Partial Order Techniques for Vehicle Collision Avoidance: Application to an Autonomous Roundabout Test-bed

V. Desaraju, H. C. Ro, M. Yang, E. Tay, S. Roth, and D. Del Vecchio

***Abstract*— In this paper, we employ partial order techniques to develop linear complexity algorithms for guaranteed collision avoidance between vehicles at highway and roundabout mergings. These techniques can be employed by virtue of the rich structure offered by such traffic systems, which constrain vehicles to advance unidirectionally along a path. The algorithms are safe by construction while maintaining the liveness of the system. The proposed algorithms are on-line implemented in a decentralized fashion on an experimental testbed composed of two in-scale communicating vehicles continuously running on an autonomous roundabout system.**

I. INTRODUCTION

Intelligent Transportation Systems (ITS) for in-vehicle cooperative active safety and related technologies continue to be examined world-wide by government and industry consortium, such as the Crash Avoidance Metrics Partnership (CAMP) [2], the Vehicle Infrastructure Integration Consortium (VIIC) [4], [3] in the U.S., the Car2Car Communications Consortium in Europe [1], the Advanced Safety Vehicle project 3 (ASV3) in Japan, and by university research centers such as the Virginia Tech Transportation Institute (VTTI) and the California PATH. In the near future, ITS is expected to become more comprehensive connecting vehicles with each other and with the surrounding road infrastructure through vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) wireless communication. Thus, complete position and speed information will be available to all vehicles in a given neighborhood for cooperative active safety purposes.

In this work, we focus on the two-vehicle collision avoidance problem as found, for example, in one-lane modern roundabout systems, where conflicts involve only two vehicles at the time through merging points. In order to guarantee safety, we employ an approach based on the computation of the capture set of an unsafe set corresponding to collision configurations (see for example [7] and the references therein). Such an approach has been employed especially in the context of collision avoidance in air-traffic control and in platooning [7], [9], [5]. It produces control maps that are guaranteed to maintain safety by construction. As it appears from these previous works, there are however two main difficulties in the computation of the capture set: Its computation does not generally scale with the size of the system and it is not guaranteed to terminate [9]. Some results to tackle these problems for discrete time hybrid automata models were proposed in [6].

In this paper, we show that the rich structure offered by automobile driving can be directly exploited to overcome the above mentioned problems. We obtain scalable algorithms that are guaranteed to terminate and that do not require expensive memory storage. In particular, the structural properties that we directly exploit are the following: (i) on its path, a vehicle can move in one direction only; (ii) for a fixed path, the higher the control force applied to a vehicle, the higher the longitudinal positions and speeds achieved on the path; (iii) for a fixed path and control force applied to the vehicle, the higher the initial speeds and positions, the higher the speeds and positions achieved at any later time on the path. These properties are mathematically formalized in the paper by introducing the concept of partial order and of order preserving dynamics. We show the real-time applicability of the obtained control algorithms by implementing them in a decentralized fashion on an experimental Roundabout Drill system composed of two in-scale autonomous vehicles that are continuously running.

This paper is organized as follows. In Section II, we model the system as a hybrid automaton, we formalize its order preserving properties and give the main computational results for the computation of the capture set. In Section III, we illustrate the experimental set up, the implementation of the control algorithm, and the experimental results.

II. SYSTEM MODEL AND PARTIAL ORDER CONTROL TECHNIQUE

A. Roundabout Drill

In order to solve the two-vehicle collision avoidance problem, we consider the Roundabout Drill system as shown in Figure 1. The blue vehicle always runs on the internal circle, while the red vehicle always runs on the external circle. Collisions occur if the two vehicles are both at the same time in the shaded area around point C. The objective is to maintain the vehicles continuously running (liveness) while avoiding collisions through the enforcement of least restrictive control actions (safety). For mathematically describing the system, we introduce coordinates for each of the vehicles along their paths, given an arbitrary reference point (denoted with 0 in Figure 1). Let $p \in \mathbb{R}$ denote the longitudinal displacement along the vehicle path in such a coordinate system. The longitudinal vehicle dynamics can thus be written as $\ddot{p} = [\mathcal{R}^2/(J_w + \mathcal{M}\mathcal{R}^2)](f_w - f_{brake} - \frac{\rho_{air}}{2} C_D A_f v^2 - C_{rr}\mathcal{M}g - \mathcal{M}g\sin(\theta_{road}))$, in which $\mathcal{R}$ is the tire radius, J_w is the wheel inertia, $\mathcal{M}$ is the mass of

This work was supported in part by NSF CAREER Award Number CNS-0642719 and by the SUROP and UROP programs at University of Michigan

V. Desaraju is with the Aeronautics Department at MIT, H. C. Ro, M. Yang, E. Tay, S. Roth, and D. Del Vecchio are with the Systems Laboratory at University of Michigan, Ann Arbor rajeswar@umich.edu

