RIM: Robust Intersection Management for Connected Autonomous Vehicles

Mohammad Khayatian, Mohammadreza Mehrabian and Aviral Shrivastava
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe, Arizona 85281
Email: {mkhayati,mmehrabi,Aviral.Shrivastava}@asu.edu

***Abstract*—Utilizing intelligent transportation infrastructures can significantly improve the throughput of intersections of Connected Autonomous Vehicles (CAV), where an Intersection Manager (IM) assigns a target velocity to incoming CAVs in order to achieve a high throughput. Since the IM calculates the assigned velocity for a CAV based on the model of the CAV, it's vulnerable to model mismatches and possible external disturbances. As a result, IM must consider a large safety buffer around all CAVs to ensure a safe scheduling, which greatly degrades the throughput. In addition, IM has to assign a relatively lower speed to CAVs that intend to make a turn at the intersection to avoid rollover. This issue reduces the throughput of the intersection even more. In this paper, we propose a space and time-aware technique to manage intersections of CAVs that is robust against external disturbances and model mismatches. In our method, RIM, IM is responsible for assigning a safe Time of Arrival (TOA) and Velocity of Arrival (VOA) to an approaching CAV such that trajectories of CAVs before and inside the intersection does not conflict. Accordingly, CAVs are responsible for determining and tracking an optimal trajectory to reach the intersection at the assigned TOA while driving at VOA. Since CAVs track a position trajectory, the effect of bounded model mismatch and external disturbances can be compensated. In addition, CAVs that intend to make a turn at the intersection do not need to drive at a slow velocity before entering the intersection. Results from conducting experiments on a 1/10 scale intersection of CAVs show that RIM can reduce the position error at the expected TOA by 18X on average in presence of up to 10% model mismatch and an external disturbance with an amplitude of 5% of max range. In total, our technique can achieve 2.7X better throughput on average compared to velocity assignment techniques.**

***Index Terms*—Connected Autonomous Vehicles, Traffic Intersection Management, Cyber-Physical Systems**

I. Introduction

According to the US Federal Highway Administration (FHA), around 30 percent of fatal crashes between 2010 and 2015 have happened in intersection areas, most of which, involved human errors. Furthermore, each person in the US spends around 42 hours stuck in the traffic per year[1]. The advent of Connected Autonomous Vehicles (CAV) promises to drastically reduce the traffic fatalities and improve throughputs of transportation infrastructures. This promise has spurned both cooperative [2], [3] and centralized [4], [5] approaches to manage traffic intersections for CAVs. Centralized approaches are relatively more popular due to security concerns of vehicle-to-vehicle communication of cooperative approaches and their need for high network bandwidth.

Existing centralized approaches to manage an intersection of CAV are categorized as: i) Query-based Intersection Management (QB-IM), and ii) Velocity-assignment Intersection Management (VA-IM). In QB-IM techniques [6], approaching CAVs send a request with their estimated Velocity of Arrival (VOA) and Time of Arrival (TOA) to the Intersection Manager (IM) in order to reserve a time-space slot in the intersection. Accordingly, IM either accepts and reserves the space-time for the requesting CAV or rejects the request. If a CAV's request is rejected, it slows down and prepares to stop before entering the intersection and re-requests after a waiting time. In Contrast, in VA-IM approaches [4], [5], [7], each CAV lets the IM know about its current position and velocity, and then, the IM assigns a target velocity to the CAV, such that it will enter and exit the intersection without any collision.

QB-IM techniques are unable to achieve the maximum throughput of the intersection since when vehicles make a request for a space-time allotment of the intersection, they have no idea of the status at the intersection. As a result, their requests are often rejected, and it takes them a lot of tries to acquire a reservation. This also increases the network traffic. On the other hand, VA-IM approaches can achieve higher throughputs since IM can specify the arrival time of vehicles. As a result, VA-IM approaches are more efficient and popular.

Irrespective of the management technique, IM must consider a safety buffer around vehicles to account for errors in the position caused by sensors (GPS, encoder, etc.). The size of the buffer depends on the accuracy of sensors and maximum velocity of the vehicle [6], [8]. Also, for correct operation of the system, all units should have the same notion of time [9], [10]. When a vehicle sends its information, IM needs to know when the request was initiated (i.e. The time at which the position was measured by the vehicle). Additionally, IM processing time and network delay are not zero and late receive of information can cause a problem. For instance, if a vehicle receives the assigned velocity from the IM with t seconds delay, it will maintain the assigned velocity t seconds later than it supposed to and hence, it will enter the intersection later or earlier that it's expected (earlier if the vehicle was supposed to slow down and actuates late). As a result, IM needs to consider an extra safety buffer for each vehicle, depending on the Worst-Case Network Delay (WCND) and Worst-Case Execution Time (WCET) of the IM [11]. The set of IM and vehicles can be interpreted as a distributed real-time system where multiple processing units (vehicles) exchange real-time data (position, velocity, etc.) with the IM.

Crossroads technique [8] solves this issue by performing clock synchronization among all nodes and sending a fixed actuation time (the time a vehicle begins maintaining the assigned velocity) along with the assigned velocity to vehicles.

Subsequently, Crossroads approach can avoid considering the extra safety buffer and achieve higher throughputs. Crossroads and generally all VA-IM approaches use a constant velocity for vehicles to be tracked, which make them vulnerable to model mismatches and external disturbances. This is because the assigned velocity is computed based on the considered model for the vehicle and the actual behavior of the vehicle can be different from the expected one. Additionally, external disturbances like wind, road slope, bumps, etc. can temporarily degrade the velocity tracking mechanism and result in errors in the expected TOA of vehicles. As a result, VA-IM approaches, including Crossroads are not robust against model mismatches and random disturbances. In addition, for vehicles that intend to make a turn at the intersection, the assigned velocity should be low enough to avoid rollover at the intersection, which reduces the throughput of the intersection.

In this paper, we first investigate the effect of model mismatches and external disturbances on the behavior of vehicles communicating with the IM and then, we propose a robust intersection management scheme for CAVs called RIM. In our approach, each vehicle sends a request to the IM containing its current position, velocity, acceleration and the corresponding timestamp upon crossing the transmit line. Then, IM calculates a safe TOA and VOA for the vehicle such that there will be no conflict in the intersection and, sends them back to the vehicle. Based on the assigned TOA and VOA, the vehicle creates an optimal position trajectory and tracks it. Since each vehicle tracks a reference position trajectory instead of constant velocity, it can compensate for the effect of external disturbances and is robust against model mismatches. Additionally, vehicles that intend to make a turn at the intersection can drive at higher velocities before entering the intersection and greatly improve the throughput of the intersection.

In order to evaluate RIM, we built a 1/10 scale intersection of RC autonomous vehicles. We conducted two types of experiments and measured the position error of CAVs at the expected TOA where the position error is caused by i) model mismatches in the parameters of the model (we added $\pm 10\%$ error to values of PID controller which changes the response time of the controller) and external disturbances (we added a constant disturbance in form of a step function with amplitude of 5% of the max range to the generated input for the motor). Results from conducting experiments on our testbed indicate that our technique, RIM, can reduce the position error by 18X on average and 25X in the best-case in comparison with the Crossroads technique. In addition, by managing the speed of vehicles that intend to make a turn at the intersection, our approach is able to increase the throughput of the intersection by 2.7X on average in comparison with Crossroads wherein the turn velocity limit varies from 0.4 to 1.4 m/s (9 mph to 31 mph for a real-world intersection).

