Available online at www.sciencedirect.com
SCIENCE DIRECT®
Transportation Research Part C 13 (2005) 320–345
TRANSPORTATION RESEARCH PART C
www.elsevier.com/locate/trc

A collaborative driving system based on multiagent modelling and simulations

Simon Hallé *, Brahim Chaib-draa

Département d'informatique et génie logiciel, Université Laval, Sainte-Foy, QC, Canada G1K 7P4

Abstract

Collaborative driving is a growing domain of intelligent transportation systems (ITS) that makes use of communications to autonomously guide cooperative vehicles on an automated highway system (AHS). In this paper, we address this issue by using a platoon of cars considered as more or less autonomous software agents. To achieve this, we propose a hierarchical driving agent architecture based on three layers (guidance layer, management layer and traffic control layer). This architecture has been used to develop centralized platoons, where the driving agent of the head vehicle coordinates other driving agents by applying strict rules, and decentralized platoons, where the platoon is considered as a group of driving agents with a similar degree of autonomy, trying to maintain a stable platoon. The latter decentralized model mainly considers an agent teamwork model based on a multiagent architecture, known as STEAM. The centralized and decentralized coordination models are finally compared using results from simulation scenarios that highlight safety, time efficiency and communication efficiency aspects for each model.

Keywords: Collaborative driving system (CDS); Intelligent vehicles; Teamwork; Multiagent; Car platoon; Intelligent transport system (ITS)

* Corresponding author.
E-mail addresses: halle@iad.ift.ulaval.ca (S. Hallé), chaib@iad.ift.ulaval.ca (B. Chaib-draa).
URLs: http://www.damas.ift.ulaval.ca/~halle/ (S. Hallé), http://www.damas.ift.ulaval.ca/~chaib/ (B. Chaib-draa).

0968-090X/$ - see front matter
doi:10.1016/j.trc.2005.07.004

1. Introduction

Transport systems all over the world are suffering from spreading problems regarding mainly their traffic flow and safety. To address these traffic problems, we generally build more highways, but this solution is greatly limited by the available land areas, which is running low in most cosmopolitan cities. An alternative solution which is growing in popularity is to develop techniques that increase existing roads' capacity by investing in intelligent transportation systems (ITS) infrastructure (Ghosh and Lee, 2000). We can cite as ITS components: advanced transportation management, advanced transportation information system, and commercial vehicle operations. Among these components, there are sub-components such as automobile collision avoidance and electronic guidance system, which are generally sustained by individual technologies as: electronic sensors, wire and wireless communications, computer software and hardware, global positioning system (GPS), etc.

As an enhancement to the very well known cruise control that most cars are equipped with, a variety of adaptive cruise control (ACC) has been proposed in the past years. These controllers have the benefit of maintaining a constant distance with the preceding vehicle using a sensor installed at the front of the vehicle. In more recent years, the cooperative adaptive cruise control (CACC) has been proposed, in order to take advantage of communication systems and allow vehicles to cooperate and increase their autonomy. By pushing the previous concept a little further, automated vehicles equipped with CACC can be regrouped inside platoons to form a network of collaborating vehicles called collaborative driving system (CDS).

In this paper, we focus on the model of platoons of collaborative vehicles organized as part of a CDS. Within such a system, we believe that platooning will ultimately be capable of increasing highways traffic density and safety. Our model is based on vehicles equipped with communicating devices, positioning systems, sensors and an onboard CDS that guides and coordinates the actions of its vehicles in a distributed way. To support this distributed system, we had to develop a coordination strategy that allows vehicles to form and maintain a safe and flexible platoon. In this paper, we present the preliminary results of our platoon coordination model, which aims at being more flexible, thus more adaptive and conceivable for the upcoming years, than previous models. But first, the concept of collaborative driving is described by presenting work relating to our system in Section 2.

2. Collaborative driving

2.1. Motivations

In a collaborative driving system (CDS) (DAMAS-Auto21, 2004), two major problems must be resolved: (i) the distributed control of autonomous vehicles; and (ii) the coordination of each vehicle controller's actions. The first problem can be resolved by a longitudinal and a lateral controller, acting on the vehicle's brake and gas throttles and steering wheel, as part of a guidance system. Such a guidance system provides functionalities as road following, inter-vehicle distance maintenance and velocity maintenance through different control laws.

In order to resolve the second problem (inter-vehicle coordination), most CDS have made the assumption that vehicles were grouped inside a platoon formation (Varaiya, 1993). A platoon is a

group of vehicles that includes a leader, which guides the platoon on the road, and followers, which follow their preceding vehicle at a short distance. In this context, the main task of the CDS is to coordinate vehicles during the entrance (merge) and exit (split) of vehicles from the platoon. In addition, a CDS should communicate information about obstacles or high deceleration ahead, thus allowing platoon members to follow each others at a closer distance. Consequently and as shown by Liang and Peng (2000), these platoons of automated vehicles have the ability of increasing traffic throughput and lowering fuel consumption due to an average higher cruising velocity and lower accelerations.

Our work on CDS comes as part of the first development phase of the Auto21 project (Auto21, 2004), which focuses on the longitudinal control of vehicles inside one platoon led by a human driver. Through the following development phases of this project, more autonomy will be given to our vehicles, in order to end up with a fully autonomous platoon configuration in which all vehicles would be controlled both laterally and longitudinal, including the leader that would be automated instead of being a human driver. Therefore, even though our initial platoon model requires a human driver as a leader, the vehicle coordination system we are developing aims the final phase of our project, in which vehicles do not rely on a specific human driver, similarly to a platoon of CACC equipped vehicles.

In this paper, we present the CDS we developed to support the first phase of the Auto21 project, as part of a decentralized vehicle coordination model. More specifically, we aim to incorporate the vision a multiagent system to the platoon architecture and coordinate vehicles through a teamwork for agents model (Tambe and Zhang, 2000). Such an approach incorporates autonomous agents in each vehicle to use their respective communication system and coordinate each others in a decentralized platoon model. Before presenting further details on our system, we first define our work in the perspective of related research projects.

2.2. Related work

Work relating to our problem can be divided in two major groups presented below: work on adaptive cruise control (ACC) and its derived versions; and work on collaborative automated vehicles.

2.2.1. Work on adaptive cruise control

The work of Ioannou and Stefanovic (2003) is an example of ACC relating to the longitudinal controller that vehicles use to maintain inter-vehicle distances within our CDS. Research on ACC mainly focused on control laws that enhanced comfort and lowered fuel consumption by moderating the vehicle's accelerations. In our application, instead of focusing on longitudinal control laws, we mostly rely on communications to adapt our controllers to changing situations ahead. This is similar to CACC systems, which integrate a communicating device to an ACC. VanderWerf et al. (2001) proposed a CACC which continuously transmitted the acceleration and braking capacity information to its follower via point-to-point vehicle–vehicle communication. Hedrick et al. (2003) used an event-driven CACC, in which a CACC equipped vehicle communicates before changing lane, entering the highway or when it senses a high deceleration. This resulted in vehicles reacting with softer acceleration/deceleration when a vehicle was merging a lane, because of the longer reaction time provided by communications. Our CDS is similar to the model of

Hedrick et al. (2003), since our vehicle model uses a longitudinal controller coupled with a communication system, which provides similar information. In contrast with Hedrick et al. (2003), our model uses communications at a higher level in our architecture, to act on the longitudinal controller and modify its behavior considering communicated information on the traffic ahead. In addition, to support platoon-oriented tasks, our communications are based on structured, but flexible protocols, instead of simple one-way conversations.

2.2.2. Work on platoons of automated vehicles

Platoon-oriented architectures are very similar to the architecture of our CDS, which was initially inspired by some of them. Those architectures address the vehicle control and guidance aspects, and introduce an inter-vehicle coordination system that can interact at different levels. We denoted three different platooning projects, which proposed solutions to the control, guidance and coordination of automated vehicles. The vehicle control aspects refer to the lower and higher-level longitudinal controllers, used to maintain specific velocities or headway. The vehicle guidance aspects refer to the a higher-level system that plans vehicle control actions in time or according to events. The coordination aspects refer to the use of communications in order to coordinate the actions of neighboring vehicles or share general traffic information. Note that in our CDS, we do not address the use of communications for traffic information sharing, since the current phase of our project focuses on the coordination of a single platoon.

California PATH project addressed the problem of platooning by developing an architecture described in Lygeros et al. (1998), which has been used in many different platooning scenarios (Howell et al., 2004). This architecture was reused and adapted to our project, as shown later in Section 4. Lygeros et al. (1998) defined their architecture as a hierarchical system, where the coordination and networking aspects are placed at the top. In the PATH project, different coordination models have been proposed. Howell et al. (2004) proposed a decentralized coordination model in which each vehicle has a supervisor that can interact with other vehicles' supervisors upon the occurrence of certain events. Bana (2001) proposed another coordination model, which addresses both the traffic management and vehicle tasks coordination aspects. In a first time, this model relies on the road-side communications system that determines the vehicle entry flow at a specific highway segment and recommends actions as lane changes to vehicles in this segment. In a second time, a networking system on each vehicle receives the previous recommendations and coordinates platoon merging and splitting actions with neighboring vehicles. In our CDS, we aim this second level of coordination, i.e., the coordination of split and merge tasks. Our coordination model, based on team of agents, is similar to the decentralized supervisors of Howell et al. (2004), but very few details were given on this supervisor model so it is very difficult to compare it to our model.

