Provable Systemwide Safety in Intelligent Intersections[*]

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

Abstract

Automation of driving tasks is of increasing interest for highway traffic management. The emerging technologies of global positioning and inter-vehicular wireless communications, combined with in-vehicle computation and sensing capabilities, 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 an interaction of centralized and distributed decision making. Intelligent intersections are representative of complex hybrid systems that are increasingly of interest, where the challenge is to design tractable distributed algorithms that guarantee safety and provide good performance.

Systems of automatically driven vehicles will need an underlying collision avoidance system with provable safety properties in order to be acceptable. This raises several challenges. We need to ensure perpetual collision avoidance so that cars do not get into future problematic positions in order to avoid an immediate collision. The architecture needs to allow distributed freedom of action to cars, yet should guard against worst case behavior of other cars to guarantee collision avoidance. The algorithms should be tractable both computationally and in information requirements, and robust to uncertainties in sensing and communication.

To address these challenges, we propose a hybrid architecture with an appropriate interplay between centralized coordination and distributed freedom of action. The approach is built around a core where each car has an infinite horizon contingency plan to avoid collisions. This contingency plan is updated at each sampling instant, distributedly by the cars, in a computationally tractable manner. We also define a dynamically changing partial order relation between cars which specifies for each car a set of cars whose worst case behaviors it should guard against. The architecture is hybrid, involving a centralized component that coordinates intersection traversals. We prove safety and liveness of the overall scheme. The mathematical challenge of quantifying performance accurately remains as a difficult challenge and so we conduct a simulation study that shows the benefits over stop signs and traffic lights.

It is hoped that efforts such as this can provide methodologies for the design of tractable solutions for complex distributed systems requiring safety and liveness guarantees.

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., concerning position, velocity, acceleration,brake pressure, etc., that are made available through the CAN bus. Additionally, cars equipped with Dedicated Short Range Communication (DSRC) radios will have the ability to exchange information with other cars

[*]This material is based upon work partially supported by the Toyota Technical Center 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. A brief version of this work has been accepted for presentation at the 11th Conference on Hybrid Systems: Computation and Control 2008 at St. Louis, MO, USA.

[†]H. Kowshik and P. R. Kumar are with the University of Illinois, Urbana-Champaign. Address: CSL, 1308 West Main Street, Urbana, IL-61801, USA. Email: kowshik2, prkumar@uiuc.edu. D. Caveney is with the Toyota Techni- cal Center, Ann Arbor. Address: 2350 Green Road, Ann Arbor, MI- 48105. Email: derek.caveney@tema.toyota.com

and the road side infrastructure. The FCC in the U.S. has allocated spectrum for such vehicle-to-vehicle and vehicle-to-infrastructure communication [1]. This wide web of available information has opened up a plethora of opportunities in the area of Intelligent Transportation Systems, and several ongoing efforts can be found in [2]. Envisaged non-safety applications include route guidance, ramp metering, congestion management and electronic toll collection. In this paper, we will focus on safety applications.

Accidents currently account for 42,000 fatalities annually [3] and an estimated 18 percent of the healthcare expenditure in the U.S. Technologies to enhance vehicular and passenger safety are of great interest, an important application being collision avoidance. Collision avoidance technologies today are largely vehicle-based systems offered by original equipment manufacturers as autonomous packages which broadly serve two functions, collision warning and driver assistance. The former warn the driver when a collision seems imminent, while the latter partially control the vehicle either for steady-state or as an emergency intervention. However, such systems are passive, and depend on the human driver to respond accurately. Automation of driving tasks is increasing, evidenced by several advanced driver assistance systems that have come to market over the past decade. These include Adaptive Cruise Control (ACC), Lane Keeping Assist (LKA) and Advanced Parking Guidance (APG), which remove aspects of longitudinal speed control and steering control from the driver workload during highway driving and parking maneuvers. Although these could be seen as comfort amenities to alleviate unnecessary workload on the driver, they may also be seen as improvements to both vehicle safety and overall driving efficiency. Additional automated systems which improve any or all three of these qualities will continue to be introduced in the automotive market, which is the motivation for this paper.

Our focus in this paper is on intelligent intersections, where conventional traffic control devices like stop signs and traffic signals are removed. Vehicles coordinate their movement across the intersection through a combination of centralized and distributed real-time decision making, utilizing global positioning, wireless communications and in-vehicle sensing and computation. The intelligent intersection is motivated by the potential benefits in comfort, safety, and efficiency. Removing the driver from negotiating the passage through complicated intersections will improve driver comfort. Furthermore, smooth coordination of vehicles through intersections will provide improvements in fuel efficiency, vehicle wear, travel time and traffic flow. It can not only provide throughput and delay benefits for low to moderate traffic arrival densities as at suburban or rural intersections, but can also be switched over seamlessly at higher traffic intensities to a more traditional traffic control operation without the need for any lights or stop signs or human intervention.

Intelligent intersections are representative of the increasing trend toward complex distributed hybrid systems which require architectures and algorithms that guarantee perpetual safety and liveness, as an essential prerequisite for acceptability, and it is hoped that the design approach presented here can be extended to more general distributed hybrid systems requiring provable safety. It is necessary to clearly specify the roles of individual cars and the information flow requirements in this architecture. Further, the algorithms for the cars must be computationally tractable, and robust to uncertainties in sensing and communication, such as noisy information and lost packets.

A significant challenge in design is that safety in intelligent intersections involves not only ensuring collision avoidance prior to entering the intersection but also within the intersection itself, and further, upon exiting the intersection. Clearly, this requires coordination between vehicles, which raises the issue of what is the interplay between centralized coordination and distributed freedom. Our approach is a hybrid one involving an appropriate separation of concerns between safety and livenss. We use distributed decision making by cars, based on their own local information to guarantee safety, and coordination between cars to get through the intersection.

We consider the design of a time slot based architecture for intersection collision avoidance. This introduces a tension between caution and aggression for a car which not only needs to get through the intersection in the given time slot, but also ensure its safety upon exiting the intersection. Thus, it is imperative that a car enters the intersection only if it can exit safely. Given this predilection towards conservative behavior, an important question is whether we can ensure liveness, i.e., a finite clearance time for a finite set of cars. Further, beyond liveness, one is also interested in high performance.

Our approach to distributed safety is built on each car possessing, at each time step, an infinite sequence of inputs called a *failsafe maneuver*, which plays the role of an infinite horizon contingency plan. At any time, if a car chooses to ignore all future information updates and simply executes its failsafe maneuver, this maneuver has the property that safety is still assured with respect to some subset of cars in the system. On the other hand, given updated state information, each car can modify its infinite horizon contingency plan from time step to time step, while still preserving the safety property. This is reminiscent of receding horizon control [4], except that we are computing infinite horizon plans at each time step. The second challenge is to develop a scheme where each car can compute its control in a distributed fashion. Our approach to distributed safety consists of inducing a partial ordering on the set of cars, which defines the subset of cars to which a given car must defer. This ordering must and does change with time as circumstances evolve, and so it is a dynamic partial ordering. We nevertheless show how this can be done so as to guarantee safety. We provide a precise description of the hybrid architecture and algorithms for distributed agents, culminating in a proof of not only systemwide safety but also liveness. For the last challenge of performance evaluation, we do not have any satisfactory theoretical method that can provide accurate results, due to the complexity of the overall system. So, as a first step, we test our overall solution by simulation and compare it with solutions based on current technology such as stop signs and traffic lights.

The rest of this paper is organized as follows. In Section 2, we provide a brief summary of related work in the area of vehicular collision avoidance.In Sections 3, 4 and 5, we consider problems of increasing complexity ranging from perpetual collision avoidance for two cars on a lane, to several cars on a single lane, and subsequently to cars on different streams crossing at an intersection. In Section 6, we provide a comparative simulation study of performance against stop signs and traffic lights.

2 Related Work

There is a vast body of literature on the problem of collision avoidance. We first provide a brief flavor of the various approaches. In the collision warning approach, a threat assessment metric is constructed based on the current scenario. Some popular metrics include time-to-collision, minimum deceleration required and time headway. Warning algorithms then 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 [5], [6] and [7]. In the driver assistance domain, technologies such as Adaptive Cruise Control (ACC), Anti-lock Brakes and lane change assistance are already available. More advanced applications like cooperative collision warning systems for blind spot and lane change assistance [8], Cooperative ACC [9] and vehicle platooning [10] have also 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, who still has to make the critical decisions.

Reference [11] helps to put the larger problem in perspective, decoupling the various technical, technological and policy issues. However, it is mainly focused on route guidance and easing congestion in a highway system, through cooperative means. There is no formal proof of safety. In [12], a cooperative collision warning system is designed based on future trajectory prediction and conflict detection. This approach closely resembles the approach taken in aircraft collision avoidance, 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. An elaborate survey of these approaches is provided in [13]. However, there is an important difference between the two domains. While aircraft can afford to be overcautious, with prediction horizons up to 30 minutes, horizons of even 5 seconds can lead to unacceptably high false alarm rates in cars. Further, the unpredictability of vehicular traffic behavior suggests that deterministic and probabilistic prediction models may be ineffective.

In the area of safety verification of multi-agent systems, [14] proposes a method to design controllers for safety specifications in hybrid systems. The safety specifications are transformed into restrictions on the system's reachable set of states. Then, analysis techniques from optimal control and non-cooperative zero-sum game theory are applied to derive provably safe control laws. In [15], a simliar game-theoretic approach is used to design provably safe conflict resolution maneuvers in air traffic management. In this approach, it is necessary to compute solutions to Hamilton-Jacobi-Isaacs partial differential equations, which could be computationally complex.

