TraInterSim: Adaptive and Planning-Aware Hybrid-Driven Traffic Intersection Simulation

Pei Lv*, Xinming Pei*, Xinyu Ren*, Yuzhen Zhang*, Chaochao Li*, and Mingliang Xu*

Abstract—Traffic intersections are important scenes that can be seen almost everywhere in the traffic system. Currently, most simulation methods perform well at highways and urban traffic networks. In intersection scenarios, the challenge lies in the lack of clearly defined lanes, where agents with various motion panning converge in the central area from different directions. Traditional model-based methods are difficult to drive agents to move realistically at intersections without enough predefined lanes, while data-driven methods often require a large amount of high-quality input data. Simultaneously, tedious parameter tuning is inevitable involved to obtain the desired simulation results. In this paper, we present a novel adaptive and planning-aware hybrid-driven method (TraInterSim) to simulate traffic intersection scenarios. Our hybrid-driven method combines an optimization-based data-driven scheme with a velocity continuity model. It guides the agent’s movements using real-world data and can generate those behaviors not present in the input data. Our optimization method fully considers velocity continuity, desired speed, direction guidance, and planning-aware collision avoidance. Agents can perceive others’ motion plannings and relative distances to avoid possible collisions. To preserve the individual flexibility of different agents, the parameters in our method are automatically adjusted during the simulation. TraInterSim can generate realistic behaviors of heterogeneous agents in different traffic intersection scenarios in interactive rates. Through extensive experiments as well as user studies, we validate the effectiveness and rationality of the proposed simulation method.

Index Terms—Collision avoidance, data-driven, heterogeneous multi-agent, traffic intersection, traffic simulation.

I. INTRODUCTION

Traffic simulation has been widely used in many typical applications such as autonomous driving, urban planning, and computer games. Traffic intersection is an important part of traffic system, however, existing works mainly focus on freeways or city-scale traffic [1], [2], [3], [4]. The intersection scenario contains traffic lights and different types of agents from various directions. In such complex environments, it is challenging to efficiently simulate plausible traffic behaviors of different types of agents.

In real world intersections, the motion planning (i.e., turn left/right, go straight) of vehicles plays an important role in determining their movements, which cannot be ignored. Many methods have been proposed to imitate the movement and interaction behavior of agents in crowd simulation and traffic simulation, such as force-based methods [5], [6], velocity-based methods [7], [8], vision-based methods [9], etc. Whereas, these methods are difficult to be directly applied to simulate traffic intersection scenarios without well-defined lanes in the central area. It is easy for agents to react implausibly to avoid possible collisions if the agents’ motion plannings are ignored (Fig. 1).

(a) Vehicles running straight from opposite directions. (b) Vehicles turning right and left from different directions.

Fig. 1. Vehicles at intersections that attempt to avoid collisions only based on the safe distance without considering the motion planning may lead to implausible behavior, such as excessive safety (orange) or unnecessary deceleration (red).

The traffic simulator [10] and [11] pre-define lanes and vehicles drive along queues by the car-following model. However, their simulation results are too regular and inflexible.

Recently, with the development of data acquisition technology, data-driven methods have gradually gained more and more attention. Researchers use ground-truth data (e.g., sensor data [2], [12], GPS data [13], [14]) to synthesize trajectories or learn the motion features of agents to reconstruct the traffic flow. Traffic simulation can also use advanced trajectory prediction techniques [15], [16], [17], [18] to reproduce traffic flow at signalized intersections. These works can make up the shortage of model-based methods which are difficult to formulate a set of perfect rules to control the motion of agents. However, their results are highly dependent on the quality and quantity of input data. If the input data is flawed (e.g., few data, non-uniform data distribution, noise artifacts), data-driven methods are difficult to obtain satisfactory results.

On the other hand, most existing methods introduce plenty of parameters to better control the agents. These parameters need
to be adjusted iteratively to achieve desired simulation results. Different kinds of agents are mixed in the traffic intersection scenario, which further increases the number of parameters. Some works use empirical data to automatically estimate the parameters of existing simulation methods [19], [20], [21], [22]. The results of the above work are also limited by the input data. Moreover, the homogeneous agents in these works using same parameters will inevitably generate similar movement patterns. And the individual flexibility of agents in real-world intersection cannot be efficiently reflected.

