Safety and Liveness in Intelligent Intersections *

Hemant Kowshik[1], Derek Caveney[2], and P. R. Kumar[1]

[1] CSL and ECE, University of Illinois, Urbana-Champaign
1308, West Main Street, Urbana, IL-61801, USA.
[2] The Toyota Technical Center,
2350 Green Road, Ann Arbor, MI-48105, USA.
kowshik2@uiuc.edu,derek.caveney@tema.toyota.com,prkumar@uiuc.edu

Abstract. Automation of driving tasks is becoming of increasing interest for highway traffic management. Technologies for on-board sensing, combined with global positioning and inter-vehicular wireless communications, can potentially provide remarkable improvements in safety and efficiency. We address the problem of designing intelligent intersections where traffic lights and stop signs are removed, and cars negotiate the intersection through a combination of centralized and distributed decision making. Such intelligent intersections are representative of complex distributed hybrid systems which need architectures and algorithms with provable safety and liveness.

We propose a hybrid architecture which involves an appropriate interplay between centralized coordination and distributed freedom for the cars. Our approach is based on each car having an open-loop infinite horizon contingency plan, which is updated at each sampling time in a distributed fashion. We also define a partial order relation between cars which specifies to each car a set of cars whose worst case behaviors it should guard against. We prove the safety and liveness of the overall scheme. Concerning performance, we conduct a simulation study that shows the benefits over stop signs and traffic lights.

1 Introduction

In the near future, cars will have access to a wide range of information from GPS and onboard sensors such as radar, lidar, camera, gyroscopes, etc. Further, vehicle to vehicle wireless communication enabled by Dedicated Short Range Communication (DSRC) radios will enable the exchange of this information with other cars. This has opened up a plethora of opportunities in the area of Intelligent Transportation systems [5]. In this paper, we will focus on safety applications. Accidents currently account for 42,000 fatalities every year and an estimated 18 percent of the healthcare expenditure in the U.S. [3]. Developing

* This material is based upon work partially supported by the Toyota Technical Center, Ann Arbor, MI, under Contract Number 2006-06246, NSF under Contract Nos. NSF ECCS-0701604, CNS-07-21992, NSF CNS 05-19535, and CCR-0325716, and USARO under Contract No. W-911-NF-0710287.

technologies to enhance vehicular and passenger safety is of great interest, and an important application vis-a-vis safety is collision avoidance. Collision avoidance technologies today are *passive*, and depend on the human driver to respond accurately. Automation of driving tasks is becoming increasingly prominent, with the introduction of Adaptive Cruise Control (ACC), Lane Keeping Assist (LKA) and Advanced Parking Guidance (APG). Such automated systems which improve safety, comfort and efficiency are the motivation for this paper.

Our focus in this paper is on *intelligent intersections*. An intelligent intersection is one in which conventional traffic control devices are removed. Vehicles coordinate their movement across the intersection through a combination of centralized and distributed real-time decision making, that leverages global positioning, wireless communications and in-vehicle sensing and computation. Smooth coordination of vehicles through intersections will provide overall improvements in fuel efficiency, vehicle wear and travel time. Most importantly, intelligent intersections can provide *guaranteed* safety.

Intelligent intersections are representative of complex distributed hybrid systems which require architectures and algorithms that guarantee safety and liveness, as a prerequisite for their acceptance. For safety, we pose the problem of collision avoidance in a worst case setting. One needs to safeguard not only against the worst-case behaviors of the other agents, but also against uncertainties in sensing and communication. *Systemwide* safety requires coordination between vehicles. This raises the issue of what is the appropriate division of functionalities between distributed agents and centralized coordination.

We consider the design of a time-slot based architecture for intersection collision avoidance. Our approach to distributed safety is built on each car possessing, at each time step, an *infinite horizon contingency plan* called the "failsafe maneuver." At any time, if the car chooses to ignore all future information updates and simply executes the failsafe maneuver, this maneuver still ensures safety with respect to some subset of cars in the system. Alternatively, given updated state information, each car can modify its infinite horizon contingency plan while still preserving the safety property. This is reminiscent of receding horizon control [14] except that we are computing *infinite* horizon plans at each time step. Systemwide safety is guaranteed by inducing an ordering on the set of cars, which clearly defines the subset of cars to which a given car must defer. We provide a precise description of the *hybrid architecture* and algorithms for distributed agents, culminating in a proof of systemwide safety and liveness. For the last challenge of performance evaluation, we test our overall solution by simulation and compare it with stop signs and traffic lights. A precise mathematical evaluation of performance may be impracticable.

2 Related work

We first provide a brief flavor of the vast literature in this area. In the collision warning approach, various warning and overriding algorithms decide thresholds to raise an alarm, or apply the brakes. These algorithms are mostly ad hoc and are designed to take into account brake system delay and driver reaction time;

see [15] and [2]. In the driver assistance domain, cooperative collision warning systems [9] and cooperative ACC [4] have been studied. These technologies already represent a move towards automatic control of vehicles. However, they do not provide any safety guarantees and are only expected to aid the human.

Reference [13] helps to put the larger problem in perspective, decoupling the various technical, technological and policy issues. In [6], a cooperative collision warning system was designed based on future trajectory prediction and conflict detection. This approach closely resembles the approach taken in aircraft collision avoidance [7], where the state of the system is estimated and propagated through a model which could be deterministic, probabilistic or worst case, and conflicts are detected. In the automotive domain, however, prediction based approaches could lead to unacceptable false alarm rates.

In the area of safety verification of multi agent systems, [11] proposes a method to design controllers for safety specifications in hybrid systems. In [12], a game-theoretic approach is used to design provably safe conflict resolution maneuvers in air traffic management. In this approach, we need to compute solutions to Hamilton-Jacobi-Isaacs partial differential equations, which could be computationally complex. In [10], the problem of systemwide safety is addressed using a cooperative avoidance control approach. Strategies which minimize worst case performance are studied in [1] where the authors use a dynamic programming like recursion can be used to arrive at the min-max strategy.

