Data-Guided Distributed Intersection Management for Connected and Automated Vehicles

Darshan Gadginmath Pavankumar Tallapragada

***Abstract*— In this paper, we seek a scalable method for safe and efficient coordination of a continual stream of connected and automated vehicles at an intersection without signal lights. To handle a continual stream of vehicles, we propose trajectory computation in two phases - in the first phase, vehicles are constrained to not enter the intersection; and in the second phase multiple vehicles' trajectories are planned for coordinated use of the intersection. For computational scalability, we propose a data-guided method to obtain the intersection usage sequence through an online "classification" and obtain the vehicles' trajectories sequentially. We show that the proposed algorithm is provably safe and can be implemented in a distributed manner. We compare the proposed algorithm against traditional methods of intersection management and against some existing literature through simulations. We also demonstrate through simulations that for the proposed algorithm, the computation time per vehicle remains constant over a wide range of traffic arrival rates.**

***Index Terms*— Autonomous intersection management, connected and automated vehicles, distributed control, data-guided control, optimized and provably safe operation**

I. Introduction

Traffic composed of Connected and Automated Vehicles (CAVs) or networked robots (such as in warehouses) can be managed using communication and coordination algorithms to achieve better efficiency and safety than traditional traffic signal based management. In this work, we propose a computationally scalable data-driven distributed algorithm for the management of an isolated intersection in the context of a continual stream traffic of CAVs or robots.

Some early works on this problem, such as [1], focused on reservation and multi-agent simulation based algorithms. These solutions are computationally demanding and centralized. A major trend in more recent literature has been to design model based, provably safe algorithms. [2] uses reservations for scheduling intersection usage times, and [3] and the references therein propose a supervisory control method where a supervisor takes over only when a collision is imminent. [4] formulates an optimal control problem to optimize the trajectory of each vehicle individually. However, intersection management problem requires coordination of multiple vehicles through a combined optimization that is combinatorial. Thus, the overall problem of trajectory optimization becomes a mixed integer program [5], [6]. The complexity of such formulations scales exponentially with the number of vehicles. For this reason, several works have sought to decompose the overall autonomous intersection management problem into simpler sub-problems to overcome the computational complexity. [7] propose a high level intersection access management by treating the vehicles on different lanes as queues and platooning vehicles for local control. Given the intersection usage schedule for the vehicles, [8] (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. [9]–[12] also decompose the problem into scheduling and trajectory optimization. [12] 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. [13] proposes to achieve coordination of vehicles by optimizing a notion of joint rewards for the vehicles. For this, it employs Q-learning based on episodic data, to minimize the average intersection delay. Thus, the resulting trajectories can turn out to be non-smooth. Although the vehicles can obtain near-optimal joint actions, the paper does not provide safety guarantees.

Comfort of passengers and generation of smooth trajectories for vehicles is also important in the context of CAVs. [14] surveys driver comfort in autonomous vehicles and highlights the inadequate research on the topic. [15] introduces a metric of comfort which is a combination of vehicle-jitter, jerk and deviation from a desired velocity.

Contributions: We propose a computationally scalable algorithm for coordinating and optimizing the trajectories of a continual stream of CAVs at and near an isolated, unsignalized autonomous intersection. Most of the papers in the literature consider only the problem of coordinating a fixed set of vehicles. Applying such solutions to a continual stream of vehicles can result in inefficiency or feasibility/safety itself may be violated in the long run. Further, the optimal coordination of vehicles at an intersection is a mixed integer problem, which scales badly with the number of vehicles and lanes. This work is the only paper, to the best of our knowledge, that addresses these issues systematically.