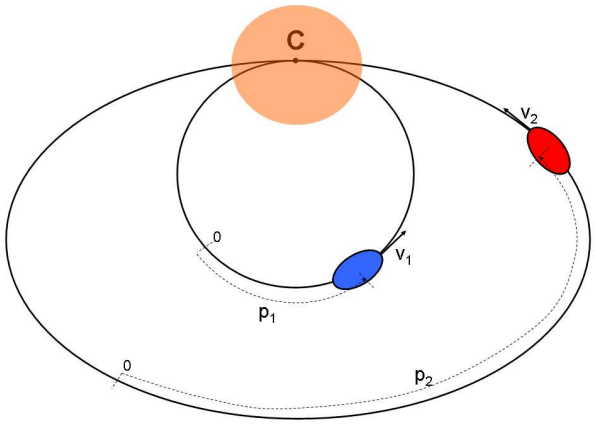


Fig. 1. Roundabout Drill system. The position of each vehicle along its path is denoted p_i, while its longitudinal speed along the path is denoted v_i. The conflict point C is where the vehicles paths intersect.

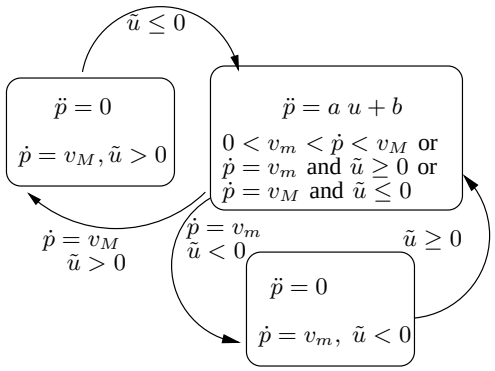


Fig. 2. Hybrid automaton modeling the longitudinal dynamics of each vehicle in the Roundabout Drill. Here $\tilde{u} := a\ u + b$. In mode 1, the dynamics of the vehicle is given by $\ddot{p} = a\ u + b$, while in modes 2 and 3 it is given by $\ddot{p} = 0$.

the vehicle, $f_w = \tau_w \mathcal{R}$ where τ_w is the drive shaft output torque, f_{brake} is the brake force, ρ_{air} is the air density, C_D is the drag coefficient, A_f is the projected front area of the vehicle, v is the longitudinal vehicle velocity, C_{rr} is the rolling resistance coefficient, g is the gravity constant, and θ_{road} is the road gradient. For more details on this model, the reader is referred to [11] and to the references therein. For automatic driving, f_w and f_{brake} are control inputs to the longitudinal dynamics of the vehicle. Assuming that the road is flat and that the air drag term is negligible, we can re-write the longitudinal dynamics as $\ddot{p} = a\ u + b$, in which $u = f_w - f_{brake}$ is the total force, which is the control input to the vehicle, $a = \mathcal{R}^2/(J_w + \mathcal{M}\mathcal{R}^2)$, and $b = -\mathcal{R}^2/(J_w + \mathcal{M}\mathcal{R}^2)\ C_{rr}\mathcal{M}g$. In order to maintain liveness, that is, to prevent a situation in which any of the vehicles stop, we implement for each of the two vehicles a hybrid controller that keeps the speed from decreasing below a low speed v_m and from exceeding a high speed v_M (Figure 2).

B. Order preserving hybrid automaton model

Each of the vehicles on the roundabout is thus modeled as a *hybrid automaton with input* $H = (X, \mathcal{Q}, \mathcal{U}, f, R)$, in which $X \subset \mathbb{R}^n$ is the set of *continuous variables*, $\mathcal{Q}$ is a finite set of *modes*, $\mathcal{U}$ is a *continuous set of inputs*, $f : X \times \mathcal{Q} \times \mathcal{U} \to X$ is a vector field, and $R : X \times \mathcal{U} \to \mathcal{Q}$ is the *mode reset map*. The mode reset map R is defined as $R(x, u) := q$ if $(x, u) \in \mathrm{Dom}(q)$, in which $\mathrm{Dom} : \mathcal{Q} \to 2^{X \times \mathcal{U}}$ is a map that attaches to a mode the set of continuous states and inputs in which the mode holds. We use the notation $t \mapsto \phi(t, x, \mathbf{u})$ to denote the flow (or trajectory) of H starting at initial condition $x \in X$ at initial time zero, when input signal $\mathbf{u}$ is applied to H. When the initial condition and input are clear from the context, we denote the flow by $x(t)$.

Let $\mathcal{U} \subseteq \mathbb{R}^n$ be compact and $\mathcal{F}(\mathcal{U})$ denote the set of all piecewise continuous functions $\mathbf{u} : \mathbb{R} \to \mathcal{U}$. We establish the partial order $(\mathcal{F}(\mathcal{U}), \leq)$ by defining $\mathbf{u^a} \leq \mathbf{u^b}$ provided that $\mathbf{u^a}(t) \leq \mathbf{u^b}(t)$ for all $t \in \mathbb{R}$, for all $\mathbf{u^a}, \mathbf{u^b} \in \mathcal{F}(\mathcal{U})$. Consider the partial order $(\mathbb{R}^n, \leq)$ defined by component-wise ordering and the partial order $(\mathcal{F}(\mathcal{U}), \leq)$ on the set of all piecewise continuous functions. Given partial orders $(P, \leq)$ and $(S, \leq)$ and a map $F : P \to S$, we say that F is order preserving if $x_1 \leq x_2$ implies $F(x_1) \leq F(x_2)$ for $x_1, x_2 \in P$. Let $e_1 = (1, 0, ..., 0) \in \mathbb{R}^n$.

