Crossroads+: A Time-Aware Approach for Intersection Management of Connected Autonomous Vehicles

MOHAMMAD KHAYATIAN, Arizona State Univeristy, USA
YINGYAN LOU, Arizona State Univeristy, USA
MOHAMMADREZA MEHRABIAN, Arizona State Univeristy, USA
AVIRAL SHIRVASTAVA, Arizona State Univeristy, USA

As vehicles become autonomous and connected, intelligent management techniques can be utilized to operate an intersection without a traffic light. When a Connected Autonomous Vehicle (CAV) approaches an intersection, it shares its status and intended direction with the Intersection Manager (IM), and the IM checks the status of other CAVs and assigns a target velocity/reference trajectory for it to maintain. In practice, however, there is an unknown delay between the time a CAV sends a request to the IM and the moment it receives back the response, namely the Round-Trip Delay (RTD). As a result, the CAV will start tracking the target velocity/reference trajectory later than when the IM expects, which may lead to accidents. In this paper, we present a time-aware approach, Crossroads+, that makes CAVs' behaviors deterministic despite the existence of the unknown RTD. In Crossroads+, we use timestamping and synchronization to ensure the both the IM and the CAVs have the same notion of time. The IM will also set a fixed start time to track the target velocity/reference trajectory for each CAV. The effectiveness of the proposed Crossroads+ technique is illustrated by experiments on a 1/10 scale model of an intersection with CAVs. We also built a simulator to demonstrate the scalability of Crossroads+ for multi-lane intersections. Results from our experiments indicate that our approach can reduce the position uncertainty by 15% in comparison with conventional techniques and achieve up to 36% better throughputs.

Additional Key Words and Phrases: Connected Autonomous Vehicles, Intersection Management, Round-trip Delay, Cyber-Physical Systems

1 INTRODUCTION

With the advent of Connected Autonomous Vehicles (CAVs), existing infrastructures for managing an intersection such as stop signs and traffic lights may be replaced by more effective frameworks to improve mobility. An Intersection Manager (IM) can increase the throughput of the intersection by exchanging information with and directing incoming CAVs [25, 30, 45, 52].

An intuitive and efficient way to implement such an IM is to assign a constant velocity to each CAV. Upon approaching the intersection, a CAV can send a request to the IM and reports its status and intended outgoing direction. The IM then checks the assigned trajectories of other present CAVs and assigns a "target velocity" to the requesting CAV to maintain. We refer to this technique as Velocity Transaction Intersection Management (VT-IM). To avoid CAV collisions and guarantee safety, an IM should consider a buffer around each CAV to account for uncertainties in its position.

Authors' addresses: Mohammad Khayatian, Arizona State Univeristy, 660 S Mill Ave, Tempe, USA, mkhayati@asu.edu; Yingyan Lou, Arizona State Univeristy, 660 S. College Avenue, Tempe, USA, Yingyan.Lou@asu.edu; Mohammadreza Mehrabian, Arizona State Univeristy, 660 S Mill Ave, Tempe, USA, mmehrabi@asu.edu; Aviral Shirvastava, Arizona State Univeristy, 660 S Mill Ave, Tempe, USA, aviral.shrivastava@asu.edu.

XXXX-XXXX/2019/9-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

The size of this buffer is determined based on errors that may come from CAV sensors and models of CAV dynamics used by the IM [3].

However, there is another source of error, namely the communication delay and processing time of the IM, which has mostly been ignored in existing studies [16, 25, 30, 32, 45, 52]. When a CAV communicates with the IM over a wireless network, the CAV's information (position, velocity, etc.) is delivered after some delay (depending on the distance, network bandwidth, etc.). Similarly, when the CAV receives the target velocity from the IM, it could be further downstream than the expected position and therefore, the assigned velocity may no longer be safe. When CAVs and the IM have different notions of time, CAVs will behave differently that IM expects and can cause accidents at the intersection. We refer to this problem as *timing problem*. We refer to the total latency, from the moment a CAV sends its information to the IM, to the moment it receives back the response, as the Round Trip Delay (RTD). The RTD is the summation of the CAV-to-IM transmission delay, IM processing time, and the IM-to-CAV transmission delay. An IM is not able to predict the RTD accurately because of two reasons: 1) the processing time of the IM is dynamic and depends on the number of CAVs that are being managed by the IM, and 2) the network delay may not be symmetric in the forward and the backward paths and cannot be compensated.

The most intuitive way to overcome this problem is to consider a larger safety buffer around each CAV, which is expanded longitudinally proportional to the Worst-Case RTD (WCRTD) and the maximum velocity of the CAV. Results from our experiments using 1/10 scale-model CAVs show that this buffer can be as ten times large as the original size of a CAV (see Section 8). Expanding the safety buffer is therefore not practical since such a large buffer will significantly reduce the throughput of the intersection.

In this paper, we present a time-sensitive VT-IM approach called Crossroads+ to address the timing problem associated with the unknown RTD. In Crossroads+, all CAVs synchronize their clock with the IM and share their status with a timestamp that corresponds to the moment when the requesting CAV has measured its status (position, velocity, etc.). When calculating the target velocity for each requesting CAV, the IM incorporates the dynamical model and parameters of the vehicle controller used for tracking the target velocity. The IM also accounts for acceleration/deceleration limits of individual CAVs. Additionally, the IM sets a fixed time for actuation (when a CAV starts to change its velocity to the target value). By setting the actuation time to be later than the request time plus WCRTD, the CAV behaviors will be deterministic and the IM schedule safe. Crossroads+ avoids considering an extra safety buffer due to unknown RTD and therefore, can achieve higher throughputs. The effectiveness of the proposed method is illustrated by conducting experiments on our testbed with 1/10 scale model CAVs. We also built a simulator to verify the scalability of our technique for multi-lane intersections. On average, Crossroads+ can reduce the uncertainty in position by 15% and achieve 36% better throughputs compared to conventional VT-IM methods with an extra safety buffer.

The rest of the paper is organized as follows. Section 3 explains the necessity of safety buffer for CAVs and the problem associated with RTD. Section 2 provides an overview of previous studies and their limitations. We present the Crossroads+ approach in Section 4 and 5. The proof of safety for CAVs in Crossroads+ is shown in Section 6. Section 7 introduces our experimental testbed and simulator; and Section 8 presents the results.

2 RELATED WORKS

In the past few years, different methodologies have been proposed to manage intersections of CAVs. However, the timing problem due to RTD was ignored in almost all previous works. A safe methodology for intersection management should be complete and clearly specify when CAVs should start communicating, what data should be communicated, how CAVs are being scheduled,

and how the desired behavior is transmitted to a CAV. Without specifying such interfaces, the design is not realistic. Below we categorize existing works based on IM methodology, IM scheduling policy, and evaluation testbed.

2.1 Methodology

AIM [5, 11, 16] is one of the first proposed methods to manage an intersection of CAVs. AIM is a query-based intersection management (QB-IM) approach where an incoming CAV queries the safe pass from the IM by sending its estimated time of arrival. The IM either reserves a time-space block in the intersection for the requesting CAV and sends back a yes, or rejects the request if the pertinent time-space block has already been reserved. Similar QB-IM methodologies [6, 20–22] have been proposed where a reservation map is used to keep track of existing CAVs. Due to the trial-and-error nature of QB-IM approaches, they have high network overhead and achieve lower throughputs.

In VT-IM approaches [15, 29, 45], the IM assigns a target velocity to each CAV and can achieve higher throughputs compared to QB-IM. In 2016, Yang *et al.* [49] proposed a methodology where IM solves an optimization problem to determine the optimal trajectory for each CAV. The optimization problem is solved iteratively every time a new CAV enters the intersection zone, come to a stop or departs the intersection. The computation and network overhead of such approaches are high because to send the reference trajectory to a CAV, an array of data and timestamps need to be transmitted. Also, when IM reschedule the CAVs for achieving better throughputs, the reference trajectory needs to be re-transmitted. There are similar works where an optimization problem is solved to determine the optimal trajectories for a set of CAVs. However, such approaches assume that the position of all CAVs are known and do not present a methodology for intersection management [18, 25, 26, 28, 41].

2.2 Scheduling Policy

Focusing on scheduling of CAVs, researchers have proposed various policies such as First-Come First-Serve (FCFS) [3, 12, 14, 37, 38], optimization-based [2, 23, 25, 30, 34–36, 44, 52, 53], and heuristic [4, 9, 27, 45] approaches. In FCFS approaches, CAVs or platoons[8, 22] are scheduled to enter the intersection in the same order as they approach the intersection; while in optimization-based ones, IM modifies the order to achieve higher throughputs.

Although better efficiency can be achieved using optimization-based approaches, they are computationally expensive. As a result, an IM has to spend more time to assign a reference trajectory to each CAV; and thus a higher RTD. It should be noted that optimization-based approaches may also face starvation. For instance, assume a CAV approaching an intersection from the north and a continuous flow of heavy traffic approaching the intersection from the west at the same time. An optimal policy to maximize the throughput will always schedule the CAVs from the west before the CAV from the north and will result in very long wait times for the CAV from the north. In the extreme case, starvation happens and the single CAV never gets a reservation.

2.3 Evaluation Testbed

Most of existing works are evaluated with simulation. VISSIM [26, 28, 50], MATLAB [8, 15, 49, 51], Sumo [20–22], AutoSIM [4, 6] are the most popular testbeds for evaluation. Such approaches, however, evaluated their methods using simulation alone, where the effect of network delay and IM processing time was not taken into account.

There are also a number of real demonstrations for intersection management. Fok *et al.* [16] developed a real platform for evaluating the AIM in 2012. They created a single-lane intersection with four 1/10 scale model CAVs. The CAVs are equipped with a camera vision sensor, an infrared

range sensor, and a Wi-Fi transceiver. The maximum speed used for their experiment was limited to 0.5 m/s (1.1 mph), which is the primary reason that ignoring the network delay did not cause any issues in their experiments. Milanes *et al.* proposed a CAV-to-CAV (V2V) method where CAVs detect the presence of other CAVs approaching an intersection [32]. A fuzzy controller was introduced to manage the intersection. Their method was verified through experiments on two fully automated Citroen C3 Pluriel, called Clavileño, at the Instituto de Automática Industria (IAI) facilities. Due to the simplicity of the testbed (just two CAVs), they possibly did not face any issue related to network delay and computation time. A heuristic intersection management technique was proposed by Ahmane *et al.* [1], where CAVs communicate with each other to decide on who has the "right of way". Their approach considers three zones, a storage zone, a conflict zone, and an exit zone. They used a regular CAV equipped with a device that guides the driver. The authors pointed out the existence of the network delay and packet loss in a real implementation, but did not discuss it. They didn't report any accidents because human drivers were used to driving the car and if there was a critical case, it has been avoided by the driver. Dedinsky *et al.*[10] propose a camera-based surveillance system to detect vehicles that do not follow IM's command and broadcasts an emergency packet to all vehicles.

Almost all previous works have ignored the communication delay between IM and CAVs or assumed it to be very small and has no effect on the safety of CAVs.

Robust Intersection Management (RIM) technique[24] takes another approach. In RIM, the IM assigns a desired Time of Arrival (TOA) and Velocity of Arrival (VOA) to an incoming CAV; and it is the CAV's responsibility to create and track a reference trajectory such that it enters the intersection at the assigned TOA while driving at the assigned VOA. Since IM is responsible for the feasibility of the assigned TOA and VOA both before and inside the intersection, it needs to check for conflicts between CAVs on the same lane before entering the intersection, which adds more computational overhead to the system. Also, the IM needs an acknowledgment from each CAV to efficiently manage other CAVs.

