Abstract—We propose an urban traffic management scheme for an all connected vehicle environment. If all the vehicles are autonomous, for example in smart city projects or future’s dense city centers, then such an environment does not need a physical traffic signal. Instead, an intersection control server processes data streams from approaching vehicles, periodically solves an optimization problem, and assigns to each vehicle an optimal arrival time that ensures safety while significantly reducing number of stops and intersection delays. The scheduling problem is formulated as a Mixed-Integer Linear Program (MILP), and is solved by IBM CPLEX optimization package. The optimization outputs (scheduled access/arrival times) are sent to all approaching vehicles. The autonomous vehicles adjust their speed accordingly by a proposed trajectory planning algorithm with the aim of accessing the intersection at their scheduled times. A customized traffic microsimulation environment is developed to determine the potentials of the proposed solution in comparison to two baseline scenarios. In addition, the proposed MILP-based intersection control scheme is modified and simulated for a mixed traffic consisting of autonomous and human-controlled vehicles, all connected through a wireless communication to the intersection controller of a signalized intersection.

I. INTRODUCTION

While traffic signals ensure safety of conflicting movements at intersections, they also cause much delay, wasted fuel, and tailpipe emissions. Frequent stops and goes induced by a series of traffic lights often frustrates passengers. However, recent studies have shown that vehicle to signal connectivity can improve this situation. Several publications have focused on uni-directional traffic signal to vehicle communication for guiding connected vehicles to arrive at green which increases their energy efficiency [1–4]. Another research direction has focused on improving intersection flow by optimizing timing of traditional traffic signals informed by uni-directional communication from connected vehicles [5],[6]. One can expect higher energy efficiency and intersection flow with bi-directional vehicle-signal communication where signals adjust their timings and vehicles their speeds [7]. Autonomous cars can further benefit from traffic signal information because they not only process the incoming information rather effortlessly but also can precisely control their speed and arrival time at a green light. The situation can get even better with 100% penetration of autonomous vehicles since a physical traffic light is not needed anymore as shown in concept papers [8–14]. Also because autonomous cars have much faster reaction times than human driven cars, the intersection controller can rapidly switch between phases [15].

Some of the benefits of eliminating traffic signals in an all autonomous vehicle environment is discussed in [9] and demonstrated by interesting simulation results in a recent publication [11]. However, optimal scheduling of vehicle arrivals at such intersections remains an open problem. Our paper attempts to address the gap in problem formulation by formalizing this scheduling problem as a Mixed Integer Linear Program (MILP) and shows its benefits in microsimulations. The proposed MILP-based controller receives information such as location and speed from each subscribing vehicle and suggests optimal access times to individual vehicles. The vehicles adjust their speed by a trajectory planning algorithm that each execute locally in order to reach the intersection at their assigned access times. The access times are computed periodically on a central server by solving a MILP. The objective of the optimization is smoothing the traffic flow and minimizing the intersection delay, while ensuring intersection safety and considering each vehicle’s desired velocity.

Preliminary results of our MILP-based intersection control were presented in [16] and this journal paper expands on those preliminary results with improved and more detailed formulation. This paper also provides a modified version of our MILP-based intersection control scheme with a physical traffic light to be applied to a mix of autonomous and human-controlled connected vehicles.

This paper is organized as follows: after a broad literature review in Section II, we introduce the scheduling problem in simple words in Section III. The notations used in this paper are explained in Section IV. Problem formulation is described in Section V followed by conversion to a MILP in Section VI. In Section VII, the nature of the MILP solution is shown in a simplified case study problem. In Section VIII, our proposed trajectory-planning algorithm to guide the autonomous vehicles for a timely arrival at intersection is explained. Microsimulation testbeds, computation load, and results are described in Sections IX, X, and XI, respectively. The modified intersection control scheme for mixed traffic conditions is provided in Section XII, followed by conclusions.
The coordination and optimal timing of traffic signals are by nature complex problems and backed by years of research in traffic engineering and operations research. In recent years, there has been a great deal of attention paid to intersection control for connected vehicles. Towards the most related work, Raravi et al. [17] determined the merge sequence in which vehicles cross the intersection region by formulating an optimization problem with constraints to ensure safety. The formulation proposed in [17] is nonlinear, and, as a result, Matlab optimization toolbox fmincon is used. The fmincon tool may only give local solutions [18] and does not guarantee a global optimum [17]. Lee et al. [19] also employed optimal control for a cooperative vehicle intersection control. In this work, the trajectories of any two conflicting vehicles are modified to minimize the overlap of trajectories in the intersection area. This would not always provide a feasible solution because of the complexity of the optimization formulation (the objective function and constraints are nonlinear) [15], [20]. For this reason, in [19], a combination of active-set, interior point, and genetic algorithms is used as backup which adds to the execution time of their intersection control.

Some other works, such as [21] and [22], used job-scheduling techniques. Colombo et al. [21] view the time interval that each vehicle spends in an intersection as the length of the job to be executed. Similarly, Xie et al. [22] view clusters in the aggregate flow representation of different routes as the jobs to be scheduled; where clusters are a basic representation of a vehicle or group of vehicles [23]. They use an approximate dynamic programing procedure, called Controlled Optimization of Phases [24], to obtain a near optimal solution. Also Ahn et al. [25] translated the intersection collision avoidance to a job-shop scheduling problem assuming first-order dynamics for the vehicles. In the field of multi-agent systems, the solution provided by Dresner et al. in [8] and [9] is based on a reservation paradigm which allows the vehicles to reserve a block in space-time in an intersection. The solution is not optimal in the sense that it is a First Come, First Serve approach, and a reservation is rejected if any part of the requested space-time block has been previously reserved or occupied by another vehicle. A detailed review of cooperative intersection management systems can be found in [26].

This paper proposes an optimization-based approach for intersection traffic management. The challenge is to provide vehicles with travel recommendations that ensure energy efficiency, safety (collision-free passage through intersection), and smoother traffic flow (less intersection delay and number of stops). Incorporating all these goals into the intersection control algorithm will complicate the formulation. However, by proposing an imaginary access area around the intersection, we managed to have a formulation based only on time of arrivals to that area. Then we converted our vehicle arrival scheduling problem to a mixed-integer linear program (MILP). The optimization problem is solved using IBM CPLEX solver and the corresponding outputs (scheduled access/arrival times) are sent to all approaching vehicles in our customized microsimulation environment. A trajectory-planning algorithm is also presented that guides individual vehicles for a timely arrival at access area.

Mixed-integer linear program (MILP) has been used in various path planning applications with collision avoidance, such as airplanes and autonomous vehicles. In the field of vehicle-intersection coordination, Zhu et al. [20] used a lane-based traffic flow model to optimize the total travel time. The output of their optimization is the traffic flow from one lane to another through the conflict point model without considering the travel times and velocities that are desirable to individual vehicles. The vehicles’ capability to meet the scheduled time of arrival is not specifically addressed by the authors in [20] and no microsimulations are provided. Our paper, however, provides microsimulations considering the interaction between vehicles and ensures that the vehicles can meet their scheduled arrival times by a trajectory-planning algorithm.

