Safe Data-Driven Distributed Coordination of Intersection Traffic

Darshan Gadginmath Pavankumar Tallapragada

***Abstract*—This work addresses the problem of traffic management at and near an isolated un-signalized intersection for autonomous and networked vehicles through coordinated optimization of their trajectories. We decompose the trajectory of each vehicle into two phases: the provisional phase and the coordinated phase. A vehicle, upon entering the region of interest, initially operates in the provisional phase, in which the vehicle is allowed to optimize its trajectory but is constrained to guarantee in-lane safety and to not enter the intersection. Periodically, all the vehicles in their provisional phase switch to their coordinated phase, which is obtained by coordinated optimization of the schedule of the vehicles' intersection usage as well as their trajectories. For the coordinated phase, we propose a data-driven solution, in which the intersection usage order is obtained through a data-driven online "classification" and the trajectories are computed sequentially. This approach is computationally very efficient and does not compromise much on optimality. Moreover, it also allows for incorporation of "macro" information such as traffic arrival rates into the solution. We also discuss a distributed implementation of this proposed data-driven sequential algorithm. Finally, we compare the proposed algorithm and its two variants against traditional methods of intersection management and against some existing results in the literature by micro-simulations.**

***Index Terms*—Intelligent transportation systems, distributed control, data-driven control, state-based intersection management, networked vehicles, optimized and provably safe operation**

I. Introduction

Signalized intersections suffer from several inefficiencies, such as low throughput and extensive braking and acceleration manoeuvres that increase the fuel consumption as well as cause discomfort to the passengers. The advent of *autonomous vehicles* (AVs) and vehicular communication presents an opportunity to rethink the problem of intersection management. Further, in recent years, relevance of intersection management has grown in non-traditional domains such as robot traffic control in warehouses. The onboard sensing, computation and communication capabilities that are available on AVs or robots allow us to implement algorithms for real time coordination of the AVs or robots to achieve a more efficient and un-signalized intersection management. In this work, we propose a computationally efficient distributed algorithm and a framework for offline data-driven tuning for the management of an isolated intersection in the context of AVs or robots.

Literature review: Un-signalized intersection management has been studied extensively in recent years using a variety of tools. Some early works [1]–[4] focused on reservation and multi-agent simulation based algorithms. Some disadvantages of such solutions is that they are computationally demanding, centralized and do not easily provide insights into the system. Since then a major trend in the literature has been to design model based, provably safe algorithms. For example, [5] uses reservations for scheduling intersection usage times while a model for the vehicle dynamics allows the design of provably safe trajectories for the vehicles. Another approach to intersection management came from supervisory control [6], [7] (see also the references therein). This work proposes a method where a supervisor takes over only when a collision is imminent. However, the trajectories followed by the vehicles can be uncomfortable for the passengers and the control does not proactively seek an efficient operation of the vehicles or the intersection. Another major trend in the field of autonomous intersection management is the use of an optimal control framework for determining the schedules and trajectories of the vehicles. Fundamentally, the problem of finding an optimal schedule is combinatorial and thus the overall problem of trajectory optimization becomes a mixed integer program [8]–[10]. However, the complexity of such formulations scales exponentially with the number of vehicles. This limits the utility of the basic formulation to a time and safety critical application such as intersection management.

Given the computational complexity of the problem, along with the motivation of designing distributed algorithms, several works have sought to decompose the overall autonomous intersection management problem into simpler sub-problems. The work in [11], [12] together proposes a high level intersection access management by treating the vehicles on different lanes as queues and uses the idea of platooning for local vehicular control. Given intersection usage schedule for the vehicles, [13], [14] (and the references therein) seek to solve the trajectory optimization problem in a decentralized manner by relaxing the rear-end collision avoidance constraints and guarantee existence of initial conditions under which the safety constraints are satisfied. Further, these works also propose a method to drive the vehicles to good "initial conditions" under which safety can be guaranteed subsequently. [15]–[19] also decompose the problem into scheduling and trajectory optimization. [15] proposes a decentralized architecture where a model-based heuristic, which relies on a notion of "time to react", provides an order in which each vehicle sequentially solves two optimal control problems to identify the best time of entry into the intersection. [16] determines the schedule using a notion called "temporal advantage" and computes an optimal control based motion planner to generate the trajectories for the vehicles. However, in order to guarantee feasibility of the trajectory planning problem, [16] assumes that the vehicles' deceleration may be unbounded. [17] proposes a polling system to obtain the schedule and then the vehicles generate their trajectories sequentially. [18] utilizes a bilevel controller

This work was partially supported by the Wipro IISc Research and Innovation Network.

Darshan Gadginmath and Pavankumar Tallapragada are with the Department of Electrical Engineering, Indian Institute of Science, Bangalore, India {darshang@iisc.ac.in, pavant}@iisc.ac.in

in which a mixed integer linear program based centralized coordinator assigns slots for intersection usage for the vehicles. The vehicles themselves are controlled by a model predictive controller. Similarly, [19] proposes an algorithm, in which a central intersection manager groups vehicles into bubbles and schedules the bubbles as a whole to use the intersection. Given the schedule, the vehicles compute provably safe trajectories using a distributed switched controller.

Comfort of passengers and generation of smooth trajectories for vehicles is another area of interest in autonomous intersection management. [20] surveys driver comfort in autonomous vehicles and highlights the inadequate research on passenger comfort in path and motion planning of autonomous vehicles. [21] studies the problem of vehicles merging into highways and uses a model predictive control architecture that optimizes comfort by minimizing the squares of both acceleration and jerk. [22] introduces a metric of comfort which is a combination of vehicle-jitter, jerk and deviation from a desired velocity. [14] and [23] focus on vehicles turning at the intersection and impose a curvature-based acceleration constraint to capture the comfort of the passengers.

Contributions: In this paper, we address the problem of coordinating and optimizing the trajectories of vehicles at and near an isolated, un-signalized autonomous intersection. We provide a complete solution that is data-driven, provably safe and distributed. Moreover, the proposed online algorithm provides near optimal solutions while being computationally very efficient. This is the major contribution of the paper and it is a fundamentally new, previously unexplored, approach to the problem. The key insight behind the proposed data-driven framework is that the computation of the optimal intersection usage order can be thought of as an online classification problem from a space of features that encode the "demand" of a vehicle and the traffic following it to the vehicle's precedence for using the intersection. The proposed framework also has the ability to incorporate both "micro" information about the individual vehicles' state as well as "macro" information such as traffic arrival rates. Such combination again has not been explored in the literature. This element of our framework is particularly useful under high traffic arrival rates.

The second set of major contributions of this paper relates to how we handle a continuous stream of vehicles. Most papers in the area essentially propose a one-shot algorithm with the implicit suggestion that the one-shot algorithm should be run repeatedly. However, not explicitly considering the continuous stream of vehicles could in general lead to loss of feasibility of safe trajectories. The solution offered in [19] to this problem is to split each incoming branch into zones and ensuring that these zones are long enough given the speed limit and the time duration between the schedule assignment instants. Thus, this approach can be limiting and may not be applicable to all intersections and parameters such as speed limit. In the proposed framework of this paper, we split the trajectory of each vehicle into two phases (1) *provisional phase* and (2) *coordinated phase*. Every vehicle operates in the provisional phase as soon as it enters the system. In this phase, vehicles are allowed to optimize their trajectories under the constraint that they maintain safety and not enter the intersection. Periodically, the vehicles in the provisional phase obtain a trajectory for their coordinated phase and start executing them. This framework thus offers a complete solution for an arbitrary intersection and for a continuous stream of vehicles while ensuring safety, feasibility and near optimality of the solutions.

Lastly, we evaluate the performance of our algorithm through an extensive collection of simulations. In particular, we compare our algorithm with that of an "optimal" algorithm, signalized intersection management, stop sign based intersection management as well as the algorithm proposed in [19]. We also demonstrate the computational efficiency of the proposed algorithm through some coarse metrics, that indicate the general trends in computation time.

Notation: We use $\mathbb{R}$ and $\mathbb{N}_0$ for the set of real and whole numbers, respectively.

II. Model and Problem Formulation

A. Model

Geometry of the Region of Interest: We consider an isolated intersection with 4 incoming branches, each with 3 possible exit lanes - left, straight and right. We denote by

$$\mathcal{L} := \{1, 2, 3, \ldots, 12\}, \quad \mathcal{L}_S := \{2, 5, 8, 11\},$$
$$\mathcal{L}_L := \{1, 4, 7, 10\}, \quad \mathcal{L}_R := \{3, 6, 9, 12\},$$

the set of all lanes and the set of lanes that go straight across the intersection, that go left and that go right, respectively. We refer to the lanes that lead to the intersection together with the intersection itself as the *region of interest*. Every lane $l \in \mathcal{L}$ has a unique path $P_l \subset \mathbb{R}^2$ associated with it. We denote by s_l the length of the portion of the path P_l, that lies within the intersection. The length of all the paths leading to the intersection is d. Along the path on lane l, we let the positions at the beginning of the region of interest, the beginning of the intersection and the end of the intersection be $-d$, 0 and s_l, respectively. We present the basic geometry of the region of interest in Figure 1. The translucent box shaded in red represents the intersection, which is the potential region of inter-lane collisions.

