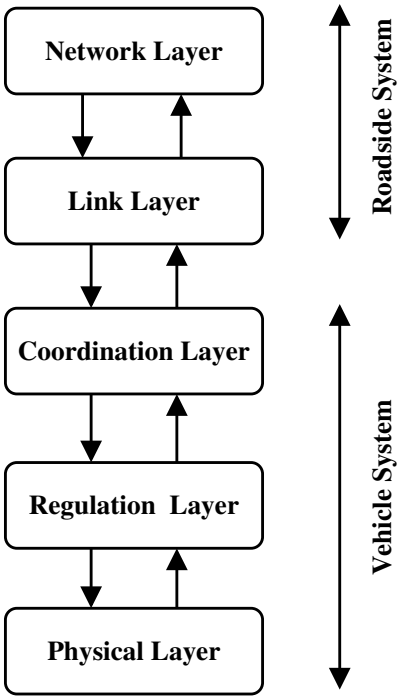

Figure 2: PATH architecture

3.1 PATH framework

The PATH architecture (Broucke and Varaiya, 1997; Varaiya and Shladover, 1991; Horowitz and Varaiya, 2000; Varaiya, 1993; Shladover, 2007) mainly focuses on the coordination of roadside-vehicle and inter-vehicle activities.

The PATH framework considers a traffic network with many interconnected highways on which the vehicles are organized in platoons. The highways in the traffic network are considered to be divided into links (about 5 km long). A link is subdivided into segments (about 1 km long) with at least one exit or one entrance. A vehicle in the PATH framework is either considered as a leader, a follower, or a free agent (i.e., a one-vehicle platoon).

The PATH framework is a hierarchical structure in which the control of the automated highway system is distributed into five functional layers as shown in Figure 2:

- physical layer,
- regulation layer,
- coordination layer,
- link layer,
- network layer.

The lower levels in this hierarchy deal with faster time scales (typically in the milliseconds range for the physical layer up to the seconds range for the coordination layer), whereas for the higher-level layers the frequency of updating can range from few times per minute (for the link layer) to once every few minutes (for the network layer). The controllers in the physical, regulation, and coordination layer reside inside the vehicles. The physical and regulation controllers govern single vehicles, whereas the coordination layer involves several vehicles. The link layer and the network layer controllers are located at the roadside, with the link layer controllers managing single freeway segments, and the network layer controllers handling entire networks.

Now we discuss each layer of the PATH framework in more detail, starting from the bottom of the hierarchy:

- The physical layer involves the actual dynamics of the vehicle. This layer has controllers that perform the actuation of the steering wheel, throttle, and brakes. It also contains the sensors in the vehicle that collect information about the speed, the acceleration, and the engine state of the vehicle, and send it to the regulation layer.

- The regulation layer controller executes the tasks specified by the higher-level coordination layer (such lane changes, and splits or merges of platoons) by converting them into throttle, steering, and braking inputs for the actuators of the vehicle. The regulation layer controller within each vehicle uses feedback control laws to execute the lateral and longitudinal maneuvers and also notifies the coordination controller in case of any failures or unsafe outcomes of the maneuvers.

- The coordination layer receives its commands from the link layer (such as set-points or profiles for the speeds, or platoon sizes). A coordination layer controller allows coordination with other neighboring platoons using messages or communication protocols, and checks which maneuvers (like lane changes, splits, or merges) have to be performed by a vehicle in order to achieve the platoon size or path trajectory specified by the link controller.

- The link layer is mainly responsible for path and congestion control. Each link controller receives commands from the network layer (such as routes for the platoons) and based on these commands, the link controller calculates the maximum platoon size, and the optimum platoon velocity for each segment in the link it is managing. The link controller also sets the local path (which lane to follow) for each platoon.

- The network layer represents the top layer in the PATH hierarchy. At this layer, the controller computes control actions that optimize the entire network. Its task is to assign a route for each vehicle or platoon that enters the highway ensuring that the capacity of each potential route is utilized properly.

3.2 A modified version of the PATH framework

In (Baskar et al., 2007; Baskar, 2009) a modified version of the PATH framework was proposed. The main motivation for this modification is the following. Although the PATH framework

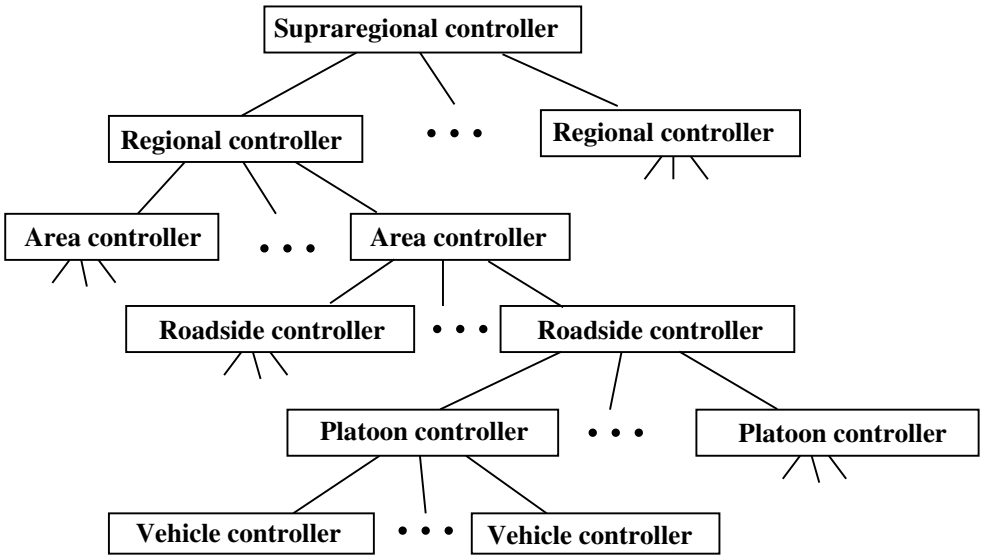

Figure 3: The hierarchical control framework of (Baskar et al., 2007; Baskar, 2009) for IVHS

includes both roadside infrastructure and vehicles, much of the research work was carried on the vehicle control side. In the PATH framework the roadside controllers determine the activities that need to be carried out in different segments. However, when the platoon size is allowed to be long enough, then it might be difficult for the roadside controller to assign the activities as the platoon resides in between two segments, and also for the vehicle controller to complete the activity within the specified space. For this reason, the modified framework uses platoon-based roadside controllers (so without segments). In addition, the framework also features some additional higher-level control layers.

The hierarchical control framework for IVHS proposed in (Baskar et al., 2007; Baskar, 2009) is also based on the platoon concept and distributes the intelligence between the roadside infrastructure and the vehicles using control measures such as intelligent speed adaption, cooperative adaptive cruise control, lane allocation, on-ramp access control, route guidance, etc. The control architecture of (Baskar et al., 2007; Baskar, 2009) consists of a multi-level structure with local controllers on-board the vehicles at the lowest level and one or more higher supervisory control levels, as shown in Figure 3.

The layers of the hierarchical control framework can be characterized as follows:

- The *higher-level controllers* (such as area, regional, and supraregional controllers) provide network-wide coordination of the lower-level and middle-level controllers. In particular, the area controllers provide area-wide dynamic route guidance for the platoons, and they supervise and coordinate the activities of the roadside controllers in their area by providing set-points, reference trajectories, or control targets. In turn, a group of area controllers can be supervised by regional controllers, and so on.

- The *roadside controllers* control a part of a highway or an entire highway. The main tasks of the roadside controllers are to assign speeds for each platoon, safe distances (to avoid

collisions between platoons), and release times at the on-ramps. The roadside controllers also give instructions for merging, splitting, and lane changes to the platoons.

- The *platoon controllers* are responsible for control and coordination of each vehicle inside the platoon. These controllers are mainly concerned with actually executing the interplatoon maneuvers (such as merging with other platoons, splitting, and lane changing) and intraplatoon activities (such as maintaining safe intervehicle distances).

- The *vehicle controllers* present in each vehicle translate commands received from the platoon controllers (e.g., reference trajectories for speeds (for intelligent speed adaption), headways (for cooperative adaptive cruise control), and paths) into control signals for the vehicle actuators such as throttle, braking, and steering actions.

Similar to the PATH framework the lower levels in this hierarchy deal with faster time scales (typically in the milliseconds range for vehicle controllers up to the seconds range for roadside controllers), whereas for the higher-level layers the frequency of updating can range from few times per minute (for area controllers) to a few times per hour (for regional and supraregional controllers).

3.3 Dolphin framework

The Japanese Dolphin framework developed in (Tsugawa et al., 2000, 2001) is similar to the PATH architecture. The Dolphin framework considers vehicles to be arranged as platoons and develops inter-vehicle communication technologies to carry out cooperative driving for the purpose of smooth merging and lane changing.

The Dolphin framework is composed of three layers

- vehicle control layer,

- vehicle management layer,

- traffic control layer.

The vehicle controller within each vehicle senses the states and the conditions ahead of the vehicle such as vehicle speed and acceleration and sends this information to the vehicle management layer. The vehicle controller also receives commands for the vehicle's steering actions and determines the actions for the vehicle actuators.

The vehicle management controller, which resides in each vehicle, receives suggestions for the movements of the vehicle from the traffic controller via road-vehicle communication and also considers the messages from the neighboring vehicles via inter-vehicle communication and the data received from the basic vehicle control layer. This controller determines the lateral and longitudinal movements of the individual vehicle under platoon-based driving.

The traffic control layer is common to all the vehicles and it is part of the roadside infrastructure. This layer consists of several distributed controllers, each of which determines advisory instructions for the vehicles in its own neighborhood and sends these instructions to the vehicle management layer.

3.4 Auto21 CDS framework

The Auto21 Collaborative Driving System (CDS) framework (Hallé and Chaib-draa, 2005) is mainly inspired by the concepts of the PATH and Dolphin architectures. The CDS architecture considers platoons of cars as autonomous agents and uses cooperative adaptive cruise control technologies to support platoon-based driving. The CDS framework employs an inter-vehicle coordination system that can ensure coordination of vehicle activities during their merge and split operations from a platoon and that can maintain stability among the vehicles in a platoon. The hierarchical architecture of the Auto21 CDS framework consists of the following three layers:

- guidance layer,
- management layer,
- traffic control layer,

with similar functionalities as the layers of the PATH and Dolphin architecture.