Also in the area of phase and timing optimization for standard two-phase or eight-phase intersection controllers, a set of MILP formulations have been proposed in the literature. Most of these formulations can be applied to mixed traffic consisting of autonomous and human-controlled vehicles. They either use off-line historical traffic data or assume a one-way communication where connected vehicles report information to an intersection controller but their travel trajectory cannot be controlled. For instance, He et al. [5] used probe vehicles’ on-line information to identify pseudo-platoons and found an optimal signal plan using MILP. Little’s MILP formulation [27], and a recent work in [7], solves the bandwidth maximization problem, and assigns optimal offsets to the standard traffic signals in a two-way arterial. Other works in this area use model predictive control for traffic signal control problem and formulate it as a MILP problem [6], [28], [29].

III. Problem Statement

We seek an intersection controller that coordinates and harmonizes the flow of the approaching connected vehicles.

The controller resides on a computational server and receives information of all subscribing vehicles and then schedules the intersection access time for each vehicle regularly. The scheduled access times (arrival times) are sent to all subscribing vehicles so that they can adjust their speed accordingly. The challenge is to find appropriate access times that ensure safety, passenger comfort, and smoother traffic flow.

We ignored all the left and right turns to simplify the presentation of ideas. We assume a two-phase/four-movement intersection. As shown in Figure 1(a), Phase $X' (\phi_X = \{X',X''\})$ corresponds to a set of two traffic movements: (1) south-bound denoted by dark letter X or $X'$; (2) north-bound denoted by light letter X or $X''$. Similarly, Phase $O (\phi_O = \{O',O''\})$ corresponds to a set of two traffic movements: (1) west-bound denoted by dark letter O or $O'$; (2) east-bound denoted by light letter O or $O''$.

Figure 1(b) demonstrates a sample scheduling problem for 9 vehicles. The remaining distances from the current position of these vehicles to the intersection access point (stop position) are indicated on the y-axis of Figure 2. Note that the y-axis values are the remaining distances on four different movements.
which are projected on one single axis. As shown on this y-axis and also in Figure 1(b), five connected vehicles denoted by $cv_2$, $cv_4$, $cv_5$, $cv_6$, and $cv_8$ are traveling on Phase X and four vehicles $cv_1$, $cv_3$, $cv_7$, and $cv_9$ are traveling on Phase O. An intersection controller is sought that is capable of scheduling the vehicle arrivals as shown on the horizontal axis of Figure 2, as an example. It is assumed that the vehicles follow the travel trajectories shown by dashed lines in Figure 2 in order to make their scheduled arrival times. The provided sample solutions for both the scheduled arrival times and the travel trajectories were plotted schematically. The solution assigns virtual green splits to Phase X and O (see green/red timings in Figure 2) in such a way that intersection stops are avoided if at all possible. The solution assumes that:

- $cv_2$ can slow down to pass the intersection with the southbound platoon on Phase X.
- $cv_3$ can speed-up so the southbound platoon on Phase X does not slow down or stop at red.
- $cv_8$ can speed-up to catch up with Phase X green light.
- $cv_7$ and $cv_9$ adjust their speed to build a westbound platoon on Phase O.

**IV. DEFINITIONS AND NOTATIONS**

Achieving an optimal solution such as the example shown in Figure 2, requires to first formalize the problem objective as well as the constraints on vehicle movements. Before providing the proposed formulation, we first introduce the notation and their definitions in this section. Table I and Table II summarize the notations used in this paper.

**Connected Vehicles.** For each intersection, we will assume a subscription process by which the approaching connected vehicles send subscription requests to the intersection control server and announce their presence, phase/movement at the intersection, and intended time of arrival. We set the subscription distance to 500 m from the intersection center. An unsubscribe message is later sent from each individual vehicle to the intersection controller server at the time the vehicle clears the intersection. Thus, data is exchanged only during the subscription period, and the intersection occupancy can be tracked by the intersection controller. We represent the list of all subscribed connected vehicles as $CV = \{cv_i\}$ where $n$ is the size of $CV$. The list of connected vehicles is sorted by distance to the intersection where $cv_1$ is the closest vehicle to the intersection at the time of subscription. The length of the vehicle is denoted by $L$, and is taken to be 5.0 meter.

**Intersection.** It is assumed that the intersection is a square with width $W=10$ m. As shown in Figure 1(a), we consider a two-phase intersection consisting of Phase X and Phase O as $\phi = \{\phi_X, \phi_O\}$. Each phase includes a set of non-conflicting movements and $M = \{O', O', X', X''\}$ is the set of movements used in this paper. (see Table I). It is assumed that all intersecting roads have the same speed limit denoted by $v_{max}$.

**Time Instances.** For each vehicle approaching an intersection, we are interested in the following time instances: (1) time when the front of the vehicle enters the intersection area at the stop-bar; (2) time when the rear of the vehicle exits the intersection area; (3) time when the front of the vehicle reaches an access distance from the intersection. As shown in Figure 3, these time instances are denoted by $t_{enter}$, $t_{exit}$, and

### Table I. Notations used to express intersection attributes.

<table>
<thead>
<tr>
<th>Description</th>
<th>$W$</th>
<th>$d_{access}$</th>
<th>$O'$</th>
<th>$O''$</th>
<th>$X'$</th>
<th>$X''$</th>
<th>$M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>width of the intersection (10 m)</td>
<td>access area distance from intersection</td>
<td>the west bound movement</td>
<td>the east bound movement</td>
<td>the south bound movement</td>
<td>the north bound movement</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table II. Notations used to express intersection attributes.

<table>
<thead>
<tr>
<th>$M$</th>
<th>$\phi_X$</th>
<th>$\phi_O$</th>
<th>$\phi$</th>
<th>$CV$</th>
<th>$n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>set of the movements in this paper</td>
<td>Phase X, the combination of non-conflicting concurrent movements</td>
<td>Phase O, the combination of non-conflicting concurrent movements</td>
<td>the combination of all phases</td>
<td>the list of all the vehicles subscribed to the intersection sorted by distance to the intersection</td>
<td>the number of all the vehicles subscribed to the intersection</td>
</tr>
</tbody>
</table>

**Fig. 1:** (a) Simplified two-phase/four-movement intersection. (b) An example of 9 vehicles approaching an intersection.

**Fig. 2:** An example of scheduling vehicle arrivals at an intersection (case study of Figure 1(b)). The depicted green/red timing is applicable to all-autonomous intersections with no physical traffic light only.