Tsugawa et al. (2001) developed a platoon architecture similar to PATH's architecture, except that they did not use a road-side infrastructure to coordinate their platoons. They developed a Wireless LAN (communication network formed by neighboring vehicles) model based on a token-ring (Sakaguchi et al., 2000), which is used to communicate each vehicle's dynamic state inside a neighborhood. Although our architecture is similar to Tsugawa et al. (2001), our communication model represents a more flexible and optimized use of the inter-vehicle communications.

The ARCOS project (Blosseville et al., 2003) aims to develop a communication infrastructure for vehicles equipped with ACC. Their inter-vehicle communication system enables vehicles to

share their dynamic state. In the ARCOS project, the collaboration among vehicles is done in order to provide the following functions: (i) notifying hazardous events to vehicles; (ii) regulating headways; (iii) anticipating collisions; (iv) preventing road departure. This project aims to resolve problems very similar to our long-term goals and the model that it sustains, based on ACC, is comparable to our architecture.

2.3. Paper outline

As it was previously presented, work relating to our CDS has mostly proposed reactive communication models based on time or event-based communications. In our approach however, we aim to: (i) develop a communication model in which the system can reason on the situation and communicate when it is necessary; and (ii) develop a more flexible platoon model that does not rely only on its leader and tries to incorporate most platoon members to support the execution of platoon-related tasks.

In this paper, we address the platoon coordination issue by first describing the simulator we used as our test environment in Section 3. Then, Section 4 presents the hierarchical control architecture we adopted to model our automated vehicles. Section 5 then describes our vehicle coordination system, implemented using centralized to decentralized approaches. Section 6 reports results based on simulation scenarios that were used to compare centralized coordination models with teamwork coordination models. Finally, Section 7 presents a discussion, followed by the conclusion.

3. Simulation environment

The environment in which our CDS has been tested is a vehicle simulation software that we developed to meet the specific needs of our project. This section first introduces our simulated environment by putting it in the perspective of other simulators. Then, the capabilities of our simulator are described, followed by the presentation of the type of scenarios we currently support.

3.1. Related simulation environment

A variety of software simulators have been developed to test autonomous driving systems. These simulators can be regrouped in two major types of environments: simulation of a single vehicle; micro-simulation of highway traffic. The first category refers to simulators like CarSim,[1] which include detailed models of the dynamics of a car. The second category refers to a higher-level kind of car simulation similar to Quadstone Paramics suite of microscopic traffic simulation tools.[2]

Our simulator targets a category in between the two previous categories of simulation environments. California PATH'S Smart AHS Simulator is a great example of such an ''hybrid'' simulation environment (Antoniotti et al., 1997). This simulator offers a great level of car dynamics

[1] For more information, visit http://www.carsim.com
[2] For more information, visit http://www.peiramics-online.com

simulation and supports traffic micro-simulations. Our simulator does not support high-level traffic simulations at this time, since our initial test only required one platoon of vehicles. On the other hand, our car dynamics model offers a great level of details that allows us to study the behavior of longitudinal and lateral vehicle controllers. Our simulator therefore focuses on the simulation of a limited amount of automated vehicles, by offering sensors and communication tools, along with a platooning scenario management tool.

3.2. HESTIA simulation environment

Our CDS simulator, called HESTIA, is based on our own 3D simulation engine which currently supports the simulation of an environment composed of about a dozen automated cars (depending on the details of their 3D model) navigating on a straight, one way, two lanes, highway segment (Hallé et al., 2003). The simulation runs in continuous time and it is controlled by a clock generating time step events. Each time step has been set to a value of 20 ms, which was considered as the highest value we could use to receive substantial results from the differential equations of the vehicle dynamics. The simulator's engine is based on JAVA 3D™'s technology, which provides the real-time 3D display of our simulation scenarios, from different angles (static or moving cameras), as shown by a screen-shot our simulator in Fig. 1. Our highway environment is mainly composed of the 3D model of a road, loaded as a static model. The 3D road model can be decomposed and assigned attributes. Therefore, it could eventually be used to build a road network, with different road conditions.

The simulated vehicle model developed in HESTIA includes longitudinal and lateral vehicle dynamics, wheel model dynamics, engine dynamics, torque converter model, automatic gear shifting, throttle/brake actuators and steering wheel actuator. To support our driver model (described in Section 4), each vehicle model possesses an interface that allows the driver to control:

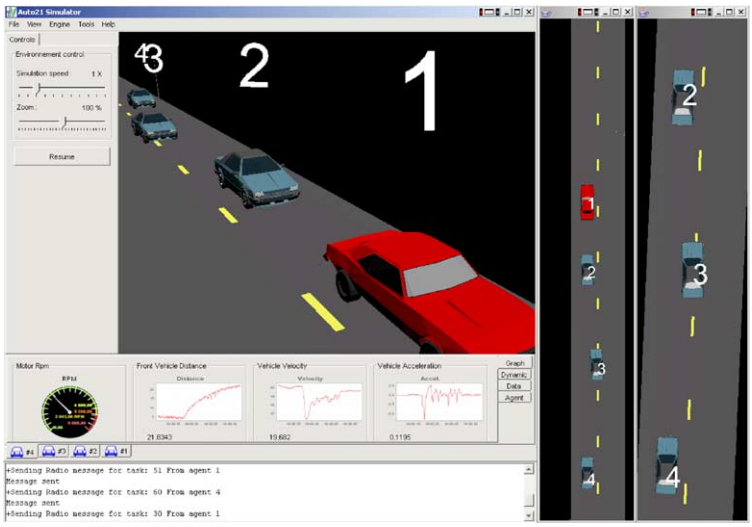

Fig. 1. The merge manoeuvre within a platoon formation in the HESTIA simulator.

(i) throttle/brake actuators; (ii) steering wheel actuator; (iii) external and internal sensors; (iv) communication receiver/transmitter.

Our sensor models were developed using the 3D engine of JAVA 3D™, and in the vehicle model presented in this paper, each follower vehicle is equipped with a vehicle-based laser sensor. This sensor provides information on the front object's (a vehicle) distance and difference of velocity, for distances up to 100 m, using an abstract model of laser. The second sensor model, used for high-level navigation, is a GPS, which gives real-time information on the vehicle's position (latitude, longitude), mapped in a two dimensions system. HESTIA finally includes the simulation model of a radio transmitter/receiver, which is included as part of our automated vehicle model to provide two ways point to multipoint communications. We will not go further on a detailed representation of the simulator's components, as it is out of scope for this paper, but the readers can refer to Hallé (2005) for more information.

3.3. Simulated driving scenarios

One of the primary goals of a CDS is to maintain the platoon formation stable, and therefore, the two simulated scenarios we focus on in this paper are the two main disturbances in this formation: a vehicle splitting and a vehicle merging the platoon. These two scenarios are represented in Fig. 2 and they can be detailed as follows for a better understanding:

(A) *A vehicle splitting* happens when a vehicle member of a platoon decides to leave it, thereby forming two non-empty platoons. The split manoeuvre is executed through three majors steps illustrated in Fig. 2(A):

*S*1: The splitting vehicle (*F*2 in Fig. 2(A)) communicates its intention of leaving the platoon. The platoon formation reacts by modifying the distances at the front and rear of the splitting vehicle.

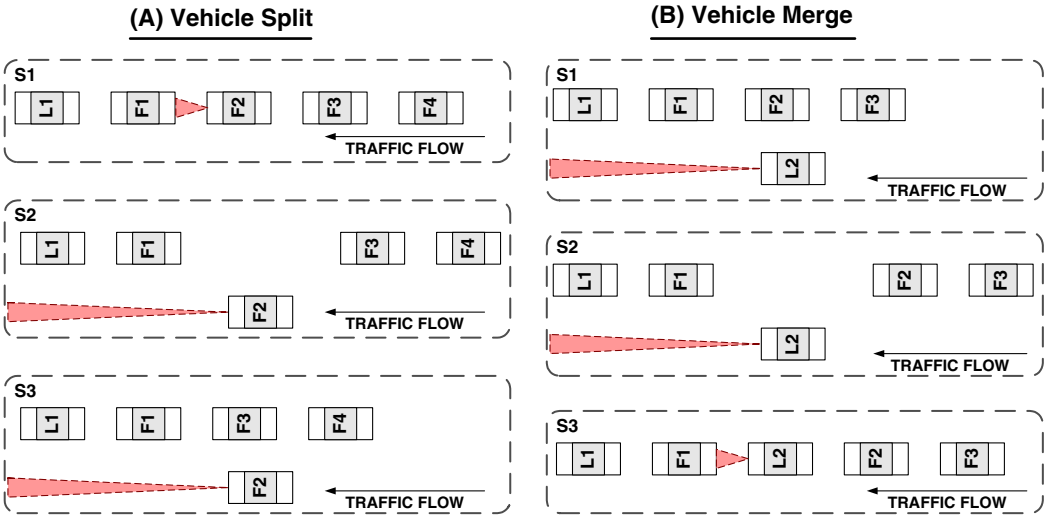

Fig. 2. The three steps of: (A) the removal (split); and (B) the insertion (merge) of a vehicle in the platoon.

*S*2: When the platoon formation gains stability after the gaps creation, the splitting vehicle *F2* changes lane, while the rest of the platoon followers keep the same distance.

*S*3: After the splitting vehicle has safely left the platoon, the gap created for its departure is closed, thus forming back the precedent platoon, minus vehicle *F2*.

(B) *A vehicle merging* is the exact opposite of a split manoeuvre: two non-empty platoons merge together to become one. To execute this manoeuvre, the merging vehicle must be part of a platoon formed of one vehicle (itself), like vehicle *L2* in Fig. 2(B). The merge manoeuvre is executed through three major steps illustrated in Fig. 2(B):

*S*1: The merging vehicle first communicates to another platoon a query to join it. In the example of Fig. 2(B), the platoon led by vehicle *L1* accepts *L2*'s query.