As it is apparent from Figure 1, the paths of some lanes do not intersect while others do. For each pair of lanes $l, m \in \mathcal{L}$, we define a notion of *compatibility* $c(l, m)$ as

$$c(l,m) := \begin{cases} 1, & P_l \cap P_m = \emptyset \\ 0, & P_l \cap P_m \neq \emptyset, \ P_l \neq P_m \\ -1, & P_l = P_m. \end{cases}$$

We say that a pair of lanes l and m are *compatible* if $c(l, m) = 1$ and *incompatible* if $c(l, m) = 0$. In the sequel, we require that vehicles on incompatible lanes not be in the intersection at the same time.

Vehicles and their Dynamics: Here we discuss about the vehicles that enter the region of interest and use the intersection. We make the following assumptions on the vehicles.

(A1) All vehicles using the intersection are autonomous vehicles (AVs) and they are equipped with vehicle to vehicle and vehicle to infrastructure communication capabilities.

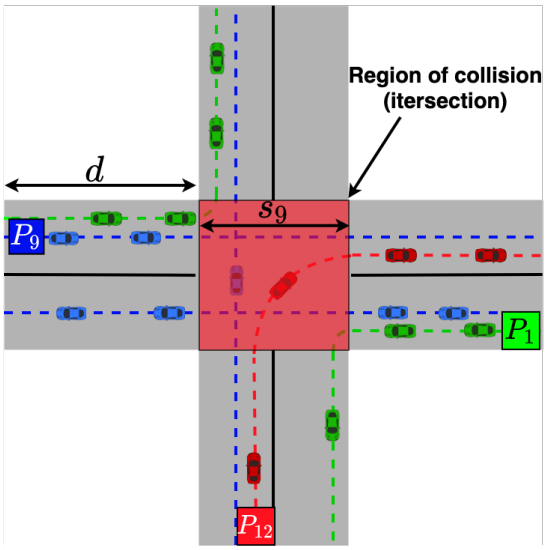


Fig. 1: This figure shows the basic geometry of the intersection and the region around it. The red translucent box represents the intersection. Each lane has a fixed path for the vehicles. For example, P_{12} denotes the path along lane 12. The parameter d is the length of each lane leading to the intersection. The length of the path P_k on lane k that passes through the intersection is s_k. In particular, the figure shows s_9.

(A2) All measurements of the state of the vehicles and the control inputs to AVs are perfect with no uncertainties. Similarly, all communication is perfect and without delay.

(A3) All vehicles are similar and are of the same length L.

(A4) Vehicles do not change lanes within the region of interest.

The AVs can enter the region of interest on any lane in $\mathcal{L}$. We denote the lane that vehicle i traverses on by $l_i \in \mathcal{L}$. Let the state of vehicle i at time t be $(x_i(t), v_i(t))$, where x_i and v_i are the position and the velocity respectively on the path P_{l_i}. Let u_i be the control input to the vehicle. We let the dynamics of the vehicle i on its lane l_i be

$$\dot{x}_i(t) = v_i(t), \quad \dot{v}_i(t) = u_i(t). \tag{1}$$

The vehicles are in the region of interest for different time durations. In particular, vehicle i enters the region of interest at the *arrival time* $t_i^{\mathcal{A}}$, that is $x_i(t_i^{\mathcal{A}}) = -d$. We call the time at which vehicle i enters the intersection as the *entry time* $t_i^{\mathcal{E}}$, that is $x_i(t_i^{\mathcal{E}}) = 0$; and the time of exit of the vehicle i from the intersection as the *exit time* $t_i^{\mathcal{X}}$, that is $x_i(t_i^{\mathcal{X}}) = s_{l_i}$.

B. Problem

We now pose the problem of intersection management. The aim is to compute the trajectories for the AVs so as to maximize the following objective function

$$J := \sum_{i \in V} \int_{t_i^{\mathcal{A}}}^{t_i^{\mathcal{A}}+T_h} \left[W_v\ v_i(t) - \left(W_a\ u_i^2(t) + W_j\ \dot{u}_i^2(t)\right)\right] \mathrm{d}t, \tag{2}$$

where $\dot{u}_i$ is the *jerk* of vehicle i and V is the set of all vehicles that arrive in the region of interest during a time interval of interest. Further, W_v, W_a and W_j are non-negative weights on velocity, acceleration and jerk terms, respectively, in the running objective. Each vehicle's contribution to the objective function is over a time horizon T_h, starting from the vehicle's arrival time $t_i^{\mathcal{A}}$. In the objective function (2), we consider the discomfort of the passengers in the vehicle i by the linear combination of the squares of acceleration and jerk. This metric of discomfort penalizes sporadic high-magnitude disturbances caused by braking and acceleration manoeuvres performed by i, as discussed in [20].

Remark 1 (Objective function)**.** In the objective function (2), the contribution of vehicle i is a linear combination of the distance traversed (integral of the velocity) and the comfort (negative of discomfort) experienced by the passengers of the vehicles. For comfort, we consider the acceleration and jerk only in the longitudinal direction along a vehicle's path. For vehicles turning left or right, we ignore the fact that forces also act in the lateral direction. However, the principles we illustrate in this paper could easily be extended to also consider 'lateral' comfort. We seek to maximize the social (over all vehicles) objective function of traversal distance and comfort. Maximizing traversal distance over a fixed horizon in each term of (2) is a proxy for minimizing traversal time for crossing the intersection. We choose this indirect metric because directly minimizing the traversal time results in a problem with a variable and unknown horizon for each vehicle. On the other hand, the fixed horizon formulation (2) provides a computational advantage. In particular, the construction of constraints online for a stream of vehicles is significantly simpler with a fixed horizon formulation. •

The constraints on the vehicles, over the horizon T_h, mainly involve the bounds on the velocity and acceleration and some constraints related to safety. We first discuss the bounds on the velocity and acceleration for each vehicle. Typically, vehicles have physical limitations that determine their acceleration and braking capabilities. Further, traffic laws impose an upper bound on the velocity at intersections. Thus, we assume that the constraints on acceleration and velocity of vehicle i are

$$u_i(t) \in [\underline{u},\ \bar{u}], \quad v_i(t) \in [\underline{v},\ \bar{v}], \tag{3}$$

for the time interval of interest. Here, $\underline{u}, \bar{u}, \underline{v}, \bar{v} \in \mathbb{R}$. $\underline{u}, \bar{u}$ are the minimum and maximum limits on acceleration and the minimum and maximum limits on velocity, respectively. Note that $\underline{u}$ captures the braking limitation, so $\underline{u} < 0$. In this paper, we let the lower bound on velocity $\underline{v}$ be 0.

We now discuss the constraints that ensure safety between the vehicles. We make a distinction between the type of collisions that can occur in the region of interest: in-lane collision (rear-ended collision) between successive cars on the same lane, and collision between cars on incompatible lanes, within the intersection. To ensure in-lane safety we impose a safe-following distance between any two successive vehicles travelling on the same lane. Consider two successive vehicles i and j on the same lane ($l_i = l_j$) such that i is the vehicle immediately following j. We use the *follower indicator function*,

$q(i,j)$, to denote this arrangement of vehicles. Formally,

$$q(i,j) := \begin{cases} 1, & \text{if } l_i = l_j,\ x_i < x_j, \\ & \nexists k \text{ s.t. } l_k = l_i,\ x_i < x_k < x_j \\ 0, & \text{otherwise.} \end{cases}$$

The minimum safe-following distance D between vehicles i and j, when $q(i,j) = 1$, is a function of the velocities of the two vehicles and is given by [19], [24]

$$D(v_i, v_j) = L + \max\left\{0, \frac{1}{-2\underline{u}}\left(v_i^2(t) - v_j^2(t)\right)\right\}.$$

Then the *rear-end safety constraint* is

$$x_j(t) - x_i(t) \geq D(v_i, v_j),\ j \text{ s.t. } q(i,j) = 1 \tag{4}$$

for the time interval of interest. Note that the rear-end safety constraint (4) is more robust to loss of coordination, either due to breakdown in communication, control or due to malicious vehicles, than mere rear-end non-collision constraints [24]. The second type of collisions are the ones that can occur when vehicles on incompatible lanes are simultaneously using the intersection. To ensure safety within the intersection, we impose the constraint that vehicles on incompatible lanes cannot be within the intersection simultaneously. Thus, the *intersection safety constraint* for a pair of vehicles i and k on incompatible lanes is

$$t_i^{\mathcal{E}} \geq t_k^{\mathcal{X}} \quad \textbf{OR} \quad t_k^{\mathcal{E}} \geq t_i^{\mathcal{X}}, \quad \text{if } c(l_i, l_k) = 0. \tag{5}$$

Then, the proposed optimal control problem for intersection management is as follows,

$$\max_{u_i(.),\ i \in V} J \tag{6a}$$

$$\text{s.t. } (1), (3), (4)\ \forall t \in [t_i^{\mathcal{A}}, t_i^{\mathcal{A}} + T_h], \forall i \in V \tag{6b}$$

$$(5)\ \forall i, k \in V \text{ s.t. } c(l_i, l_k) = 0. \tag{6c}$$

In this paper, we exercise no control over the vehicles after $t_i^{\mathcal{A}} + T_h$. In fact, we assume that the vehicles can proceed unhindered after crossing the intersection. We choose to do this purely for ease of exposition and to keep the focus of the paper on intersection management.