In order to overcome above limitations, we propose a novel adaptive and planing-aware hybrid-driven simulation method for traffic intersection scenarios. Inspired by [23], our method combines an optimization-based data-driven approach with a velocity continuity model to generate plausible simulation results even the input dataset is flawed. Specifically, our optimization function fully considers velocity continuity, desired speed, direction guidance, and collision avoidance. In collision avoidance, we integrate not only relative distances between agents but also their motion plannings. In particular, we do not directly update the agent with the velocity in the input dataset that minimizes the optimization function. Instead, we design indicators to decide whether this velocity can be used to update the movement of an agent. If this velocity is unacceptable, we will feed back a set of supplementary candidate velocities based on the agent’s current velocity through the continuity of velocity into the optimization process. To reduce tedious parameter tuning and allow agents to have more flexible trajectories, we dynamically estimate the parameters of the proposed method for each agent during the simulation. The input in our method can be different types of trajectory data collected by various devices. Afterwards, we convert those trajectories in the input dataset into the candidate velocity dataset. To improve the quality of these initial trajectories, we further design a curve fitting method to optimize the input trajectories and segment them according to their spatial distribution.

The main contributions of our work can be summarized as follows:

- We propose a hybrid-driven method to control the movement of agents in traffic intersection scenarios. It combines an optimization-based data-driven scheme with a velocity continuity model, which can effectively drive agents to move realistically even if the input data is defective.
- We introduce a novel collision avoidance energy function integrating the motion planning of agents. Different plausible behaviors of agents can be simulated at traffic intersections without pre-defined lanes.
- We present an adaptive parameter adjustment mechanism to dynamically optimize the parameters of our hybrid-driven method during the simulation. It can reflect the flexibility of agent motion and the diverse responses of agents to traffic lights efficiently.

Through extensive experiments and user studies, it is demonstrated that our method can generate more realistic traffic simulation results than existing methods. Fig. 2 shows some snapshots generated by the proposed method on different traffic intersection scenarios.

II. RELATED WORK

In this section, we first overview the prior traffic simulation methods, including rule-based methods and data-driven methods. Then we review the works related to multi-agent navigation and parameter estimation.

A. Traffic Simulation

Early work mainly used rule-based methods to control the motion of agents. In computer graphics community, they can be divided into two categories based on the level of simulation details. Microscopic methods mainly focus on the details of individual behaviors, vehicles are treated as discrete autonomous agents that satisfy certain constraints [24], [25], [26], [27]. The Intelligent Driver Model (IDM) [28] and the lane-changing model [29] are noteworthy, which have been improved and expanded by many subsequent works [3], [30], [31]. Macroscopic methods focus on the overall traffic trend, where traffic flows are viewed as continuous dynamics such as fluid or gas. The regulations of vehicle motion are described by solving partial differential equations of fluid dynamics [32], [33], [34], [35], [36]. However, these techniques mainly focus on the decision-making process of vehicles on freeways (i.e., acceleration/deceleration or lane-changing). It is difficult to formulate a complete set of rules for agents to match the diverse behaviors in real-world intersections.

Recently, with the development of data acquisition techniques, more attention has been paid to data-driven simulation methods. Researchers extract information from ground-truth data to reconstruct traffic flow. Spatio-temporal data from inroad sensors are used to reconstruct realistic traffic flow [2], [12]. Li et al. [13], [14] estimate traffic dynamics using sparse GPS...
tracks. The specific driving characteristics of drivers are learned from videos in [37]. Bi et al. [38] simulate the lane-changing process using the vehicle trajectory data. They use randomized forest and back-propagation neural network to learn lane-changing characteristics. Chao et al. [1] propose a novel data-driven approach using texture synthesis. They use trajectory data to calculate the velocity in each frame that minimizes the traffic energy. Trajectory prediction technique can also be used as a novel data-driven scheme. Skarkar et al. [15] and Huang et al. [16] use video and multiple sensors data to learn the behavior of agents at intersections to predict their trajectories. Zhang et al. [39] proposed a trajectory prediction method for intersection scenarios considering the impact of traffic lights on vehicles. Based on an in-house collected intersectional traffic dataset, Bi et al. [40] propose a deep learning-based editable simulation framework combined with trajectory prediction. Data-driven methods can generate more precise traffic flow, but their simulation results are highly correlated with the input data. Our method combines data-driven and model-based methods to plausibly guide the agents’ movement even the input data is flawed.