In [16], the problem of systemwide safety is addressed using a cooperative avoidance control approach proposed in [17]. It considers a non-cooperative setting, where avoidance control laws are computed using value functions that resemble barrier functions in static optimization. These value functions can simply be appended to the original cost function and the avoidance control law obtained by minimizing the augmented Hamiltonian. This scheme has the desirable property that the avoidance laws are active only in the bounded sensing regions of each individual agents, and do not interfere with the agents' individual optimal control laws outside these regions. This approach is suitable for multi-agent systems which are free to move around on the plane, but does not carry over directly to cars on a road network. Strategies which minimize worst case performance are studied in [18] where the authors show that a dynamic programming [19] like recursion can be used to arrive at the min-max strategy.

3 Two cars on a lane

In an automated vehicle system, a critical design goal is to ensure collision avoidance in *perpetuity*; that is, cars must avoid moving into positions that create problems in the future, even while avoiding collisions in the short term. One can develop a general set-theoretic formulation of the problem and study if safety is indeed a *perpetually maintainable relation* [20]. This has been omitted for brevity of presentation. Instead, we directly study the problem of perpetual collision avoidance for a sequence of increasingly complex situations of specific interest, which are elements of the much more complex systems we consider in the sequel. Again, we note that this approach leads to a distributed implementation of safety.

3.1 Point Cars with Bounded Acceleration and Nonnegative Velocity

Consider two cars A and B on a single lane, where the rear car, car B, is responsible for perpetual safety, in spite of worst case behavior of the front car A. Each car is restricted to nonnegative velocity, i.e., it cannot travel backwards, and also has both an upper bound as well as a lower bound on its acceleration, where the latter is a negative quantity. Here and through the rest of this paper, all cars are treated as being *on rails*, with no lateral movement. [1] 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 $x_B \equiv \binom{s_B}{v_B}$. Thus $s_B < s_A$, $v_A \geq 0$ and $v_B \geq 0$. The control input is the acceleration of the rear car a_B. Let $\underline{a}_A < 0$ be the minimum acceleration that A is capable of applying, and similarly let $\underline{a}_B < 0$ be the minimum acceleration that B is capable of applying. Note that the dynamics of car A is given by $\dot{s}_A(t) = v_A(t)$, $\dot{v}_A(t) = a_A(t)\mathbf{1}\{v_A(t) > 0$ or $a_A(t) \geq 0\}$, where $\mathbf{1}(E)$ is the indicator function of the event E, and likewise for car B.

To begin, we consider point cars of zero area and declare a collision if $s_A = s_B$.

Theorem 1 (Perpetual safety for two cars on a lane) *For initial conditions $s_A > s_B$, the necessary and sufficient condition for perpetual collision avoidance (perpetual safety) by car B, is*

$$s_B + \int_0^t (v_B + \underline{a}_B\tau)^+ d\tau \quad < \quad s_A + \int_0^t (v_A + \underline{a}_A\tau)^+ d\tau \qquad \forall t \geq 0, \text{ where } r^+ = max(r, 0). \tag{1}$$

Physically, this is saying that if the front car A brakes maximally, then the rear car B should be in such a state as to be able to avoid collision by also braking maximally. It can be simplified to $s_B + \int_0^t (v_B + \underline{a}_B\tau)^+ d\tau < s_A + \int_0^t (v_A + \underline{a}_A\tau)^+ d\tau, \forall t \in \left[0, \frac{v_B}{-\underline{a}_B}\right]$, for ease of computation. If (1) is satisfied, we say that car B is in the *safety set* of car A.

Above we have considered a discrete time version where information on position and velocity is obtained by the follower car at the beginning of each sampling interval, based on which it has to choose its next position to be assumed at the next sampling instant. However the condition does not explicitly involve the frequency (or sampling interval) at which information refreshes. Rather, it has to do with the configuration of the cars and does not have anything to do with how often information refreshes. However, the rate of information exchange will affect the throughput of cars on a street, i.e., performance, with slower refresh rate yielding lower throughput.

Proof of Theorem 1

Necessity: Note that $s_A > s_B$. Suppose that the above condition is violated, i.e., $\exists\ t^*$ such that $s_B + \int_0^{t^*} (v_B + \underline{a}_B\tau)^+ d\tau \geq s_A + \int_0^{t^*} (v_A + \underline{a}_A\tau)^+ d\tau$. Now consider the situation when the front car A brakes at maximum with an acceleration of $\underline{a}_A$. B's best choice is to brake at maximum with an acceleration of $\underline{a}_B$. However, even this will cause a collision by time t^*.

Sufficiency: Assuming condition (1), it is enough to show that there exists a safe input $\{a_B(t) : t \geq 0\}$ for the rear car, for all time. This is trivial since the rear car can choose $a_B(t) \equiv \underline{a}_B$ for all $t \geq 0$. The condition for worst case collision avoidance along this trajectory is equivalent to the condition in (1),

[1]We are grateful to the reviewer for comments on this issue

i.e., given any arbitrary input $\{a_A(t) : t \geq 0\}$ which results in the velocity trajectory $\{v_A(t) : t \geq 0\}$ for the front car,

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

Note that, in particular there is an *open-loop input* which maintains safety. Denote by $\{s_B(\tau) : \tau \geq 0\}$ and $\{v_B(\tau) : \tau \geq 0\}$ the resulting position and velocity trajectories of car B when input $\{a_B(\tau) : \tau \geq 0\}$. Less conservatively, we can choose any open-loop input $\{a_B(\tau) : \tau \geq 0\}$ for which

$$s_B + \int_0^t v_B(\tau) d\tau < \quad s_A + \int_0^t (v_A + \underline{a}_A \tau)^+ d\tau \qquad \forall t \geq 0. \tag{2}$$

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 behavior. This is the issue of *liveness*. It will be guaranteed by an aggressive choice of rear car strategy $a_B(\cdot)$, as described in Section 3.3.

Suppose cars need to maintain a minimum separation distance $K > 0$, so that cars separated by less than K meters are said to "collide". Then (2) simply becomes

$$K + s_B + \int_0^t v_B(\tau) d\tau < \quad s_A + \int_0^t (v_A + \underline{a}_A \tau)^+ d\tau \qquad \forall t \geq 0. \tag{3}$$

3.2 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 nT, $a_B(\cdot)$ is chosen identically equal to $\underline{a}_B$, except on the interval $[nT, (n+1)T)$.

3.3 Maximally Aggressive but Safe Strategies

We are now interested in computing strategies $a_B^*(.)$ which are *maximally aggressive* in the set of safe strategies. This is of interest for two reasons. First, as noted earlier in Section 1, aggressive but safe action is needed to ensure that cars exit from the intersection in time. Second, aggressive behavior is also needed to ensure liveness and to ensure high throughput.

Definition 1 (Maximally aggressive strategy) *A* maximally aggressive strategy $a_B^*(\cdot)$ *is a safe strategy that maximizes the distance travelled in each interval* $[nT, (n+1)T)$.

The problem of determining the maximally aggressive strategy can be formulated as:
Maximize $J(a_B(\cdot),T) = \int_{nT}^{(n+1)T} \left(v_B + \int_{nT}^{\tau} a_B(s)ds\right)^+ d\tau$ over $a_B : [nT,(n+1)T) \to [\underline{a}_B, \overline{a}_B]$, subject to the constraint (3).

In order to simplify this, we restrict attention to a *constant control strategy*, where acceleration input is kept constant over this interval, i.e.,

$$a_B(s) = \begin{cases} a_B & nT \le s < (n+1)T, \\ \underline{a}_B & s \ge (n+1)T. \end{cases}$$

This restriction is reasonable, being readily implementable in a digital control setting. The constant accleration a_B^* in $[nT,(n+1)T]$ for the *maximally aggressive constant control input strategy* is the maximal acceleration in $[\underline{a}_B, \overline{a}_B]$ which satisfies the following two conditions derived from (3):

$$K + s_B(nT) + \int_0^t (v_B(nT) + a_B^* \tau)^+ d\tau \le s_A(nT) + \int_0^t (v_A(nT) + \underline{a}_A \tau)^+ d\tau \quad 0 \le t \le T, \text{ and}$$
$$K + s_B(nT) + \int_0^T (v_B(nT) + a_B^* \tau)^+ d\tau + \int_0^t ((v_B(nT) + a_B^* T)^+ + \underline{a}_B \tau)^+ d\tau$$
$$\le s_A(nT) + \int_0^{T+t} (v_A(nT) + \underline{a}_A \tau)^+ d\tau \text{ for } 0 \le t \le \frac{v_B(nT) + a_B^* \cdot T}{-\underline{a}_B}.$$

Here, the first condition corresponds to checking the safety condition for $[nT,(n+1)T)$, while the second condition corresponds to checking safety for all times after $(n+1)T$. We thus arrive at a *piecewise constant input*, $a_B^*(t) = a_{B,n}$ for $nT \le t < (n+1)T$, with $a_{B,n} \in [\underline{a}_B, a_{B,n}^*] \subseteq [\underline{a}_B, \overline{a}_B]$, reminiscent of receding horizon control [4] with the key difference that the horizon is infinite to guarantee perpetual safety.

3.4 Robustness of the Scheme

The collision avoidance strategy designed above is robust to the nominal assumptions that noiseless, undelayed information about the front car is instantly available periodically, or the resort to maximal braking. We can relax these assumptions as below:

- While we have assumed a constant control strategy which involves maximum braking after T seconds, we could instead use some other more "graceful" maneuver such as gradual braking. More precisely, we can replace the constant control strategy $(a_B^*, \underline{a}_B, \underline{a}_B \ldots)$ with the strategy $(a_B^{**}, a_B^{**} - \Delta, a_B^{**} - 2\Delta, \ldots, \underline{a}_B, \underline{a}_B, \ldots,)$. We now need to ensure that this gradual braking strategy satisfies the safety condition (3). This will not affect any of the results above, but will only decrease the maximum admissible acceleration (i.e., we have $a_B^{**} \le a_B^*$), thus increasing the spacing between adjacent cars, as it necessarily must.