Remark 2 (Intersection safety constraints)**.** The constraints in (5) require that a vehicle i not occupy the intersection as long as a vehicle j on a lane incompatible with that of vehicle i is within the intersection. In general, this constraint may be unnecessarily strict. Instead, one can easily impose the constraint that vehicles i and j on incompatible lanes cannot simultaneously occupy a small conflict region around the point of intersection of the paths P_{l_i} and P_{l_j}. The algorithms we present in the sequel hold even for the refined constraints. In this paper, we present the simpler constraints (5) only for ease of exposition and simplicity of notation. •

Remark 3 (Challenges in solving Problem (6))**.** There are several challenges in solving Problem (6). First, vehicles arrive randomly in a stream into the system and the information about their arrival and state is revealed only incrementally. Thus, Problem (6) cannot be "solved" in the usual sense of the word. Hence, we seek an algorithm that satisfies the constraints in the problem and we utilize (2) as a metric for evaluating the performance of an algorithm after it makes all the decisions. Though the exact arrival times of the vehicles are not available a priori, we allow for the knowledge of the statistical data about the arrival times. In the simulations, we assume vehicles arrive into the region of interest according to a Poisson process, with additional safety constraints. Second, Problem (6) is a mix of large scale optimal control and combinatorial optimization. In particular, the number of optimal control subproblems that constraints (5) generate scales exponentially with the number of vehicles and lanes. This is a serious issue since intersection management is a time and safety critical problem. Hence, we seek algorithms that are computationally scalable and provide near optimal performance. •

III. Overview of Algorithm

Considering the complexities and time-criticality associated with Problem (6), we propose a computationally efficient algorithm to compute a schedule of intersection usage as well as the trajectories for the vehicles. To overcome the randomness in the arrival of traffic, and the challenges associated with incremental revelation of information, we split the trajectory of each vehicle into two phases: *provisional* and *coordinated*. In the provisional phase, which begins as soon as a vehicle arrives into the region of interest, the vehicle seeks to maximize its objective under the constraint of a safe approach towards the intersection. At a prescribed time, the vehicle switches to its coordinated phase from its provisional phase. The vehicles in their coordinated phase use the intersection safely while aiming to optimize the overall objective.

In this section, we give an overview of the proposed algorithm to solve Problem (6). For ease of exposition, we initially assume the presence of a central *intersection manager* (IM) that has communication and computation capabilities with which it carries out the coordination of the traffic. Subsequent to detailing the full algorithm, we discuss how essentially all the functions of the IM can be carried out in a distributed manner. We present the overview of the algorithm in two parts: from the perspectives of an arbitrary vehicle i and the IM in Algorithm 1, and in Algorithm 2, respectively.

A vehicle i starts execution of Algorithm 1 at $t_i^{\mathcal{A}}$, its time of arrival into the region of interest. In this algorithm, as soon as vehicle i arrives at $t_i^{\mathcal{A}}$, it communicates with the IM. The IM

Algorithm 1: Algorithm from a vehicle i's perspective

```
if t = t_i^A then
    receive t_i^C and trajectory of vehicle
     preceding i in its lane
    prov_phase(i)
    send provisional trajectory to IM
    start provisional phase
end
if t = t_i^C then
    receive new trajectory from IM for
     coordinated phase
    start coordinated phase
end
```

prescribes $t_i^{\mathcal{C}}$, the *start time of coordination phase* for vehicle i, and also informs about the planned trajectory of the vehicle (if any) that precedes vehicle i on its lane. This is sufficient for vehicle i to plan its trajectory for the provisional phase, which ends at $t_i^{\mathcal{C}}$. In particular, vehicle i computes its trajectory for the provisional phase by solving optimal control Problem (9), which we refer to in Algorithm 1 by `prov_phase`(i). Vehicle i communicates its trajectory for the provisional phase back to the IM and starts executing it at $t_i^{\mathcal{A}}$. At $t_i^{\mathcal{C}}$, vehicle i receives a new trajectory from the IM for the coordinated phase, which allows the vehicle to utilize the intersection safely.

Now, we describe Algorithm 2, which is from the IM's perspective. As soon as a vehicle i enters the region of interest,

Algorithm 2: Algorithm from IM's perspective

```
if t = t_i^A then
    t_i^C ← kT_c, with k = min{k ∈ N_0 : kT_c ≥ t_i^A}
    send to vehicle i, t_i^C and trajectory
     of vehicle preceding i in its lane
    receive vehicle i's provisional
     trajectory
end
if t = kT_c then
    V_c(k) ← {i : t_i^A ∈ ( (k−1)T_c, kT_c ]}
    coord_phase(V_c(k))
    send trajectories to vehicles V_c(k)
    k ← k + 1
end
```

the IM sends to vehicle i, the next instance of coordinated trajectory planning as $t_i^{\mathcal{C}}$ and the trajectory of the vehicle preceding i on its lane l_i. In this paper, for simplicity, we assume that IM carries out coordinated planning periodically with period $\mathcal{T}_c$. That is, the IM performs coordinated trajectory planning at the instances $k\mathcal{T}_c$, where $k \in \mathbb{N}_0$. At $t_i^{\mathcal{A}}$, the IM also sends to vehicle i the trajectory that is being executed by the vehicle preceding i in its lane. This information is sufficient for vehicle i to compute its provisional trajectory, which the IM receives from vehicle i.

At the coordinated trajectory planning time instance $k\mathcal{T}_c$, the IM first considers $V_c(k)$, the set all the vehicles that have arrived during the interval $\big(\ (k-1)\mathcal{T}_c,\ k\mathcal{T}_c\ \big]$. Then, it computes a trajectory for each vehicle in $V_c(k)$ that together achieve coordination and allow the vehicles to cross the intersection safely, while also seeking to optimize the objective in (6). We denote the problem of planning for the coordinated phase at the instance $k\mathcal{T}_c$ by `coord_phase`$(V_c(k))$. The IM communicates the trajectories for the coordinated phase to the vehicles in $V_c(k)$, which then execute them.

Remark 4 (Computation instances)**.** The instances of coordination planning need not be periodic and could adapt to the traffic. Choosing the "best" coordination time instances or even the period $\mathcal{T}_c$ of coordination planning is a non-trivial problem and is out of the scope of this paper. In Algorithm 1, and in Algorithm 2, we have presented various functions to be executed at specific time instances. However, this is purely for ease of exposition and one could easily modify these algorithms so that the execution of each step starts sufficiently before the time at which it is required to be finished. •

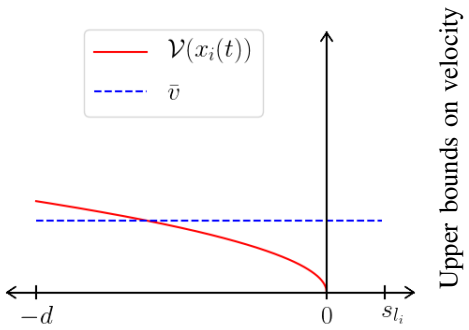


Fig. 2: The figure shows the upper bound on the velocity of vehicle i enforced by $\bar{v}$ and $\mathcal{V}(x_i(t))$ as a function of $x_i(t)$, the position of vehicle i along the path P_{l_i}. The horizontal axis also shows the arrival position $-d$, position at the beginning of the intersection, 0, and the position at the end of the intersection, s_{l_i}, on the path P_{l_i}.

Section IV details `prov_phase`(i), the algorithm for the computation of the trajectory for the provisional phase. In Section V, we present `coord_phase`$(V_c(k))$, the algorithm for planning the trajectories in the coordinated phase.

IV. Provisional Phase

In this section we describe `prov_phase`(i), the method that vehicle i utilizes to compute the trajectory for its provisional phase. As described in Algorithm 1, vehicle i communicates with the IM as soon as it arrives into the region of interest. The IM communicates $t_i^{\mathcal{C}}$, the start time of the coordination phase for vehicle i, and the trajectory of the vehicle preceding vehicle i on its lane. Using this information, vehicle i computes an optimal trajectory under a constraint that forces the vehicle to not enter the intersection apart from other constraints such as rear-end safety (4). To ensure the former requirement, we impose a position dependent upper bound, $\mathcal{V}(x_i(t))$, on the velocity $v_i(t)$ of the vehicle i. The upper bound $\mathcal{V}(x_i(t))$ is the maximum velocity that vehicle i may have at position $x_i(t)$ such that with maximum braking ($u_i(t) = \underline{u}$) vehicle i can come to a stop at most at the start of the intersection. In particular, the *intersection entry prevention constraint* is

$$v_i(t) \leq \mathcal{V}(x_i(t)) := \sqrt{2\underline{u}x_i(t)}, \tag{7}$$

for t in the time interval of interest. Notice that the upper bound $\mathcal{V}(0) = 0$, i.e., the upper bound on the velocity is zero at the beginning of the intersection. In this way, the constraint prevents the entry of the vehicle while also satisfying the bounds on the control at all times. Figure 2 depicts the position dependent upper bound $\mathcal{V}(x_i(t))$ along with the absolute upper bound $\bar{v}$ on the velocity of vehicle i.

