A Dynamic Geocast Solution to Support Cooperative Adaptive Cruise Control (CACC) Merging

Wouter Klein Wolterink, Georgios Karagiannis, Geert Heijenk
Department of Computer Science, University of Twente, The Netherlands
{w.kleinwolterink, g.karagiannis, geert.heijenk}@utwente.nl

Abstract. Cooperative Adaptive Cruise Control (CACC) is a type of cruise control in which the speed of a vehicle is controlled based on wireless communication between vehicles. In this paper we tackle the communication needed in case of fully automatic CACC merging at a junction. The first contribution of our paper is to show that to target the vehicles involved we need a special kind of geocast that takes both the geographical location and the dynamics (speed, acceleration) of a vehicle into account. The second contribution is to give a first approach to such a geocast solution. The resulting geocast protocol is able to target multiple destination sets that are geographically dispersed and that are persistent in time. This paper does not yet include an analysis of the protocol, but analyses by means of simulation and real-world testing have already been planned.

Keywords: VANET, CACC, V2V, geocast, merging

1 Introduction: the Problem of CACC Merging

With Cooperative Adaptive Cruise Control (CACC) the longitudinal speed of a vehicle is automatically controlled based on the behaviour of vehicles up to a certain distance ahead of the vehicle. Information about this behaviour is obtained by having each CACC vehicle regularly (≥ 5 Hz) broadcast a so-called beacon: a small message containing all information (including the position, speed, and acceleration of a vehicle) needed for CACC to operate. The goal of CACC is to anticipate earlier and more accurate to traffic disturbances than a human driver can, and thus improve overall traffic efficiency by dampening out any traffic disturbances.

Within the Connect & Drive project we are currently working on automatic CACC merging at a freeway, where the merging vehicle is a “normal” vehicle (without CACC) and the freeway vehicles are a mix of CACC vehicles and normal vehicles. We focus on gap creation: the CACC freeway vehicles must be told in advance to create a gap for the merging vehicle. We use a so-called road side unit (RSU) to detect oncoming merging vehicles and to tell the freeway vehicles to make a gap. Such a gap is created by having a CACC vehicle decrease its speed, so that its

---

1 This work is supported by the Dutch NL Agency/HTAS project Connect & Drive, Project no. HTASD08002
headway will increase. The goal of our project is to minimize the resulting speed disturbances caused by this deceleration.

Fig. 1 illustrates the problem: two merging vehicles, $M_0$ and $M_1$, are approaching the merge area ($a_1$ to $a_2$) where they both want to join the freeway traffic. Vehicle $D_1$ has created a gap for $M_0$, while one of the vehicles of the platoon $D_3$-$D_7$ must create a gap for $M_1$. We define the moments in time when a merging vehicle reaches $a_1$ and $a_2$ as $t_{a,1}$ and $t_{a,2}$. The destination sets drawn in the picture can be ignored so far, they are discussed in Section 3.

![Fig. 1. Merging vehicles wish to join the downstream freeway vehicles. The arrows denote the direction of travel. The dotted lines represent the borders of geocast destination sets. The subnetworks of downstream travelling vehicles are $D_0$, $D_1$, $D_2$, and $D_3$-$D_7$.](image)

Informing the right CACC vehicles (henceforth also referred to as nodes) in advance to their arrival at the merge area involves quite a number of problems, two of which are relevant here: (i) how can be decided in advance which vehicles are responsible for creating a gap, (ii) how should these vehicles be informed of this responsibility.

The first goal of this paper is to show that to inform these responsible nodes, we need a novel geocast approach that is able to target vehicles based on their location and their vehicle dynamics. To do this, we first introduce a simple approach to defining which vehicles should be informed: instead of explicitly defining which vehicles are responsible and targeting only them, we choose to inform every freeway vehicle that may be responsible, and defer the question of who should actually create the gap till later.

The second goal of our paper is to give a first approach to a geocast solution able to target multiple, geographically dispersed destination sets along a straight road, which are defined by the nodes’ location and dynamics. These destination sets are persistent in time, similar to abiding geocast, but also dynamic in size: as time progresses the destination sets shrink, and nodes automatically fall out of them.