II. Related Work

In the literature, many researchers have focused on managing intersections of autonomous vehicles. We categorize the previous works as centralized and cooperative techniques. In centralized approaches, a local infrastructure coordinates the status of incoming vehicles [6]–[8] while in cooperative ones, vehicles communicate with each other to decide on who goes first [2], [3], [12]–[14]. Cooperative approaches are, however, less popular due to security reasons. Centralized techniques are categorized into QB-IM and VA-IM approaches. Figure 1 shows an abstract comparison between existing intersection management approaches.

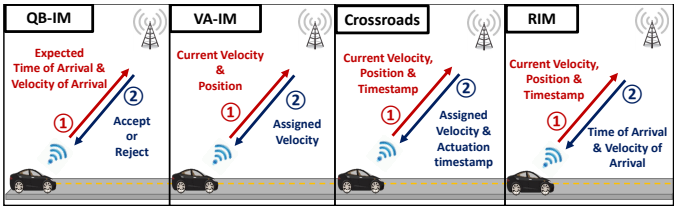

Fig. 1. In QB-IMs, approaching vehicles propose a time-space slot in the intersection, and the IM replies with a yes or no. In VA-IMs, vehicles report their position, velocity, and timestamp when they arrive, and the IM assigns them a velocity (hence velocity assignment) at which to drive. In Crossroads, a timestamp is used to make the system robust against network delay. In the proposed approach RIM, vehicles report their position, velocity, and timestamp, and the IM assigns them a VOA and TOA. It is upon the vehicle on how to arrive at the intersection at TOA at VOA. RIM is robust against model mismatches and external disturbances, as the vehicles can compensate for them.

Autonomous Intersection Management (AIM) [6], [15]–[18] technique is a QB-IM approach introduced by Stone *et al.*, where vehicles query a safe pass from the IM. In AIM, each vehicle sends a request to the IM upon approaching the intersection to reserve a set of time-space tiles in the intersection. Accordingly, IM checks a binary reservation map and either accepts or rejects the request by sending a yes/no response. If a request is rejected, the vehicle slows down and prepares to stop behind the intersection line and will request again after a set timeout. Otherwise, it will continue at its current speed. AIM has a high communication overhead because vehicles have to re-request when their request is rejected. In addition, AIM cannot achieve the maximum throughput of the intersection since vehicles have to slow down and sometimes come to a complete stop if they cannot reserve a spot. In AIM technique, there is a trade-off between throughput and network overhead. Shorter timeouts for re-requesting will result in better throughputs but higher network traffic and, longer timeouts will result in lower network traffic but worse throughputs. Another query-based technique was introduced in 2012 by Jin *et al.* [19] where vehicles constantly send a request to the IM until they get a reservation. Authors used many unrealistic assumptions like no network drop, unlimited network capacity, etc.

To overcome the communication overhead and low throughput issues of QB-IM methods, researchers have proposed Velocity Assignment Intersection Management (VA-IM) techniques where IM coordinates the incoming vehicles by assigning a target velocity to them [4], [5], [7]. In VA-IM approaches, each vehicle sends its position, velocity, and other information to the IM, and IM computes a target velocity based on a First-Come-First-Served (FCFS) policy or other efficient policies like [20]–[22] such that all vehicles enter

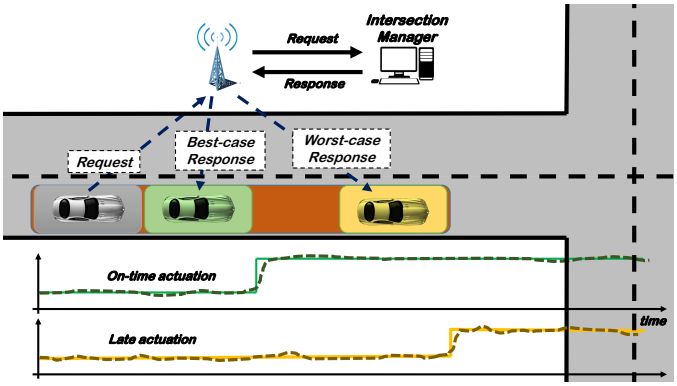

Fig. 2. The effect of network delay and IM processing time on the behavior of a vehicle in VA-IM techniques. All VA-IM techniques require considering an extra safety buffer (the brown box) to ensure the safety of vehicles.

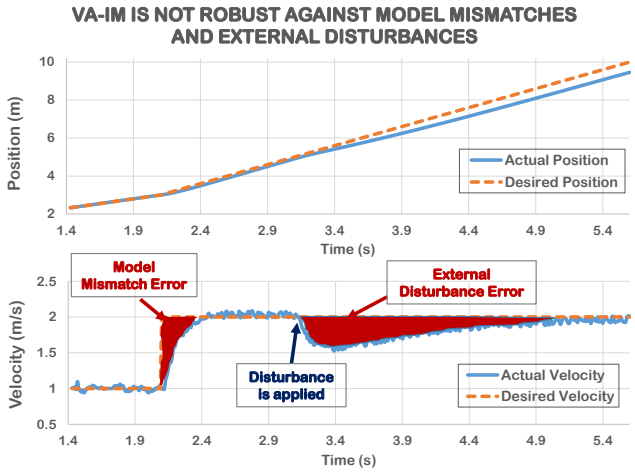

Fig. 3. All VA-IM techniques and even Crossroads are vulnerable to model mismatch and external disturbances. The top figure shows the actual and expected position of the vehicle and the bottom one shows the actual and expected velocity of the vehicle in presence of model mismatch and an external disturbance.

the intersection safely and sends them back to the requesting vehicle. However, VA-IM techniques neglect the effect of network delay due to communication and the delay caused by the IM when processing the request. As a result, a vehicle will receive the target velocity with an unknown delay and the time it starts accelerating/decelerating is delayed. We refer to the summation of WCND in the forward and backward path and, WCET of the IM as Worst-Case Round-trip Delay (WCRTD). Since the value of WCRTD is not known beforehand, it cannot be compensated by the IM. Hence, the eventual position of the vehicle (and therefore its TOA) is erroneous. Figure 2 shows a scenario where the assigned velocity is greater than the current velocity of the vehicle and network delay and IM processing time cause a late actuation. As a result, IM should consider an extra safety buffer (the brown buffer in Figure 2) for each vehicle to ensure their safety. The result buffer has the same lateral size as the safety buffer while the longitudinal size is extended relative to the WCRTD and maximum velocity of the vehicle. The longitudinal size of the extra safety buffer can be as large as 3X of a vehicle size [8]. Considering such a large buffer greatly reduces the throughput of the intersection and makes such VA-IM techniques impractical to use. In 2017, Andert *et al.* proposed a velocity assignment approach, Crossroads [8], that can skip considering an extra buffer due to WCRTD. In this technique, all vehicles first synchronize their local clock with the IM and, then send their information (position, velocity, etc.) with a timestamp to the IM. As a result, Both IM and the requesting vehicle have the same notion of time. Accordingly, IM calculates an actuation timestamp along with the target velocity to fix the actuation time of the vehicle (When to start accelerating/decelerating). Crossroads can achieve 1.6X better throughput in comparison with regular VA-IM approaches thanks to avoiding considering an extra safety buffer.