Definition 1: We say that $H = (X, \mathcal{Q}, \mathcal{U}, f, R)$ is *order preserving* provided there exist constants $u_m, u_M \in \mathbb{R}$ and a positive constant γ such that the following hold:

(i) $\mathcal{U} = [u_m, u_M] \subset \mathbb{R}$;

(ii) The flow $\phi(t, x, \mathbf{u})$ is order preserving with respect to the x variable and with respect to the $\mathbf{u}$ variable;

(iii) $< f(x, R(x, u), u), e_1 > \geq \gamma$ for all $x, u \in X \times \mathcal{U}$.

One can check that the hybrid automaton of Figure 2 modeling each vehicle in the Roundabout Drill is order preserving. The overall Roundabout Drill system can thus be modeled as the parallel composition of two order preserving hybrid automata. That is, $H_{\text{roundabout}} = H_1 || H_2$, in which H_i are represented in Figure 2. For each H_i, we have that $x_i = (p_i, v_i) \in \mathbb{R}^2$ with $x_{i,1} := p_i$ and $x_{i,2} := v_i$, the mode q_i can be in one of the three possible modes of Figure 2, the vector fields f_i are given by $f_i(x_i, q_i, u_i) = (x_{i,2}, a_i\ u_i + b_i)$ for q_i in mode 1 and $f_i(x_i, q_i, u_i) = (x_{i,2}, 0)$ for q_i in modes 2 and 3.

C. Control design

We seek to determine control laws that guarantee that the continuous state of system $H = H_1 || H_2$ never enter the bad set $B := \{x \in X \mid x_{i,1} \in]L_i, U_i[\text{ for } i = 1, 2\}$. A system whose trajectories do not enter set B is said to be safe. For the roundabout system, entering such a set B corresponds, for suitable L_i and U_i, to having both of the vehicles of Figure 1 be at the same time in the shaded ball.

In order to avoid bad set B, we determine the set of all initial system configurations that independently of the control input lead to trajectories of H entering B in finite time. This set is called the *capture set* and it is denoted by $\mathcal{C}$. It is mathematically characterized by $\mathcal{C} := \{x \in X \mid \forall\ \mathbf{u} \in \mathcal{F}(\mathcal{U})\ ,\ \exists\ t \geq 0 \text{ s.t. } \phi(t, x, \mathbf{u}) \in B\}$. A control map that makes the system safe is one that allows all control inputs

if the system configuration is outside of the capture set, while it allows only safe control actions on the boundary of the capture set. The main bottleneck in applying this approach to nonlinear and hybrid systems is the computation of the capture set itself. Computation usually does not scale with the size of the system and is not guaranteed to terminate in finite time. The structure provided by the parallel composition of order preserving hybrid automata allows us to overcome these difficulties. We provide scalable algorithms to compute the capture set, which are also guaranteed to terminate. These algorithms are obtained by virtue of the following central result. Let $x = (x_1, x_2) \in X$ be the state of $H_1||H_2$ with H_i order preserving hybrid automata. Let $x_i = (x_{i,1}, ..., x_{i,n_i})$, $u_B := (u_{1,m}, u_{2,M})$, and $u_C := (u_{1,M}, u_{2,m})$. For a constant input signal $\mathbf{u}(t) = u$ for all t, define $\mathcal{C}_u = \{x \in X \mid \exists t \geq 0 \text{ s.t. } \phi(t, x, u) \in B\}$. This set represents all x configurations that are taken to B by the flow of the system when the input is constant and equals u. The next result shows that the capture set $\mathcal{C}$ can be computed by computing only the two sets $\mathcal{C}_{u_C}$ and $\mathcal{C}_{u_B}$.

Theorem 1: $\mathcal{C} = \mathcal{C}_{u_B} \cap \mathcal{C}_{u_C}$.