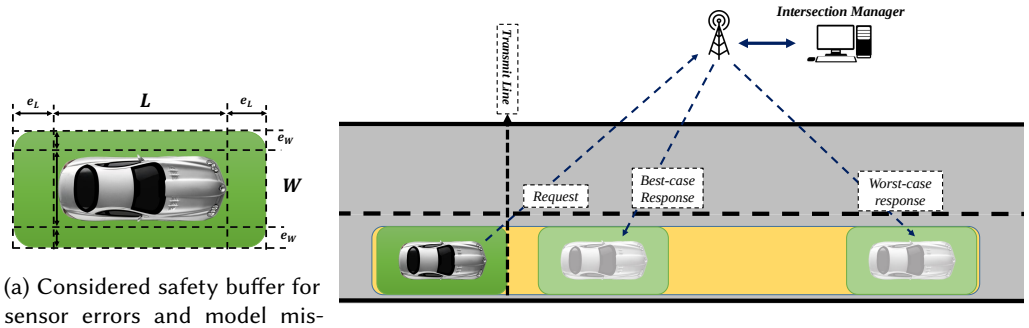
3 BACKGROUND

3.1 Safety Buffer

All localization devices used for autonomous driving like GPS, encoder, and odometer are subject to inherent errors. It's shown that Kalman Filter [39, 40] and Particle Filter[19, 47] based approaches can be used to improve the accuracy of localization for an autonomous agent. Nowadays, SLAM (Simultaneous Localization and Mapping) algorithms [7, 13] are widely used to estimate CAV location with the help of landmarks in the environment. However, it has been shown that current SLAM approaches are still not perfect [17]. Noise, drift, unknown disturbance, nonlinear behavior, and model mismatch are some common causes that can result in discrepancies between the actual and estimated positions of a CAV. Different error models exist to account for uncertainties in position. For instance, GPS error can be modeled as a circle around the CAV; while shaft encoder's error drifts over time and is typically modeled as a longitudinal buffer in front of/behind a CAV. Due to these errors, the IM should consider a buffer around each CAV to avoid potential collisions, which we refer to as *Safety Buffer*. The size of the safety buffer is related to the accuracy and precision of positioning sensors (odometer, IMU, GPS sensor, etc.) as well as the estimation algorithms. In this paper, we consider the error model to be similar to Figure 1a.

3.2 Round-trip Delay

In a real deployment of an intersection management algorithm, CAVs and IM communicate over a wireless network and therefore, are prone to unknown communication delays. This delay is directly



(a) Considered safety buffer for sensor errors and model mismatch

(b) Added safety buffer to account for RTD.

Fig. 1. Safety Buffer and its extension to account for the nondeterministic RTD

related to the amount of data that needs to be transmitted. For instance, sending two arrays of data (e.g., reference position and timestamps) will incur higher delays compared to transmitting a single variable (e.g., constant velocity).

In conventional VT-IM approaches, the target velocity is safe only if it is received by a CAV at where the original request was sent and is executed right away. In reality, however, the communication delay and IM processing time are not zero and the CAV receives the velocity later than when the IM expects. As a result, the CAV's trajectory will be different from what the IM has calculated and this may cause an accident inside the intersection. Based on the initial velocity

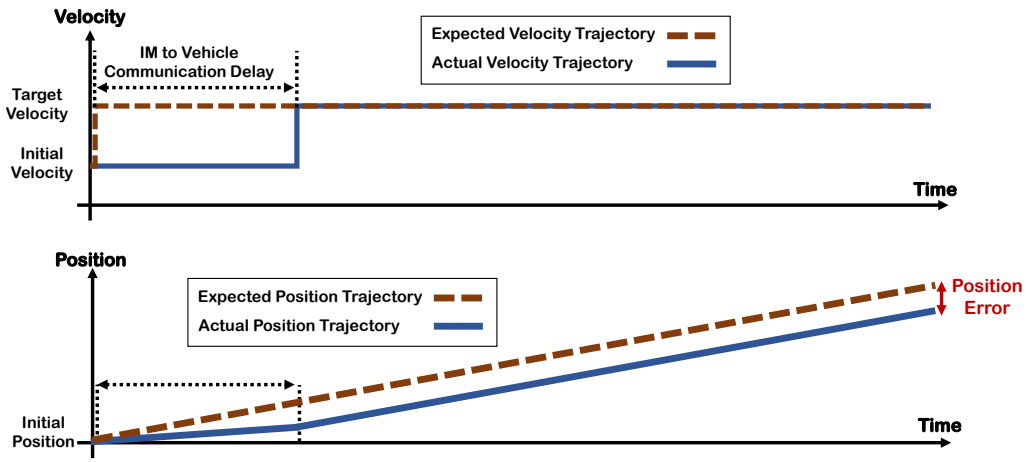


Fig. 2. Ignoring the network delay will result in position error and can cause accidents.

and the assigned target velocity, a CAV may be ahead of or behind the expected position. If the assigned velocity is greater than the initial velocity and the RTD is not zero, then the actuation (acceleration) will be delayed and the CAV will enter the intersection later than expected and vice versa. Figure 2 shows an example of a CAV lagging behind the expected trajectory due to RTD (assuming instant velocity change).

One way to model the effect of communication delay and the processing time of the IM is to consider a larger safety buffer around each CAV. Figure 1b illustrates how the safety buffer needs to

be extended longitudinally. The size of the buffer (B_{RTD}) depends on the WCRTD and the maximum possible velocity of CAVs

$$B_{RTD} = \Delta t_{WCRTD} \times v_{max} \quad (1)$$

Δt_{WCRTD} is defined as the duration of the worst-case delay from the time a CAV sends its information to the IM to the time it receives the response. The shorter the WCRTD, the smaller the added buffer, and consequently, the better the throughput. This is because a much larger virtual size of the CAV is considered with an extended safety buffer.

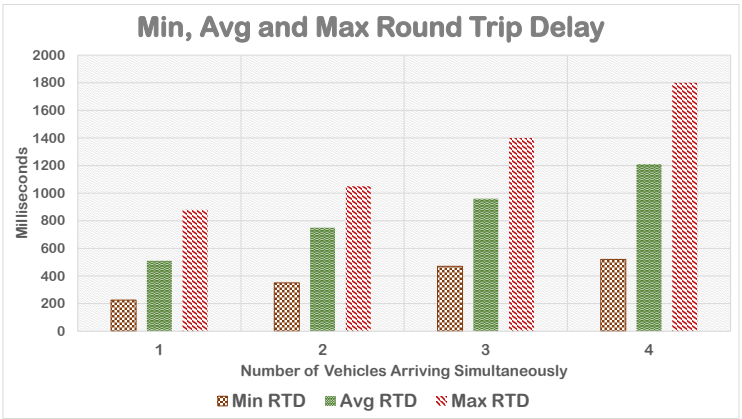


Fig. 3. Round Trip Delay (RTD) for the best-case, average-case and worst-case network delay for 1-4 CAV(s) sending a request to the IM simultaneously.

We measured the RTD when there are multiple CAVs sending requests to an IM simultaneously using our 1/10 scale model testbed (see Section 7). Figure 3 depicts the average, minimum and the maximum values of measured RTD for 4 scenarios, where the number of CAVs sending a request to the IM is equal to 1, 2, 3 and 4 respectively. We repeated each experiment 10 times. Note that for a 4-way intersection with one lane per road, the maximum number of CAVs that can send a request at the same time is 4. We can observe that the WCRTD increases as the number of CAVs increments.

4 CROSSROADS+

In this section, we present our Crossroads+ methodology. Algorithms 1 and 2 show the Crossroads+ interface for CAVs and the IM. Upon reaching the *sync line* (depicted in Figure 4), CAVs communicate with the IM to synchronize their local clock. Synchronization is needed to have the same notion of time among all nodes [31, 42, 43]. When a CAV reaches the designated *transmit line*, it captures the time and sends its ID, position, velocity, maximum acceleration and deceleration rates, destination lane and the captured timestamp to the IM. After sending the request, CAVs will continue traveling with their initial velocity until either receiving a response from the IM or reaching the slow-down line. When the IM receives a request, it computes a target velocity and an actuation time based on the received information and status of other CAVs. When the CAV receives the target velocity and actuation time, it continues traveling at its current velocity until the actuation time, at which point it will take action to reach the assigned target velocity. If a CAV cannot synchronize its clock with the IM before reaching the *1st slow-down line*, it will apply the brake to stop behind the *transmit line*. If a CAV does not receive a response from the IM after the set timeout, it will request again. If no response is received from the IM before reaching the *2nd slow-down line*, the CAV will apply

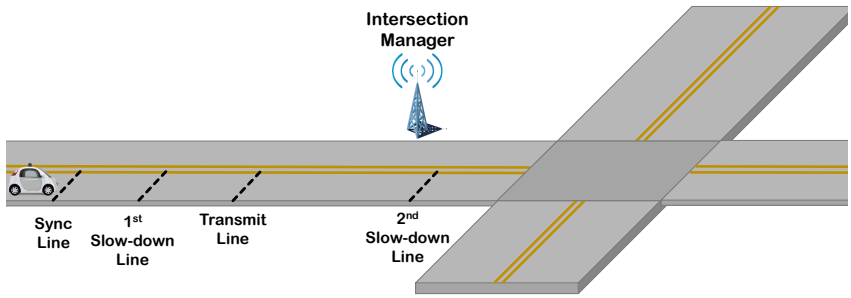


Fig. 4. An overview of Crossroads+ interface. IM uses a FCFS policy for scheduling of CAVs.

the brake and stop behind the intersection line. Slow-down lines are imaginary lines and are set for fail-safe operation of CAVs. Recovery from an event (such as a problem in the communication system or when an emergency car approaches) is out of the scope of this paper, but is an important issue to be studied further.

Algorithm 1: Crossroads+ Approach - CAV

```
if Sync line is crossed then
    Synchronize clock with IM;
    while Waiting for Sync do
        if reached the 1st slow-down line then
            Apply brake;
        end
        if sync is successful then
            Exit the loop;
        end
    end
end
if Transmit line is crossed then
    Send a request to IM;
    while waiting for the response do
        if reached the 2nd slow-down line then
            Apply brake;
        end
        if no response within the timeout then
            Prepare to stop when the 2nd slow-down line is reached;
        else
            Exit the while loop;
        end
    end
    Receive the response;
    Wait until actuation time;
    Accelerate or decelerate to achieve the target velocity;
end
```

Algorithm 2: Crossroads+ Approach - IM

```
if a new msg is received then
    if msg is a request then
        Set the actuation time (t_act);
        Find potential conflicts (x_c, t_c) w.r.t. existing CAVs ;
        Calculate the target velocity (v_T^new) w.r.t. x_c, t_c;
        Send back t_Act and v_T^new;
        Add the requesting CAV's information to the list of active CAVs;
    end
    if msg is a leave notification then
        remove the CAV's information from the list of active CAVs;
    end
end
```

4.1 Crossroads+ Scheduling Policy

Crossroads+ adopts an FCFS-based policy to ensure fairness among all CAVs. If two or more CAVs send their requests at the same time, the IM will use the CAVs' IDs to break the tie. When the IM receives a request from a CAV, it adds the CAV's information to the list of active CAVs. Accordingly, when a CAV leaves the intersection, it will be removed from the list of active CAVs by the IM.