The objective function for vehicle i's provisional phase is

$$J_i^p := \int_{t_i^{\mathcal{A}}}^{t_i^{\mathcal{A}}+T_p} \Big(W_v \; v_i(t) - \big[W_a \; u_i^2(t) + W_j \; \dot{u}_i^2(t) \big] \Big) \; \mathrm{d}t. \tag{8}$$

The time horizon for the provisional phase is T_p, which must be at least $t_i^{\mathcal{C}} - t_i^{\mathcal{A}}$. Then, the optimal control problem for the provisional phase of vehicle i is

$$\max_{u_i(.)} \; J_i^p \tag{9a}$$

$$\text{s.t. } (1), (3), (4), (7) \; \forall t \in [t_i^{\mathcal{A}}, t_i^{\mathcal{A}} + T_p]. \tag{9b}$$

Remark 5 (Computation of provisional phase trajectory)**.** To compute the rear-end safety constraint (4), vehicle i needs the trajectory of only vehicle k in front of it, i.e., the vehicle k such that $q(i,k) = 1$. With the trajectory of vehicle k, vehicle i is capable of computing the trajectory for the provisional phase locally. •

V. Coordinated Phase

This section presents the trajectory optimization for the coordinated phase, which implicitly also includes the schedule of the intersection usage for the vehicles. We propose two methods to obtain the schedule and perform trajectory optimization. The first method, which we call as the *combined optimization*, is a naive centralized method and is not computationally scalable. Taking combined optimization as the starting point, we propose a second method, the sequential weighted algorithm, that is significantly superior in terms of computational requirements. Moreover, as we demonstrate through simulations in Section VI, the sequential weighted algorithm performs almost as well as the computationally demanding approach of combined optimization.

In each of the two methods, the planning for the coordinated phase is carried out periodically with period $\mathcal{T}_c$. In particular, at the instance $k\mathcal{T}_c$, scheduling and trajectory planning is carried out for the vehicles that arrive into the region of interest during the interval $\big(\; (k-1)\mathcal{T}_c, \; k\mathcal{T}_c \;\big]$. Recall, from Step 7 of Algorithm 2, that this set of vehicles is denoted by $V_c(k)$. In this section, we discuss the methods for coordinated planning at an arbitrary but fixed instance $k\mathcal{T}_c$. We also introduce the set V_s that contains all the *vehicles that have received a trajectory for the coordinated phase*. The vehicles in $V_c(k)$ are added to V_s after they receive their respective trajectories for the coordinated phase. For brevity, we omit the argument k for $V_c(k)$ in the rest of this section. Further, notice that $t_i^{\mathcal{C}}$ is the same for all vehicles in $V_c(k)$. Hence, in the sequel, we drop the index i from $t_i^{\mathcal{C}}$.

A. Combined Optimization

This first approach of trajectory optimization for the coordinated phase is centralized in nature. In this method, the IM computes an optimal schedule and optimal trajectories of all the vehicles in V_c simultaneously. Hence the name *combined optimization*. After the IM computes the optimal trajectories for all the vehicles in V_c, the vehicles in V_c are added to V_s. The optimal control problem for the combined optimization method is a variation of the problem (6). The time horizon for the coordinated phase is T_c and the set of vehicles participating in the optimal control problem is V_c. The objective function and the optimal control problem for combined optimization is

$$J^c = \sum_{i \in V_c} \int_{t^{\mathcal{C}}}^{t^{\mathcal{C}}+T_c} \Big(W_v \; v_i(t) - \big[W_a \; u_i^2(t) + W_j \; \dot{u}_i^2(t) \big] \Big) \; \mathrm{d}t$$

$$\max_{u_i(.),\; i \in V_c} \; J^c \tag{10a}$$

$$\text{s.t. } (1), (3), (4) \; \forall t \in [t^{\mathcal{C}}, t^{\mathcal{C}} + T_c], \; \forall i \in V_c \tag{10b}$$

$$(5) \; \forall i, k \in V_c \cup V_s \text{ s.t. } c(l_i, l_k) = 0. \tag{10c}$$

Subsequent to solving (10) and updating the trajectories for the vehicles, V_s is updated to $V_s \cup V_c$.

Computation of the schedule and the trajectories for the coordinated phase in this manner requires the IM to compute optimal trajectories for each feasible order in which the vehicles could use the intersection and then pick the best schedule and the corresponding optimal trajectories that maximize the objective function. However, the number of feasible orders grows exponentially with the number of vehicles on incompatible lanes. Thus, the combined optimization method is not scalable and is not well suited for a time and safety critical application as autonomous intersection management. Hence, in the next subsection, we propose a computationally scalable and efficient method for computing near optimal schedules and trajectories for the coordinated phase.

Remark 6. Choosing the length of the time horizon T_c for the coordinated phase is a complex issue and is out of the scope of this paper. In this work, we choose a T_c that is long enough for every vehicle in V_c to exit the intersection.

B. Data Driven Sequential Weighted Algorithm (DD-SWA)

In this section, we propose a second method for optimizing the schedules and trajectories of vehicles in the coordinated phase. We call this method as the *data-driven sequential weighted algorithm* (DD-SWA). Note that a consequence of Assumption **(A4)** is that vehicles on the same lane do not overtake one another within the region of interest. This assumption leads to one of the main ideas behind DD-SWA: among the unscheduled vehicles, only those closest to the intersection on each lane are in contention for using the intersection first. Thus, if we have an efficient method to determine which of the unscheduled vehicles must go first and its trajectory, we may apply the method iteratively or sequentially to determine schedules and trajectories for the coordinated phase in a scalable manner. The second main idea is to use a data-driven approach for determining the schedule and the trajectories sequentially. We propose a scheme where one can use extensive offline simulations to arrive at an efficient online sequential method for determining near optimal schedules and trajectories. In fact, the method scales linearly with the number of vehicles. Moreover, as we discuss in

the sequel, this method is very amenable to a distributed implementation.

An overview of DD-SWA is presented in Algorithm 3. The algorithm begins with the set of unscheduled vehicles, V_c. Then the set of vehicles in V_c that are closest to the intersection, $\mathcal{F}$, is identified. Formally,

$$\mathcal{F} := \{i \in V_c \mid x_i \geq x_j,\ \forall j \in V_c \text{ s.t. } l_j = l_i\}.$$

In Step 8 of the algorithm, the *precedence index* p_i is computed

Algorithm 3: DD-SWA

1 **if** $t = 0$ **then**
2 $\quad V_s \leftarrow \emptyset$ {set of scheduled vehicles}
3 **end**
4 **if** $t = k\mathcal{T}_c$ **then**
5 $\quad$ **while** $|V_c| > 0$ **do**
6 $\quad\quad \mathcal{F} \leftarrow \{i \in V_c \mid x_i \geq x_j,\ \forall j \in V_c \text{ s.t. } l_j = l_i\}$
7 $\quad\quad$ **for** $i \in \mathcal{F}$ **do**
8 $\quad\quad\quad p_i \leftarrow$ precedence(i)
9 $\quad\quad$ **end**
10 $\quad\quad i^* \leftarrow \text{argmax}\{p_i \mid i \in \mathcal{F}\}$
11 $\quad\quad$ traj_opti(i^*)
12 $\quad\quad V_c \leftarrow V_c \setminus i^*$ {remove i^* from V_c }
$\quad\quad V_s \leftarrow V_s \cup i^*$ {i^* is scheduled}
13 $\quad$ **end**
14 **end**

for every vehicle i in $\mathcal{F}$. The precedence indices p_i serve as a metric for precedence of vehicle i to use the intersection. The precedence index p_i is computed based on various features that capture the "demand" of vehicle i itself and of the vehicles following it on lane l_i as well as the minimum time that vehicle i must wait before entering the intersection. The vehicle i^* with the highest precedence index p_{i^*} among the vehicles in $\mathcal{F}$ is given the highest precedence (after arbitrarily resolving any potential ties) to use the intersection before any other vehicle in $\mathcal{F}$. A trajectory for the coordinated phase is then computed for i^* in Step 11 and subsequently i^* is removed from V_c and added to V_s. Note that a single vehicle i^* is added to V_s after every iteration, unlike in combined optimization where all the vehicles in V_c were added to V_s at once. This whole process is carried out iteratively for as long as V_c is non-empty. The vehicles optimize their trajectories sequentially so as to satisfy the intersection safety constraint (5). Next, we describe the computation of the precedence indices precedence(i) and the trajectory optimization.

1) Computation of the Precedence Index precedence(i)*:* We let the precedence index p_i be a weighted linear combination of certain *scheduling features* related to the vehicle $i \in \mathcal{F}$ at the time of trajectory optimization for the coordinated phase. Table I lists the specific scheduling features that we utilize in this paper.

Three of the scheduling features are purely based on the state at time $t_i^{\mathcal{C}}$ and history of the vehicle i, namely, distance traveled since arrival $d + x_i(t_i^{\mathcal{C}})$, velocity $v(t_i^{\mathcal{C}})$ and time since arrival $(t_i^{\mathcal{C}} - t_i^{\mathcal{A}})$ of vehicle i. Three other features capture the "demand" on lane l_i that is "following" vehicle i. First of these features is the number of vehicles $|Q_i|$ that are following i on lane l_i at time $t_i^{\mathcal{C}}$, where Q_i is the *set of vehicles that follow vehicle i on lane l_i at time $t_i^{\mathcal{C}}$*. The second feature is the average separation of vehicles in Q_i from vehicle i, i.e.,

$$\frac{\sum_{j \in Q_i}(x_i(t_i^{\mathcal{C}}) - x_j(t_i^{\mathcal{C}}))}{|Q_i|}.$$

The third feature in this group is the average rate of arrival of vehicles σ_{l_i} on lane l_i. The final feature is the *minimum wait time to use the intersection*, τ_i, for vehicle i. Specifically,

$$\tau_i := \max\{t_m^{\mathcal{X}} - t_i^{\mathcal{C}} \mid m \in V_s \ s.t.\ c(l_i, l_m) = 0\}.$$

In the sequel, we penalize high minimum wait times τ_i in the precedence indices thus penalizing frequent switching of the right of way between incompatible lanes.