*S*2: *L1*'s platoon reacts by creating a larger space between *F1* and *F2* and communicates the dynamic position of this space to the merging vehicle. The merging vehicle then modifies its velocity to join the meeting point (be parallel to the gap).

*S*3: *L2* verifies if it is safe to merge and it changes lane to enter the platoon led by vehicle *L1*. Once the merged vehicle has stabilized its inter-vehicle distance, the platoon reaches its precedent formation plus one vehicle (*L2*), by diminishing the distances in front and behind *L2*.

In order to test the previous scenarios, our vehicles' lateral automated control is simulated to enable us to perform the lane changes involved in the merge and split manoeuvres. The lateral guidance system of our current system could then be seen as the simulation of the human driver's steering behavior or as the first phase of the lateral guidance system. This subject being out of scope for this paper, which focuses on communications, we will not give further details on this lateral controller.

4. Hierarchical architecture for collaborative driving

The architecture we adopted for our driving system is based on a hierarchical control model (Auto'21, 2003). This model uses a more reactive system as the bottom of the architecture and moves forward to a more deliberative system as it raises to the upper levels. Finally, as we related to other collaborative driving models, our hierarchical architecture was also inspired by Tsugawa's architecture (Tsugawa et al., 2000) and other concepts coming mainly from the PATH project (Lygeros et al., 1998). The resulting architecture has three major layers: *guidance layer*, *management layer and traffic control layer*, as indicated in Fig. 3.

4.1. Guidance layer

This layer has the function of sensing the conditions and states ahead and around the vehicle and activating the longitudinal or the lateral actuators. For the *intelligent sensing* sub-layer, the inputs come from sensors for speed, acceleration, raw rate, machine vision, etc. The *guidance* layer

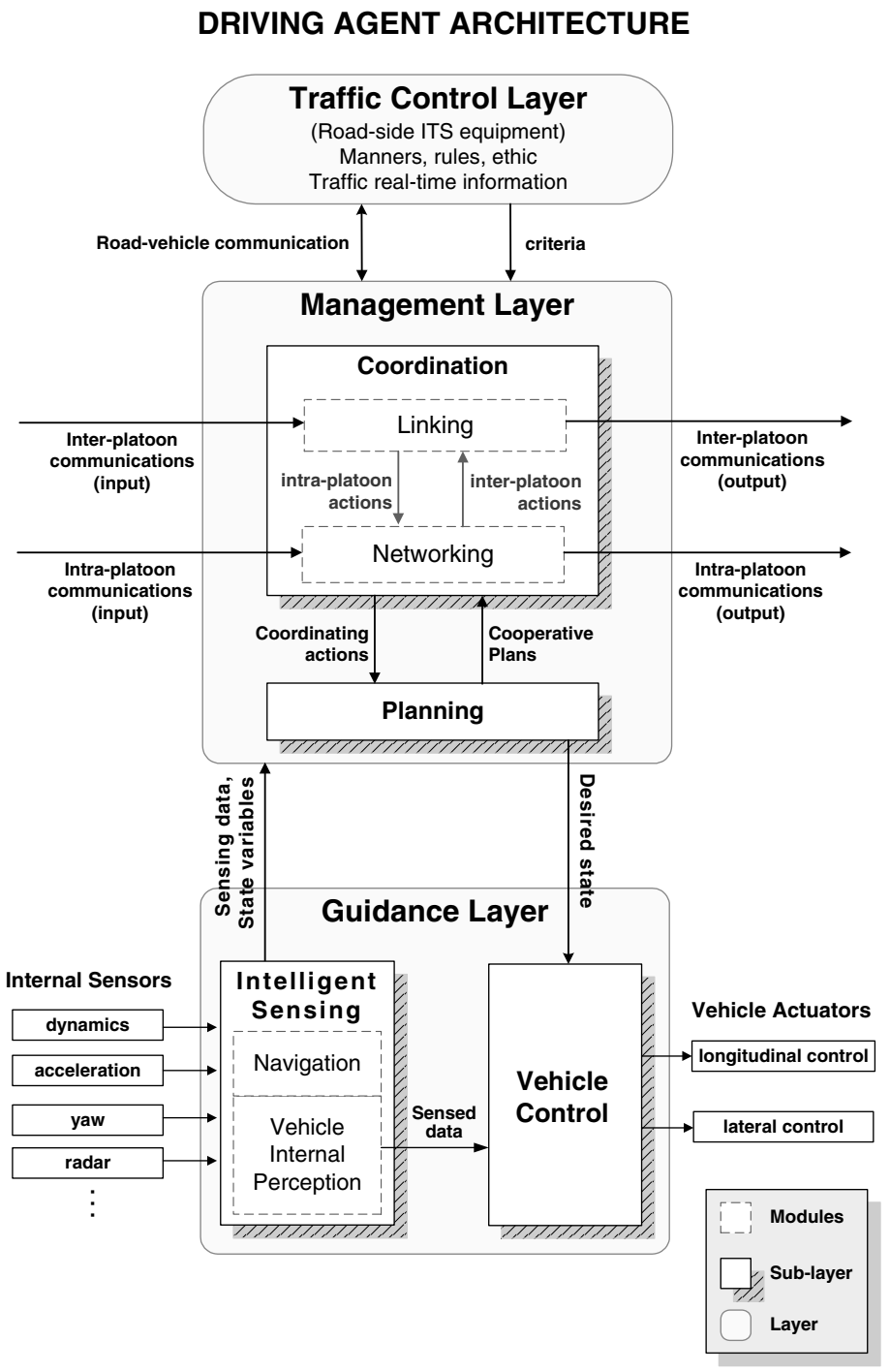

Fig. 3. Hierarchical driving architecture.

outputs sensing data and vehicles state variables to the vehicle *management* layer, which in turn sends back "desired state" queries in the form of steering and vehicle velocity queries to the *guidance* layer. These queries are finally applied by the *vehicle control* sub-layer, which includes lateral controllers and the longitudinal controllers developed by our partners at Sherbrooke University (Huppe et al., 2003).

4.2. Management layer

This layer determines the movement of each vehicle under the cooperative driving constraints using data from: (a) the *guidance* layer; (b) vehicles coordination constraints through the inter-vehicle communication; and (c) the *traffic control* layer through the road–vehicle communication. To determine the movement of each vehicle under the cooperative constraints, the *management* layer needs to reason on the place of the vehicle in its platoon when the vehicle stays in the same lane (intra-platoon coordination), and its place in a new platoon when the vehicle should change lane (inter-platoons coordination). The first type of coordination is handled by the *networking* module and the second by the *linking* module, together forming the *coordination* sub-layer. Generally, the task of the *linking* module is to communicate with the *traffic control* layer to receive suggestions on actions to perform. Resulting from these suggestions, the architecture's *linking* module reasons about the place of its vehicle on the highway and it coordinates lane change actions with neighboring vehicles (inter-platoons coordination). Note that the inter-platoon coordination refers to the coordination of lane change actions which result in the lower-level coordination of a platoon split or merge task (intra-platoon coordination). We did not develop any inter-platoon coordination at this time, but the reader can refer to Bana (2001), which details such a model, based on vehicle cost in a highway segment considering traffic flow.

Following the synchronization of neighbors' lane change requests (inter-platoon coordination), vehicles have to coordinate the actions involved in the execution of specific lane change (intra-platoon coordination). For example, if a vehicle leaves a platoon to enter a new one during its lane change, it has to coordinate both a split and a merge manoeuvre. This coordination task is handled by the *networking* module, which is responsible of the intra-platoon coordination and thus, the platoon formation's stability. The *networking* module coordinates driving actions with other driving agents involved in the manoeuvre (split or merge) to finally plan a series of local actions using the *planning* sub-layer. This sub-layer schedules driving actions (in time or according to events) that are locally executed by sending "desired state" queries to the *guidance* layer.

4.3. Traffic control layer

This layer is a road-side system composed of infrastructure equipments like sign boards, traffic signals and the road–vehicle communications as well as a logical part including: social laws, social rules, weather-manners and other ethics (more specific to Canada), etc. As mentioned in Section 4.2, this layer communicates traffic information to the *Linking* module, in order to support the inter-platoon coordination.

5. Communication and coordination methodologies for platoons of vehicles

Communication has proven to be very useful for collaborative driving systems (CDS), by providing faster response time, more efficiency and by enhancing safety (Xu et al., 2002), but we must define the most efficient way of using it, in order to take full advantage of this technology. The different possible communication methodologies for the platoon of vehicles are implemented in the *coordination* sub-layer of Fig. 3. For the coordination of the platoon and its two main manoeuvres (split and merge) we describe four coordination models and outline their differences in Section 6. Those four coordination models are presented in the following order, by focusing on the teamwork approach: (i) hard-centralized; (ii) centralized; (iii) decentralized; and (iv) teamwork. Note that in each of our coordination models, the guidance and control systems are decentralized for every vehicle involved (Huppe et al., 2003).