Firstly, we deal with a continual stream of vehicles by splitting the trajectory of the vehicles into two phases 1) *provisional phase* and (2) *coordinated phase*. Every vehicle operates in the provisional phase as soon as it enters the system and it is restricted from entering the intersection. Periodically, the vehicles in the provisional phase obtain a

This work was partially supported by the Wipro IISc Research and Innovation Network. Darshan Gadginmath (dgadg001@ucr.edu) is with the Department of Mechanical Engineering, University of California, Riverside. Pavankumar Tallapragada (pavant@iisc.ac.in) is with the Department of Electrical Engineering, Indian Institute of Science, Bangalore, India.

trajectory for their coordinated phase and start executing them. Secondly, for the coordinated phase, we propose a data-driven framework that uses an online "classification" from a space of features that encode the state of the traffic to schedule vehicles to use the intersection. Then, we compute the trajectories of the vehicles for the coordinated phase sequentially. [9] proposes the use of different features to establish a sequence of interesction usage. However, the best set of features and their importance for obtaining this sequence is unexplored, which motivates the idea of using data-guided methods to obtain the weights for a wide set of features. Further, the complete algorithm can be run in a distributed manner. The proposed framework thus offers a complete and scalable traffic management for a continual 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. We compare our algorithm with that of an "optimal" algorithm, signalized intersection management, first-in first-out based intersection management as well as the algorithm proposed in [12]. We also demonstrate the computational scalability of the proposed algorithm through simulations. In particular, we show that for the proposed algorithm the computation time per vehicle essentially remains constant for a wide range of traffic arrival rates.

Notation: We use $\mathbb{R}$ and $\mathbb{N}_0$ for the set of real and whole numbers, respectively. For a discrete set V, we let $|V|$ be the cardinality of the set V.

II. Model and Problem Formulation

A. Model

Geometry of the Region of Interest: Consider an isolated intersection with a set of lanes $\mathcal{L}$. We refer to the lanes along with the intersection as the *region of interest*. Each lane $l \in \mathcal{L}$ has a fixed 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. Figure 1 presents the basic geometry of an example region of interest, where the set of lanes is $\mathcal{L} = \{1, 2, 3, \ldots, 12\}$ with 3 lanes (for going left, straight and right) on each branch. Figure 1 labels lanes 1, 8 and 12 and skips the rest for clarity. The translucent box shaded in red represents the conflict region or the intersection, which is the region of potential inter-lane collisions. The paths of some lanes intersect, while others do not. For each pair of lanes $l, m \in \mathcal{L}$, we let *compatibility* $c(l, m)$ be equal to 1 if $P_l \cap P_m = \emptyset$, or 0 if $P_l \cap P_m \neq \emptyset$. A pair of lanes l and m are *compatible* if $c(l, m) = 1$ and *incompatible* if $c(l, m) = 0$. We require that vehicles on incompatible lanes not be in the intersection at the same time.

Vehicles and their Dynamics: We assume that all vehicles are CAVs - they can communicate with each other and the infrastructure, and are automated. We also assume that the vehicles do not change lanes within the region of interest.

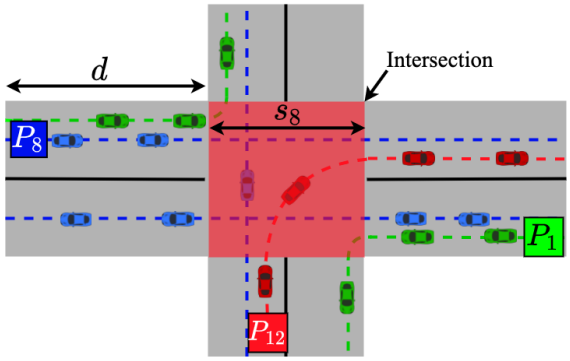

Fig. 1: Region of interest and the geometry of the intersection. Here only 3 lanes with numbers 1, 8 and 12 have been labeled. d is the length of the path to the intersection. s_l denotes the length of lane l within the interesection.