- We can handle *aperiodic* sensing by planning over the worst case, longest update interval. Furthermore, we can adaptively change the horizon, according to the current sensing period.

- Sensing jitter, delay in acquiring information and delay in actuation can all be incorporated into the scheme.

  Theorem 2 *Consider a scenario in which the sensor samples at instants $T_1, T_2, \ldots T_k \ldots$ where is $T_k \in (kT - \delta_j, kT + \delta_j)$. Suppose there is a bounded communication delay d_c, with $0 \le d_c \le \delta_c$. Finally, suppose there is a bounded actuation delay d_a, with $0 \le d_a \le \delta_a$. Then, the rear car can ensure worst-case safety by planning over the worst case interval between two successive actuations, i.e., $T + 2\delta_j + \delta_c + \delta_a$.*

  Proof: First, observe that if we consider delayed information from the front car as current information, we are only being more cautious, since the worst case behavior of the front car is maximum braking. The worst case interval between two actuations is $T_{wc} = T + 2\delta_j + \delta_c + \delta_a$, and the rear car must plan for this interval. Hence, at time of actuation, say t_{act}, if a_B^* is chosen so that the acceleration input given by $a_B(s) = a_B^*$ for $t_{act} \le s \le t_{act} + T_{wc}$ and $a_B(s) = \underline{a}_B$ for $s \ge t_{act} + T_{wc}$ maintains the safety condition, we are done.

- *Packet losses* can also be handled, since in the event of nonreceipt of a packet, the rear car can brake at maximum and stay safe with respect to the front car. Alternately, to avoid such sudden braking events, we can restrict attention to constant control strategies where $a_B(s) = a_B$ for $0 \le s \le \beta T$ and $a_B(s) = \underline{a}_B$ for $s > \beta T$, for some integer $\beta > 1$. Thus, we can handle upto $(\beta - 1)$ consecutive packet losses without resorting to maximum braking which is undesirable from the viewpoint of passenger comfort. We have not considered packet errors here since we suppose that the architecture is implemented using a suitable error correcting code.[2]

- *Noisy information* from the sensors, with bounded noise, can be handled by assuming that each sensor measurement is corrupted by the worst case noise. The modified safety condition is

$$K + (s_B + \Delta s_B) + \int_0^t ((v_B + \Delta v_B) + \underline{a}_B \tau)^+ \, d\tau \le (s_A - \Delta s_A) + \int_0^t ((v_A - \Delta v_A) + \underline{a}_A \tau)^+ d\tau \quad \forall t \ge 0.$$

  where $\Delta s_A, \Delta v_A, \Delta s_B, \Delta v_B$ are the magnitudes of the worst case errors in position and velocity for cars A and B, respectively.

[2]We are grateful to the reviewer for comments on this issue

4 Multiple cars on a lane

The solution to the problem of two cars on a lane can be easily extended to the case of several cars on a lane; see Figure 1. This is done by simply following the rule that *each car makes worst case assumptions about the car immediately in front of it.* In the figure, there is an arrow of "responsibility" between any pair of adjacent cars, and the car at the head of the arrow is supposed to make worst case assumptions about the car at the tail of the arrow, and ensure that it does not collide with it. This results in a *distributed solution.*

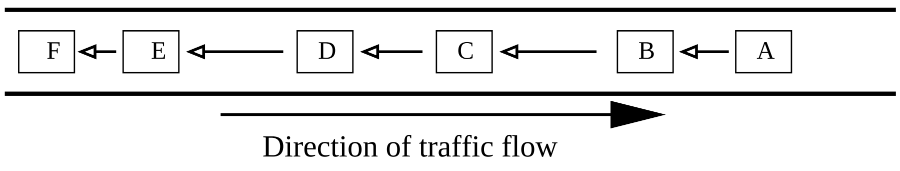


Figure 1: Many cars on a lane

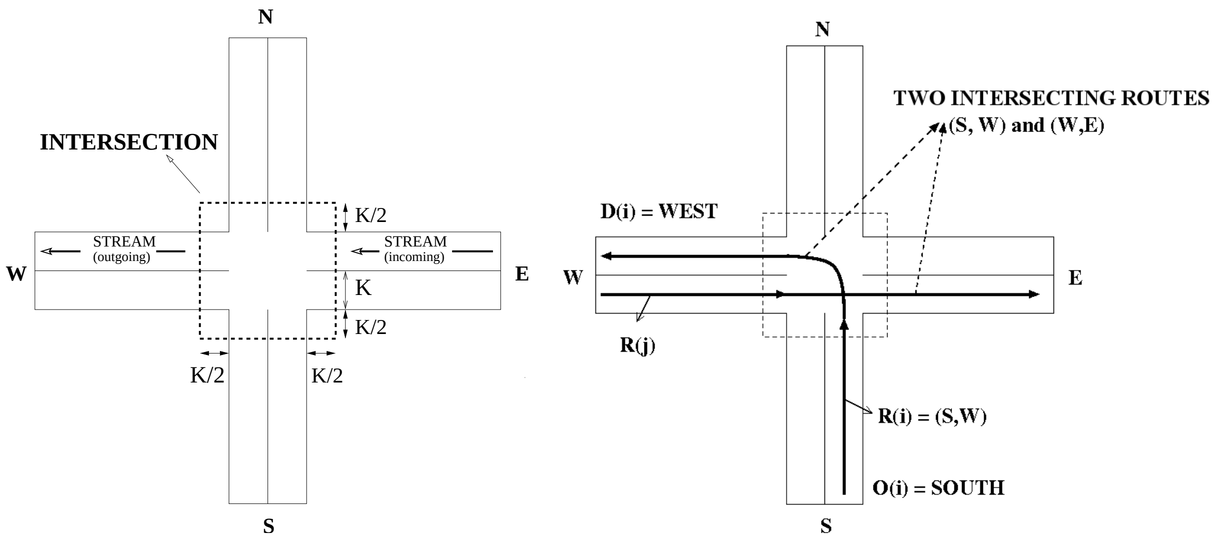
5 Collision avoidance at intersections

Now we turn to the intelligent intersection problem where there are two or more streams of cars crossing at an intersection. We would like to create the "electronic equivalent" of a traffic light, which ensures safety. We need to devise a scheme which ensures that all cars reach their respective destination lanes after crossing the intersection, without collisions, which are defined as occurring when cars are separated by less than $K > 0$ meters. Further, we need to ensure that once cars reach their destination lanes after crossing the intersection, the perpetual safety condition continues to be satisfied for any pair of adjacent cars, so that they can continue along their trajectories. Another issue of interest is liveness and deadlock avoidance.

To solve all these problems, we introduce a "hybrid" architecture. It is based on an interaction between cars and the intersection infrastructure, utilizing time slot assignment by the intersection infrastructure for crossing the intersection, with the cars responsible for distributed safety with respect to collisions as well as timely crossing. It should be noted that safey is handled in a worst case sense, and through distributed actions, which appears to be appropriate to this by issue.

5.1 Description of Intersection

We consider a four road intersection with four incoming and four outgoing roads shown in Figure 2(a). Each road has only one lane in each direction, making for a total of 8 lanes. The incoming and outgoing roads are indexed by the directions N,W,E, and S, as shown. Consider a system of m cars indexed $\{1, 2, 3 \dots m\}$. Car i's acceleration is constrained to lie in $[\underline{a}_i, \overline{a}_i]$ 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

Figure 2: Description of intersection

By *intersection* we refer to the square consisting of the intersection proper as well as an area encompassed by $K/2$ meters along each incoming and outgoing lane; see Figure 2(a). The *route* taken by a car i is described by an ordered pair $R(i) = (O(i), D(i))$, of origin and destination respectively; see Figure 2(b).

Two routes are said to be *intersecting* if they cross each other; e.g., (W, E) and (S, W) are intersecting routes in Figure 2(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, let s_i^α and s_i^β denote the position coordinates of the beginning and end of the intersection, respectively, along $R(i)$.

Implicit in our formulation is the assumption that every car has collision avoidance technology installed. Hence, we do not address the problem of how an equipped car will avoid collisions with unequipped cars. We assume perfect clock synchronization among the cars. If this is not true, a guard band will be required in order to handle clock discrepancies. At time zero, we suppose that all cars are at least a braking distance away from the intersection and that each car is in the safety set of its lead car. We treat all routes as straight lines, and use simple kinematic equations to describe the one-dimensional motion of each car.

5.2 A Hybrid Architecture

We propose a *hybrid architecture* for collision avoidance at intersections. It consists of an interaction between the intersection infrastructure and the cars, that is, an interaction between a centralized component and distributed agents.

The intersection infrastructure will function 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 assigned 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. Rather, a car is only required to be outside the intersection at all time instants that are not in its time slot.

This brings us to the roles of the cars in the architecture. Given a time slot, a car has to determine not only if it can go through the intersection in the assigned time slot, but also if it can enter its destination lane *without* violating perpetual safety with respect to cars already on that lane.If it cannot, then it prepares to come to a halt before the intersection and requests a new slot. All this is done by the *Intersection Crossing Algorithm* implemented by each car in the system.

