Experimental Testing of Semi-autonomous Multi-vehicle Control for Collision Avoidance at Intersections

Heejin Ahn[1], Andrea Rizzi[2], Alessandro Colombo[3], and Domitilla Del Vecchio[1]

***Abstract*— This paper describes the implementation of a multi-vehicle supervisor to prevent collisions at intersections. The experiments are performed on an intersection testbed consisting of three RC cars. Here, we account for uncertainty in car dynamics and state measurement, and the presence of an uncontrolled car, which is human-driven. The supervisor overrides the controlled cars only when necessary to avoid a possible future collision. From the experiments, we demonstrate that intersection collisions are averted, that is, the cars continuously and safely run on the paths without stopping 92.8% of times.**

I. Introduction

According to the National Highway Traffic Safety Administration (NHTSA), 33,561 people were killed, and 2,362,000 people were injured in road crashes reported in 2012 in the United States. In particular, 27% of the fatalities and 51% of the injuries occurred in crashes at or near intersections [1]. This indicates that one important challenge to traffic safety is to prevent intersection collisions. To reduce the number of road crashes, the government and consortia of automotive companies have been developing a connected vehicle environment [2], [3], [4].

By exploiting the connected vehicle environment, characterized by vehicle-to-vehicle and vehicle-to-infrastructure communication, many researches have suggested the design of controllers to maintain road safety. A global coordination scheme of fully autonomous vehicles was introduced in a model predictive control framework [5], [6] and utilizing time-slot assignment [7]. In more practical scenarios, where full autonomy is not an option, a controller should continuously verify the safety of the system until override is necessary. In order to verify safety, a path-planning approach has been used [8], [9], [10], which computes the possible paths of a vehicle and checks whether the paths collide with an obstacle or another vehicle. Since this approach is mostly concerned with full vehicle dynamics, it has computational difficulties in dealing with collisions that involve a large number of vehicles, such as intersection collisions. To address this problem of scalability, scheduling has been studied using only the longitudinal dynamics of vehicles [11], [12], [13]. Supervisory control theory based on discrete event systems [14], [15] has also been used, where system abstractions are exploited to simplify the verification of safety [16], [17]. However, those results required heavy computation and were applicable only to first-order dynamics of vehicles.

In this paper, we consider a semi-autonomous controller, called a supervisor, that employs a scheduling approach to maintain intersection safety. When a possible future collision is detected, the supervisor overrides cars to guarantee safety regardless of uncertainty in the dynamic model of the cars and state measurement. This supervisor is validated through experiments performed on an intersection testbed composed of three RC cars. One of the cars is human-driven and "uncontrolled" by the supervisor. The other cars are controlled by on-board computers, which drive them at a constant speed, until the supervisor intervenes to avoid collisions. This work extends the studies of [18] and [19], where a collision avoidance algorithm that is applicable only to two cars at the time was tested. Also, this work merges the designs of the supervisors of [12] where the presence of uncontrolled vehicles was considered, and of [13] where modeling and measurement uncertainty was considered. For the first time, those results are tested in an experimental setting.

This paper is structured as follows. In Section II, we describe an intersection testbed and the dynamic model of RC cars. Then, the design of a supervisor is illustrated in Section III, where three functions that constitute the supervisor are introduced. In Section IV, experimental setups and results are described. Then, Section V concludes the paper with suggestions on future work.

II. System model

A. An intersection testbed

We consider the intersection testbed in Figure 1 consisting of three RC cars. Each car runs on a different prescribed path. The paths intersect at one conflict point, and the area around the conflict point is an intersection, which is represented by the shaded area in Figure 1a. If at least two cars are inside the intersection at the same time, we consider that a collision occurs. The objective of the supervisor is to make the cars cross the intersection without any collisions without ever stopping, while intervening only when necessary.

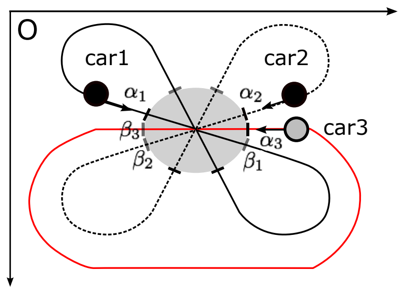
The supervisor overrides the controlled cars only when it detects possible future collisions. Otherwise, the controlled cars are programmed to maintain a constant motor input, called a desired input. In the experiments, car 1 and car 2

This work was in part supported by NSF Award #1239182.

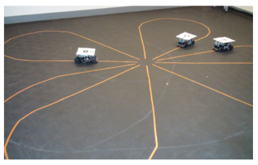
[1]Heejin Ahn and Domitilla Del Vecchio are with the Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA. Email: `hjahn@mit.edu` and `ddv@mit.edu`

[2]Andrea Rizzi is with the Department of Biological Statistics and Computational Biology, Cornell University, 1198 Comstock Hall, Ithaca, NY 14853, USA. Email: `ar963@cornell.edu`

[3]Alessandro Colombo is with the Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, via Ponzio 34/5, 20133, Milano, Italy. Email: `alessandro.colombo@polimi.it`



(a)



(b)

Fig. 1: (a) Three prescribed paths on which three RC cars run. The paths are defined as a sequence of points in the global coordinates with an origin O. The shaded area in the middle is the intersection, represented on each path by an interval (α_i, β_i) for $i = \{1, 2, 3\}$. The small black circles represent two controlled cars while the small grey circle represents the uncontrolled car. (b) The intersection testbed. The orange lines on the ground correspond to the prescribed paths in (a).

are controllable by the supervisor, while car 3 is driven by a human and not controllable by the supervisor.

