Automated Vehicle-to-Vehicle Collision Avoidance at Intersections

M. R. Hafner[1], D. Cunningham[2], L. Caminiti[2] and D. Del Vecchio[3]

1. Department of Electrical and Computer Engineering, University of Michigan
1301 Beal Ave., Ann Arbor, MI 48109
Phone: (805)698-2584, Fax: (734)763-8041, E-mail: mikehaf@umich.edu

2. Toyota Technical Center, Toyota Engineering and Manufacturing North America
1555 Woodridge Ave., Ann Arbor, MI 48105
Phone: (617)452-2275, Fax: (734)995-3053, E-mail: lorenzo.caminiti@tema.toyota.com

3. Department of Mechanical Engineering, Massachusetts Institute of Technology
77 Massachusetts Ave., Cambridge, MA 02139
Phone: (734)995-9330, Fax: (617)258-6156, E-mail: ddv@mit.edu

***Abstract*—We present experimental results for an active control Intersection Collision Avoidance (ICA) system implemented on modified Lexus IS250 test vehicles. The system utilizes vehicle-to-vehicle (V2V) Dedicated Short-Range Communications (DSRC) to share safety critical state information, allowing for distributed implementation of our provably safe algorithms. Safety is achieved in potential collision scenarios by controlling the velocities of both vehicles with automatic brake and throttle commands. Automatic commands can never cause the violation of predefined upper and lower speed limits.**

I. Introduction

In this paper, we present experimental results from an Intelligent Transportation System (ITS), which we call the Intersection Collision Avoidance (ICA) application, designed to prevent collisions between two vehicles passing through an intersection. According to data taken in 2009 by the Fatality Analysis Reporting System (FARS) and the General Estimates System (GES), about 40.1 percent of all crashes in the United States were intersection-related crashes [18]. A study conducted by the National Highway Traffic Safety Administration (NHTSA) on intersection-related crashes in 2010 [19] concluded that 44.1 percent of intersection-related crashes were most directly attributed to inadequate driver surveillance, compared to only 7.3 percent in non-intersection-related crashes.

Previous work concerning formal approaches to ITS involved the automated highway systems (AHS) by the California PATH project in the 90s. The objective of the AHS project was to develop fully autonomous highway systems, with the goal of decreasing congestion, increasing safety, and improving fuel efficiency [12, 13, 20, 23]. Most of this work pertaining to safety involved the development of vehicle platooning, where formal modeling and control solutions were employed based on the computation the largest unsafe set, drawing from techniques in optimal control and game theory [1–3, 10, 14, 16, 17, 22]. Recent work concerning the design of intelligent intersections revolves around provably safe scheduling algorithms for large numbers of vehicles [15].

The aim of ICA is to prevent collisions at intersections after normal traffic control mech-

The authors would like to thank Jeff Lovell, Jeff Rogers, Chris Peplin and Chris Hammond for their contribution to hardware and software development, and Keiji Shibata for help with experimentation. This work has been in part funded by NSF-GOALI Award CMMI-0854907.

anisms have failed. The ICA application could be employed in a real-world scenario where a driver turns right at a red light without properly (or incorrectly) surveying oncoming traffic. If the right turning vehicle enters the intersection before the oncoming vehicle, ICA will issue a throttle command to the vehicle turning right, while simultaneously issuing a brake command to the oncoming vehicle. These commands are issued in a manner that does not cause the violation of predefined speed limits (either traffic laws or comfort levels), more precisely, the braking command will not cause the vehicle velocity to become less than a minimum speed, while the accelerating command will not cause the vehicle velocity to become greater than a maximum speed. This implies that automatic control commands applied to prevent collisions at the intersection do not create hazardous driving conditions for other vehicles not directly involved.

Central to our approach is the assumption that, after making high level route decisions, drivers follow predefined paths when driving through intersections. These paths are defined by driving lanes, and drivers do not change lanes once they enter the intersection. This assumption allows us to model collisions between two vehicles as a set of vehicle configurations on these paths, as seen in Figure 1. Collisions between two vehicles are prevented by *only* controlling the longitudinal velocity and displacement of each vehicle along its path, never controlling vehicle steering. We assume each vehicle is equipped with sensors for state measurement (absolute position, heading, velocity, acceleration, brake torque, and pedal position), non-interrupted V2V communication, and the ability to automatically actuate the throttle and brake. Under the above assumptions, the safety algorithms implemented have been mathematically shown to guarantee the two vehicles passing through the intersection never collide [6–8, 11].

This paper is organized as follows. We describe the test vehicles including sensors, actuators and computational resources in Section II. In Section III, we provide the modeling approach taken on the two vehicles approaching the intersection. In Section IV, we introduce the formal collision avoidance problem, and provide the formal solution. In Section V, we describe the software implementation of our formal solution. In Section VI, the experimental setup is provided including the intersection configuration considered. Lastly, in Section VII we present our experimental results.

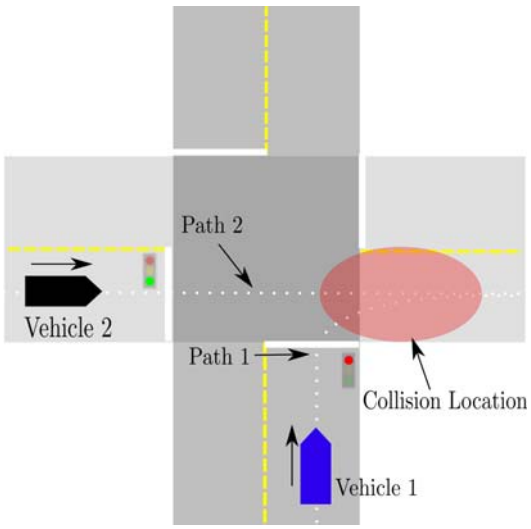


Fig. 1. Example of two adjacent vehicles approaching a four way intersection along predefined paths, the location of potential collisions is indicated around the intersection of the paths.