Our goal is to design the overall system so as to ensure that all cars can get through the intersection safely and to ensure "liveness" of traffic flow. We now formulate and specify the different aspects of the architecture, culminating in a theorem which proves worst case safety and liveness.

5.2.1 The 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. This will result 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 can therefore potentially modify the time slot allotted to each of the cars every T seconds.

We call a maximal contiguous period of time $[kT, mT)$ during which the time slot allotted to car i remains frozen, as an *allocation epoch* for car i. We will say that a car i *conforms* to the slot $\sigma(i)$ if it never occupies the intersection at any time outside $\sigma(i)$. Extending this, we will say that a car

i conforms to the slot sequence $\{\sigma^{(0)}(i), \sigma^{(1)}(i), \ldots\}$ if, in each interval $[nT, (n+1)T)$, it conforms to $\sigma^{(n)}(i)$ in the sense that it is not in the intersection at any time in the interval $[nT, (n+1)T)$ if $\sigma^{(n)}(i)$ does not permit it.

We define a *strict partial ordering relation* "$\prec$" on the set of slots according to their slot start times; that is, we say, $\sigma(i) \prec \sigma(j)$ if and only if $t_{start}(i) < t_{start}(j)$.

Definition 2 (Admissible Time Slot Assignment) *We say that a time slot assignment σ is* admissible *if it satisfies the following properties:*

- *For any two cars i and j, if $R(i) \;\mathcal{I}\; R(j)$, then $\sigma(j) \cap \sigma(i) = \emptyset$ for all n. This ensures that two cars with intersecting routes have non-intersecting slots.*
- *For any car i, define $J(i) := \{j : D(j) = D(i), O(j) \neq O(i)\}$. Then we must have $\sigma(i) \prec \sigma(j)$ or $\sigma(j) \prec \sigma(i)$, for all i and for all $j \in J(i)$.*

Note that merely having an admissible slot assignment is not sufficient for collision avoidance, since cars on non-intersecting routes may still collide. E.g., (E, N) and (S, N).

Definition 3 (Failsafe Maneuver) *For each car i at time nT, we will maintain an* open loop sequence *of inputs called the* failsafe maneuver*, denoted by $\{a_i^{F,n}(k)\}_{k \geq n}$. This specifies at time nT an open-loop future trajectory for car i, which serves as an infinite horizon open-loop contingency plan that it can thereafter follow and stay perpetually safe. We initialize the failsafe maneuver for each car to be maximum braking at time zero.*

In the sequel, we will discuss some properties to be satisfied by the failsafe maneuver, which are crucial to establishing systemwide safety.

Definition 4 (Admissible Slot Reallocation Policy) *Suppose car i is beginning a new allocation epoch at time nT, i.e., $\sigma^{(n-1)}(i) \neq \sigma^{(n)}(i)$. We say that a slot reallocation policy is* admissible *if it satisfies the following conditions:*

(i) The reallocated slot must be in the future, i.e., $t^{(n)}_{start}(i) \geq nT$.

(ii) If the available failsafe maneuver at time nT is implemented at time nT, then car i will come to a stop before the intersection.

(iii) Slot reallocation cannot be done too early; we need $nT \geq t^{(n-1)}_{end}(i) - \tau_{max} + T$, where τ_{max} is the length of time enough for any car starting from rest to get through the intersection. This ensures that cars do not simply postpone their slots.

(iv) *Consider the set of cars* $\{j : R(j) = R(i), t_{end}^{(n-1)}(j) > nT\}$*; let* ζ *be the car in this set which is earliest in the ordering* $\prec$*(this need not be unique). Then we must have,*

$$\sigma^{(n-1)}(k) \prec \sigma^{(n)}(i) \text{ for all } \{k \in J(i) : \sigma^{(n-1)}(k) \prec \sigma^{(n-1)}(\zeta)\}. \tag{4}$$

If such a car ζ *does not exist, we must have* $\sigma^{(n-1)}(k) \prec \sigma^{(n)}(i)$ *for all* $k \in J(i)$*.*

This is saying that the slot $\sigma^{(n-1)}(\zeta)$ *is the earliest slot available for car* i *on route* $R(i)$*. For cars with slots later than the slot of car* ζ*, they will not be affected by the addition of another earlier slot. However, one must ensure that the cars whose slots are ordered earlier than car* ζ *are also ordered earlier than the new slot of car* i*, which is ensured by (4).*

In the sequel, we shall assume throughout that we have a sequence of admissible time slot assignments and an admissible slot reallocation policy. A candidate admissible slot assignment is to assign time slots using a sequential greedy algorithm. We assign to each subsequent car, a time slot of predefined length, which begins immediately after the last timeslot ends. More intelligent and efficient ways to assign timeslots are described in Section 6.

5.2.2 Three primitive 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 safe behavior by cars. Such an approach suggesting a language for composing motion is in [21].

We adopt a discrete-time viewpoint and suppose that information about other cars in the system refreshes periodically every T seconds. This service is to be provided over an underlying wireless communication layer. We denote by $\{s_i(n)\}$, $\{v_i(n)\}$ the sampled position and velocity of car i in car i's coordinate system, and by $\{a_i(n)\}$ its piecewise constant input, in the time interval $[nT, (n+1)T)$.

We introduce some notation to describe the three primitive maneuvers. We say that a *car* j *is on car* i*'s route at time* nT, if either

(i) $O(j) = O(i)$ and car j is located strictly less than K meters from any point on $R(i)$ at time nT,

(ii) Or, $D(j) = D(i)$ and car j is located at a point on $R(i)$ at time nT and $t_{end}^{(n)}(j) \leq nT$.

Thus it accounts for cars from the same origin as car i which are still within K meters of $R(i)$ and cars from other origin streams which have crossed the intersection and are now on $R(i)$.

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 a *virtual* position for car j at $s_{ji}(n) := s_i^\alpha + K$ and a virtual speed $v_{ji}(n) := 0$. If car j is on i's route at time t, we define virtual position and virtual speed as follows. If $O(j) = O(i)$, we set $s_{ji}(n) := s_i^\alpha + s_j(n) - s_j^\alpha$ and $v_{ji}(n) := v_j(n)$; while if $D(j) = D(i)$, we set $s_{ji}(n) := s_i^\beta + s_j(n) - s_j^\beta$ 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 nT. If there is no such 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.

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 m 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_j^α, after which j is no longer on i's route. Otherwise $K_{ji} = K$.

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 future time. We note that such a maneuver may be infeasible for certain values of s_{park}. It is feasible if the braking distance for car i at time nT satisfies $s_{park} - s_i(n) \geq \frac{(v_i(n))^2}{-2\underline{a}_i}$. An extremal parking maneuver of specific interest is the *minimum-time 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 $s_{ji}(t) - s_i(t) \geq K$ for all $t \geq nT$ under worst case assumptions (viz., maximum braking) on car j, and results in car i parking at $s_{ji}(n) + \frac{v_{ji}(n)^2}{-2\underline{a}_j} - K$. An extremal tailing maneuver of interest is the *minimum-time tailing maneuver* which stops in minimum time.

Lemma 1 *Consider two cars i and j with $R(i) = R(j)$. Let us suppose that $s_{ji}(n) \geq s_i(n) + K$ and $\underline{a}_j \leq \underline{a}_i$. Then, at time nT, the minimum-time tailing maneuver for car i behind car j is exactly the minimum-time parking maneuver stopping at $s_{ji}(n) + \frac{v_{ji}^2(n)}{-2\underline{a}_j} - K$.*

Proof: First, observe that every tailing maneuver is a parking maneuver stopping at $s_{ji}(n) + \frac{v_{ji}^2(n)}{-2\underline{a}_j} - K$, by definition. For the converse, consider a parking maneuver stopping at $s_{ji}(n) + \frac{v_{ji}^2(n)}{-2\underline{a}_j} - K$, and let $s_i(t)$ and $s_{ji}(t)$ denote the positions of cars i and j as a function of time. We need to show that, under worst case assumptions (viz., maximum braking) on car j, the distance between the two cars, $f(t) := s_{ji}(t) - s_i(t) \geq K$ for all $t \geq nT$. Suppose not. Then $f(t)$ attains a minimum at some t^* with $f(t^*) < K$. Then the first order necessary condition for minimality of $f(t^*)$ is $v_i(t^*) = v_j(t^*)$ and the second order condition is $\underline{a}_j > a_i$. However, since $\underline{a}_j \leq \underline{a}_i$, we must have $t^* = nT$ or $t^* = t_{park}$, the time of parking of car i. However, by assumption, $f(nT) > K$ and the fact that it is a parking maneuver stopping at $s_j(n) + \frac{v_j^2(n)}{-2\underline{a}_j} - K$ implies $f(t_{park}) > K$. Therefore, the sets of parking maneuvers and tailing maneuvers are exactly equal. Hence, minimum-time maneuvers in each set must also be exactly the same. □

5.2.3 Downstream cars: Real and virtual

Our approach to guaranteeing safety is based on each car taking the responsibility of avoiding collisions with the other cars that are "ahead" of them. However due to the fact that there is an intersection, and that paths intersect, we define more precisely the notion of "downstream cars."

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 all of the following:

(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 $\underline{a}_0 = 0$ (to ensure the set is nonempty).

We now define one car, the *immediate downstream car*, that car i is responsible 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:

$$\begin{aligned} s_{d(i)}(n) &= \min_{j \in \mathcal{D}(i,nT)} s_{ji}(n), \\ s_{d(i)}(n) + \frac{v_{d(i)}^2(n)}{-2\underline{a}_i} - K &= \min_{j \in \mathcal{D}(i,nT)} \left\{ s_{ji}(n) + \frac{v_{ji}^2(n)}{-2\underline{a}} - K_{ji} \right\}, \end{aligned} \tag{5}$$

where $\underline{a} = \min \underline{a}_j$ over all $j \in \{1, 2, \ldots m\}$. Further, we define $\underline{a}_{d(i)}(n) := \underline{a}_i$. Since $\mathcal{D}(i, nT)$ is nonempty for all i and all n, the immediate downstream car is well defined.