3 Perpetual collision avoidance

Consider two agents A and B, with state spaces X_A and X_B respectively. Their states $x_A \in X_A$ and $x_B \in X_B$ could be vectors containing various elements of interest like position, velocity, etc. We adopt a discrete time perspective and describe their dynamics through *One-step reachability set mappings.*

Definition 1 (One-step reachability set mapping). *The one-step reachability set for A is specified by $\mathcal{R}_A : X_A \to 2^{X_A}$, which specifies the set of points $\mathcal{R}_A(x_A) \subseteq X_A$ reachable from a state x_A in one discrete time step. Similarly, the reachability set for B is specified by $\mathcal{R}_B : X_B \to 2^{X_B}$.*

We can extend this to the set of states reachable from a *set of initial states* in one time step. Thus, $\mathcal{R}_A(\Gamma_A)$ is the set of states which can be reached from some point in Γ_A. $\mathcal{R}_A(\Gamma_A) := \bigcup_{x_A \in \Gamma_A} \mathcal{R}_A(x_A), \qquad \mathcal{R}_B(\Gamma_B) := \bigcup_{x_B \in \Gamma_B} \mathcal{R}_B(x_B).$

We define a "collision relation" between points in the sets X_A and X_B, corresponding to some notion such as "being within K meters of each other."

Definition 2 (Collision relation). *Let $\mathcal{C}_{AB}$ be a subset of $X_A \times X_B$. We say that agents A and B have* collided *if their states x_A and x_B are such that $(x_A, x_B) \in \mathcal{C}_{AB}$. Alternatively, we can simply define a "collision set" $\mathcal{C}_A(x_A) := \{x_B : (x_A, x_B) \in \mathcal{C}_{AB}\}$.*

3.1 Perpetually maintainable relations

In the sequel, we address a scenario where A is free to move around, while B has to make worst case assumptions about A's behaviour and is responsible for perpetual collision avoidance. We suppose that we are given a "desirable

relation," i.e., a set valued mapping $\mathcal{P}_A : X_A \to 2^{X_B}$ between states x_A and x_B in the sense that we want $x_B \in \mathcal{P}_A(x_A)$ maintained at *all* times. We extend this mapping to sets by taking an intersection: $\mathcal{P}_A(\Gamma_A) := \bigcap_{x_A \in \Gamma_A} \mathcal{P}_A(x_A)$ for any $\Gamma_A \subseteq X_A$. This is the set of points in X_B that forms a desirable relationship with *every* point in Γ_A.

Given a desirable relationship $\mathcal{P}_A$, the question that arises is whether it can be indefinitely maintained by Agent B if one were to start with initial states that satisfied the relationship. This is to be guaranteed under worst case assumptions on Agent A, provided only that $x_A(t+1) \in \mathcal{R}_A(x_A(t))$, and that Agent B gets to observe $(x_A(t), x_B(t))$ at each time t before picking $x_B(t+1)$, which is constrained to lie in $\mathcal{R}_B(x_B(t))$. If this is so, then we will say that the relation $\mathcal{P}_A$ is *perpetually maintainable* by Agent B.

Theorem 1. *A relation $\mathcal{P}_A$ is* perpetually maintainable *by Agent B if and only if $\mathcal{R}_B(x_B) \cap \mathcal{P}_A(\mathcal{R}_A(x_A)) \neq \phi$ for all $x_A \in X_A$ and $x_B \in \mathcal{P}_A(x_A)$.*[3]

3.2 Perpetually avoiding collisions

We would like to ensure that there are never any collisions, i.e., $x_B(t) \notin C_A(x_A(t))$ for all t. Hence we would like to *determine* a perpetually maintainable relation $\mathcal{P}_A$ with the additional property that $\mathcal{P}_A(x_A) \cap \mathcal{C}_A(x_A) = \phi$ for all $x_A \in X_A$. Given, $x_A \in X_A$, we can try to compute such a $\mathcal{P}_A(x_A)$ as follows:

(i) We create a first iterate for $\mathcal{P}_A(x_A)$: $\mathcal{P}_A^{(0)}(x_A) := \{x_B \in X_B : x_A \mathcal{C} x_B\}^c$.

(ii) Given $\mathcal{P}_A^{(k)}(x_A)$, calculate $\mathcal{P}_A^{(k+1)}(x_A) := \mathcal{R}_B^{-1}(\mathcal{P}_A^{(k)}(\mathcal{R}_A(x_A))) \bigcap \mathcal{P}_A^{(k)}(x_A)$.

(iii) $\mathcal{P}_A(x_A) := \lim_{k\to\infty} \mathcal{P}_A^{(k)}(x_A)$.

Theorem 2. *(i) $\mathcal{P}_A^{(k+1)}(x_A) \subseteq \mathcal{P}_A^{(k)}(x_A)$ for all $x_A \in X_A$.*
(ii) Hence $\lim_{k\to\infty} \mathcal{P}_A^{(k)} =: \mathcal{P}_A(x_A)$ exists for each $x_A \in X_A$.
(iii) If $\overline{\mathcal{P}}_A : X_A \to 2^{X_B}$ is also perpetually maintainable and it satisfies $\overline{\mathcal{P}}_A(x_A) \cap \mathcal{C}_A(x_A) = \phi$ for all $x_A \in X_A$, then $\mathcal{P}_A(x_A) \supseteq \overline{\mathcal{P}}_A(x_A)$ for all $x_A \in X_A$.

Definition 3 (Safety relation). *We shall call a relation $\mathcal{P}_A$ satisfying:*
(i) $\mathcal{P}_A$ is perpetually maintainable,
(ii) $\mathcal{P}_A(x_A) \cap \mathcal{C}_A(x_A) = \phi$ for all $x_A \in X_A$,
a safety relation*, and such a set $\mathcal{P}_A(x_A)$ a* safety set *for $x_A \in X_A$.*

4 Cars on a lane

Consider two point cars A and B on a single lane, where the rear car B, makes worst case assumptions about the front car A, and is responsible for perpetual safety. Each car is restricted to non-negative velocity, i.e., it cannot travel backwards. Each car also has both an upper bound as well as a lower bound on its acceleration, where the latter is a negative quantity. Let $x_A \equiv \binom{s_A}{v_A}$ be the state vector for the front car with position s_A and velocity v_A; similarly for $x_B \equiv \binom{s_B}{v_B}$. Note that $s_B < s_A$, $v_A \geq 0$ and $v_B \geq 0$. Let $\underline{a}_A$, $\underline{a}_B$ be the minimum acceleration that A and B are capable of applying, respectively.