3.5 CVIS

CVIS (Cooperative Vehicle-Infrastructure Systems) (Toulminet et al., 2008; CVIS web site) is a European research and development project that aims to design, develop, and test technologies that allow communication between the cars and with the roadside infrastructure, which in turn will improve road safety and efficiency, and reduce environmental impact.

CVIS operates with existing traffic control and management centers, roadside infrastructures, and vehicle systems. The complete system can be considered as a single-level architecture with the existing systems and CVIS operating at the same level. Various networks and communication protocols have been developed within CVIS to enable communication between different subsystems. The time scale for this architecture ranges from minutes to hours.

A CVIS system is composed of four subsystems: central, handheld, vehicle, and roadside subsystems. The central subsystem is a basically a service provider for the vehicle or the roadside infrastructure. Typical examples of central subsystems include traffic control and service centers, and authority databases. The handheld subsystem provides services such as pedestrian safety and remote management of other CVIS subsystems by allowing access to the CVIS system using PDAs and mobile phones. The vehicle subsystem is comprised of on-board systems and includes vehicle sensors and actuators, and equipment for vehicle-vehicle and vehicle-infrastructure communication. The roadside subsystem corresponds to the intelligent infrastructure that operates at the roadside and includes traffic signals, cameras, variable message signs, etc.

3.6 SafeSpot

SafeSpot (Toulminet et al., 2008; SafeSpot web site) is a research project funded by the European 6th Framework Program on Information Society Technologies. The main objective of this project is to improve road safety using advanced driving assistance systems and intelligent

roads. The safety margin assistant developed by the SafeSpot project uses advanced communication technologies to obtain information about the surrounding vehicles and about the roadside infrastructure. This safety margin assistant can detect dangerous situations in advance and can make the driver aware of the surrounding environment using a human machine interface. The time scale for this architecture ranges from seconds to minutes.

3.7 PReVENT

PReVENT (PReVENT web site) is a European automotive industry activity co-funded by the European Commission. The main focus of the PReVENT project is to develop preventive applications and technologies that can improve the road safety. These safety applications use in-vehicle systems to maintain safe speeds and distances depending on the nature and severity of the obstacles, and to provide instructions and to assist the drivers in their driving tasks so as to avoid collisions and accidents.

The PReVENT architecture features a three-layer approach with the following layers: perception layer, decision layer, and action layer. All these layers are located within the vehicle. From the perception layer upward to the action layer, the time complexity and update frequency of states typically ranges from milliseconds to seconds.

The perception layer uses on-board sensors (such as radar, cameras, and GPS receivers) in conjunction with digital maps and allows vehicle-to-vehicle and vehicle-to-infrastructure communication. The decision layer assesses dangerous situations ahead of the vehicle and determines relevant actions that are needed to avoid such situations. The controller then passes this decision to the action layer. The action layer then issues warnings to the driver about the severity of the situation through an appropriate human machine interface or through vehicle actuators such as the steering wheel or the brakes.

3.8 Comparison of IVHS frameworks

The main differences between the frameworks consist in the control objectives considered, the type of formation control (platoons or single cars), the location of the intelligence (roadside and/or in-vehicle), and communication and coordination mechanism.

The PATH, Dolphin, AUTO21 CDS, and CVIS frameworks have developed control methodologies to be implemented in the roadside infrastructure to improve the traffic flow or in vehicles to allow automation of driving tasks. On the other hand, SafeSpot and PReVENT focus on improving the road safety by avoiding or preventing accidents, and they aim at integrated safety, with an emphasis on the potential of communication between vehicles and between vehicles and roadside systems.

The frameworks usually consider the vehicles to be controlled either as part of higher-level entities such as platoons, or as individual vehicles. The PATH, Dolphin, and Auto21 CDS frameworks allow platooning. On the other hand, SafeSpot, PReVENT, and CVIS do not use platoons.

The PATH framework allows involvement of both roadside infrastructure and vehicles for improving traffic performance. Although the Dolphin and the Auto21 CDS frameworks consider distributed intelligence between roadside infrastructure and vehicles, the roadside infrastructure

only provides suggestions and instructions to the vehicles. The platoons are not obliged to follow these suggestions. CVIS and SafeSpot incorporate intelligence in both vehicle and roadside infrastructure. PReVENT also includes distributed intelligence but with the main focus on vehicle intelligence.

Almost all the frameworks and projects have designed and developed technologies for intervehicle and roadside-vehicle communication for coordination of activities. More specifically, PATH has developed dedicated communication protocols and the Dolphin framework has developed a wireless local access network model for vehicle following, and intervehicle communication technologies for coordination of platoon maneuvers. For the coordination of tasks within the platoons, the AUTO21 CDS framework allows both a centralized set-up (i.e., the platoon leader instructs intra-platoon maneuvers) or a decentralized set-up (i.e., all the members of the platoon are involved in the coordination) SafeSpot, PReVENT, and CVIS focus on the issue of developing communication techniques that can be implemented in existing traffic networks and that can also be extended to AHS.

A more detailed comparison of the above frameworks is presented in (Baskar et al., 2011; Baskar, 2009).

4 Control of vehicle platoons

The traffic control frameworks presented in the previous section either explicitly identify a vehicle platoon as a layer in the hierarchical structure (PATH, Dolphin, Auto21 CDS) or have a more generic design, such that vehicle platoons can be an important component in the overall control framework (CVIS, PReVENT). Consequently, vehicle platoons are expected to play an important role in traffic control. Therefore, this section focuses on the developments in the field of control of vehicle platoons. To this end, we will first formally introduce the platoon control problem after which a very important aspect, being the notion of "string stability" is reviewed.

4.1 Problem formulation

The concept of platooning refers to a string of vehicles that aim to keep a specified, but not necessarily constant, inter-vehicle distance. Consequently, the control of vehicles in a platoon can be characterized as a vehicle-following control problem. This section will state the formal control objective and identifies the components of the control system, as reported in literature.

4.1.1 Vehicle following

Let us consider an arbitrary (and possibly infinite) number of vehicles that drive behind each other as depicted in Figure 4. Note that this configuration is referred to as a "string" of vehicles and not as a "platoon" because the latter might suggest the presence of a platoon leader, which is certainly not a prerequisite.

In Figure 4 the position (measured with respect to the rear bumper) of vehicle i at time t is denoted by $s_i(t)$. The distance ("headway") $d_i(t)$ of vehicle i, with length L_i, at time t to the

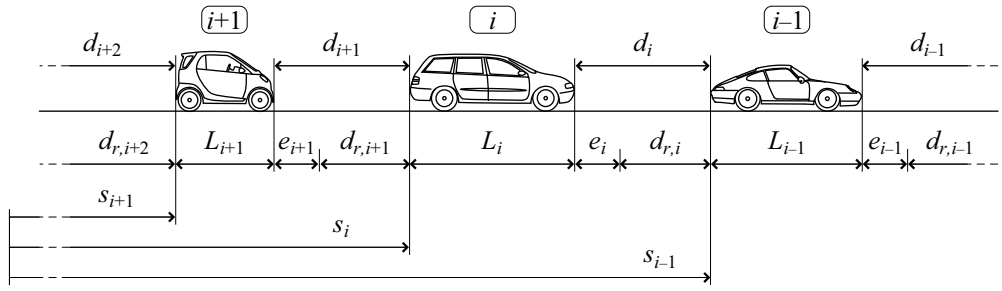

Figure 4: A string of vehicles on a straight lane

preceding vehicle $i-1$ therefore equals

$$d_i(t) = s_{i-1}(t) - (s_i(t) + L_i). \tag{1}$$

The distance error $e_i(t)$ is taken as

$$e_i(t) = d_i(t) - d_{\mathrm{r},i}(t) \tag{2}$$

where $d_{\mathrm{r},i}(t)$ is the desired headway of vehicle i that follows from the so-called *spacing policy* (see below). For the ease of notation we will not explicitly indicate the time argument t any more in the sequel if it is clear from the context.

The control objective $\mathcal{O}$, referred to as (asymptotic) *vehicle following*, can now be formulated in its most basic form as regulating e_i to zero in the presence of disturbances induced by

- initial condition errors of any vehicle in the string,
- perturbations in the velocity of other vehicles in the string,
- velocity variations of the lead vehicle,
- setpoint changes with respect to the desired distance,

or, in other words,

$$\mathcal{O} : \lim_{t\to\infty} e_i(t) = 0, \quad \forall\, t \geq t_0, \forall i \in \{2,\ldots,m\} \tag{3}$$

where control starts at time $t = t_0$. Here, the string is assumed to consist of m vehicles in total, of which $m-1$ vehicles have a vehicle-following objective. The lead vehicle, with index 1, not being subject to a vehicle following objective, ultimately defines the desired speed of the entire string. To this end, the lead vehicle is assumed to be velocity controlled with a given, possibly time-varying set speed. For persistently time-varying perturbations in the velocity of other vehicles in the string and/or of the lead vehicle, which must certainly be included for the vehicle following problem, it might not be feasible to achieve asymptotic disturbance rejection or tracking, respectively. In such cases, the control objective can be formulated as

$$\mathcal{O} : \|e_i(t)\|_p \leq \varepsilon, \quad \forall\, t \geq T, \forall i \in \{2,\ldots,m\} \tag{4}$$

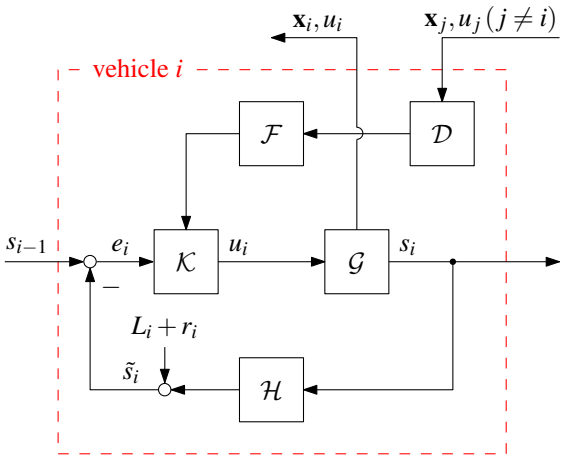

Figure 5: General control structure of the vehicle following problem

where ε is an a priori chosen (small) number, and $T > t_0$ is introduced to ignore transient effects due to initial condition errors. In (4) $\|\cdot(t)\|_p$ denotes the vector p-norm, defined as

$$\|\mathbf{u}\|_p \triangleq \left(\sum_i |u_i|^p\right)^{1/p} \tag{5}$$

for a vector $\mathbf{u}$. Whichever objective is used, the vehicle following problem can essentially be regarded as a standard asymptotic tracking problem that occurs in many applications. One additional requirement that can be formulated however, is related to how the responses of the vehicles to disturbances evolve not only in time, but also across the vehicles in the string. This disturbance evolution across a string of vehicles, or, in general, across a number of interconnected subsystems, is covered by the notion of *string stability*. Note that the problem formulation can be further extended with additional performance requirements based on comfort, fuel consumption and other important criteria. These are however outside the scope of this section.