The proof of this theorem is in [10]. For the Roundabout Drill system, this theorem implies the following. If a configuration x (speeds and positions of both vehicles) is taken to B by having vehicle 1 apply maximal acceleration and vehicle 2 apply maximal deceleration and it is also taken to B by having vehicle 1 apply maximal deceleration and vehicle 2 apply maximal acceleration, then any other (non-constant) control inputs will also take x to B. By virtue of this result, to compute $\mathcal{C}$, we can compute only the sets $\mathcal{C}_{u_B}$ and $\mathcal{C}_{u_C}$. Due to the order preserving property of the flow, sets C_{u_B} and C_{u_C} can be on-line computed with a linear complexity algorithm, which is also guaranteed to terminate. This algorithm is provided in Section III-D. The resulting control map that renders the system safe is obtained as

$$g(x) := \begin{cases} \{u_B\} & \text{if } x \in \mathcal{C}_{u_C} \cap \partial\mathcal{C}_{u_B} \\ \{u_C\} & \text{if } x \in \mathcal{C}_{u_B} \cap \partial\mathcal{C}_{u_C} \\ \{u_C, u_B\} & \text{if } x \in \partial\mathcal{C}_{u_B} \cap \partial\mathcal{C}_{u_C} \\ \mathcal{U} & \text{otherwise.} \end{cases} \quad (1)$$

It applies control actions only when the system configuration is on the boundary of the capture set $\mathcal{C}$. When the system configuration is not on the boundary of $\mathcal{C}$, any control input is allowed.

III. EXPERIMENTS

A. Experimental setup

The testbed where the experiments are performed consists of a 6m x 6m arena in which the vehicles are driven and a positioning system, which serves as an in-lab version of GPS. A local 802.11b wireless network provides access to the vehicles from a workstation using a Secure Shell terminal. It also allows inter-vehicle communication via UDP.

The positioning system employs a Hexamite Hx11 ultrasonic system configured for guidance applications. The system uses frequencies in the band of 40 kHz $\pm$ 1 kHz. Each vehicle has a transponder mounted on the front, and each transponder is assigned a different delay value, which determines its transmission frequency. There are 24 passive units on the ceiling that act like mirrors and reflect these transmissions back to the vehicles along with the ceiling unit's ID number. These units are arranged in a grid, forming equilateral triangles with 4ft long sides. The transponder on the vehicle returns the round trip time for each received reflection to the CPU mounted on each vehicle. These times are then used to determine the distance from the transponder to the corresponding ceiling units. When three or more valid distances are received, the vehicle's position is calculated via trilateration using the algorithm described in [8]

The vehicles (Figure 3) are custom built on a Tamiya TT-01R model car chassis modified to be front-wheel drive. The drive motor is a 7.2V DC motor. Mounted on the chassis are the microcontroller and the main CPU. Power is supplied by two 16.5 V batteries in parallel.

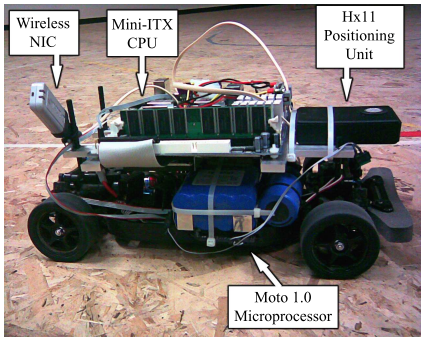


Fig. 3. Vehicles employed in the lab experiments on the Roundabout Drill.

The motor and the steering servo are controlled by an Acroname Moto 1.0 microcontroller. The Moto 1.0 has a 40 MHz processor and 368 bytes of RAM. The software on the Moto 1.0 is used to implement a motor map, allowing for torque control rather than speed control. For details on the motor map, the reader is referred to [11].

The main CPU is a VIA EPIA Mini-ITX with a 600 MHz processor, 512 MB of RAM, and a 40 GB hard drive. All control, positioning, and communications algorithms are written in C and run on the Mini-ITX on a Linux Fedora Core 5 operating system. The resulting torque commands from the control algorithms are transmitted to the Moto 1.0 via shared scratchpad memory to be applied to the motors. The software on both the Moto 1.0 and the Mini-ITX is asynchronously triggered every 100ms in order to provide enough time for the Moto 1.0 code to perform the motor map calculation and the necessary I/O to the hardware and the scratchpad memory.

B. Vehicle longitudinal dynamics model

At full charge, the vehicles are able to reach speeds up to 2.5 m/s. The longitudinal dynamics are modeled as a second order model of the form $\ddot{p} = au + b$, where $\ddot{p}$ is the vehicle acceleration and u is the torque command. Parameters a and b for each vehicle are experimentally determined from data by employing standard least squares

estimation techniques. In particular, we ran each vehicle with constant torque commands, recorded the speed profile as it accelerated, and employed the least squares method to fit a line to the speed profile. The slope of this line was taken as the acceleration for that torque command. We then fixed the value of b by measuring the deceleration with a command of zero torque and determined the parameter a using a least squares linear fit on the acceleration versus torque data. The resulting models for vehicles 1 and 2 are, respectively, $\ddot{p}_1 = 1.20u_1 - 0.90$, $\ddot{p}_2 = 1.26u_2 - 1.15$. The torque command, u, is issued as a percentage from 0 to 100, with 100 corresponding to a torque of 0.09 N m. The longitudinal dynamics model given by the hybrid automaton of Figure 2 is obtained by implementing a speed limiter. In particular, we set $v_M = 0.80m/s$ and $v_m = 0.25m/s$. The speeds v_M and v_m in modes 2 and 3 are maintained through the employment of a proportional/derivative (PD) speed control.