**Fig. 3:** Phase X Timing and Phase O Timing.
taccess, respectively. In this figure, the intersection area and
the access area are shown by a shaded area and a solid box,
respectively. Border of the access area is defined by daccess
that is the estimated stopping distance of a vehicle in case of
a safety concern and is calculated as a function of the road
average speed \( v_{avg} \):

\[
d_{\text{access}} = t_{\text{res}}v_{\text{avg}} - \frac{v_{\text{avg}}^2}{2a_{\text{dec,max}}} \tag{1}
\]

where \( t_{\text{res}} = 0.5 \) sec is assumed to be the response time
of an autonomous vehicle \([11]\), and \( a_{\text{dec,max}} = -4 \) (m/s²)
the maximum deceleration considered for passenger cars in
emergency braking. We obtain \( d_{\text{access}} \approx 38 \) m by setting
\( v_{\text{avg}}=56.3 \) kph (35 mph).

It should be emphasized that intersection access time \( t_{\text{access}} \)
for each vehicle is the time the vehicle enters the access area;
all other vehicles in the opposing movement must access the
access area at a sufficiently later time. If the intersection is
not yet cleared for safe passage of a vehicle in the opposing
movement then the access area provides enough stopping
distance to avoid a collision. The vehicles can be notified of
the presence of an opposing vehicle at the intersection not
only by their on-board sensors but also by the intersection
controller.

**Vehicle Attributes.** The attributes of each vehicle \( cv_i \in CV \)
\((1 \leq i \leq n)\) that is subscribed to the intended intersection
controller are described by:

\[
cv_i = (m_i, \phi_i, d_i, v_i, t_{\text{access,des},i}) \tag{2}
\]

and explained in Table II. Please note that in this paper, we
assume that all vehicles prefer to travel at the average velocity
\( v_{\text{avg}}=56.3 \) kph (35 mph); and as a result, their distance divided
by \( v_{\text{avg}} \) approximately yields their desired access times with
respect to current time \( t(0) = 0 \) sec.

If a vehicle has successfully followed the intersection con-
troller commands, it reports its previously assigned access time
as its new desired access time in each communication with the
intersection controller. In other words:

\[
t_{\text{access,des},i}(l) = t_{\text{access,des},i}(l-1) \tag{3}
\]

where \( t_{\text{access,des},i}(l) \) is the desired access time at the \( l^{th} \) execution
of the vehicle controller of \( cv_i \); and \( t_{\text{access,des},i}(l-1) \) is the
most recently updated access time previously communicated to
the vehicle \( cv_i \). This is intended to minimize the deceleration
and acceleration required each time the vehicle receives a new
assigned access time which is more fuel efficient.

**V. Problem Formulation**

**A. Objective**

The objective of increasing intersection throughput will be
formalized here as an optimization problem. The optimization
goal is to find the sequence and times of arrival for the vehicles
such that the maximum (latest) access time assigned to the
subscribed vehicles is minimized and any potential collision
is prevented. Furthermore, in defining the objective function
we take into account the desired arrival time of the vehicles
in such a way that vehicles would not face extreme delay or
expedition compared to their desired arrival times.

We formulate the main goal as to find the optimal sequence
and time of arrivals \( t_{\text{access}} \) for the subscribed vehicles such
that the difference between the current time \( t(0) \) and the
expected arrival time of the last vehicle passing the intersection
in a given time window is minimized:

\[
J_1 = t_{\text{access},j} - t(0) \tag{4}
\]

Minimizing the aforementioned objective could force the
vehicles to travel near the speed limit against their preference.
To avoid such a scenario, we define a cost on the difference
between assigned and desired access times for all vehicles:

\[
J_2 = \sum_{i=1}^{n} |t_{\text{access},i} - t_{\text{access,des},i}| \tag{5}
\]

The total cost function is then:

\[
J = w_1J_1 + w_2J_2 \tag{6}
\]

where \( w_1 \) and \( w_2 \) are penalty weights. We hypothesize that
this optimization will result in reduced fuel consumption and
intersection delay, even though these factors are not explicitly
incorporated into the objective function.

**B. Constraints**

Several constraints are imposed to ensure safety. The main
challenge is expressing the constraints as a function of access
times so that a linear constrained optimization problem can be
derived at the end. In this study, we are assuming that all ve-
hicles have same length, weight, and acceleration/deceleration
capabilities. In a real-world condition, these information needs
to be communicated at the time that vehicles send subscription
requests to the intersection controller.

---

**TABLE II. Notations used to express connected vehicles’ attributes.**

<table>
<thead>
<tr>
<th>Description</th>
<th>( m_i )</th>
<th>the vehicle movement ( m_i \in M )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \phi_i )</td>
<td>the phase ( \phi_i \in \mathcal{P} ) that ( cv_i ) movement is associated with</td>
<td></td>
</tr>
<tr>
<td>( d_i )</td>
<td>the distance of ( cv_i ) to the intersection access point</td>
<td></td>
</tr>
<tr>
<td>( v_i )</td>
<td>the velocity of ( cv_i )</td>
<td></td>
</tr>
<tr>
<td>( t_{\text{access,des},i} )</td>
<td>the ( cv_i )’s desired access time</td>
<td></td>
</tr>
</tbody>
</table>

---

**Fig. 3:** A schematic of different regions of the proposed intersection.
1) **Speed limit and maximum acceleration:** For each vehicle $cv_i$, we should consider the speed limit requirement $v_i \leq \text{vmax}$ as well as the maximum acceleration constraint $a_{i} \leq a_{\text{acc,max}}$; where $v_i$ and $a_i$ are the velocity and acceleration of the vehicle, $\text{vmax}$ is set based on the speed limit of the road, and $a_{\text{acc,max}}$ is set to $+3 \text{ m/s}^2$. We introduce $t_{\text{access,min},i}$ as the earliest time that $cv_i$ can access the intersection, if it travels at maximum acceleration and maximum speed possible. This earliest possible access time is calculated as explained in Figure 4. Then we rephrase our aforementioned speed and acceleration constraints as:

$$t_{\text{access},i} \geq t_{\text{access,min},i}$$  \hspace{1cm} (7)

2) **Safety gap on the same movement:** Two consecutive vehicles that are traveling on the same movement (e.g. east bound) should be separated by a safety gap (headway) that is denoted by $t_{\text{gap}1}$ in this paper. This time gap is independent of the vehicles’ speed [11] (except at very low speeds), and is the minimum following time gap to avoid a rear end collision. The computer control in autonomous vehicles will eliminate human reaction times; and assuming a very small communication delay, a time headway of 0.2 seconds may be sufficient for safe automatic vehicle following [30]. Adding a safety factor, it is suggested in [11] that a $h=1$ sec headway provides a reasonable upper bound for the response time of an autonomous vehicle. To enforce the safety gap of $t_{\text{gap}1} = h = 1$ sec at access point, we add the following constraint on any two consecutive vehicles traveling on the same movement:

$$t_{\text{access},j} - t_{\text{access},k} \geq t_{\text{gap}1}$$  \hspace{1cm} \forall cv_j, cv_k \in CV, \hspace{0.5cm} d_{j} \geq a_{k}, \hspace{0.5cm} m_{j}, m_{k} \in M, \hspace{0.5cm} m_{j} = m_{k}$$ \hspace{1cm} (8)