II. Test Vehicle

The test vehicles used in these experiments are modified Lexus IS250 (2007) test vehicles. The modifications include: (a) computer running a Linux operating system; (b) Differential Global Positioning System (DGPS) for position, absolute time and heading; (c) DENSO Wireless Safety Unit (WSU) capable of V2V and Vehicle-to-Infrastructure (V2I) Dedicated Short-Range Communications (DSRC); (d) connection to the Controller-Area Network (CAN) bus to read information from vehicle sensors (velocity, acceleration, brake pedal position, transmission state, etc.); (e) CAN bus interface with brake and throttle actuators. We provide illustrations of some of these systems in Figure 2.

(a) (b)

(c)

Fig. 2. Modified Lexus IS250 vehicles used in the experiments. In (a), a front profile of one test vehicle is provided. In (b), the top mounted WSU antenna is shown. In (c), system components are highlighted: main computer running the application, DGPS receiver, computer interface with the CAN bus, and Denso WSU.

A. Positioning and Inertial Measurements

The on-board DGPS unit is capable of .45 m accuracy for absolute position, 1.5° accuracy for absolute heading, and .1 s accuracy for absolute time. The measurement update rate is 10 Hz.

Other sensors include: (i) accelerometer, based on MEMS technology, capable of .5 m/s^2 accuracy; (ii) speedometer, measuring average speed at the wheel, capable .5 m/s accuracy; (iii) throttle pedal measurement, capable of .5 % accuracy; (iv) brake torque applied at wheel, capable of .5 Nm accuracy.

B. Communication System

A Denso WSU is used for V2V communication, which utilizes the industry standard DSRC message standard. The WSU has a CPU for eventual implementation of our algorithms, however at the moment is only used for sending and receiving messages.

Due to the requirement of uninterrupted V2V communication, we mount the WSU antenna on the vehicle roof, thus preventing packet loss often attributed to non-line-of-sight (NLOS). See Section 4.3.1 of [4] for an investigation of DSRC 5.9 GHz performance under varying line-of-sight (LOS) conditions.

C. Computation

The on-board computer mounted in the wheel well runs the Ubuntu open-source Linux distribution, which allows for rapid development and testing of Driver Safety Systems (DSS). The console display is connected to the computer via s-video, and a remote keyboard allows operation of the computer. The CPU is an Intel Core-Duo 2.0 GHz processor, with 1 GB of memory and a 150 GB hard drive. All software is compiled and run on this computer.

III. Modeling Formalism

We introduce the mathematical formalism used to describe the two-vehicle system. We rely on the fundamental assumption: as vehicles approach an intersection with high level route decided, traffic control mechanisms cause vehicles to take predefined trajectories through the intersection. In Figure 3, example trajectories are given for two adjacent vehicles coming together at a four way intersection. In the sequel, we will assume the collision scenario under study is the one presented in Figure 1, however ICA can be employed for any fixed choice of path.

To develop this formalism, we first present some basic mathematical notation. We then develop a single vehicle model, and extend it to the two-vehicle model. The utility of this modeling formalism lies in the ability of representing all collisions as a set of two-vehicle configurations, which will be developed in Section IV.

A. Mathematical Notation

We denote the real numbers $\mathbb{R}$ and the non-negative real numbers $\mathbb{R}_+$. For the set $U \subset \mathbb{R}$,

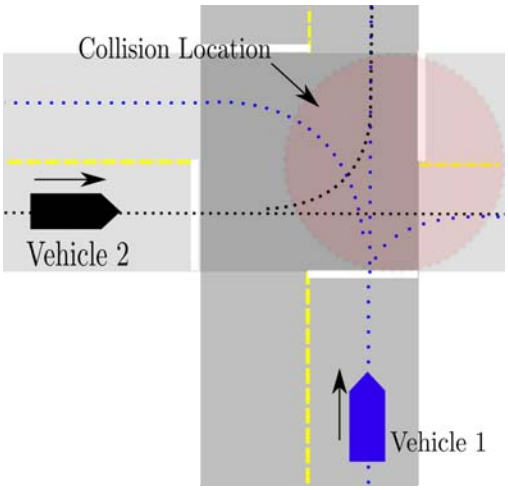


Fig. 3. Set of possible trajectories that could generate collisions between two adjacent vehicles passing through a four way intersection. The location of possible collisions is indicated around the intersections of paths.

we denote the set of causal signals as $S(U)$, that is, the set of all functions $f : \mathbb{R} \to U \cup \{0\}$ such that $f(t) = 0$ for all $t < 0$. For the function $f : X \to Y$, and set $A \subset X$, we will use the notation $f(A) := \bigcup_{x \in A} f(x)$. We use the interval notation $]a, b[$ to represent an open interval, that is, we have $x \in]a, b[$ if and only if $a < x < b$.

B. Single Vehicle Model

We use the *state space* $X \times V$ to model the dynamic state of the vehicle, where X represents the set of all possible longitudinal displacements and V represents the set of all possible longitudinal velocities. The corresponding vector $(x, v) \in X \times V$ denotes the *state* of the vehicle, where the x is the longitudinal displacement of the vehicle center of mass and v is the tangential velocity of the vehicle center of mass. We use the control space $U := [u_L, u_H] \subset \mathbb{R}$ to represent the scalar combination of all possible pedal and brake torque inputs, where u_L represents the maximum brake torque command, and u_H represents the maximum throttle torque command. We will denote the set of control signals $S(U)$ where $\mathbf{u} \in S(U)$ is called the *control signal*.