Fig. 4 illustrates the differences in the communicative behavior of each coordination model by showing which vehicles are involved in the split and merge manoeuvres, and whether each vehicle "surely" communicates or not. In Fig. 4, vehicle *L* represents the platoon leader, *Fi* represents followers (with *i* as an index incremented following the leader), *S* represents a vehicle splitting from the platoon and *M* represents a vehicle merging in the platoon. Inter-vehicle (point-to-point) communications are represented by black arrows with an arrow shape which indicates that: these two vehicles exchange one to many message; or these two vehicles do not always exchange messages (0 to many). Accordingly, Fig. 4 outlines the fact that every communication in the centralized model involves the leader, while the communications in the decentralized model only involve two vehicles. Moreover, the communications in the teamwork model involve most of the platoon

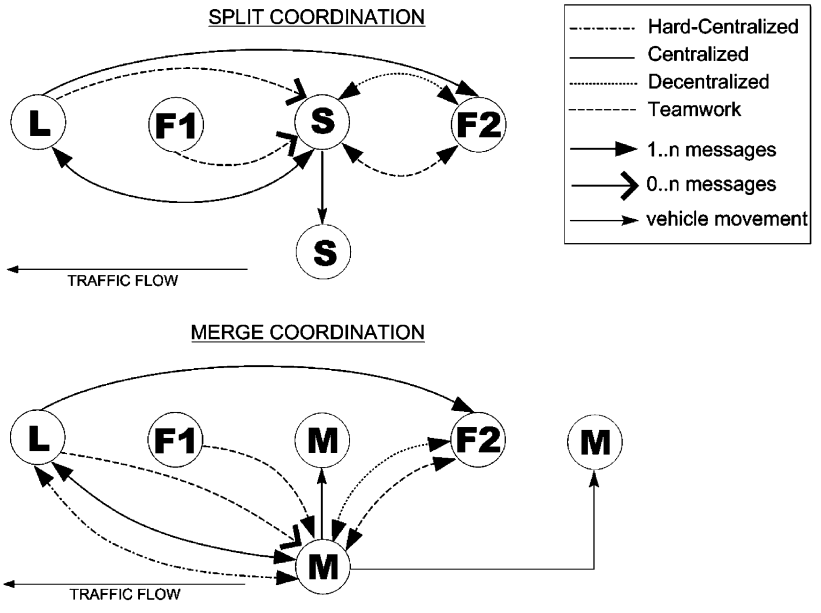

Fig. 4. Four coordination models of the merge and split manoeuvres.

members, but they do not necessarily communicate (0–*n* messages) during the manoeuvre's execution, as it is shown in Section 5.3.

5.1. Centralized platoon coordination

In a centralized platoon coordination model, the communications are centered on one "master vehicle" giving orders to the rest of the platoon: the leader. In this case the leader is the head vehicle of the platoon, and as mentioned earlier, this vehicle is driven by a human (simulated) in our first phase of development. To maintain the platoon formation, the leader is the only entity that can give orders, in which case the followers only apply requested changes.

During a split manoeuvre, three vehicles are involved: (1) the leader; (2) the splitting vehicle; and (3) the vehicle following the splitting vehicle (vehicle *F3* in Fig. 2(A)). During a merge, the same configuration of vehicles is involved: (1) the leader; (2) the merger; and (3) the vehicle which will follow the merged vehicle after its lane change (vehicle *F2* in Fig. 2(B)). For both of these manoeuvres, the merger or splitter first communicates its need to do a manoeuvre, and then, the leader communicates requests for inter-vehicle distance, change of lane, meeting point or velocity to involved vehicles.

For the merge manoeuvre, we have defined two sub-models: hard-centralized and centralized. The hard-centralized model simplifies the task and only requires two vehicles to communicate, by requesting the merging vehicle to always merge at the end of the platoon. In the centralized model, the leader specifies the optimal in-platoon merging position, considering the merging vehicle's position (parallel to the platoon). Thus, the centralized model requires three vehicles to execute a merge (leader, merger and gap creator), while the hard-centralized model requires only two (leader and merger).

5.2. Decentralized platoon coordination

In the concept of a decentralized platoon coordination, the leader is still the platoon representative, but this is only for inter-platoon coordination. Thus, every platoon member has a knowledge of the platoon formation and is able to react autonomously, communicating directly with each others. An agent's common knowledge is initialized when it enters the platoon and is updated using the broadcasted information about new vehicles' merge or split (done at the end of such manoeuvres). Common knowledge refers to a database (knowledge base) of information about platoon members. This database is initialized by setting the ID and "in platoon position" (position following the leader) of each vehicle. The information about platoon members dynamic state (position, velocity, acceleration, etc.) is also initialized/updated if a vehicle communicates such information, i.e., in the merge manoeuvre, the merger communicates its position and velocity to platoon members.

In the decentralized model, platoon members use a set of social laws to determine each member's task in a manoeuvre, instead of being assigned tasks by the leader. Social laws restrict or dictate the driving agent's behaviors and its relative actions (Shoham and Tennenholtz, 1995). In our application, social laws are used to locally dictate which communicative action is possible for a given agent, considering its state in the platoon, without requiring further communications.

The decentralized model represents the simplest decentralization approach and does not rely on any existing framework or complex distributed planning theory, but it tries to lower the

communications as much as possible by simply using social laws. In this model, the leader is only in charge of maintaining the platoon safety by notifying others of any emergencies ahead, similarly to the centralized model. For the split manoeuvre, only two vehicles are involved: the splitter and the vehicle following the splitter (vehicle *F3* in Fig. 2(A)). For the merge, once the merging vehicle has chosen a platoon, only two vehicles are involved as well: the merger, the vehicle which will follow the merged vehicle after its lane change (vehicle *F2* in Fig. 2(B)). For these two manoeuvres, we eliminate the intermediate, i.e., the leader in the centralized model, because every platoon member has the knowledge of its platoon configuration. Thus, by using a set of social laws for our driving agents, the actions of creating a safe gap for the split and merge manoeuvres can be handled by a platoon member without being assigned by the leader.

5.3. Multiagent teamwork for platoons

The previous decentralized model can be improved using a better structured organization, as the teamwork for agents, a concept gaining in popularity these recent years, in the field of multiagent systems. In this context, a teamwork architecture like STEAM (Tambe and Zhang, 2000) can be used to assign roles to platoon members within a predefined team hierarchy. The teamwork concept results in most vehicles of a platoon to be involved in tasks and communicate if necessary, as shown by the dotted lines of Fig. 4, representing "possible" communication. For the Auto21 project, we adapted STEAM's communication framework (based on STEAM operators) considering our specific needs. We then defined the required CDS team structures and developed a set of driving plans as domain-level operators, which can be used inside STEAM's framework. This section presents the previous aspects starting with the Auto21 team formations, followed by the description of STEAM framework and domain-level operators.

5.3.1. Auto21 team formations

For the Auto21 project, we have defined three major teams: (i) the "platoon formation" team; (ii) the "split task" team; and (iii) the "merge task" team. The "platoon formation" team is a persistent team using persistent roles for long-term assignments, as it is the case for the platoon formation. The two latter types of team are task teams using task-specific roles, for shorter-term assignments, since the teams crated to support the split and merge stop existing after the task completion. Each of these teams is formed of different agents that must fill all the roles that are specified in the team's definition. As an example, the "split task" team is defined by Fig. 5's tree, in which the leaf nodes represent roles and the internal node (only one in this case) represents a sub-team (the task observers).

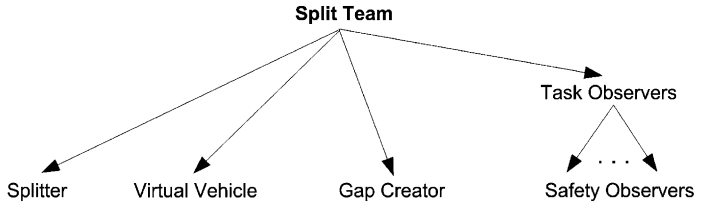

Fig. 5. Split task team's role organization.

The ''platoon formation'' team is the simplest of our three teams, in which each driving agent holds the intention of maintaining a stable and safe platoon formation. At the moment, we consider that each member of the ''platoon formation'' has the same task, which consists of following the front vehicle in a safe manner. Hence, this formation only requires two persistent (long-term assignments) roles:

- *Leader*: a role filled by the head vehicle, which mainly communicates with others using selective communication (SC) operators (defined in Section 5.3.2). Since the goal here, is to maintain a stable platoon formation, an unsafe deceleration can be seen as a percept that could endanger the goal achievement, therefore influencing the leader to inform others of this fact using SC operators. The probability of such a communicative act is discussed later.
- *Follower*: a role filled by all the platoon members that are not at the head. At the moment, each follower's goal consists of maintaining a safe time headway (distance in time) with the preceding vehicle, in order to maintain the platoon's stability. An agent in this role does not need to communicate any information since the automated driving system of each vehicle is capable of maintaining the platoon stable in the context of a ''platoon formation'' team. Thus, the task of maintaining a safe time headway is realized by using the vehicle's front sensor and possible information from the leader.

The ''merge task'' team being similar to the ''split task'' team, we only depict the ''merge task'' team in this paper. As shown in Fig. 5, the split team (similar to the merge team) is formed of four different roles and a sub-team: the *task observers* sub-team. Each role is described below by referring to the actions they execute and the place they occupy in the platoon illustrated in Fig. 2(B).