The CAVs can enter the region of interest on any lane in $\mathcal{L}$. We denote the lane that vehicle i traverses on by $l_i \in \mathcal{L}$ and the vehicle's length by L_i. The state of the vehicle at time t is $(x_i(t), v_i(t))$, where x_i and v_i are the position of the front bumper of the vehicle and the vehicle's velocity respectively on the path P_{l_i}. The dynamics of the vehicle i is

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

where u_i is the acceleration input to vehicle i. 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*, $t_i^{\mathcal{A}}$, enters the intersection at the *entry time*, $t_i^{\mathcal{E}}$, and leaves the intersection at the *exit time*, $t_i^{\mathcal{X}}$. Thus, $x_i(t_i^{\mathcal{A}}) = -d$, $x_i(t_i^{\mathcal{E}}) = 0$. and $x_i(t_i^{\mathcal{X}}) = s_{l_i}$.

B. Problem

The aim of the autonomous intersection management problem is to compute safe trajectories for the CAVs while maximizing 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 V is the set of all vehicles that arrive in the region of interest during a time interval of interest, $\dot{u}_i$ is the *jerk* of vehicle i and W_v, W_a and W_j are non-negative weights. We model the instantaneous discomfort of the passengers in vehicle i by the linear combination of the squares of acceleration and jerk. This metric penalizes sporadic high-magnitude disturbances caused by braking and acceleration manoeuvres performed by a vehicle [14]. Thus, each vehicle's contribution to the objective function is a linear combination of the distance it travels and the comfort of passengers in a time horizon T_h, starting from the vehicle's arrival time $t_i^{\mathcal{A}}$.

Constraints: The first set of constraints on the CAVs are bounds on their acceleration $u_i(t)$ and velocity $v_i(t)$, i.e.,

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

for all $\forall i \in V$ over an appropriate time interval. We assume that $\underline{u} < 0$, and $\underline{v} = 0$.

The second set of constraints ensure safety between the vehicles. Two types of collisions can occur in the region of interest: (1) rear-ended collision between successive vehicles on the same lane, and (2) collision between vehicles 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 vehicles i and j on the same lane ($l_i = l_j$) such that i is the vehicle immediately following j. This arrangement is formally denoted using the *follower indicator function* as,

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

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 [12], [16],

$$D(v_i, v_j) = L_j + r + \max\left\{0, \frac{1}{-2\underline{u}}\left(v_i^2(t) - v_j^2(t)\right)\right\}. \quad (4)$$

Here, r is a robustness parameter which is a constant distance to account for measurement and communication errors and delays. Then the *rear-end safety constraint* is

$$x_j(t) - x_i(t) \geq D(v_i, v_j),\ j \text{ s.t. } q(i,j) = 1 \quad (5)$$

for the time interval of interest. Note that the rear-end safety constraint (5) is more robust to loss of coordination, either due to breakdown in communication, control or due to malicious vehicles, than rear-end non-collision constraints [12], [16]. To ensure safety within the intersection, we also impose the constraint that vehicles on incompatible lanes not be within the intersection simultaneously. Thus, the *intersection safety constraint* for a pair of vehicles i and k is

$$t_i^{\mathcal{E}} \geq t_k^{\mathcal{X}} \quad \textbf{OR} \quad t_k^{\mathcal{E}} \geq t_i^{\mathcal{X}}, \quad \text{if } c(l_i, l_k) = 0. \quad (6)$$

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

$$\max_{u_i(.),\ i \in V} J \quad (7a)$$

$$\text{s.t. } (1), (3), (5)\ \forall t \in [t_i^{\mathcal{A}}, t_i^{\mathcal{A}} + T_h], \forall i \in V \quad (7b)$$

$$(6)\ \forall i, k \in V \text{ s.t. } c(l_i, l_k) = 0. \quad (7c)$$

Remark 1 (Challenges in solving (7) and problem statement)**.** There are several challenges in solving Problem (7). First, vehicles arrive randomly in a stream into the system and the information about their arrival and state is revealed only incrementally. Thus, Problem (7) cannot be "solved" in the usual sense. 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. Further, although the exact arrival times of the vehicles are not known a priori, we allow for the knowledge of the statistical data such as the mean arrival rate of vehicles. We seek to leverage this information for more efficient traffic management. Second, Problem (7) is a mix of large scale optimal control and combinatorial optimization. In particular, the number of optimal control subproblems that constraints (6) 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 yet provide near optimal performance. Lastly, the objective function used in 2 is not restrictive. Other objective functions could also be chosen in the problem 7. •