4.1.2 Control system components

In order to further explain the various components in the control system that is designed to fulfill the objective (3) or (4), Figure 5 shows a block scheme of a controlled vehicle in a string, taken from (Naus et al., 2009, 2010c). Although this block scheme will not hold for all possible control solutions, it serves the purpose of identifying the various components, to be explained in further detail below.

First of all, the subsystem $\mathcal{G}$ represents the vehicle to be controlled. In frequency-domain oriented approaches, such as applied in (Naus et al., 2009, 2010c), a general linear vehicle model is formulated, according to

$$\mathcal{G} : G_i(s) = \frac{k_i}{s^2(\tau_i s + 1)} e^{-\theta_i s} \tag{6}$$

where τ_i is a time constant representing the lumped vehicle actuator dynamics, θ_i is a time delay, caused by the throttle and brake system, and k_i a gain. The input u_i can be interpreted as the desired acceleration, whereas the output is the resulting vehicle position s_i (i.e., $k_i \approx 1$). Consequently, the model (6) includes in fact a low-level acceleration controller. This acceleration controller can be characterized as a linearizing pre-compensator whose input is the desired acceleration and whose output is the throttle valve position or brake pressure. This pre-compensator aims to linearize the vehicle drive line at the lowest level possible, thereby greatly simplifying the design of higher-level vehicle following controllers. Moreover, the pre-compensator "hides" specific vehicle characteristics such as the mass, aerodynamic drag, and rolling resistance, even more simplifying the higher-level control design. The following parameter values for a Citroën Grand C4 Picasso combined with a custom design pre-compensator are mentioned in (Naus et al., 2010c): $k_i = 0.72$, $\tau_i = 0.38$ s and $\theta_i = 0.18$ s.

Another, frequently used vehicle model is obtained by means of input-output linearization by state feedback (Stanković et al., 2000), resulting in

$$\mathcal{G}: \begin{cases} \dot{s}_i = v_i \\ \dot{v}_i = a_i \\ \dot{a}_i = -\tau_i^{-1}(v_i)a_i + \tau_i^{-1}(v_i)u_i \end{cases} \tag{7}$$

where v_i is the vehicle speed, a_i the acceleration, and u_i the external input (desired acceleration); $\tau_i(v_i)$ is a velocity-dependent time constant representing the engine dynamics. For ease of notation, the time argument t is omitted. Note that (7) is in fact only partly linearized since the state v_i still occurs in a nonlinear fashion. Obviously, it is easily possible to complete the model linearization by introducing a new input η_i and choosing $u_i = \tau_i(v_i)\eta_i + a_i$, resulting in

$$\mathcal{G}: \begin{cases} \dot{s}_i = v_i \\ \dot{v}_i = a_i \\ \dot{a}_i = \eta_i \end{cases} \tag{8}$$

which is applied in, e.g., (Sheikholeslam and Desoer, 1992; Ioannou and Chien, 1993; Sheikholeslam and Desoer, 1993). Alternatively, τ_i could be approximated by a constant value. The model (7) then becomes linear and equal to the frequency-domain model (6) with $k_i = 1$ and $\theta_i = 0$.

The subsystem $\mathcal{H}$ describes the *spacing policy*, which refers to the choice of desired headway $d_{\mathrm{r},i}$. Originally, $d_{\mathrm{r},i}$ has been determined using physical considerations (Ioannou and Chien, 1993), taking into account the distance it takes for a vehicle to adapt its speed to a preceding decelerating vehicle, such that a collision is just avoided. This approach results in a desired headway according to

$$d_{\mathrm{r},i} = c_{0,i} + c_{1,i}v_i + c_{2,i}(v_{i-1}^2 - v_i^2) \tag{9}$$

where the coefficients $c_{k,i}$ $(k = 0,1,2)$ depend on vehicle specifications, such as maximum deceleration and jerk, and a possible reaction time delay. Obviously, for tight vehicle following, i.e., v_i close to v_{i-1}, (9) can be simplified to $d_{\mathrm{r},i} = c_{0,i} + c_{1,i}v_i$, which is commonly denoted as

$$d_{\mathrm{r},i} = r_i + h_i v_i. \tag{10}$$

The variable h_i is known as the (desired) *time headway* and r_i is the *standstill distance*, since it can be interpreted as such. The latter is desired to prevent a near collision at standstill. Note that according to (International Organization for Standardization, 2002), h_i must be greater than or equal to 1.0 s for commercially available road vehicles equipped with Adaptive Cruise Control (ACC)[2]. There is no legal upper limit to h_i, but in practice this turns out to be chosen as high as 3.6 s by system manufacturers, probably stemming from comfort requirements. This is significantly higher than common human driver behavior would incur. For Cooperative ACC (see below), string stability can be achieved for values of h_i smaller than 1.0 s (Naus et al., 2009, 2010c); safety is however not taken into account here. The spacing policy (10) can be formulated in terms of a transfer function as follows. First write the distance error e_i as defined in (2) as

$$\begin{aligned} e_i &= d_i - d_{\mathrm{r},i} \\ &= s_{i-1} - (s_i + L_i) - (r_i + h_i v_i) \\ &= s_{i-1} - \tilde{s}_i \end{aligned} \tag{11}$$

with

$$\tilde{s}_i = L_i + r_i + s_i + h_i v_i. \tag{12}$$

Note that $\tilde{s}_i$ can be interpreted as the "virtual control point" of vehicle i, whose position must be as close as possible to the actual position s_{i-1} of the preceding vehicle $i-1$. Reformulating (12) as a system with input s_i and output $\tilde{s}_i$, while omitting the constants L_i and r_i, yields the spacing policy transfer function

$$\mathcal{H} : H_i(s) = 1 + h_i s \tag{13}$$

which clearly indicates the differential action contributing to a well damped behavior of the controlled vehicle. This explains why this spacing policy is widely used in literature and has become a de facto standard for commercial ACC. Note that the chosen spacing policy also influences the traffic flow stability, having consequences for throughput. These aspects are not taken into account, even though it is known that the above constant time headway spacing policy might have adverse effects on traffic flow stability. Some research has been done into this area, see, e.g., (Swaroop and Rajagopal, 1999), who analyze the effects of the spacing policy on traffic flow stability and propose alternative spacing policies.

The feedback controller $\mathcal{K}$ in the block diagram of Figure 5 can be regarded as an ACC controller, based on direct measurement of the distance to the preceding vehicle. In general, this distance is measured by an on-board environmental sensor such as radar or scanning laser (lidar)[3]. PD-like controllers are often used here, see, e.g., (Ioannou and Chien, 1993; Venhovens et al., 2000), where the differential action does not have to be explicitly implemented since both radar and lidar directly measure the relative velocity. In (Naus et al., 2010a) an ACC controller is designed based on Explicit MPC, which is a rather promising approach in view of constraints such as maximum velocity, acceleration and jerk.

[2]The letter 'A' in ACC turns out to be rather versatile, as it might represent 'Adaptive', 'Automatic', 'Autonomous', 'Advanced', or 'Active', depending on the car brand.

[3]A radar or laser directly measures the distance d_i. The block diagram is however rather efficiently formulated due to the introduction of the spacing policy $\mathcal{H}$, having the drawback that d_i is not explicitly available in the diagram anymore.