However, a 1 sec headway is not sufficient at very low speeds such as when discharging from a queue. As observed in our simulations, when discharging from a queue, the headway between the first vehicle and the second vehicle accessing the access point can be as large as 2.3 sec. This time gap can be significant for trucks or other longer vehicles [31] and is limited to the time period that front bumper and rear bumper of the lead vehicle pass and clear the access point. As an example, for a vehicle of length $L=5$ meter, stopping right at access point, and accelerating with $a_i = 2 \text{ m/s}^2$, this time is $\frac{\sqrt{2L/a_i}}{a_i} = 2.24$ sec which is consistent with our simulations. For vehicles stopped further in the queue, this time is shorter than 2.24 sec. As a result, for a standstill vehicle we set $t_{\text{gap}1}$ to the time it would take the vehicle body, with length of $L$, to completely pass over the access point or to 1 sec whichever is larger. For a very slow moving leading vehicle, we set $t_{\text{gap}2}$ to the time it would take a following vehicle to decelerate and maintain a minimum safe distance and time headway.

Please note that $t_{\text{gap}1}$ defines a lower bound on the headway between vehicles at an access point only. It means that: i) The actual headway at access point is a decision variable determined by the intersection controller and can be larger than $t_{\text{gap}1}$ after each intersection controller execution; and ii) The actual headway between two vehicles on their way to access area can be different from $t_{\text{gap}1}$. For example, if the spacing (a function of time headway) to the vehicle in front is getting dangerously short, a car-following subroutine that is incorporated into each vehicle trajectory control takes control of the vehicle, and does not let the spacing between vehicles go below a certain threshold based on their velocities. The car-following subroutine is explained shortly in Subsection VIII.

3) **Safety gap on conflicting movements:** Two vehicles traveling on conflicting movements also need to be separated by a safety gap. This time gap, if selected properly, guarantees that a vehicle can only enter the access area after all conflicting vehicles have left the intersection area. Considering two vehicles $cv_j$ and $cv_k$ that are on different phases of $\phi_j \in \phi$ and $\phi_k \in \phi (\phi_j \neq \phi_k)$, there are four possible situations with just enough safety gap between the vehicles. These situations are shown in a space-time diagram in Figure 5 by projecting the vehicles trajectories on one single distance axis. It can be concluded from this figure that the following constraints cover all the possible situations; here $\lor$ is the OR operator:

$$t_{\text{access},j} - t_{\text{exit},k} \geq 0$$  \hspace{1cm} \forall cv_j, cv_k \in CV, \hspace{0.5cm} \phi_j, \phi_k \in \phi, \hspace{0.5cm} \phi_j \neq \phi_k$$ \hspace{1cm} (9)

Although Equation (9) mandates at least 0 sec gap between the accessing and exiting time instances of the first and second vehicles, it does not explicitly state the gap needed between the accessing times of those vehicles. We are specifically interested in the time gap between access timestamps so that, at the end, we can formulate the optimization problem based on access times only. For this reason, we define $t_{\text{exit}} = t_{\text{access}} + \Delta t_{\text{travel}}$ where $\Delta t_{\text{travel}}$ is the travel time between access point and exit point of a vehicle. To simplify, we assume the intersection is a square and the two intersecting roads have the same speed limits. As a result, the travel time does not depend on the phase. By substituting $t_{\text{exit}} = t_{\text{access}} + \Delta t_{\text{travel}}$ into constraint (9), we can rephrase this constraint as:

$$t_{\text{access},j} - t_{\text{access},k} \geq t_{\text{gap}2}$$  \hspace{1cm} \forall cv_j, cv_k \in CV, \hspace{0.5cm} \phi_j, \phi_k \in \phi, \hspace{0.5cm} \phi_j \neq \phi_k$$ \hspace{1cm} (10)

where $t_{\text{gap}2} = \Delta t_{\text{travel}}$ is the safety gap we need between access times and is equal to the time period that a vehicle needs to first pass the access area, then pass the intersection area, and finally exit the intersection completely. The longest travel time a vehicle could take is when it is stopped behind the access
area and accelerates at its assigned access time as shown in Figure 6(a). We set an average acceleration of 2 m/s² and obtain Δtravel=7.3 sec as calculated in Figure 6(b) as the longest travel time. Consequently, we set tgap2=7.5 sec in our simulations as a conservative value. With the price of reducing the intersection throughput, this conservative safe gap will not only allow safe left and right turns with respect to conflicting traffic but also reduces the possibility of collisions in worst case scenarios.

VI. HANDLING OF REMOVABLE DISCONTINUITIES

We plan to solve the problem proposed in the previous section by linear programming. As a result, any discontinuity and disjunction in the formulation needs to be removed first. There are three disjunctions found in the previous section:

A. Discontinuity in Constraint

Constraint (10) has discontinuity and is not linear because it includes the OR logic operator (∨). The goal here is to convert this constraint into an AND-combination of two or more inequalities in such a way that if one equation holds true then the other equations are always redundant. The most widely known method to handle this disjunctions is the big-M method that, in our application, requires a binary variable B and a constant Mbig [32]. For each set of constraints in (10) for cvj and cvk, we add one artificial binary variable Bi (1 ≤ i ≤ # of constraints) to take care of the discontinuity as:

\[ t_{access,j} - t_{access,k} + Mbig B_i \geq t_{gap2} \]
\[ t_{access,k} - t_{access,j} + Mbig (1 - B_i) \geq t_{gap2} \]

\[ \forall cvj, cvk \in CV, \ \phi_j, \phi_k \in \Phi, \ \phi_j \neq \phi_k, \ B_i \ \text{binary} \]

where Bi can be either 0 or 1, and Mbig is a large enough number. If Bi=0 then the first equation of the above constraint holds true if \( t_{access,j} - t_{access,k} \geq t_{gap2} \) and the second equation \( (t_{access,k} - t_{access,j} \geq (t_{gap2} - Mbig)) \) is redundant and always holds if Mbig is large enough. If Bi=1 then the first equation \( t_{access,j} - t_{access,k} \geq (t_{gap2} - Mbig) \) is redundant and always holds true if Mbig is large enough, and the second equation holds true if \( t_{access,k} - t_{access,j} \geq t_{gap2} \).