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 2 *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*[3]

$$\left(s_{d(i)}(nT) + \int_0^\tau (v_{d(i)}(nT) + \underline{a}_i s)^+ ds - K \right) \leq s_{ji}(nT + \tau) - K_{ji} \quad \forall j \in \mathcal{D}(i, nT), \forall \tau \geq 0. \tag{6}$$

[3]Here, we abuse notation slightly by using $s_j(t)$, $v_j(t)$ to denote the continuous time evolution of these quantities.

Proof: Consider the expression

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

This is the separation distance between car j and car $d(i)$ as a function of time, under worst case assumptions on cars j and $d(i)$. We claim that we must have $f_j(\tau) \geq 0$ for all $\tau \geq 0$ and all $j \in \mathcal{D}(i, nT)$. Suppose not. Then for some car $\bar{j}$ and some τ^*, the worst case trajectories of car $\bar{j}$ and car $d(i)$ must cross at τ^*. Hence, we must have $f_{\bar{j}}(\tau^*) = 0$ and $v_{\bar{j}i}(\tau^*) < v_{d(i)}(\tau^*)$. However, since $\underline{a} \leq \underline{a}_i$, this would mean that

$$\begin{aligned}
0 > f(\infty) &= \left(s_{\bar{j}i}(nT) + \frac{v_{\bar{j}i}^2(nT)}{-2\underline{a}} - K_{ji}\right) - \left(s_{d(i)}(nT) + \frac{v_{d(i)}^2(nT)}{-2\underline{a}_i} - K\right) \\
&\geq \min_{j \in \mathcal{D}(i,nT)} \left(s_{ji}(nT) + \frac{v_{ji}^2(nT)}{-2\underline{a}} - K_{ji}\right) - \left(s_{d(i)}(nT) + \frac{v_{d(i)}^2(nT)}{-2\underline{a}_i} - K\right) \\
&= \left(s_{d(i)}(nT) + \frac{v_{d(i)}^2(nT)}{-2\underline{a}_i}\right) - \left(s_{d(i)}(nT) + \frac{v_{d(i)}^2(nT)}{-2\underline{a}_i}\right) = 0,
\end{aligned}$$

which is a contradiction. Hence we must have $f_j(\tau) \geq 0$ for all $\tau \geq 0$ and all $j \in \mathcal{D}(i, nT)$. To complete the proof, we have $0 \leq f_j(\tau) \leq (s_{ji}(nT + \tau) - K_{ji}) - \left(s_{d(i)}(nT) + \int_0^\tau (v_{d(i)}(nT) + \underline{a}_i s)^+ ds - K\right)$.

5.2.4 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.

P1: Each car i conforms to its slot sequence $\{\sigma^{(0)}(i), \sigma^{(1)}(i), \ldots\}$.

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.

We now show that these two properties ensure perpetual collision avoidance. Hence, in the sequel, it suffices to establish just these two properties as far as safety is concerned.

Lemma 3 *Given a sequence of admissible time slot assignments $\{\sigma^{(0)}(\cdot), \sigma^{(1)}(\cdot), \ldots\}$, 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.*

Proof: First, two cars i and j with $R(i)\ \mathcal{I}\ R(j)$ can collide only if both cars are in the intersection. However, given an admissible time slot assignment $\sigma^{(n)}(\cdot)$ at each time nT, we have $\sigma^{(n)}(i) \cap \sigma^{(n)}(j) = \emptyset$ for all n. Using $P1$, we are guaranteed that i and j cannot collide.

For two cars i and j with non-intersecting routes, if $O(i) \neq O(j)$ and $D(i) \neq D(j)$, their routes are physically separated by distance K and we are done. If $O(i) = O(j)$ or $D(i) = D(j)$, there are two possibilities. If car i is on j's route at time nT or vice versa, we must have $s_{ji}(n) \geq s_i(n)$ or $s_{ij}(n) \geq s_j(n)$, and consequently we have $i \in \mathcal{D}(j, nT)$ or $j \in \mathcal{D}(i, nT)$, respectively. Using $P2$, we are guaranteed that i and j cannot collide. At time nT, if i and j are not on the same route, they can collide in the interval $[nT, (n+1)T)$ only if $t_{end}^{(n)}(i) > nT$ and $t_{end}^{(n)}(j) > nT$. Under an admissible slot assignment $\sigma^{(n)}(\cdot)$, due to the strict ordering between $\sigma^{(n)}(i)$ and $\sigma^{(n)}(j)$, we have $i \in \mathcal{D}(j, nT)$ or $j \in \mathcal{D}(i, nT)$. Using $P2$, we are guaranteed that i and j cannot collide in the interval $[nT, (n+1)T)$ for all n. $\square$

There is, of course, a trivial algorithm that ensures perpetual collision avoidance; immediately after the initial slot assignment $\sigma^{(0)}$ has been made, each car brakes at maximum and stops before the intersection (recall that cars are at least a braking distance away from the intersection at time zero). However, such an algorithm is of no value in practice since eventually all traffic comes to a halt; that is, it does not ensure *liveness*. Hence we will also address the issue of liveness below.

5.2.5 Update algorithm for Failsafe Maneuver

Our scheme for perpetual collision avoidance is predicated upon each car always having a so called *failsafe maneuver* at every time, which it can apply from that time forward in an open-loop fashion, and still guarantee safety. This failsafe maneuver will be updated at every time step. It can be thought of as a "rolling baseline contingency plan."

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 $(n+1)T$. We provide an elaborate proof of its safety in Section 5.2.7.

Algorithm for update of Failsafe Maneuver

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

Suppose car i executes the 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 3)[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 allocated slot. In*

[4]In particular, one can choose the maximally aggressive strategy, as described in Section 3.3

particular, if, even under maximum braking, car i will penetrate the intersection at the beginning of its time 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. [5]

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

Car i executes the current failsafe maneuver; i.e., it 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. From Lemma 1, this is equivalent to a minimum time parking maneuver, which can be easily calculated as a bang-bang control. 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 executes the current failsafe maneuver; i.e., it 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 executes the MTT maneuver; i.e., it 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 a convex combination of 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:

$$a_i^*(k) = \lambda . a_i^{MTT}(k) + (1-\lambda).\underline{a}_i \qquad \forall k \in \{n, \ldots, \frac{t_{start}^{(n)}(i)}{T} - 1\}.$$

[5]We note here that this condition is in fact never satisfied. It has only been included here for completeness of the proof.

where $\lambda = \frac{s_i^{\alpha} - s_i^{MB}(\frac{t_{start}^{(n)}(i)}{T})}{s_i^{MTT}(\frac{t_{start}^{(n)}(i)}{T}) - s_i^{MB}(\frac{t_{start}^{(n)}(i)}{T})}$ is the relative contribution of the MTT maneuver. It results in $v_i^*(k) = \lambda.v_i^{MTT}(k) + (1-\lambda).v_i^{MB}(k)$ and $s_i^*(k) = \lambda.s_i^{MTT}(k) + (1-\lambda).s_i^{MB}(k)$ for all $k \in \{n, \ldots, \frac{t_{start}^{(n)}(i)}{T}\}$

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 executes the current failsafe maneuver; i.e., it 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 $a_i^{F,n+1}(k) = a_i^*(k) \quad \forall k \geq n+1$.

It is worth noting that the failsafe maneuver is an open-loop sequence. Since it is safe, it can be used even if sensors and communication fail in the network, a critical property.

5.2.6 The Intersection Crossing Algorithm

Now we are ready to specify the algorithm for cars in the hybrid architecture, the *Intersection Crossing Algorithm*:

- 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 subsequent time step nT, car i has two choices. It can

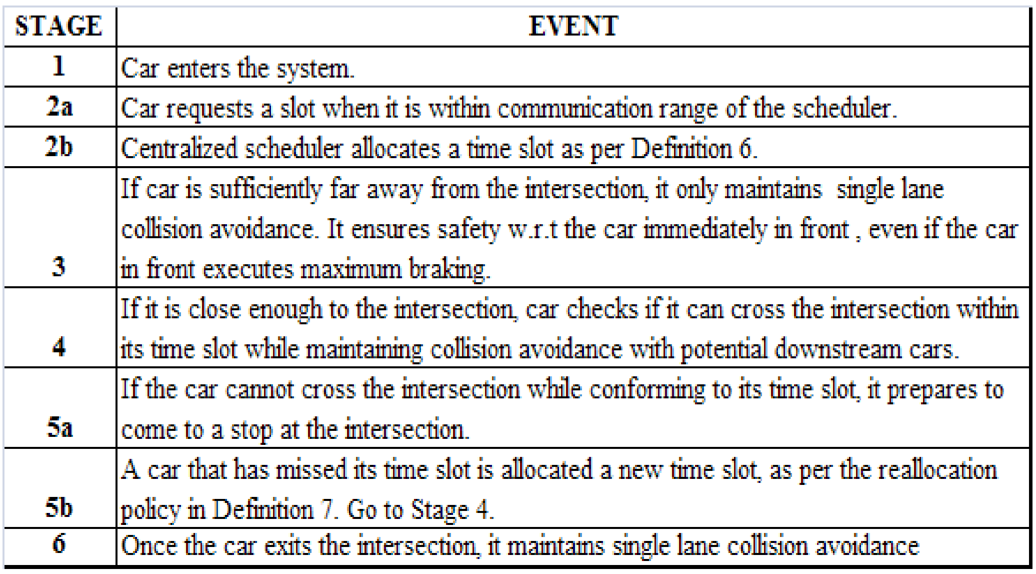
  (i) Choose the current input $a_i(n)$ and update the failsafe maneuver by running the failsafe maneuver update algorithm using information from other cars at time nT.