III. Overview of the Algorithm

Considering the complexities and time-criticality associated with Problem (7), we propose a computationally efficient algorithm to compute a sequence for 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*. The provisional phase 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 (7). 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. At the end of Section IV, 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's Perspective

A vehicle i starts execution of Algorithm 1 at $t_i^{\mathcal{A}}$, its time of arrival into the region of interest. Vehicle i communicates with the IM as soon as it arrives at $t_i^{\mathcal{A}}$. The IM 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 as `prov_phase`(i). Vehicle i communicates its provisional phase trajectory 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.

Provisional Phase of Vehicle i: Here we describe `prov_phase`(i), the method that vehicle i utilizes to compute the trajectory for its provisional phase. At $t_i^{\mathcal{A}}$, vehicle i obtains $t_i^{\mathcal{C}}$, the start time of its coordination phase, and

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
```

the trajectory of the vehicle preceding it on its lane. Vehicle i computes an optimal trajectory under several constraints including the *intersection entry prevention constraint*,

$$v_i(t) \leq \mathcal{V}(x_i(t)) := \sqrt{2\underline{u}x_i(t)}, \tag{8}$$

for all t in the time interval of interest. The upper bound $\mathcal{V}(x_i(t))$ is the maximum velocity that vehicle i may have at position $x_i(t)$ so that with maximum braking ($u_i(t) = \underline{u}$) vehicle i can come to a stop before entering the intersection. Thus this constraint prevents the vehicle from entering the intersection under the bounded control constraint. Then, the optimal control problem for vehicle i's provisional phase is

$$\max_{u_i(.)} \int_{t_i^{\mathcal{A}}}^{t_i^{\mathcal{A}}+T_p} \Big(W_v v_i(t) - \big[W_a \ u_i^2(t) + W_j \ \dot{u}_i^2(t) \big] \Big) \ \mathrm{d}t$$
$$\text{s.t. } (1), (3), (5), (8) \ \forall t \in [t_i^{\mathcal{A}}, t_i^{\mathcal{A}} + T_p]. \tag{9}$$

B. Intersection Manager's Perspective

Now, we describe Algorithm 2, which is from the IM's perspective. As soon as a vehicle i enters the region of interest, 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 vehicle i on its lane l_i so that vehicle i can compute its provisional phase trajectory and communicate it back to the IM. In this paper, for simplicity, we assume that IM carries out coordinated planning periodically at the instances $k\mathcal{T}_c$, where $k \in \mathbb{N}_0$. And we let $t_i^{\mathcal{C}} = k\mathcal{T}_c$, where k is the smallest integer such that $k\mathcal{T}_c \geq t_i^{\mathcal{A}}$.

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 l_i
    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
```

At each 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 $V_c(k)$ seeking to achieve optimized coordination and ensures the vehicles cross the intersection safely. We denote the coordinated phase planning problem at the instance $k\mathcal{T}_c$ by $\texttt{coord\_phase}(V_c(k))$. The IM communicates the trajectories for the coordinated phase to the vehicles in $V_c(k)$, which then execute them. In Section IV, we present $\texttt{coord\_phase}(V_c(k))$, the algorithm for planning the trajectories in the coordinated phase.

IV. Coordinated Phase

This section presents the trajectory optimization for the coordinated phase. We first present *combined optimization*, which is a naive centralized method and is not computationally scalable. Based on this method, we present the data-guided sequential weighted algorithm, which is significantly superior in terms of computational requirements. Further, as we demonstrate through simulations in Section V, this algorithm performs almost as well as the combined optimization.