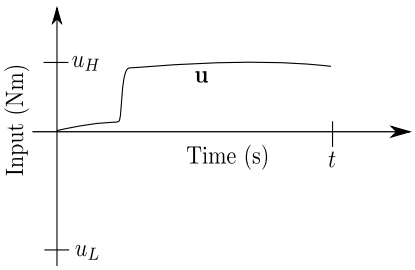
We introduce the *flow* of the system, given by the function $\phi : \mathbb{R}_+ \times X \times V \times S(U) \to X \times V$, which defines the evolution of the vehicle state as time goes forward [9], assumed to be generated by a controlled dynamical system $f : X \times V \times U \to X \times V$. That is, for an initial state $(x, v) \in X \times V$, control signal $\mathbf{u} \in S(U)$, and time $t \geq 0$, the state of the vehicle at time t is given as $\phi(t, (x, v), \mathbf{u}) \in X \times V$, where $\phi_1(t, (x, v), \mathbf{u}) \in X$ is the longitudinal displacement, and $\phi_2(t, (x, v), \mathbf{u}) \in V$ is the longitudinal velocity. The flow history over an interval of time, mathematically given as $\phi([0, t], (x, v), \mathbf{u})$, will be called a *trajectory* of the system. We provide an example in Figure 4, where the control signal $\mathbf{u} \in S(U)$ is provided, along with the trajectory both in the state space $X \times V$ and from a top-down view.

C. Two-Vehicle Model

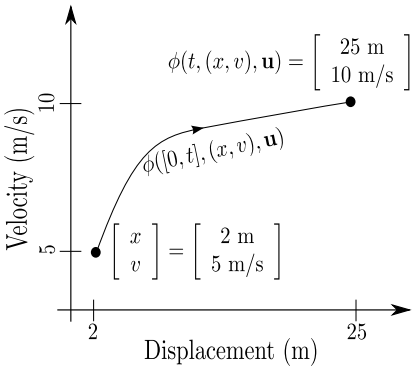
We next develop a two-vehicle modeling formalism, which will be used extensively to formulate the collision avoidance problem, describe the algorithm implementation, and interpret experimental data. We adopt the formalism of *parallel composition* to construct the two-vehicle model, where we can define a state vector for the *entire* system as $(x, v) := (x^1, x^2, v^1, v^2) \in X \times V := X^1 \times X^2 \times V^1 \times V^2$, where $(x^1, v^1) \in X^1 \times V^1$ is the state of vehicle 1, and $(x^2, v^2) \in X^2 \times V^2$ is the state of vehicle 2 (see [5] for more details). The control signal for the entire system is now given as $\mathbf{u} := (\mathbf{u}^1, \mathbf{u}^2) \in S(U) := S(U^1) \times S(U^2)$, where $\mathbf{u}^1 \in S(U^1)$ is the control signal to vehicle 1, and $\mathbf{u}^2 \in S(U^2)$ is the control signal to vehicle 2. The flow of the entire system is given by $\phi(t, (x, v), \mathbf{u}) \in X \times V$, where $\phi_1(t, (x, v), \mathbf{u}) \in X$ represents a vector of displacements, and $\phi_2(t, (x, v), \mathbf{u}) \in V$ represents a vector of velocities. We denote the flow of vehicle 1 as $\phi^1(t, (x, v), \mathbf{u}) \in X^1 \times V^1$ and the flow of vehicle 2 as $\phi^2(t, (x, v), \mathbf{u}) \in X^2 \times V^2$. To help illustrate this formalism, Figure 5 depicts an example of two vehicles approaching the intersection, both in the physical environment, and within the abstract state space $X \times V$.

IV. Problem Statement and Solution

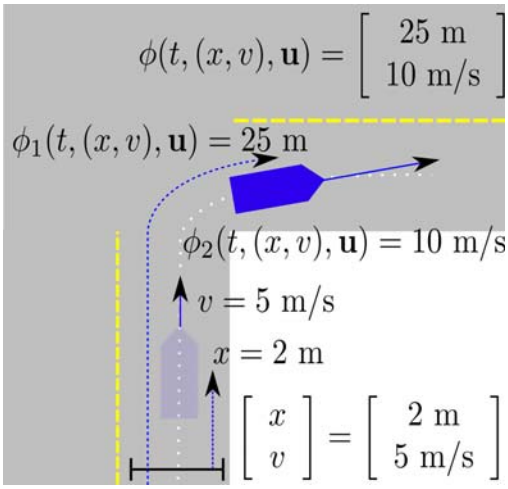
In this section, we present both the formal collision avoidance problem, and the solution we



(a)

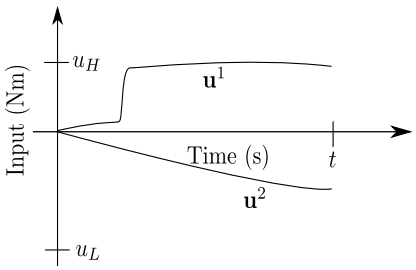


(b)

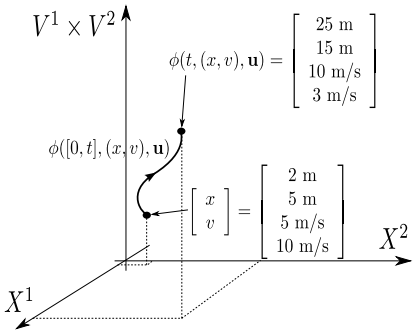


(c)

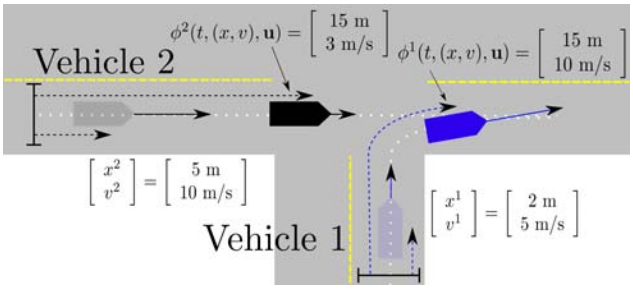
Fig. 4. Example trajectory of a single vehicle with initial condition $(x,v) = (2,5)$ and accelerating control $\mathbf{u} \in S(U)$ as shown. In (a), the control signal $\mathbf{u}$ is shown up to time t. In (b), the following are shown within the state space $X \times V$: the trajectory $\phi([0,t],(x,v),\mathbf{u})$; the initial condition (x,v); and the final state $\phi(t,(x,v),\mathbf{u})$. In (c), a top down snapshot of the system is shown at time 0 and time $t > 0$.