C. Path following algorithm implementation

Vehicle control has two main components: maintaining the vehicles on the corresponding roundabout paths and applying the appropriate control torque u to the longitudinal dynamics to prevent collisions at point C (Figure 1). In general, the longitudinal and lateral dynamics of a vehicle are coupled. However, since the radii of the paths are much greater than the length of the vehicles and the speeds are low, it is possible to assume low coupling. This allows us to decouple the path following task, using a steering control input, from the longitudinal dynamics control, using the torque control input u.

In order to make the positioning of the vehicles more accurate and reliable and to obtain heading estimates, we implement a state estimator, which employs the kinematic model of the vehicle

$$\dot{s}_x = v\cos(\zeta+\gamma),\ \dot{s}_y = v\sin(\zeta+\gamma),\ \dot{\zeta} = \tfrac{v}{B}\sin(\gamma), \tag{2}$$

where s_x and s_y are the vehicle's coordinates, ζ is its heading angle, γ is its steering angle, v is its speed, and B is its length, as depicted in Figure 4.

In every iteration, the estimator uses local steering and

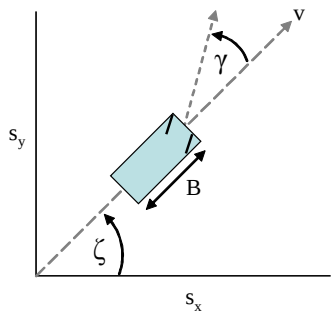


Fig. 4. Kinematic model parameters relative to the vehicle.

speed measurements to evolve this basic bicycle model over time. Whenever valid position data is received, the estimator adjusts the position estimate by taking a weighted average of the prediction and the measurement. Due to the absence of a heading measurement, the heading angle estimate is updated by normalizing the three most recent estimated displacement vectors, adding them, and taking the angle formed by the result. This is essentially a moving average filter on the heading angle, where normalization removes the unwanted weighting introduced by the vector magnitudes. Let $\Delta T > 0$ be the sampling time (100ms). Denote by $\hat{s} = (\hat{s}_x, \hat{s}_y)$ the position estimate and by $\hat{\zeta}$ the heading estimate. Define $s_{x,pred}(k) := \hat{s}_x(k-1) + v(k-1)\cos(\hat{\zeta}(k-1) + \gamma(k-1))\Delta T$ and $s_{y,pred}(k) := \hat{s}_y(k-1)+v(k-1)\sin(\hat{\zeta}(k-1)+\gamma(k-1))\Delta T$. Similarly, let $s_{x,meas}$ and $s_{y,meas}$ denote the measurement obtained by the positioning system. Then, we have the update laws $\hat{s}_x(k) = (c\ s_{x,pred}(k) + d\ s_{x,meas}(k))/(c+d)$, $\hat{s}_y(k) = (c\ s_{y,pred}(k)+d\ s_{y,meas}(k))/(c+d)$ for the position estimates and $\hat{\zeta}(k) = \text{angle}\left(\sum_{i=k-2}^{k}\left(\frac{\hat{s}(i)-\hat{s}(i-1)}{||\hat{s}(i)-\hat{s}(i-1)||_2}\right)\right) + \frac{v(k-1)}{B}\sin(\gamma(k-1))\Delta T$, for the heading estimates. The weights c and d that are found to provide the best estimator performance are 3 and 2, respectively.

The position and heading estimates feed directly into the path following algorithm. This controller accesses an array of position coordinates that define the desired trajectory for the vehicle. Each vehicle has a different array, each corresponding to one of the two loops of Figure 1. The control algorithm determines the closest point in the array that the vehicle is approaching and uses a proportional feedback to adjust its steering such that the vehicle's heading is adjusted toward the target point. Letting $s_{x,target}$ and $s_{y,target}$ denote the coordinates of the target point, we thus have $\zeta_{target}(k) = \arctan\left(\frac{s_{y,target}(k)-\hat{s}_y(k)}{s_{x,target}(k)-\hat{s}_x(k)}\right)$, $\gamma(k) = K_P(\zeta_{target}(k)-\hat{\zeta}(k))$.

D. Collision avoidance algorithm implementation

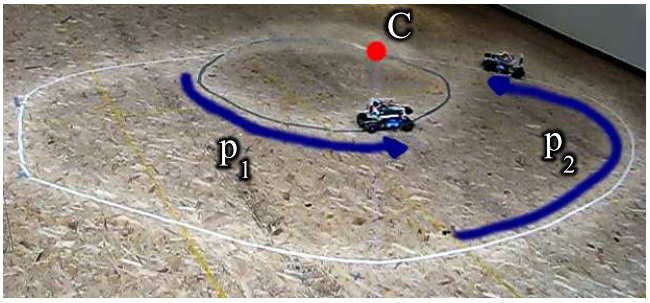


Fig. 5. Vehicles running on the testbed. Their longitudinal displacements with respect to a reference point along the corresponding paths are indicated by p_1 and p_2.