The planning for the coordinated phase is carried out periodically with period $\mathcal{T}_c$. In particular, at the instance $k\mathcal{T}_c$, trajectory planning is carried out for the set of vehicles $V_c(k)$ that arrive into the region of interest during the interval $\big(\ (k-1)\mathcal{T}_c,\ k\mathcal{T}_c\ \big]$. 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

In this method, the IM computes the trajectories for all the vehicles in V_c simultaneously. The optimal control problem for the combined optimization method is a variation of the problem (7). The only differences are: the time horizon for the coordinated phase is T_c and the set of participating vehicles is V_c. Specifically, the objective function is

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

and the combined optimization problem is

$$\max_{u_i(.),\ i \in V_c} \ J^c \tag{10a}$$
$$\text{s.t. } (1), (3), (5) \ \forall t \in [t^{\mathcal{C}}, t^{\mathcal{C}} + T_c], \ \forall i \in V_c \tag{10b}$$
$$(6) \ \forall i, k \in V_c \cup V_s \text{ s.t. } c(l_i, l_k) = 0. \tag{10c}$$

Subsequent to solving (10) and updating the trajectories for the vehicles, V_s is updated to $V_s \cup V_c$. Combined optimization (10) requires the IM to compute optimal trajectories for each feasible intersection usage sequence and then pick the best sequence and the corresponding optimal trajectories. However, the number of feasible sequences grows exponentially with the number of vehicles on incompatible lanes. Thus, this method is not scalable and is not well suited for the time and safety critical problem of autonomous intersection management. Hence, we next propose a computationally scalable and efficient method for computing near optimal sequences and trajectories for the coordinated phase.

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

We next propose a scalable method for optimizing the intersection usage sequence and trajectories of vehicles in the coordinated phase. We call it *data-driven sequential weighted algorithm* (DD-SWA). The method scales linearly with the number of vehicles and it is amenable to a distributed implementation. We present an overview of DD-SWA in Algorithm 3. The algorithm begins with the set of unscheduled vehicles, V_c. In Step 6, we identify $\mathcal{F}$, the set of vehicles in V_c that are closest to the intersection. In Step 8

Algorithm 3: DD-SWA

1 **if** $t = 0$ **then**
2 $\quad V_s \leftarrow \emptyset$ {set of scheduled vehicles}
3 **end**
4 **if** $t = k\mathcal{T}_c$, *for* $k \in \mathbb{N}_0$, **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**

of the algorithm, the *precedence index* p_i is computed for every vehicle i in $\mathcal{F}$. The vehicle $i^* \in \mathcal{F}$ with the highest precedence index (after arbitrarily resolving any potential ties) is allowed 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, after which i^* is removed from V_c and added to V_s. This process is repeated until V_c is empty. The vehicles optimize their trajectories sequentially so as to satisfy the intersection safety constraint (6). 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 linear combination of certain *scheduling features* related to the vehicle $i \in \mathcal{F}$

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

Three of the features are 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 features capture the "demand" on lane l_i that is "following" vehicle i. First of these features is the number of vehicles $|Q_i|$, 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. 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, $\tau_i := \max\{t_m^{\mathcal{X}} - t_i^{\mathcal{C}} \mid m \in V_s\ s.t.\ c(l_i, l_m) = 0\}$. To prevent frequent switching of the right of way between incompatible lanes, minimum wait times are weighted negatively. The weighted linear combination of the features makes the computation of the precedence indices extremely simple. We propose tuning the weights based on offline simulations.

2) Trajectory Optimization: In Algorithm 3, the trajectories of the vehicles are computed sequentially. In each iteration, the vehicle i^* with the greatest precedence index is selected for trajectory optimization. However, notice from the combined optimization problem (10) that the optimization of the trajectory of vehicle i is coupled to the optimization of the other vehicles' trajectories through the constraints. One of the purposes of the precedence indices is to set a precedence in the constraint (6). Even then, the coupling is not fully eliminated. In order to compute the trajectories sequentially, we seek to decouple (10) into several optimization problems - one per vehicle. We have to do this in a manner that ensures we still get near optimal solutions to (10). Such a method aids in developing a distributed algorithm. A naive starting point for constructing such a decoupled problem is to consider only the term involving i^* in J of (10). 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. Let $\mathcal{D}_i$ be the *demand* 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

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

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

$$\max_{u_{i^*}(.)} J_{i^*}^c \text{ s.t. } (1), (3), (5)\ \forall t \in [t_{i^*}^{\mathcal{C}}, t_{i^*}^{\mathcal{C}} + T_c] \quad (12a)$$

$$t_{i^*}^{\mathcal{E}} \geq \tau_{i^*} + t_{i^*}^{\mathcal{C}}, \quad (12b)$$

with $i = i^*$ in (1) , (3) and (5).