- *Merger*: a role filled by the agent which initiates the ''merge task'' team by broadcasting its intention to merge a platoon (vehicle *L2* in Fig. 2(B)). When an agent fills this role, it instantiates a series of plans dependent of contextual issue, which allow it to move beside the gap created inside the platoon and change lane. To support a safe execution of the lane change required to merge a platoon, the *Merger* uses a virtual representation of its preceding vehicle. For instance, when *L2* is in state *S2*, it creates a virtual representation of *F1* before changing lane. This allows its longitudinal controller to follow *F1*, even though its front laser cannot sense it until it is half-way through its lane change. To support the creation of this virtual vehicle representation, *L2* uses information that *F1* communicates about its dynamic state, through the *Virtual Vehicle* role presented below.
- *Gap Creator*: a role filled by the agent driving the vehicle behind the merging position, in the platoon (vehicle *F2* in Fig. 2(B)). Within this role, an agent defines the entry position for the *Merger*, since its vehicle will be behind the *Merger* after the lane change. When an agent fills this role, it monitors the gaps between the distances sensed by its front laser. This allows the *Gap Creator* to conclude on the arrival of the merging vehicle in the platoon and react by requesting a new headway to its longitudinal controller.
- *Virtual Vehicle*: a role that was introduced to ensure a stable execution of our manoeuvres. This role helps the manoeuvre executor (splitter or merger), when it is in a different lane, to follow the vehicle that was or will be in front of it. For the split manoeuvre, the *Virtual Vehicle* role is filled by the vehicle preceding the splitter (*F1* in Fig. 2(A)). For the merge manoeuvre, the same

role is filled by the vehicle that will precede the merging vehicle, after it has changed lane (*F1* in Fig. 2(B)). In the *Virtual Vehicle* role, vehicle *F1* communicates information about its velocity, if this velocity comes to change while the splitting/merging vehicle is changing lane.

Since our vehicles only use a front laser to follow their preceding vehicle, a splitting vehicle looses the perception of its preceding vehicle at the end of its lane change and a merging vehicle does not have the perception of its preceding vehicle at the beginning of its lane change. Therefore, using the information communicated by *F1*, the splitting and merging vehicles can create and update a virtual representation of their preceding vehicle (*F1*). This allows our manoeuvering vehicles to follow *F1* even though it cannot be sensed by their laser.

- *Safety observers*: a role filled by one or more agents. The constraint on the role fillers, is that they must be in a position ahead from the manoeuvre executor, so they can monitor dangers in advance. Using the selective communication operator presented in Section 5.3.2, agents in the *safety observers* role communicate their belief about dangers or unsafe deceleration to others, by taking in account the dangers of sudden movements during the execution of a manoeuvre.

5.3.2. Team operators

Agents evolving in the STEAM framework are able to execute two possible types of operator: domain-level; and architecture-level (STEAM operators). Domain-level operators have been defined for our application in the form of a hierarchical operator tree where top nodes refer to team operators and bottom nodes refer to local operators. Each team domain-level operator represents a synchronized action (plan) executed by more than one team member. This kind of operator is based on the team's mutual beliefs, which supports the coordination of team members' local actions considering the team's belief.

To ensure the safe execution of the domain-level operators, STEAM introduced its framework operators, detailed in Tambe and Zhang (2000). STEAM's operators include: coherence preserving (CP) operators; monitor and repair (MR) operators; selective communication (SC) operators. CP operators preserve the team coherence by forcing team members to communicate beliefs that would make the current team's goal unachievable, achieved, or irrelevant. MR operators monitor the relation among team members to react to an agent's failure to fill in its role and replace it, when a "replacement agent" can be found. SC operators have been customized for our application, as they allow our "split task" and "merge task" teams to manage their communications with the splitting or merging vehicle. Therefore, this operator is detailed below for a better understanding of its use in our CDS.

The SC operator synchronizes team members' mutual beliefs during the execution of domain-level operators. A SC operator simply monitors the agent's local beliefs and compares them with the team's mutual beliefs. If it considers that the difference between these two beliefs is important enough, it automatically communicates the agent local belief to other team members and makes its local belief a mutual belief. In order to determine if a new belief should be communicated, the SC operator uses the decision tree presented in Fig. 6. According to this tree, the SC operator communicates the new belief considering a reward based on the following variables:

- ρ: the probability that the new belief is not known by its teammates.
- σ: the probability that the new belief opposes a threat to the execution of the current team operator.

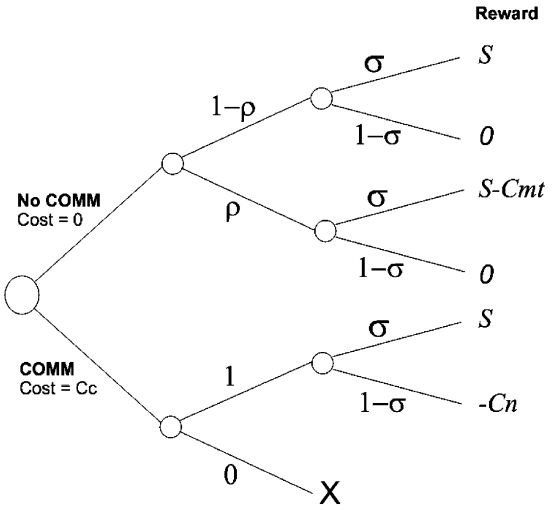

Fig. 6. Decision tree on team communicative acts, from Tambe and Zhang (2000).

- C_c: the cost of communication.
- C_n: the cost of nuisance.
- C_{mt}: the cost for miscoordination.
- S: a reward for synchronization of the team's belief during the execution of a team operator.

By using the decision tree in Fig. 6, the SC operator chooses to communicate if and only if the reward of making a new belief "mutual" is higher than the cost of communications. With a probability of $1-\sigma$, the new belief is not a threat to the team operator, and therefore, there is no reward relating to this belief. If the agent chooses not to communicate (*No COMM* (*NC*)) and the belief is a threat, with a probability $1-\rho$, this belief was already known by its teammates so the agent receives the standard reward of S. However, with a probability ρ, this belief was not mutual, so the agent receives the standard reward of $S - C_{mt}$, where, in our application, C_{mt} depends on the difference between the local and mutual belief (if the local belief is much different from the mutual belief, C_{mt} is high). Therefore, by referring to Fig. 6, the expected utility of not communicating can be defined as $EU(NC) = \sigma * S - (\rho * \sigma * C_{mt})$.

In the situation where the agent decides to communicate (*COMM* (*C*)), a cost of communication C_c is applied to further possible rewards and this cost is fixed considering the current available communication bandwidth. Following a decision to communicate, the new belief is definitely mutual, so the second branch is irrelevant in Fig. 6 (probability 0). However, the new belief can be a threat with a probability σ, in which case the agent receives a reward of S. Nevertheless, with a probability $1-\sigma$, this was not a threat and the communication of this belief has a cost of nuisance C_n to other team members. This cost depends on the type of information that is communicated, but it is usually set to discourage agents from communicating information that may disturb other team members, when it is not necessary. According to this definition, the expected utility of communicating a new belief to team members can be summarized as $EU(C) = \sigma * S - (C_c + (1-\sigma) * C_n)$. The two previous equations on expected utility can be merged to represent the

decision of the selective communication (SC) operator, which communicates when $EU(C) > EU(NC)$, i.e., iff:

$$\rho * \sigma * C_{\mathrm{mt}} > (C_{\mathrm{c}} + (1 - \sigma) * C_{\mathrm{n}})$$

This kind of decision-theoretic selector being part of the STEAM framework, we use it for all of the roles presented earlier. Probabilities, cost and rewards used by the communication decision tree are initialized according to common knowledge on collaborative driving system (CDS) and should be adapted through testing using offline learning approach on patterns of communication within the team. For the moment, the SC operators have been very useful to determine when a *safety observers or a virtual vehicle* should communicate its new beliefs. For instance, if the *virtual vehicle* (vehicle *F1* in Fig. 2(B)) has to modify its velocity during the merge, the probability ρ that this new information on *F1*'s velocity is commonly known mainly depends on the probability $P(L2, F1)$ that the merging vehicle *L2* has *F1* in its sensor's range (if *L2* is in the platoon). Furthermore, the probability σ that this information opposes a threat to the merge manoeuvre depends on the difference between *F1*'s local belief on its velocity and the team's mutual belief. If the team is highly out of synchronization, the agent will communicate at a higher probability. Finally, cost C_{n} will be low for this type of belief, while cost C_{mt} will be higher, since changes on the *virtual vehicle*'s velocity affect the platoon safety.

6. Experiments and results

To develop the previous coordination models, we have used an agent development toolkit called JACK Intelligent Agents™ (AOS, 2004), which supports the belief desire intention (BDI) agent model (Rao and Georgeff, 1995), as well as teamwork oriented modelling. In the teamwork vision relating to the STEAM architecture, a non-negligible advantage is the reusability and flexibility of the operators (Tambe and Zhang, 2000), which are not directly related to the domain level. Thus, using JACK's representation of an agent's, we managed to develop collaborative driving teams based on the STEAM framework. In this section, we present the behaviors of these agents by comparing different coordination models and focusing on the centralized and teamwork models. First, the simulation scenarios' metrics are described, followed by the presentation of graphical results and an analysis of our coordination models.

6.1. Metrics

The evaluation model used to analyze the results of the teamwork and centralized coordination models is based on platoon splitting and merging scenarios. Each scenario has been simulated in the HESTIA simulator, on a straight road, with the same dynamics, sensing and communication models. For the test scenarios presented in this paper, the platoon's cruising velocity is 20 m/s. The time headway maintained by platoon members when they follow each others is a gap in time of 0.2 s, while the time headway created to allow a lane change (in front and behind the lane changing vehicle) is 0.5 s. The time distance of 0.2 s is referenced as what may be called a "safe time headway" when driving inside a platoon, while 0.5 s is a "safe time headway" when creating a gap between a vehicle that changes lane.