Several research activities have focused on how vehicles could use communication to realize merging manoeuvres, see e.g., [4] and [5]. However, their goal was to optimize the merging procedure from the point of the merger’s benefits. Our approach focuses on the realization of a merging manoeuvre where the disturbances on the highway are minimized.

Geocast is a routing paradigm that supports the dissemination of information to a certain geographical area, rather than to a certain address. Data is typically only sent once. In abiding geocast a geographical area is targeted for a specified duration: every node that enters the destination region during a certain period is part of the destination set. For more work on (abiding) geocast see [1], [2], and [3]. The authors of this paper could not find any geocasting solution developed to support merging manoeuvres.
Transportation systems that utilize ICT are called Intelligent Transport Systems (ITS) and are receiving quite some research attention. A good general resource is [6].

The outline of this paper is as follows. In Section 2 the set of vehicles that may be responsible is defined. In Section 3 a design is presented that is able to target these vehicles in the context of our merging application. Section 4 concludes the paper by giving a summary of the presented work and by presenting our future plans extending this work.

2 Determining the Destination Set

In this section we give a definition of the set of vehicles that may be responsible for creating a gap. It is this set that we want to target with our dynamic geocast protocol presented in the next section. We show that the destination set depends on a vehicle’s geographical location and its vehicle dynamics, and that its size is dynamic in time. Note that the definition given here is conceptual – its implementation is shown in the next section.

We define that every vehicle that may fall within a certain margin ($\mu$) of the merging vehicle’s estimated position during the period $[t_{a1}, t_{a2}]$ may be responsible, and is therefore part of the destination set. The margin is not of constant size: it is a function of the (un)certainty about the merger’s position during that period. As the merging vehicle approaches the merge area uncertainty should normally lessen, as a result of which the margin should decrease in size.

Estimating a vehicle’s future position is fraught with uncertainty – especially when the driver is human. For the freeway vehicles we use the formula for linear motion with constant acceleration, see Eq. (1). Here the position at a future time $t$, $s(t)$, is calculated using the position ($s_i$) and speed ($v_i$) at $t_i$ and a constant acceleration $a$. Let $t$ range over the period $[t_{a1}, t_{a2}]$, and $a$ between the maximum deceleration and acceleration values of the CACC control algorithm. We can then theoretically calculate all future positions of the vehicle during $[t_{a1}, t_{a2}]$, and whether or not the vehicle may fall inside the merger’s margin (or $\mu$).

We do not specify how the time of arrival of the merging vehicle (i.e., $t_{a1}$ and $t_{a2}$) should be estimated but this could be done in the same way as with the freeway vehicles.

$$s(t) = s_i + v_i \cdot (t - t_i) + \frac{1}{2} \cdot a \cdot (t - t_i)^2.$$

(1)

Fig. 2 shows the estimated trajectories of five of the freeway vehicles in Fig. 1., whose estimated distance to the merge area are shown for the period $[t_{0}, t_{a2}]$ for different acceleration values: maximal deceleration ($a_{\text{min}}$), no acceleration ($a_0$), and maximum acceleration ($a_{\text{max}}$). The solid lines indicate the estimated trajectories of the vehicles when they were to start accelerating at $t_0$. The dotted lines show the actual trajectories between $t_0$ and $t_1$ (assuming that the vehicles move at constant speed), and beyond $t_1$ the estimated trajectories of the vehicles if they were to start accelerating at $t_1$. The width of the margin at $t_0$ is $\mu_0$, and at $t_1$ its width is $\mu_1$. The parallelograms
show the margins $\mu_0$ and $\mu_1$ for the period $[t_{a,1}, t_{a,2}]$. Note that as the merger nears the merge area $\mu$ will continually decrease.

It is important to note that as vehicles get closer to the destination area the destination set will decrease because (i) $\mu$ continuously decreases, and (ii) because even for a static $\mu$ fewer vehicles will be able to fall inside it (since the effect of accelerating/decelerating hard becomes less). A vehicle must therefore not only be informed when it has entered a destination set, but also when it has left it. Fig. 2 shows how for a static $\mu$ the destination set becomes smaller as vehicles get closer to the merge area: at $t_0$ (solid lines) vehicles $D_4$, $D_5$, and $D_6$ can all fall inside the wider $t_0$ margin, while at $t_1$ (dotted lines) only $D_5$ can still fall inside it. We can also see how the destination set becomes smaller as the margin decreases: at $t_0$ vehicles $D_4$, $D_5$, and $D_6$ can all fall inside the wider margin, but $D_5$ is the only vehicle that can fall inside the smaller margin.