Remark 2 (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. The computation of the trajectory of the vehicles is also of lesser complexity since for each vehicle we 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 combined optimization while the computation time per vehicle essentially stays constant for a wide range of traffic arrival rates. On the other hand for combined optimization, the computation time per vehicle increases exponentially with the traffic arrival rate. •

In the following result we show that if the vehicles arrive into the region of interest in a safe configuration then safety is guaranteed for all pairs of vehicles in the region of interest.

Theorem 1 (Sufficient condition for system wide inter-vehicle safety)**.** *If every vehicle i satisfies the rear-end safety constraint (5) at the time of its arrival, $t_i^{\mathcal{A}}$, and its initial velocity is such that that $v_i(t_i^{\mathcal{A}}) \leq \min\{\bar{v}, \mathcal{V}(-d)\}$, feasiblility of problems (9), (10) and (12) is guaranteed. Consequently, safety of all the vehicles is also guaranteed for all time.* •

Proof. The assumptions that at time $t_i^{\mathcal{A}}$ vehicle i satisfies the rear-end safety constraint (5) and $v_i(t_i^{\mathcal{A}}) \leq \min\{\bar{v}, \mathcal{V}(-d)\}$ ensures the feasibility for the problem for the provisional phase (9). The feasibility of problem (9) ensures the satisfaction of rear-end safety constraint (5) and the intersection entry prevention constraint (8) at the start of the coordinated phase $t_i^{\mathcal{C}}$. This ensures the feasibility of problems (10), and (12). Since feasibility of the problems ensures safety constraints, safety between every pair of vehicles is satisfied. □

C. Distributed Implementation of the Algorithm

Note that each vehicle i can implement its provisional phase in a distributed manner by communicating only with the vehicle preceding it in its lane. The design of DD-SWA is also 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.

Each vehicle can obtain information such as distance travelled, velocity and time since arrival locally. The other scheduling features, the safety constraints (5) and (6) and the weights for the scheduling features require communication. We make a distinction between the three types of communication required for this purpose: (1) intra-lane, (2) inter-lane and (3) central communication. 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. Similarly, the vehicle immediately in front of i^* can communicate its position and velocity trajectory which are sufficient to compute (5). 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. We denote such communication as inter-lane communication. Lastly, 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 communication.

V. Simulations

To evaluate the proposed algorithm, a simulation framework using Casadi [17] and Python was created. All the simulations were performed on an Intel i9-9900k 3.6GHz processor with 128GB of RAM. In the simulations, we assume that vehicles arrive according to a Poisson process with an average arrival rate of σ_l on lane $l \in \mathcal{L}$. To evaluate the proposed algorithm, we compare combined optimization and DD-SWA against a signalized intersection, the Hierarchical-Distributed algorithm [12] and the coordinated phase with a first-in first-out (FIFO) protocol for the sequence of intersection usage. The simulation results are for the particular case where vehicles only pass straight across the intersection. However, the proposed algorithms hold even when turning is allowed.

Here, we present the algorithms and the comparisons in greater detail. [1]. Firstly, we consider the case of a 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 lane 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). The cycle times and green times for the signals are obtained using Webster's method [18] corresponding to the arrival rate σ_l in each lane. Next, we consider the Hierarchical-Distributed (HD) algorithm presented in [12]. Lastly, we also compare with a FIFO protocol for the coordinated phase in these simulations.