B. Dynamic model of a RC car

The cars are programmed to adjust steering to follow the specified path [18]. To prevent collisions, we consider only the longitudinal dynamics of the car along its path. To describe the longitudinal dynamics of car i, we introduce the state $x_i = (y_i, v_i)$ where y_i and v_i are position and speed along the path, respectively. Let u_i denote the input to the car. With model uncertainties $d_{y,i}$ and $d_{v,i}$, the dynamic model can be written as follows:

$$\begin{aligned} \dot{y}_i &= v_i + d_{y,i}, \\ \dot{v}_i &= a_i v_i + b_i + f_i u_i + d_{v,i}, \end{aligned} \tag{1}$$

where a_i and b_i are model parameters, and f_i is a motor gain [20]. Here, we assume $\dot{y}(t) > 0$ for all t so that the cars do not reverse or stop. Since $f_i > 0$, the model has the following monotonicity property. If $u_i \leq u_i'$, then $y_i(t, u_i, d_i, x_i(0)) \leq y_i(t, u_i', d_i, x_i(0))$ for all $t \geq 0$ where $y_i(t, u_i, d_i, x_i(0))$ is the output at time t starting from an initial condition $x_i(0)$ with an input u_i and disturbance $d_i := (d_{y,i}, d_{v,i})$. Similarly, if $d_{y,i} \leq d'_{y,i}$ and $d_{v,i} \leq d'_{v,i}$, then $y_i(t, u_i, d_i, x_i(0)) \leq y_i(t, u_i, d_i', x_i(0))$ for all $t \geq 0$. If $x_i(0) \leq x_i'(0)$, then $y_i(t, u_i, d_i, x_i(0)) \leq y_i(t, u_i, d_i, x_i'(0))$ for all $t \geq 0$. The state $x_i(t, u_i, d_i, x_i(0))$ also follows the same notation and satisfies the same monotonicity property. The aggregate state is denoted by $\mathbf{x} := (x_1, x_2, x_3)$ and the aggregate output is $\mathbf{y} := (y_1, y_2, y_3) \in Y \subset \mathbb{R}^3$ where Y is the output space.

In addition, we assume that speed is bounded, that is, $v_i \in [v_{i,min}, v_{i,max}]$. The input signal u_i is also bounded, that is, $u_i(t) \in \mathcal{U}_i := [u_{i,min}, u_{i,max}]$ for all t. Similarly, the disturbance signal d_i is bounded, that is, $d_{y,i}(t) \in [d_{y,i,min}, d_{y,i,max}]$ and $d_{v,i}(t) \in [d_{v,i,min}, d_{v,i,max}]$ for all t.

The state measurement is subject to measurement uncertainty $\delta_i := (\delta_{y,i}, \delta_{v,i})$. That is, the state measurement $x_{m,i} := (y_{m,i}, v_{m,i})$ is as follows:

$$y_{m,i} = y_i + \delta_{y,i}, \qquad v_{m,i} = v_i + \delta_{v,i},$$

where $y_{m,i}$ is the position measurement and $v_{m,i}$ is the speed measurement. The measurement uncertainty δ_i is bounded, that is, $\delta_i \in [\delta_{i,min}, \delta_{i,max}]$, where $\delta_{i,min} := (\delta_{y,i,min}, \delta_{v,i,min})$ and $\delta_{i,max} := (\delta_{y,i,max}, \delta_{v,i,max})$. In Section IV-B, we will show several factors that contribute to the uncertainties.

III. Supervisor Design

The supervisor is executed in discrete time τ and has the high level structure shown in Figure 2. Each function of the

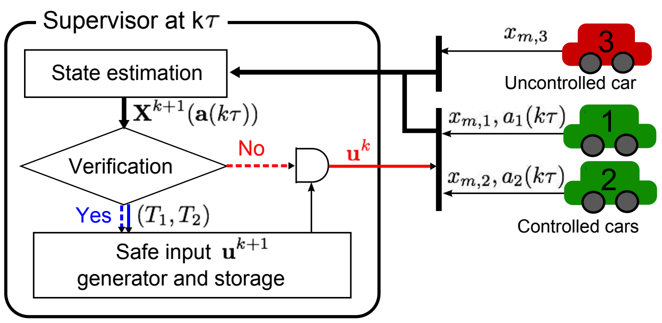


Fig. 2: The structure of a supervisor. At step k, the supervisor receives a state measurement $\mathbf{x}_m := (x_{m,1}, x_{m,2}, x_{m,3})$ and a desired input $\mathbf{a}(k\tau) := (a_1(k\tau), a_2(k\tau))$ from the controlled cars. These are used to compute a one step ahead state prediction $\mathbf{X}^{k+1}(\mathbf{a}(k\tau))$. Then, the verification function determines whether there exists a safe input signal for this state prediction to avoid future collisions. If the answer is "yes" with a schedule (T_1, T_2), a safe input signal $\mathbf{u}^{k+1}$ is generated and stored for possible use at the following step, and the supervisor does not override the cars. Otherwise, the supervisor overrides the controlled cars using the safe input $\mathbf{u}^k$, which was computed and stored at the previous step.

supervisor is introduced in Section III-A, III-B, III-C, and the formal integration of these functions is provided in Section III-D.