It is possible to predict how large Mbig must be because either of the redundant equations discussed above needs to be always fulfilled; and this requires that Mbig ≥ tgap2 + taccess,j − taccess,k. Considering the fact that tgap2 is small, and can be neglected compared to Mbig, a lower bound to Mbig is equal to the latest possible access time with respect to current time. As a worst-case scenario, a vehicle would travel the whole subscription distance (e.g. 2 km) at a very low speed (e.g. 20 kph) and would stop at access point waiting for a long time (e.g. 500 sec); that leads to a 860 sec interval before the vehicle can access the intersection. Although Mbig can be 860 sec, we set Mbig to a large enough constant just to consider all possible scenarios such as longer subscription distances and longer waiting delays. We set Mbig to 2000 in our formulation.

B. Discontinuity in Cost Function J1

The cost function J1, introduced in Equation (4), is discontinuous in the sense that it includes a decision variable (the largest access time). One solution, as used in our previous work [16] and [33], is to always assign the largest (last) access time to the furthest subscribed vehicle that is cvj, considering the fact that the list of all subscribed vehicles (CV) is sorted by distance to the intersection. We then rephrased the optimization objective and constraint as given in Equation (12) where taccess,n is the access time assigned to the cvj and should be larger than or equal to the access times of all other vehicles.

\[ J_1 = t_{access,n} - t_0 \]
\[ \text{s.t. } t_{access,n} \geq (\{t_{access,1}, \ldots, t_{access,n-1}\}) \]

However, in this manuscript, we restate Equation (4) as \( J_1 = \max(\{t_{access,1} - t_0, \ldots, t_{access,n} - t_0\}) \) which is a min-max problem. As a result, we can replace J1 with a new so-called slack variable Δaccess,latest and n extra constraints as follows:

\[ J_1 = \Delta_{access,latest} \]
\[ \text{s.t. } \Delta_{access,latest} \geq (t_{access,i} - t_0) \ \forall i \in \{1, \ldots, n\} \]
C. Discontinuity in Cost Function J2

The absolute value signs cannot be included in a linear programming formulation; as a result, the cost function $J_2$, introduced in Equation (5), needs to be restated. We restate $J_2$ by adding a new so-called slack variable $\Delta t_{\text{access},i} = |t_{\text{access},i} - t_{\text{access},i,\text{des}}|$. Then, considering the fact that $|x| = \max\{x, -x\}$ for any real number $x$, we can add two constraints of $\Delta t_{\text{access},i} \geq (t_{\text{access},i} - t_{\text{access},i,\text{des}})$ and $\Delta t_{\text{access},i} \geq -(t_{\text{access},i} - t_{\text{access},i,\text{des}})$ in order to ensure that our added slack variable is equal to $|t_{\text{access},i} - t_{\text{access},i,\text{des}}|$. In summary, the restated cost function and the added constraints are as follows:

$$J_2 = \sum_{i=1}^{n} \Delta t_{\text{access},i,\text{abs}}$$

s.t. $\quad \Delta t_{\text{access},i,\text{abs}} \geq (t_{\text{access},i} - t_{\text{access},i,\text{des}})$

$$\Delta t_{\text{access},i,\text{abs}} \geq -(t_{\text{access},i} - t_{\text{access},i,\text{des}})$$

VII. MILP CASE STUDY

The linear objective function/constraints and mixed-integer variables in the optimal solution make our problem a Mixed Integer Linear Program (MILP) for which efficient methods exist. Our detailed microsimulation results with measure of effectiveness (MOE) analysis will be given later in Section XI; however, this section employs a simplified case study only to observe an example numerical output of the proposed formulation.

To solve this MILP problem in this section, we use IBM’s CPLEX optimization package. We simulate the same example as previously shown in Figure 2 including $n=9$ connected vehicles $c_i$ ($1 \leq i \leq 9$). We set the speed limit to $v_{\text{max}}=72.4$ kph (45 mph), and average arterial road speed to $v_{\text{avg}}=56.3$ kph (35 mph). We assume that the current state of all the vehicles is available: they are all traveling at $v_{\text{avg}}$ and their distance to access area are [690, 750, 780, 900, 990, 1080, 1170, 1230, 1290] meters, respectively.

Figure 7(a-d), demonstrates the optimal solutions to the aforementioned problem found by MILP. As shown in Figure 7(a), by setting $w_1=100\%$ and $w_2=0\%$, all the weight is given to intersection throughput improvement ($J_1$, cost function (13)), and, as a result, some vehicles end up traveling near the speed limit (solid red lines). On the other hand, by setting $w_1=0\%$ and $w_2=100\%$ in Figure 7(b), all the weight is given to satisfying the desired speeds of all vehicles ($J_2$, cost function (14)); and, as a result, the intersection clearance time was increased by 13 sec compared with Figure 7(a). Two compromised solutions are also demonstrated in Figure 7(c) and (d) for comparison purposes. Because of the updated cost function introduced in (13), the intersection clearance time of these solutions are improved compared to our results previously presented in [16]. For our microsimulation, we set equal weights as $w_1=50\%$ and $w_2=50\%$.

A feature of our formulation is that overlapped arrival-times can be assigned to two vehicles that are traveling on different parallel traffic movements (e.g. $c_4$ and $c_2$ in Figure 2 and 7(b-d)). This also holds true for two opposing vehicles both making left or right turns because they are not considered conflicting movements and they can be served at the same time. However, a safety gap of $t_{\text{gap2}}$ is needed to separate a left- or right-turning vehicle from its conflicting traffic movements. Most important feature that can also be identified in the Figure 7(a), is that since $t_{\text{gap2}} > t_{\text{gap1}}$, platoon formations will be encouraged, otherwise lone vehicles have to clear the intersection with the longer safe gap $t_{\text{gap2}}$ with respect to opposing movements which reduces intersection capacity.

VIII. AUTONOMOUS TRAJECTORY-PLANNING

In order to guide the autonomous vehicles, a trajectory-planning algorithm needs to run locally on individual vehicles’ control system. Our trajectory-planning engine has two subroutines: i) an access-time-tracking subroutine that computes a feasible trajectory for the vehicle in such a way that it can reach and pass the access area at its assigned access time; and, ii) a car-following subroutine that constrains the vehicle’s
speed to obtain a safe space headway to the vehicle in front, if necessary. Note that in this paper, to simplify the model of an autonomous vehicle, we assume there is no obstacle to avoid and lane-changing is not allowed.