To compare the behavior of our different platoon coordination models, we use a metric based on a "headway error" referenced as Δh. This factor is the difference between a vehicle's time headway h and the "safe time headway" h^*, i.e., $\Delta h = h - h^*$. Using this metric, we can analyze the fluctuations of the absolute value of Δh and determine that a vehicle shows the wanted behavior if Δh is low (close to 0) or the unwanted behavior if Δh is high. If Δh is lower than -0.1 s, our vehicle does not show a safe behavior. On the other hand, if Δh is higher than 0.1 s, our vehicle does not optimize the inter-vehicle distances, which should lower the overall traffic density. Note that in order to make a fair analysis, the "headway error" Δh will be analyzed for the splitting and merging vehicles, from the time they change lane to the time the manoeuvre ends.

The simulation scenarios in which we use this metric are noisy merge and split scenarios. In both the merge and split manoeuvres, the noise is added by modifying the leader's velocity about 2 s before the splitter or merger changes lane to leave or enter the platoon. This velocity change creates a wave in the platoon, since all the followers react by modifying their respective velocities to keep the same time headway with the preceding vehicle. Therefore, the analysis of the Δh value in a noisy scenario allows us to compare the ability of our coordination models to handle uncertain events.

6.2. Simulation results

Simulation scenarios involving a vehicle merging and splitting a platoon have been executed to analyze the behavior of the merging and splitting vehicles with the "headway error" metric under different coordination models. The results we present here only relate to the centralized and teamwork coordination models, since these two models are very different one from another and they represent the comparison of a multiagent systems versus the standard centralized system. Moreover, the current implementation of our decentralized model does not fully support our noisy simulation scenarios and the hard-centralized model presents a behavior which is fairly the same as the centralized model when comparing it with the "headway error" metric. Therefore, the two latter coordination models are only analyzed in Section 6.3.

Fig. 7 first presents the comparison of a normal merge scenario, with noisy merge scenarios based on both the centralized and teamwork models. This figure shows the three major different results on the "headway error" value for vehicle *L2* (refer to Fig. 2(B)) during its merge. Fig. 7's scenario presents the following characteristics:

(1) At time 24 s: A platoon formed of four vehicles is driving in lane 1. This platoon is in the state represented as *S2* in Fig. 2(B): the platoon is stable and a merge gap has been created to let a vehicle (*L2*) merge this platoon.
(2) At time 24 s: vehicle *L2* is driving in lane 2 and it adjusts its velocity to become parallel to the gap created in the platoon led by *L1*.
(3) From time 32 s to time 37 s: *L1* gradually decelerates of -0.5 m/s^2 (noise).
(4) From time 37 s to time 42 s: *L1* gradually accelerates of 0.5 m/s^2 (noise).
(5) At time 36 s: *L2* is parallel to the merge gap and changes from lane 2 to lane 1.
(6) From time 24 s to time 44 s: $h^* = 0.5$ s.
(7) From time 44 s to time 50 s: $h^* = 0.2$ s.
(8) At time 50 s: the platoon lead by *L1* is in state *S3* of Fig. 2(B).

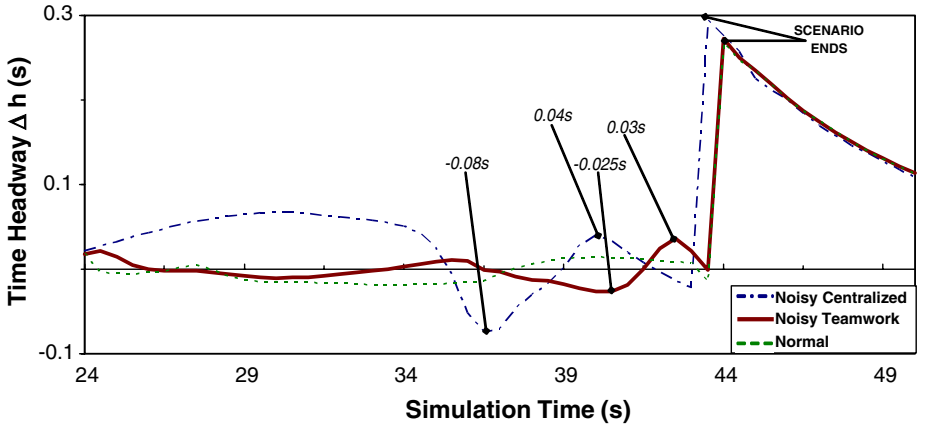

Fig. 7. Difference with the time headway and the safe headway of the merging vehicle, in three merge scenarios using different coordination models.

During the normal merge scenario (without any noise) referred as 'Normal' in Fig. 7, vehicle *L2*'s Δh reaches -0.015 s and 0.015 s, which is the behavior we requested. On the other hand, a noisy merge scenario makes it more difficult for vehicle *L2* to keep a safe time headway during the lane change occurring around time 36 s. By comparing the peak values of Δh in the interval of time 36 s to 44 s, we show that the "headway error" on the teamwork model is less than half the value of Δh with the centralized model in the same situation: -0.08 s and 0.04 s for the centralized model; -0.025 s and 0.03 s for the teamwork model. Another interesting difference between these two coordination models is the time they require to execute the noisy merge scenario. Indeed, the centralized model manages to execute the merge manoeuvre faster than the teamwork model, since the teamwork model required more time (1.25 s more) at the very beginning of the merge manoeuvre to form the merge team. On the other hand, the teamwork model does not require as much time (0.7 s less) to stabilize the platoon after the entrance of vehicle *L2* in the platoon. Therefore, the teamwork model finishes the merge manoeuvre only 0.5 s later than the centralized model.

In the scenario of a vehicle splitting from a platoon, similar conclusions can be drawn. In Fig. 8, the results of vehicle *F2*'s (the splitting vehicle of Fig. 2(A)) "headway error" are compared when using a scenario without noise (normal), the teamwork coordination model with noise and the centralized model with noise. Fig. 8's scenario presents the following characteristics:

(1) At time 15 s: a platoon formed of five vehicles is driving in lane 1. This platoon is in the state represented as *S1* in Fig. 2(A): the platoon is stable and a merge gap has been created to let *F2* split from this platoon.
(2) At time 15 s: *F2* and *F3* are decelerating to create a gap for *F2* to change lane (split).
(3) From time 21 s to time 26 s: *L1* gradually decelerates of -0.5 m/s^2 (noise).
(4) From time 26 s to time 31 s: *L1* gradually accelerates of 0.5 m/s^2 (noise).
(5) At time 21 s: *F2* has met the safe headway ($h = h^*$), so it changes from lane 1 to lane 2.
(6) From time 15 s to time 30 s: $h^* = 0.5$ s.
(7) At time 30 s: *F2* and the platoon led by *L1* should be in state *S2* of Fig. 2(A).

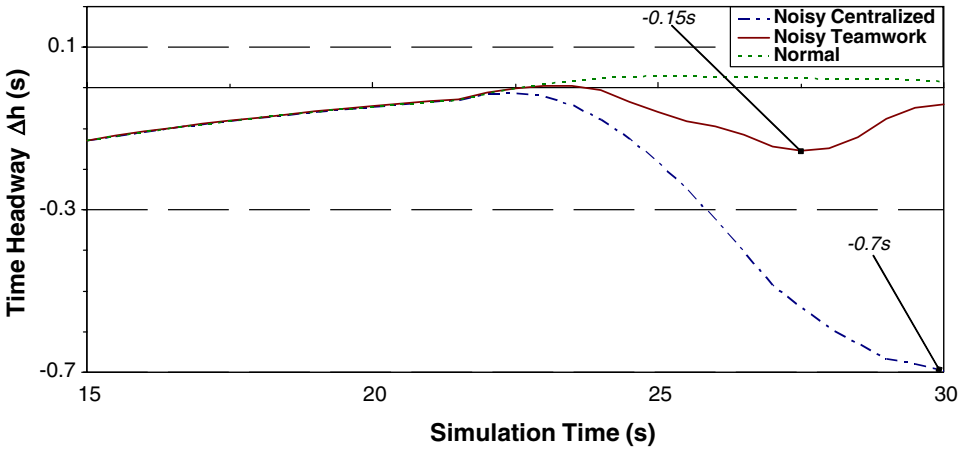

Fig. 8. Difference with the time headway and the safe headway of the splitting vehicle, in three split scenarios using different coordination models.

The results of the split scenario of Fig. 8 differ greatly when using the centralized and teamwork coordinations. When the lane change is executed, it should be recalled that the splitting vehicle must maintain a safe headway with its preceding vehicle in the platoon it just left, until the splitting vehicle completely and safely reaches the other lane. Thus, after *F2* has changed lane (following time 21 s), Δh should be kept to a value close to 0 s, as it is the case for the "Normal" curve. Using the teamwork model, Δh reaches a value of -0.15 s, because of the noise created by *L1*'s deceleration. However, with the centralized model, the value of Δh reaches a much higher value: -0.7 s. Thus, instead of being parallel to the gap created in *L1*'s platoon, *F2* is parallel to *F3*, which is dangerous since the split manoeuvre has not ended (vehicle *F2*'s lateral controller is still unstable).

In a different analysis, we decided to show two possible "turn of events" for the centralized coordination model used in a noisy merge scenario. In the previous noisy merge scenario, the centralized model managed to realize the merge manoeuvre in a time comparable to the teamwork model, but the results were not as safe. In contrast, if the noise created by the leader's deceleration appears a slight second before the previous scenario's noise, the leader can conclude that this deceleration is dangerous before commanding the merger to change lane. In this case, referred as the "Centralized-Slow" scenario, the leader waits to gain stability before commanding the merger to change lane. In Fig. 9, we compare the results of a fast, but unsafe centralized merge with a slow, but safe centralized merge. This scenario has the same characteristics as Fig. 8's scenario, expect for the following changes:

(1) For the "Centralized-Fast" scenario, from time 32 s to time 37 s: *L1* gradually decelerates of -0.5 m/s^2 (noise).
(2) For the "Centralized-Slow" scenario, from time 31 s to time 36 s: *L1* gradually decelerates of -0.5 m/s^2 (noise).
(3) For the "Centralized-Fast" scenario, from time 37 s to time 42 s: *L1* gradually accelerates of 0.5 m/s^2 (noise).