A. State estimation

The state estimation is updated through a prediction-correction scheme as follows. Suppose that at $t = (k-1)\tau$ the state $x_i(t)$ of car i is known to be in the interval $[x_i^a((k-1)\tau), x_i^b((k-1)\tau)]$. Also, suppose that an input $u_i(t)$ is applied for $t \in [(k-1)\tau, k\tau)$. Then, the state prediction at time $k\tau$ is a set $X_i^k(u_i) \subset \mathbb{R}^2$ that is in the form of an interval $X_i^k(u_i) = [\min(X_i^k(u_i)), \max(X_i^k(u_i))]$ with the following lower and upper bounds, for $i \in \{1, 2\}$,

$$\begin{aligned} \min(X_i^k(u_i)) &:= x_i(\tau, u_i, d_{i,min}, x_i^a(k\tau)), \\ \max(X_i^k(u_i)) &:= x_i(\tau, u_i, d_{i,max}, x_i^b(k\tau)). \end{aligned} \tag{2}$$

The same definitions hold for the uncontrolled car 3, except that the input can take any values between $u_{3,min}$ and $u_{3,max}$:

$$\begin{aligned}\min(X_3^k(\cdot)) &:= x_3(\tau, u_{3,min}, d_{3,min}, x_3^a(k\tau)),\\ \max(X_3^k(\cdot)) &:= x_3(\tau, u_{3,max}, d_{3,max}, x_3^b(k\tau)).\end{aligned} \tag{3}$$

At $t = k\tau$, this state prediction is corrected using the measurement $x_{m,i}(k\tau)$ for all $i \in \{1, 2, 3\}$. The resulting state estimation at time $k\tau$ is a set with lower and upper bounds as follows:

$$\begin{aligned}x_i^a(k\tau) &= \max(x_{m,i}(k\tau) + \delta_{i,min}, \min(X_i^k(u_i))),\\ x_i^b(k\tau) &= \min(x_{m,i}(k\tau) + \delta_{i,max}, \max(X_i^k(u_i))).\end{aligned} \tag{4}$$

If $k = 0$, let $x_i^a(0) = x_{m,i}(0)+\delta_{i,min}$ and $x_i^b(0) = x_{m,i}(0)+\delta_{i,max}$. The aggregate state prediction $\mathbf{X}^{k+1}(\mathbf{a}(k\tau)) \subset \mathbb{R}^6$ with a desired input $\mathbf{a}(k\tau)$ is used in the verification function (Figure 2).

B. Verification of safety

Let (α_i, β_i) denote the intersection on the path of car i as shown in Figure 1a. The output configurations corresponding to collision is called *Bad set* $B \subset Y$. The Bad set is defined as $B := \{\mathbf{y} \in Y : y_i \in (\alpha_i, \beta_i)$ and $y_j \in (\alpha_j, \beta_j)$ for some $i \neq j \in \{1, 2, 3\}\}$.

Let $\mathbf{x}(0) \in [\mathbf{x}^a(0), \mathbf{x}^b(0)]$. Then, the verification problem is as follows:

Problem 1: Given a set of initial conditions $[\mathbf{x}^a(0), \mathbf{x}^b(0)]$, determine if there exists an input signal $\mathbf{u}$ that guarantees $\mathbf{y}(t, \mathbf{u}, \mathbf{d}, \mathbf{x}(0)) \notin B$ for all $\mathbf{x}(0) \in [\mathbf{x}^a(0), \mathbf{x}^b(0)]$, for all $\mathbf{d}$, for all $t \geq 0$.

To address Problem 1, we employ an equivalent Inserted Idle-time Scheduling problem, which is formulated and solved in [12]. In this problem, one has to determine whether the intersection can be crossed by at most one car at the time. In this paper, we redefine scheduling variables to account for uncertainty, which were not considered in [12].

Definition 1: Given a set of initial conditions $[\mathbf{x}^a(0), \mathbf{x}^b(0)]$, release times R_i, deadlines D_i, and process times $P_i(T_i)$ are defined as follows. For $i \in \{1, 2\}$,

$$\begin{aligned}R_i &:= \{t \geq 0 : y_i(t, u_{i,max}, d_{i,max}, x_i^b(0)) = \alpha_i\},\\ D_i &:= \{t \geq 0 : y_i(t, u_{i,min}, d_{i,max}, x_i^b(0)) = \alpha_i\}.\end{aligned}$$

Given a real non-negative number T_i,

$$\begin{aligned}P_i(T_i) := \min_{u_i \in \mathcal{U}_i} \{&t \geq 0 : y_i(t, u_i, d_{i,min}, x_i^a(0)) = \beta_i\\ &\text{with constraint } y_i(T_i, u_i, d_{i,max}, x_i^b(0)) = \alpha_i\}.\end{aligned}$$

If $y_i^a(0) \geq \beta_i$, then set $R_i = 0, D_i = 0$, and $P_i(T_i) = 0$. If $y_i^b(0) \geq \alpha_i$, then $R_i = 0, D_i = 0$, and $P_i(T_i) = \{t : y_i(t, u_{i,max}, d_{i,min}, x_i^a(0)) = \beta_i\}$. If the constraint is not satisfied, set $P_i(T_i) = \infty$. For the uncontrolled car 3, idle-time $(\bar{R}_3, \bar{P}_3)$ is defined as follows.

$$\begin{aligned}\bar{R}_3 &:= \{t \geq 0 : y_3(t, u_{3,max}, d_{3,max}, x_3^b(0)) = \alpha_3\},\\ \bar{P}_3 &:= \{t \geq 0 : y_3(t, u_{3,min}, d_{3,min}, x_3^a(0)) = \beta_3\}.\end{aligned}$$

If $y_3^a(0) \geq \beta_3$, set $\bar{R}_3 = 0$ and $\bar{P}_3 = 0$. If $y_3^b(0) \geq \alpha_3$, set $\bar{R}_3 = 0$.

In this definition, the release time R_i and the deadline D_i are the soonest and the latest times at which a controlled car i can enter the intersection for any d_i for any $x_i(0) \in [x_i^a(0), x_i^b(0)]$, respectively. Provided that a controlled car i enters the intersection no earlier than T_i, the process time $P_i(T_i)$ is the soonest time at which the car can exit the intersection. The idle-time is the time at which the uncontrolled car 3 is possibly inside the intersection for some u_3, d_3, and $x_3(0) \in [x_3^a(0), x_3^b(0)]$. The Inserted Idle-Time Scheduling problem with uncertainty is formulated as follows.