In order to reference vehicle position as a displacement along the path, we project each vehicle's position onto its assigned path (Figure 5). The error introduced by assuming the vehicle is exactly on the path is minimal as long as the steering controller and state estimator work correctly. The resulting longitudinal dynamics of each vehicle along this path is modeled as described in Section III-B.

The collision avoidance algorithm implements the feedback map of equation (1), which establishes what torque command, if any, is required to prevent the vehicles from entering the capture set $\mathcal{C}$. If no special torque command is required to guarantee safety (the last case of map (1) is verified), a cruise control algorithm comes into effect to maintain the vehicle speeds about some set points. For the roundabout implementation, vehicle 1 tracks a speed of 0.4 m/s, while vehicle 2 tracks a speed of 0.5 m/s. A proportional plus derivative (PD) control law is employed for this tracking task. Each vehicle broadcasts its current position coordinates, heading angle, torque command, speed, steering angle, and vehicle number to all vehicles on the local wireless network. Each vehicle then employs the received data in addition to its local data to determine whether the current system configuration is going to be mapped at the next iteration in $\mathcal{C}_{u_B} \cap \mathcal{C}_{u_C}$ if a safe control action is not enforced. We next provide an algorithm for the symbolic computation of the sets $\mathcal{C}_{u_B}$ and $\mathcal{C}_{u_C}$ for a general hybrid automaton $H = H_1||H_2$ given by the parallel composition of two order preserving hybrid automata.

Let $\Delta T > 0$ denote the discretization time and $\bar{x}_i := (x_{i,2}, ..., x_{i,n_i})$. The discrete time version of the dynamics of each order preserving hybrid automaton H_i is given by $x'_{i,1} = x_{i,1} + F_{i,1}(\bar{x}_i, u_i)$, $\bar{x}'_i = \bar{F}_i(\bar{x}_i, u_i)$, where primed variables denote updated variables, $F_{i,1}(\bar{x}_i, u_i) = f_{i,1}(\bar{x}_i, R_i(\bar{x}_i, u_i))\Delta T$, $\bar{F}_i(\bar{x}_i, u_i)$ and $F_{i,1}(\bar{x}_i, u_i)$ are order preserving in the state and in the input, and we have assumed that the $\bar{x}_i$ dynamics do not depend on the $x_{i,1}$ variable. Set $\bar{F}_i^0(\bar{x}_i, u_i) := \bar{x}_i$ and $\bar{F}_i^{k+1}(\bar{x}_i, u_i) := \bar{F}_i(\bar{F}_i^k(\bar{x}_i, u_i), u_i)$ for $k = 0, 1, \ldots$.

Algorithm 1: For each $i \in \{1, 2\}$ and $k \in \mathbb{N}$, let $L_i^k(\bar{x}_i, u_i) = L_i - \sum_{j=0}^{k-1} F_{i,1}(\bar{F}_i^j(\bar{x}_i, u_i), u_i)$, $U_i^k(\bar{x}_i, u_i) = U_i - \sum_{j=0}^{k-1} F_{i,1}(\bar{F}_i^j(\bar{x}_i, u_i), u_i)$.Then, $\mathcal{C}_u = \left\{x \in X \mid \exists\, k \geq 0 \text{ with } L_i^k(\bar{x}_i, u_i) < x_{i,1} < U_i^k(\bar{x}_i, u_i)\ \forall i\right\}$. According to this algorithm, $\mathcal{C}_{u_B}$ and $\mathcal{C}_{u_C}$ are each computed for a given pair of speeds $(\bar{x}_1, \bar{x}_2)$ as a union of rectangles in the position $(x_{1,1}, x_{2,1})$ plane. Checking whether a point x is in $\mathcal{C}_{u_B} \cap \mathcal{C}_{u_C}$ can be performed by simply comparing $(x_{1,1}, x_{2,1})$ against the lower and upper bounds L_i^k, U_i^k (which depend on the values of the speeds $(\bar{x}_1, \bar{x}_2)$) for $i \in \{1, 2\}$ for all k. Also, the sequences $\{L_i^k\}_{k\geq 0}$ and $\{U_i^k\}_{k\geq 0}$ are strictly decreasing due to the increasing property of the flow (property (iii) of Definition 1). Therefore, to check whether $x_{i,1}$ is contained in any of the intervals (L_i^k, U_i^k), it is enough to compute such intervals only until $U_i^k < x_{i,1}$. Hence, only a finite number of such intervals needs to be computed and as a consequence the algorithm that checks whether a configuration is in $\mathcal{C}_{u_B}$ or $\mathcal{C}_{u_C}$ terminates. Furthermore, computation scales with the number of continuous variables.