Despite the fact that vehicles have a deterministic behavior in Crossroads technique, our experiment on our 1/10 scale intersection with autonomous vehicles (explained in Section IV) shows that accidents can happen. This is because IM needs an accurate model of a vehicle in order to compute a safe target velocity for the vehicle and any mismatches between the actual model and the considered one by the IM can cause errors in the expected TOA of the vehicle and cause accidents. Specifically, the response time of a vehicle (acceleration or deceleration time) can be different from the expected one, which depends on the dynamics of the vehicle as well as the control algorithm. IM should know how long acceleration/deceleration time of a vehicle is and how long it takes to maintain the assigned velocity. Moreover, vehicles track a constant velocity in VA-IM and Crossroads approaches and not a position trajectory. Thus, external disturbances like wind, slope, etc. can temporarily prevent them from tracking the assigned velocity and therefore, the eventual arrival time of the vehicle varies and can cause accidents. In order to demonstrate this issue, we performed an experiment on our 1/10 scale model autonomous vehicle (Section IV) and measured the position and velocity of a vehicle in presence of an external disturbance (a step input with 0.1X amplitude was added to the controller's input to the motor) and model mismatches (up to 10% in parameters of the PID controller and actuator gain). Figure 3 shows the position and velocity trajectories of the vehicle and the effect of model mismatches and the external disturbance on the eventual position of the vehicle. One can observe that the vehicle has a 0.54 m position error when it enters the intersection, which is almost equal to the size of the vehicle (vehicle length is 0.6m). This experiment shows that VA-IM approaches including Crossroads should consider an extra safety buffer due to the model mismatch and possible disturbances to ensure the safety of vehicles.

Another practical issue of VA-IM approaches and Crossroad is associated with the speed limit for vehicles that intend to make a turn at the intersection. A vehicle should not make the turn at high velocities to avoid rollover. Since in Crossroad IM assigns a constant velocity to all vehicles to maintain (including those that intend to make a turn) and vehicles are supposed to keep the assigned velocity until entering the intersection, the assigned velocity for vehicles that intend to make a turn will be bounded by the turn speed limit. As a

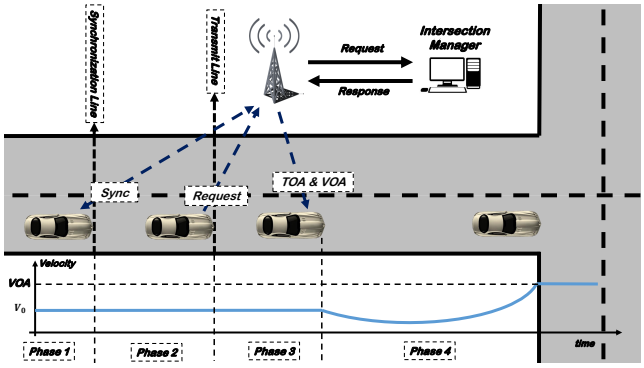

Fig. 4. Different phases of a vehicle in our technique (RIM). Phase 1) Synchronization, Phase 2) Sending a request, Phase 3) Receiving the response, Phase 4) Trajectory calculation and tracking

result, turning vehicles have to drive at a slow speed before reaching the intersection and this reduces the throughput of the intersection.

III. Robust Intersection Management (RIM)

In our approach, RIM, we divide the status of an approaching vehicle into four phases: 1) when the vehicle is within the range of the intersection and before reaching the synchronization line, 2) after the synchronization line and before the transmit line and 3) after sending the request and before receiving the response, 4) after receiving the response until entering the intersection. Figure 4 shows the status of a vehicle at different phases.

In phase 1, all vehicles synchronize their local clock by either communicating with the IM or receiving a reference clock from a GPS sensor (GPS satellites broadcast very accurate clocks). If the synchronization is successful, the vehicle enters phase 2 and sends its position (P), velocity (V), acceleration (a) and the corresponding timestamp (TS), as well as the outgoing lane (LO), max/min acceleration (a_{max} and a_{min}) and the ID to the IM upon crossing the transmit line. In phase 3, IM processes the request and calculates a feasible TOA and VOA, based on the status of the vehicle (V-Info) and the scheduling policy (FCFS, BATCH[20], etc.). Variety of scheduling policies are studied in the literature [**?**], [20], [22]. Since the effectiveness of the scheduling policy is not the main focus of this paper, we use an FCFS scheduling policy for simplicity. Then, IM sends them back to the requesting vehicle. In this phase, the vehicle maintains its initial velocity until it receives the response. In phase 4, the vehicle creates a reference trajectory and follows it until it enters the intersection. We consider the following model for the behavior of vehicles in 2D:

$$\begin{cases} \dot{x} = v cos(\phi) \\ \dot{y} = v sin(\phi) \\ \dot{\phi} = \frac{v}{L} tan(\psi) \\ \dot{v} = u(t) \\ u(t) = K_a\big(- K_p e - K_i \int e - K_d \dot{e} + d(t)\big) \end{cases} \quad (1)$$

where x, y are longitude and latitude of the vehicle in Cartesian coordinates respectively, ϕ is the heading angle of the vehicle from the x-axis, v is the linear velocity of the vehicle, L is vehicle's wheelbase distance, ψ is the steering angle of front tires and u is the control input for the motor. K_p, K_i and K_d are PID controller gains, e, $\int e$ and $\dot{e}$ are the error between actual velocity and target velocity, its integral and derivative respectively and $d(t)$ is the applied disturbance. K_a is a constant to model actuator's gain. The input for the motor is $u(t)$, which is generated as a Pulse Modulation Width (PWM) signal. We assume that the values of the PID controller and the actuator gain have model mismatches.

Algorithm 1: Vehicle Controller

```
1  if Sync line is crossed then
2      result = synchronize();
3      if result is not OK then
4          if distance to transmit line is less than d_min then
5              update(Trajectory, SD); /* slow down */
6          end
7          Goto Line 3;
8      end
9  end
10 if Transmit line is crossed then
11     V-Info = [P, V, a, TS, LO, a_max, a_min, ID];
12     send(V-Info);
13     Wait for the response;
14     if response is timed out then
15         if distance to intersection is less than d_min then
16             update(Trajectory, SD); /* slow down */
17         end
18         Goto line 12;
19     else
20         [TOA, VOA] = getPacket(response) ;
21         [A_0, B_0] = calculateTrajectory(TOA, VOA);
22         update(Trajectory, [A_0, B_0]); /* set the Ref
               Trajectory */
23     end
24 end
```

A. Vehicles

When the vehicle receives the TOA and VOA, it computes an optimal reference position trajectory and a PID controller is utilized to track the trajectory. Each vehicle has a specified timeout to bound its waiting time when waiting the response. Algorithm 1, shows a pseudocode of the vehicle's controller. The value of d_{min} is calculated based on a_{min} and v_{max}, i.e., the distance a vehicle needs for stopping. In order to compute the reference trajectory, each vehicle stores its current position, velocity, and the timestamp as initial position (x_0), velocity (v_0) and time (t_0). Additionally, final position (x_f), velocity (v_f) and TOA (t_f) of the reference trajectory are known (received from the IM). Any position trajectory that satisfies the initial and final position condition ($x(t_0) = x_0$ and $x(t_f) = x_f$) and its derivative (velocity trajectory) satisfies the initial and final velocity conditions ($v(t_0) = v_0$ and $v(t_f) = v_f$) can be a candidate for the reference trajectory. However, we are looking for an optimal trajectory for the vehicle. So,