Problem 2: Given a set of initial conditions $[\mathbf{x}^a(0), \mathbf{x}^b(0)]$, determine whether there exists a schedule $\mathbf{T} = (T_1, T_2) \in \mathbb{R}_+^2$ such that for all $i \in \{1, 2\}$,

$$R_i \leq T_i \leq D_i, \tag{5}$$

$$(T_1, P_1(T_1)) \cap (T_2, P_2(T_2)) = \emptyset, \tag{6}$$

$$(T_i, P_i(T_i)) \cap (\bar{R}_3, \bar{P}_3) = \emptyset. \tag{7}$$

In this problem, the schedule T_i represents the time at which car i enters the intersection for some u_i, d_i and $x_i(0) \in [x_i^a(0), x_i^b(0)]$. Then, each condition has an intuitive meaning. Condition (5) comes from the fact that the input is bounded. Condition (6) means that the times at which car 1 and car 2 are inside the intersection do not overlap, which guarantees collision avoidance between the controlled cars. Condition (7) implies that a controlled car i and car 3 never meet inside the intersection for any u_3 and d_3. This condition guarantees that collision between the controlled car and the uncontrolled car is avoided. Based on these meanings, if there exists a schedule satisfying the conditions, we can find a safe input signal that makes the cars avoid collisions. Thus, by solving Problem 2, we can address Problem 1.

We adopt the algorithm given in [12] to solve Problem 2 because the algorithm will be the same except for the fact that the scheduling variables are defined as given above. We call the algorithm $\textsc{Scheduling}([\mathbf{x}^a(0), \mathbf{x}^b(0)])$, whose output is either $(\mathbf{T}, yes)$ or $(\cdot, no)$, where $\mathbf{T} \in \mathbb{R}_+^2$ is a schedule satisfying the above conditions if it exists.

C. Safe input

If the verification problem has a positive answer at $t = k\tau$, then the supervisor generates a safe input signal and stores it for potential use at the next step. In particular, given a schedule $\mathbf{T} = (T_1, T_2)$ and a set of initial conditions $[\mathbf{x}^a(0), \mathbf{x}^b(0)]$, a safe input signal $u_i^{k+1,\infty}$ for $i \in \{1, 2\}$ is defined as follows:

$$\begin{aligned}u_i^{k+1,\infty} &:= \sigma_i([x_i^a(0), x_i^b(0)], T_i)\\ &\in \{u_i : y_i(P_i(T_i), u_i, d_{i,min}, x_i^a(0)) = \beta_i\\ &\qquad \text{and } y_i(T_i, u_i, d_{i,max}, x_i^b(0)) = \alpha_i\}.\end{aligned}$$

That is, a safe input signal $u_i^{k+1,\infty}$ is one of the input signals that make the car enter the intersection no earlier than T_i and make it exit no later than $P_i(T_i)$ for any d_i for any $x_i(0) \in [x_i^a(0), x_i^b(0)]$. This input signal is used at step $k + 1$ to override the controlled cars if necessary to avoid a collision as described in the next section.

D. Supervisor

In this section, we combine the results of Section III-B and III-C to provide an algorithm to implement the supervisor, which is informally described in Figure 2.

Algorithm 1 Implementation of the supervisor

```
1: procedure SUPERVISOR($\mathbf{x}_m, \mathbf{a}(k\tau)$)
2: $(\mathbf{T}_1, answer_1) \leftarrow$ SCHEDULING($\mathbf{X}^{k+1}(\mathbf{a}(k\tau))$)
3: if $answer_1 = yes$ then
4:     $\mathbf{u}^{k+1,\infty} \leftarrow \sigma(\mathbf{X}^{k+1}(\mathbf{a}(k\tau)), \mathbf{T}_1)$
5:     $\mathbf{u}^{k+1}(t) \leftarrow \mathbf{u}^{k+1,\infty}(t)$ for $t \in [(k+1)\tau, (k+2)\tau)$
6:     return $\mathbf{a}(k\tau)$
7: else
8:     $(\mathbf{T}_2, answer_2) \leftarrow$ SCHEDULING($\mathbf{X}^{k+1}(\mathbf{u}^k)$)
9:     $\mathbf{u}^{k+1,\infty} \leftarrow \sigma(\mathbf{X}^{k+1}(\mathbf{u}^k), \mathbf{T}_2)$
10:    $\mathbf{u}^{k+1}(t) \leftarrow \mathbf{u}^{k+1,\infty}(t)$ for $t \in [(k+1)\tau, (k+2)\tau)$
11:    return $\mathbf{u}^k$
```

At $t = k\tau$, the supervisor in Algorithm 1 receives a state measurement $\mathbf{x}_m$ and a desired input $\mathbf{a}(k\tau)$, which are used to compute the state prediction $\mathbf{X}^{k+1}(\mathbf{a}(k\tau))$ as defined in (2) and (3). The algorithm SCHEDULING($\mathbf{X}^{k+1}(\mathbf{a}(k\tau))$) solves the Inserted Idle-Time Scheduling problem (Problem 2) given the set of initial conditions $\mathbf{X}^{k+1}(\mathbf{a}(k\tau))$. The scheduling variables in Definition 1 are computed by the Newton-Raphson method [21].

If $answer_1$ in line 2 of Algorithm 1 is "yes", then a safe input $\mathbf{u}^{k+1,\infty}$ is generated based on a schedule $\mathbf{T}_1$. In line 5, the input $\mathbf{u}^{k+1}(t)$, which is the truncated safe input signal $\mathbf{u}^{k+1,\infty}$ defined for $t \in [(k+1)\tau, (k+2)\tau)$, is stored for potential use at the following step. In this case, the supervisor returns $\mathbf{a}(k\tau)$, that is, it allows the controlled cars to run with their desired inputs.