When a request is received, the IM sets the actuation time for the CAV, t_{Act}, to be:

$$t_{Act} = t_0 + \Delta t_{WCRTD} \tag{2}$$

where t_0 is the timestamp corresponding to the moment at which the requesting CAV has measured its status (position, velocity, etc.) and Δt_{WCRTD} is the considered WCRTD. We will later discuss how to determine a reasonable value of Δt_{WCRTD} for the intersection manager. The actuation time corresponds to a particular position along the vehicle's travel path, x_{Act}, since the CAV will drive at its initial velocity until t_{Act}:

$$x_{Act} = x_0 + v_0 \times \Delta t_{WCRTD} \tag{3}$$

where x_0 is the position of CAV at request time (t_0), and v_0 is the initial velocity of the CAV at t_0. The IM then computes all potential conflict points and corresponding conflict times along the requesting CAV's path with respect to each existing active CAV i. x_c and t_c are the sets of conflicts position (x_c^i) and times (t_c^i) calculated based on the set target velocities of existing active CAVs. For example, Figure 5 illustrates the potential conflict positions and times calculated for the travel path of the requesting CAV, CAV #5. The travel path of CAV #5 is depicted by the dashed line. Other CAVs (#1, #2, #3 and #4) are on the list of active CAVs and have already received a target velocity and actuation time. Their paths are shown by solid lines. The intersections of paths of all CAVs with CAV #5 are denoted by solid dots. For example, the blue dot indicates the intersection between the paths of CAV #1 and CAV #5, i.e. conflict point x_c^1. t_c^1 is the time when CAV #1 reaches this point x_c^1. For CAVs on the same lane as the requesting CAV, the edge of the intersection is considered as the conflict point. With all potential conflict points determined, the IM projects each conflict time (t_c^i) to a safe reach time (t_s^i):

$$t_s^i = t_c^i + \Delta t_{safety} \tag{4}$$

where Δt_{safety} is the delay buffer ensuring that the requesting CAV reaches the conflict position Δt_{safety} time units after the existing active CAV i. One way to determine the Δt_{safety} in order to

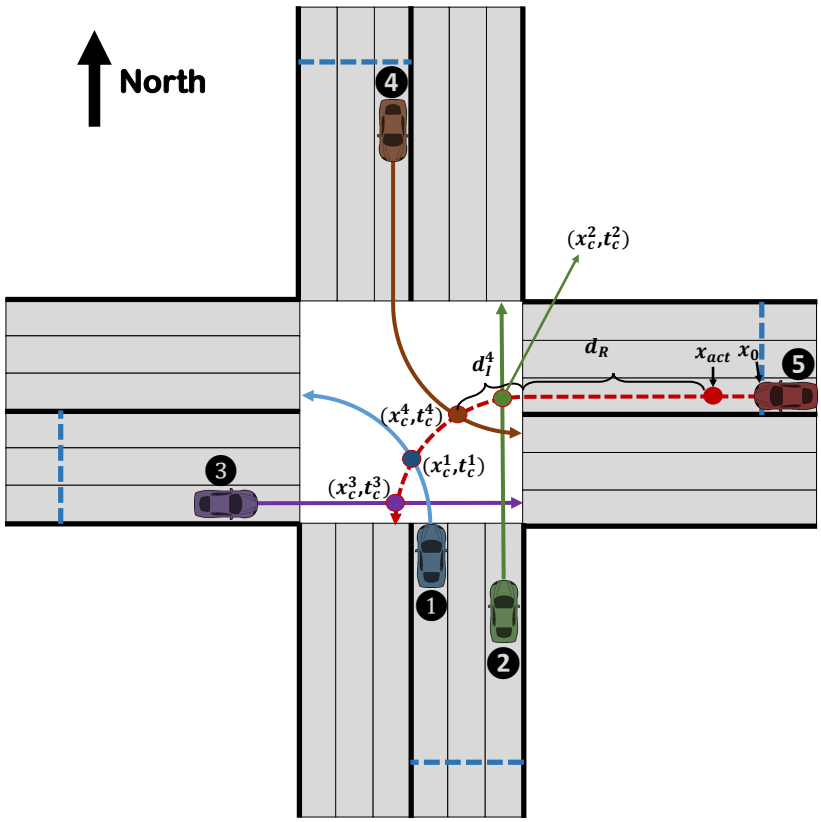


Fig. 5. A view of a four-way intersection with three lanes per road. Four conflict points are determined for the CAV number 5 regarding the existing ones.

achieve guaranteed safety is

$$\Delta t_{safety} = \frac{l_{max} + l_B}{v_{min}} \tag{5}$$

where l_B is the longitudinal size of the safety buffer due to sensor error, l_{max} is the length of the longest CAV among all CAVs and v_{min} is the slowest velocity that IM assigns to a CAV i.e., IM will not assign a velocity less than v_{min} to a CAV. The proof of safety will be presented in Section 6. The target velocity v_T^i corresponding to a pair of safe time and conflict position (t_s^i, x_c^i) is calculated as:

$$v_T^i = \frac{d^i}{t_s^i - t_{Act}} \tag{6}$$

where

$$d^i = x_c^i - x_{Act}$$

is the total distance the requesting CAV travels from its actuation position until reaching the conflict position x_c^i. We can rewrite d^i as $d^i = d_I^i + d_R$. d_I^i is the distance the requesting CAV travels *inside* the intersection to reach the conflict position x_c^i and d_R is the distance between the actuation position and edge of the intersection. The computation of d_I^i and d_R is straight-forward when the dimension of the intersection is known. For instance, it is trivial to show that d_I^4, the distance CAV #5 travels

inside the intersection to reach x_c^4 in the example of Figure 5, is equal to $acos(3/3.5) \times 3.5LW$, where LW is the lane width. Since an instant velocity change is impractical, the target velocity v_T^i needs to be modified to account for the requesting CAV's acceleration/deceleration. In the next section, we will explain in detail how the compensated target velocity $v_T^{new^i}$ is calculated based on v_T^i, taking into consideration the vehicle dynamics of the requesting CAV. After calculating all target velocities ($v_T^{new^i}$) the IM selects the slowest velocity and assigns it to the requesting CAV.

5 SAFE TARGET VELOCITY CALCULATION

This section discusses how to calculate $v_T^{new^i}$ based on v_T^i, taking into consideration the vehicle dynamics of the requesting CAV. The superscription i is dropped in this section to simplify the notations. Without loss of generality, we only discuss the case where v_T is higher than v_0. The case of deceleration can be analyzed similarly.

In order to compute the compensated target velocities (v_T^{new}) to account for CAV acceleration/deceleration time, we need to know what the actual velocity and acceleration profiles of a CAV under v_T are. They can be calculated using a model for the CAV dynamics, with explicit considerations of acceleration limits and the underlying CAV controller.

In subsection 5.1, we first compute the velocity profile with realistic vehicle dynamics but unbounded acceleration, which will serve as a baseline for computing the compensated target velocity. Subsection 5.2 explains how v_T^{new} is calculated in the case that the resulting CAV acceleration/deceleration is always within the acceleration/deceleration limit. Subsection 5.3 discusses the case where the resulting CAV acceleration/deceleration from the baseline unbounded profile exceeds the limit.

5.1 Solving for Velocity Profile with Realistic Vehicle Dynamics and Unbounded Acceleration

The following differential equations are used to model CAV motion in the 2D space:

$$\begin{cases} \dot{x} = vcos(\phi) \\ \dot{y} = vsin(\phi) \\ \dot{\phi} = \frac{v}{L} tan(\psi) \\ \dot{v} = a \end{cases} \tag{7}$$

where x, y are the longitude and latitude of a CAV in Cartesian coordinates respectively, ϕ is the heading angle from the x-axis, v and a are linear velocity and acceleration of the CAV respectively, L is CAV's wheelbase distance and ψ is steering angle of front tires with respect to the heading of the CAV. In order to account for maximum acceleration and brake capability of each CAV, we consider the following saturation function to bound the acceleration. Therefore, Equation (7) is replaced by:

$$\dot{v} = \begin{cases} a_{max}, & \text{if } a > a_{max} \\ a_{min}, & \text{if } a < a_{min} \\ a, & \text{otherwise} \end{cases} \tag{8}$$

Without loss of generality, consider an eastbound CAV (driving along the x-axis before entering the intersection) that uses a PID controller to achieve and maintain the assigned target velocity.

The vehicle dynamics presented in Equation (7) for this vehicle can be written as:

$$\begin{cases} \dot{x} = v \\ \dot{y} = 0 \\ \dot{\phi} = 0 \\ \dot{v} = K_P e + K_I \int e + K_D \dot{e} \end{cases} \tag{9}$$

where e is the velocity error ($e = v_T - v$) and K_P,K_I and K_D are Proportional, Integral and Derivative gains of the PID controller. By substituting $\dot{e} = -\dot{v}$ and taking the derivative of Equation (9), we have:

$$\ddot{v} = -K_P \dot{v} + K_I(v_T - v) - K_D \ddot{v}$$

simplifying, we get:

$$\ddot{v} + \frac{K_P}{1 + K_D}\dot{v} + \frac{K_I}{1 + K_D}v = \frac{K_I V_T}{1 + K_D} \tag{10}$$

The solution to Equation (10) is the actual velocity profile of a CAV under the target velocity v_T.

Once the values of the PID gains are determined, we can find the homogeneous solution to Equation (10) by obtaining the characteristic equation of the system. To do so, we should replace $\ddot{v}$, $\dot{v}$ and v in Equation (10) with v^2, v and 1 respectively and set the left hand side of the equation equal to zero:

$$(1 + K_D)v^2 + K_P v + K_I = 0 \tag{11}$$

This equation can have real or complex roots, but we only consider the case with real roots. This is because complex roots correspond to overshoot in the controller response and are not suitable for velocity tracking. The homogeneous solution to Equation (10), when roots of the characteristic equation of the system are real, can be written as:

$$v_H(t) = c_1 e^{-At} + c_2 e^{-Bt} \tag{12}$$

where A and B are roots of the characteristic equation (Eq. (11)), and c_1 and c_2 are constants to be determined (computed later). The particular solution to Equation (10) can be calculated by setting all derivatives to zero.

$$v_P(t) = v_T \tag{13}$$

Using initial conditions $\dot{v}(0) = 0$ and $v(0) = v_0$ (without loss of generality, let $t_{Act} = 0$), the complete solution can be written as:

$$v(t) = v_T + c_1 e^{At} + c_2 e^{Bt} \tag{14}$$

where

$$c_1 = \frac{-B}{A - B}(v_0 - v_T)$$

and

$$c_2 = \frac{A}{A - B}(v_0 - v_T)$$

5.2 Case 1, no saturated acceleration

With the baseline velocity profile under a target velocity v_T calculated as in Equation (14), let us first consider the case where the corresponding acceleration to achieve this velocity profile is always within the acceleration limit.

In this case, the baseline velocity profile (Equation (14)) is the actual velocity profile; but the expected velocity profile the IM adopted when calculating v_T assumes instantaneous velocity change. The position discrepancy at the expected time of arrival t_s at a potential conflict position

x_c can be calculated as the difference between the area under the expected and the actual velocity profiles:

$$e = \int_0^{t_s} \big(v_T - v(t)\big)dt \tag{15}$$

We can replace the upper bound of the integral with ∞, assuming that the CAV has achieved v_T at time t_s. This requires the response time of the controller is short enough (see Appendix A for a discussion on controller design to achieve a short response time) and/or the transmit line is sufficiently far away from the edge of the intersection (see Appendix C.2 for more details). As a result, we can rewrite the position discrepancy as:

$$\begin{aligned} e &= \int_0^{\infty} \big(v_T - (c_1 e^{At} + c_2 e^{Bt} + v_T)\big)dt \\ &= \frac{c_1}{A} + \frac{c_2}{B} \end{aligned} \tag{16}$$

Equation (16) means the CAV will travel e units less than expected if the assigned velocity is higher than its initial velocity (or $|e|$ unit more than expected if the assigned velocity is lower than the initial velocity) during the time interval from t_{Act} to t_s. To compensate this discrepancy, we modify the calculated velocity v_T as:

$$v_T^{new} = \frac{d+e}{t_s - t_{Act}} = v_T + \frac{e}{t_s - t_{Act}} \tag{17}$$

where v_T^{new} is the compensated velocity and d is the traveled distance from t_{Act} to t_s.