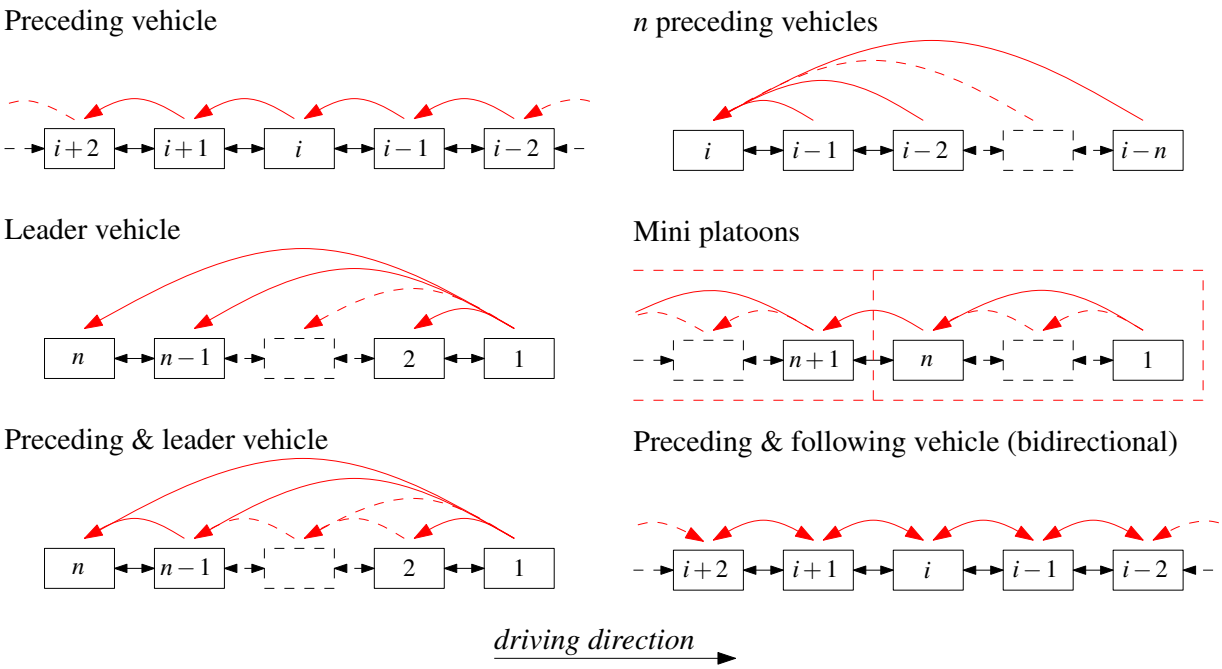

Figure 6: Communication structures for CACC

The feedforward controller, denoted by $\mathcal{F}$, provides the extension from ACC to so-called Cooperative ACC (CACC). To this end, wireless communication is applied to obtain motion data, represented by the vehicle state $\mathbf{x}_j$ in Figure 5, from other vehicles than the directly preceding one, and/or to obtain data $\mathbf{x}_{i-1}$ from the directly preceding vehicle that cannot be measured by the on-board environment sensor. In literature, a very wide range of communication structures is found, e.g., communication with the directly preceding vehicle, communication with a platoon leader or even bidirectional communication. Figure 6 provides an overview of structures reported in literature.

The main objective here is to obtain or enhance string stability compared to ACC. The (possibly varying) communication delay involved in this method of data exchange is represented by $\mathcal{D}$. The combined controllers $\mathcal{K}$ and $\mathcal{F}$ constitute a CACC controller. In the literature, a PD-like controller with feedforward is very often used, see, e.g., (Sheikholeslam and Desoer, 1992, 1993; Tsugawa et al., 2001; Naus et al., 2009, 2010c). Also sliding mode controllers are regularly encountered in literature, see, e.g., (Swaroop et al., 1994; Gehring and Fritz, 1997). Note that (Levine and Athans, 1966) is probably the first paper on the subject. In that paper, optimal control is applied, which is not surprising given the developments at the time. Although $\mathcal{K}$ and $\mathcal{F}$ obviously need to be synthesized in an integral approach, and usually are, it is still possible to distinguish both functions in the controller and even to guarantee string stability in case only the "ACC-part" $\mathcal{K}$ is active, i.e., when the communication link is not functioning properly, albeit with a significantly larger time headway. The latter feature might prove of great importance to obtain a certain robustness against communication impairments such as latency (caused by among others queuing delay, transmission delay, and propagation delay) and packet loss (due to

packet collisions related to the so-called “hidden-node problem”, and interference).

4.2 String stability literature review

This section provides as short literature overview on the notion of string stability. Three main approaches can be distinguished here, being a Lyapunov-like approach, focusing on generalization of the notion of stability, a performance-oriented approach, and finally a linear stability approach for strings of infinite length, based on the bilateral $\mathscr{Z}$-transform.

4.2.1 Platooning requirements

From a practical perspective, control design for a string of vehicles in order to obtain vehicle following behavior, entails in the first place achieving *individual vehicle stability* (Rajamani, 2006). This is commonly interpreted as achieving stable vehicle following behavior with the preceding vehicle driving at constant velocity. Taking the block diagram of Figure 5, with vehicle model $G_i(s)$, controller $K_i(s)$, and spacing policy $H_i(s)$, the system with complementary transfer function $T_i(s) = (1 + G_i(s)K_i(s)H_i(s))^{-1} G_i(s)K_i(s)$ should be stable. Obviously, this is a basic practical requirement, that is assumed to be met in the remainder of this section.

Besides individual vehicle stability, an important requirement is the ability of the string of vehicles to attenuate disturbances, or at least guarantee boundedness, introduced by an arbitrary vehicle in the string as we “move away” from that vehicle. Assume for instance a vehicle string where each vehicle is controlled based on information of one or more preceding vehicles, i.e., a unidirectional communication link between vehicles in upstream direction. If the first vehicle in the string introduces some disturbance, e.g. a variation in velocity, then the states of the following vehicles should be bounded as a result or, preferably, get smaller in some sense in upstream direction, ultimately leading to a smooth traffic flow. The notion of string stability refers exactly to this property. Note that (Swaroop et al., 1994) also mentions that spacing errors should not be amplified in the platoon in order to avoid collisions, thereby providing another motivation for string stability[4].

4.2.2 The Lyapunov-like approach

Although in the majority of the literature, string stability is not explicitly defined, a formal approach of the subject can be found in the work of Swaroop (Swaroop and Hedrick, 1996; Swaroop et al., 2001). As opposed to system stability, which is essentially concerned with the evolution of system states *over time*, string stability focuses on the propagation of states *over subsystems*. Recently, new results appeared (Klinge and Middleton, 2009), related to the stability analysis in case of a one-vehicle look-ahead control architecture and a homogeneous string. The resulting string stability definition is given below.

[4]It might be questioned whether collision avoidance is a valid argument for requiring string stability, since this is likely to require dedicated controllers that do not aim to optimize string behavior with respect to smoothness. Moreover, collision avoidance cannot be guaranteed with linear controllers, as regularly used in controller design for vehicle platoons.

Definition 4.1 ($\mathcal{L}_p$ String Stability)**.** *Consider a string of m dynamic systems of order n described by*

$$\begin{aligned}\dot{\mathbf{x}}_1 &= \mathbf{f}(\mathbf{x}_1, \mathbf{0}) \\ \dot{\mathbf{x}}_i &= \mathbf{f}(\mathbf{x}_i, \mathbf{x}_{i-1}) \quad \forall\, i \in \{2, \ldots, m\}\end{aligned}$$

with $\mathbf{x}_i \in \mathbb{R}^n$, $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, *and* $\mathbf{f}(\mathbf{0}, \mathbf{0}) = 0$, *and* $\mathbf{x}_i(0) = \mathbf{0}$ *for* $i = 2, \ldots, m$. *Then, the origin is* $\mathcal{L}_p$*-string stable if for each* $\varepsilon > 0$, *there exist a* $\delta > 0$ *such that*

$$\|\mathbf{x}_1(0)\|_p < \delta \Rightarrow \sup_i \|\mathbf{x}_i(t)\|_{\mathcal{L}_p} < \varepsilon \tag{14}$$

Here, $\|\cdot\|_p$ denotes the vector p-norm and $\|\cdot\|_{\mathcal{L}_p}$ denotes the $\mathcal{L}_p$-norm for vector-valued signals. Obviously, according to this definition, homogeneous, linear strings of finite length are string stable, provided that the vehicles are individually stable. However, as m approaches infinity, it appears that the string stability requirement leads to a lower bound on the time headway h. The above definition nicely illustrates that string stability is concerned with the propagation of states over the string.

4.2.3 The performance-oriented approach

Despite the existence of the Lyapunov-like approach, which may be thought of as being rigorous, a frequency-domain approach for string stability is also adopted since this appears to directly offer tools for controller synthesis (Sheikholeslam and Desoer, 1992; Ioannou and Chien, 1993; Sheikholeslam and Desoer, 1993; Swaroop et al., 1994; Gehring and Fritz, 1997; Stanković et al., 2000; Naus et al., 2009, 2010c). In the performance-oriented approach, string stability is evaluated by analyzing the amplification in upstream direction of either distance error, velocity and/or acceleration. This immediately leads to the following definition, (implicitly) used in the above literature references.

Definition 4.2 (Bounded Propagation String Stability)**.** *Consider a string of* $m \in \mathbb{N}$ *dynamic systems, then the string is considered string stable if and only if*

$$\|z_i(t)\|_{\mathcal{L}_\infty} \leq \|z_{i-1}(t)\|_{\mathcal{L}_\infty}, \quad \forall\, i \in \{2, \ldots, m\},\ t \geq 0$$

where $z_i(t)$ *can either be the distance error* $e_i(t)$, *the velocity* $v_i(t)$ *or the acceleration* $a_i(t)$ *of vehicle i,* $z_1(t) \in \mathcal{L}_\infty$ *is any given input signal, and* $z_i(0) = 0$ *for* $i = 2, \ldots, m$.

Here, $\|\cdot\|_{\mathcal{L}_\infty}$ denotes the signal ∞-norm, which for scalar signals comes down to taking the highest peak value over time. Definition 4.2 thus states that the peak value of either distance error, velocity or acceleration must decrease in upstream direction. In literature, the choice between distance error, velocity or acceleration seems a little arbitrary. Note that $z_i(t)$ is thus assumed to be a scalar signal.