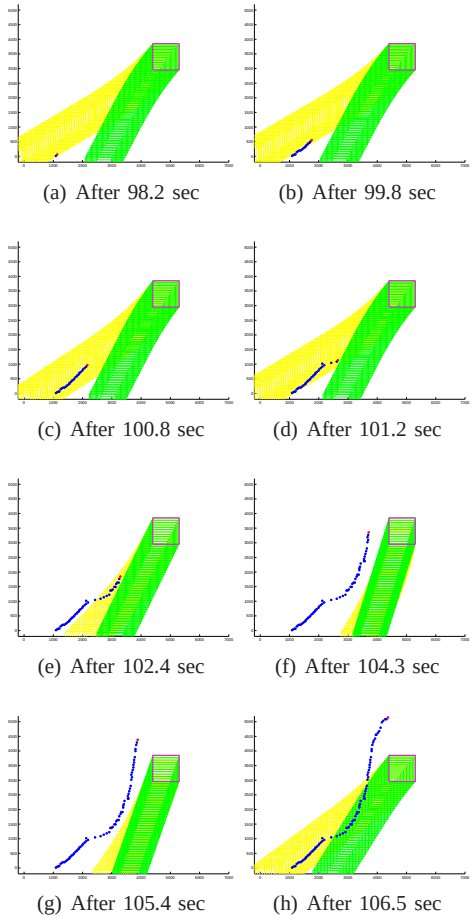
For the Roundabout Drill system, we apply Algorithm 1 with the longitudinal dynamics model from Section III-B. According to such a model, we have that $n_i = 2$ for $i \in \{1, 2\}$, $\bar{x}_i = x_{i,2}$ is the speed of vehicle i, $f_{i,1}(\bar{x}_i, R_i(\bar{x}_i, u_i)) = x_{i,2}$, $\bar{F}_i(\bar{x}_i, u_i) = 0$ if the vehicle is in modes 2 or 3, and $\bar{F}_i(\bar{x}_i, u_i) = x_{i,2} + (a_i\ u_i + b_i)\Delta T$ if the vehicle is in mode 1. In order to take uncertainty on the identified parameters a_i, b_i into account in the computation of the set $\mathcal{C}_{u_B}$ and $\mathcal{C}_{u_C}$, we add a modeling uncertainty to the dynamics of the system in mode 1, that is, $\bar{F}_i(\bar{x}_i, u_i) = x_{i,2} + (a_i\ \ u_i + b_i)\Delta T + \Delta_i$, with $\Delta_i \in [\Delta_{i,m}, \Delta_{i,M}]$. For implementing safety control, this is equivalent to having $\bar{F}_i(\bar{x}_i, u_i) = x_{i,2} + \bar{u}_i \Delta T$, where $\bar{u}_i \in [\bar{u}_{i,m}, \bar{u}_{i,M}]$, $\bar{u}_{i,m} = a_i u_{i,m} + b_i + \Delta_{i,m}$, and $\bar{u}_{i,M} = a_i u_{i,M} + b_i - \Delta_{i,M}$. That is, the uncertainty can be viewed as an adversary that reduces the degree of freedom of the input. The values of $[\Delta_{i,m}, \Delta_{i,M}]$ for vehicles 1 and 2 were selected to be [0.6, 19.1] and [0.85, 24.85], respectively. These values were chosen so as to compensate for the modeling error while not being too conservative. As a consequence, $\mathcal{C}_{u_C}$ is computed with $(\bar{u}_1, \bar{u}_2) = (\bar{u}_{1,M}, \bar{u}_{2,m})$ and $\mathcal{C}_{u_B}$ is computed with $(\bar{u}_1, \bar{u}_2) = (\bar{u}_{1,m}, \bar{u}_{2,M})$.

Ideally, both vehicles should have access to the same data for each iteration, thus resulting in identical computations. Since communication delays and the asynchronous nature of the vehicles may result in up to a three-iteration difference in the data each vehicle is using, it is necessary that the vehicles agree on the same view of the system configuration so they apply the correct control. Therefore, before applying control, the vehicles communicate to each other the case of the control map (1) in which they think the configuration of the system is. The instances in which there is a disagreement are typically due to a system configuration x being in the proximity of the tip of the capture set. Since in the proximity of the tip of the capture set both u_B and u_C are allowed, one of these two inputs is arbitrarily chosen.

E. Experimental results

Six separate experiment runs, each involving 3-4 potential collision scenarios were performed for a total of 23 potential collision scenarios. The vehicles configuration (in $\mathbb{R}^4$) was started at arbitrary locations (but never in the capture set) for each run. Figure 6 shows the slices of the four dimensional sets $\mathcal{C}_{u_B}$ and $\mathcal{C}_{u_C}$ in the position plane corresponding to the current speeds of the vehicles. Note that due to the dynamics of the vehicles, even if vehicle 1 is closer to the conflict point than vehicle 2, the collision avoidance algorithm may decide to let vehicle 2 accelerate to pass first depending on the speeds of the vehicles. The trajectories in Figure 7 indicate that the configuration comes very close to, but never enters, the bad set. Thus the control algorithm not only maintains safety, but it also does so without being conservative. The experiments also show that the vehicles can usually follow their paths without much deviation. Since we project vehicle positions onto the paths, we are even able to maintain safety if the vehicles are within 0.5 m of their projected positions and are moving approximately tangent to the paths.