We declare a collision if the two cars are separated by less than K meters.

[3] For details of all proofs, we refer the reader to [8].

Theorem 3 (Perpetual safety for two cars on a lane). *The necessary and sufficient condition for perpetual collision avoidance (perpetual safety) of two cars A and B, with $s_A > s_B$, is*

$$K + s_B + \int_0^t (v_B + \underline{a}_B \tau)^+ d\tau \quad \leq \quad s_A + \int_0^t (v_A + \underline{a}_A \tau)^+ d\tau \qquad \forall t \geq 0.$$

Less conservatively, it is easy to see that we can choose any open-loop input $\{a_B(t) : t \geq 0\}$ resulting in velocity trajectory $\{v_B(t) : t \geq 0\}$ which satisfies:

$$K + s_B + \int_0^t v_B(\tau) d\tau \leq s_A + \int_0^t (v_A + \underline{a}_A \tau)^+ d\tau \quad \forall t \geq 0. \tag{1}$$

Note that the above theorem only guarantees *safety*. However, there is also the issue of whether traffic will actually flow on a street where cars follow such safe behaviour. This is the issue of *liveness*. Liveness will be guaranteed by an aggressive choice of the rear car strategy $a_B(\cdot)$, as described in Section 4.2.

4.1 Sampling with intermediate safety

Suppose the acceleration of A is constrained to lie in the interval $[\underline{a}_A, \overline{a}_A]$ and that of B in $[\underline{a}_B, \overline{a}_B]$, and that the rear car B receives updates on the state of the front car every T seconds, the *sampling interval*. Based on the information about the lead car A at time nT, B chooses an acceleration input $\{a_B(t) : t \in [nT, (n+1)T)\}$. For simplicity, let us restrict ourselves to *T-horizon strategies*, where if the current time is 0, $a_B(\cdot)$ is chosen identically equal to $\underline{a}_B$, except on the interval $[0, T]$.

4.2 Maximally aggressive but safe strategies

Definition 4 (Maximally aggressive strategy). *A* maximally aggressive strategy $a_B^*(\cdot)$ *is one which maximizes the distance travelled in each interval $[nT, (n+1)T)$, while still satisfying the safety condition (1) at each time nT.*

Let a_B^* denote the constant accleration in [0,T] for the *maximally aggressive constant control input strategy*. The condition (1) yields that a_B^* is the maximal acceleration in $[\underline{a}_B, \overline{a}_B]$ which satisfies the following two conditions :

$$K + s_B + \textstyle\int_0^t (v_B + a_B^* \tau)^+ d\tau \leq s_A + \int_0^t (v_A + \underline{a}_A \tau)^+ d\tau \quad 0 \leq t \leq T,$$

$$K + s_B + \textstyle\int_0^T (v_B + a_B^* \tau)^+ d\tau + \int_0^t ((v_B + a_B^* T)^+ + \underline{a}_B \tau)^+ d\tau$$

$$\leq \quad s_A + \textstyle\int_0^{T+t} (v_A + \underline{a}_A \tau)^+ d\tau \qquad \text{for } 0 \leq t \leq \frac{v_B + a_B^* \cdot T}{-\underline{a}_B}.$$

Note that all accelerations in the interval $[\underline{a}_B, a_B^*]$ yield perpetual safety. When car B receives fresh information at time T, it can simply repeat the same procedure, as in receding horizon control [14]. We thus arrive at a *piecewise constant input*, $a_B^*(t) = a_{B,n}$ for $nT \leq t < (n+1)T$, with $a_{B,n} \in [\underline{a}_B, a_{B,n}^*] \subseteq [\underline{a}_B, \overline{a}_B]$.

Remark 1. The above approach can be easily extended to multiple cars on a lane by simply following the rule that *each car makes worst case assumptions about the car immediately in front of it*; see [8].

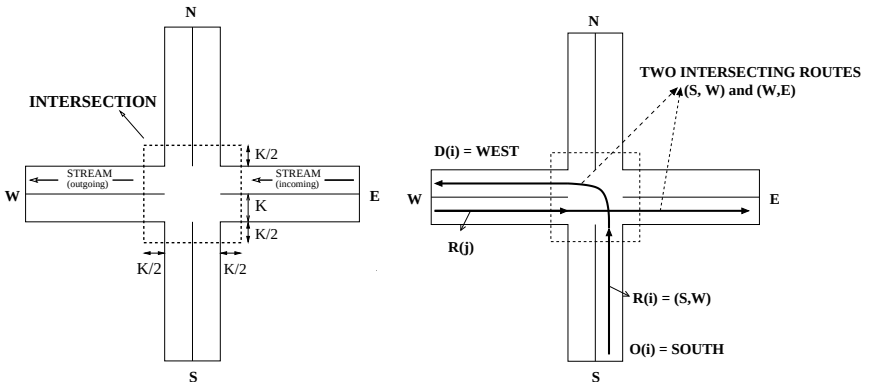
Remark 2. We can handle noisy information, random delays and packet loss, with minor modifications to the scheme described above; see [8].

5 Collision avoidance at intersections

Now we turn to the more general problem where there are two or more streams of cars crossing at an intersection. We need to devise a scheme which achieves two objectives. First, all cars cross the intersection without collisions and second, once cars are on their destination lanes, the perpetual safety condition continues to be satisfied for any pair of adjacent cars. Another issue of interest is liveness, in terms of avoiding deadlock. To solve these problems, we introduce a "hybrid" architecture. It is based on an interaction between cars and the intersection infrastructure based on time slot assignment by the intersection infrastructure, with the cars responsible for distributed safety with respect to collisions.

5.1 Description of system

We consider a four road intersection, with four incoming and four outgoing roads as shown in Figure 1(a). The incoming and outgoing roads are indexed by the directions N, W, E and S, as shown. Consider a system of m cars indexed $\{1, 2, 3 \ldots m\}$. Car i's acceleration is constrained to lie in an interval $[\underline{a}_i, \overline{a}_i]$ where $\underline{a}_i < 0$ and $\overline{a}_i > 0$. We assume that each car employs a *piecewise constant input*. We also assume that all cars have a *maximum speed limit* v_M. The following vocabulary will be useful.