As a small example, we simulated a case where the IM does not compensate for CAV's actuation delay and a case where it does. Figure 6 shows the expected (ideal) and actual trajectories and velocity profiles. We can see that the CAV's position at time $t = 2$ (when the CAV is expected to reach the conflict point) will be ahead of the expected position, if the assigned velocity is $v_T = 1m/s$, which is calculated without considering CAV dynamics. However, by modifying the assigned velocity to $v_T^{new} = 0.84m/s$, we can compensate for the position error caused by the actuation time.

5.3 Case 2, with Saturated Acceleration

We now consider the case where the acceleration/deceleration required to track the baseline velocity (Equation (14)) exceeds the acceleration limit. The expected acceleration in order to achieve v_T (derivative of Equation (14)) is:

$$a(t) = Ac_1 e^{At} + Bc_2 e^{Bt} \tag{18}$$

In reality, however, the acceleration rate of a CAV is bounded (Equation (8)). In this case, a CAV faces saturated acceleration and its behavior will be different from what was discussed in the previous section. To determine the compensated target velocity v_T^{new}, we need to simultaneously calculate v_T^{new} and how long the vehicle will accelerate at maximum value. The position discrepancy, on the other hand, is not used in calculating v_T^{new}, unlike in the case of unsaturated acceleration. Readers interested in learning more about position discrepancy in the case of saturated acceleration are referred to Appendix B. Same as in the previous section, our discussion below will focus on the case where v_T is higher than v_0. The case of deceleration can be analyzed similarly.

When a CAV needs to speed up ($v_T > v_0$), and the acceleration exceeds the a_{max}, the CAV will accelerate at a_{max} and maintain it until the input from the controller is less than the limit. We refer to the time when saturated acceleration ends as t_{SAT}. Assuming the initial acceleration is large enough, Equation (18) when the target velocity is v_T^{new} can be written as:

$$a'(t) = \begin{cases} a_{max}, & \text{if } t \le t_{SAT} \\ Ac_1' e^{At} + Bc_2' e^{Bt}, & \text{if } t \ge t_{SAT} \end{cases} \tag{19}$$

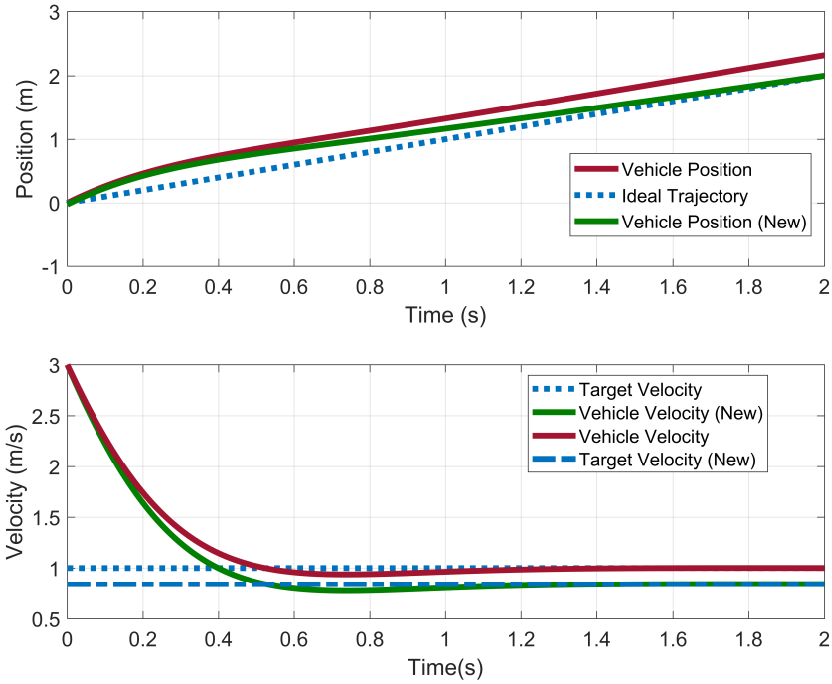


Fig. 6. Position error of a CAV due to neglecting CAV dynamics. The new target velocity compensates the effect of response time on the position error.

where

$$c_1' = \frac{-B}{A-B}(v_{SAT} - v_T^{new})$$

and

$$c_2' = \frac{A}{A-B}(v_{SAT} - v_T^{new})$$

From Equation (19), the velocity profile can be computed as:

$$v'(t) = \begin{cases} v_0 + a_{max}t, & \text{if } t \le t_{SAT} \\ c_1' e^{At} + c_2' e^{Bt} + v_T^{new}, & \text{if } t \ge t_{SAT} \end{cases} \tag{20}$$

We define v_{SAT} as the velocity at time t_{SAT} assuming the initial acceleration is zero:

$$v_{SAT} = t_{SAT} a_{max} + v_0$$

The distance traveled by the CAV can be derived from Equation (20) as:

$$\Delta x = \int_0^{t_{SAT}} (v_0 + a_{max}t)dt + \int_{t_{SAT}}^{t_s} c_1' e^{At} + c_2' e^{Bt} + v_T^{new} dt \tag{21}$$

We define $t' = t - t_{SAT}$ (note that $dt' = dt$) and replace it in the second integral as:

$$\Delta x = \int_0^{t_{SAT}} v_0 + a_{max}t dt + \int_0^{t_s - t_{SAT}} c_1' e^{At'} + c_2' e^{Bt'} + v_T^{new} dt'$$

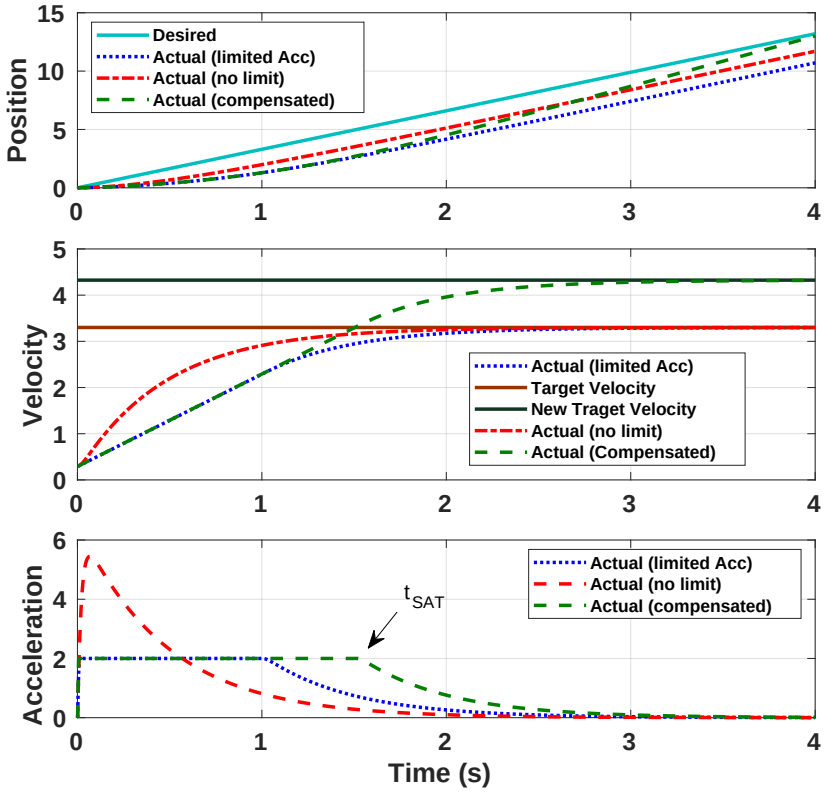


Fig. 7. The behavior of a CAV using when acceleration is limited and when there is no limit on acceleration. Crossroads+ is able to compensate for the effect of acceleration delay in the actuation of CAV.

Solving the integral assuming the response time is fast enough, we have:

$$\Delta x = v_0 t_{SAT} + \frac{a_{max} t_{SAT}^2}{2} + \frac{c_1'}{A} + \frac{c_2'}{B} + v_T^{new}(t_s - t_{SAT})$$

By setting the travelled distance equal to $v_T t_s$, we have:

$$v_T t_s = v_0 t_{SAT} + \frac{a_{max} t_{SAT}^2}{2} + \frac{c_1'}{A} + \frac{c_2'}{B} + v_T^{new}(t_s - t_{SAT}) \tag{22}$$

There are two unknown variables in this equation: t_{SAT} and v_T^{new}. We know that the acceleration is continuous. Therefore, from Equation (19), we can write:

$$a_{max} = Ac_1' e^{At_{SAT}} + Bc_2' e^{Bt_{SAT}} \tag{23}$$

The new target velocity v_T^{new} can now be determined from solving Equation (22) and Equation (23).

Figure 7 shows an example of the actual acceleration profile $a'(t)$ (dotted blue line) compared to the baseline $a(t)$ (red dotted curve) in the bottom sub-figure, the corresponding velocity profiles in the middle sub-figure, and the trajectories in the top sub-figure. By adopting the new target velocity (v_T^{new}), the position error due to acceleration limit will be compensated (green dashed lines).

6 SAFETY PROOF

In this section, we prove that the proposed Crossroads+ methodology is safe. We start by making the following assumptions for the system:

- All CAVs are connected and autonomous (no human driver).
- All CAVs have built-in Adaptive Cruise Control (ACC) system.
- The spatial travel paths of all CAVs are predetermined based on the dimensions of the intersection.
- The position error due to tracking of predetermined paths is within the safety buffer.

Instead of showing that CAV trajectories don't overlap in 2D, we map their trajectories into multiple 1D ones where the longitudinal movement of the CAV matters. We assume that $x(t)$ corresponds to the center of each CAV. For a CAV of length l, a conflict point x_c (as depicted in Figure 5) now becomes a conflict interval:

$$I = [x_c - l, x_c + l]$$

We use induction for the proof and define k to be the number of existing CAVs on the active list.

When $k = 0$, i.e. no existing vehicle, any velocity for the requesting vehicle will be safe.

When $k = 1$, i.e. there is one CAV on the active list, three scenarios can happen: 1) the travel paths of the two CAVs do not intersect each other; 2) the travel paths of the two CAVs intersect at a single point; and 3) the travel paths of two CAVs intersect at more than one point (paths of two CAVs on the same lane will overlap). Since the IM considers the edge of the intersection as the conflict point for two CAVs on the same lane, we can treat this case similar to the single conflict point scenario. For the first scenario, there is no conflict. For the second scenario, we assume the length of the existing CAV is l_E, its velocity is v_E, and its distance from the conflict point is d_E. The conflict time is then:

$$t_c = \frac{d_E}{v_E}$$

Recalling from Equation (1), the IM schedules the requesting vehicle to reach the conflict point at

$$t_s = t_c + \Delta t_{safety}$$

where

$$\Delta t_{safety} = \frac{l_{max} + l_B}{v_{min}}$$

We calculate the position of the existing vehicle at time $t_c + \frac{1}{2}\Delta t_{safety}$:

$$x_E = v_E\left(t_c + \frac{1}{2}\Delta t_{safety}\right) = v_E\left(\frac{d_E}{v_E} + \frac{l_{max} + l_B}{2v_{min}}\right)$$

or

$$x_E = d_E + (l_{max} + l_B)\frac{v_E}{2v_{min}}$$

We know that $v_{min} \leq v_E$ is always true. Therefore, we have:

$$x_E \geq d_E + \frac{l_{max} + l_B}{2}$$

which means the existing vehicle will be completely out of the conflict interval at time $t_c + \frac{1}{2}\Delta t_{safety}$ since $l_{max} \geq l_E$ and therefore,

$$x_E \notin [x_c - l_E, x_c + l_E]$$

On the other hand, assume the distance of the requesting vehicle from the conflict point is d_R. Similarly, we can calculate the position for the requesting CAV at time $t_c + \frac{1}{2}\Delta t_{safety}$ as

$$x_R = v_R\left(t_c + \frac{1}{2}\Delta t_{safety}\right) = v_R\left(t_s - \frac{1}{2}\Delta t_{safety}\right) = v_R\left(\frac{d_R}{v_R} - \frac{l_{max} + l_B}{2v_{min}}\right)$$

or

$$x_R = d_R - (l_{max} + l_B)\frac{v_R}{2v_{min}}$$

considering the fact $v_{min} \leq v_R$,

$$x_R \leq d_R - \frac{l_{max} + l_B}{2}$$

As a result, the requesting vehicle will be outside of the conflict interval:

$$x_R \notin [x_c - l_R, x_c + l_R]$$

because $l_{max} \geq l_R$.