Simulation Parameters: Table I lists the parameters of the intersection and the vehicles that are common to all the algorithms. We conducted simulations using all the algorithms 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 obtained from the Webster's method [18]. In each of the simulations, 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

[1] A video describing the main features of the proposed algorithm and simulations is available at: http://www.ee.iisc.ac.in/~pavant/files/figs/Data-Driven-IM.mp4.

TABLE I: 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_i	$4.3\ m$
Robustness parameter (4)	r	0.2 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	$30\ s$

function value per vehicle over the 20 trials for each value of σ across all the algorithms.

A. Results

We present 3 sets of comparisons between the various algorithms mentioned previously. Table II indicates the weights on the scheduling features used for computing the precedence index (11) in DD-SWA in each of the comparative simulations.

TABLE II: DDSWA Weights for comparisons

Weight Symbol	**C1 and C2**	**C3**
w_x	0.1	0.8
w_v	5	7
w_n	4.5	5
w_t	3	5
w_σ	40	40
w_s	6	7
w_w	0.5	5
w_l	0.02	0.02

1) Comparison 1 (C1): In this comparison, we compare the average time to cross (TTC) for the vehicles under the different algorithms. The weights on acceleration and jerk (W_a and W_j respectively) were set to 0 and the weight on velocity (W_v) was set to 1. In the HD algorithm, the fuel cost represented by $F_i(\bar{v}_i)$ in Equation (5) of [12] 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 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 2(a) shows that the average TTC for vehicles with combined optimization and DD-SWA is comparable for all arrival rates in the considered range. FIFO performance is marginally poor compared to DD-SWA and Combined Optimization as we are only considering low arrival rates. The HD algorithm's performance is comparable for arrival rates initially but performs poorly beyond 0.04 vehicles/s. The signalized algorithm performs better than the HD algorithm after 0.08 vehicles/s.

2) Comparison 2 (C2): In Comparison 2, similar to comparison 1, there was no emphasis on comfort, but the arrival rates (σ) vary 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 2(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. FIFO is initially better than the signalized intersection but it's performance rapidly deteriorates as the arrival rate increases. The signalized algorithm outperforms FIFO initially but outperforms the HD algorithm in this range. However, it does not perform better than the DD-SWA. The HD algorithm performs significantly worse than the other algorithms due to it's nature of creating bubbles with multiple vehicles.

3) Comparison 3 (C3): Figures 2(c) and (d) depict the average TTC and the average objective value for combined optimization and DDSWA when there is an equal weight of 1 on acceleration, jerk and velocity. 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 marginally outperforms DD-SWA both in terms of the average objective value and the average TTC. FIFO performs significantly poorly compared to the other two algorithms. The blue and red bars in the figure correspond to DD-SWA and combined optimization respectively.

Computation time comparison: Here we demonstrate the computational advantage of DD-SWA over combined optimization. In Figure 3, the lower and the upper edges of the boxes represent the first third quartiles respectively. The whiskers represent the maximum and the minimum of the data. Small circles beyond the whiskers represent outliers. The mean of the data is represented by the bold black line. Figures 3(a) and (b) compare the computation time per vehicle for combined optimization and DD-SWA for various arrival rates. Although the trend of $|V_c|$ is similar for both the algorithms, the computation time for combined optimization increases exponentially with the arrival rate. Figure 3(b) shows that DD-SWA has a nearly constant value of computation time per vehicle. We attribute this to the low computational effort required to determine the sequence of intersection usage and for sequentially optimizing the trajectories of the vehicles in V_c.