(a) Intersection (b) Intersecting routes

Fig. 1. Description of system

By *intersection* we refer to the square consisting of the intersection proper as well as $K/2$ meters along each incoming and outgoing lane; see Figure 1(a). The *route* taken by a car i is described by an ordered pair $R(i) = (O(i), D(i))$, consisting of origin and destination respectively. Two routes are said to be *intersecting* if they cross each other; see Figure 1(b). For clarity, we add that two routes $R(i)$ and $R(j)$ are considered to be non-intersecting if $O(i) = O(j)$ or $D(i) = D(j)$. Thus, we have an "intersection relation" $\mathcal{I}$ defined on the set of routes. If two routes $R(i)$ and $R(j)$ are intersecting, we say $R(i)\ \mathcal{I}\ R(j)$.

We associate with each route a one-dimensional coordinate system, assuming that the position coordinate increases in the direction of traffic flow along the route. Hence each car i has its own coordinate system associated with its route $R(i)$. Let $s_i(t)$ and $v_i(t)$ denote the position and velocity of car i at time t in i's coordinate system. Further, along $R(i)$, let s_i^α and s_i^β denote the position coordinates of the beginning and end of the intersection, respectively.

5.2 Hybrid Architecture

We propose a *hybrid architecture* for collision avoidance at intersections. The intersection infrastructure functions as a *scheduler* which assigns a *time slot* to a car when it comes within communication range of the intersection, with the instruction that the car should be *strictly outside* the area covered by the intersection during all times *other* than its time-slot. This is done by the *Time Slot Allocation Algorithm* implemented at the intersection. Note that this does *not* mean that the car is required to be *in* the intersection during its time slot. Given a time slot, the car has to determine if it can safely get through the intersection and onto its destination lane in the allotted time slot, and compute acceleration inputs appropriately. If the car cannot get through the intersection in its time slot, or cannot safely get onto the destination lane, then it prepares to come to a halt before the intersection and receives a new slot. All this is done by the *Intersection Crossing Algorithm* implemented by each car in the system.

Time slot assignment policy The *time slot assignment* is a mapping σ which maps each car i in $\{1, 2, \ldots m\}$ to an interval of time $\sigma(i) = [t_{start}(i), t_{end}(i))$, called the *time slot* allocated to i. We will also allow for a *new* (or "revised") slot to be assigned to a car that has missed its earlier assigned time slot. This corresponds to modifying the time slot assignment. Hence we are interested in a sequence of time slot assignments $\{\sigma^{(0)}(\cdot), \sigma^{(1)}(\cdot), \ldots\}$, with the understanding that $\sigma^{(n)}(\cdot) : i \to [t^{(n)}_{start}(i), t^{(n)}_{end}(i))$ is the time slot assignment applicable during the time interval $[nT, (n+1)T)$. We will say that a car i *conforms* to the slot $\sigma(i)$ if it never occupies the intersection at any time outside $\sigma(i)$.

Suppose that we have a *strict partial ordering relation* "$\prec$" on the set of slots, established by comparing slot start times, that is, we say, $\sigma(i) \prec \sigma(j)$ if and only if $t_{start}(i) < t_{start}(j)$.

Definition 5 (Admissible Time Slot Assignment sequence). *We say that a time slot assignment sequence* $\{\sigma^{(0)}(\cdot), \sigma^{(1)}(\cdot), \ldots\}$ *is* admissible *if it satisfies the following properties.*
(a) *For any two cars i and j, if $R(i)\ \mathcal{I}\ R(j)$, then $\sigma^{(n)}(j) \cap \sigma^{(n)}(i) = \phi$ for all n. This ensures that two cars with intersecting routes have non-intersecting slots.*
(b) *For any car i, define $I(i) := \{j : D(j) = D(i), O(j) \neq O(i)\}$. Then we must have $\sigma^{(n)}(i) \prec \sigma^{(n)}(\overline{i})$ or $\sigma^{(n)}(\overline{i}) \prec \sigma^{(n)}(i)$, for all n, for all i and for all $\overline{i} \in I(i)$.*

In the sequel, for each car i at time nT, we will have an available open loop sequence *of inputs called the* failsafe maneuver*, denoted by $\{a_i^{F,n}(k)\}_{k \geq n}$. This determines an open-loop future trajectory for car i, which it can thereafter follow and stay perpetually safe.*

Reallocation Policy: *Suppose car i gets a "revised" slot at time nT, i.e., $\sigma^{(n-1)}(i) \neq \sigma^{(n)}(i)$. Then the following conditions must be satisfied:*
(i) *The reallocated slot must be in the future, i.e., $t^{(n)}_{start}(i) \geq nT$.*
(ii) *Under the available failsafe maneuver at time nT, if it is implemented at time nT, then car i will come to a stop before the intersection.*
(iii) *Reallocation cannot be done too early; reallocation is permitted at time nT only if $nT \geq t^{(n-1)}_{end}(i) - \tau_{max} + T$, where τ_{max} is the length of time enough for any car starting from rest to get through the intersection.*
(iv) *Consider the set of cars $\{j : R(j) = R(i), t^{(n-1)}_{end}(j) > nT\}$; let ζ be the car in this set which is highest in the ordering $\prec$ (this need not be unique). Then we must have, $\sigma^{(n-1)}(k) \prec \sigma^{(n)}(i)$ for all $\{k \in I(i) : \sigma^{(n-1)}(k) \prec \sigma^{(n-1)}(\zeta)\}$. If such a ζ does not exist, we must have $\sigma^{(n-1)}(k) \prec \sigma^{(n)}(i)$ for all $k \in I(i)$.*

Three Maneuvers We now define three maneuvers, a "braking" maneuver, a "parking" maneuver and a "tailing" maneuver. These maneuvers will be used in what follows to compose more complex behavior that ensures safety.