If $answer_1$ is "no", the safe input signal $\mathbf{u}^k$, which was generated at the previous step, is used to compute another state prediction $\mathbf{X}^{k+1}(\mathbf{u}^k)$. Since the input $\mathbf{u}^k$ is a safe input, $answer_2$ in line 8 is always "yes". Then, a safe input signal $\mathbf{u}^{k+1,\infty}$ is generated using $\mathbf{T}_2$ and stored for potential use at the following step. The supervisor returns the safe input $\mathbf{u}^k$, which means that the supervisor overrides the controlled cars using $\mathbf{u}^k$.

Thus, the supervisor is designed such that it overrides the controlled cars only when using the desired input will lead to an unavoidable future collision.

IV. EXPERIMENTS

A. Experimental setups

The supervisor is tested on an intersection testbed using three RC cars as shown in Figure 1. The size of the testbed is $6m \times 6m$, and the length of the RC car is $30cm$. A Tamiya scaled RC car chassis is used. The motor and the steering servo are controlled by a microcontroller (Acroname Moto 1.0) through a power amplifier (Acroname 3A Back EMF H-bridge). Motor and steering inputs are commanded by an on-board computer (Mini ITX running Linux, Fedora Core) mounted on the top of each RC car [20]. Both Algorithm 1 and a path following algorithm are implemented on the on-board computer.

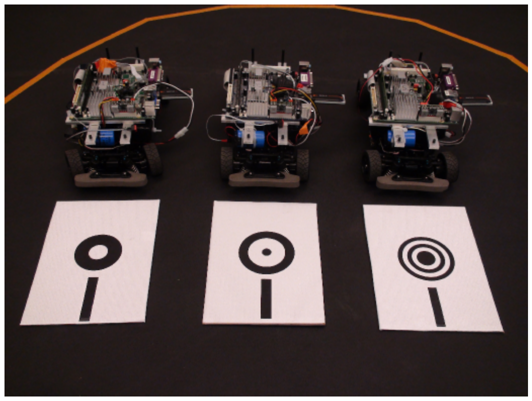


Fig. 3: Three RC cars that are used in the experiments. Each car is recognized with its symbol by a vision system.

Two batteries (Tenergy Li-Ion 14.8V 4400mAh) provide power to the car. A power relay (Omron G5SB) is connected to a capacitor (Aluminium Electrolytic Capacitor 12000uF 25volts) and the two batteries. The batteries are connected to the power amplifier through a switch. Because of this power connection, the motor input shows a first-order dynamic behavior, which will be considered as a disturbance in Section IV-B.

An over-head vision system is used to measure the position and direction of all cars. The vision system consists of six cameras on the ceiling and three computers that process the images taken by the cameras. The position and direction measurements are extracted from the images by recognizing the symbols on the top of each car (see Figure 3). A quadrature encoder mounted on the rear axle of the wheels measures the speed.

A computer collects all the available information including the positions and directions of all cars, which are measured by the vision system, and the local information of each controlled car, such as the speed, the desired input, and the model parameters. This information is then distributed to the controlled cars through a 802.11b wireless communication network.

All cars follow the prescribed paths in Figure 1a by implementing a feedback controller for the steering input using information from the vision system. The supervisor can override the motor input, but cannot change the steering input.

B. Modeling of car dynamics

The block diagram of the dynamic model (1) is shown is Figure 4, where $d_{y,i} = d_{proj,i}$ and $d_{v,i} = d_{power,i} + d_{steer,i} + d_{slope,i}$ for all $i \in \{1, 2, 3\}$. In this section, we explain the origin of the disturbances, and introduce a compensator in order to reduce their effects.

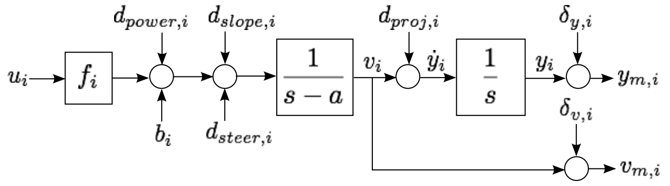


Fig. 4: The block diagram of the dynamic model (1). The model disturbances are $d_{proj,i}, d_{power,i}, d_{steer,i}$, and $d_{slope,i}$, and the measurement errors are $\delta_{y,i}$ and $\delta_{v,i}$.

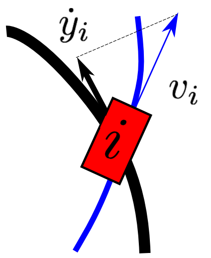


Fig. 5: Car i does not perfectly follow the prescribed path (thick black line), which causes the uncertainty $d_{proj,i} = \dot{y}_i - v_i$.

The disturbance $d_{proj,i}$ is due to the fact that the cars do not perfectly follow the prescribed paths as shown in Figure 5. Here, v_i is the longitudinal speed along the actual path that the car follows (thin blue line) while $\dot{y}_i$ is the speed along the prescribed path (thick black line). When the vehicle axis is not aligned with the prescribed path, $\dot{y}_i \neq v_i$. The disturbance $d_{proj,i}(t)$ quantifies such a difference.

The power connection, which includes the capacitor, the power relay, the batteries, and the power amplifier, introduces additional dynamics, which we model by the time-varying disturbance $d_{power,i}$. In particular, we take the form $d_{power,i} = g_i e^{-t/h_i} u_i$, where g_i is a gain, h_i is a time constant, and u_i is the motor input. This form is justified by the experimental data acquired when car i runs in circles with a constant input (see [22] for the data). The parameters g_i and h_i are also determined using those data by finding the best fitted values. The motor gain f_i is estimated on-line using "MIT rule"[23] because it can be time-varying depending on the battery charge level. This on-line estimation is executed using the speed measurement $v_{m,i}$ before Algorithm 1.