(a)



(b)



(c)

Fig. 5. Example trajectory of the two-vehicle system with initial condition $(x^1,v^1) = (2,5)$ and $(x^2,v^2) = (5,10)$, accelerating input $\mathbf{u}^1$ to vehicle 1 and braking input $\mathbf{u}^2$ to vehicle 2 as shown. In (a), the input signals $\mathbf{u}^1$ and $\mathbf{u}^2$ are shown up to time t. In (b), the initial condition (x,v), trajectory $\phi([0,t],(x,v),\mathbf{u})$, and final state $\phi(t,(x,v),\mathbf{u})$ are shown within the state space $X \times V$ (which is in 4-dimensional space). In (c), a top down snapshot of the system is shown at time 0 and time t.

implement. We start by formulating the problem in a mathematical manner consistent with the two-vehicle modeling approach introduced in Section III-C. We then present a solution to this problem with many technical details omitted, for which the interested reader is referred to [6].

A. Problem

We formalize collision avoidance as the problem of avoiding a *bad set* of two-vehicle states $\mathcal{B} \subset X$, which represents the set of all displacements along the path corresponding to a collision. With the modeling formalism introduced in Section III-C, we can define the bad set directly

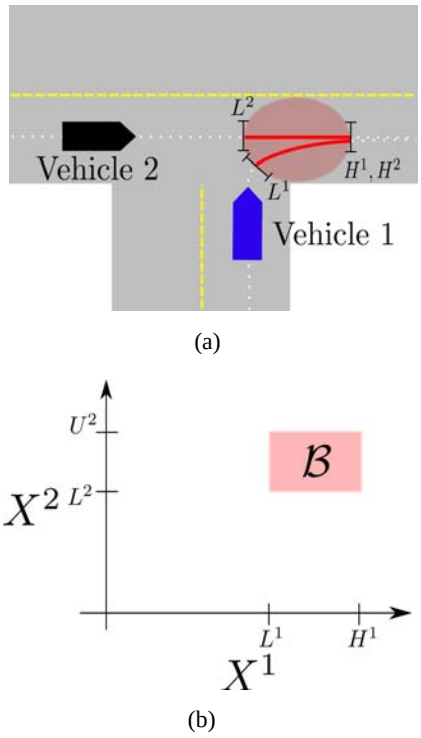


Fig. 6. Construction of the bad set $\mathcal{B}$ from the path geometry. In (a), the red circle denotes the location of potential intersection collisions, where the upper and lower bounds, H^1, H^2 and L^1, L^2 respectively, are shown along the paths. The paths are white dotted lines, with the sections inside the bad set highlighted bright red. In (b), the bad set $\mathcal{B}$ is shown within the space of two-vehicle displacements X. If the displacement of the two-vehicle system is inside the bad set, mathematically given by $x \in \mathcal{B}$, then a collision has occurred.

from the path geometry. In particular, for each path we can find a lower and upper displacement, such that if both vehicles are within their respective bounds, then we have a collision. The bad set can thus be defined as

$$\mathcal{B} := \left\{ x \in X \mid x^1 \in \,]L^1, H^1[\text{ and } x^2 \in]L^2, H^2[\right\},$$

where $L^1 < H^1$ and $L^2 < H^2$ are displacements along the respective paths, as seen in Figure 6. Therefore, the formal collision avoidance problem involves designing a controller that prevents the flow of the system from entering the bad set $\mathcal{B}$.

B. Solution

To prevent a collision between the two vehicles, we apply control based on the construction of a set $C(v) \subset X$ called the *capture* set, which is parametrized by the current velocity of the two-vehicle system. As the name indicates, the capture set corresponds to the set of vehicle displacements $x \in X$, such that given the current vehicle velocities $v \in V$, no control signal $\mathbf{u} \in S(U)$ can prevent an eventual collision. This is mathematically expressed as

$$C(v) := \left\{ x \in X \;\middle|\; \begin{array}{l} \forall\ \mathbf{u} \in S(U),\ \exists\ t \geq 0 \\ \text{s.t. } \phi_1(t, (x, v), \mathbf{u}) \in \mathcal{B} \end{array} \right\}.$$

An example is given in Figure 7, where a two-vehicle configuration is outside of the bad set, but within the capture set.

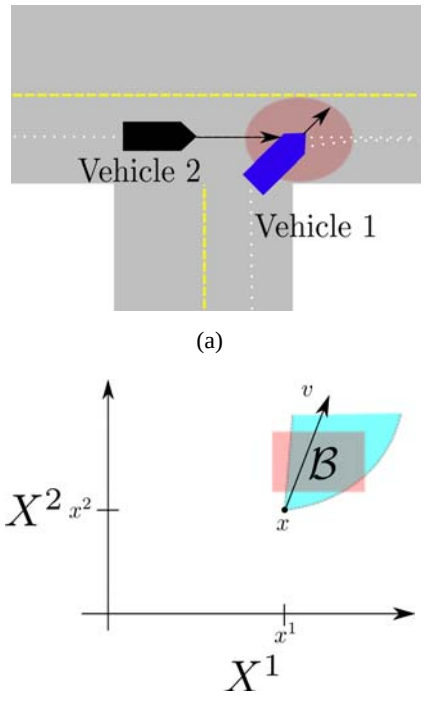


Fig. 7. Example of a two-vehicle system state outside of the bad set, but inside of the capture set. In (a), vehicle 1 is in the intersection with low velocity, while vehicle 2 is approaching the intersection with high velocity. In (b), the corresponding state x is shown in the space of displacements X. All possible future trajectories within X, mathematically given as $\phi_1([0, t], (x, v), S(U)) \subset X$, are depicted as the blue cone. From this depiction, we observe that every trajectory enters the bad set $\mathcal{B}$, implying a collision is immanent. By definition, this implies the current state is inside the capture set, mathematically given by $x \in C(v)$.