We adopt a discrete-time viewpoint and suppose that information about other cars in the system refreshes periodically every T seconds. We denote by $\{s_i(n)\}$, $\{v_i(n)\}$ the sampled position and velocity in car i's coordinate system, and by $\{a_i(n)\}$ the piecewise constant input, all of car i, in the time interval $[nT, (n+1)T)$. We say that a *car j is on i's route at time nT*, if either
(i) $O(j) = O(i)$ and car j is located $< K$ meters from a point on $R(i)$, or
(ii) $D(j) = D(i)$ and car j is located at a point on $R(i)$ and $t^{(n)}_{end}(j) \leq nT$.

Given a car i, consider any other car j with $O(j) = O(i)$ or $D(j) = D(i)$. We can project the position and velocity of car j onto i's coordinate system as follows. If car j is not on i's route at time nT, we set $s_{ji}(n) := s^{\alpha}_i + K$ and $v_{ji}(n) := 0$. If car j is on i's route at time t, we have two cases. If $O(j) = O(i)$, we set $s_{ji}(n) := s^{\alpha}_i + s_j(n) - s^{\alpha}_j$ and $v_{ji}(n) := v_j(n)$; if $D(j) = D(i)$, we set $s_{ji}(n) := s^{\beta}_i + s_j(n) - s^{\beta}_j$ and $v_{ji}(n) := v_j(n)$.

For a car i at time nT, we define its *lead car* $l(i, nT)$ as the car immediately in front of car i on i's route at time t. If there is no such lead car in front of car i on i's route, a *virtual lead car* is assumed to be situated at $+\infty$ along i's route. We declare a *collision* between two cars i and j if they are less than K meters apart. Consider two cars i and j with $O(j) = O(i)$ or $D(j) = D(i)$. If $s_{ji}(n) > s_i(n)$, the minimum value of $s_{ji}(n) - s_i(n)$ required to ensure a "physical" separation of K meters between car i and car j at time nT, is denoted by K_{ji}. If $O(j) = O(i)$ and $D(j) \neq D(i)$, K_{ji} is the distance along $R(j)$ beyond s^{α}_j, after which j is no longer on i's route, or $K_{ji} = K$ otherwise.
Maximum braking maneuver: A car i is said to execute the *maximum braking* (MB) maneuver at time nT, if $a_i(k) = \underline{a}_i \quad \forall k \geq n$.
Parking maneuver: For car i at time nT, a *parking maneuver* stopping at s_{park} consists of a choice of $\{a_i(k)\}_{k \geq n}$ and an $n^* \geq n$, such that $s_i(k) = s_{park}$ for all $k \geq n^*$. Applying this sequence of inputs will result in car i parking (i.e., coming to a standstill) at s_{park} and staying there for all further time. We note that such a maneuver may be infeasible for certain values of s_{park}. The *minimum-time*

parking maneuver is the parking maneuver with the smallest value of n^*.
Tailing maneuver: Consider two cars i and j with $R(i) = R(j)$. A *tailing maneuver* for car i behind car j at time nT is a sequence of acceleration inputs $\{a_i(k)\}_{k\geq n}$ which guarantees $s_j(t) - s_i(t) \geq K$ for all $t \geq nT$ under worst case assumptions (viz., maximum braking) on car j, and results in car i parking at $s_j(n) + \frac{v_j(n)^2}{-2\underline{a}_j} - K$. A specific extremal tailing maneuver of interest is the *minimum-time tailing maneuver* which stops in minimum time.

Downstream cars: Real and Virtual It is necessary for cars to take responsibility to avoid collisions with the other cars that are "ahead" of them.
Potential Downstream Cars: The set of potential downstream cars $\mathcal{D}(i, nT)$ for car i at time nT, is defined as the set of cars that consists of:
(i) All cars $j \neq i$ on i's route at time nT, with $s_{ji}(n) \geq s_i(n)$.
(ii) All cars j not on i's route at time nT, with $D(j) = D(i)$, $nT < t_{end}^{(n)}(j)$ and $\sigma^{(n)}(j) \prec \sigma^{(n)}(i)$.
(iii) A virtual car 0 with $\{s_0(\cdot)\} \equiv \infty$, $\{v_0(\cdot)\} \equiv \infty$ and $\underline{a}_0 = 0$.
The above is a *set* of cars. We now define one car, the *immediate downstream car*, that car i will need to take responsibility for avoiding.
Immediate Downstream Car: The immediate downstream car $d(i)$ for a car i is a *virtual car* with location and velocity given as follows:

$$s_{d(i)}(n) = \min_{j\in\mathcal{D}(i,nT)} s_{ji}(n), \qquad a_{d(i)}(n) = \underline{a}_i,$$
$$s_{d(i)}(n) + \tfrac{v_{d(i)}^2(n)}{-2\underline{a}_i} - K = \min_{j\in\mathcal{D}(i,nT)} \left\{ s_{ji}(n) + \tfrac{v_{ji}^2(n)}{-2\underline{a}} - K_{ji} \right\},$$

where $\underline{a} = \min_{1\leq j\leq m} \underline{a}_j$. Now we show that it is enough to make worst case assumptions on the virtual car $d(i)$ instead of all cars in $\mathcal{D}(i, nT)$.
Lemma 1. *Given a car i at time nT, the continuous time evolution of the state $(s_j(t), v_j(t))$ of any car $j \in \mathcal{D}(i, nT)$ satisfies*

$$\left(s_{d(i)}(nT) + \int_0^\tau (v_{d(i)}(nT) + \underline{a}_i s)^+ ds - K \right) \leq s_{ji}(nT + \tau) - K_{ji} \ \forall \tau \geq 0.$$

Outline of Algorithm for Perpetual Safety Let us suppose that the following two properties hold for each car i. In the sequel we will show how to maintain them.
Property P1: Each car i conforms to its slot sequence $\{\sigma^{(0)}(i), \sigma^{(1)}(i), \ldots\}$.
Property P2: Each car i does not collide with any car in $\mathcal{D}(i, nT)$ in the interval $[nT, (n+1)T)$ for all n.
Lemma 2. *Given an admissible time slot assignment sequence, if the two properties $P1$ and $P2$ are satisfied by each car i, then there is perpetual collision avoidance for all cars in the system.*
Failsafe maneuver update algorithm Our scheme for perpetual collision avoidance is predicated upon each car having a so called *failsafe maneuver* at every time, which it can apply from that time forward in an open-loop fashion, and guarantee safety. The algorithm to ensure perpetual collision avoidance for all cars in the system is based on the iterative update of the available failsafe maneuver at each time nT. Given a failsafe maneuver $\{a_i^{F,n}(k)\}_{k\geq n}$, for car i at

time nT, and information about other cars at time nT, we prescribe the current input $a_i(n)$ and the failsafe maneuver $\{a_i^{F,n+1}(k)\}_{k\geq n+1}$ at time $(n+1)T$.