Fig. 2. The trajectories of five freeway vehicles of Fig. 1 through time and space. The merger’s trajectory has also been projected on the figure as a line between the points $(t_{a,1}, a_1)$ and $(t_{a,2}, a_2)$. The shaded parallelograms represent the margin $\mu$ around the merging vehicle during $[t_{a,1}, t_{a,2}]$, calculated at $t_0$ and $t_1$. The dotted trajectory for $D_5$ is hidden beneath the solid trajectory.

3 Design of a Dynamic Geocast Approach for CACC Merging

This section presents a design for a dynamic geocast approach that is able to disseminate messages to multiple, geographically dispersed destination sets at once (assuming a straight road). The destination set defined in the previous section is used. The resulting protocol can be seen as an abiding geocast approach, in which location and time are mutually dependent: the location moves as time progresses. An important difference with abiding geocast is that the location also depends on a node’s velocity, and is therefore different per node. The destination sets are furthermore regularly updated by specific update messages.

For ease of reading we refer in this section to CACC vehicles that travel downstream as class 1 nodes, and to CACC vehicles that travel in another direction as class 2 nodes. A set of class 1 nodes that can all reach each other via only class 1 nodes is called a class 1 subnetwork. Fig. 1 shows three such networks: $D_0$, $D_1D_2$, and $D_3D_4$. Nodes that are within each other’s transmission range are called neighbours.
Due to the information contained in the CACC beacons nodes know the location and the direction of travel of all their neighbours (as well as their own of course).

**System Operation.** As the merger approaches the merge area the RSU regularly estimates $t_{a,1}$ and $t_{a,2}$. The first estimation must be made well enough in advance to be sure that there are freeway vehicles that have enough time to create a gap. How far in advance this should be is still subject of research. After each estimation the RSU creates a *merge request*: a message containing $a_1$, $a_2$, $t_{a,1}$, $t_{a,2}$, and $\mu$. The merge request is then disseminated to its own destination set (defined further below) and to the destination set of the previous merge request for the same merger (if applicable). The latter is necessary to inform vehicles that were part of the previous destination set but not of the current one that they have fallen out. A vehicle that receives a merge request and that is part of the destination set will inform the CACC software layer of the merge request. Because a vehicle may fall out of the destination set as it approaches the merge area it has to regularly check whether it is still part of the destination set. If not then it should again inform the CACC software layer.

**Joining of Merge Requests.** When there are multiple merging vehicles at the same time, the RSU will combine all merge requests into a single *merge message*. Each merge message must therefore be disseminated to multiple destination sets, which may or may not overlap. We thus want to target multiple destination areas that may be geographically dispersed, see for example Fig. 1 where the destination sets for $M_0$ and $M_1$ are shown.

**The Destination Set.** A class 1 node considers itself part of the destination set of a merge request if any of the following statements hold:

1. it may itself fall inside $\mu$ (by using Eq. (1) with $a$ ranging over the possible CACC values);
2. it has an upstream and downstream neighbour that may fall inside $\mu$;
3. it is estimated to always fall in front (i.e., further downstream) of $\mu$, but it has a downstream neighbour that may fall inside the margin;
4. it is estimated to always fall behind (i.e., further upstream) of the margin, but it has an upstream neighbour that may fall inside the margin.

Nodes check regularly whether they are still part of a destination set. A node that is not (or no longer) part of the destination set will discard the message after having forwarded it. A node that is part of the destination set will keep a received merge request as long as it has not passed $a_2$ (in space) or $t_{a,2}$ (in time), whichever comes first.

**Forwarding by the RSU.** The RSU forwards a merge message to every neighbour that is part of the destination set. If this set is empty it will forward it to its most upstream class 1 node. If it has no class 1 nodes at all it will act as if it is the most upstream node of a class 1 subnetwork (see below).

**Forwarding inside a class 1 subnetwork.** Forwarding between class 1 nodes is done reliably, either by means of unicast, by means of broadcast with implicit
acknowledgement, or by piggybacking on the CACC beacons (also with implicit acknowledgement).