TABLE I: Scheduling Features

Feature	Weight
Distance traveled since arrival	w_x
Velocity	w_v
Time since arrival	w_t
No. of vehicles following vehicle i on lane l_i	w_n
Average separation of vehicles from vehicle i on lane l_i	w_s
Average arrival rate on lane l_i	w_σ
Minimum wait time to enter the intersection	w_w

We expect the relationship between the scheduling features and an optimal intersection usage order (or equivalently the precedence) to be quite nonlinear and complicated. There may also exist other features of the vehicles that determine the optimal precedence. A systematic study of such relationship and identifying the best choice of features is beyond the scope of this paper. Here, we define the precedence index p_i as a weighted linear combination of the chosen features,

$$p_i := w_x(d + x_i(t_i^{\mathcal{C}})) + w_v v_i(t_i^{\mathcal{C}}) + w_t(t_i^{\mathcal{C}} - t_i^{\mathcal{A}}) + w_n|Q_i| + w_s\frac{\sum_{j \in Q_i}(x_i(t_i^{\mathcal{C}}) - x_j(t_i^{\mathcal{C}}))}{|Q_i|} + w_\sigma\ \sigma_{l_i} - w_w\ \tau_i. \quad (11)$$

Note that all the weights are positive and thus we weigh all features positively except the wait time, which we weigh negatively. The weighted linear combination of the features makes the computation of the precedence indices extremely simple. However, even in this formulation, finding the best choice of the weights is non-trivial and beyond the scope of this paper. In this work, we propose tuning the weights based on offline simulations. Recall from Algorithm 3 that the vehicle i^* with the greatest precedence index p_i is selected for trajectory optimization.

2) Trajectory Optimization: Notice from the original optimal control problem (6) that the optimization of the trajectory of vehicle i is coupled to the optimization of the other vehicles' trajectories through constraints as well as the objective function (2). One of the purposes of the precedence indices is to set a precedence in the constraint (5). With the precedence set, we seek an optimization problem in which the objective function of vehicle i^* is not coupled with those of the other vehicles. Such a feature aids in developing a distributed implementation. A natural starting point for constructing such an objective function is to consider only the term involving i^* in J of (2).

However, this ignores the "demand" for the intersection usage. Thus, we seek to modify the "marginal" cost function of the vehicle i^* by incorporating a measure of the demand.

We first introduce a notion of *demand*, $\mathcal{D}_i$, from vehicle i and those following it on the lane l_i. Specifically,

$$\mathcal{D}_i := p_i + w_w \tau_i.$$

Then, we let the objective function for generating a trajectory for vehicle i^* to be

$$J^c_{i^*} = \int_{t^{\mathcal{C}}_{i^*}}^{t^{\mathcal{C}}_{i^*}+T_c} \Big(\overline{W}_v v_{i^*}(t) - \big[W_a u^2_{i^*}(t) + W_j \dot{u}^2_{i^*}(t) \big] \Big) \mathrm{d}t, \tag{12}$$

where $\overline{W}_v := w_l \frac{\sum_{i \in \mathcal{F}} \mathcal{D}_{i^*}}{|\mathcal{F}|} W_v$. Here, w_l is a scaling factor for the average demand. Then, `traj_opti`(i^*) in Step 11 of Algorithm 3 is

$$\max_{u_{i^*}(.)} \; J^c_{i^*} \tag{13a}$$

$$\text{s.t. } (1), (3), (4) \; \forall t \in [t^{\mathcal{C}}_{i^*}, t^{\mathcal{C}}_{i^*} + T_c] \tag{13b}$$

$$t^{\mathcal{E}}_{i^*} \geq \tau_{i^*} + t^{\mathcal{C}}_{i^*}, \tag{13c}$$

with $i = i^*$ in (1) , (3) and (4).

Remark 7 (Computational complexity of DD-SWA)**.** In DD-SWA, we obtain the intersection usage order by computing the precedence indices for vehicles in $\mathcal{F}$ as a weighted linear combination of the scheduling features and selecting the maximizer of the precedence indices. Also, note that $|\mathcal{F}| \leq |\mathcal{L}|$, the number of lanes. These aspects make the computation of the intersection usage order very simple. Further, the computation of the trajectory of the vehicles is also of significantly lesser complexity since for each vehicle we essentially need to solve an optimal control problem in which the only decision variables are those related to the vehicle itself and the constraints are significantly simplified. In the sequel, we use simulations to demonstrate that DD-SWA performs only marginally worse compared to the combined optimization while there is at least an order of magnitude decrease in the computation time. •

We see that in all the three optimal control problems (9), (10) and (13) feasibility implies safety. In the following result we show that if the vehicles arrive into the region of interest in a safe configuration then they are always in a safe configuration (both in-lane and within the intersection) for all time during the provisional as well as the coordinated phases.

Theorem 1 (Sufficient condition for system wide inter-vehicle safety)**.** *If every vehicle i satisfies the rear-end safety constraint (4) at the time of its arrival, $t^{\mathcal{A}}_i$, and its initial velocity is such that that $v_i(t^{\mathcal{A}}_i) \leq \min\{\bar{v}, \mathcal{V}(-d)\}$, feasiblility of problems (9), (10) and (13) is guaranteed. Consequently, safety of all the vehicles is also guaranteed for all time.*

Proof. We have assumed that each vehicle i satisfies the rear-end safety constraint (4) at $t^{\mathcal{A}}_i$, which ensures that there exists a control trajectory to ensure rear-end safety with the vehicle that precedes i on its lane l_i. Further, the assumption that $v_i(t^{\mathcal{A}}_i) \leq \min\{\bar{v}, \mathcal{V}(-d)\}$ ensures that there exists a control trajectory that additionally guarantees that vehicle i has a feasible control trajectory that ensures that the vehicle can come to a stop before the beginning of the intersection. Thus, the optimization problem for the provisional phase (9) is feasible.

If problem (9) is feasible, the trajectory for the provisional phase guarantees that vehicle i satisfies the rear-end safety constraint (4) and the intersection entry prevention constraint (7) at $t^{\mathcal{C}}_i$, the start time of the coordinated phase of vehicle i. This property ensures that the intersection safety constraints (10c) and (13c) for combined optimization and DD-SWA respectively are also feasible as the vehicle can come to a stop before the beginning of the intersection if necessary. Thus, feasibility of problem (9) guarantees the feasibility for problems (10) and (13). Since feasiblility of the problems ensure rear-end safety and intersection safety, safety of all vehicles is also guaranteed. □

TABLE II: Modes of obtaining information necessary for DD-SWA

Features and constraints	**Method**
Distance travelled since arrival	Local information
Velocity	Local information
Time since arrival	Local information
Average arrival rate in l_i	Communication with IM
Weights on Scheduling Features $(w_x, \ldots, w_w)$	Communication with IM
No. of vehicles following i in l_i	Intra-lane communication
Average separation of vehicles from i in l_i	Intra-lane communication
Rear-end safety constraints	Intra-lane communication
Minimum wait time to use the intersection	Inter-lane communication

C. Implementation of DD-SWA

As mentioned in Section V-B, the design of DD-SWA is amenable to a distributed implementation. The information required to calculate a vehicle's precedence index and to solve its trajectory optimization problem can be obtained with distributed communication and local computation. Table II lists the methods that the vehicles in $\mathcal{F}$ can utilize to acquire the information necessary to compute their precedence indices and optimize their trajectories.

Each vehicle can obtain information such as distance travelled, velocity and time since arrival locally or through technology such as GPS or odometry or some other infrastructure for localization. The other scheduling features, the safety constraints (4) and (5) and the weights for the scheduling features require communication. We make a distinction between the types of communication required for this purpose. The three types of communication required are: (1) intra-lane, (2) inter-lane and (3) central communication. We elaborate on each mode of communication in greater detail. In intra-lane communication, a vehicle $i \in \mathcal{F}$ needs to communicate with only a vehicle j such that $q(i,j) = 1$ or $q(j,i) = 1$, *i.e.*, i needs to communicate with just the vehicles immediately preceding or following it on its lane l_i. The number of vehicles following i, $|Q_i|$, can be counted in a distributed manner and can be communicated from one car to the next in Q_i, the vehicles following i and ultimately to the vehicle i itself. Intra-lane communication can be employed to compute the rear-end safety constraints (4) as well. The vehicle immediately in front

of i^* can communicate its position and velocity trajectory which is sufficient to compute (4). The intersection safety constraint and the minimum wait time feature require i^* to communicate and receive the exit time of the vehicle on an incompatible lane that has the greatest exit time. We denote such communication as inter-lane communication. Apart from communication between vehicles, intersection-specific information such as the weights for the scheduling features $w_x, \ldots, w_s$ and the average arrival rate of traffic σ_{l_i} need to be communicated to the vehicles in $\mathcal{F}$ from a central infrastructure, such as an IM. Thus the central infrastructure's or IM's function is essentially restricted to only communication. With reduced burden on the central infrastructure, the IM need not host major computational capabilities unlike in the case of a centralized algorithm.

VI. Simulations

To evaluate the proposed algorithm with combined optimization and DD-SWA, a simulation framework using Casadi [25] and Python was created. All the simulations were performed on an Intel i7-9700 3.6GHz processor with 32GB of RAM. Next, the simulations are discussed in detail.