Now, we assume that there are n CAVs on the active list that have already received a target velocity and there is no conflict among them (case $k = n$), i.e.

$$x_i \notin I_{i,j}, \forall x_i, \quad i = 1, ..., n, \quad j = 1, ..., n \quad i \neq j.$$

where x_i is the longitudinal position of a CAV and $I_{i,j}$ is the conflict interval around the conflict point between CAVs i and j projected on travel path of CAV i:

$$I_{i,j} = [x_{i,j} - l_i, x_{i,j} + l_i]$$

where $x_{i,j}$ is the conflict point between CAV i and j. We will show that if a new CAV approaches the intersection and makes a request (case $k = n + 1$), the assigned target velocity to the requesting CAV will be safe.

$$x_i \notin I_{i,j}, \forall x_i, \quad i = 1, ..., n + 1, \quad j = 1, ..., n + 1 \quad i \neq j. \tag{24}$$

We already know that existing vehicles ($i = 1, ...n$) do not have any conflicts. As a result, we can simplify the Equation (24) and rewrite it as:

$$x_i \notin I_{i,n+1}, \quad i = 1, ..., n.$$

and

$$x_{n+1} \notin I_{n+1,i}, \quad i = 1, ..., n.$$

In other words, the new CAV (denoted by index $n + 1$) should not have any conflicts with existing ones. Recalling from Section 4, the IM calculates target velocities with respect to each existing vehicle as:

$$v_T^i = \frac{d^i}{t_c^i + \Delta t_{safety}} \tag{25}$$

and assigns the slowest target velocity among all calculated target velocities to the requesting CAV:

$$v_T \leq v_T^i$$

As a result, the actual reach time of the requesting vehicle to each conflict point, t_{reach}^i will satisfy

$$\frac{d^i}{t_{reach}^i} \leq \frac{d^i}{t_s^i}$$

Since all variable are positive, we can write:

$$t_{reach}^i \geq t_s^i$$

As a result, there won't be any conflict between existing CAVs and the new one.

■

7 OUR TESTBED FOR EVALUATION

We evaluated our algorithm using two testbeds: 1) 1/10 scale model CAVs and 2) our multi-lane simulator.

7.1 1/10 scale RC Cars

In order to demonstrate the effectiveness of our approach, we built a single lane 1/10 scale model intersection (Figure 8) with 8 CAVs [24]. The width of each lane is 0.6 m and the size of the intersection is $1.2 \times 1.2\ m^2$. Our CAVs are RWD (Rear Wheel Drive) cars with Traxxas Slash RC chassis. The size of each car is $0.296\ m \times 0.568\ m$ and has $3.5\ m/s$ maximum speed. $3.5\ m/s$ for our 1/10 scale model corresponds to 78mph in real-life. We used DC motors with built-in quadrature encoder to measure the longitudinal position of the CAV. Encoder data is processed by an Arduino Nano board and the processed data is then sent to the main microcontroller (Arduino Mega 2560). We utilized a Bosch BNO055 absolute orientation sensor which has built-in sensor fusion and fuses the data gathered from a 3-axis magnetometer, accelerometer, and gyroscope. Each car communicates to the server via NRF24L01+, 2.4GHz wireless module and the wireless module communicates with the Arduino Mega via the SPI protocol.

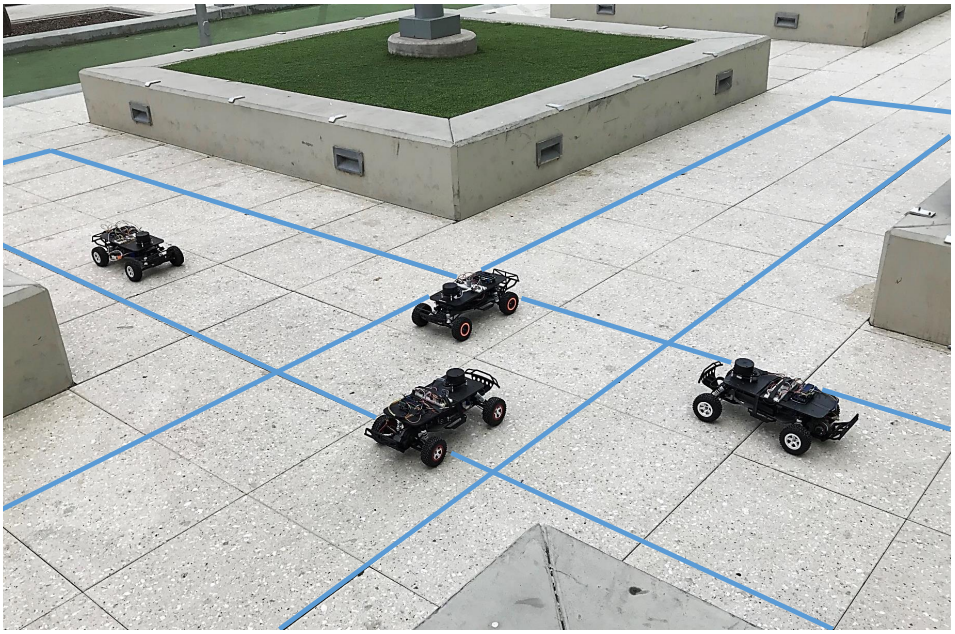


Fig. 8. 1/10 scale CAVs in our experiment. Top speed is 3.5 m/s (78mph in full scale). Intersection and road lines are overlayed for a better intuition.

We set the transmit line to be 3 meters away from the edge of the intersection. CAVs are set to communicate with the IM when they reach the transmit line. The IM consists of a communication station with Arduino Mega 2560 that talks with the other nodes via NRF24L01+,2.4GHz wireless module. Communication station sends the received data to a laptop via serial over the USB port. The IM program is executed on the laptop and is written in Matlab R2016. The laptop has 10 GB

memory, Core i7 -3517u @1.9/2.4GHZ CPU and Windows 8.1 64-bit OS. Each car exchanges 44 bytes of data with IM for communication. The internal clock of all CAVs is synchronized to the IM's clock before reaching the transmit line. We developed Network Time Protocol (NTP) [33] for synchronization and we achieved 1ms accuracy. We measured the WCRTD for our testbed empirically (Figure 3).

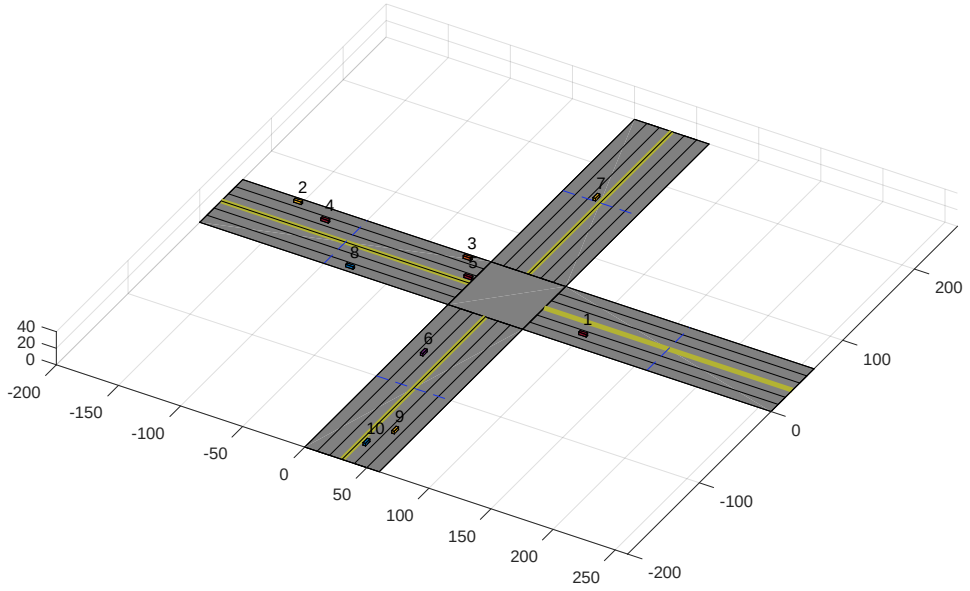


Fig. 9. A snapshot of our simulator in Matlab with three-lane per road.

7.2 Simulator

In order to validate the proposed method for multi-lane intersections and for arbitrary flow rates of approaching CAVs, we developed a simulator in MATLAB [1]. We simulated the IM and CAVs as separate computation nodes and communication between IM and CAVs is done using the network model. The modeled network is actually a buffer of packets with the delivery time. This allows for modeling the network delay by easily adding a random variable to the set delivery time. The random variable (D) is selected from the interval $[0, RTD_{max}/2]$. The simulated CAVs in our simulator are 6 m long and 2 m wide, and the vehicle wheelbase length is 5m. The speed limit is 50 mph ($\approx$ 22.3 m/s). Roads connected to the intersection are 200 m long and lane width is 10 meters. The transmit line is 100 m away from the edge of the intersection. An overview of our simulator is depicted in Figure 9. In this demonstration, left turns are possible from the leftmost lane and right turns are allowed from the rightmost lane. The numbers next to each CAV in Figure 9 corresponds to the CAV's ID.

8 RESULTS

In this section, we first show the result of experiments conducted on our testbed followed by the results of a simulated multi-lane intersection in our simulator.

[1]The simulator is available at https://github.com/mkhayatian/Traffic-Intersection-Simulator-for-Autonomous-Vehicles

8.1 Experiment on our testbed

We compared the throughput of VT-IM and Crossroads+. The throughput is measured in terms of the number of CAVs (8 in our case) divided by the total wait time. The wait time of a CAV is measured as the difference between the time a CAV crosses the transmit line and the time it leaves the intersection. Since existing VT-IM approaches may lead to potential collisions due to ignoring the RTD, we considered an extra buffer of B_{RTD} around each CAV to implement the VT-IM technique without accidents (see Section 3.2). Based on our experiments for measuring the RTD in of 1/10 scale model RC cars, Δt_{WCRTD} is 1800ms (Figure 3). Since the v_{max} is 3.5m/s in our testbed, the required longitudinal safety buffer for VT–IM is 630cm, which almost 10 times of vehicle size (60cm). We tested 10 different traffic scenarios (from light to heavy traffic) using our 1/10 scale model intersection and each scenario was repeated 10 times. In the light traffic scenarios, vehicles are set to drive toward the intersection such that there is the least number of conflicts between approaching CAVs. On the other hand, for heavy traffic scenarios, CAVs are set to reach the transmit line in a short time interval. An example of a light traffic scenario and a heavy traffic scenario is depicted in Figure 10.