A node that is part of the destination set and that has previously received a merge message will include either the entire merge message in its CACC beacons, or only the message’s identifier. If forwarding is done by means of piggybacking then the entire message is included, else only its identifier. A node will include the message (identifier) in its beacons for as long as the node is still part of the destination set and the message is still valid.

A node will forward a message to any new neighbour it might get (after the initial forwarding) that is also part of the destination set, but that has not yet included the message in its CACC beacons. It will not forward a message to a node that has already included that message in its beacon. Class 1 nodes discard duplicate messages; class 2 nodes do not.

A class 1 node that received a message from a downstream class 1 node will forward it in the upstream direction, and vice versa. If the message came from a class 2 node then it will forward the message in both directions.

A class 1 node will not forward in the upstream direction if all its upstream class 1 neighbours fall behind every destination set. Similarly it will not forward in the downstream direction if all its downstream class 1 neighbours fall in front of every destination set.

A node that considers itself part of the destination set will forward a message to its nearest class 1 neighbour in the direction it wants to forward. Outside the destination set forwarding between class 1 nodes is done in greedy fashion by forwarding a message to the class 1 neighbour that is located furthest away (in a certain direction).

**Forwarding between subnetworks.** Although the direction of forwarding inside subnetworks is arbitrary (i.e., upstream or downstream) we assume that by following the rules below the subnetworks themselves are targeted in an upstream matter.

If the most upstream node of a class 1 subnetwork is either part of or situated in front of at least one destination set, it has the responsibility to forward the message to the next upstream class 1 subnetwork. If possible this can be done using reliable multi-hop (geo-)routing (i.e., the original sender receives an acknowledgement if the intended receiver has received the message), see for instance Fig. 1 where node D₂ has a connected path to subnetwork D₅-D₇ using the upstream nodes U₂ and U₃. If this is not possible however the node should resort to store-carry-forward: it forwards the message to every class 2 node it encounters, for as long as the message is valid. The class 2 node forwards the message to the first upstream class 1 node it encounters (unless that node has included the message in its beacon), after which it will discard the message. A class 2 node will also discard the message when it has travelled so far upstream that even a class 1 vehicle travelling at maximum speed can no longer fall inside any of the margins. Forwarding between subnetworks is work in progress: we still have to research whether a combination of reliable multi-hop and store-carry-forward is viable.

Nodes decide for themselves whether they are the most upstream node of subnetwork, so responsibility for reaching the next subnetwork shifts automatically when necessary.
**Delimiting the destination set.** The forwarding protocol presented here aims to keep the destination area purposefully small, to increase the system’s bandwidth efficiency. As a result some vehicles may receive a merge request too late to be able to create a gap. Suppose for example that behind and outside the transmission range of $D_7$ (in Fig. 1) another class 1 vehicle is travelling, $D_8$. If this vehicle’s speed is high enough then it will also be part of the destination set, even if $D_7$ is not, but it will initially not receive the message. However, such a vehicle will be forced to either adapt its speed to the vehicles in front (that are not part of the destination set) so that it will no longer be part of the set anyway, or to change to the faster lane (when possible), in which case it should simply treat the merge request as a warning to do just so.

### 4 Conclusions & Future Work

We have shown that CACC merging on a freeway requires a geocast protocol capable of targeting a destination set whose make-up depends on the location and dynamics of both the merging vehicle and the freeway vehicles. A first (rather simple) approach to defining the destination set was given. Using this definition a geocast solution capable of targeting multiple such destination sets at once was discussed.

Within Connect & Drive we will first focus on defining which vehicles are responsible for creating a gap. Following that we will finalize our merging protocol and analyse it using e.g. simulations, and later on implement it and subject it to real-world testing in the first half of 2011. Parallel to our project-related work we plan to generalize the idea of a dynamic geocast protocol whose target set depends on a vehicle’s location and dynamics.

**Acknowledgements.** We thank the members of Connect & Drive for their close cooperation, especially Fei Liu and Rattaphol Pueboobpaphan of the Centre for Transport Studies of the University of Twente. We also thank NL Agency and HTAS for their financial support of the Connect & Drive project.

**References**