Algorithm 2: IM's Scheduling algorithm

```
Input: Request;
Outputs: [TOA, VOA];
while Request buffer is not empty do
    V-Info = read(buffer[first]);
    [TOA, VOA] = Schedule(V-Info, I-Info);
    Result = F-Check(TOA, VOA, V-Info, I-Info);
    if Result is OK then
        Send(TOA,VOA,Vehicle Info);
        update(I-Info)
    else
        Increase(TOA);
        Goto Line 6;
    end
end
```

we define a functional J to minimize the acceleration of the trajectory:

$$J = \int_{t_0}^{t_f} a^2 dt \tag{2}$$

where a is the acceleration of a vehicle. After solving Equation (2) using the *Fundamental Lemma of the Calculus Variation* [23], the solution (acceleration trajectory) is found to be in the form of:

$$a(t) = A_0 t + B_0 \tag{3}$$

A_0 and B_0 are constant variables to be determined. Taking integral from (3), we have:

$$v(t) = \frac{1}{2}A_0 t^2 + B_0 t + v_0 \tag{4}$$

Taking integral from (4) results in a cubic function as:

$$x(t) = \frac{1}{6}A_0 t^3 + \frac{1}{2}B_0 t^2 + v_0 t + x_0 \tag{5}$$

Without loss of generality, we assume that the initial time t_0 for the reference trajectory is zero. By substituting t, $x(t)$ and $v(t)$ for boundary condition values, t_f, x_f and v_f in Equations (4) and (5), following equations are derived:

$$x_f = \frac{1}{6}A_0 t_f^3 + \frac{1}{2}B_0 t_f^2 + v_0 t_f + x_0 \tag{6}$$

and

$$v_f = \frac{1}{2}A_0 t_f^2 + B_0 t_f + v_0 \tag{7}$$

Solving Equations (6) and (7) for A_0 and B_0, yields:

$$\begin{cases} A_0 = \frac{6(2x_0 - 2x_f + t_f v_0 + t_f v_f)}{t_f^3} \\ B_0 = \frac{-2(3x_0 - 3x_f + 2t_f v_0 + t_f v_f)}{t_f^2} \end{cases} \tag{8}$$

Each vehicle computes the value of A_0 and B_0 and creates its reference trajectory according to Equation (5). If a vehicle receives the target TOA and VOA within the worst-case delay (due to the IM's computation time and network delay), it's still able to create a feasible trajectory that meets the final conditions (TOA and VOA).

1) Case study: To have a better understanding, we simulated position and velocity trajectories of a vehicle (Using Equation (1)) that is 15 m away from the intersection while driving at 3 m/s. The worst-case delay from IM to the vehicle is 1350 ms and the assigned TOA and VOA are 4 s and 2.5 m/s respectively. Dashed lines in Figure 5 show position and velocity trajectories for the best-case round-trip delay (BCRTD) and solid lines depict position and velocity trajectories for the worst-case round-trip delay (WCRTD) respectively. Delay in

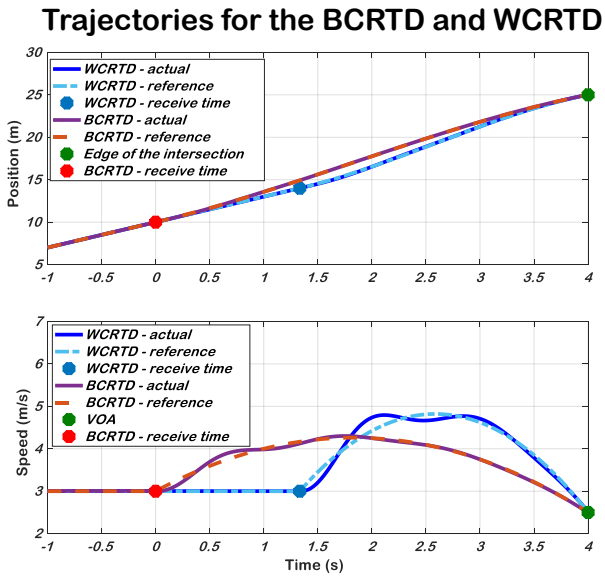

Fig. 5. Velocity and position trajectories for the best-case and worst-case round-trip delay (BCRTD and WCRTD) in the network.

the network or IM processing time may affect the trajectory of the vehicle. However, no matter how much is the delay, as long as it's smaller than the WCET plus WCND, the arrival time and velocity of the vehicle remains unaffected.

B. Intersection Manager

When IM receives a request, it computes a TOA and VOA based on the status of the requesting vehicle (V-Info) and the status of other vehicles that have already received a TOA and VOA (I-Info). Before sending back the computed TOA and VOA to the requesting vehicle, IM verifies the feasibility of the computed TOA and VOA using the "F-Check" function. Algorithm 2 shows the pseudo-code for the IM. In order to check the feasibility of assigned TOA and VOA, IM has to check the future trajectory of the vehicle and verify that road specifications ($V < V_{max}$), vehicle specifications ($a < a_{max}$) and safety specifications (No front-back accident before entering the intersection) are not violated. From Figure 5, one can observe that the area under the velocity profile is the same for both best-case and worst-case RTD. This is because the TOA and VOA are fixed. As a result, the vehicle will experience higher/lower velocities (a higher peak/a lower trough), as the receive time increases. Based on this observation, we can conclude that if the worst-case trajectory does not violate the maximum/minimum velocity threshold,

Algorithm 3: F-Check function

```
v = calculateVelocity ;     /* based on Eq.(4) */
a = calculateAcceleration; /* based on Eq.(3) */
inLane = getLane(V-Info);
if max(v) < v_max and min(v) < v_min then
    if max(a) < a_max then
        For all cars ∈ I-Info s.t. V-Info.inLane = inLane
        distance = distanceBetweenCar1andCar2;
        if min(distance) > distance Threshold then
            Result = OK;
        else
            Result = Not OK;
        end
    else
        Result = Not OK;
    end
else
    Result = not OK;
end
```

the best-case trajectory never exceeds such values. This way, we can check if requirements are being met only by verifying the worst-case trajectory.

C. Safety Analysis

IM needs to verify that the assigned TOA and VOA are safe. As a result, it performs a feasibility analysis for the best-case and worst-case scenarios. F-Check function in Algorithm (2) computes the values of A_0 and B_0 based on the WCND and WCET and, checks if the max value of the worst-case delay trajectory is smaller than road speed limit (V_{max}) and the min value is greater than a threshold $V_{min} > 0$. Additionally, F-Check verifies if the maximum acceleration of the worst-case trajectory is smaller than a_{max}. For different values of VOA and TOA, we simulated the position and velocity trajectories of a vehicle and depicted them in Figure 6 where green trajectories are feasible and red ones are infeasible. Algorithm 3 shows details of the F-Check function. Since the extreme acceleration/deceleration cases occur only at boundary conditions, IM can verify the feasibility of the worst-case reference trajectories by just checking the acceleration at the initial time. We simulated the behavior of a vehicle driving at 3 m/s for different pairs of VOA and TOA when the intersection is 15 meters away. Figure 6 depicts the position and velocity trajectories of the vehicle. If the velocity trajectory for the WCRTD scenario exceeds the speed limit or its slope exceeds the acceleration limit (a_max), the trajectory is not feasible and IM extends the TOA of the vehicle. However, if the velocity trajectory goes under the minimum velocity, it means that the vehicle should drive at a very slow velocity, which is not practical. Once the vehicle calculates the values of A_0 and B_0, it sends them to the IM in order to confirm that it has received the assigned TOA and VOA and, lets the IM know how the trajectory would be.