Arrival of Vehicles: Recall that the existence of feasible trajectories for the provisional and coordinated phases is guaranteed if the conditions mentioned in Theorem 1 are satisfied. Thus to ensure feasibility, we restrict the maximum velocity of the vehicles at the time of their arrival, to $\min\{\bar{v}, \mathcal{V}(-d)\}$. In the simulations here, we assume that vehicles arrive in each lane according to a Poisson process with an average arrival rate of σ_l for a lane $l \in \mathcal{L}$. However, a specific realization of arrival times of the vehicles may cause a violation of the rear-end safety constraint at the arrival time itself. To avoid this, we check the separation between the successive vehicles at the time of arrival. If the constraint (4) is violated, the arrival of the vehicle is delayed until the constraint is satisfied.

To evaluate the proposed algorithm, we compare combined optimization and DD-SWA against two traditional methods of intersection management: a signalized intersection and a stop-sign based intersection. We also make a comparison against the Hierarchical-Distributed algorithm proposed in [19]. The simulations results that are presented here are for the particular case of vehicles only passing straight across the intersection, *i.e.* vehicles arrive in lanes $l \in \mathcal{L}_s = \{2, 5, 8, 11\}$. However, the proposed algorithms hold even when turning is allowed. We present the algorithms and the comparisons in greater detail below.

Signalized Intersection: In this algorithm, every vehicle i that enters the region of interest performs `prov_phase`(i) to approach the intersection. When a lane l receives a green signal, all the vehicles in l are considered to be a part of V_c and they are given a *green trajectory* to exit the intersection by solving problem (10). If a vehicle i cannot exit the intersection before the end of the green time using the green trajectory, the vehicle performs `prov_phase`(i) instead. We consider 2 phases per cycle for the traffic signals. The cycle times and green times for the signals are obtained using Webster's method [26] corresponding to the arrival rate σ_l in each lane. The yellow time in each phase has been set to 0. Thus, this algorithm assumes autonomous vehicles with communication capabilities, which is well beyond today's state of the art.

Stop-Sign Based Intersection: As the name suggests, this algorithm mimics the traffic rules at an intersection with stop-signs. The vehicles come to a complete stop at the beginning of the intersection before they compute a trajectory to cross the intersection. Every vehicle i performs `prov_phase`(i) to approach the intersection. When a vehicle i is at the beginning of the intersection, *i.e.* $x_i(t) = 0$, and has the right-off-way to cross the intersection, vehicle i computes a trajectory to exit the intersection by solving problem (10). Once i computes its trajectory to use the intersection, every vehicle j following i in l_i recomputes `prov_phase`(j) sequentially starting from the vehicle immediately following i. The vehicles get the right-of-way to use the intersection in the same order as they arrive at the intersection. If two vehicles on different lanes arrive at the beginning of the intersection at the same time, the vehicle with the lower lane-number l is given the right-of-way to use the intersection.

Hierarchical-Distributed (HD) Algorithm: This is the algorithm presented in [19]. The HD algorithm was implemented on Matlab on an i7-9700 processor with 32GB of RAM.

Simulation Parameters: Table III lists the parameters of the model of the intersection and the vehicles that were common to all the algorithms. For the signalized and stop-sign based algorithms, we set the time horizon for the provisional phase $T_p = 20$s. For the proposed algorithm involving combined optimization and DD-SWA, we set the time period of the coordination phase computation instances $\mathcal{T}_c = 3$s, the time horizon for the provisional phase $T_p = t_i^{\mathcal{C}} - t_i^{\mathcal{A}}$ and the time horizon for the coordinated phase $T_c = 30$s. We conducted simulations for all the comparisons for several arrival rates of traffic. We chose the simulation time for each simulation to be equal to the time duration of 10 cycles of a signalized intersection corresponding to the particular arrival rate σ. Recall that we chose the cycle time for the signalized intersection for each arrival rate σ according to the Webster's method [26]. In each of the simulations we discuss here, we conducted 20 trials for each of the algorithms for each arrival rate σ. Then, we compared the average time to cross and average objective function value per vehicle over the 20 trials for each value of σ across all the algorithms.

TABLE III: General Simulation Parameters

Intersection Parameters		
Parameter	**Symbol**	**Value**
Length of branch	d	60 m
Length of intersection (Straight)	s_l	20 m
Length of vehicle	L	3.5 m
Min. Acceleration	$\underline{u}$	-3 m/s^2
Max. Acceleration	$\bar{u}$	3 m/s^2
Max. Velocity	$\bar{v}$	11.11 m/s
Proposed Algorithm Parameters		
Time interval for coordinated phase	$\mathcal{T}_c$	3 s
Time horizon for provisional phase	T_p	$t_i^{\mathcal{C}} - t_i^{\mathcal{A}}$
Time horizon for coordinated phase	T_c	30 s
Time horizon for objective function (2)	T_h	30 s

A. Results

We present 4 sets of comparisons between the various algorithms mentioned previously. Table IV lists the weights in the objective functions (2), (10) and (13) for each comparison. It also indicates the weights on the scheduling features used in DD-SWA in each of the comparative simulations. Next, we discuss in detail each of the four comparisons.

1) Comparison 1: In this comparison, there is no weight on comfort of the passengers in the objective functions and the vehicles aim to only maximize the distance travelled. In particular, in the "running cost", the weights on acceleration and jerk terms (W_a and W_j respectively) are set to 0 and the weight on velocity term (W_v) is set to 1 for trajectory optimization problems for the provisional phase and the coordinated phase. In the HD algorithm, the fuel cost represented by $F_i(\bar{v}_i)$ in Equation 5 of [19] is set to 0 so that the vehicles only aim to minimize the time spent within the intersection. This ensures a fair comparison between the HD algorithm and other algorithms. We compare the average time to cross (TTC) for the vehicles under the different algorithms. We show simulation results for arrival rates (σ) in the range of 0.01 to 0.09 vehicles/s per lane with an increment of 0.01 vehicles/s per lane. Figure 3(a) shows that the average TTC for vehicles with combined optimization and DD-SWA is comparable for all arrival rates in the considered range. The HD algorithm's performance is comparable for arrival rates between 0.01 and 0.04 vehicles/s per lane but performs poorly beyond 0.05 vehicles/s. The signalized algorithm performs better than the HD algorithm at 0.09 vehicles/s. The stop-sign algorithm performs very poorly compared to the other algorithms.

2) Comparison 2: In Comparison 2, the objective is again to maximize only the distance travelled by the vehicles, but the comparison is made for arrival rates (σ) from 0.1 to 0.9 vehicles/s per lane. As the computation time for combined optimization is significantly higher compared to the other algorithms, we choose to not include it for comparison 2. Figure 3(b) shows that DD-SWA continues to perform better than all the other algorithms. Although the time to cross initially increases for DD-SWA, it saturates at 0.4 vehicles/s per lane. The signalized algorithm outperforms the HD algorithm but does not perform better than the DD-SWA. The HD algorithm and stop-sign based intersection performs significantly worse than the other algorithms.

3) Comparison 3: In Comparison 3, we compare between combined optimization and DD-SWA when the arrival of traffic is inhomogeneous, *i.e.*, when the arrival rates are not the same for all the lanes in consideration. In particular, we set $\sigma_2 = \sigma_8 = \sigma$ and $\sigma_5 = \sigma_{11} = \frac{\sigma}{2}$ for different values of σ. We again set the weights on acceleration and jerk terms to 0 and the weight on the velocity term to 1 in this comparison. Figure 3(c) shows the average time to cross for the vehicles for combined optimization and DD-SWA. The performance of DD-SWA is only marginally poor compared to that of combined optimization. Figure 3(d) shows the average objective value (2) over a period of t_i^A to $t_i^A + T_h$ for every vehicle. To compute the objective value, only the vehicles that crossed the intersection by the end of the simulation time were considered to be in the set V in the objective function (2). The average objective value per vehicle is depicted in 3(d). Note that the weights on the scheduling features in DD-SWA were tuned to improve its performance.

4) Comparison 4: Combined optimization is compared against DD-SWA with a non-zero weight on comfort. The weights on the velocity, acceleration and jerk terms are all set to 1. Figures 4(a) and (b) depict the average TTC and the average objective value for the two methods from t_i^A to $t_i^A + T_h$. A decrease in the average objective value and an increase in the average TTC can be observed as there is an emphasis on both comfort and the distance travelled by the vehicles. It can be observed that combined optimization outperforms DD-SWA both in terms of the average objective value and the average TTC. Similar to the legend in Figure 3, the blue and red bars in Figure 4 correspond to DD-SWA and combined optimization respectively.