From linear system theory (Desoer and Vidyasagar, 2009), it is well known that the $\mathcal{L}_\infty$-norms of input and output are related through the $\mathcal{L}_1$-norm of the impulse response matrix $\gamma_i(t)$

with respect to the "input" z_{i-1} and the "output" $z_i(t)$, according to

$$\|\gamma_i(t)\|_{\mathcal{L}_1} = \max_{z_{i-1}\neq 0} \frac{\|z_i(t)\|_{\mathcal{L}_\infty}}{\|z_{i-1}(t)\|_{\mathcal{L}_\infty}} \tag{15}$$

where the $\mathcal{L}_1$-norm $\|\gamma_i(t)\|_{\mathcal{L}_1}$ for scalar impulse responses equals the integral over time of the absolute value $|\gamma_i(t)|$. Hence, the above definition yields the *necessary* and *sufficient* string stability requirement (Rajamani, 2006)

$$\|\gamma_i(t)\|_{\mathcal{L}_1} \leq 1, \quad \forall\, i \in \{2,\ldots,m\},\, t \geq 0. \tag{16}$$

In practice, however, the application of impulse response functions is not particularly easy and intuitive. Using another fact from linear system theory, allows translation of the string stability requirement to the frequency domain, to some extent. Consider to this end the relation between the norm of $\gamma_i(t)$ and its corresponding transfer function $\Gamma_i(s)$ on the imaginary axis:

$$\|\Gamma_i(j\omega)\|_{\mathcal{H}_\infty} \leq \|\gamma_i(t)\|_{\mathcal{L}_\infty} \tag{17}$$

where the $\mathcal{H}_\infty$ norm for scalar transfer functions equals the peak over the frequency ω of the gain $|\Gamma_i(j\omega)|$. This immediately leads to the *necessary* condition for string stability

$$\|\Gamma_i(j\omega)\|_{\mathcal{H}_\infty} \leq 1, \quad \forall\, i \in \{2,\ldots,m\} \tag{18}$$

which is far more convenient for controller synthesis in general. In (Swaroop et al., 1994) it is shown that if $\gamma_i(t) \geq 0$, (17) becomes an equality, thus making (18) a necessary and sufficient condition. From a physical perspective, $\gamma_i(t) \geq 0$ means that the time response shows no overshoot to a step input, indicating a sufficient level of damping. Commonly, the requirement (18) is considered necessary and sufficient for string stability. This can be motivated by the fact that the $\mathcal{H}_\infty$-norm is induced by the $\mathcal{L}_2$-norms of input and output which, in turn, are measures for energy. As a consequence, the condition $\|\Gamma_i(j\omega)\|_{\mathcal{H}_\infty} \leq 1$ can be interpreted as requiring energy dissipation in upstream direction.

The string stability transfer function thus equals $\Gamma_i(j\omega) = Z_i(s)/Z_{i-1}(s)$, with $Z(s)$ being the Laplace transform of $z(t)$. In general, however, $Z_{i-1}(s)$ is not an independent input, since it is determined by other downstream vehicles or even, in case of bidirectional communication, by upstream vehicles in the string. Consequently, the string stability transfer function should in fact be regarded as the product of transfer functions which actually have independent inputs, being the first vehicle in the string:

$$\Gamma_i(s) = \frac{Z_i(s)}{Z_1(s)} \left(\frac{Z_{i-1}(s)}{Z_1(s)} \right)^{-1}. \tag{19}$$

$\Gamma_i(s)$ is capable of describing the simplest one-vehicle look-ahead communication structure, but also more complex communication structures as shown in Figure 6. In the latter case however, $\Gamma_i(s)$ not only includes the dynamics of two neighboring vehicles, but also the dynamics of vehicles further away.

4.2.4 The z-domain approach

Within the framework of analysis of string stability of infinite-length vehicle strings, or any other system consisting of identical interconnected subsystems, the model of such a system is formulated in state-space as

$$\dot{\mathbf{x}}_l(t) = \sum_{j=-\infty}^{\infty} \left(\mathbf{A}_{l-j}\mathbf{x}_j(t) + \mathbf{B}_{l-j}\mathbf{u}_j(t) \right) \tag{20a}$$

$$\mathbf{y}_l(t) = \sum_{j=-\infty}^{\infty} \mathbf{C}_{l-j}\mathbf{x}_j(t) \tag{20b}$$

where $\mathbf{x}_l(t)$ denotes the state of subsystem $l \in \mathbb{Z}$ and $\mathbf{u}_j(t)$ the input of subsystem j. The matrices $\mathbf{A}_{l-j}$, $\mathbf{B}_{l-j}$, and $\mathbf{C}_{l-j}$ are the state, input, and output matrix, respectively, regarding the influence of subsystem j on subsystem l. In general, the influence of other subsystems decreases when they are "further away", meaning that the state-space matrices approach zero for $j \to \pm\infty$. Moreover, a distributed linear output-feedback controller is assumed, according to

$$\mathbf{u}_j(t) = \mathbf{K}\mathbf{y}_j(t). \tag{21}$$

In order to analyze this system, it can be transformed using the bilateral $\mathscr{Z}$-transform. Consider to this end a sequence $\{a_k(t)\}_{k=-\infty}^{\infty}$. The bilateral $\mathscr{Z}$-transform $\mathscr{Z}(a_k(t)) = \hat{a}(z,t)$ is then defined by

$$\hat{a}(z,t) \triangleq \sum_{k=-\infty}^{\infty} a_k(t) z^{-k} \tag{22}$$

where z is a complex variable. The model (20) is already formulated as a convolution, which makes the application of the $\mathscr{Z}$-transform particularly easy, resulting in

$$\hat{\dot{\mathbf{x}}}(z,t) = \hat{\mathbf{A}}(z)\hat{\mathbf{x}}(z,t) + \hat{\mathbf{B}}\hat{\mathbf{u}}(z,t) \tag{23a}$$

$$\hat{\mathbf{y}}(z,t) = \hat{\mathbf{C}}(z)\hat{\mathbf{x}}(z,t) \tag{23b}$$

$$\hat{\mathbf{u}}(z,t) = \mathbf{K}\hat{\mathbf{y}}(z,t). \tag{23c}$$

Defining $\mathbf{D}(z) = \hat{\mathbf{A}}(z) + \hat{\mathbf{B}}(z)\hat{\mathbf{K}}(z)\hat{\mathbf{C}}(z)$, the closed-loop system thus reads

$$\hat{\dot{\mathbf{x}}}(z,t) = \mathbf{D}(z)\hat{\mathbf{x}}(z,t). \tag{24}$$

Using this approach, a third type of string stability definition can be formulated (El-Sayed and Krishnaprasad, 1981; Chu, 1974; Barbieri, 1993).

Definition 4.3 (Interconnected System String Stability)**.** *Assume a dynamical interconnected system described by an infinite number of identical n^{th}-order subsystems* (20) *with feedback control law* (21)*, such that the bilateral $\mathscr{Z}$-transform of the closed-loop system is given by* (24)*. Then this system is string stable if all the eigenvalues λ_i $(i = 1,\ldots,n)$ of $\mathbf{D}(z)$ are in the left-half complex plane, for all z on the unit circle, i.e.,*

$$\mathrm{Re}\left(\lambda_i\left(\hat{\mathbf{D}}\left(e^{j\theta}\right)\right)\right) \leq 0$$

for $i = 1,\ldots,n$ and for $0 \leq \theta < 2\pi$.

This approach, although elegant in itself, has severe limitations due to the assumed infinite length of the interconnected system and due to the fact that it only covers homogeneous strings, i.e., all subsystems must be identical.

4.2.5 Concluding remarks on string stability

It can probably be stated that Definition 4.1 (or generalized versions thereof) regarding $\mathcal{L}_p$ string stability, formulates a true string stability definition. In itself, however, this definition provides little support for controller synthesis, as opposed to Definition 4.2. The latter has the character of a *condition* for string stability rather than a *definition* of this notion. It should therefore be possible to rigorously derive this condition using for instance the notion of input-output stability (Khalil, 2000). As far as the string stability of interconnected systems, given in Definition 4.3 is concerned, as already mentioned, this approach mainly has theoretical value regarding the stability analysis and controller synthesis of infinitely long strings of identical subsystems. In practice, the performance-oriented approached is often adopted, which is nicely illustrated in the case study, shortly summarized in the next section.

4.3 Case study

To validate the theory, especially with respect to the performance-oriented string stability approach, experiments have been performed with two vehicles, as described in (Naus et al., 2010b,c). Vehicle 1 communicates its actual acceleration to vehicle 2. The latter is equipped with an electro-hydraulic braking (EHB) system, facilitating brake-by-wire control and an electronically controlled throttle valve; both serve as actuators for the custom-made lower-level acceleration controller. Finally, vehicle 2 is equipped with a laser radar.

The vehicle model (6) is adopted, which is supposed to also include the lower-level acceleration controller. After experiments, the following parameter values have been identified: $k_i = 0.72$, $\tau_i = 0.38\,\mathrm{s}$ and $\theta_i = 0.18\,\mathrm{s}$. The model is validated using step responses, depicted in Figure 7. From this figure, it can be concluded that the model (6) adequately describes the relevant dynamics.

The controller $\mathcal{K}$ (refer to Figure 5) is chosen as a lead-lag filter, according to

$$K_i(s) = \frac{\omega_{\mathrm{K},i}}{k_i}\frac{\omega_{\mathrm{K},i}+s}{\omega_{\mathrm{f},i}+s}, \quad \text{for } i>1 \tag{25}$$

with $\omega_{\mathrm{K},i} = 0.5\,\mathrm{rad/s}$ and $\omega_{\mathrm{f},i} = 100.0\pi\,\mathrm{rad/s}$. These parameters are mainly based on ensuring a sufficient closed-loop bandwidth and phase margin of the individually controller vehicle, characterized by the complementary sensitivity $T_i(s) = (1+G_i(s)K_i(s))^{-1}G_i(s)K_i(s)$ (see Section 4.2.1). The identified vehicle gain k_i is compensated for by including this gain in the controller as well. As mentioned, the preceding vehicle acceleration is communicated, which then serves as the input for the feedforward filter $\mathcal{F}$. Hence, the feedforward filter should compensate for the vehicle dynamics $G_i(s)$ and take the spacing policy into account as well. Since it cannot compensate for time delay, the following filter is implemented:

$$F_i(s) = \left(H_i(s)G_i(s)s^2\right)^{-1}, \quad \text{for } i>1 \tag{26}$$

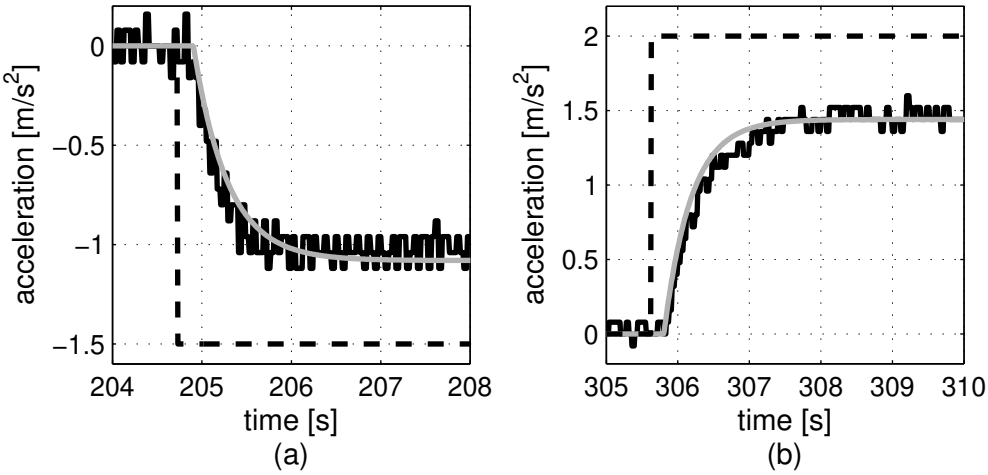

Figure 7: Validation step-response results for (a) braking and (b) accelerating. The reference step input (dashed black), the measurement results (solid black), and the corresponding simulation results (solid grey) are shown.

using the constant time headway spacing policy (13).