A. Planning an Uninterrupted Passage of a Vehicle

The basic fact to be considered before explaining the access-time-tracking algorithm is that if a vehicle’s remaining travel time to the intersection access area at current speed is equal to the remaining time to its assigned access time, then, obviously, the vehicle doesn’t need to adjust its speed. As shown in Figure 8(a), this means that the vehicle can continue driving at its current speed if \( t_{\text{access}, i} - t_0 = \frac{d_i}{v_i} \), where \( d_i \) and \( v_i \) are the current distance and velocity at current time \( t_0 \), and \( t_{\text{access}, i} \) is the access time assigned to the vehicle by the intersection controller. Most often, however, acceleration or deceleration is necessary to adjust the vehicle speed as explained below:

- **Acceleration:** If \( t_{\text{access}, i} < t_0 + \frac{d_i}{v_i} \), the vehicle needs to accelerate to a cruising speed in order to make it to the access point at its assigned timestamp. As shown in Figure 8(b), the vehicle should keep that speed \((v_{\text{cruise}, i})\) till it reaches the access area. Knowing the desired acceleration \( a_{\text{acc}} \), the cruising speed is calculated by:

\[
v_{\text{cruise}, i} = v_i + a_{\text{acc}} \Delta t - \sqrt{2a_{\text{acc}} \Delta t^2 + v_i \Delta t - d_i}
\]

where \( \Delta t \) is the remaining time to the assigned access time, \( d_i \) is the remaining distance to the access point, and \( v_i \) is the vehicle’s velocity at current time. We set \( a_{\text{acc}} = 2m/s^2 \) as the desired acceleration of vehicles, only if it satisfies the condition of \( 2m/s^2 \geq \frac{2(d_i-v_i\Delta t)}{\Delta t^2} \) to avoid taking the square root of negative number in Equation (15); otherwise, we set \( a_{\text{acc}} = \frac{2(d_i-v_i\Delta t)}{\Delta t^2} \).

- **Deceleration:** If \( t_{\text{access}, i} > t_0 + \frac{d_i}{v_i} \), the vehicle would be early at access point if it maintains its current speed. Thus, the vehicle needs to decelerate to a cruising speed \((v_{\text{cruise}, i})\) as shown in Figure 8(c). Knowing the desired deceleration \( a_{\text{dec}} \), the cruising speed is calculated by:

\[
v_{\text{cruise}, i} = v_i + a_{\text{dec}} \Delta t + \sqrt{2a_{\text{dec}} \Delta t^2 + v_i \Delta t - d_i}
\]

We set \( a_{\text{dec}} = -2m/s^2 \) as the desired deceleration of vehicles only if it satisfies the condition of \(-2m/s^2 \leq \)

\[2(d_i-v_i\Delta t) \leq \Delta t^2 \]

\[a_{\text{dec}} = \frac{2(d_i-v_i\Delta t)}{\Delta t^2} \]

B. Planning a Stop at Queue or Access-Point

The aforementioned equations do not always have solutions. In other words, it is not always possible for a vehicle to pass without stopping. If the assigned access time is far away in time such that even a very slow-moving vehicle reaches the intersection before that time, then the vehicle should be prepared to stop. After stopping at an access point or at a queue behind access point, the vehicle waits for its reserved access time to come before it starts to proceed to the access area and the intersection. To formulate this into our trajectory-planning algorithm, we introduce \( t_{\text{access}, \text{max}, i} \) as the latest time that the vehicle can potentially access the intersection if it travels at minimum cruising speed \((v_{\text{min}})\) and average deceleration. Then, the vehicle needs to be prepared for a complete stop if:

\[t_{\text{access}, i} \geq t_{\text{access}, \text{max}, i}\]

where \( t_{\text{access, max, i}} \) is locally computed by the trajectory-planning engine of each vehicle as explained in Figure 9.

C. Car-Following Subroutine

There is a close interaction between the car-following and access-time-tracking subroutines. As an example, let \( j - 1 \) and \( j \) be a pair of vehicles, with \( j \) following \( j - 1 \). If \( j - 1 \) is moving slower and not obeying intersection controller commands then it may interfere with \( j \). This interference does not cause collision because vehicle \( j \)'s trajectory-planning engine has a car-following subroutine that takes the control of the vehicle and avoids collision if the spacing between \( j \) and \( j - 1 \) is getting dangerously short. The access-time-tracking subroutine of \( j \) takes the control back from the car-following if \( j - 1 \) accelerates increasing the inter-vehicle gap.

To illustrate this, Figure 10(a) shows the travel trajectories and access times of these vehicles if they both obey the intersection controller commands. Figure 10(b), on the other hand, shows a simulation where \( j - 1 \) slows down to 24.1 kph (15 mph) at 10 sec and doesn’t obey the intersection controller commands until 60 sec of the simulation.

It should be emphasized here that no vehicle is allowed to cross the intersection (access area) at a time different from the access time assigned to the vehicle. This avoids side impacts while the car-following subroutine prevents rear-end collisions. Figure 11 demonstrates a two-vehicle case study simulation with similar conditions of Figure 10 but for vehicles on two
after the other conflicting vehicle (Fig. 11: Scheduling of two conflicting vehicles

\[ t_{\text{access}}, j \] doesn’t follow the intersection controller commands between 10 sec and 60 sec of the simulation.

conflicting phases. In Figure 11(b), the slow moving vehicle (cvk) receives an updated access time commanding it to pass after the other conflicting vehicle (cvj) and avoiding a side impact. If this updated access time is not sent to cvk, this vehicle does not pass the access area and waits at the access point to receive a feasible access time.

D. Managing Access Time Infeasibility

If an unwanted change or disturbance happens locally to a vehicle (e.g. because of a slow-moving vehicle ahead), a previously assigned access time may become infeasible, not allowing the vehicle to pass the intersection. Infeasibility can be identified when a vehicle cannot reach the access area at its assigned timestamp even if it finds the opportunity to travel at its maximum acceleration and speed possible. In other words, an assigned access time of \( t_{\text{access}}, j \) is infeasible if:

\[ t_{\text{access}}, j < t_{\text{access}, \text{min}}, j \]  \hspace{1cm} (18)

where \( t_{\text{access}, \text{min}}, j \) is previously explained in Figure 4.

With the intersection controller, the car following and access-time-tracking subroutines all running algorithms continuously, the vehicles are capable of making real-time adjustments even in the infeasible situation. If no feasible access time is available, the vehicle is commanded to travel at its desired velocity (free flow) which may be constrained by the car following subroutine if there is a slow-moving vehicle ahead. This continues till the next execution of the intersection controller where it assigns a new feasible access time to the vehicle and also updates the access times of other vehicles accordingly. If no appropriate access time is assigned by the intersection controller well ahead before the access area, the trajectory-planning engine commands the vehicle for a complete stop at access point or at a queue behind access point. The stopped vehicle then needs to wait till the next execution of the intersection controller.

It should be emphasized that it is probable that the disturbance (e.g. a slow-moving vehicle) fades away before Equation (18) warns an infeasible situation. In this case, the access-time-tracking subroutine takes the control back and recommends an updated travel trajectory to the vehicle by rerunning its algorithms (subsections VIII-A). The updated trajectory probably recommends larger acceleration and speed to compensate the time wasted.