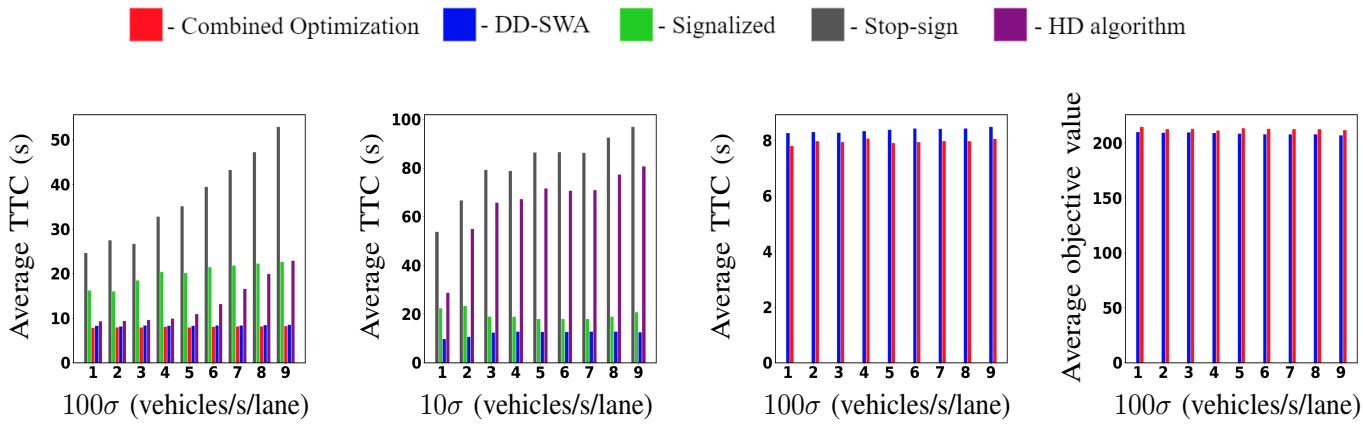
5) Comparison 5: In this comparison, the weight on both acceleration and jerk terms is set to 10 and the weight on the velocity term is set to 1. Figures 4(c) and (d) show a significant decrease in the average objective value and an increase in the average TTC for the two methods. It can be observed that combined optimization outperforms DD-SWA in terms of the average objective value but there is a significant increase in the average TTC for combined optimization. DD-SWA outperforms combined optimization in terms of the average TTC. Note that the performance of DD-SWA is heavily dependent on the weights on the scheduling features. Hence, it is possible that one may achieve better performance for the algorithm with further tuning of the weights on the scheduling features.

Computation time comparison: We make a comparison of the computation time per vehicle for combined optimization and DD-SWA to emphasize the computational advantage of DD-SWA. We initially compare the size of V_c for every round of trajectory optimization for the coordinated phase. To inspect the variation in $|V_c|$, we use box plots to vizualize the data. In the Figure 5, The lower and the upper edges of the boxes represent the first quartile ($Q1$, 25^{th} percentile) and the third quartile ($Q3$, 75^{th} percentile) respectively. The whiskers above and below the boxes represent the maximum and the minimum of the data. The maximum is calculated as $Q3+1.5(Q3-Q1)$ and the minimum as $Q1 - 1.5(Q3 - Q1)$. Data beyond the maximum and the minimum are considered as outliers and they are represented by small circles. The mean of the data is represented by the bold black line. In Figures 5(a) and (b), the box plots represent $|V_c|$ for combined optimization and DD-SWA respectively and in Figures 5 (c) and (d), we compare the computation time per vehicle for combined optimization and DD-SWA for increasing arrival rates. Although the trend of $|V_c|$ is similar for both the algorithms, the trend of computation time per vehicle is significantly different. The computation time for combined optimization increases exponentially as $|V_c|$ increases with the arrival rate. In Figure 5(d), we can see that DD-SWA has a nearly constant value of computation time per vehicle. This can be attributed to its its structure of sequentially optimizating the trajectory of the vehicles in V_c.

Saturation of arrival rate: In Comparison 2, at high arrival rates of traffic between 0.1 and 0.9 vehicles/s per lane, the true

TABLE IV: Weights for comparisons

General Weights					
Parameter	**Symbol**	**Comparisons 1 and 2**	**Comparison 3**	**Comparison 4**	**Comparison 5**
Weight on acceleration term	W_a	0	0	1	10
Weight on jerk term	W_j	0	0	1	10
Weight on velocity term	W_v	1	1	1	1
DD-SWA Weights					
Scheduling Features	**Symbol**	**Comparison 1 and 2**	**Comparison 3**	**Comparison 4**	**Comparison 5**
Distance travelled since arrival	w_x	0.1	0.4	0.2	0.8
Velocity	w_v	5	4	6	6
No. of vehicles following i in l_i	w_n	4	2	5	7
Time since arrival	w_t	3	4.5	3	3
Average arrival rate in l_i	w_σ	40	60	40	50
Average separation of vehicles from i in l_i	w_s	5	6	3	3
Minimum wait time to use the intersection	w_w	0.5	1	1	4
Average demand scaling factor	w_l	0.02	0.02	0.02	0.02



(a) (b) (c) (d)

Fig. 3: Results of Comparisons 1, 2 and 3 for various arrival rates σ. In Comparisons 1 and 2, the arrival rate is homogeneous across all lanes. In Comparison 3, the arrival rate is inhomogeneous. (a) Comparison 1 - the average time to cross (TTC) for low arrival rates. (b) Comparison 2 - the average time to cross (TTC) for high arrival rates. Results for combined optimization algorithm are excluded here because of very high computation times. (c) Comparison 3 - the average time to cross(TTC) for the vehicles. (d) Comparison 3 - the average objective value, which is the average distance traversed by the vehicles from the time of their arrival.

arrival rate of the vehicles is potentially reduced due to the feasibility conditions mentioned in Theorem 1. As mentioned earlier, if the rear-end safety constraint is violated at the time of arrival of a vehicle, the arrival time is delayed until the constraint is satisfied. At high arrival rates, the rear-end safety constraint is violated often. If necessary, the actual arrival of the vehicle is delayed until the constraint is satisfied. In Figure 6, we present this idea by plotting the true arrival rate per lane versus the set arrival rate per lane for DD-SWA. We make use of the box plot to capture the variation of the true arrival rate on the y-axis. This plot illustrates that the true arrival rate saturates beyond 0.4 vehicles/s per lane. In Figure 3(b), the average TTC of vehicles in DD-SWA also saturates at 0.4 vehicles/s per lane which is consistent with the saturation of the true arrival rate.

Remark 8. Numerical computation for combined optimization can potentially be parallelized as the trajectory optimization problem can be solved in parallel for each feasible intersection usage order. However, DD-SWA is a distributed algorithm. In DD-SWA, the only operation that truly requires information about the whole traffic state is the computation of the precedence indices and the intersection usage order or precedence. These operations have only a marginal contribution to the computation time of trajectories in DD-SWA. Most of the computation time is in fact taken by the optimization of the trajectories of the vehicles given the precedence indices. However, this operation can be carried out by each vehicle with only local communication. This property of the DD-SWA algorithm, wherein the schedule is determined based on data using computationally efficient online operations and local computation of the trajectories has another very significant advantage. It is that the computational complexity of the online part of DD-SWA is essentially unaffected even if we consider AVs with significantly more complicated dynamics than (1). This is surely not the case with full optimization based algorithms such as combined optimization.

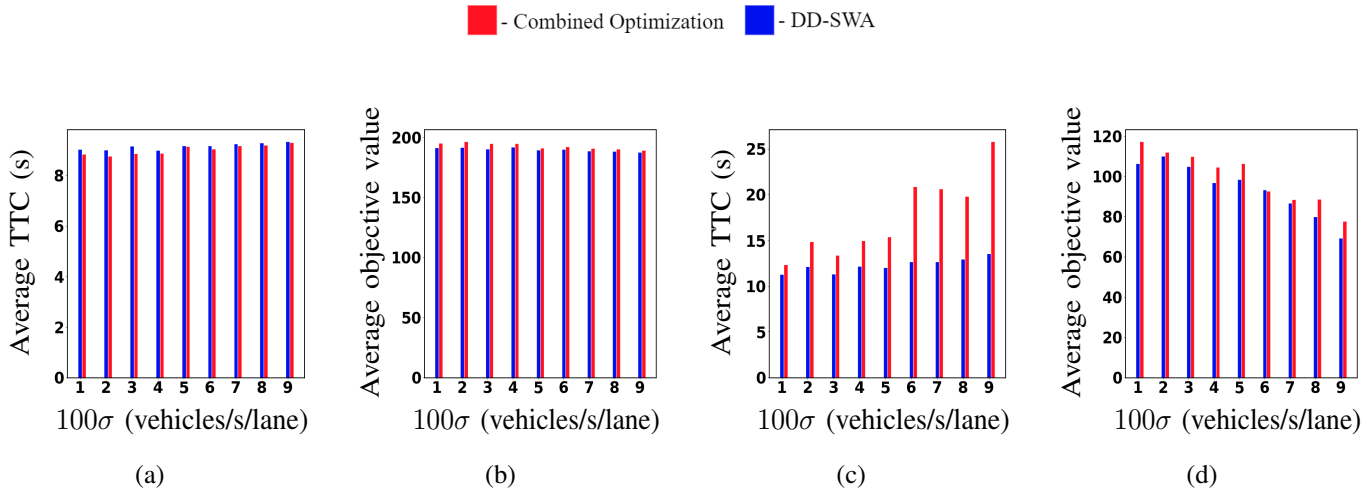


Fig. 4: Results of Comparisons 4, in (a) and (b), and Comparison 5, in (c) and (d). In comparison 4, the weights on the velocity, acceleration and jerk terms are equal to 1 and in comparison 5, the weight on velocity term is 1 while the weights on acceleration and jerk terms are equal to 10.