For the wireless inter-vehicle communication, the standard WiFi protocol IEEE 802.11g is used, with an update rate of 10 Hz. The acceleration of vehicle 1 is derived from the built-in anti-lock braking system (ABS), which is available on the CAN-bus, and communicated to vehicle 2. A zero-order-hold approach is adopted for the communicated signal, introducing a corresponding average delay of about 50 ms. To synchronize the measurements of both vehicles, GPS time stamping is adopted. Correspondingly, an additional communication delay of about 10 ms is identified. Combination of these values yields $\psi_i \approx 60\,\text{ms}$ as a total delay. hence, the communication delay model $\mathcal{D}$, shown in Figure 5, is defined by

$$D_i(s) = e^{-\psi_i s}. \tag{27}$$

Having determined all control system components, the string stability transfer function $\Gamma_i(s)$ can be analyzed. In (Naus et al., 2010b,c), it is shown that for a homogeneous vehicle string, i.e., a string with all identical subsystems, the string stability transfer functions are identical, regardless of whether the distance error, the acceleration, or the control input is chosen as input/output. The resulting string stability transfer function reads

$$\begin{aligned}\Gamma_i(s) &= \frac{\left(F_i(s)D_i(s)s^2 + K_i(s)\right)G_i(s)}{1 + H_i(s)G_i(s)K_i(s)} \\ &= \frac{D_i(s) + H_i(s)G_i(s)K_i(s)}{H_i(s)\left(1 + H_i(s)G_i(s)K_i(s)\right)}\end{aligned} \tag{28}$$

which reduces to $\Gamma_i(s) = 1/H_i(s)$ in case there is no communication delay, i.e., $D_i(s) = 1$. Figure 8 shows the resulting magnitude for various values of the time headway h_i. Also the ACC equivalent is depicted, which in fact equals the CACC with $F_i(s) = 0$. It can be clearly seen

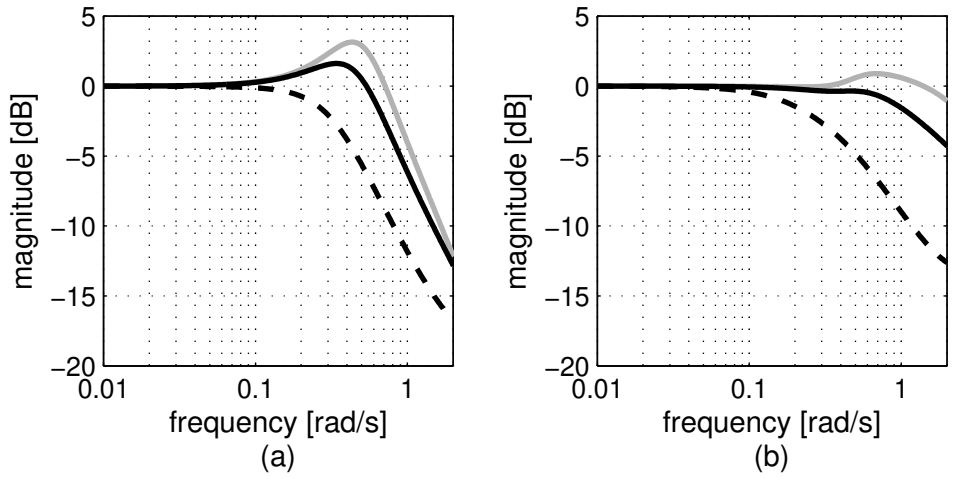

Figure 8: Bode magnitude plots of $\Gamma_i(j\omega)$, in the case of (a) ACC, and (b) CACC, for time headway $h_i = 0.5$ s (solid grey), $h_i = 1.0$ s (solid black), and $h_i = 3.0$ s (dashed black).

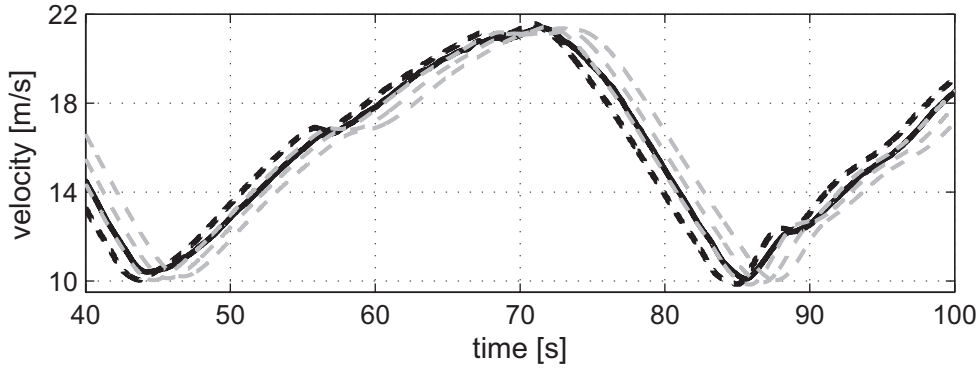

Figure 9: Measured velocity of vehicle 1 (dashed black) and vehicle 2 (solid black), as well as the resulting velocities for three simulated follower vehicles (dashed grey).

that ACC provides string stable behavior only for $h_i = 3.0\,\mathrm{s}$, which is considered quite large in practice. CACC provides string stable behavior for values as small as $h_i = 1.0\,\mathrm{s}$, given the current parameters of the controller $K_i(s)$; (Naus et al., 2010c) mentions $h_i = 2.6\,\mathrm{s}$ as minimum string-stable value for ACC and $h_i = 0.8\,\mathrm{s}$ for CACC.

Figure 9 contains some experiment results, showing the measured velocity of the first and second vehicle, extended with simulated results of three more follower vehicles for CACC with a time headway $h_i = 1.0\,\mathrm{s}$. From this figure, it can be clearly seen that the disturbance, introduced by the first vehicle, is attenuated, albeit barely.

5 Roadside and area control

In this section we summarize the approach proposed in (Baskar, 2009; Baskar et al., 2008, 2009a,b) for the roadside controllers to determine optimal speeds, lane allocations, and on-ramp release times for the platoons, and for the area controllers to determine optimal flows and speeds on links. As control approach we adopt the model predictive control (MPC) scheme presented in Section 2.2.2.

5.1 MPC for roadside controllers

In order to the make the MPC approach tractable the roadside controllers do not consider each individual vehicle in each platoon separately, but they consider each platoon as a single unit and they monitor the movements of the platoons in the highway stretch under their control. More specifically, the platoons are modeled using the so-called "big car" model, i.e., as a single (long) vehicle with a speed-dependent length:

$$L_{\text{plat},p}(k) = (n_p - 1)S_0 + \sum_{i=1}^{n_p-1} T_{\text{head},i} v_{\text{plat},p}(k) + \sum_{i=1}^{n_p} L_i \ ,$$

where $L_{\text{plat},p}(k)$ is the length of platoon p at time step k, n_p is the number of vehicles in the platoon, S_0 the minimum safe distance that is to be maintained at zero speed, $T_{\text{head},i}$ is the desired time headway for vehicle i in the platoon, $v_{\text{plat},p}(k)$ is the speed of the platoon (leader), and L_i is the length of vehicle i. In this way a good trade-off is obtained between computational speed and accuracy.

The control inputs determined for each platoon are its speed, lane assignment, size, as well as release time (at on-ramps) and route choice (at bifurcations). The objective function and constraints can correspond to general traffic performance criteria such as total time spent, throughput, emissions, etc., or they could reflect tracking of targets set forth by the area controllers.

In general, this results in a mixed-integer nonlinear optimization problem (if lane allocation and/or size are included in the MPC optimization) or in a real-valued nonlinear optimization problem (if lane allocation and size are assigned using heuristics or logic rules). Mixed-integer optimization problems could be solved using genetic algorithms, simulated annealing, or branch-and-bound methods. Continuous optimization problems can be solved using multi-start sequential quadratic programming, genetic algorithms, simulated annealing, or pattern search.

5.2 MPC for area controllers

In principle, the optimal route choice control problem in IVHS consists in assigning an optimal route to each individual platoon in the network. However, this results in a huge nonlinear integer optimization problem with high computational complexity and requirements, making the problem in fact intractable in practice. Since considering each individual platoon is too computationally intensive for on-line real-time control, the area controllers consider a more aggregate model based on flows of platoons. In this context two approaches have been pursued, viz. one based on

a flow-and-queue model (Baskar et al., 2009a; Baskar, 2009) and one based on a METANET-like model for platoons in an IVHS (Baskar et al., 2009b; Baskar, 2009).

In the first approach the evolution of the flows (on highway stretches) and queue lengths (at junctions) in the network is described using simple queuing models and assuming a fixed average speed in each highway stretch. The control decisions are then the assignment of flows to the links. Although in general this results in a nonlinear, non-convex, and nonsmooth optimization problem, it was shown in (Baskar et al., 2009a; Baskar, 2009) that the resulting optimization problem can be approximated using mixed integer linear programming (MILP), for which efficient branch-and-bound solvers are currently available (Fletcher and Leyffer, 1998). The MILP solution can then be applied directly to the IVHS or it can be used as a good initial starting point for a local optimization of the original nonlinear, non-convex optimization problem.