  (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" available at time nT, possibly due to nonreceipt of information from other cars at time nT.

- 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, since car i can always stop before the intersection (now at $+\infty$) under the Modified Maximum Braking maneuver.[6]

With this algorithm, we have now provided a complete description of individual car behavior. We now proceed to prove systemwide safety and liveness.

[6]Upon exit from an intersection, we could set the values of s_i^{α} and s_i^{β} to correspond to the next intersection. This will allow the treatment of systems of many intersections.



STAGE	EVENT
1	Car enters the system.
2a	Car requests a slot when it is within communication range of the scheduler.
2b	Centralized scheduler allocates a time slot as per Definition 6.
3	If car is sufficiently far away from the intersection, it only maintains single lane collision avoidance. It ensures safety w.r.t the car immediately in front , even if the car in front executes maximum braking.
4	If it is close enough to the intersection, car checks if it can cross the intersection within its time slot while maintaining collision avoidance with potential downstream cars.
5a	If the car cannot cross the intersection while conforming to its time slot, it prepares to come to a stop at the intersection.
5b	A car that has missed its time slot is allocated a new time slot, as per the reallocation policy in Definition 7. Go to Stage 4.
6	Once the car exits the intersection, it maintains single lane collision avoidance

Figure 3: Operation of Intelligent Intersection

5.2.7 Safety of the Intersection Crossing Algorithm

In order to establish safety, we need to carefully 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;

$$\pi^{n|x(n-1)} := \{p_1^{n|x(n-1)}, p_2^{n|x(n-1)}, \ldots, p_m^{n|x(n-1)}\} \text{ where } p_i^{n|x(n-1)} \equiv \{a_i^{F,n}(k)\}_{k\geq n}.$$

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 $(\underline{x(n)}, \pi^{n|x(n-1)}, \sigma^{(n)}) \in \mathcal{X} \times \Pi \times \Sigma$. The underlining of $x(n)$ is a convention that we will use to indicate that this information is *private*, and that the cars have not yet communicated this information to each other at time nT. The superstate evolves in four steps as follows:

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 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 systemwide choice $u(n) \in U(n)$.

Step C: Consequently, the physical state of the system evolves according to the kinematics, from $x(n)$ to $x(n+1)$.

Step D: Finally, the time slot assignment is updated from $\sigma^{(n)}$ to $\sigma^{(n+1)}$.

$$(\underline{x(n)}, \pi^{n|x(n-1)}, \sigma^{(n)}) \overset{StepA}{\longrightarrow} (x(n), U(n) \times \pi^{n+1|x(n)}, \sigma^{(n)}) \overset{StepB}{\longrightarrow} (x(n), u(n) \in U(n), \pi^{n+1|x(n)}, \sigma^{(n)})$$
$$\overset{StepC}{\longrightarrow} (\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n)}) \overset{StepD}{\longrightarrow} (\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n+1)}). \quad (7)$$

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 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 $\sigma^{(n)}(i)$ 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 3 *Suppose we have a sequence of admissible time slot assignments* $\{\sigma^{(0)}(\cdot), \sigma^{(1)}(\cdot), \ldots\}$ *and an admissible slot reallocation policy, 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 $a_i(n)$ *by the method 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 an admissible slot reallocation policy, we have $(\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n+1)}) \in \mathcal{A}$*. That is, under the maneuver* $p_i^{n+1|x(n)}$ *for car i at time* nT*, we have:*

$(C1^{(n+1)})$: Car i conforms to the (frozen) time slot $\sigma^{(n+1)}(i)$.

$(C2^{(n+1)})$: Car i does not collide with any car in $\mathcal{D}(i, (n+1)T)$ *under worst case assumptions on cars in* $\mathcal{D}(i, (n+1)T)$.

Proof: **(a)** It is enough to show that for each car i, under any maneuver in the set $\{U_i(n) \times \pi^{n+1|x(n)}\}$, conditions $C1^{(n)}$ and $C2^{(n)}$ are satisfied. In the intersection crossing algorithm, if car i chooses to simply execute the failsafe maneuver at time nT, the result is immediate. However, if car i executes the failsafe maneuver update algorithm, we need to do a step-by-step analysis.

Observe that in Steps 2, 3, and 5, the proof is trivial. We now look at the other steps.

Step 1: First, note that since each car has a failsafe maneuver at time nT, we can definitely find $a_i(n)$ such that car i stays in the safety set of its lead car. Since $s_i^{MMB}(\infty) < s_i^{\alpha}$, the prescribed maneuver $\{a_i(n), p_i^{n+1|x(n)}\}$ ensures that car i conforms to the slot $\sigma^{(n)}(i)$, which proves $(C1^{(n)})$. Further, under this maneuver, car i does not collide with its lead car $l(i, nT)$. Suppose, to the contrary, car i does collide with some other car $j \in \mathcal{D}(i, nT)$, where $j = l^a(i, nT)$ for some $a > 1$. Now, since $(x(n), \pi^{n|x(n-1)}, \sigma^{(n)}) \in \mathcal{A}$, we deduce that $l^{a-1}(i, nT)$ is in the safety set of $l^a(i, nT)$, $l^{a-2}(i, nT)$ is in the safety set of $l^{a-1}(i, nT)$, and so on. Hence, at time nT, $l(i, nT)$ is in the safety set of $l^a(i, nT)$, which means that if $l(i, nT)$ performs maximum braking, it does not collide with $l^a(i, nT)$. This contradicts our supposition that car i collided with $l^a(i, nT)$.

Step 4: Under Step 4 of the failsafe maneuver update algorithm, the following is true:

$$s_i^{MAT}\left(\frac{t^{(n)}_{end}(i)}{T}\right) > s_i^{\beta} \quad \text{and} \quad s_i^{MAT}\left(\frac{t^{(n)}_{start}(i)}{T}\right) \leq s_i^{\alpha}.$$

Hence we already see that if car i executes the MTT maneuver behind $d(i)$, it conforms to the slot $\sigma^{(n)}(i)$, proving $(C1^{(n)})$. Further, from Lemma 2 we know that avoiding collisions with the worst case trajectory of the immmediate downstream car is a sufficient condition for avoiding collisions with the worst case trajectory of every car $j \in \mathcal{D}(i, nT)$. Under the MTT maneuver behind $d(i)$, we have $s_{ji}(t) - s_i(t) \geq K_{ji}$, $\forall j \in \mathcal{D}(i, nT)$, under worst case assumptions on cars in $\mathcal{D}(i, nT)$. This consequently ensures a minimum *physical separation* of K m, and $(C2^{(n)})$ is ensured.

Step 6: Observe that the maneuver $\{a_i^*(k)\}$ has been specifically constructed to ensure that $s_i^*(\frac{t^{(n)}_{start}(i)}{T}) = s_i^{\alpha}$. Further, under Step 6 of the failsafe maneuver update algorithm, we have $s_i^*(\frac{t^{(n)}_{end}(i)}{T}) \geq s_i^{\beta}$, and this establishes that under this maneuver, car i conforms to the slot $\sigma^{(n)}(i)$, which proves $(C1^{(n)})$. Now, notice that $\{a_i^*(k)\}_{k \geq n}$ is a parking maneuver behind $s_{d(i)}(n) + \frac{v_{d(i)}(n)^2}{-2\underline{a}_i}$, and hence a tailing maneuver behind car $d(i)$ (see proof of Lemma 1). Using Lemma 2 and exactly the same arguments as in Step 4, $(C2^{(n)})$ is ensured.

(b) From (a), we see that if each car chooses its acceleration input at time nT according to the intersection crossing algorithm, then the following are true: First, each car i conforms to the time slot $\sigma^{(n)}(i)$ in the time interval $[nT, (n+1)T)$, and, second, car i does not collide with any car in $\mathcal{D}(i, nT)$ in the time interval $[nT, (n+1)T)$. From Lemma 3, under an admissible time slot assignment $\sigma^{(n)}(\cdot)$, we have systemwide safety in the time interval $[nT, (n+1)T)$. Further, at time $(n+1)T$, the resulting superstate $(\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n)}) \in \mathcal{A}$.

(c) Given a car i, we have two cases. If $\sigma^{(n+1)}(i) = \sigma^{(n)}(i)$, then $(C1^{(n+1)})$ is equivalent to $(C1^{(n)})$, which is satisfied since $(x(n+1), \pi^{n+1|x(n)}, \sigma^{(n)}) \in \mathcal{A}$. From (a), we already know that, under the maneuver $p_i^{n+1|x(n)}$, car i does not collide with any car in $\mathcal{D}(i, nT)$. Further, since there were no collisions in $[nT, (n+1)T)$, $\mathcal{D}(i, (n+1)T) \setminus \mathcal{D}(i, nT)$ can only consist of cars not on i's route at time nT. However, due to the admissible slot reallocation policy (iii)-(iv), $\mathcal{D}(i, (n+1)T) \setminus \mathcal{D}(i, nT) \neq \emptyset$ only if

- Car i has not yet entered the intersection at $(n+1)T$, and
- There is a car $j \in \mathcal{D}(i, nT)$ which is not on i's route at time nT.