Step 1 *(Determine if car i will stop before the intersection if it executes the one-step Modified Maximum Braking maneuver $\equiv \{\overline{a}_i, \underline{a}_i, \underline{a}_i, \ldots\}$ at time nT).*

Suppose car i executes the one step Modified Maximum Braking (MMB) maneuver at time nT. This will result in car i stopping at $s_i^{MMB}(\infty)$.

If $s_i^{MMB}(\infty) < s_i^{\alpha}$, then

Car i chooses any $a_i(n)$ which ensures that car i stays in the safety set of the lead car $l(i, nT)$ (as described in Section 4), and sets $a_i^{F,n+1}(k) = \underline{a}_i \quad \forall k \geq n+1$.

Else, go to Step 2

Step 2 *(Ensure that car i does not enter the intersection before the start of the assigned slot. In particular, if, even under maximum braking, car i is inside the intersection at the start of its slot, it must execute the failsafe maneuver).*

Let $\{s_i^{MB}(k)\}_{k\geq n}$, $\{v_i^{MB}(k)\}_{k\geq n}$ and $\{a_i^{MB}(k)\}_{k\geq n}$ denote the resulting position, velocity and acceleration profiles of car i if car i were to execute the Maximum Braking (MB) maneuver at time nT.

If $s_i^{MB}(\frac{t_{start}^{(n)}(i)}{T}) > s_i^{\alpha}$, then

Car i does go ahead and execute the current failsafe maneuver, i.e., car i chooses $a_i(n) = a_i^{F,n}(n)$ and sets $a_i^{F,n+1}(k) = a_i^{F,n}(k) \quad \forall k \geq n+1$.

Else, go to Step 3.

Step 3 *(Ensure that car i exits the intersection before $t_{end}^{(n)}(i)$, and is in the safety set of its lead car upon exit. This is done by checking if car i can safely tail the immediate downstream car).*

Construct the Minimum-Time Tailing (MTT) maneuver behind the immediate downstream car $d(i)$, for car i at time nT. For convenience, let $\{s_i^{MTT}(k)\}_{k\geq n}$, $\{v_i^{MTT}(k)\}_{k\geq n}$ and $\{a_i^{MTT}(k)\}_{k\geq n}$ denote the resulting position, velocity and acceleration profiles of car i, if car i were to execute the MTT maneuver.

If the tailing maneuver behind $d(i)$ is infeasible, or if $s_i^{MTT}\left(\frac{t_{end}^{(n)}(i)}{T}\right) \leq s_i^{\beta}$, then

Car i goes ahead and does execute the current failsafe maneuver, i.e., car i chooses $a_i(n) = a_i^{F,n}(n)$ and sets $a_i^{F,n+1}(k) = a_i^{F,n}(k) \quad \forall k \geq n+1$.

Else, go to Step 4.

Step 4 *(Given that the MTT maneuver behind $d(i)$ is feasible, check if, under this maneuver, car i conforms to its time slot).*

If $s_i^{MTT}(\frac{t_{start}^{(n)}(i)}{T}) \leq s_i^{\alpha}$, then

Car i goes ahead and executes the MTT maneuver, i.e., car i chooses $a_i(n) = a_i^{MTT}(n)$ and sets $a_i^{F,n+1}(k) = a_i^{MTT}(k) \quad \forall k \geq n+1$.

Else, go to Step 5.

Step 5 *(Synthesize a failsafe maneuver using the MB maneuver and the MTT maneuver behind $d(i)$. Check that, under this synthesized maneuver, car i conforms to its time slot).*

Define a sequence of acceleration inputs $\{a_i^*(k)\}_{k\geq n}$ as follows. First, define $a_i^*(k) := \lambda.a_i^{MTT}(k) + (1-\lambda).\underline{a}_i$ for all $k \in \{n, \ldots, \frac{t_{start}^{(n)}(i)}{T} - 1\}$ where

$\lambda = \frac{s_i^{\alpha} - s_i^{MB}(\frac{t_{start}^{(n)}(i)}{T})}{s_i^{MTT}(\frac{t_{start}^{(n)}(i)}{T}) - s_i^{MB}(\frac{t_{start}^{(n)}(i)}{T})}$. For $k \geq \frac{t_{start}^{(n)}(i)}{T}$, $a_i^*(k)$ is the sequence of acceleration inputs corresponding to the Minimum-Time Parking maneuver behind $s_{d(i)}(n) + \frac{v_{d(i)}^2(n)}{-2\underline{a}_{d(i)}}$ for car i, with initial time set to $\frac{t_{start}^{(n)}(i)}{T}$, initial position $s_i^*(\frac{t_{start}^{(n)}(i)}{T})$ and initial velocity $v_i^*(\frac{t_{start}^{(n)}(i)}{T})$.
If $s_i^*(\frac{t_{end}(i)}{T}) \leq s_i^{\beta}$, then
Car i goes ahead and executes the current failsafe maneuver, i.e., car i chooses $a_i(n) = a_i^{F,n}(n)$ and sets $a_i^{F,n+1}(k) = a_i^{F,n}(k) \quad \forall k \geq n+1$.
Else, go to Step 6.
Step 6
Car i simply chooses $a_i(n) = a_i^*(n)$ and sets $a_i^{F,n+1}(k) = a_i^*(k) \quad \forall k \geq n+1$.