VI. Conclusion

In this work, we introduced a provably safe data-driven algorithm for intersection management. By decomposing into two phases, we ensured system wide safety and feasibility of vehicle trajectories. Simulations suggest that the proposed algorithm performs significantly better than traditional methods such as signalized intersections and first-in first-out algorithms. We also demonstrated through simulations that DD-SWA takes significantly less computational effort compared to the centralized implementation with only marginal loss in the objective value. Future work can be focused on developing learning-based methodologies to automate the tuning of the parameters in DD-SWA for various traffic scenarios. Other promising directions include extension to

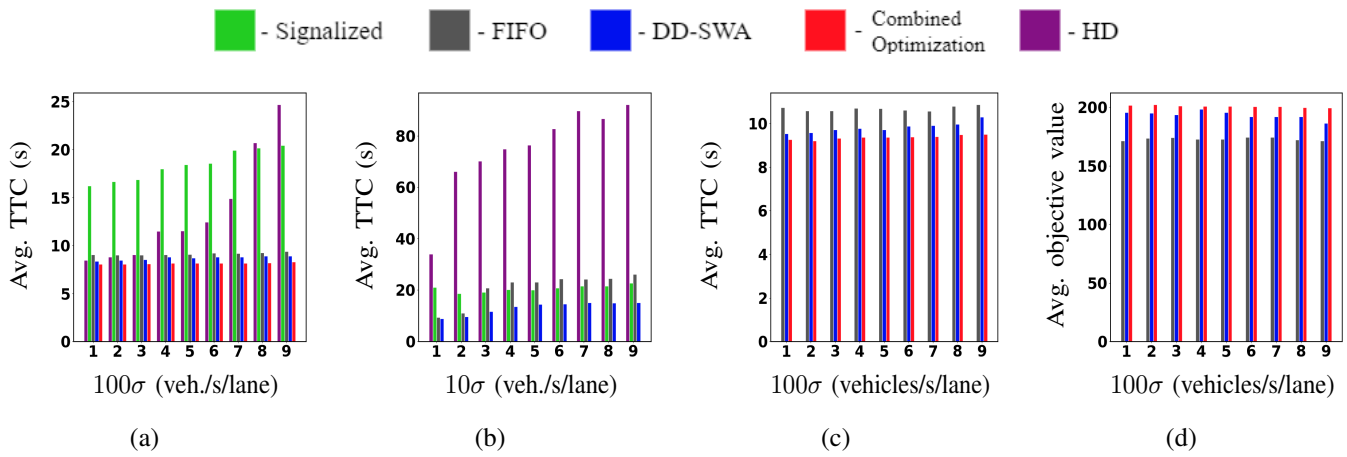

Fig. 2: (a) Comparison 1 - the average time to cross (TTC) for low arrival rates. (b) Comparison 2 - the average time to cross (TTC) for high arrival rates. (c) Comparison 3 - the average time to cross (TTC) for the vehicles. (d) Comparison 3 - the average objective value, when weight on acceleration, jerk and velocity (W_v) are equal to 1.

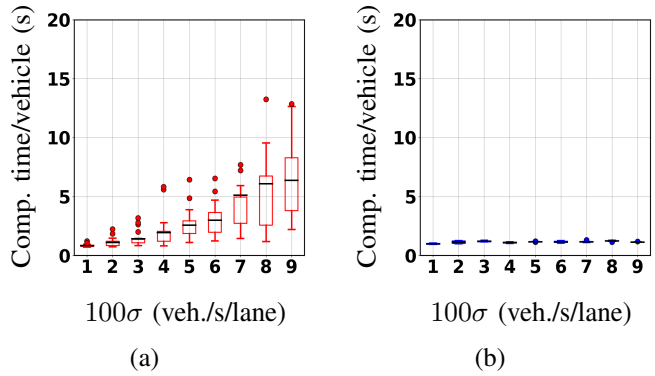

Fig. 3: Results of the computation time comparison. Figure (a) and (b) correspond to combined optimization and DD-SWA, respectively. The plots depict computation time per vehicle for the two methods.

traffic management for a network of intersections and 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] 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.

[3] 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.

[4] Y. Bichiou and H. A. Rakha, "Developing an optimal intersection control system for automated connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1908–1916, 2018.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

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

[13] Y. Wu, H. Chen, and F. Zhu, "DCL-AIM: Decentralized coordination learning of autonomous intersection management for connected and automated vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 103, pp. 246–260, 2019.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.