Hence, if $\mathcal{D}(i,(n+1)T)\backslash\mathcal{D}(i,nT) \neq \emptyset$, then under the failsafe maneuver $p_i^{n+1|x(n)}$, car i must stop before the intersection, which ensures that car i does not collide with any car in $\mathcal{D}(i,(n+1)T) \setminus \mathcal{D}(i,nT)$. Hence $(C2^{(n+1)})$ is ensured.

Now, let us consider the case where $\sigma^{(n+1)}(i) \neq \sigma^{(n)}(i)$. From the admissible slot reallocation policy (i), we know that under the failsafe maneuver $p_i^{n+1|x(n)}$, car i stops before the intersection, which automatically ensures $(C1^{(n+1)})$. Further, since $\mathcal{D}(i,(n+1)T) \setminus \mathcal{D}(i,nT)$ can only consist of cars not on i's route at time nT, $(C2^{(n+1)})$ is trivially satisfied. $\square$

5.3 Perpetual Systemwide Safety

It remains to prove the main result, the *perpetual systemwide safety* of the hybrid architecture. We make the following assumptions:

(A1) We have a sequence of admissible time slot assignments $\{\sigma^{(0)}(\cdot), \sigma^{(1)}(\cdot), \ldots\}$ and an admissible slot reallocation policy.

(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 4 ***Systemwide Safety of the Intersection Crossing Algorithm***
Under assumptions (A1) and (A2), if each car follows the Intersection Crossing Algorithm as described above, there is perpetual collision avoidance for the whole system of cars.

Proof: At time zero, for each car i, the failsafe maneuver $\{a_i^{F,0}(k)\}_{k\geq 0}$ is set to be the maximum braking maneuver. Using (A2), it is easy to check that under this maneuver, car i conforms to its (frozen) time slot $\sigma^{(0)}(i)$, and car i does not collide with any car in the set $\mathcal{D}(i,0)$. Hence, the superstate $(\underline{x(0)}, \pi^{0|x(-1)}, \sigma^{(0)}) \in \mathcal{A}$. We proceed by induction. Let us suppose that $(\underline{x(n)}, \pi^{n|x(n-1)}, \sigma^{(n)}) \in \mathcal{A}$. From Theorem 3, we know that, if all cars follow the Intersection Crossing Algorithm, then there is systemwide safety in the interval $[nT, (n+1)T)$, and also that at time $(n+1)T$, the superstate $(\underline{x(n+1)}, \pi^{n+1|x(n)}, \sigma^{(n+1)}) \in \mathcal{A}$. This completes the induction. $\square$

5.4 Liveness of the Intersection Crossing Algorithm

In addition to safety, it is also important to ensure that the system does not fall into deadlock.

Definition 5 (Liveness) *We say that the system is live if for any given finite system of cars, every car crosses the intersection.*

Liveness requires that cars utilize the freedom given by the Intersection Crossing Algorithm to make progress on their routes. We can ensure liveness under the following stronger assumptions:

(B1) 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 $0 < \alpha \leq 1$. (Note that the other steps are already designed to be maximally aggressive.)

(B2) 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.

(B3) For each car, a new slot is reallocated within a finite time Δ if it misses its earlier slot.

Theorem 5 (Liveness of the Intersection Crossing Algorithm) *Under conditions (A1,A2) and (B1-B3), if each car in the system follows the Intersection Crossing Algorithm as described above, then there is liveness.*

Proof: First, we claim that once a car crosses the intersection, it continues to make positive progress, such that, given any rectangle enclosing the intersection, the car crosses this rectangle in finite time. Clearly, due to condition (B1), the lead car on each destination lane will cross any such rectangle in finite time. We proceed by induction. Suppose the k upstream cars on a destination lane cross any given rectangle in finite time. If the $(k+1)^{th}$ car never crosses some given rectangle, then the distance between the k^{th} car and the $(k+1)^{th}$ car is increasing with time and unbounded. Hence, at some point maximum acceleration is feasible for the $(k+1)^{th}$ car, and, due to condition (B1), car is forced to apply an acceleration which is bounded away from zero. This contradicts the assumption that the $(k+1)^{th}$ car does not cross the given rectangle.

Suppose there is some subset of cars which never cross the intersection. Consider one such car i at the head of some origin lane. Car i never executes Steps 4 or 6 of the update algorithm since both these result in crossing the intersection. It alternates between executing Step 1 of the update algorithm and executing the failsafe maneuver (Steps 2,3, and 5). However there can only be finitely many switches between the two modes, and car i comes to rest in bounded time, at the edge of the intersection[7]. Since all cars on destination lanes are α-aggressive, there exists some time t_{clear}, after

[7]Actually, car i comes to a stop no further than $\frac{\overline{a}_i^2 T^2}{2}\left(\frac{1}{\overline{a}_i} - \frac{1}{\underline{a}_i}\right)$ from the intersection

which the lead car of car i will be sufficiently far away from the intersection. By this we mean that car i can apply maximum acceleration, traverse the intersection, and apply maximum braking upon exiting the intersection, and still not collide with the lead car. From condition (B3), we have a bounded interval between missing a slot and being assigned a new slot. Hence, within a bounded number of slot reallocations, we must have some $\sigma^*(i)$ assigned to car i, such that $t^*_{start}(i) \geq tclear$.

From condition (B2), each slot for car i must be at least τ_{max} seconds long. At time $t^*_{end}(i) - \tau_{max}$, from condition (B2), the set of potential downstream cars for a car i can only consist of cars on i's route, which are sufficiently far away since $t^*_{start}(i) > t_{clear}$. Since τ_{max} is a sufficient length of time for any car to get through the intersection, the failsafe maneuver update algorithm can construct a minimum time tailing maneuver behind the immediate downstream car. From the admissible slot reallocation policy (ii), the next reallocation cannot happen before $t^*_{end}(i) - \tau_{max} + T$. However,under the available failsafe maneuver at $t^*_{end}(i) - \tau_{max} + T$, car i will actually cross the intersection and the reallocation is not permitted. Hence car i does manage to get through the intersection within a bounded interval of time $\square$.

6 Performance Evaluation

The algorithm and architecture described above ensure systemwide safety and liveness, while still providing freedom in the design space for policies to enhance throughput or reduce delay. In this section, we attempt to explore this design space and devise schemes for enhancing performance. In particular, we would like to find admissible time slot assignments which satisfy conditions (A1)-(A2), and locally maximize an appropriate performance metric.

In order to find an efficient slot assignment, we use a descent approach based on forward simulation. Given that the mechanical system of cars is slowly moving, it may even be possible for a faster than real time simulation methodology to be used as a “model predictive” controller to enhance system performance. We run the simulation with some arbitrary initial slot assignment and record the average travel time and the final slot assignment. We then perform two operations on the final slot assignment obtained:

SWAP: Given any car, identify the car on a conflicting route with the immediately earlier slot, and swap these two slots.

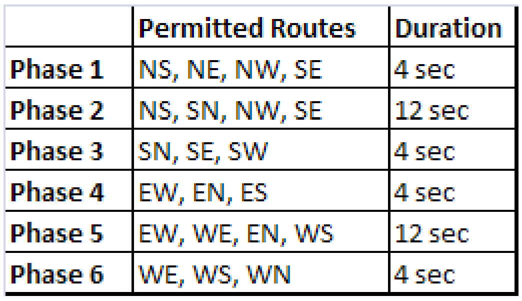
SQUEEZE: Given any car, ensure that its slot is squeezed as close as possible to the slot occupied by the immediately preceding car which has a conflicting route or shares the same origin.

Using these operations, we obtain modified time slot assignments, which are evaluated again by forward simulation, and accepted if they result in lower average travel time. The algorithm terminates when all such modifications result in higher average travel time. The feasibility of the descent approach depends on the computational ability to run faster than real-time simulations. We consider two candidate behaviors, maximally aggressive behavior and just-in-time behavior where cars plan to arrive just in time for their slots.

We have built a simulator of the entire system in MATLAB. The performance metric under consideration is the average time taken by a system consisting of m cars to travel from 200 m away from the start of the intersection to 200 m away from the end of the intersection. The routes of all cars are independent and identically distributed with probability 1/6 for each of the four straight line routes, and 1/24 for each of the eight turning routes. All cars start from at least 200 m away from the start of the intersection, with each subsequent car being positioned at a distance of $(K+\ exp(\lambda))$ behind its lead car with $K = 6$ m, where $exp(\lambda)$ is a random variable that is exponentially distributed with mean $\frac{1}{\lambda}$. All cars start at maximum velocity equal to 25 m/s, and are assumed to have equal braking power equal to -3.5 m/s^2, for simplicity. A tangible measure of the load of the intersection is the average number of cars entering the system per second, calculated as $\frac{4.v_M}{K+\frac{1}{\lambda}}$, where the factor of 4 arises since there are 4 input streams. Cars exchange state information and time-slot information every $T = 0.2$s.

We first perform a post-hoc optimization algorithm over all m cars, where we assign slots to all cars at time zero, irrespective of how far these cars are from the intersection. The drawback with this algorithm is that it requires information about future car arrivals and hence does not allow for the scheduler to have a finite range of communication. We would like to integrate this forward simulation engine to create an online optimization algorithm with a causal information structure. Suppose that each car makes first contact with the scheduler when it is within 200 m of the intersection. Each car requesting a slot provides information about itself and its one-hop neighbors. The scheduler runs the descent algorithm and assigns slots to the cars requesting slots, and reserves slots for the cars which are not yet within communication range, but whose information has reached the scheduler. When these cars indeed arrive, the descent algorithm is run again starting with the reserved slots as the initial slot assignment.