The Intersection Crossing Algorithm Now we are ready to specify the algorithm that cars use in the hybrid architecture.
At time zero, each car i sets the failsafe maneuver $\{a_i^{F,0}(k)\}_{k\geq 0}$ to be the maximum braking maneuver. At every time step nT, car i has two choices. It can
(i) Choose the current input $a_i(n)$ and update the failsafe maneuver by running the update algorithm using information from other cars at time nT,
(ii) Or, it can simply execute the failsafe maneuver at time nT, i.e., $a_i(n) = a_i^{F,n}(n)$ and set $\{a_i^{F,n+1}(k)\}_{k\geq n+1} = \{a_i^{F,n}(k)\}_{k\geq n+1}$. This corresponds to following the "contingency plan" at time nT, possibly due to lost packets.
Once car i exits the intersection, it sets $s_i^{\alpha}, s_i^{\beta} = +\infty$, and continues the failsafe maneuver update. This will simply amount to repeatedly executing Step 1 of the update algorithm.[4]

Safety of the Intersection Crossing Algorithm In order to establish safety, we need to analyze the evolution of positions of the cars, their failsafe maneuvers, and the time slots allocated to them. Let $x(n) \in \mathcal{X}$ represent the "physical state" of the system at time nT, which includes position, velocity, etc., of all cars in the system. Let $\pi^{n|x(n-1)} \in \Pi$ be the "planning state" at time nT, which is the set of failsafe maneuvers at time nT; $\pi^{n|x(n-1)} = \{p_1^{n|x(n-1)}, p_2^{n|x(n-1)}, \ldots, p_m^{n|x(n-1)}\}$. Finally, let $\sigma^{(n)} \in \Sigma$ be the time slot assignment during $[nT, (n+1)T)$.

The *superstate* of the system at time nT is given by $(\underline{x(n)}, \pi^{n|x(n-1)}, \sigma^{(n)}) \in \mathcal{X} \times \Pi \times \Sigma$, where the underlining of $x(n)$ is to indicate that this information is *private*. The superstate evolves in four steps as follows:
$(\underline{x(n)}, \pi^{n|x(n-1)}, \sigma^{(n)}) \stackrel{StepA}{\longrightarrow} (x(n), U(n) \times \pi^{n+1|x(n)}, \sigma^{(n)})$
$\stackrel{StepB}{\longrightarrow} (x(n), u(n) \in U(n), \pi^{n+1|x(n)}, \sigma^{(n)}) \stackrel{StepC}{\longrightarrow} (\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n)})$
$\stackrel{StepD}{\longrightarrow} (\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n+1)})$.
Step A: When the cars exchange physical state information at time nT, each car i can run the Intersection Crossing Algorithm, which prescribes a set of feasible current inputs $U_i(n)$ and the updated failsafe maneuver $p_i^{n+1|x(n)}$ for car i. This results in a set of feasible inputs $U(n) = U_1(n) \times U_2(n) \ldots U_m(n)$ for the system

[4] In a road network, set the values of s_i^{α} and s_i^{β} to correspond to the next intersection.

and an updated planning state $\pi^{n+1|x(n)}$.
Step B: Each car i can then choose a particular input $a_i(n) \in U_i(n)$, which results in the system choosing $u(n) \in U(n)$.
Step C: The physical state of the system evolves from $x(n)$ to $x(n+1)$.
Step D: Finally, the time slot assignment is updated from $\sigma^{(n)}$ to $\sigma^{(n+1)}$.

We prove the safety property of the intersection crossing algorithm by showing that there is an *invariant set* $\mathcal{A} \subset \mathcal{X} \times \Pi \times \Sigma$ for the superstate. $\mathcal{A}$ will comprise of all those superstates $(x(n), \pi^{n|x(n-1)}, \sigma^{(n)})$ for which, under the maneuver $p_i^{n|x(n-1)}$ for each car i at time nT, it results that
($C1^{(n)}$): Car i conforms at all future times to the slot $\sigma^{(n)}(i)$, i.e., if the time slot $\sigma^{(n)}(i)$ is never changed in the future and is kept "frozen."
($C2^{(n)}$): Car i does not collide with any car in $\mathcal{D}(i, nT)$ under worst case assumptions on cars in $\mathcal{D}(i, nT)$.

Theorem 4. *Suppose we have an admissible Time Slot Assignment Sequence and that all cars are following the intersection crossing algorithm described above. Let us suppose that* $(\underline{x(n)}, \pi^{n|x(n-1)}, \sigma^{(n)}) \in \mathcal{A}$ *at time* nT. *Then,*

(a) *The set of superstates* $(x(n), U(n) \times \pi^{n+1|x(n)}, \sigma^{(n)}) \subseteq \mathcal{A}$.

(b) *If each car* i *chooses the current input* $a_i(n)$ *as described, then there are no collisions in* $[nT, (n+1)T)$. *Further* $(\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n)}) \in \mathcal{A}$.

(c) *Under the slot reallocation policy, we have* $(\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n+1)}) \in \mathcal{A}$.

Perpetual Systemwide safety It remains to prove *perpetual systemwide safety* of the hybrid architecture. We make the following assumptions:
(A1) We have an admissible time slot assignment sequence.
(A2) At time zero, all cars are at least a braking distance away from the intersection. Further, each car i is in the safety set of its lead car at time zero.

Theorem 5. ***Systemwide Safety of the Intersection Crossing Algorithm*** *Under conditions (A1) and (A2), if each car follows the Intersection Crossing Algorithm, there is perpetual collision avoidance for the whole system of cars.*

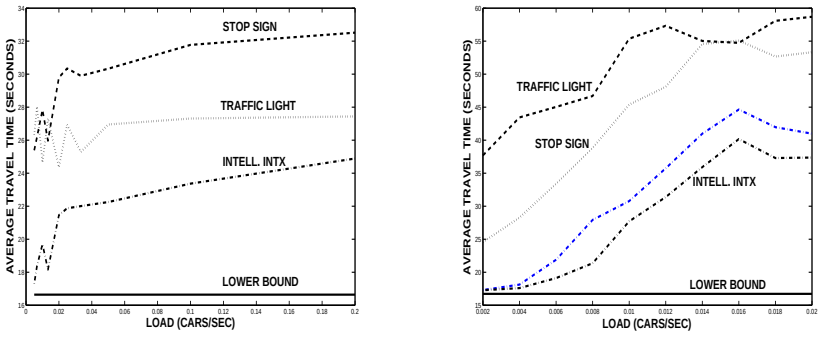
Liveness of the Intersection Crossing algorithm As noted earlier, in addition to safety, it is also important to ensure that any finite system of cars does not fall into deadlock, i.e., every car must cross the intersection in finite time. We can ensure liveness under the following additional conditions:
(A3) Suppose that all cars in the system are α-*aggressive*, i.e., in Step 1 of the failsafe maneuver update algorithm, if the maximum permissible acceleration $(a_i^{max}(n))$ for a car i at time nT is positive, then the chosen input must be at least $\alpha \cdot a_i^{max}(n)$ where $0 < \alpha \leq 1$.
(A4) The partial ordering relation "$\prec$" satisfies the condition:
$\sigma(i) \prec \sigma(j) \Rightarrow l(\sigma(j) \cap \sigma^c(i)) \geq \tau_{max}$, where $l(A)$ is the maximum length of an interval contained in A, and τ_{max} is a length of time enough for any car starting from rest to traverse through the intersection.
(A5) A new slot is reallocated within Δ seconds of missing an earlier slot.