It's also possible that the trajectory of a vehicle conflicts with another vehicle in the same lane before reaching the

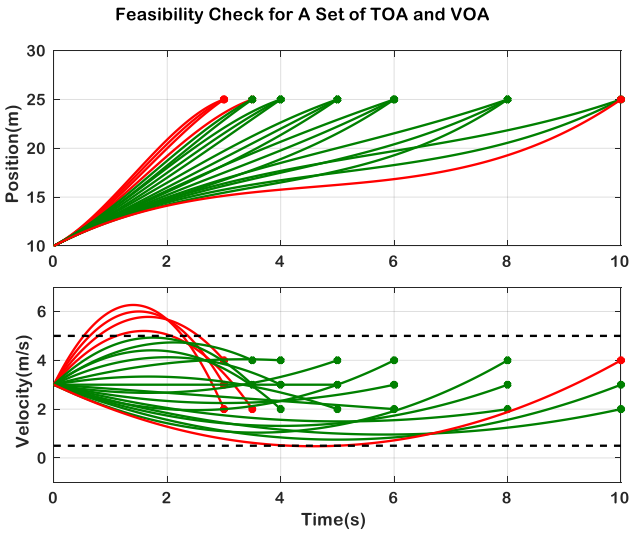

Fig. 6. An example of feasibility checking for a set of the VOA and TOA. Based on the specified maximum and minimum velocity thresholds, IM rejects a pair of TOA and VOA. Green points on the velocity Figure correspond to feasible TOAs and VOAs.

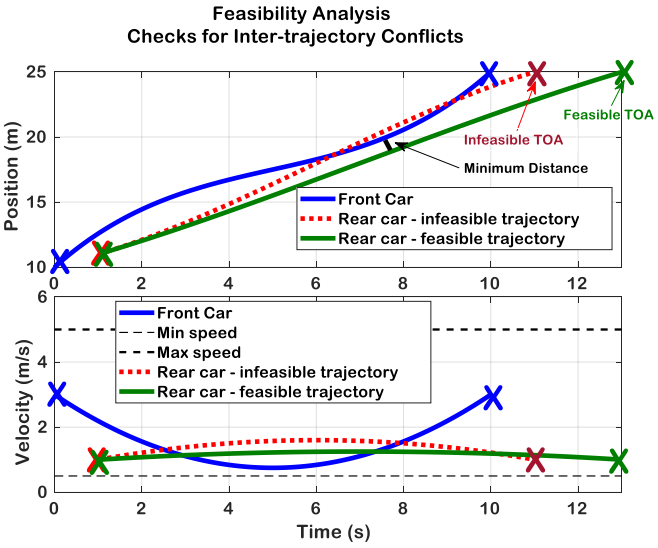

Fig. 7. A scenario where F-Check fails. The assigned TOA and VOA cause a front-back accident (the blue position trajectory crosses the red one). IM then assigns another TOA (green) that is safe.

intersection. We simulated a case where two vehicles driving in the same lane have a conflict on their position trajectory and depicted their trajectories in Figure 7. Blue trajectories belong to the front vehicle and the red and green ones belong to the rear car. Red trajectories are not feasible while the green ones are feasible. IM can find a feasible trajectory for the rear vehicle by increasing the TOA. If the distance between trajectories of two vehicles in the same lane is always greater than a threshold, the value of the result is "OK". Otherwise, the result will be "not OK" and the IM has to increase the TOA and verify the TOA and VOA using the F-Check function again.

D. Practical issues

Since vehicles and IM interact with each other, both should follow some rule as a prerequisite to the correct functionality of the system. For instance, the system will not work if the processing time of the IM is very high or if a vehicle takes a trajectory that fails to satisfy the assigned TOA and VOA. Therefore, we discuss some of the necessary requirements that should be met. It is challenging to find an upper bound for the network request because the delay in the network can be infinite. To address this issue, vehicles use a timeout mechanism to bound the waiting time of a vehicle. This ensures that a vehicle either receives the response within the expected delay or it will ignore the response if it's received afterward. The value of the timeout can be determined by measuring the average delay of the network and WCET of the IM. WCET can be calculated statically using existing WCET analysis methods [11], [24], [25]. Similarly, if a vehicle fails to synchronize its clock with the IM or cannot get it from the GPS before reaching the transmit line, it should slow down and stop behind the intersection line.

As another requirement, vehicles must always retain a safe distance from their front vehicle. Typically, the Adaptive Cruise Control (ACC) system is responsible to maintain a safe distance from the front vehicle by adjusting the velocity. Based on the Responsibility-Sensitive Safety (RSS) model [26], maintaining a minimum distance from the front vehicle requires having a bounded response time (from sensing to actuation). In order to guarantee the safety of the intersection, we can express a set of requirements for vehicles and IM. One way to formally express such safety requirements for each processing unit is specifying them using temporal logic (like Timestamp Temporal Logic (TTL) [27], [28]). Here's the list of requirement:

- WCET of the IM when responding to a request should be less than a threshold, say t_{IM}.
- Settling time of the PID controller should be short enough (Settling time is referred to the time it takes for the vehicle to reach and maintain the assigned trajectory).
- The network delay should be less than a threshold, t_N.
- The response time of the ACC system should be less than a threshold to avoid accidents before reaching the transmit line and after exiting the intersection, t_{ACC}.

Thresholds are determined based on specification of the intersection (intersection size, the distance of transmit line from the intersection, turn speed limit, wireless network, etc.), IM (WCET), network (WCND) and vehicles (size, max/min acceleration rate, etc.).

IV. Experimental Testbed

In order to evaluate our technique, we built a $1/10$ scale model 4-way intersection (Figure 8) with 8 fully autonomous RC vehicles that communicate with a stationary IM. The width of each lane is 60 cm and the transmit line is located 3 m away from the edge of the intersection. All autonomous vehicles are built on Traxxas Slash chassis. The size of each vehicle is 30 $cm \times 57$ cm and can drive up to 5 m/s. Wheel-base size of each vehicle is 53.5 cm and its maximum steering angle is 45 degrees. Transmit line is located 3 meters away from the vehicle's initial position and the edge of the intersection is 6 meters away from the starting point of the vehicle.

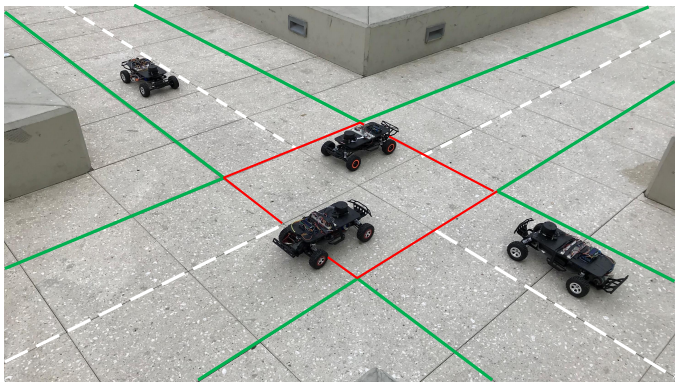

Fig. 8. An overview of our 1/10 scale model intersection of autonomous vehicles. Intersection and road lines are overlayed for a better intuition.