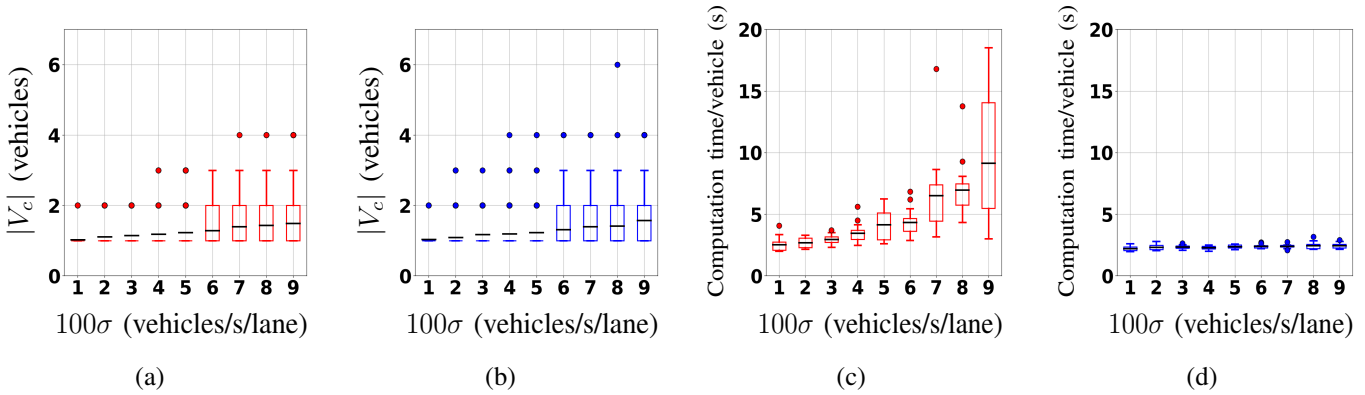


Fig. 5: Results of the computation time comparison. Figures (a) and (c) correspond to combined optimization and (b) and (d) correspond to DD-SWA. In (a) and (b), the box plots represent the number of vehicles that participate in the trajectory optimization problem for the coordinated phase, which is denoted by $|V_c|$. In (c) and (d), computation time per vehicle is compared for combined optimization and DD-SWA. The lower and upper edge of the boxes represent the first and third quartile of the data respectively. The minimum and maximum of the data is represented by whiskers beyond the edges of the boxes. The outliers of the data are represented by circles beyond the whiskers.

VII. Conclusion

In this work, we have introduced a provably safe data-driven algorithm that aims to minimize the travel times of vehicles and the discomfort of the passengers. By appropriately decomposing the general problem of intersection management into two phases, we have ensured safety and feasibility for the control of vehicles. Our data-driven algorithm DD-SWA relies on offline computation to tune the online algorithms for scheduling and trajectory optimization. This architecture of using offline tuning of the online algorithm significantly reduces the online computation effort, as demonstrated by the simulations. We have shown that DD-SWA takes significantly less computational effort compared to the centralized implementation with only marginal loss in the objective value. We have also shown that DD-SWA can be implemented in a distributed manner with vehicle to vehicle and vehicle to infrastructure communication. Further, simulations suggest that the proposed algorithm performs significantly better than traditional algorithms such as signalized and stop-sign based intersection management even for very high traffic arrival rates. This is again made possible by the data-driven approach of our algorithm, which allows us to seamlessly incorporate "micro" information about individual vehicles as well as "macro" statistical information about the traffic arrival rates in the intersection management system.

The general performance of DD-SWA for intersection management depends on appropriately tuned parameters for decision making. In this work, we have used extensive offline simulations to improve the performance of DD-SWA. Future work can be focused on developing learning-based methodologies to automate the tuning of the parameters in DD-SWA

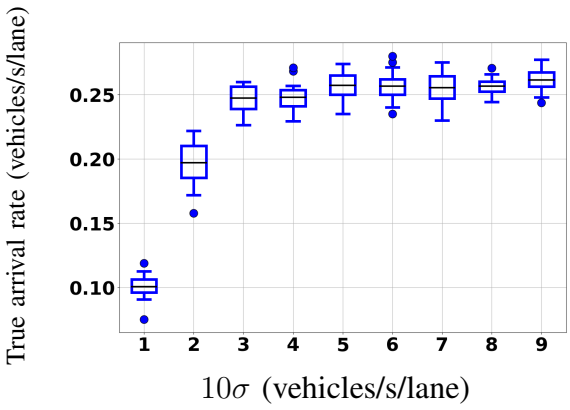


Fig. 6: The true arrival rate versus the desired σ recorded for DD-SWA in Comparison 2. The upper and lower edges of the boxes represent the third and first quartile respectively. The whiskers of the boxes represent the maximum and minimum of the data. The circles beyond the whiskers are the outliers.

for various traffic scenarios. Our framework is very efficient computationally and is capable of incorporating diverse kind of features in the schedule and trajectory optimization. Due to this, we believe that another promising direction for future research is extension of our framework to network wide traffic management with multiple intersections. Computational efficiency of DD-SWA algorithm also suggests that it is worthy to explore hardware implementation on multi-robot systems in regulated environments.

REFERENCES

[1] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *Journal of artificial intelligence research*, vol. 31, pp. 591–656, 2008.

[2] D. Fajardo, T.-C. Au, S. T. Waller, P. Stone, and D. Yang, "Automated intersection control: Performance of future innovation versus current traffic signal control," *Transportation Research Record*, vol. 2259, no. 1, pp. 223–232, 2011.

[3] M. Hausknecht, T.-C. Au, and P. Stone, "Autonomous intersection management: Multi-intersection optimization," in *IEEE International Conference on Intelligent Robots and Systems*, 09 2011, pp. 4581–4586.

[4] D. Carlino, S. D. Boyles, and P. Stone, "Auction-based autonomous intersection management," in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 10 2013, pp. 529–534.

[5] H. Kowshik, D. Caveney, and P. Kumar, "Provable systemwide safety in intelligent intersections," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 3, pp. 804–818, 2011.

[6] A. Colombo and D. Del Vecchio, "Least restrictive supervisors for intersection collision avoidance: A scheduling approach," *IEEE Transactions on Automatic Control*, vol. 60, no. 6, pp. 1515–1527, 2015.

[7] 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.

[8] M. W. Levin and D. Rey, "Conflict-point formulation of intersection control for autonomous vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 85, pp. 528 – 547, 2017.

[9] F. Altché and A. de La Fortelle, "Analysis of optimal solutions to robot coordination problems to improve autonomous intersection management policies," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016, pp. 86–91.

[10] S. A. Fayazi and A. Vahidi, "Mixed-integer linear programming for optimal scheduling of autonomous vehicle intersection crossing," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 3, pp. 287–299, 2018.

[11] A. I. Morales Medina, F. Creemers, E. Lefeber, and N. van de Wouw, "Optimal access management for cooperative intersection control," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2019.

[12] A. I. Morales Medina, N. van de Wouw, and H. Nijmeijer, "Cooperative intersection control based on virtual platooning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 6, pp. 1727–1740, 2018.

[13] A. A. Malikopoulos, C. G. Cassandras, and Y. J. Zhang, "A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections," *Automatica*, vol. 93, pp. 244–256, 2018.

[14] Y. Zhang and C. G. Cassandras, "Decentralized optimal control of connected automated vehicles at signal-free intersections including comfort-constrained turns and safety guarantees," *Automatica*, vol. 109, p. 108563, 2019.

[15] G. R. de Campos, P. Falcone, R. Hult, H. Wymeersch, and J. Sjöberg, "Traffic coordination at road intersections: Autonomous decision-making algorithms using model-based heuristics," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 8–21, 2017.

[16] C. Liu, C.-W. Lin, S. Shiraishi, and M. Tomizuka, "Distributed conflict resolution for connected autonomous vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 1, pp. 18–29, 2017.

[17] D. Miculescu and S. Karaman, "Polling-systems-based autonomous vehicle coordination in traffic intersections with no traffic signals," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 680–694, 2020.

[18] R. Hult, M. Zanon, S. Gros, and P. Falcone, "Optimal coordination of automated vehicles at intersections: Theory and experiments," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 6, pp. 2510–2525, 2018.

[19] P. Tallapragada and J. Cortés, "Hierarchical-distributed optimized coordination of intersection traffic," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[20] M. Elbanhawi, M. Simic, and R. Jazar, "In the passenger seat: investigating ride comfort measures in autonomous cars," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 3, pp. 4–17, 2015.

[21] I. A. Ntousakis, I. K. Nikolos, and M. Papageorgiou, "Optimal vehicle trajectory planning in the context of cooperative merging on highways," *Transportation research part C: emerging technologies*, vol. 71, pp. 464–488, 2016.

[22] P. Dai, K. Liu, Q. Zhuge, E. H. . Sha, V. C. S. Lee, and S. H. Son, "Quality-of-experience-oriented autonomous intersection control in vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 1956–1967, 2016.

[23] R. Hult, M. Zanon, S. Gros, and P. Falcone, "Optimal coordination of automated vehicles at intersections with turns," in *2019 18th European Control Conference (ECC)*, 2019, pp. 225–230.

[24] P. Tallapragada and J. Cortés, "Distributed control of vehicle strings under finite-time and safety specifications," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1399–1411, 2018.

[25] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, In Press, 2018.

[26] T. Urbanik, A. Tanaka, B. Lozner, E. Lindstrom, K. Lee, S. Quayle, S. Beaird, S. Tsoi, P. Ryus, D. Gettman, S. Sunkari, K. Balke, and D. Bullock, *Signal Timing Manual - Second Edition*. Washington, DC: The National Academies Press, 2015.