The disturbance $d_{steer,i}$ is due to the fact that the steering input affects, to some extent, the speed of the four-wheeled car [24]. The fact that the testbed is not completely flat appears as the disturbance $d_{slope,i}$. The disturbance $d_{steer,i} + d_{slope,i}$ is position-dependent because the slope of the testbed and the steering input are about the same at the same point of the prescribed path. This disturbance is estimated off-line from data acquired when the car runs on the path with a constant input.

To reduce the effect of these disturbances, we introduce a compensating input $c_i(t, y_i)$ such that $u_i = u'_i + c_i(t, y_i)$ where u'_i is the input that the supervisor returns. This compensating input $c_i(t, y_i)$ is defined as follows:

$$c_i(t, y_i) = -\frac{g_i e^{-t/h_i} + d_{steer,i}(y_i) + d_{slope,i}(y_i)}{g_i e^{-t/h_i} + f_i}.$$

Thus, the dynamic model becomes

$$\dot{v}_i = a_i v_i + b_i + f_i u'_i + g_i e^{-t/h_i}(u'_i - 1),$$

in which we have assumed that the error due to the estimation of $g_i, h_i, f_i, d_{steer,i}$, and $d_{slope,i}$ is small enough to be negligible. With this assumption, the effective disturbance after the compensation becomes $d'_{v,i} = g_i e^{-t/h_i}(u'_i - 1)$.

The design of the supervisor assumes that the disturbances $d_{y,i}$ and $d'_{v,i}$ are bounded. However, the empirical distribution of the disturbances shows a quite long tail. We chose to ignore the tail and take tighter bounds.

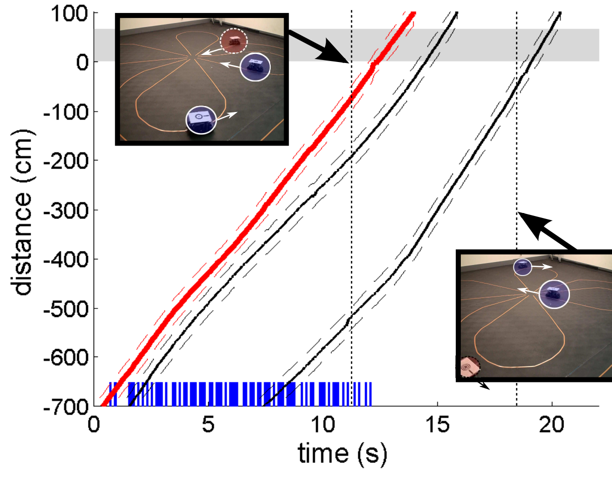
C. Experimental results

Algorithm 1 is implemented on each car. The model parameters are $a = (-0.53, -0.30, -0.43)/s$, $b = (-84.68, -66.43, -49.64)cm/s^2$, $g = (0.99, 0.44, 0.95)$, and $h = (9.17, 6.84, 5.95)s$. These are the best fitted parameters for the data of each car running in circles in advance of the implementations of the supervisor. The motor gain $f_i(t)$ is time-varying for all i. As a reference, the average of each gain is $\bar{f}_1 = 9.07$, $\bar{f}_2 = 7.02$, and $\bar{f}_3 = 5.93$.

We find the bounds for the measurement errors by manually measuring the position and comparing it with the measurement data: $\delta_{y,i,min} = -25cm$, $\delta_{y,i,max} = 25cm$, $\delta_{v,i,min} = -25cm/s$, and $\delta_{v,i,max} = 16cm/s$ for all i. The experimental disturbance bound of $d_{y,i}$ is $\pm 30cm/s$, and that of $d_{v,i}$ is $\pm 35cm/s^2$. These large bounds are due to the long tail of the disturbance distribution as mentioned in Section IV-B. In the experiments, we use tighter bounds, which are chosen as $d'_{y,min} = (-5, -3, -3)cm/s$, $d'_{y,max} = (3, 4, 3)cm/s$, $d'_{v,min} = (-4, -2, -3)cm/s^2$, and $d'_{v,max} = (2, 3, 2.5)cm/s^2$. With these tighter bounds, we still obtain a 92.8% success rate.

The total length of the paths in Figure 1a is $22m$ for the 8-figure paths for car 1 and 2 and $14m$ for the 0-figure path for car 3. Because the position is reset after a car exits the intersection, we consider the half of the length of the 8-figure path, which is $11m$. The intersection is located at $(906.5, 971.5)cm \times (911.5, 976.5)cm \times (1241.2, 1306.2)cm$. The speed is bounded with $v_{min} = (10.5, 10.5, 13)cm/s$ and $v_{max} = (17, 16.5, 15)cm/s$, and the motor input (PWM) is bounded with $u_{min} = (105, 105, 130)$ and $u_{max} = (170, 165, 150)$, whose maximum output is 250. The time interval τ is $0.1s$.