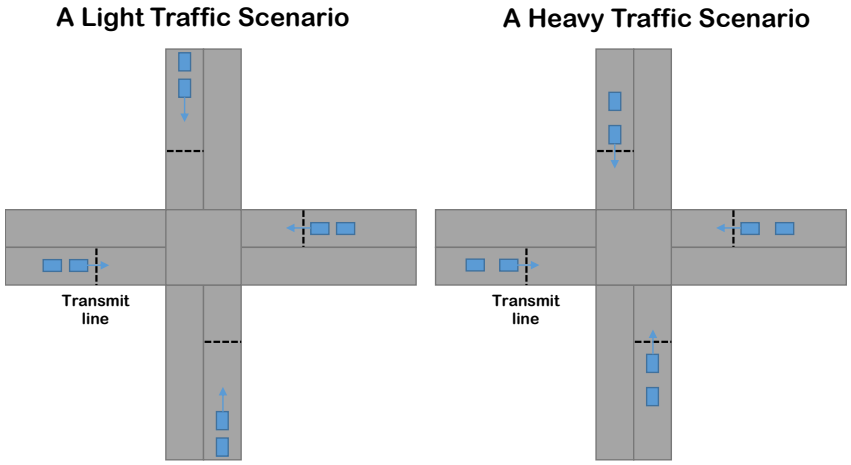


Fig. 10. An example of a light traffic scenario and a heavy traffic scenario created by setting the initial position and velocities of CAVs.

All CAVs first synchronize their clock with the IM. The IM then broadcasts the "start time" of the experiment to all CAVs. CAVs start driving when their local timer is equal to the set start time. This way, all CAVs start driving at the same time no matter where they are placed. Each CAV detects the transmit line by comparing its shaft encoder's value with the hard-coded position of the transmit line. We used the same initial position and velocity for CAVs in both VT-IM and Crossroads+ experiments. The improvement in the average wait time of CAVs in Crossroads+ is depicted in Figure 11. The improvement is calculated by dividing the average wait time for VT-IM by the average wait time for Crossroads+. The first scenario represents a heavy traffic case where CAVs arrive at the transmit line with short time headways. Conversely, scenario 10 represents a light traffic scenario where CAVs are spaced enough to arrive at the transmit line with large time headways. Other scenarios are generated randomly and are sorted based on the reduction in the wait time of CAVs that is achieved in our method compared to VT–IM. Since the size of the intersection and the location of the transmit line is fixed for our experiment, less wait times result

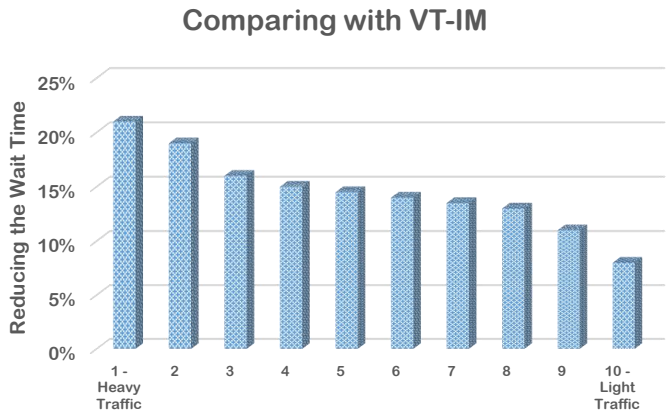


Fig. 11. Improvement in the wait time for 10 different scenarios from heavy to light traffic conducted using our 1/10 scale model CAVs (explained in Section 7). Scenario 1 represents the heaviest traffic case and scenario 10 represents the lightest one.

in higher throughputs. Based on our results, Crossroads+ can achieve 15% better throughput on average in comparison with VT-IM. The main advantage of Crossroads+ is avoiding consideration of an extra safety buffer due to the unknown RTD.

8.2 Simulation of Multi-Lane Intersections

We compared QB-IM, VT-IM, and Crossroads+, in terms of both network overhead and throughput in our simulator. Since the VT-IM and QB-IM are not proposed to account for RTD, we added an extra safety buffer (Equation 5) for each CAV to operate the intersection without any accidents. The size of this buffer is determined from the multiplication of maximum velocity and WCRTD for a realistic case (v_{max} = 22.3 m/s or 50 mph). The maximum latency of DSRC (Dedicated Short-range Communication) is 100 ms [48]. The WCET (Worst-case Execution time) of the FCFS scheduling

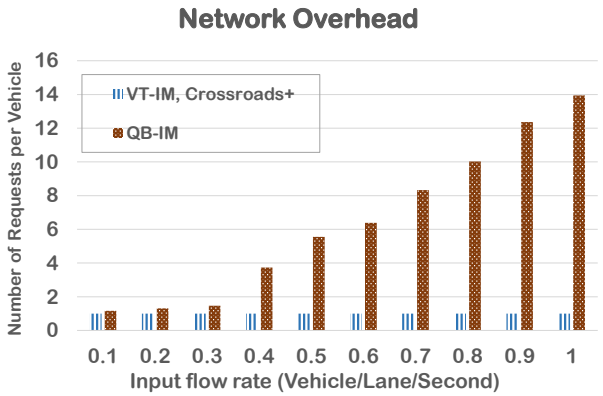


Fig. 12. Comparison of the network overhead of QB-IM approach with VT-IM and Crossroads+ in terms of the average number of messages exchanged between CAVs and IM.

algorithm is 900 ms. Therefore, WCRTD is 0.1 + 0.9 + 0.1 = 1.1 s and the required safety buffer is 1.1 * 22.3 = 24.53 m, which is almost four times of the length of a vehicle. To provide a fair comparison, we conducted a simulation of all approaches with the same configuration. In particular, we used the same time vector and an initial velocity vector for generating CAVs. In terms of network overhead, VT-IM, and Crossroads+ have the same performance since the data is exchanged once unless a packet is dropped. However, due to the trial-and-error nature of the QB-IM approach, its communication overhead for heavy traffic cases is very high. We considered a 1-second timeout between two consecutive requests for the QB-IM method and counted the number of requests from CAVs. Figure 12 shows the average number of requests per CAV for different input flow rates of CAVs. As the input flow rate of the intersection increase in QB-IM, more CAVs fail to get a reservation, which results in a re-request and more network traffic. Based on the results, QB-IM has 14X communication overhead in the worst case.

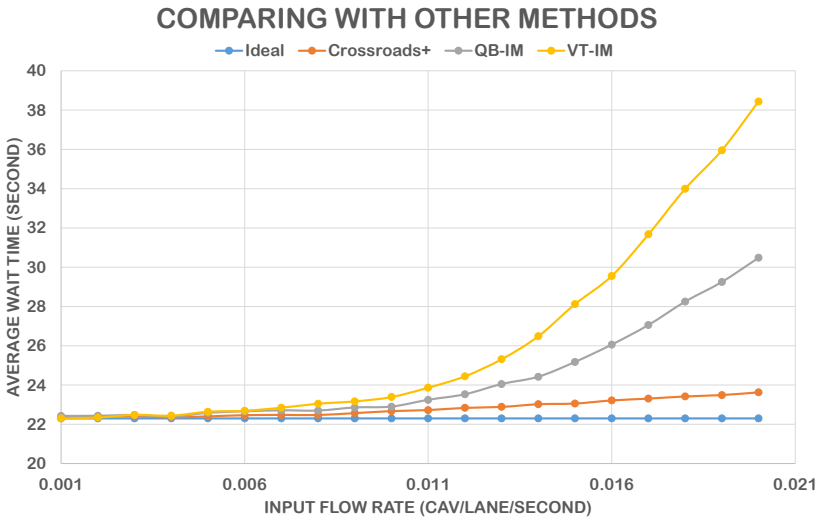


Fig. 13. Comparing Crossroads+ with other techniques implemented in our Simulator. Crossroads+ performs better than other approaches especially in heavy traffic scenarios.

We measured the average travel time of CAVs for different input flow rates. Due to the limitation of VT-IM, it cannot operate for flow rates more than 0.02 CAV/lane/Second. This is because the intersection becomes congested and IM cannot assign a positive velocity to a CAV. Figure 13 depicts the average travel time of the ideal case, QB-IM, VT-IM, and Crossroads+ (CAV per lane per second). The ideal case is indicated by a blue solid line, which represents an intersection with separated roads so that CAV can always pass the intersection while driving at the max velocity. VT-IM performs better than QB-IM because it has the advantage of assigning higher velocities if possible and can avoid the extra safety buffer. On average, Crossroads+ achieves 36% better throughputs in comparison with VT-IM and 16% in comparison with QB-IM.

In Figure 14, we depicted the position and velocity of a CAV for VT-IM, QB-IM and Crossroads+. In VT-IM, the CAV starts tracking the target velocity (v_T = 6) as soon as it's received. In QB-IM, the CAV comes to a complete stop before entering the intersection (at x = 200 m) and then accelerate after 3 second. In Crossroads+, the CAV waits for WCRTD = 1.1 second and then starts tracking

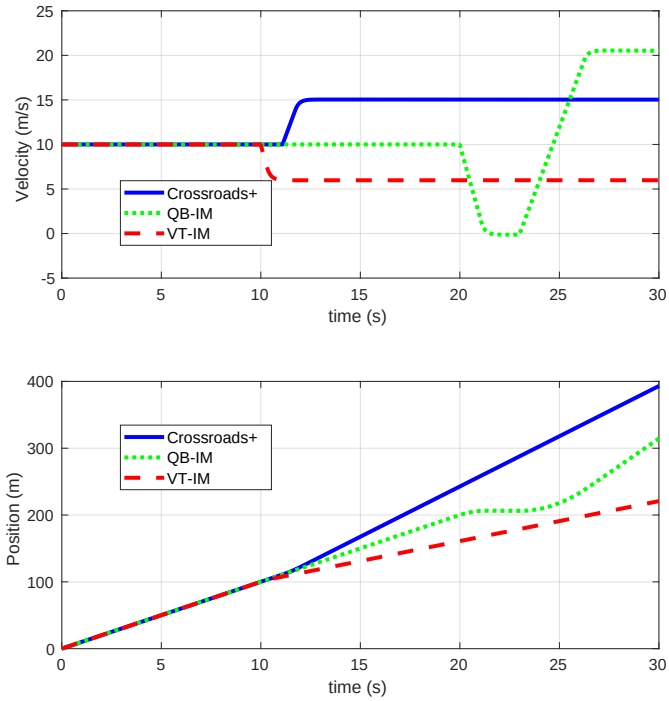


Fig. 14. Comparing behavior of a CAV for VT-IM, QB-IM and Crossroads+ approaches.

the target velocity. The target velocity for the Crossroads+ is most likely greater than VT-IM since VT-IM requires an extra safety buffer due to ignoring the RTD.

We also developed an experiment to observe the effect of the safety buffer size, speed limit, WCRTD and distance of transmit line from the edge of the intersection on the throughput of an intersection. In the first experiment, the average travel time is measured for different value of WCRTD. In the original case, the WCRTD is 1100 ms and in the rest of the cases, it's decreased by 90%, 89%, 70%, 60%, and 50%. We can observe that a smaller WCRTD can slightly increase the throughput and this is because a CAV travels at its initial velocity for a shorter time. In the second experiment, we measured the average travel time of the CAVs for the base case (Vmax = 50 mph) and a reduction in the speed limit (45, 40, 35, 30 and 25 mph). A lower speed limit results in a significant increase in the average travel time of CAVs. This is because with a slower assigned velocity, a CAV occupies the intersection area for a longer time and therefore, delays the scheduling of next CAVs. In the third experiment, the distance of transmit line from the edge of the intersection varies between the original length (100 m) and a 25% increase in the length (125m). It can be observed that the average wait time is reduced when the transmit line is placed farther from the intersection. The reason behind this reduction is better flexibility that is achieved. When the distance of the transmit line increases, the IM can assign a target velocity to a CAV earlier. In the last experiment, we measured the average travel time of the CAVs when the safety buffer is increased by 10%, 20%, 30%, 40% and 50%. Figure 15 shows the sensitivity of throughput on the WCRTD, speed limit, transmit line and safety buffer.