Our control approach is entirely based upon the capture set. We compute the capture set online and apply control if the flow enters the capture set boundary. If the flow enters the lower

boundary of the capture set, which corresponds to a configuration where vehicle 1 will enter the intersection before vehicle 2, the control signal $\mathbf{u}_* = (u_H^1, u_L^2)$ must be applied to avoid the collision, where vehicle 1 accelerates and vehicle 2 brakes. If the flow enters the upper boundary of the capture set, which corresponds to a configuration where vehicle 2 will enter the intersection before vehicle 1, the control signal $\mathbf{u}_* = (u_L^1, u_H^2)$ must be applied to avoid the collision, where vehicle 1 brakes and vehicle 2 accelerates. By construction, the resulting control prevents the flow from entering the capture set $C(v)$ and thus the bad set $\mathcal{B}$, which is formally shown in [6]. An example is provided in Figure 8, where the capture set is shown along with the evaluated control signal applied to prevent a collision.

V. ICA Application

This section contains an overview of the software components making up ICA. We separate the major software components into estimation, communication, and control. We first provide some notes regarding the general operation of all algorithms.

A. General Overview

Due to the short-time scales encountered in high-velocity collision scenarios, all algorithms update with a minimum periodic run time of 200 ms. From the distributed nature of the control problem, certain tasks, such as state estimation and state communication, run at higher frequency.

Message size must be small due to bandwidth limitations, therefore, vehicles cannot share detailed information such as UTM path waypoints, dynamic model used for prediction, or capture set parameterizations. Thus, the algorithm is designed such that memory intensive tasks are handled only on the local vehicle. We provide a high level software diagram in Figure 9.

The driver has the ability override automatic control commands by pressing either the brake or throttle pedal beyond a configurable threshold. However, ICA can no longer guarantee collision avoidance in the event of driver override.

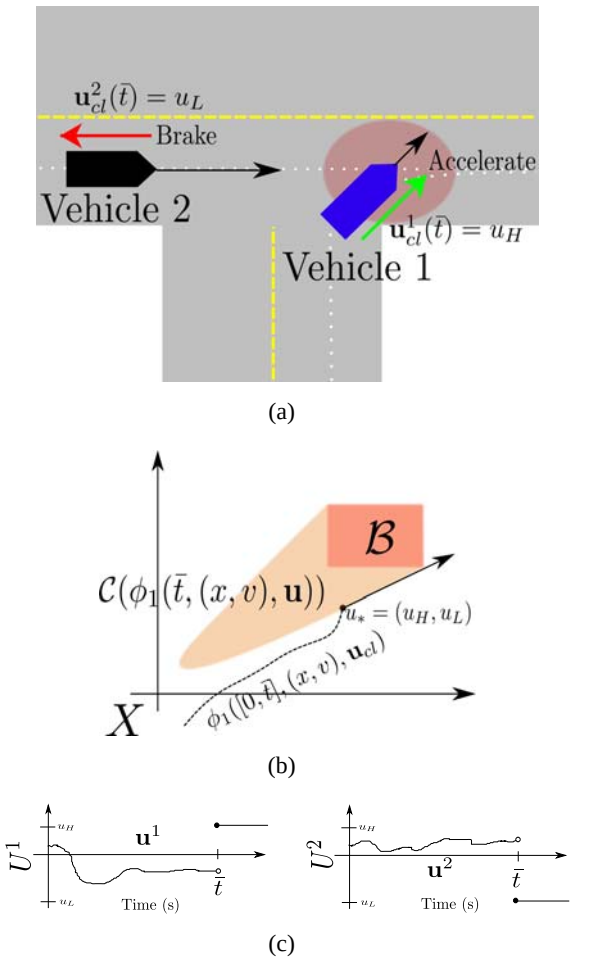


Fig. 8. An example two-vehicle configuration is given at time $\bar{t} > 0$. In (a), we see that vehicle 1 is close to the intersection with a low velocity, while vehicle 2 is far away with high velocity. In (b), the flow is on the lower boundary of the capture set $C(\phi_2(\bar{t}, (x, v), \mathbf{u}))$ at time $\bar{t}$, at which point control evaluates to $u_* = (u_H, u_L)$. As can be seen in the bottom figures, the control signal $\mathbf{u}$ is equal to u_* for all time $t \geq \bar{t}$ until vehicle 1 has passed through the intersection.

B. Estimation

For local vehicle state estimation, we employ a Hybrid Kalman Filter (details can be found in [21]), which combines measurements and an internal state model to estimate the current longitudinal state by minimizing the mean-squared error. The filter is also used to construct trajectory predictions over a configurable finite-time horizon, which are needed in control evaluation. The internal state model is learned off-line from experimental data trials, using a weighted least squares approach. The filter constructs predic-

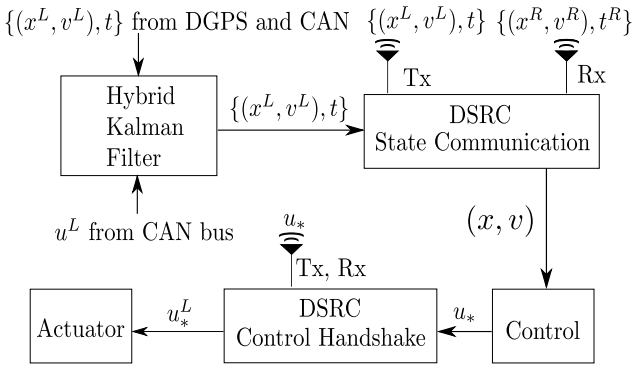


Fig. 9. Overview of ICA software modules, where the superscript L denotes the local vehicle and the superscript R denotes the remote vehicle. On the far left, the Hybrid Kalman Filter module comprises the estimator, where noisy measurements taken from DGPS and CAN are used to construct a state estimate. The state communication module is used to send local state information and receive remote state information. The control module uses the state of the entire system to determine if and what control action shall be taken to avoid collisions. If control action u_* is commanded by the control module, the control handshake module is utilized to make sure the vehicles agree on the control action to be taken. The local output u_*^L of this negotiation is sent to the local actuator for control application.

tions at a rate of 40 ms, due to the low computational burden and the need for state estimates to be as current as possible. Correction steps occur every 100 ms, corresponding to the update rate of the DGPS device.