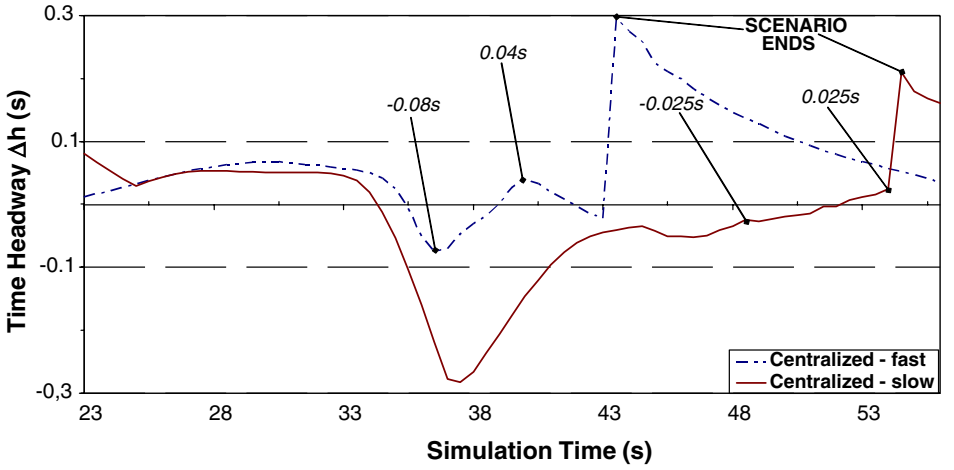

Fig. 9. Difference with the time headway and the safe headway of the merging vehicle, in two merge scenarios using the centralized model.

(4) For the "Centralized-Slow" scenario, from time 36 s to time 41 s: *L1* gradually accelerates of 0.5 m/s^2 (noise).
(5) For the "Centralized-Fast" scenario, at time 36 s: *L2* changes from lane 2 to lane 1.
(6) For the "Centralized-Slow" scenario, at time 48 s: *L2* changes from lane 2 to lane 1.
(7) For the "Centralized-Fast" scenario, from time 24 s to time 44 s: $h^* = 0.5$ s.
(8) For the "Centralized-Slow" scenario, from time 24 s to time 54 s: $h^* = 0.5$ s.
(9) For the "Centralized-Fast" scenario, from time 44 s to time 57 s: $h^* = 0.2$ s.
(10) For the "Centralized-Slow" scenario, from time 54 s to time 57 s: $h^* = 0.2$ s.
(11) For both scenarios, at time 57 s: the platoon led by *L1* is in state *S3* of Fig. 2(B).

In the two merge scenarios of Fig. 9, the merging vehicles (*L2*) must keep a value of Δh close to zero after they started changing lane. For the "Centralized-Slow" scenario, the value of Δh only reaches -0.025 s and 0.025 s from time 48 s to time 54 s, while the "Centralized-Fast" scenario presents values of Δh that reach -0.08 s and 0.04 s from time 36 s to time 44 s. However, the "Centralized-Slow" scenario ends at time 44 s, while the "Centralized-Fast" scenario ends at time 54 s. Note that the Δh value of -0.28 s reached by the "Centralized-Slow" scenario at time 37.5 s is not critical since at that moment, *L2* has not begun the lane change.

The safest results in a noisy merge scenario can be achieved by using the teamwork model with more communications from the *Virtual Vehicle*. This fact is shown in Fig. 10, which presents the different value of Δh kept by vehicle *L2* in the teamwork model. This figure compares a scenario where the *virtual vehicle* communicates three messages ("More Messages") about its position and velocity, with a *Virtual Vehicle* communicating only one message ("Less Messages"). This scenario has the same characteristics as Fig. 8's scenario, expect for the following changes:

(1) For the "More Messages" scenario, at time 36 s: *L2* changes from lane 2 to lane 1.
(2) For the "Less Messages" scenario, at time 39 s: *L2* changes from lane 2 to lane 1.
(3) For the "More Messages" scenario, from time 24 s to time 44 s: $h^* = 0.5$ s.

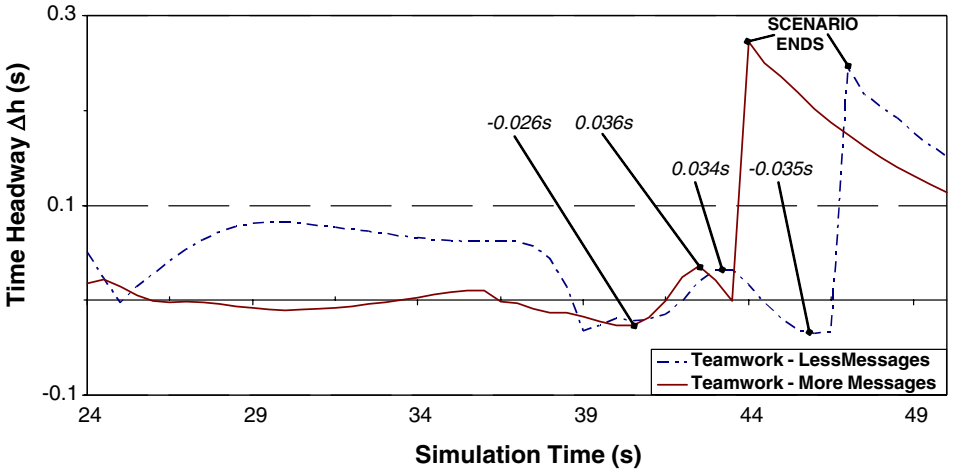

Fig. 10. Difference with the time headway and the safe headway of the merging vehicle, in two merge scenarios using the teamwork model.

(4) For the "Less Messages" scenario, from time 24 s to time 47 s: $h^* = 0.5$ s.
(5) For the "More Messages" scenario, from time 44 s to time 50 s: $h^* = 0.2$ s.
(6) For the "Less Messages" scenario, from time 47 s to time 50 s: $h^* = 0.2$ s.
(7) For both scenarios, at time 57 s: the platoon led by *L1* is in state *S3* of Fig. 2(B).

Fig. 10 shows that a merge scenario coordinated with a teamwork model using more messages results in a Δh reaching -0.026 and 0.036 from time 39 s to 44 s, while using less messages results in a Δh reaching -0.035 and 0.034 from time 39 s to 47 s. This difference is not very significative, but the most important difference between these two scenarios is the time they require to execute the merge manoeuvre: the "Less Messages" scenario requires three more seconds (executed at time 47 s versus 44 s).

By summarizing the previous results, our test scenarios showed that the teamwork coordination model was more appropriate when the platoon becomes unstable or when uncertain events arise. Apart from being safer than the centralized model, the teamwork model diminishes platoon instability when noises appear ahead, which helps enhancing the highway's overall traffic capacity.

6.3. Results analysis

In order to complete the previous results on our coordination models and give a better overview of each model's advantages and disadvantages, Table 1 presents the amount of messages and plans required by each model. In this table, "average number of messages" represents the average number of messages, of any size, that have been broadcasted to one or many vehicles during a specific manoeuvre. "Messages size" is the average total amount of bits that have been transmitted through messages during the manoeuvre. Note that these two average values have been calculated using scenarios with and without noise (noise similar to scenarios presented in Fig. 7).

Table 1
Average messages and total plans used by each coordination model

Coordination model	Average number of messages		Messages size (bits)		JACK plans	
	Merge	Split	Merge	Split	Merge	Split
Centralized	9	6	506	112	18	14
Hard-centralized	7	–	486	–	16	–
Decentralized	10	7	674	290	20	13
Teamwork	11.25	8.25	882	454	10	8

Finally, "JACK plans" represents the amount of plans in JACK Intelligent Agents™language,[3] which were required to support each coordination model, showing the flexibility of their respective framework. These JACK plans refer to the coordination plans that made it possible to support a specific manoeuvre, with a specific communication protocol.

By using the results presented in Table 1, along with the results presented in Section 6.2, each coordination model were analyzed and their respective advantage and disadvantage have been summarized.

6.3.1. Hard-centralized model analysis

Advantage: this model exchanges the lowest amount of messages for the merge manoeuvre since it forces the merging vehicle to insert itself at the end of the platoon (refer to Table 1). This model sometimes has a faster execution time than the centralized model, since it does not wait for the platoon to create a merging gap before changing lane.

Disadvantage: depending on its position and the current traffic on the highway, this model may require more time for the merging vehicle to place itself at the right position. Moreover, further test scenarios including more platoons should prove that a merge manoeuvre, coordinated with the hard-centralized model, will more likely create traffic waves that will disturb the overall highway traffic.

6.3.2. Centralized model analysis

Advantage: this model requires a low amount of messages of small sizes to coordinate the split and merge tasks, as shown in Table 1. By using a single vehicle to coordinate less autonomous vehicles (followers), the centralized model is less propitious to uncertain or conflicting actions that may be executed by the followers in other models as the decentralized and teamwork models. Finally, the centralized model presents the best execution time when the platoon is maintained stable (without noise) and therefore it is the most efficient coordination model if we make the assumption that our platoons will always maintain their stability.