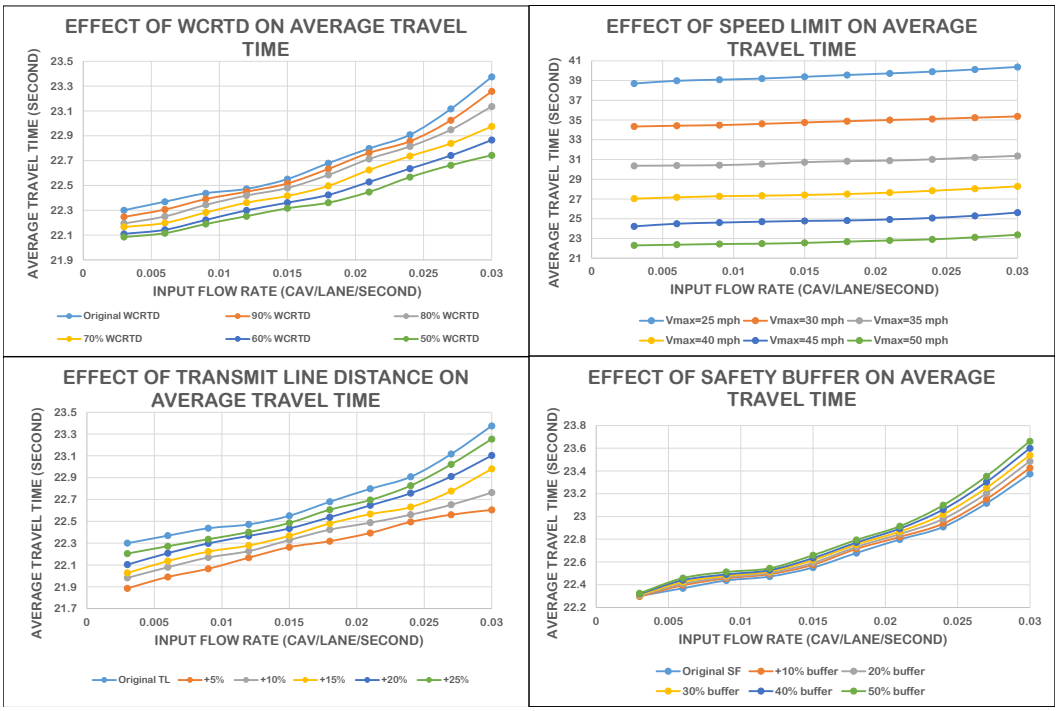


Fig. 15. The effect of WCRTD, speed limit, transmit line and safety buffer on the throughput of the intersection.

We can observe that by increasing the speed limit, we can improve the throughput of the intersection for all input flow rate. However, the increase in the safety buffer only affects the higher input flow rates.

9 CONCLUSION

In this paper, we presented Crossroads+, a time-sensitive approach for intersection management of CAVs that accounts for network delays, IM computation time and CAVs' physical behaviors including acceleration and deceleration capabilities and controller parameters. In Crossroads+, a fixed actuation time is assigned to each CAV to overcome nondeterminism caused by network delay and IM processing time. We verified the effectiveness of Crossroads+ through conducting experiments on a 1/10 scale model intersection of CAVs. We also verified the scalability of our approach using a multi-lane simulator that models network delay. Results from our experiments and simulation indicate that Crossroads+ can achieve higher throughputs in comparison with other approaches while guaranteeing the safety.

Despite the advantages of Crossroads+, there are other challenges that are left open for researchers to tackle. First, Crossroads+ uses an FCFS scheduling policy. Throughput of the intersection can be further improved if an optimization-based scheduling policy is implemented. Second, we assumed that all CAVs follow IM's command, while in reality, a vehicle may break down and fail to follow the assigned trajectory. There are a number of fault models that should be considered before deploying an intersection management technique in real life. Third, in case of an emergency situation, all CAVs will stop. To resume the operation of the intersection once the emergency situation is resolved, a recovery mechanism should be developed and added to both CAVs and IM algorithm.

10 ACKNOWLEDGEMENT

This work was partially supported by funding from NIST Award 70NANB19H144, and by National Science Foundation grants CNS 1525855, CPS 1645578, and CCF 172346 - the NSF/Intel joint research center for Computer Assisted Programming for Heterogeneous Architectures (CAPA).

REFERENCES

[1] Mourad Ahmane, Abdeljalil Abbas-Turki, Florent Perronnet, Jia Wu, Abdellah El Moudni, Jocelyn Buisson, and Renan Zeo. 2013. Modeling and Controlling an Isolated Urban Intersection based on Cooperative Vehicles. *Transportation Research Part C: Emerging Technologies* 28 (2013), 44–62.

[2] Heejin Ahn and Domitilla Del Vecchio. 2016. Safety Verification and Control for Collision Avoidance at Road Intersections. *arXiv preprint arXiv:1612.02795* (2016).

[3] Edward Andert, Mohammad Khayatian, and Aviral Shrivastava. 2017. Crossroads: Time-Sensitive Autonomous Intersection Management Technique. In *Proceedings of the 54th Annual Design Automation Conference 2017*. ACM, 50.

[4] Shunsuke Aoki and Ragunathan Rajkumar. 2018. Dynamic intersections and self-driving vehicles. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 320–330.

[5] Tsz–Chiu Au, Neda Shahidi, and Peter Stone. 2011. Enforcing Liveness in Autonomous Traffic Management. In *24th AAAI Conference on Artificial Intelligence*.

[6] Reza Azimi, Gaurav Bhatia, Ragunathan Raj Rajkumar, and Priyantha Mudalige. 2014. STIP: Spatio-temporal intersection protocols for autonomous vehicles. In *2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 1–12.

[7] Tim Bailey and Hugh Durrant-Whyte. 2006. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robotics & Automation Magazine* 13, 3 (2006), 108–117.

[8] Masoud Bashiri and Cody H Fleming. 2017. A platoon-based intersection management system for autonomous vehicles. In *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 667–672.

[9] Dustin Carlino, Stephen D Boyles, and Peter Stone. 2013. Auction-Based Autonomous Intersection Management. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. IEEE, 529–534.

[10] Rachel Dedinsky, Mohammad Khayatian, Mohammadreza Mehrabian, and Aviral Shrivastava. 2019. A Dependable Detection Mechanism for Intersection Management of Connected Autonomous Vehicles (Interactive Presentation). In *Workshop on Autonomous Systems Design (ASD 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[11] Kurt Dresner and Peter Stone. 2004. Multiagent Traffic Management: A Reservation-based Intersection Control Mechanism. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2*. IEEE Computer Society, 530–537.

[12] Kurt Dresner and Peter Stone. 2008. A Multiagent Approach to Autonomous Intersection Management. *Journal of artificial intelligence research* 31 (2008), 591–656.

[13] Hugh Durrant-Whyte and Tim Bailey. 2006. Simultaneous localization and mapping: part I. *IEEE robotics & automation magazine* 13, 2 (2006), 99–110.

[14] Mourad Elhadef. 2015. An adaptable inVANETs-based intersection traffic control algorithm. In *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*. IEEE, 2387–2392.

[15] Mohammed Elhenawy, Ahmed A Elbery, Abdallah A Hassan, and Hesham A Rakha. 2015. An intersection game-theory-based traffic control algorithm in a connected vehicle environment. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, 343–347.

[16] Chien-Liang Fok *et al.* 2012. A platform for evaluating autonomous intersection management policies. In *ICCPS*.

[17] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. 2015. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review* 43, 1 (2015), 55–81.

[18] Jean Gregoire, Silvère Bonnabel, and Arnaud de La Fortelle. 2013. Optimal cooperative motion planning for vehicles at intersections. *arXiv preprint arXiv:1310.7729* (2013).

[19] Jason Gross, Yu Gu, Srikanth Gururajan, Brad Seanor, and Marcello Napolitano. 2010. A comparison of extended Kalman filter, sigma-point Kalman filter, and particle filter in GPS/INS sensor fusion. In *AIAA guidance, navigation, and control conference*. 8332.

[20] Qiu Jin, Guoyuan Wu, Kanok Boriboonsomsin, and Matthew Barth. 2012. Advanced intersection management for connected vehicles using a multi-agent systems approach. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*. IEEE, 932–937.

[21] Qiu Jin, Guoyuan Wu, Kanok Boriboonsomsin, and Matthew Barth. 2012. Multi-agent intersection management for connected vehicles using an optimal scheduling approach. In *Connected Vehicles and Expo (ICCVE), 2012 International Conference on*. IEEE, 185–190.

[22] Qiu Jin, Guoyuan Wu, Kanok Boriboonsomsin, Matthew J Barth, et al. 2013. Platoon-based multi-agent intersection management for connected vehicle.. In *ITSC*. 1462–1467.
[23] Alexander Katriniok, Stefan Kojchev, Erjen Lefeber, and Henk Nijmeijer. 2018. Distributed Scenario Model Predictive Control for Driver Aided Intersection Crossing. In *2018 European Control Conference (ECC)*. IEEE, 1746–1752.
[24] Mohammad Khayatian, Aviral Shrivastava, and Mohammadreza Mehrabian. 2018. RIM: Robust Intersection Management for Connected Autonomous Vehicles. In *Proceedings of the 39th Real-Time Systems Symposium, RTSS*. IEEE.
[25] Joyoung Lee and Byungkyu Park. 2012. Development and Evaluation of A Cooperative Vehicle Intersection Control Algorithm under The Connected Vehicles Environment. *IEEE ITS Transactions* (2012).
[26] Joyoung Lee, Byungkyu Brian Park, Kristin Malakorn, and Jaehyun Jason So. 2013. Sustainability assessments of cooperative vehicle intersection control at an urban corridor. *Transportation Research Part C: Emerging Technologies* 32 (2013), 193–206.
[27] Li Li and Fei-Yue Wang. 2006. Cooperative driving at blind crossings using intervehicle communication. *IEEE Transactions on Vehicular technology* 55, 6 (2006), 1712–1724.
[28] Peiqun Lin, Jiahui Liu, Peter J Jin, and Bin Ran. 2017. Autonomous vehicle-intersection coordination method in a connected vehicle environment. *IEEE Intelligent Transportation Systems Magazine* 9, 4 (2017), 37–47.
[29] Jennie Lioris, Ramtin Pedarsani, Fatma Yildiz Tascikaraoglu, and Pravin Varaiya. 2017. Platoons of connected vehicles can double throughput in urban roads. *Transportation Research Part C: Emerging Technologies* 77 (2017), 292–305.
[30] Andreas A. Malikopoulos, Christos G. Cassandras, and Yue J. Zhang. 2016. A Decentralized Energy-Optimal Control Framework for Connected Automated Vehicles at Signal-Free Intersections. *arXiv preprint arXiv:1602.03786* (2016).
[31] Mohammadreza Mehrabian, Mohammad Khayatian, Aviral Shrivastava, John C Eidson, Patricia Derler, Hugo A Andrade, Ya-Shian Li-Baboud, Edward Griffor, Marc Weiss, and Kevin Stanton. 2017. Timestamp Temporal Logic (TTL) for Testing the Timing of Cyber-Physical Systems. *ACM Transactions on Embedded Computing Systems (TECS)* 16, 5s (2017), 169.
[32] Vicente Milanés, Joshué Pérez, Enrique Onieva, and Carlos González. 2010. Controller for Urban Intersections Based on Wireless Communications and Fuzzy Logic. *IEEE Transactions on Intelligent Transportation Systems* 11, 1 (2010), 243–248.
[33] David Mills et al. 1985. *Network time protocol*. Technical Report. RFC 958, M/A-COM Linkabit.
[34] Nikolce Murgovski, Gabriel Rodrigues de Campos, and Jonas Sjöberg. 2015. Convex modeling of conflict resolution at traffic intersections. In *2015 54th IEEE conference on decision and control (CDC)*. IEEE, 4708–4713.
[35] Norbert Neuendorf and Torsten Bruns. 2004. The vehicle platoon controller in the decentralised, autonomous intersection management of vehicles. In *Mechatronics, 2004. ICM'04. Proceedings of the IEEE International Conference on*. Ieee, 375–380.
[36] Enrique Onieva, Unai Hernández-Jayo, Eneko Osaba, Asier Perallos, and Xiao Zhang. 2015. A multi-objective evolutionary algorithm for the tuning of fuzzy rule bases for uncoordinated intersections in autonomous driving. *Information Sciences* 321 (2015), 14–30.
[37] Jianglin Qiao, Dongmo Zhang, and Dave de Jonge. 2018. Virtual Roundabout Protocol for Autonomous Vehicles. In *Australasian Joint Conference on Artificial Intelligence*. Springer, 773–782.
[38] Michael Quinlan, Tsz-Chiu Au, Jesse Zhu, Nicolae Stiurca, and Peter Stone. 2010. Bringing Simulation to Life: A Mixed Reality Autonomous Intersection. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 6083–6088.
[39] Peyman Setoodeh, Alireza Khayatian, and Ebrahim Farjah. 2007. Attitude estimation by divided difference filter-based sensor fusion. *The journal of navigation* 60, 1 (2007), 119–128.
[40] Peyman Setoodeh, Alireza Khayatian, and Ebrahim Frajah. 2004. Attitude estimation by separate-bias Kalman filter-based data fusion. *The Journal of Navigation* 57, 2 (2004), 261–273.
[41] Jiahe Shi, Yi Zheng, Yuning Jiang, Mario Zanon, Robert Hult, and Boris Houskal. 2018. Distributed control algorithm for vehicle coordination at traffic intersections. In *2018 European Control Conference (ECC)*. IEEE, 1166–1171.
[42] Aviral Shrivastava, Patricia Derler, Ya-Shian Li Baboudr, Kevin Stanton, Mohammad Khayatian, Hugo A Andrade, Marc Weiss, John Eidson, and Sundeep Chandhoke. 2016. Time in cyber-physical systems. In *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2016 International Conference on*. IEEE, 1–10.
[43] Aviral Shrivastava, Mohammadreza Mehrabian, Mohammad Khayatian, Patricia Derler, Hugo Andrade, Kevin Stanton, Ya-Shian Li-Baboud, Edward Griffor, Marc Weiss, and John Eidson. 2017. A testbed to verify the timing behavior of cyber-physical systems. In *Proceedings of the 54th Annual Design Automation Conference 2017*. ACM, 69.
[44] Erik Steinmetz, Robert Hult, Zhenhua Zou, Ragne Emardson, Fredrik Brännström, Paolo Falcone, and Henk Wymeersch. 2018. Collision-Aware Communication for Intersection Management of Automated Vehicles. *IEEE Access* 6 (2018), 77359–77371.
[45] Remi Tachet, Paolo Santi, Stanislav Sobolevsky, Luis Ignacio Reyes-Castro, Emilio Frazzoli, Dirk Helbing, and Carlo Ratti. 2016. Revisiting street intersections using slot-based systems. *PloS one* 11, 3 (2016), e0149607.