C. Communication

The wireless V2V DSRC is comprised of two major modules: communication of state information and communication of control information.

State information is sent between vehicles to determine if automatic control action need be taken to prevent a collision. The state information sent is comprised of: (1) The longitudinal state of the local vehicle along with the current absolute time; (2) The predicted trajectory of the local vehicle state over a finite-horizon (configurable); (3) Bad set bounds for the local vehicle path; (4) Dynamic model for capture set construction.

When automatic control is evaluated by the control module, a wireless V2V DSRC handshake mechanism is utilized to guarantee both vehicles agree on the automatic control. This module serves a critical failsafe function, as two vehicles may not necessarily evaluate control simultaneously, which is an unfortunate artifact of the distributed nature of the control problem. Once the handshake module reaches agreement with the remote vehicle, a control command is sent to the local vehicle actuators for automatic control application.

D. Control

As mentioned in Section IV-B, control evaluation is determined completely by the capture set. If the current displacement of the two-vehicle system at time $t \geq 0$, given by $\phi_1(t, (x, v), \mathbf{u}) \in X$, is on the boundary of the capture set $C(v)$, then control must be applied. If this is not the case, no control is needed. In this section we introduce the algorithms used to compute the capture set, and the logic used to determine control action.

It can be shown that the longitudinal dynamics used to describe each vehicle generate *order preserving* systems. In [6], it has been shown that the capture set $C(v)$ can be computed using a back propagation algorithm of linear complexity with respect to system state. This algorithm is based on the computation of two sets, called restricted capture sets $C(v, u^*)$, which are defined to be the capture set when the input is fixed to $u^* \in U$ for all time. Rather than providing algorithmic details, we introduce a visualization of the algorithm in Figure 10.

Control action can be uniquely determined by checking membership of the state $\phi_1(t, (x, v), \mathbf{u}) \in X$ and the prediction $\phi_1(t + \delta, (x, v), \mathbf{u}) \in X$ within the capture sets $C(\phi_2(t, (x, v), \mathbf{u}))$ and $C(\phi_2(t + \delta, (x, v), \mathbf{u}))$ respectively. The choice of control will depend on whether the flow is on the upper or lower boundary of the capture set. In each case, evasive control will involve one vehicle applying a maximum brake torque input, with the second vehicle applying a maximum throttle torque input. The automatic control turns off on-board both vehicles after either vehicle has passed through the intersection.

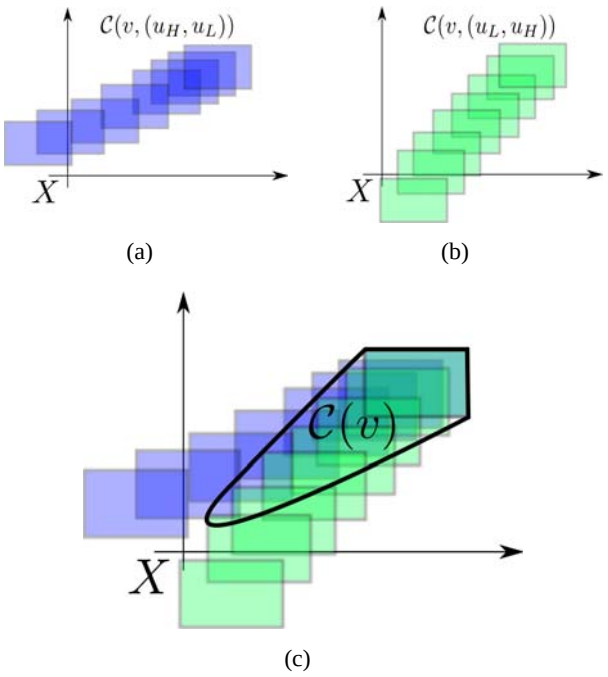


Fig. 10. Algorithm used to compute the capture set $C(v)$ in real time with an integration time step of T_s. In (a), the computation of the restricted capture set $C(v,(u_H,u_L))$ is depicted as a family of blue rectangles. In (b), the computation of the restricted capture set $C(v,(u_L,u_H))$ is shown as a family of green rectangles. In (c), the capture set is computed as $C(v) = C(v,(u_H,u_L)) \cap C(v,(u_L,u_H))$. The black outline represents the capture set $C(v)$ as the time step T_s used in the computation the restricted capture sets is taken to 0.

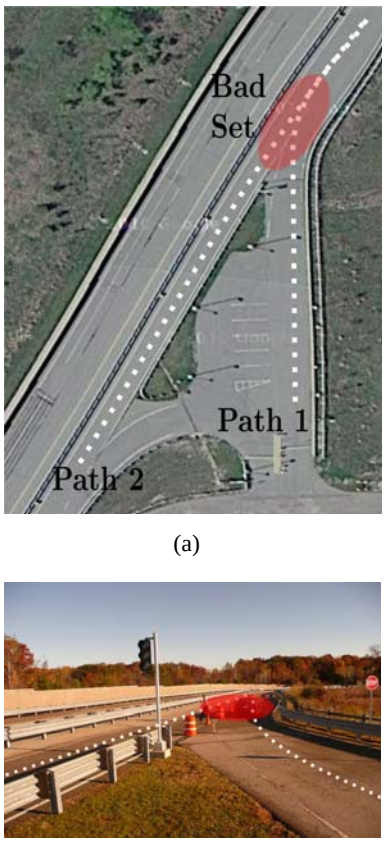


Fig. 11. Pictures of the test track section at TTC (Ann Arbor, Michigan) used for testing, where (a) is top down, and (b) is a side profile. All experiments carried out use this intersection topology. The paths are shown as colored dotted lines, and the bad set is shown as the red circle.

VI. Experimental Setup

All experiments are carried out on a test track at the Toyota Technical Center (TTC) in Ann Arbor, Michigan. In particular, we run the experiment on a section of the track containing two merging paths. A schematic of the layout is provided in Figure 11.