Disadvantage: the lower amount of messages sent in a centralized model is an advantage that results in a disadvantage mentioned in Section 6.2. During our "noisy" merge and split scenarios, the merging and splitting vehicles had more problems keeping safe headways when they changed lane. This resulted in dangerous situations or in a manoeuvre that required a lot more time to be

[3] In JACK, Plans are pre-compiled procedures based on the PRS architecture (Georgeff and Ingrand, 1990). A plan can only be executed when it answers to internal or external events using *relevance* and belief *context* criterions.

executed, as shown in Fig. 9. Finally, a non-negligible disadvantage of the centralized model is the fact that in average, more than three quarters of the messages were sent or received by the leader, creating a bottleneck for this vehicle.

6.3.3. Decentralized model analysis

Advantage: this model has more flexibility than the centralized model since the leader and its followers have a similar degree of autonomy. The decentralized model also has the advantage of involving only two vehicles, while the rest of the platoon only has to update its knowledge at the beginning and end of a manoeuvre.
Disadvantage: being based on the theory of social laws, the decentralized coordination model is not easily expandable since it requires more plans, as shown in Table 1. As we mentioned it for the centralized model, the decentralized model is not as safe as the teamwork model, which can update the platoon state through the *Virtual Vehicle*. Finally, this model needs to communicate to initialize and maintain common knowledge within the platoon, thus it requires a higher amount of messages than the centralized model.

6.3.4. Teamwork model analysis

Advantage: by using this coordination model, more vehicles are involved in a manoeuvre, but this has the advantage of dividing equally the communication load involved in the coordination. As mentioned previously, and as we explained it in Section 6.2, the teamwork model can more easily handle uncertain situations or other types of instability in the platoon. The teamwork model also has the advantage of being expendable and because of the generic and reusable framework it is based on, it requires less plans than other models (refer to Table 1), which makes it more flexible. Finally, the teamwork coordination model will be more efficient and safe if we insert our automated vehicles in mixed traffic (including non-automated vehicles), where the centralized and decentralized models will react to uncertain events at the cost of poor time efficiency and safety.
Disadvantage: the most important disadvantage of the teamwork may be the amount and size of the messages it requires. In order to assign roles and maintain a common belief inside the teams, vehicles in the teamwork model exchange more messages than the centralized model, as shown in Table 1.

7. Conclusion and future work

Collaborative driving is emerging in the domain of ITS and it will ultimately be part of the every day vehicle's automation system, since it is the next step, following the adaptive cruise control (ACC). A complete collaborative driving system (CDS) used with platoons of vehicles can be seen as a long-term goal, but communication infrastructures are already being incorporated to highways, and a CDS would present the most efficient use for this type of infrastructure. CDS can therefore be easily included in the development plans of ITS for the upcoming years, as it will evolve until vehicle-level technologies like ACC meet AHS infrastructures technologies and increase ITS benefits on safety, efficiency and environment.

In this paper, we presented a hierarchical driving architecture that can be coupled with different platoon coordination models in the context of collaborative driving. To determine the most propitious coordination model, we investigated four different inter-vehicle communication models and showed the simulation results of two contrasting approaches: a centralized model based on the coordination of the platoon's leader and a decentralized model based on the coordination of the platoon as a team of agents. We then analyzed our coordination models and concluded on the possible choice of a model for our future works. The hard-centralized and centralized models present a good option if the most important issue is to minimize communications and lower the autonomy of the followers, in order to prevent " unwanted" behaviors. The decentralized and teamwork models, on the other hand, are more flexible since they are not linked to the platoon leader and could be used to form groups of CACC equipped vehicles. The autonomy given to followers, particularly in the teamwork model, often results in increasing safety, since each agent can make its own choices considering local and shared beliefs. If we consider that the traffic environment in which we want to incorporate our CDS is unpredictable and makes it difficult to maintain the platoon's stability, then the teamwork model is probably the best coordination model for our automated vehicles.

Considering the great results on safety issues we received from the teamwork model and its ability to be easily expandable, our future works should consider the extension of this model. However, we should also continue the improvement of our longitudinal guidance system, which could help improving the overall results of our CDS. We should finally investigate the possibility of extending the teamwork model with role-based multiagent team decision problem (RMTDP) role re-allocation strategies (Nair et al., 2003) and reinforcement learning applied to the decisions on communication.

Acknowledgement

This work has been carried out as part of the Automobile of the 21st Century (Auto21) project supported by the Government of Canada through the Networks of Centres of Excellence (NCE).

References

Antoniotti, M., Deshpande, A., Girault, A. Nov. 1997. Microsimulation analysis of multiple merge junctions under autonomous ahs operation. In: Proceedings of the IEEE Conference on Intelligent Transportation System (ITSC) 97, pp. 147–152.

AOS, 2004. JACK Intelligent Agents™4.1. Software Agents Development Framework.

Auto'21, 2003. Architectures for collaborative driving vehicles: From a review to a proposal. Tech. Report, Université Laval, Ste-Foy, Québec.

Auto21, April 2004. Online available from: <http://www.auto21.ca/> (accessed the 30th of April 2004).

Bana, S.V., September 2001. Coordinating automated vehicles via communication. Ucb-its-prr-2001-20, University of California, Berkeley.

Blosseville, J.M., Hoc, J.M., Riat, J.C., Wautier, D., d. 1. Bourdonnaye, A., Artur, R., Tourni, E., Narduzzi, C., Gerbenne, E., 2003. French contribution to the functional analysis of 4 key active safety functions (arcos project). In: Proceedings of the 10th ITS World Congress. Madrid, Spain.

DAMAS-Auto21, April 2004. Online accessed the 30th of April 2004.

Georgeff, M., Ingrand, F., May 1990. Real-time reasoning: the monitoring and control of spacecraft systems. In: Press, C.S. (Ed.), Proceedings of the Sixth IEEE Conference on Artificial Intelligence Applications (CAIA 90). vol. 1. Los Alamitos, CA, pp. 198–205.

Ghosh, S., Lee, T., 2000. Intelligent Transportation Systems: New Principles and Architectures. CRC Press, London.

Hallé, S., January 2005. Automated highway systems: platoons of vehicles viewed as a multiagent system. Master's thesis, Université Laval, Québec, Canada.

Hallé, S., Chaib-draa, B., Laumonier, J., April 2003. Car platoons simulated as a multiagent system. In: Muller, J.-P., Seidel, M.-M. (Eds.), Proceedings of Agent Based Simulation, vol. 4 (ABS4). pp. 57–63.

Hedrick, J., Sengupta, R., Xu, Q., Kang, Y., Lee, C., 2003. Enhanced ahs safety through the integration of vehicle control and communication. California PATH Research Report UCB-ITS-PRR-2003-27, University of California, Berkeley.

Howell, A.S., Girard, A.R., Hedrick, J.K., Varaiya, P.P., January 2004. A real-time hierarchical software architecture for coordinated vehicle control. In: Proceedings of Automotive Software Architecture Workshop 2004. San Diego, CA.

Huppe, X., de Lafontaine, J., Beauregard, M., Michaud, F., 2003. Guidance and control of a platoon of vehicles adapted to changing environment conditions. In: Proceedings of IEEE International Conference on Systems Man and Cybernetics, 2003. vol. 4. pp. 3091–3096.

Ioannou, P., Stefanovic, M., 2003. Evaluation of the ACC vehicles in mixed traffic: lane change effects and sensitivity analysis. PATH Research Report UCB-ITS-PRR-2003-03, University of Southern California.

Liang, C.-Y., Peng, H., 2000. String stability analysis of adaptive cruise controlled vehicles. JSME International Journal Series C 43 (3), 671–677.

Lygeros, J., Godbole, D.N., Sastry, S.S., 1998. Verified hybrid controllers for automated vehicles. IEEE Transactions on Automatic Control 43, 522–539.

Nair, R., Tambe, M., Marsella, S., 2003. Role allocation and reallocation in multiagent teams: towards a practical analysis. In: Proceedings of the second International Joint Conference on Agents and Multiagent Systems (AAMAS). pp. 552–559.

Rao, A.S., Georgeff, M.P., 1995. BDI agents: from theory to practice. Proceedings of the First International Conference on Multiagent Systems. The MIT Press, San Francisco, pp. 312–319.

Sakaguchi, T., Uno, A., Kato, S., Tsugawa, S., October 2000. Cooperative driving of automated vehicles with inter-vehicle communications. In: Proceedings of the IEEE Intelligent Vehicles Symposium. Dearbsorn, USA, pp. 516–521.

Shoham, Y., Tennenholtz, M., 1995. On social laws for artificial agent societies: off-line design. Artificial Intelligence 73 (1–2), 231–252.

Tambe, M., Zhang, W., 2000. Towards flexible teamwork in persistent teams: extended report. Journal of Autonomous Agents and Multi-agent Systems 98 (3), 159–183, Special Issue on Best of ICMAS.

Tsugawa, S., Kato, S., Matsui, T., Naganawa, H., October 2000. An architecture for cooperative driving of automated vehicles. In: Proceedings of the IEEE Intelligent Vehicles Symposium 2000. Dearbsorn, MI, USA, pp. 422–427.

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

VanderWerf, J., Kourjanskaia, N., Shladover, S., Krishnan, H., Miller., M., January 2001. Modeling the effects of driver control assistance systems on traffic. Technical Report Paper No. 01-3475, US National Research Council Transportation Research Board 80th Annual Meeting, Washington, DC.

Varaiya, P., 1993. Smart cars on smart roads: problems of control. IEEE Transactions on Automatic Control, 32.

Xu, Q., Hedrick, K., Sengupta, R., VanderWerf, J., September 2002. Effects of vehicle–vehicle/roadside–vehicle communication on adaptive cruise controlled highway systems. In: Proceedings of IEEE Vehicular Technology Conference (VTC). Vancouver, Canada, pp. 1249–1253.