The main microcontroller is an Arduino Mega 2560 which performs trajectory tracking. We utilized a Bosch BNO055 absolute orientation sensor for measuring the heading angle of the vehicle and making a turn. Each vehicle communicates to the IM via an NRF24L01+, 2.4GHz wireless module. We used a hall effect shaft encoder to measure the longitudinal position of the vehicle. Encoder data are processed by another microcontroller (Arduino Nano board) and the position data are sent to the main microcontroller over an I2C communication. We implemented a Proportional Integral Derivative (PID) controller for each vehicle. We measured the maximum acceleration/deceleration of each vehicle numerically by performing some tests. IM station includes an Arduino Mega 2560 and an NRF24L01+, 2.4GHz wireless module for communication. We used Network Time Protocol (NTP) [29] synchronization technique and the accuracy of synchronization is 10 ms. Synchronization packet has a size of 7 bytes (1 byte for message type, 4 bytes for timestamps and 2 bytes for ID). The size of a request packet is 30 bytes, which includes ID, message type, velocity, position, captured timestamp, lane out, max acceleration, max deceleration, and max speed. The

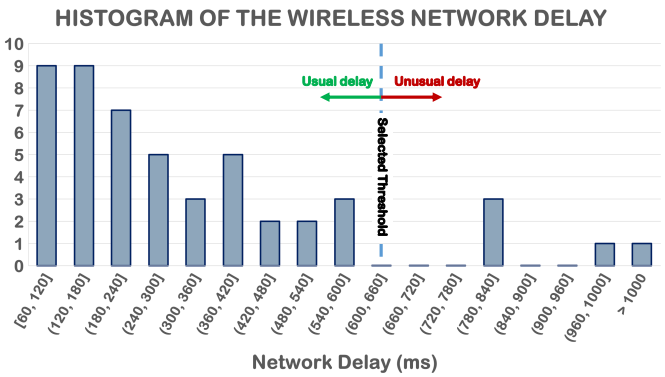

Fig. 9. Histogram of the measured network delay in our 1/10 scale model testbed. Based on the collected data, the selected threshold value for a communication with a usual delay is set to be 600 ms.

response packet has a size of 16 bytes, which includes ID,

message type, TOA, VOA and transmit line distance (the distance of transmit line from the edge of the intersection). The acknowledgement packet is 8 bytes and contains A_0 and B_0. For the experiment, vehicles are placed at arbitrary positions and start driving with arbitrary initial velocities. Before reaching the transmit line, vehicles synchronize their local clock with the IM by sending a sync packet. Each vehicle monitors its position and upon crossing the synchronization line or transmit line, it sends a synchronization message or a request to the IM respectively. To estimate the worst-case delay for the IM, we need to find a reasonable value for communication delay and estimate the WCET of the IM. Figure 9 shows the histogram of the measured delay for the wireless network in 50 experiments. Based on the collected data, we set the network threshold to be 600 ms. As a result, the value of timeout for each vehicle (discussed in Section III) can be calculated as:

$$t_{Timeout} = WCET + 2WCND$$

The WCET of the IM is estimated based on the maximum capacity of the intersection, which is related to the maximum number of vehicles that fit in the intersection and roads before it. The estimated WCET of the IM for the microcontroller (Atmega2560 with the clock frequency of 16 MHz) is 56 ms. As a result, we set the timeout to be 1256 ms. Since vehicles ignore a response after the timeout, we can claim that the WCRTD is 1256 ms.

V. Experimental Results

We conducted two types of experiments: i) safety-related and ii) throughput-based experiments. The first one highlights the effectiveness of the RIM technique in reducing the position error and the second one shows the usefulness of the RIM in improving the throughput of the intersection. In safety experiments, we evaluated the impact of external disturbances and model mismatch on the eventual position of the vehicle in 3 different experiments:

- **Effect of External Disturbances (ED) on TOA**

To model the external disturbance, we added a step function with the amplitude of up to 5% of the maximum range to the PWM signal (generated by the controller for the motor) and measured the position error at the expected TOA for Crossroads approach and RIM. Figure 10 depicts the position and velocity trajectories of a vehicle under RIM interface in presence of an external disturbance with the amplitude of 10 % of the max value. Despite the fact that the velocity trajectory of the vehicle is deviated by the external disturbance, it is still able to meet the set TOA and VOA.

- **Effect of Model Mismatches (MM) on TOA**

In Crossroads, IM has to account for the response time of vehicles when computing the target velocity and actuation time. However, the response time calculation is done based on the considered model and can be inaccurate. To see how much model mismatches can affect the TOA, we added up to 10% error to parameters of the PID controller (K_P, K_I, and K_D), which is related to the estimated actuation time by the

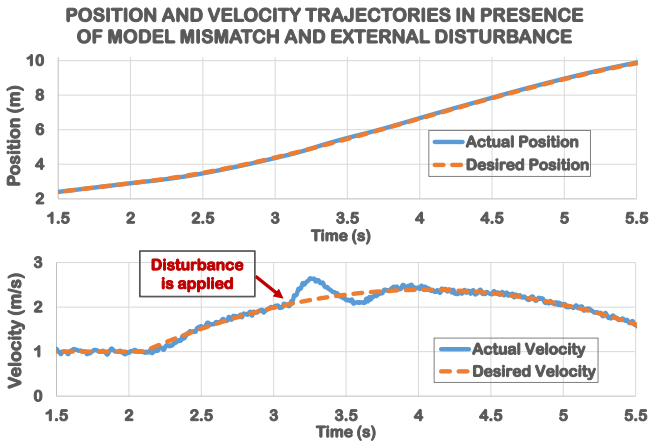

Fig. 10. An external disturbance is applied to the vehicle that causes a temporary degradation in the velocity. However, the vehicle is able to compensate for the effect of the disturbance and meet the assigned TOA and VOA.

IM. We measured the position error at the expected TOA for both Crossroads and RIM techniques and reported the result in Figure 3 and Figure 10.

- **Effect of combined MM and ED on TOA**

In this experiment, we modeled both the external disturbance and model mismatch similar to the first and second experiments and recorded the measured position error at the expected TOA. Then, we compared the result for the Crossroads approach and RIM technique. We repeated each experiment 50 times for a different set of initial velocities and positions and, the position error is reported by storing the position of vehicles along with a timestamp on the EEPROM memory of their microcontroller. Figure 11 shows the average and the worst-case position error of vehicles at the expected TOA for Crossroads and RIM, normalized to the size of the vehicle. Results

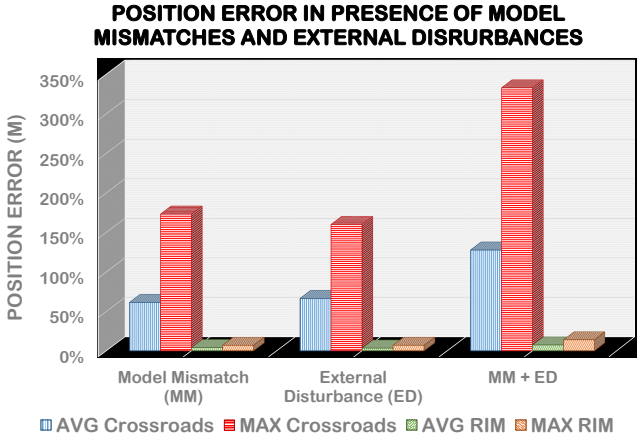

Fig. 11. The average and worst-case position error of vehicles at designated TOA in presence of i) Model Mismatches (MM), ii) External Disturbances (ED) and iii) MM and ED together.