[46] Teng-Tiow Tay, Iven Mareels, and John B Moore. 2012. *High performance control.* Springer Science & Business Media.
[47] Seong-hoon Peter Won, Wael William Melek, and Farid Golnaraghi. 2010. A Kalman/particle filter-based position and orientation estimation method using a position sensor/inertial measurement unit hybrid system. *IEEE Transactions on Industrial Electronics* 57, 5 (2010), 1787–1798.
[48] Zhigang Xu, Xiaochi Li, Xiangmo Zhao, Michael H Zhang, and Zhongren Wang. 2017. DSRC versus 4G-LTE for connected vehicle applications: A study on field experiments of vehicular communication performance. *Journal of Advanced Transportation* 2017 (2017).
[49] Kaidi Yang, S Ilgin Guler, and Monica Menendez. 2016. Isolated intersection control for various levels of vehicle technology: Conventional, connected, and automated vehicles. *Transportation Research Part C: Emerging Technologies* 72 (2016), 109–129.
[50] Yue J Zhang, Andreas A Malikopoulos, and Christos G Cassandras. 2016. Optimal control and coordination of connected and automated vehicles at urban traffic intersections. In *2016 American Control Conference (ACC)*. IEEE, 6227–6232.
[51] Xing Zhao, Jinxiang Wang, Yifeng Chen, and Guodong Yin. 2018. Multi-objective Cooperative Scheduling of CAVs at Non-Signalized Intersection. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 3314–3319.
[52] Ismail H Zohdy, Raj Kishore Kamalanathsharma, and Hesham Rakha. 2012. Intersection Management for Autonomous Vehicles using iCACC. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*. IEEE, 1109–1114.
[53] Ismail H Zohdy and Hesham A Rakha. 2016. Intersection management via vehicle connectivity: The intersection cooperative adaptive cruise control system concept. *Journal of Intelligent Transportation Systems* 20, 1 (2016), 17–32.

A CONTROLLER DESIGN

For controller design, there are three gains to be determined: K_P, K_I, and K_D. Different values of K_P, K_I, and K_D will lead to different response times to achieve the target velocity. It is desired that the PID gains are such that the target velocity is achieved quickly. We use settling time as a metric for the CAV response time, which is defined as "the time it takes to reach and stay within 2% of the final value" and can be calculated as [46]:

$$T_s = \frac{3.9}{\zeta \omega_n} \tag{26}$$

where ζ is the damping ratio and ω_n is the natural frequency of the system [46]:

$$\zeta = \frac{K_P}{\sqrt{K_I(1+K_D)}}$$

$$\omega_n = \sqrt{\frac{K_I}{1+K_D}}$$

We define a cost function to minimize the settling time of the system:

$$min \ \ \frac{1}{\zeta \omega_n} \tag{27}$$

Based on the values of the PID controller (K_P, K_I, K_D), roots of the characteristic Equation (11) can be pure real or complex that correspond to overdamped and underdamped responses respectively. An underdamped response is not suitable for controlling the velocity because CAV's velocity will oscillate (around the set target velocity) before reaching the steady state. As a result, we only consider an overdamped situation (no oscillation). We rewrite the this requirement as a constraint for the optimization problem:

$$K_P^2 - 4K_I(1+K_D) > 0$$

Also, to ensure the stability of the closed-loop system, we have:

$$K_P > \sqrt{K_P^2 - 4K_I(1+K_D)}$$

B POSITION DISCREPANCY FOR SATURATED CASE

The acceleration at time zero for the saturate case will be:

$$\dot{a}(0) = A^2 c_1 + B^2 c_2$$

$$= \frac{(v_T - v_0)}{A-B}(-BA^2 + AB^2)$$

$$= (v_0 - v_T)(A^2 + B^2)$$

Since the value of A or B is selected to be large, the initial acceleration will be large enough.

We can calculate the position error for the actual behavior with saturated acceleration:

$$e = \int v_T dt - \int v'(t) dt$$

$$= \int_0^{t_s} v_T dt - \tag{28}$$

$$\left(\int_0^{t_{SAT}} (v_0 + a_{max} t) dt + \int_{t_{SAT}}^{t_s} (c_1' e^{At} + c_2' e^{Bt} + v_T) dt \right)$$

We can use a change of variable ($t' = t - t_{SAT}$) in the last integral as:

$$\int_0^{t_s - t_{SAT}} (c_1' e^{At'} + c_2' e^{Bt'} + v_T) dt'$$

Finally, we can calculate the error as:

$$e = v_T t_s - \left(\frac{a_{max}(t_{SAT})^2}{2} + v_0 t_{SAT} + v_T(t_c - t_{SAT} - \frac{c_1'}{A} - \frac{c_2'}{B})\right) \tag{29}$$

C REAL-LIFE PROJECTION

In this section, we elaborate on details of how Crossroads+ can be implemented in real-life.

C.1 Slow-down Line

Slow-down lines are computed according to the max velocity and maximum deceleration rate of each CAV. Assuming a CAV is driving at velocity v and has maximum deceleration a_{min}, it will take

$$t_{SD} = \frac{v}{a_{min}} \tag{30}$$

seconds to come to a complete stop if the settling time of the response is short. Plugging in the deceleration time into the equation of motion of CAV, we can calculate the travel distance:

$$D_{SDL} = \frac{1}{2} a_{min} \left(\frac{v}{a_{min}}\right)^2 + v\left(\frac{v}{a_{min}}\right) \tag{31}$$

Simplifying, we can obtain the distance of the slow-down line (D_{SDL}) from the stop lines (transmit line and edge of the intersection):

$$D_{SDL} = \frac{3v^2}{2a_{min}}$$

C.2 Transmit Line

As we have discussed in previous sections, a key assumption is that CAVs are able to reach the assigned velocity before they reach the conflict point (x_c). As a result, we need to ensure that the transmit line is set sufficiently back from the intersection. By doing so, the IM would also be more flexible to assign a safe velocity to CAVs because not all CAVs are able to accelerate/decelerate quickly when the distance for the response is short.

According to Equation (3), the actuation time will be set Δt_{WCRTD} seconds after the request time. As a result, the CAV requires $v_0(\Delta t_{WCRTD})$ meters space while it's waiting for the response. Additionally, a CAV requires some time to maintain the target velocity which will be the summation of acceleration time under saturated acceleration (if any) and settling time of the controller. We can find the required distance as:

$$D_{TL} = v_0(\Delta t_{WCRTD}) + \frac{v_T - v_0}{a_{min}} \left(\frac{v_T - v_0}{2}\right) + T_s(v_S^{AVG}) \tag{32}$$

In this equation, D_{TL} is the required distance for the transmit line, v_0 is the initial velocity of the CAV, v_T is the target velocity assigned to the CAV, a_{max} is the maximum acceleration during the saturated behavior, T_s is the settling time of the controller (see Equations (26) and (27)) and v_s^{AVG} is the average velocity of the CAV during the settling time. We can find the worst-case scenario (maximum required distance) as:

$$D_{TL} = v_{max}(\Delta t_{WCRTD}) + \frac{v_{max}}{a_{min}^{slowest}} \left(\frac{v_{max}}{2}\right) + T_s(v_{max}) \tag{33}$$

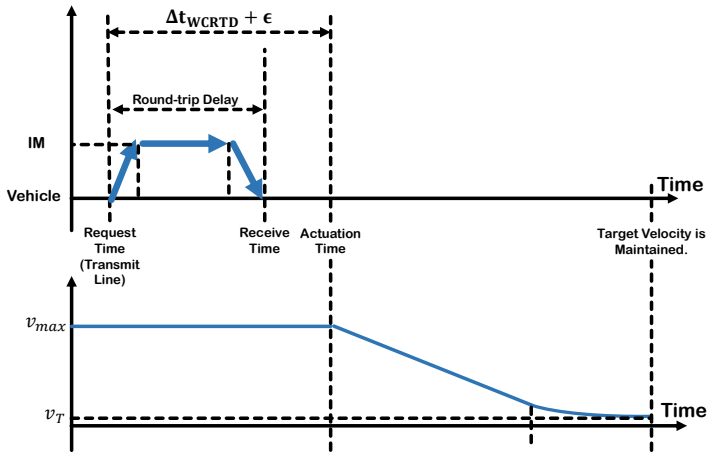


Fig. 16. In the worst-case behavior, a CAV is driving at the maximum velocity and the assigned target velocity is the minimum possible velocity.

$a_{min}^{slowest}$ is the slowest deceleration rate among all CAVs. We use an upper bound for v_S^{AVG} and replace it with v_{max} to make the calculation easier. Figure 16 shows the scenario where the traveled distance is longest. Assuming that $v_{max} = 20m/s$ ($\approx$ 45 mph), $a_{min}^{slowest} = 3m/s^2$, $\Delta t_{WCRTD} + \epsilon = 1s$ and $T_s = 0.1s$, we can find the distance of the transmit line from the edge of the intersection for a real scenario as $D_{TL} = 88.6m$.