The above simulation framework has been extended to simulate worst case stop signs and worst case traffic lights with guaranteed safety. While this may be somewhat different from actual stop signs and traffic lights, it enables a fair comparison. For stop signs, the scheduler assigns time slots in a first-come first serve (FCFS) fashion to cars only when they are very close to the intersection. For traffic lights,



	Permitted Routes	Duration
Phase 1	NS, NE, NW, SE	4 sec
Phase 2	NS, SN, NW, SE	12 sec
Phase 3	SN, SE, SW	4 sec
Phase 4	EW, EN, ES	4 sec
Phase 5	EW, WE, EN, WS	12 sec
Phase 6	WE, WS, WN	4 sec

Figure 4: Phase plan for Traffic Light

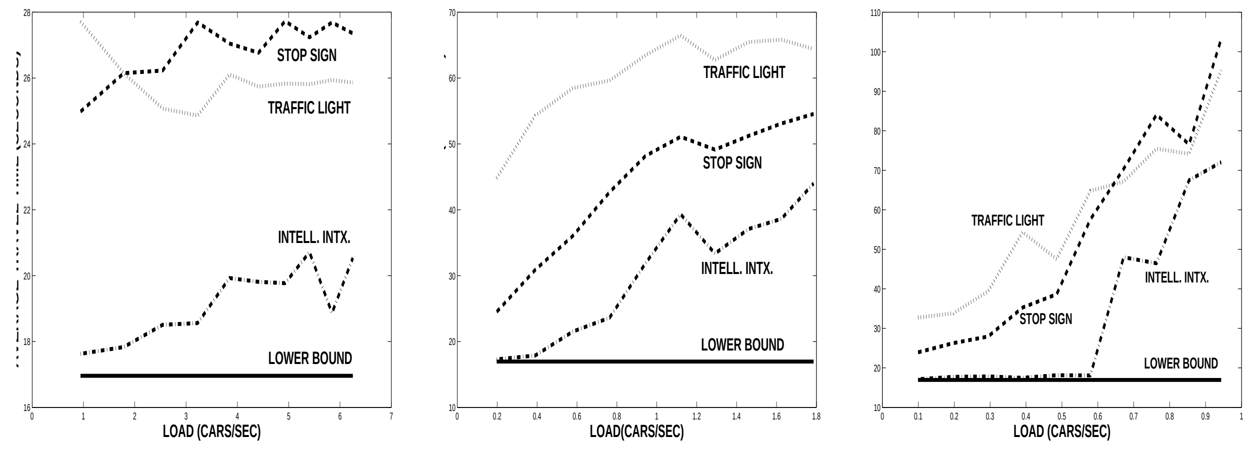
we have implemented a *pretimed signal operation*, using a phase plan with six phases including leading and lagging greens (see Chap. 18.1 in [22]) as described in Figure 4. In practice, the cycle length and duration of green periods are set by measuring critical lane volumes, headway saturation and lost time (see Chap. 18.2 in [22]). Since we do not have access to these parameters in our simulations, we determined the appropriate cycle length and green periods by an examination of what values work best for a given load. The phase durations are chosen so as to balance the load generated by probabilistic route assignments. Each input stream sees a green signal for 4/10 of the entire cycle. In our simulations, we implemented a phase plan with a cycle length of 40 seconds, where each input lane gets a 16 second green period, which includes a 4 second green *protected left turn.* For a fixed set of routes, we compare the performance of our scheduler against these traffic regulation mechanisms, for a variety of loads. The comparisons are presented in Figure 5 below.

We note here that many modern traffic lights employ *actuated signal control.* Under this operation, detectors indicate the presence of cars on one or several lanes, thus enabling the traffic signal to adaptively change the phase durations so as to improve efficiency or fairness or both. Such a traffic light can be semi-actuated or fully actuated, depending on whether there are detectors on the minor approaches to the intersection alone, or on every approach of every lane (Chap. 20 of [22]). In our context, we are unable to implement semi-actuated or fully actuated traffic signals due to the following reason. In our problem, we are interested not only in traffic intersections, but also in provable safety in the road segments indefinitely before and indefinitely after the traffic intersection, as well as within the traffic intersection itself. Hence, in our simulation framework, traffic lights and stop signs are implemented *over and above* the architecture and algorithm which guarantees provable safety. It is not clear how, if at all, to merge automated cars with provable safety, with conventional actuated traffic lights. Technically, for instance, it may not be feasible to dynamically change the phase plan by extending or shortening the green phase for a particular stream since these reallocations might affect the slots of cars with conflicting routes. In particular, the reallocations need to satisfy the conditions in Definitions 2 and 4, which makes actuated control infeasible in our current simulation framework.[8]

We can also model persistent traffic by running the simulation for a large number of cars and then ignoring edge effects. The load is varied from low (0.2 cars/sec) to moderate (1 car/sec) and high (2 cars/sec). In Figure 5, we see that the intelligent intersection consistently 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. This suggests that a dynamic slot assignment mechanism

[8]We are grateful to the reviewer for comments on this issue.

outperforms a FCFS service discipline as in stop signs, or a periodic service discipline as in traffic lights. For this same reason, a fully actuated traffic light may have performance that is closer to that of the intelligent intersection, especially at high loads. However, we should note that all these conclusions deserve a much more thorough simulation study than we have conducted.



(a) Clearing 10 cars (b) Clearing 30 cars (c) Persistent traffic

Figure 5: Average travel time comparison

7 Concluding Remarks

Technological developments in in-vehicle sensing and computation, along with wireless networking of automobiles with other automobiles and the roadside infrastructure, and global positioning, are leading to automated vehicle systems. This paper has examined an important safety application, intelligent intersections, that can also potentially provide improved efficiency. Intelligent intersections are representative of the class of complex, distributed hybrid systems, for which it is necessary to provide provably safe designs, both architecturally and algorithmically, that are furthermore robust and tractable. We have proposed a provably safe design based on distributed updating of infinite horizon contingency plans by distributed agents, with centralized mediation. We have demonstrated by a simulation study the potential performance benefit of our approach over worst case stop signs and worst case traffic lights. It is important to consider alternative designs which can provide guarantees against adverse behavior of cars after they enter the intersection, and approaches for system reinitialization by revoking time slot assignments. Another area for future work is to explore provably safe strategies that take into account passenger comfort. It would also be of interest to develop mathematical performance evaluation methodologies that can model situations of practical interest and yield answers to with the accuracies that are desired. This appears to be a difficult challenge.

It is hoped that approaches such as this may be useful in the design of other tractable complex, distributed, hybrid systems.

References

[1] M. K. Powell and J. S. Adelstein. Report and order: Amend rules regarding dedicated short range communicacalitions services and rules for mobile service for dedicated short range communications for intelligent transportation services. Technical report, United States Federal Communications Commission, December 2003.

[2] California Center for Innovative Transportation. Intelligent transportation systems. http://www.calccit.org/itsdecision.

[3] National Center for Statistics and Analysis. 2006 Traffic Safety Annual Assessment - A Preview. DOT HS 810 791, July 2007.

[4] B. E. Ydstie and L. K. Liu. Single-and multi-variable control with extended prediction horizons. In *American Control Conference*, volume 21, pages 1303–1308, 1982.

[5] 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*, pages 968–975, Toronto, Canada, 2006.

[6] 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*, 15:335–340, 1994.

[7] Y. Fujita, K. Akuzawa, and M. Sato. Radar brake system. In *Proceedings of the 1995 Annual Meeting of ITS America*, volume 1, pages 95–101, 1995.

[8] 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*, Toronto, Canada, 2005.

[9] 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 IEEE Conference on Decision and Control*, 2001.

[10] D. Swaroop and J. K. Hedrick. Constant spacing strategies for platooning in automated highway systems. *Journal of Dynamic Systems, Measurement, and Control*, 121(3):462–470, 1999.

[11] P. Varaiya. Smart cars on smart roads: Problems of control. *Automatic Control, IEEE Transactions on*, 38:195–207, 1993.

[12] J. Huang and H. S. Tan. Design and implementation of a cooperative collision warning system. In *Proceedings of the IEEE Intelligent Transportation Systems Conference*, pages 1017–1022, Toronto, Canada, 2006.

[13] J.K. Kuchar and L. C. Yang. A review of conflict detection and resolution modeling methods. *IEEE Transactions on Intelligent Transportation Systems*, 1:179–189, 2000.

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

[15] C. Tomlin, G. Pappas, and S. Sastry. Conflict resolution of air traffic management: A study in multiagent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, 1998.

[16] D. M. Stipanovic, P. F. Hokayem, M. W. Spong, and D. D. Siljak. Avoidance control for multiagent systems. *ASME Journal of Dynamic systems, measurement and control*, 129(5):699–707, 2007.

[17] G. Leitmann and J. Skowronski. Avoidance control. *Journal of Optimization Theory and Applications*, 23:581–591, 1977.

[18] T. Başar and P. R. Kumar. On worst case design strategies. *Comput. Math. Applic.*, 13(1–3):239–245, 1987.

[19] P. R. Kumar and P. Varaiya. *Stochastic systems: Estimation, identification and control.* Prentice Hall Inc., Englewood Cliffs, N.J., 1986.

[20] H. Kowshik. Provable systemwide safety in intelligent intersections, m.s. thesis. *University of Illinois, Urbana-Champaign*, 2008.

[21] E. Frazzoli, M.A. Dahleh, and E. Feron. A maneuver-based hybrid control architecture for autonomous vehicle motion planning. *Software Enabled Control: Information Technology for Dynamical Systems, G. Balas and T. Samad editors*, 2003.

[22] R. P. Roess, E. S. Prassas, and W. R. McShane. *Traffic Engineering.* Prentice Hall PTR, USA, 2004.