The second approach is based on a reformulation of the macroscopic traffic flow model METANET (Messmer and Papageorgiou, 1990; Kotsialos et al., 2002) for IVHS. The resulting IVHS-METANET model describes the evolution of the traffic flows through average densities, flows, and speeds in the highway segments. The control decisions in this case are the splitting rates at the network nodes and possibly also the speeds on the links. This then results in a nonlinear non-convex optimization problem with real-valued variables. To solve the nonlinear optimization problem we can use a global or a multi-start local optimization method such as multi-start sequential quadratic programming, pattern search, genetic algorithms, or simulated annealing.

5.3 Interfaces between the different control layers

The higher-level controllers can influence the controller in the level immediately below them in two ways: by specifying weights, set-points, or reference signals in the objective function, or by specifying targets or thresholds in the constraints. The lower-level controller then has to solve an optimization problem of the form

$$\min_{\mathbf{u}(k),\ldots,\mathbf{u}(k+N_\mathrm{p}-1)} J(k) = J_\mathrm{high}(k) + \lambda J_\mathrm{local}(k) \tag{29}$$

$$\text{s.t. } \mathbf{x}(k+j+1) = \mathbf{f}(\mathbf{x}(k+j), \mathbf{u}(k+j), \mathbf{d}(k+j)) \quad \text{for } j = 0,\ldots,N_\mathrm{p}-1 \tag{30}$$

$$\mathbf{u}(k+j) = \mathbf{u}(k+N_\mathrm{c}-1) \quad \text{for } j = N_\mathrm{c},\ldots,N_\mathrm{p}-1 \tag{31}$$

$$\mathbf{C}_\mathrm{high}(\mathbf{x}(k),\ldots,\mathbf{x}(k+N_\mathrm{p}),\mathbf{u}(k),\ldots,\mathbf{u}(k+N_\mathrm{c}-1)) \tag{32}$$

$$\mathbf{C}_\mathrm{local}(\mathbf{x}(k),\ldots,\mathbf{x}(k+N_\mathrm{p}),\mathbf{u}(k),\ldots,\mathbf{u}(k+N_\mathrm{c}-1)) \tag{33}$$

where J_high and $\mathbf{C}_\mathrm{high}$ represent respectively the objectives and constraints (in the form of a system of equations and/or inequalities) imposed by the higher-level controller, J_local is the local, additional objectives that have to be optimized, $\lambda > 0$ is a weighting factor, $\mathbf{C}_\mathrm{local}$ contains the local constraints, and $\mathbf{x}(k+j)$ is the prediction of the state of the traffic system (region, area, highway stretch, depending on the control level) at time step $k+j$, $\mathbf{u}(k+j)$ is the control input at time step $k+j$, and $\mathbf{d}(k+j)$ is the estimate of the traffic demand at time step $k+j$. In addition, the model equations (30) and the control horizon constraint (31) are also included.

The control variables determined by the area controllers are the flows on the links and/or the splitting rates at the nodes with more than one outgoing link (and if speed limits are included, also these speed limits). Once the optimal flows, splitting rates, and speeds have been determined by the area controller, they are sent to the lower-level roadside controllers, which can then translate them into actual speed, route, and lane allocation instructions for the platoons. So in this case the communication goes through the performance criterion J_{high}.

The roadside controllers can provide lane allocation commands and speeds in order to realize the target flows and speeds in the links. These control measures can then slow down or speed up platoons in the links and also steer the platoons in certain directions depending on the imposed splitting rates for the flows. At the nodes, the roadside controller will additionally provide routing instructions for every platoon on the stretch under its supervision. The roadside controller will determine these routing instructions by taking into account the destinations of the platoons and also the imposed splitting rates or the target flows on the adjacent highways.

The roadside controller can combine the speed and route guidance control measures along with on-ramp access timing to control the platoons that enter from on-ramps. The platoon length will play a crucial role while providing routing instructions to the platoons at internal nodes or bifurcation junctions. So if necessary, the roadside controllers can then also provide commands for platoon splits and merges, and determine new platoon compositions and platoon lengths.

6 Challenges and open issues

Now we discuss the main technological, economical, and societal challenges that will have to be addressed when actually implementing an IVHS system.

Although several authors have indicated how control design methods such as static feedback control, MPC, and AI-based control could be used for IVHS and traffic management and control systems based on intelligent vehicles and platoons (see the preceding sections), real integration of these methods is still lacking. This is one of the challenges that still have to be addressed.

Moreover, since IVHS and IVs are nonlinear and often even hybrid (i.e., they exhibit both continuous dynamics and discrete-event behavior (switching)), properties such as stability and robustness of the traffic system have also to be investigated further. In addition, in particular at the platoon level and higher, there is also a need for performance guarantees supported by solid fundamental results.

Two other important remaining open problems in control of IVHS are platoon formation and control, and scalability.

In literature there are no strict rules available on how to form platoons and on how many vehicles to include in a platoon. This can either be specific to a given road or to a destination. There are few articles that deal with vehicle sorting with respect to platoon sizes and platoon formation time, and also on the design of platoon maneuver protocols Hsu et al. (1991); Hall and Chin (2005). Moreover, the design of platoon controllers is certainly not standardized. Especially the particular choice of spacing policy will heavily influence the overall IVHS performance with respect to throughput, safety and fuel consumption. Although some research has been done, this is still an open issue.

Some of IVHS frameworks such as the PATH, Auto21 CDS, and Dolphin frameworks are by nature hierarchical and offer thus a certain degree of scalability with regard to network size. Other frameworks such as PReVENT, SafeSpot, and CVIS are not explicitly hierarchical and are thus not inherently scalable with regard to network size. However, none of the frameworks *explicitly* addresses scalability and the scalability of the frameworks has not yet been investigated in detail in literature. So this is also a topic for future research.

The technical issues outlined above are still open and need to be addressed. Moreover, the IVHS approach requires major investments to be made by both the government (or the body that manages the highway system) and the constructors and owners of the vehicles. Since few decisions are left to the driver and since the AHS assumes almost complete control over the vehicles, which drive at high speeds and at short distances from each other, a strong psychological resistance to this traffic congestion policy is to be expected. In addition, the fact that vehicles can be tracked through the entire network may raise some concerns regarding privacy and liability issues.

Another important question is how the transition of the current highway system to an AHS-based system should occur and — once it has been installed — what has to be done with vehicles that are not yet equipped for IVHS. Other transition issues that have to be taken into account are Fenton (1994): How will the system be funded? What types of accidents can be expected to occur in AHS, in what numbers, and with what consequences? Will bi-directional communication and transfer of information be allowed between IVs and roadside infrastructure? What are the legal implications of an accident, especially if it were caused by system error or a system oversight? How will an AHS implementation be coordinated on an international level? etc.

7 Conclusions

In this chapter we have presented on overview of traffic management and control frameworks for IVHS. First, we have given a short survey of the main control design methods that could be used in IVHS. Next, we have discussed various traffic management architectures for IVHS such as PATH, Dolphin, Auto21 CDS, CVIS, SafeSpot, and PReVENT. Subsequently, we have focused in more detail on the platoon, roadside, and area layers. Finally, we have identified some open issues and future challenges in the further implementation and actual deployment of IVHS traffic management systems, in particular, integration, stability, scalability, and transition issues.

References

A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, March 1994.

K. J. Åström and B. Wittenmark. *Computer-Controlled Systems — Theory and Applications*. Prentice-Hall, Upper Saddle River, New Jersey, 3rd edition, 1997.

E. Barbieri. Stability analysis of a class of interconnected systems. *ASME Journal of Dynamic Systems, Measurement, and Control*, 115(3):546–551, September 1993.

L. D. Baskar. *Traffic Management and Control in Intelligent Vehicle Highway Systems*. PhD thesis, Delft University of Technology, Delft, The Netherlands, November 2009. TRAIL Thesis Series T2009/12.

L. D. Baskar, B. De Schutter, and H. Hellendoorn. Hierarchical traffic control and management with intelligent vehicles. In *Proceedings of the 2007 IEEE Intelligent Vehicles Symposium (IV'07)*, pages 834–839, Istanbul, Turkey, June 2007.

L. D. Baskar, B. De Schutter, and J. Hellendoorn. Model-based predictive traffic control for intelligent vehicles: Dynamic speed limits and dynamic lane allocation. In *Proceedings of the 2008 IEEE Intelligent Vehicles Symposium (IV'08)*, pages 174–179, Eindhoven, The Netherlands, June 2008.

L. D. Baskar, B. De Schutter, and H. Hellendoorn. Optimal routing for intelligent vehicle highway systems using mixed integer linear programming. In *Proceedings of the 12th IFAC Symposium on Transportation Systems*, pages 569–575, Redondo Beach, California, September 2009a.

L. D. Baskar, B. De Schutter, and J. Hellendoorn. Optimal routing for intelligent vehicle highway systems using a macroscopic traffic flow model. In *Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems (ITSC 2009)*, pages 576–581, St. Louis, Missouri, October 2009b.

L. D. Baskar, B. De Schutter, Z. Papp, and J. Hellendoorn. Traffic control and intelligent vehicle highway systems: A survey. *IET Intelligent Transport Systems*, 2011. To appear.

R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, New Jersey, 1957.

R. Bishop. *Intelligent Vehicles Technology and Trends*. Artech House, 2005.

M. Broucke and P. Varaiya. The automated highway system: A transportation technology for the 21st century. *Control Engineering Practice*, 5(11):1583–1590, November 1997.

S. H. Chen, A. J. Jakeman, and J. P. Norton. Artificial intelligence techniques: An introduction to their use for modelling environmental systems. *Mathematics and Computers in Simulation*, 78(2):379–400, July 2008.

K.-C. Chu. Optimal dencentralized regulation for a string of coupled systems. *IEEE Transactions on Automatic Control*, 19(3):243–246, June 1974.

CVIS web site. http://www.cvisproject.org/. Last visited on November 15, 2010.

C. F. Daganzo. *Fundamentals of Transportation and Traffic Operations*. Pergamon Press, 1997.

C. A. Desoer and M. Vidyasagar. *Feedback Systems: Input-Output Properties*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pennsylvania, 2009.