In Figure 6a, the uncontrolled car (thick red line, or dotted red circle in the pictures) approaches the intersection (shaded area) earlier than the others with slow speed. The supervisor overrides the controlled cars (thin black lines, or solid blue circles) to decelerate them ($\sim 11s$, intermittently). In the case of Figure 6b, the supervisor overrides the controlled cars for a shorter time. Notice that the last car's position upper bound enters the intersection right after the uncontrolled car's position lower bound has exited, indicating that the override



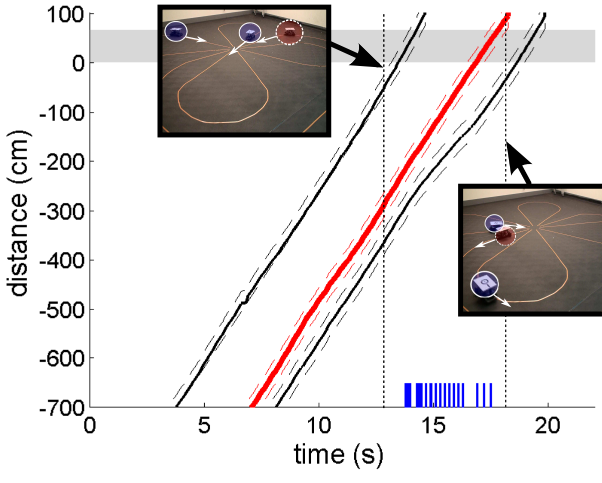


Fig. 6: The results of collision avoidance. The intersection is the shaded region located at $(0, 65)cm$. The thin black lines represent the controlled cars, car 1 and car 2, and the thick red line represents the uncontrolled car, car 3. The dotted lines are the lower and upper bounds of the state estimation defined in (4). They are updated using measurements every $0.1s$. The dotted vertical lines relate the graphs with the corresponding pictures of the actual experiments. In the pictures, the dotted red circle is the uncontrolled car, and the solid blue circles are the controlled cars, where the arrows show the directions of the cars' speeds. The blue boxes on the x-axis represent the times when the supervisor intervenes. These demonstrate that the cars are not inside the intersection at the same time.

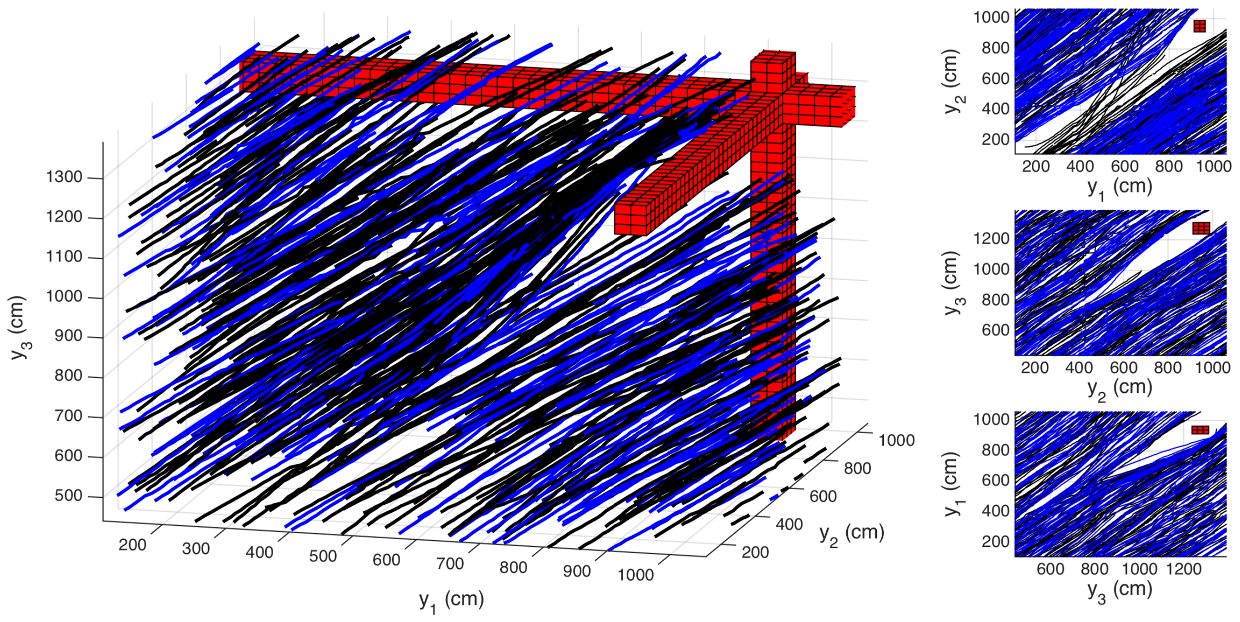


Fig. 7: Trajectories near the Bad set in the 3D output space. The red blocks are the Bad set, and the black lines are the experimental data of crossing the intersection. The blue lines represent the trajectories in which the supervisor overrides the controlled cars for at least one step. The three figures on the right-hand side are 2D projections, which confirm that no two cars are ever inside the intersection (red boxes) at the same time.

was applied because deemed necessary to avoid the collision. In both cases, intersection collisions are averted.

Figure 7 shows the trajectories of the positions of each car and the Bad set in the 3D output space. The figures on the right-hand side are the trajectories projected on 2D planes, which confirm that no trajectory enters the Bad set (red boxes). The 3D graph contains 591 trajectories that include at least one car crossing the intersection, among which 208 trajectories are overridden by the supervisor for at least one time step.

In a small number of instances, the system is subject to a disturbance outside of the tighter bounds, and as a consequence, the state prediction becomes incorrect. If $answer_2$ in line 8 of Algorithm 1 is "no" because of the incorrect prediction, then the controlled cars are programmed to stop. This occurs only 7.2% of times. The results are summarized in Table I.

TABLE I: Summary of the experiments.

Trajectories of at least one car crossing the intersection	Trajectories where the supervisor intervenes	Collision	Stop
591	208	0	15

V. Conclusions

We have implemented a supervisor that overrides controlled cars only when future collisions would otherwise become unavoidable. This supervisor accounts for 1) modeling and measurement uncertainty, and 2) the presence of an uncontrolled vehicle. In particular, the inclusion of these uncertainty sources in the algorithm allows us to implement the system on an experimental platform. The supervisor is implemented on an intersection testbed using three RC cars. Here, two of the cars are cooperating with each other and execute the supervisor to actively prevent collisions. From the experiments, we demonstrated that collisions are averted without substantial conservatism.