IX. BENCHMARKING TESTBEDS

In order to conduct comprehensive evaluations in this paper, we implemented two simulation testbeds with pre-timed traffic signal control systems. Autonomous vehicles approaching a pre-timed traffic signal control provide a baseline testbed, against which we can benchmark and compare our MILP-based intersection controller. The signal phase and timing for the benchmark pre-timed traffic signals were obtained off-line from SYNCHRO (Trafficware 2011) optimization program. This optimized signal phase and timing were then used in the microsimulation to model the current state of the traffic light (cycle time = 100 sec, green split = 44.5 sec, and yellow interval = 3.5 sec).

A. Testbed A: Camera-based Trajectory-Planning at Pre-timed Signalized Intersection

In this testbed, we assume that there is no communication between the autonomous vehicles and the traffic signal devices, and the vehicles can observe the current state of the traffic signals ahead only with their on-board cameras. In our simulations, as soon as a vehicle is within the range of its imaginary camera (300 m), the current state of the simulated traffic light is fed into the vehicle’s trajectory-planner.

B. Testbed B: Communication-based Trajectory-Planning at Pre-timed Signalized Intersection

We also consider another benchmark algorithm where all autonomous vehicles are assumed to be able to receive the deterministic future state of traffic signals via unidirectional wireless communications when they are within the range of 500 m from the intersection. As soon as this information is received, the trajectory of each individual autonomous vehicle, in free flow, is planned based on the speed advisory algorithm proposed by our group in [1], [2]. This algorithm is modified and its details are not included here in consideration of the length of the paper.
Nevertheless, an example is given in Figure 12 to clarify the general features of our modified speed advisory system. Figure 12 shows the feasible speed intervals that a vehicle traveling at a speed of \( v_1 \) can follow without stopping at red. These speed intervals are limited to the speed limit \( v_{\text{max}} \) (e.g. 45 mph) and the minimum cruising speed possible \( v_{\text{min}} \) (e.g. 20 mph). In this paper, we add a constant buffer \( \Delta t = 4 \text{ sec} \) after each start-of-green so that a vehicle would never be advised to cross the stop-bar at red. Among the available speed intervals shown in Figure 12, we choose a target speed as close as possible to the desired average speed \( v_{\text{avg}} \).

At this testbed, we are assuming that the autonomous vehicles have information about the instantaneous queue size when they are within 300 m of the intersection. If a queue exists then the vertical axis of Figure 12 represents the distance to the rear end of the queue. In this case, the safety buffer \( \Delta t \) compensates, in part, for the queue dissipation time.

### X. Computational Load

In microsimulations of our MILP-based intersection control scheme, the MILP problem was solved by IBM’s CPLEX optimization package running on Intel Core i5@2.5 GHz Windows 7 laptop with 8 GB of RAM. By tuning this package for performance improvement and by reducing preprocessing computational load, we were able to achieve an average intersection controller execution time of 120 ms for an average number of 50 subscribed vehicles over one hour of simulation. The intersection controller execution time varied between 28 ms and 2400 ms for 50 subscribed vehicles. It should be emphasized that MILP is running by the intersection controller every 4 sec; as a result, the system is able to adapt its recommendations to unwanted interference or delays, e.g. change in speed because of a slow-moving vehicle ahead.

The aforementioned execution time includes the MILP solver execution time plus the time needed for preprocessing the probe vehicle data and expressing the problem in canonical form of: minimizing \( J \) subject to \( Ax \leq b \). In this form, \( x \) is the vector of decision variables to be determined defined as \( x = \{ t_{\text{access,1}}, \ldots, t_{\text{access,n}}, \Delta t_{\text{access,abs,1}}, \ldots, \Delta t_{\text{access,abs,n}}, B_1, \ldots, B_m, \Delta t_{\text{access,latest}} \} \); where \( n \) is the number of all subscribed vehicles, and \( m \) is the number of artificial binary variables of Equation (11). Also, \( A \) is a matrix and \( b \) is a vector of coefficients that together represent the constraints and bounds that appeared in Equations (7), (8), (11), (13), and (14).

### XI. Simulation Results

In this section, we try to determine the potential in a MILP controlled intersection. Towards this objective, in our simulation, we assume that instantaneous vehicle information is available to the intersection controller, no disturbance input exists in the simulated traffic, there is no obstacle to avoid and lane-changing is not allowed, all vehicles prefer to travel at the average velocity, and they all have same length and acceleration capabilities. The microsimulation was implemented using Java. The screenshots of the implemented simulation environments for three testbeds are shown in Figure 13.

Each simulated testbed includes an intersection with four legs, each 500 m long. The simulated vehicles arrive based on a probabilistic generation method: a negative Exponential distribution [34] was used for 750 vehicles/hour for all four approaches. The vehicles’ arrival pattern is recorded and replayed for each testbed. In this way, the same arrival pattern was replicated for each testbed. Three simulations were conducted for Testbeds A, B, and MILP. The average and maximum speeds were set to \( v_{\text{avg}} = 56.3 \text{ kph} \) (35 mph) and \( v_{\text{max}} = 72.4 \text{ kph} \) (45 mph). For our communication-based testbeds (testbed B and MILP), the subscription range is 500 m.

The MOE results of the simulations are given in Table III, where the performance improvements achieved by our MILP-based intersection controller are also provided compared to Testbeds A and B. The results of our MILP-controlled intersection were obtained by setting \( w_1 = 50\% \) and \( w_2 = 50\% \) in Equation (6); however, it may be possible to tune these parameters for improved results. The MOEs studied in our simulations and given in Table III are: (1) the intersection total number of stops, (2) the intersection total stopped delay, (3) the average stopped delay per stopped vehicle, and (4) the average travel time per vehicle. As shown in Table III, by using our MILP-based control at Testbed MILP, the intersection delay and stops were significantly reduced compared to pre-timed intersection benchmarks; and travel times were not compromised. These improvements were obtained although the road capacities were reduced in Testbed MILP. The capacity is reduced because the queue starting point for Testbed MILP is the access point, while for other testbeds it is the stop-bar at intersection.

![Fig. 13: Screenshots of simulation testbeds. Testbed A: Pre-timed signalized intersection with no communication. Testbed B: Pre-timed signalized intersection with unidirectional communication and speed-advisory. Testbed MILP: MILP-controlled intersection with bidirectional communication.](image)
MILP results are also affected by the conservative safe gap value of $t_{gap} = 7.5$ sec; however, this amount of gap reduces the chance of collisions in worst case scenarios.

By using the speed advisory at Testbed B, the total duration and number of stops at intersection is decreased compared with that of Testbed A; however, the average travel time has not been improved (shown in Table III). The reason is that, in Testbed B, most of the vehicles that manage to pass the intersection without stopping have been commanded to travel at slow speeds in order to avoid a red light.