Theorem 6. ***Liveness of the Intersection Crossing Algorithm*** *Under conditions (A1) through (A5), if each car follows the Intersection Crossing Algorithm, then there is guaranteed liveness for the whole system of cars.*

6 Performance Evaluation

The algorithm and architecture described above ensure systemwide safety and liveness, while still providing freedom in the design space. We would like to find time slot assignments which satisfy conditions (A1)-(A2), and locally maximize an appropriate performance metric. In order to find such efficient slot assignments, we use a local improvement heuristic, specifically a gradient approach based on forward simulation. We run the simulation with an arbitrary initial slot assignment and record the average travel time and the final slot assignment. We systematically tweak this to obtain modified time slot assignments, which we again evaluate by forward simulation. When we obtain an assignment which entails lower average travel time, we switch to it, and continue this procedure recursively. The algorithm terminates when we obtain a slot assignment, all of whose modifications result in higher average travel time.

We have built a simulator of the entire system. We consider a system of m cars which desire to get through the intersection. The performance metric under consideration is the average time taken to travel from 200m away from the start of the intersection to 200m beyond the end of the intersection. The routes of all cars are independent and identically distributed according to a probability mass function which assigns a mass of $1/6$ to each of the four straight line routes, and a mass of $1/24$ to each of the eight turning routes. All cars start from at least 200m away from the start of the intersection, with each subsequent car being positioned at a distance of $(K + exp(\lambda)RV)$ behind its lead car with $K = 5m$. The parameter λ of the exponential distribution is a measure of the *load on the intersection*. All cars start at maximum velocity equal to $25m/s$, and are assumed to have braking power equal to $-3.5m/s^2$.



(a) Clearing 10 cars (b) Clearing 30 cars

Fig. 2. Average travel time comparison

Using this simulation framework, we can compare the performance of our scheduler against traffic regulation mechanisms such as stop signs and traffic lights; see Figure 2. We see that our intelligent intersection outperforms both traffic lights and stop signs at low and moderate loads by fairly good margins. It even appears to perform comparably or better at high loads. However, we should note that these conclusions deserve a much more thorough simulation study.

7 Concluding Remarks

This paper has examined an important safety application, intelligent intersections, that can provide provable safety with improved efficiency. We have proposed a design based on distributed updates of infinite horizon contingency plans by distributed agents, with centralized coordination. We have also demonstrated by a simulation study the performance benefit of our approach over stop signs and traffic lights. It is hoped that this approach may be useful in the design of other tractable complex, distributed, hybrid systems.

References

1. T. Başar and P. R. Kumar, "On worst case design strategies," *Computers and Mathematics with Applications*, vol. 13(1–3), pp. 239–245, 1987.
2. A. Doi, T. Butsuen, T. Niibe, T. Yakagi, Y. Yamamoto, and H. Seni, "Development of a rear-end collision avoidance system with automatic braking control," *JSAE Review*, vol. 15, pp. 335–340, 1994.
3. National Center for Statistics and Analysis, *2006 Traffic Safety Annual Assessment - A Preview,* DOT HS 810 791, July 2007.
4. A. R. Girard, J. B. de Sousa, J. A. Misener, and J. K. Hedrick, "A control architecture for integrated cooperative cruise control and collision warning systems," in *Proceedings of the 40th Conference on Decision and Control*, vol. 2, pp. 1491–1496.
5. California Center for Innovative Transportation, "Intelligent transportation systems," August 2007. [Online]. Available: http://www.calccit.org/itsdecision.
6. J. Huang and H. S. Tan, "Design and implementation of a cooperative collision warning system," in *Proceedings of the IEEE Intelligent Transportation Systems Conference*, Toronto, Canada, 2006, pp. 1017–1022.
7. J. K. Kuchar and L. C. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, pp. 179–189, 2000.
8. H. Kowshik, D. Caveney, and P. R. Kumar. Provable Systemwide Safety in Intelligent Intersections. *Submitted to IEEE Transactions Automatic Control*, 2007.
9. J. A. Misener, R. Sengupta, and H. Krishnan, "Cooperative collision warning: Enabling crash avoidance with wireless technology," in *12th World Congress on Intelligent Transportation Systems*, 2005, San Francisco, paper 1960.
10. D. M. Stipanovic, P. F. Hokayem, M. W. Spong, and D. D. Siljak, "Avoidance control for multi-agent systems," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 129(5), pp. 699–707, 2007.
11. C. Tomlin, J. Lygeros, and S. Sastry, "A game theoretic approach to controller design for hybrid systems," *Proceedings of the IEEE*, vol. 88(7), pp. 949–970, 2000.
12. C. Tomlin, G. Pappas, and S. Sastry, "Conflict resolution of air traffic management: A study in multiagent hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43(4), pp. 509–521, 1998.
13. P. Varaiya, "Smart cars on smart roads: Problems of control," *IEEE Transactions on Automatic Control*, vol. 38, pp. 195–207, 1993.
14. B. E. Ydstie and L. K. Liu, "Single- and multi-variable control with extended prediction horizons," in *American Control Conference*, vol. 21, pp. 1303–1308.
15. Y. Zhang, E. K. Antonsson, and K. Grote, "A new threat assessment measure for collision avoidance systems," in *Proceedings of the IEEE Intelligent Transportation Systems Conference*, Toronto, Canada, 2006, pp. 968–975.