While this supervisor is validated through experiments, several theorems that support the design, such as the nonblocking property of the supervisor, must still be provided as in [25], [13], [12]. In addition, while the architecture presented here prevents collisions at one conflict point, future work will consider multiple conflict points such as found in real traffic intersections. As the intersections are modeled in more detail, the challenge is to overcome significantly increasing computational complexity. Also, since the routes of vehicles may not be known in advance, a supervisor will have to consider all possible routes of vehicles and progressively discard infeasible ones.

References

[1] National Highway Traffic Safety Administration, U.S. Department of Transportation, "Traffic Safety Facts 2012," http://www-nrd.nhtsa.dot.gov/Pubs/812032.pdf, 2012.

[2] U.S. Department of Transportation, "Connected Vehicle Technology," http://www.its.dot.gov/landing/cv.htm, 2015.

[3] "Vehicle Infrastructure Integration Consortium," http://www.vehicle-infrastructure.org/, 2010.

[4] Crash Avoidance Metrics Partnership, "Driver Worklad Metrics Task 2 Final Report," http://www.nhtsa.gov/Research/Crash+Avoidance/Office+of+Crash+Avoidance+Research+Technical+Publications, 2006.

[5] M. Kamal, J. Imura, T. Hayakawa, A. Ohata, and K. Aihara, "A Vehicle-Intersection Coordination Scheme for Smooth Flows of Traffic Without Using Traffic Lights," *IEEE Transactions on Intelligent Transportation Systems*, 2014.

[6] K.-D. Kim and P. Kumar, "An MPC-Based Approach to Provable System-Wide Safety and Liveness of Autonomous Ground Traffic," *IEEE Transactions on Automatic Control*, vol. 59, no. 12, pp. 3341–3356, Dec. 2014.

[7] H. Kowshik, D. Caveney, and P. Kumar, "Provable Systemwide Safety in Intelligent Intersections," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 3, pp. 804–818, 2011.

[8] G. Aoude, B. Luders, K. Lee, D. Levine, and J. How, "Threat assessment design for driver assistance system at intersections," in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, Sep. 2010, pp. 1855–1862.

[9] A. Gray, Y. Gao, T. Lin, J. Hedrick, H. Tseng, and F. Borrelli, "Predictive control for agile semi-autonomous ground vehicles using motion primitives," in *American Control Conference (ACC), 2012*, Jun. 2012, pp. 4239–4244.

[10] J. Bialkowski, S. Karaman, M. Otte, and E. Frazzoli, "Efficient Collision Checking in Sampling-Based Motion Planning," in *Algorithmic Foundations of Robotics X*, ser. Springer Tracts in Advanced Robotics. Springer Berlin/Heidelberg, 2013, no. 86, pp. 365–380.

[11] A. Colombo and D. Del Vecchio, "Least Restrictive Supervisors for Intersection Collision Avoidance: A Scheduling Approach," *IEEE Transactions on Automatic Control*, 2014.

[12] H. Ahn, A. Colombo, and D. Del Vecchio, "Supervisory control for intersection collision avoidance in the presence of uncontrolled vehicles," in *American Control Conference (ACC)*, Jun. 2014, pp. 867–873.

[13] L. Bruni, A. Colombo, and D. Del Vecchio, "Robust multi-agent collision avoidance through scheduling," in *IEEE Conference on Decision and Control (CDC)*, Dec. 2013, pp. 3944–3950.

[14] P. J. Ramadge and W. M. Wonham, "Supervisory Control of a Class of Discrete Event Processes," *SIAM Journal on Control and Optimization*, vol. 25, no. 1, pp. 206–230, Jan. 1987.

[15] C. G. Cassandras, *Introduction to discrete event systems*, 2nd ed. New York: Springer Science+Business Media, 2008.

[16] A. Colombo and D. Del Vecchio, "Supervisory control of differentially flat systems based on abstraction," in *IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, 2011, pp. 6134–6139.

[17] E. Dallal, A. Colombo, D. Del Vecchio, and S. Lafortune, "Supervisory control for collision avoidance in vehicular networks using discrete event abstractions," in *American Control Conference (ACC)*, 2013.

[18] V. Desaraju, H. Ro, M. Yang, E. Tay, S. Roth, and D. Del Vecchio, "Partial order techniques for vehicle collision avoidance: Application to an autonomous roundabout test-bed," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2009, pp. 82–87.

[19] J. Duperret, M. Hafner, and D. Del Vecchio, "Formal design of a provably safe robotic roundabout system," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2010, pp. 2006–2011.

[20] R. Verma, D. Del Vecchio, and H. Fathy, "Development of a Scaled Vehicle With Longitudinal Dynamics of an HMMWV for an ITS Testbed," *IEEE/ASME Transactions on Mechatronics*, vol. 13, no. 1, pp. 46–57, 2008.

[21] E. Kreyszig, *Advanced Engineering Mathematics*, 10th ed. Hoboken, NJ: Wiley, Aug. 2011.

[22] A. Rizzi, "Analysis and optimization of an experimental apparatus to test active safety systems in vehicles," Apr. 2014. [Online]. Available: https://www.politesi.polimi.it/handle/10589/92232

[23] H. P. Whitaker, J. Yamron, and A. Kezer, *Design of model-reference adaptive control systems for aircraft*. Massachusetts Institute of Technology, Instrumentation Laboratory, 1958.

[24] P. Falcone, M. Tufo, F. Borrelli, J. Asgari, and H. Tsengz, "A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems," in *IEEE Conference on Decision and Control (CDC)*, Dec. 2007, pp. 2980–2985.

[25] A. Colombo and D. Del Vecchio, "Efficient algorithms for collision avoidance at intersections," in *Hybrid Systems: Computation and Control (HSCC)*, 2012, pp. 145–154.