### XII. Modified Design for Mixed Traffic Conditions

In this section, we provide a modified version of our MILP-based intersection control scheme that can be applied to a mixed traffic consisting of automated and human-controlled vehicles. Assuming all the vehicles are connected to the intersection controller, we make the following modifications:

- Physical traffic signals are added and simulated at the access points. The status of the traffic light (green, yellow, and red) controls the traffic flow to the access area at each of four approaches to the intersection. It is possible to have the traffic lights located at the stop-bars, controlling the traffic flow to the intersection area instead. However, this requires major modification to our intersection control scheme.

- All autonomous vehicles travel based on their assigned access time with the desired velocity of $v_{avg}$. The non-autonomous vehicles desire to travel at a randomly selected velocity between $v_{min}$ and $v_{max}$.

- Both autonomous and non-autonomous vehicles decide to pass or stop at the access point based on the traffic light status only. When autonomous vehicles are in close proximity of the traffic light, they ignore their assigned access times and pass at green or stop at red signal. Thus, an autonomous vehicle has the flexibility to pass at a time different than its assigned access time as long as the light is green.

- Considering a slower response time for drivers, the MILP-based intersection controller assumes $t_{gap} = 2$ sec if a following vehicle is not autonomous.

#### TABLE III: The simulation results, and the overall performance improvements achieved by MILP-based intersection controller when all vehicles are connected and autonomous.

<table>
<thead>
<tr>
<th>MOE1 (for all simulated vehicles)</th>
<th>Testbed A</th>
<th>Testbed B</th>
<th>Testbed MILP2</th>
<th>Gain achieved comparing to Testbed A</th>
<th>Testbed B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intersection traversals</td>
<td>2900</td>
<td>2900</td>
<td>2900</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Test duration</td>
<td>1h 1min</td>
<td>1h 1min</td>
<td>1h 1min</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Total intersection number of stops</td>
<td>1171</td>
<td>872</td>
<td>13</td>
<td>98.9%</td>
<td>98.5%</td>
</tr>
<tr>
<td>Total intersection stopped delay</td>
<td>6h 40min</td>
<td>3h 43min</td>
<td>1min 51sec</td>
<td>99.5%</td>
<td>99.2%</td>
</tr>
<tr>
<td>Average stopped delay per Stopped Vehicle</td>
<td>20sec</td>
<td>15sec</td>
<td>9sec</td>
<td>55%</td>
<td>40%</td>
</tr>
<tr>
<td>Average travel time3 per Vehicle</td>
<td>50sec</td>
<td>51sec</td>
<td>36sec</td>
<td>28%</td>
<td>29%</td>
</tr>
</tbody>
</table>

1 All MOEs are reported for one intersection with four approaches (each approach is 500 m long).
2 The MILP results were obtained by setting $w_1=50\%$ and $w_2=50\%$.
3 Travel time over a distance of 500 m to the intersection.

#### Fig. 14: The green/red timing output of the modified intersection control, applicable to mixed traffic of non-autonomous and autonomous connected vehicles at a signalized intersection.

At 100% penetration, during one hour simulation, we observed 685 stops, 1h and 42min total stopped delay, 9 sec average stopped delay per stopped vehicle, and 47 sec average travel time. Please note, because of having a physical traffic signal, we should use our modified intersection control scheme presented in this section to simulate a 100% penetration of autonomous vehicles. Otherwise, the traffic signal switches rapidly between phases and it doesn’t turn from red to green till the moment an autonomous vehicle crosses the access point (see Figure 2). That is why the aforementioned results for an all-autonomous environment are different from Table III.

### XIII. Conclusions & Future Work

We proposed a novel intersection control scheme for an all autonomous vehicle environment. Our five key contributions are in: i) an intersection control algorithm that anticipates vehicle arrivals and facilitates uninterrupted passage of them at intersections, ii) reducing the vehicle-intersection coordination problem to a mixed-integer linear program, iii) the
improvements in traffic flow, compared to [16], by introducing an updated cost function, iv) a trajectory planning algorithm that guides autonomous vehicles for a timely arrival at intersections, and v) developing a customized microsimulation test environment in which simulated vehicles are guided by our trajectory-planning algorithm. Microsimulation results demonstrated that our linear formulation not only significantly reduced the intersection delay and stops compared to pre-timed intersection benchmarks, but also ensured that no crash occurred and did not compromise the travel time. Our recent work, on implementing a Vehicle-In-the-Loop (VIL) platform [33], showed that our intersection-vehicle coordination scheme provided 19.5% benefit in fuel consumption compared to our baseline testbed with a pre-timed traffic signal.

The initial proposed algorithm considers a futuristic all autonomous driving environment and eliminates the need for physical traffic signals (100% penetration rate of equipped vehicles). This can be utilized in future’s dense city centers where only autonomous vehicles would be allowed to travel. However, we also modified the proposed algorithm to be applied to mixed traffic consisting of autonomous and human-controlled vehicles, all connected to the intersection controller of a signalized intersection. More modifications are required to make this intersection control scheme applicable to situations with purely human-driven vehicles that have no wireless connectivity; as an example, a maximum green time needs to be considered for each phase of the traffic light. Other future work includes simulating left- and right-turning vehicles as well as multi-lane intersections.

ACKNOWLEDGMENT

This research was partially sponsored by a research award from BMW Information Technology Research Center (ITRC) in Greenville, SC, USA. The authors are thankful for the support provided by Dr. Andre Luckow from BMW ITRC.

REFERENCES


---

S. Alireza Fayazi is currently working as a post-doctoral fellow at Clemson University, Clemson, SC, USA, where he obtained PhD in dynamic systems and controls (mechanical engineering). He received his M.Sc. from University of Tehran, Tehran, Iran and his B.Sc. from K. N. Toosi University of Technology, Tehran, Iran both in Electrical and Electronics Engineering. In 2012-2013, he was a visiting researcher at the University of California, Berkeley and was also working as a visiting researcher in BMW Group Technology Office located in Mountain View, California. Before joining Clemson University, he was a research and development engineer at Kerman Tablo Corp. for three years where he worked on discrete control systems and digital control for embedded applications.

Ardalan Vahidi is a Professor of Mechanical Engineering at Clemson University, South Carolina. He received his Ph.D. in mechanical engineering from the University of Michigan, Ann Arbor, in 2005, M.Sc. in transportation safety from George Washington University, Washington, DC, in 2002, and B.S. and M.Sc. in civil engineering from Sharif University, Tehran in 1996 and 1998, respectively. In 2012-2013 he was a Visiting Scholar at the University of California, Berkeley. He has also held scientific visiting positions at BMW Technology Office in California, and at IFP Energies nouvelles, in France. His research is at the intersection of energy, vehicular systems, and automatic control. His recent publications span topics in alternative vehicle powertrains, intelligent transportation systems, and connected and autonomous vehicle technologies.