Both vehicles run ICA as they approach the intersection. The velocity of approach is not fixed, however must be within safe limits. Each path is stored as a list of UTM co-ordinates on the respective vehicle. The speed limits for path 1 are v_{min} = 0 MPH and v_{max} = 25 MPH, while the speed limits for path 2 are v_{min} = 20 MPH and v_{max} = 40 MPH. The bad set parameters chosen are L^1 = 55 m, L^2 = 75 m, U^1 = 65 m and U^2 = 85 m.

The maximum brake u_L input and maximum throttle u_H input were chosen independently for each vehicle such that the vehicle did not accelerate in a potentially unsafe manner. During experimentation, drivers can override automatic control commands for safety purposes.

The testing procedure is provided as a series of steps:

1) Start the application at rest;
2) Accelerate each vehicle to a reasonable cruising velocity;
3) Upon automatic control activation, driver does not cancel commands;
4) Once one vehicle passes through the intersection, automatic control is disabled and the drivers bring the vehicles to a stop.

VII. Experimental Results

We now provide experimental results from a trial conducted at the TTC test track, as described in Section VI. The safety of the system

is verified in the experiment by checking that the trajectory $\phi_1(\mathbb{R}_+, (x, v), \mathbf{u})$ has empty intersection with the bad set $\mathcal{B}$. In addition, the adherence to the speed limits can be verified by checking that the velocity of each vehicle is within the interval $[v_{min}, v_{max}]$ for all time.

The experimental data provided corresponds to a scenario where vehicle 1 has entered the intersection at low velocity, such as a driver turning right on red, and neglected to check for oncoming traffic. The solution is maximum throttle control u_H applied to vehicle 1, and maximum brake control u_L applied to vehicle 2. In Figure 12, we provide four snap shots of the evolution of the system as seen in the set of two vehicle longitudinal displacement (left) and from the top-down world-view perspective. In Figure 13, signals depicting the dynamic evolution of each vehicle are provided for both vehicles. Note that the speed limits, represented as black lines in the velocity signal subplots, are never violated by automatic control application. Until automatic control is applied at 18.6 seconds, all vehicle inputs are provided by the drivers. After the collision is avoided (after one vehicle has passed through the respective bad interval), the automatic control was terminated and the drivers resumed applying nominal inputs.

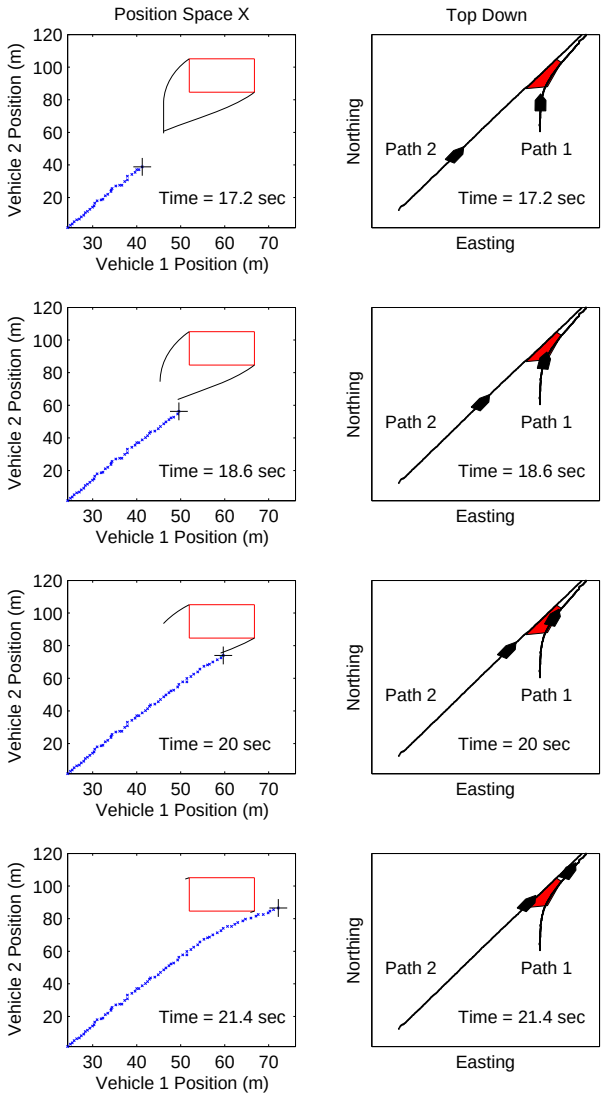


Fig. 12. Experimental results from a single trial. In the left column, snap-shots of the bad set, capture set, and trajectory of two-vehicle system are shown. In the right column, snap-shots are given from a top-down perspective, with the bad set, paths and absolute vehicle positions. A collision is predicted at 18.6 seconds, corresponding to the second row of figures. Experimental success can be verified from the empty intersection of the bad set $\mathcal{B}$ and the trajectory $\phi_1([0, 21.4], (x, v), \mathbf{u})$, as seen in the last row.

VIII. Conclusion

In this paper, we have provided experimental results for an automatic control system capable of preventing collisions between two vehicles at an intersection. Data from an experimental trial is provided, depicting both the on-line algorithmic implementation and dynamic evolution of the system as it successfully prevents a collision.

In the future, we plan on including a human machine interface (HMI) for implementing a warning system. We are also working on implementing this collision avoidance system in cases where V2V communication is not available, as would be the case if one of the vehicles did not have the safety system installed [24, 25]. Lastly, we are researching the extension of our methods to intersection collisions involving more than 2 vehicles, with potentially more general collision topologies.

References

[1] L. Alvarez and R. Horowitz. Analysis and verification of the PATH AHS coordination-regulation layers hybrid system. In *American Control Conference*, pages 2460–2461, Albuquerque, New Mexico, 1997.