from Figure 11 indicate that on average, RIM can reduce the position error by 18X compared to the Crossroads technique. Since Crossroads and generally all VA-IM techniques ignore the effect of model mismatch and external disturbances, they

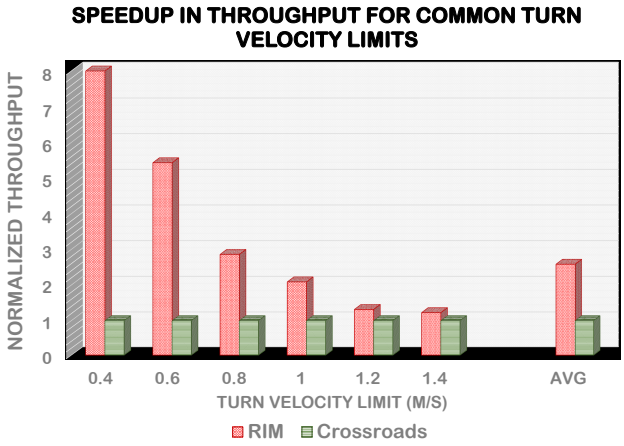

Fig. 12. Rim achieves Speedups in throughput of he intersection by controlling the speed before entering the intersection. The turn velocity limit varies between 0.4 to 1.4 m/s (9 mph to 31 mph for a real-sized intersection)

are not safe and accidents can happen. In order to safely manage by just vehicles using a constant velocity, IM should consider a larger safety buffer around all vehicles to avoid accidents. Results from our experiment show that the size of the extra safety buffer can be as large as 3.2X of the vehicle length in the worst-case (MM and ED together). Considering such a large buffer around each vehicle guarantees the safety of the vehicle but is impractical since it reduces the throughput of the intersection greatly.

- **Velocity management for vehicles making a turn**

In intersections with a separate road for a right turn, the turn speed limit can be as high as 31 mph. However, for small intersections vehicles have to make a sharp right turn and therefore, the turn speed limit is as low as 9 mph. In this experiment, we measured the wait time of all vehicles, from transmit line to the departure of the intersection, by storing entrance and departure timestamps on the EEPROM memory of the vehicle's microcontroller. The maximum allowed velocity for making a turn in our 1/10 scale model varies from 0.4 m/s to 1.4 m/s (9 mph to 31 mph for a real intersection [30]) and, the speed limit (for the road) is 2.5 m/s (55 mph). Figure 12 shows the throughput of RIM and Crossroads normalized to the throughput of the Crossroads.

Results show that RIM can achieve 2.7X better throughputs on average in comparison with Crossroads and other VA-IM techniques and, 8X in the best-case (lowest turn speed limit). The great difference in the throughput at low turn speeds has two main reasons: i) the scheduling policy of the IM and ii) induced behavior from the front vehicle. Since the scheduling policy is FCFS, a vehicle that tends to go straight will be slowed down if it is behind another vehicle that is making a turn at the intersection. For other scheduling policies like BATCH[20], the difference can be lower. Since setting arbitrary input flow rates in real experiments is hard, we will study the effect of considering the extra safety buffer on the throughput of the intersection using our simulator.

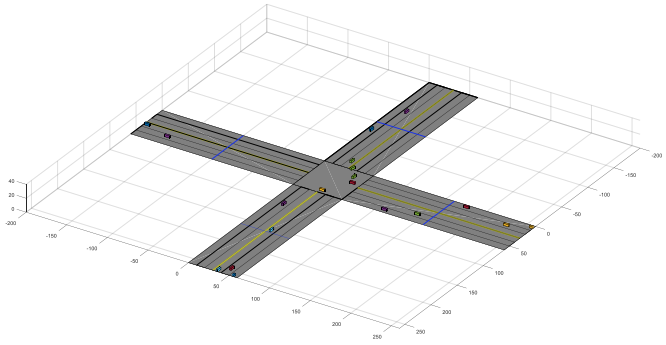

Fig. 13. A view of our 3D simulator in MATLAB®. The blue line on each road shows the location of the transmit line.

A. Extension to Multi-lane Intersections

In order to show that our method can be easily be scaled to multi-lane intersections, we built a 3D simulator in MATLAB®. The simulator considers a separate processing unit for vehicles and the IM and all data exchanging is done through the communication over a network. The network has the capability of modeling a random network delay and packet loss. Figure 13 shows a view of our simulator. In our simulator[1], we created a four-way intersection with 3 lanes per road. Intersection size is 60 x 60 m and lane width is 10 m. The size of simulated vehicles is 6 x 2 m with the wheelbase of 5 m. The maximum value of acceleration is 5 m/s^2 and deceleration is -8 m/s^2. We used the result of the experiment

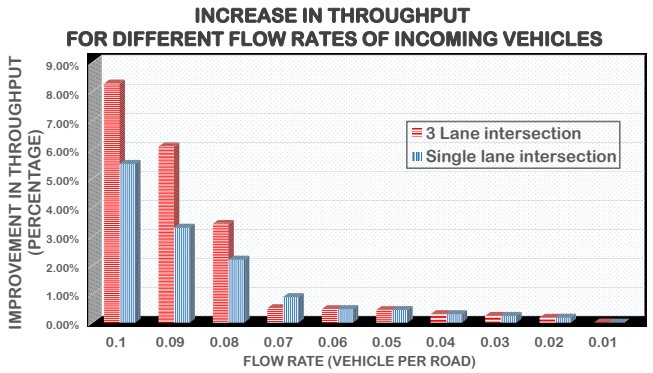

Fig. 14. Increase in the throughput of the intersection for different values of input flow rate of incoming vehicles. RIM can achieve improvement in the throughput since the extra safety buffer for model mismatch and external disturbance is skipped.

on our 1/10 scale model autonomous vehicle to estimate the size of the extra safety buffer for the Crossroads technique and VA-IM approaches. Since the length of the vehicle is 6 m and the error due to model mismatch and possible external disturbances can be as large as 3.3X of the length of the vehicle, the extra safety buffer size is calculated as 20m (10 m in front of the vehicle and 10 m behind it). The transmit line is 200 m away from the intersection and the sync line is 250 m away from the intersection. We implemented an FCFS policy for the IM and requests are processed based on their arrival

[1]Available Online: https://github.com/mkhayatian/Traffic-Intersection-Simulator-for-Autonomous-Vehicles

time. Figure 14 shows the degradation of the throughput in a single lane intersection and in a multi-lane intersection (3 lanes per road) due to considering an extra safety buffer around vehicles. Results from Figure 14 show that we can improve the throughput of the intersection by up to 8% for a multi-lane intersection and up to 5% for a single lane intersection when there is no need for considering an extra safety buffer for model mismatches and external disturbances. In order to fairly compare the throughput of the Crossroads technique and other VA-IM techniques against RIM, we should add the improvement result from both Figure 12 and Figure 14. This is because RIM can increase the throughput by managing the speed of vehicles making a turn at the intersection and avoid considering an extra safety buffer.