M. L. El-Sayed and P. S. Krishnaprasad. Homogeneous interconnected systems: An example. *IEEE Transactions on Automatic Control*, 26(4):894–901, August 1981.

R. E. Fenton. IVHS/AHS: Driving into the future. *IEEE Control Systems Magazine*, 14(6): 13–20, December 1994.

J. Ferber. *Multi-Agent Systems — An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, Harlow, England, 1999.

R. Fletcher and S. Leyffer. Numerical experience with lower bounds for MIQP branch-and-bound. *SIAM Journal on Optimization*, 8(2):604–616, May 1998.

O. Gehring and H. Fritz. Practical results of a longitudinal control concept for truck platooning with vehicle to vehicle communication. In *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, pages 117–122, November 1997.

M. Gelfand and S. V. Fomin. *Calculus of Variations*. Dover Publications, New York, 1991.

R. Hall and C. Chin. Vehicle sorting for platoon formation: Impacts on highway entry and throughput. *Transportation Research Part C*, 13(5-6):405 – 420, 2005.

S. Hallé and B. Chaib-draa. A collaborative driving system based on multiagent modelling and simulations. *Transportation Research Part C: Emerging Technologies*, 13(4):320–345, August 2005.

D. Hammerstrom. Working with neural networks. *IEEE Spectrum*, 30(7):46–53, July 1993.

F. Hayes-Roth. Rule-based systems. *Communications of the ACM*, 28(9):921–932, 1985.

J. K. Hedrick, M. Tomizuka, and P. Varaiya. Control issues in automated highway systems. *IEEE Control Systems Magazine*, 14(6):21–32, December 1994.

M. R. Hestenes. *Calculus of Variations and Optimal Control Theory*. John Wiley and Sons, New York, 1966.

R. Horowitz and P. Varaiya. Control design of an automated highway system. *Proceedings of the IEEE: Special Issue on Hybrid Systems*, 88(7):913–925, July 2000.

A. Hsu, F. Eskafi, S. Sachs, and P. Varaiya. Design of platoon maneuver protocols for IVHS. Technical Report 96-21, California Partners for Advanced Transit and Highways PATH, University of California, Berkeley, California, 1991.

A. Hsu, F. Eskafi, S. Sachs, and P. Variaya. Protocol design for an automated highway system. *Discrete Event Dynamic Systems: Theory and Applications*, 2(1):183–206, 1993.

P. A. Ioannou and C. C. Chien. Autonomous intelligent cruise control. *IEEE Transactions on Vehicle Technology*, 42(4):657–672, November 1993.

P. Kachroo and K. Özbay. *Feedback Control Theory for Dynamic Traffic Assignment*. Advances in Industrial Control. Springer-Verlag, Berlin, 1999.

H. K. Khalil. *Nonlinear Systems*. Prentice-Hall, Inc., Pearson Eductation International, New Jersey, USA, 3rd edition, 2000.

D. E. Kirk. *Optimal Control Theory: An Introduction*. Prentice-Hall, Englewood Cliffs, New Jersey, 1970.

S. Klinge and R. H. Middleton. Time headway requirements for string stability of homogeneous linear unidirectionally connected systems. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 1992–1997, Shanghai, P. R. China, December 2009.

G. J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice-Hall, Upper Saddle River, New Jersey, 1995.

A. Kotsialos, M. Papageorgiou, C. Diakaki, Y. Pavlis, and F. Middelham. Traffic flow modeling of large-scale motorway networks using the macroscopic modeling tool METANET. *IEEE Transactions on Intelligent Transportation Systems*, 3(4):282–292, December 2002.

W. S. Levine and M. Athans. On the optimal error regulation of a string of moving vehicles. *IEEE Transactions on Automatic Control*, 11(3):355–361, July 1966.

K. Li and P. Ioannou. Modeling of traffic flow of automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 5(2):99–113, June 2004.

J. M. Maciejowski. *Predictive Control with Constraints*. Prentice-Hall, Harlow, England, 2002.

International Organization for Standardization. Transport information and control systems – Adaptive Cruise Control systems – Performance requirements and test procedures, ISO 15622. Technical report, Transport information and control systems, October 2002.

A. Messmer and M. Papageorgiou. METANET: A macroscopic simulation program for motorway networks. *Traffic Engineering and Control*, 31(9):466–470, 1990.

G. J. L. Naus, R. Vugts, J. Ploeg, M. J. G. van de Molengraft, and M. Steinbuch. Towards on-the-road implementation of cooperative adaptive cruise control. In *Proceedings of the 16th World Congress & Exhibition on Intelligent Transport Systems and Services*, Stockholm, Sweden, September 2009.

G. J. L. Naus, J. Ploeg, M. J. G. van de Molengraft, W. P. M. H. Heemels, and M. Steinbuch. Design and implementation of parameterized adaptive cruise control: An explicit model predictive control approach. *Control Engineering Practice*, 18(8):882–892, August 2010a.

G. J. L. Naus, R. Vugts, J. Ploeg, M. J. G. van de Molengraft, and M. Steinbuch. Cooperative adaptive cruise control, design and experiments. In *Proceedings of the American Control Conference*, pages 6145–6150, Baltimore, Maryland, USA, June–July 2010b.

G. J. L. Naus, R. Vugts, J. Ploeg, M. J. G. van de Molengraft, and M. Steinbuch. String-stable CACC design and experimental validation: A frequency-domain approach. *IEEE Transactions on Vehicle Technology*, 59(9):4268–4279, November 2010c.

H. T. Nguyen and E. A. Walker. *A First Course in Fuzzy Logic*. Chapman & Hall Press, Boca Raton, Florida, 2nd edition, 1999.

M. Papageorgiou. *Applications of Automatic Control Concepts to Traffic Flow Modeling and Control*. Lecture Notes in Control and Information Sciences. Springer-Verlag, Berlin, Germany, 1983.

P. M. Pardalos and M. G. C. Resende. *Handbook of Applied Optimization*. Oxford University Press, Oxford, England, 2002.

PReVENT web site. http://www.prevent-ip.org/. Last visited on November 15, 2010.

R. Rajamani. *Vehicle Dynamics and Control*. Mechanical Engineering Series. Springer, New York, NY, 2006.

B. S. Y. Rao and P. Varaiya. Roadside intelligence for flow control in an IVHS. *Transportation Research Part C*, 2(1):49–72, 1994.

J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, Madison, Wisconsin, 2009.

S. G. Ritchie. A knowledge-based decision support architecture for advanced traffic management. *Transportation Research Part A*, 24(1):27–37, January 1990.

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, New Jersey, 2003.

SafeSpot web site. http://www.safespot-eu.org/. Last visited on November 15, 2010.

S. Sheikholeslam and C. A. Desoer. A system level study of the longitudinal control of a platoon of vehicles. *ASME Journal of Dynamic Systems, Measurement, and Control*, 114(2):286–292, June 1992.

S. Sheikholeslam and C. A. Desoer. Longitudinal control of a platoon of vehicles with no communication of lead vehicle information: A system level study. *IEEE Transactions on Vehicle Technology*, 42(4):546–554, November 1993.

S. E. Shladover. PATH at 20 – History and major milestones. *IEEE Transactions on Intelligent Transportation Systems*, 8(4):584–592, December 2007.

S. S. Stanković, M. J. Stanojević, and D. D. Šiljak. Decentralized overlapping control of a platoon of vehicles. *IEEE Transactions on Control Systems Technology*, 8(5):816–832, September 2000.

J. M. Sussman. Intelligent vehicle highway systems: Challenge for the future. *IEEE Micro*, 1 (14-18):101–104, June 1993.

H. J. Sussmann and J. C. Willems. 300 years of optimal control: From the brachystochrone to the maximum principle. *IEEE Control Systems Magazine*, 17(3):32–44, June 1997.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, Massachusetts, 1998.

D. Swaroop and J. K. Hedrick. String stability of interconnected systems. *IEEE Transactions on Automatic Control*, 41(3):349–357, March 1996.

D. Swaroop and K. R. Rajagopal. Intelligent cruise control systems and traffic flow stability. *Transportation Research Part C: Emerging Technologies*, 7(6):329–352, December 1999.

D. Swaroop, J. K. Hedrick, C. C. Chien, and P. Ioannou. A comparison of spacing and headway control laws for automatically controlled vehicles. *Vehicle System Dynamics*, 23(1):597–625, 1994.

D. Swaroop, J. K. Hedrick, and S. B. Choi. Direct adaptive longitudinal control of vehicle platoons. *IEEE Transactions on Vehicle Technology*, 50(1):150–161, January 2001.

G. Toulminet, J. Boussuge, and C. Laurgeau. Comparative synthesis of the 3 main european projects dealing with cooperative systems (CVIS, SAFESPOT and COOPERS) and description of COOPERS demonstration site 4. In *Proceedings of the 11th IEEE Conference on Intelligent Transportation Systems*, pages 809–814, Beijing, China, 2008.

S. Tsugawa, S. Kato, K. Tokuda, T. Matsui, and H. Fujii. An architecture for cooperative driving of automated vehicles. In *Proceedings of the IEEE Intelligent Transportation Symposium*, pages 422–427, Dearborn, Michigan, 2000.

S. Tsugawa, S. Kato, K. Tokuda, T. Matsui, and H. Fujii. A cooperative driving system with automated vehicles and inter-vehicle communications in demo 2000. In *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, pages 918–923, Oakland, California, August 2001.

P. Varaiya. Smart cars on smart roads: Problems of control. *IEEE Transactions on Automatic Control*, 38(2):195–207, February 1993.

P. Varaiya and S. E. Shladover. Sketch of an IVHS systems architecture. In *Vehicle Navigation and Information Systems*, pages 909–922, Dearborn, Michigan, October 1991.

P. Venhovens, K. Naab, and B. Adiprasito. Stop and go cruise control. *International Journal of Automotive Technology*, 1(2):61–69, 2000.

G. Weiss, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, Cambridge, Massachusetts, 1999.

X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.