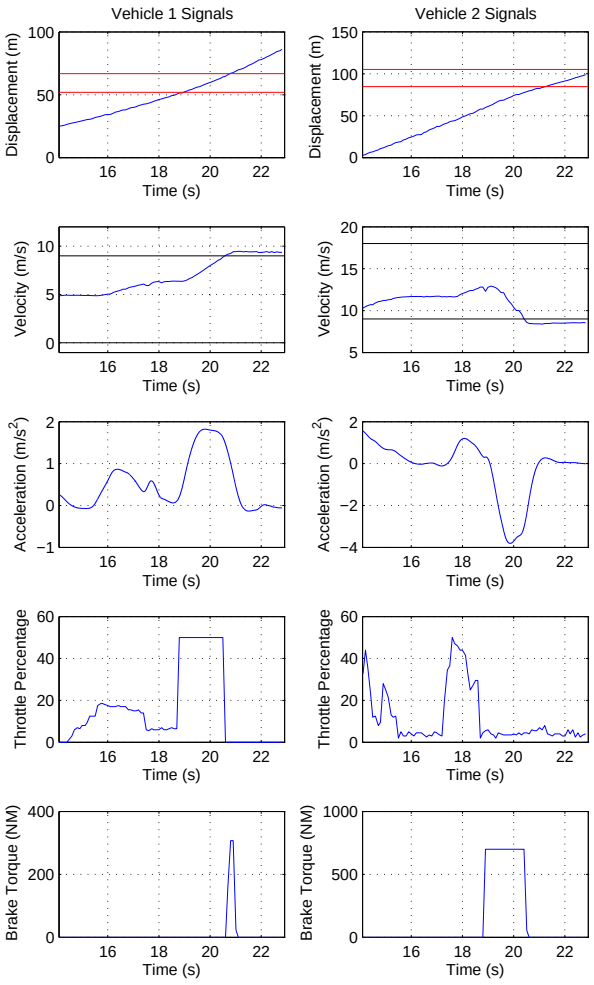


Fig. 13. Experimental results from a single trial. The signals in the left column correspond to vehicle 1, while the right column correspond to vehicle 2. The red lines within the displacement subplots correspond to upper and lower bounds of the bad interval. The black lines within the velocity subplots correspond to speed limits. Automatic control is first applied at 18.6 seconds.

[2] L. Alvarez and R. Horowitz. Hybrid controller design for safe maneuvering in the PATH AHS architecture. In *American Control Conference*, pages 2454–2459, Albuquerque, New Mexico, 1997.

[3] L. Alvarez and R. Horowitz. Safe platooning in automated highway systems. *California Partners for Advanced Transit and Highways (PATH). Research Reports: Paper UCB-ITS-PRR-97-46*, Jan 1997.

[4] S. Andrews and M. Cops. Vehicle Infrastructure Integration Proof of Concept Technical Description - Vehicle. *U.S. Department of Transportation*. Report: FHWA - JPO-09-043. http://ntl.bts.gov/lib/31000/31100/31135/14477.htm, 2009.

[5] C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Springer, 2nd edition, 2007.

[6] D. Del Vecchio, M. Malisoff, and R. Verma. A separation principle for a class of hybrid automata on a partial order. In *American Control Conference*, 2009.

[7] V. Desaraju, H. C. Ro, M. Yang, E. Tay, S. Roth, and D. Del Vecchio. Partial order techniques for vehicle collision avoidance: Application to an autonomous roundabout testbed. In *International Conference on Robotics and Automation*, pages 82–87, 2009.

[8] J. Duperret, M. Hafner, and D. Del Vecchio. Formal design of a provably safe robotic roundabout system. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[9] G. F. Franklin, J. D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice Hall, fourth edition, 2002.

[10] J. A. Haddon, D. N. Godbole, A. Deshpande, and J. Lygeros. Verification of hybrid systems: Monotonicity in the AHS control system. In *Hybrid Systems III*. Lecture Notes in Computer Science, vol. 1066. Springer, 1996.

[11] M. R. Hafner and D. Del Vecchio. Computation of safety control for uncertain piecewise continuous systems on a partial order. In *Conf. on Decision and Control*, pages 1671–1677, 2009.

[12] J. K. Hedrick, Y. Chen, and S. Mahal. Optimized vehicle control/communication interaction in an automated highway system. *California Partners for Advanced Transit and Highways (PATH). Research Reports: Paper UCB-ITS-PRR-2001-29*, 2001.

[13] R. Horowitz and P. Varaiya. Control design of an automated highway system. *Proceedings of the IEEE*, 88(7):913–925, Jul 2000.

[14] R. Horowitz and P. Varaiya. Control design of an automated highway system. *Proceedings of the IEEE*, 88(7):913–925, 2000.

[15] H. Kowshik, D. Caveney, and P. R. Kumar. Provable systemwide safety in intelligent intersections. *Transactions on Vehicular Technology*, 60(3):804–818, March 2011.

[16] J. Lygeros, D. N. Godbole, and S. Sastry. A verified hybrid controller for automated vehicles. In *Conf. on Decision and Control*, pages 2289–2294, Kobe, Japan, 1996.

[17] J. Lygeros and N. Lynch. Strings of vehicles: Modeling and safety conditions. pages 273–288, 1998.

[18] U.S. DOT National Highway Traffic Administration (NHTSA). Traffic Safety Facts. http://www-nrd.nhtsa.dot.gov/Pubs/811402EE.pdf, 2009.

[19] U.S. DOT National Highway Traffic Administration (NHTSA). Crash Factors in Intersection-Related Crashes: An On-Scene Perspective. http://www-nrd.nhtsa.dot.gov/Pubs/811366.pdf, September 2010.

[20] A. Puri and P. Varaiya. Driving safely in smart cars. In *American Control Conference*, pages 3597–3599, Seattle, WA, 1995.

[21] D. Simon. *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley-Interscience, 2006.

[22] C. J. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, 2000.

[23] P. Varaiya. Smart cars on smart roads. *IEEE Transactions on Automatic Control*, 38(2):195–207, Feb 1993.

[24] R. Verma and D. Del Vecchio. Safety control of hidden mode hybrid systems. *IEEE Transactions on Automatic Control*, Accepted, 2011.

[25] R. Verma and D. Del Vecchio. Safety in semi-autonomous multi-vehicle systems: A hybrid control approach. *IEEE Robotics Magazine*, Accepted, 2011.