VI. Conclusion

In this paper, we investigated the safety and performance challenges of implementing an intersession of autonomous vehicles. We proposed a time and space aware technique, that is robust against model mismatches, external disturbances, and nondeterministic delay of network and processing time of the IM. By efficiently managing vehicle, our management protocol, RIM, can reduce the effect of uncertainties and greatly improve the throughput of the intersection. We built a 1/10 scale model intersection of autonomous vehicles to compare RIM with the existing ones. Results from our experiments show that on average, RIM is able to reduce the position error of vehicles due to model mismatch and external disturbances by 18X on average for up to 10% error in the parameters of the model and an external disturbance with 0.1X amplitude. In addition, RIM is able to increase the throughput of the intersection by managing the velocity of turning vehicles before entering the intersection by 2.7X on average.

Acknowledgment

This work was partially supported by funding from NIST Award 70NANB16H305, and by NSF grant CNS 1525855, and CPS 1645578.

References

[1] US Department of Transportation - Federal Highway Administration, "Roadway Safety Data Dashboards," Online: accessed [5/15/2017].

[2] A. Morales Medina *et al.*, "Cooperative Intersection Control based on Virtual Platooning," 2017.

[3] F. Perronnet, A. Abbas-Turki, and A. El-Moudni, "Cooperative Intersection Management: Using Mini-Robots to Compare Sequenced-based Protocols," *Les Journées Nationales des Communications dans les Transports (JNCT)*, 2013.

[4] A. A. Malikopoulos *et al.*, "A Decentralized Energy-Optimal Control Framework for Connected Automated Vehicles at Signal-Free Intersections," *arXiv preprint arXiv:1602.03786*, 2016.

[5] J. Lee and B. Park, "Development and Evaluation of A Cooperative Vehicle Intersection Control Algorithm under The Connected Vehicles Environment," *IEEE ITS Transactions*, 2012.

[6] K. Dresner and P. Stone, "A Multiagent Approach to Autonomous Intersection Management," *Journal of artificial intelligence research*, vol. 31, pp. 591–656, 2008.

[7] I. H. Zohdy and H. A. Rakha, "Intersection Management via Vehicle Connectivity: The Intersection Cooperative Adaptive Cruise Control System Concept," *Journal of Intelligent Transportation Systems*, vol. 20, no. 1, pp. 17–32, 2016.

[8] E. Andert, M. Khayatian, and A. Shrivastava, "Crossroads: Time-Sensitive Autonomous Intersection Management Technique," in *Proceedings of the 54th Annual Design Automation Conference 2017*, p. 50, ACM, 2017.

[9] A. Shrivastava, P. Derler, Y.-S. L. Baboud, K. Stanton, M. Khayatian, H. A. Andrade, M. Weiss, J. Eidson, and S. Chandhoke, "Time in Cyber Physical Systems," in *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2016 International Conference on*, pp. 1–10, IEEE, 2016.

[10] A. Shrivastava *et al.*, "A Testbed to Verify the Timing Behavior of Cyber-Physical Systems," in *Proceedings of the 54th Annual Design Automation Conference 2017*, p. 69, ACM, 2017.

[11] J. Gustafsson, A. Ermedahl, C. Sandberg, and B. Lisper, "Automatic Derivation of Loop Bounds and Infeasible Paths for WCET Analysis using Abstract Execution," in *Real-Time Systems Symposium, 2006. RTSS'06. 27th IEEE International*, pp. 57–66, IEEE, 2006.

[12] L. Li and F. Wang, "Cooperative Driving at Blind Crossings using Inter-vehicle Communication," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 6, pp. 1712–1724, 2006.

[13] L. Chen and C. Englund, "Cooperative Intersection Management: A Survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 570–586, 2016.

[14] V. Milanés, J. Pérez, E. Onieva, and C. González, "Controller for Urban Intersections Based on Wireless Communications and Fuzzy Logic," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 1, pp. 243–248, 2010.

[15] D. Carlino, S. D. Boyles, and P. Stone, "Auction-Based Autonomous Intersection Management," in *ITSC, 16th International IEEE Conference on*, pp. 529–534, IEEE, 2013.

[16] C. Fok, M. Hanna, S. Gee, T. Au, P. Stone, C. Julien, and S. Vishwanath, "A Platform for Evaluating Autonomous Intersection Management Policies," in *Proceedings of IEEE/ACM 3rd International Conference on Cyber-Physical Systems*, IEEE Computer Society, 2012.

[17] M. Quinlan, T. Au, J. Zhu, N. Stiurca, and P. Stone, "Bringing Simulation to Life: A Mixed Reality Autonomous Intersection," in *Intelligent Robots and Systems (IROS)*, pp. 6083–6088, IEEE, 2010.

[18] K. M. Dresner and P. Stone, "Sharing the Road: Autonomous Vehicles Meet Human Drivers," in *International Joint Conference on Artificial Intelligence*, vol. 7, pp. 1263–1268, 2007.

[19] Q. Jin, G. Wu, K. Boriboonsomsin, and M. Barth, "Advanced Intersection Management for Connected Vehicles using a Multi-agent Systems Approach," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pp. 932–937, IEEE, 2012.

[20] R. Tachet, P. Santi, S. Sobolevsky, L. I. Reyes-Castro, E. Frazzoli, D. Helbing, and C. Ratti, "Revisiting Street Intersections using Slot-based Systems," *PloS one*, vol. 11, no. 3, 2016.

[21] H. Ahn and D. Del Vecchio, "Safety verification and control for collision avoidance at road intersections," *IEEE Transactions on Automatic Control*, vol. 63, no. 3, pp. 630–642, 2018.

[22] S. A. Fayazi, A. Vahidi, and A. Luckow, "Optimal Scheduling of Autonomous Vehicle Arrivals at Intelligent Intersections via MILP," in *American Control Conference (ACC), 2017*, pp. 4920–4925, IEEE, 2017.

[23] I. M. Gelfand, R. A. Silverman, *et al.*, *Calculus of Variations*. Courier Corporation, 2000.

[24] D. Hardy and I. Puaut, "WCET Analysis of Multi-level non-inclusive Set-associative Instruction Caches," in *Real-Time Systems Symposium, 2008*, pp. 456–466, IEEE, 2008.

[25] S. Chattopadhyay and A. Roychoudhury, "Unified Cache Modeling for WCET Analysis and Layout Optimizations," in *Real-Time Systems Symposium, 2009, RTSS 2009. 30th IEEE*, pp. 47–56, IEEE, 2009.

[26] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a Formal Model of Safe and Scalable Self-driving Cars," *arXiv preprint arXiv:1708.06374*, 2017.

[27] M. Mehrabian *et al.*, "Timestamp Temporal Logic (TTL) for Testing the Timing of Cyber-Physical Systems," in *Proceedings of the Annual Embedded System WEEK (ESWEEK)*, ACM, 2017.

[28] M. Mehrabian *et al.*, "An Efficient Timestamp-based Monitoring Approach to Test Timing Constraints of Cyber-physical Systems," in *Proceedings of the 55th Annual Design Automation Conference*, p. 144, ACM, 2018.

[29] D. L. Mills, "Computer Network Time Synchronization," in *Report Dagstuhl Seminar on Time Services Schloß Dagstuhl, March*, vol. 11, p. 332, Springer, 1997.

[30] K. Fitzpatrick, W. Schneider, and H. William, "Turn Speeds and Crashes within Right-turn Lanes," tech. rep., Texas Transportation Institute, Texas A & M University System, 2005.