Among the 23 instances of collision avoidance, only one collision occurred. This was due to a series of bad position measurements that the estimator could not filter before the vehicle configuration reached the capture set.



(a) After 98.2 sec (b) After 99.8 sec

(c) After 100.8 sec (d) After 101.2 sec

(e) After 102.4 sec (f) After 104.3 sec

(g) After 105.4 sec (h) After 106.5 sec

Fig. 6. Experiment data showing the trajectory in the position plane (p_1, p_2) of the vehicles configuration as it approaches a potential collision scenario. The red box is the projection of B in the position plane. In each panel, the green set represents a slice of the four dimensional set $\mathcal{C}_{u_B}$ corresponding to the current vehicles speeds. The yellow set represents a slice of the four dimensional set $\mathcal{C}_{u_C}$ corresponding to the current vehicles speeds. The red dot indicates the current vehicles positions. Control is applied at (d) to avoid the capture set, and the vehicles resume normal operation after passing the bad set (in (g) and (h)). The capture set slices are updated at every iteration on the basis of the vehicles speeds.

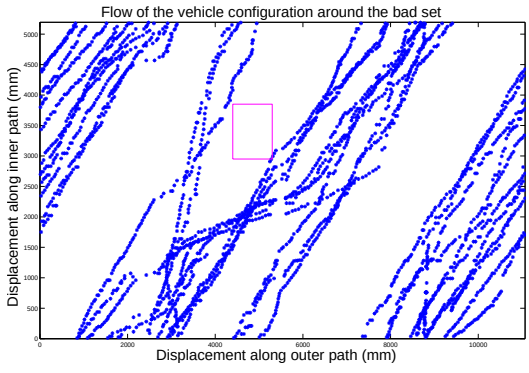


Fig. 7. Roundabout system trajectories for several laps around their paths projected on the position plane. The red box denotes the projection of set B in position plane.

Position and speed information errors deriving from either measurement noise or communication delays occasionally cause the configuration of the system to enter the capture set. Nevertheless, the vehicle configuration quickly exits the capture set and never enters the bad set. Thus, we can conclude that the modeling uncertainty confers enough robustness to the algorithm with respect to these errors. This problem could be prevented by formally accounting for these errors by adding a set-valued estimator to keep track of the resulting state uncertainty.

Though we only considered the two-vehicle collision avoidance case here, we plan to expand this to involve multiple vehicles running simultaneously. This would require the addition of an adaptive cruise control algorithm that maintains safety for the remainder of the trajectories, without disrupting collision avoidance.

IV. CONCLUSIONS

In this paper, we have presented a new computationally efficient control algorithm for the two-vehicle collision avoidance problem as it occurs at mergings on highways and roundabouts. Our algorithm guarantees safety by design and is scalable in the number of continuous variables. We applied the proposed algorithm to an experimental Roundabout Drill, which is collision free and enjoys liveness properties. The experimental results show that (a) the proposed algorithm is well suited for fast real time computation and that (b) the algorithm is robust to uncertainty while not being conservative. In the future, we plan to extend the algorithm to incorporate set-valued state estimators that can formally handle uncertainty deriving from measurement noise and communication delays. Also, we will apply these collision avoidance techniques to a roundabout test-bed with several vehicles and to a full-scale experimental system.

REFERENCES

[1] Car 2 Car Communication Consortium. http://www.car-to-car.org.

[2] Cooperative Intersection Collision Avoidance Systems (CICAS). http://www.its.dot.gov/cicas.

[3] Vehicle Infrastructure Integration Consortium (VIIC). http://www.vehicle-infrastructure.org.

[4] Vehicle Infrastructure Integration (VII). http://www.its.dot.gov/vii.

[5] 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.

[6] D. Del Vecchio. Observer-based control of block triangular discrete time hybrid automata on a partial order. *International Journal of Robust and Nonlinear Control*, page To Appear, 2008.

[7] J. Lygeros, C. J. Tomlin, and S. Sastry. Controllers for reachability specifications for hybrid systems. *Automatica*, 35(3):349–370, 1999.

[8] D. E. Manolakis. Efficient solution and performance analysis of 3-d position estimation by trilateration. *IEEE Transactions on Aerospace and Electronic systems*, 32(4):1239–1248, 1996.

[9] O. Shakernia, G. J. Pappas, and Shankar Sastry. Semi-decidable synthesis for triangular hybrid systems. In *Hybrid Systems: Computation and Control*, Lecture Notes in Computer Science, vol. 2034, M. D. Di Benedetto and A. Sangiovanni-Vincentelli (Eds.), Springer Verlag, 2001.

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

[11] R. Verma, D. Del Vecchio, and H. Fathy. Development of a scaled vehicle with longitudinal dynamics of a HMMWV for an its testbed. *IEEE/ASME Transactions on Mechatronics*, 13:46–57, 2008.