B. Multi-Agent Navigation

There are many researches involving multi-agent path navigation in traffic simulation and crowd simulation. Local navigation is the most popular research area in the microscopic paradigm, especially collision avoidance between agents [41]. Many approaches have been proposed to simulate the interactions between agents by modeling natural human characteristics. Helbing et al. [42], [43] first propose the social force model (SFM), assuming that the agent is subjected to the resultant “forces” of the environment and other agents, which drives the agent’s movement. Han et al. [44] and Chao et al. [5] use the force-based method to simulate mixed traffic on freeways. Velocity-based models employ the concept of prediction, where the agent selects the best option from many candidate velocities [7], [8]. Karamouzas et al. [45] combine force-based methods to provide guaranteed collision-free motion. Ren et al. [23] formulate the motion decision of heterogeneous agents as an optimization problem. They construct a velocity dataset, and the agent chooses the velocity that minimizes the energy function. Furthermore, Ondřej et al. [9] propose a vision-based method, each agent makes decisions based on the information rendered onto the virtual retina. To further simulate the movement of real-world pedestrians, human psychology is also incorporated into the crowd simulation [46], [47], [48]. These works have achieved collision-free motion, however, they cannot be directly applied to intersectional traffic simulation due to the lack of pre-defined lanes. Our planning-aware collision avoidance scheme injects agents’ motion plannings, which can be integrated with previous work to simulate intersection scenarios.

C. Parameter Estimation

Most of existing simulation methods use plenty of parameters to control the details of the simulation, and the choice of parameters can significantly affect the simulation results which require empirical iterative tuning to find the best combination of parameters. Researchers refer to real-world data for parameter estimation by searching for a set of parameters to minimize the evaluation function. Wolinski et al. [20] propose several general evaluation metrics for crowd simulation algorithms. Berseth et al. [19] propose a general method to automatically find the optimal parameters for steering algorithms in crowd simulation. Zhou et al. [49] model pedestrian dynamics and then learn the parameters of the model from crowd videos. In traffic simulation, Kesting et al. [22] calibrate the parameters of IDM and velocity difference model using genetic algorithm (GA). Yang et al. [50] use GA to estimate the parameters of their modified SFM to simulate the intersection with an island work zone. Chao et al. [21] employ an adaptive genetic algorithm (AGA) to estimate the parameters of their force-based model. These methods need to extract a fixed set of parameters offline using a large amount of sampled data. Different from them, the parameters of each agent in our method are dynamically adjusted at each time step.

III. OVERVIEW

Our adaptive and planning-aware hybrid-driven method can simulate the heterogeneous agents in different traffic intersection scenarios with defective input data and without tedious parameter tuning manually. Fig. 3 provides the overview of our method.

We use traffic trajectory dataset as the input of our method. Since noise is inevitable in the input dataset, we design a curve fitting approach to remove them (Section IV-A). It can also be used for other dataset with significant noise or lower sampling rates. The denoised trajectories are grouped according to their spatial distribution. Afterwards, they are converted to velocities, which are used to construct candidate velocity datasets in our method (Section IV-B). Each candidate velocity dataset is sorted by the magnitude of the velocity. The intersection scene consists of static lanes and dynamic traffic lights. Before the simulation starts, we initialize the position and velocity information for each agent.

Inspired by [23], we also treat the movement decision-making progress of the agent at each time step as an optimization problem. Each agent searches the dataset for a velocity to minimize the energy function (Section V-A). The difference with [23] is that we do not directly use this velocity to update the agent’s state. Instead, we design indicators to confirm that a velocity is acceptable, including energy terms, velocity magnitude, and direction. If the optimal velocity in the constructed dataset exceeds any threshold, our method will repeat the optimization process by supplementing a series of candidate data with the velocity continuity (Section V-C). The agent then updates its state with the final result and adds that velocity to its corresponding dataset. Therefore, our method can simulate the behavior that does not exist in the input data, and the dataset is expanded dynamically during simulation. Our energy function comprehensively considers the continuity of velocity, direction guidance, expected velocity, and collision avoidance (Section V-B). The agent’s motion planning is injected into our collision avoidance energy function to exhibit plausible agent interaction in intersection areas without predefined lanes. The characteristics of different
types of agent behavior are reflected by different parameters of energy functions. We allocate a set of initial parameters that agents can move normally in the sparse traffic flow. In order to show the individual flexibility and dynamics of homogeneous agents, at each time step, we update the parameters of each agent based on the agent’s environmental information (Section V-D).

IV. DATA PRE-PROCESSING

In this section, we describe the details of the input data pre-processing. The input of our method comes from trajectory datasets in different scenarios. To improve the quality of simulation results, we denoise the sampling trajectories and then construct the candidate velocity dataset.

A. Trajectory Denoising

Noise is inevitably introduced in traffic data collection and processing. A large amount of noise data may cause agents to be unable to find velocities suitable for updating their own motion state. We use curve fitting to reduce noise in sampled trajectories. Trajectory curves are interpreted as parametric curves taking frame \( t \) as the parameter. The \( \Gamma^S \) specifies the set of sampling trajectories. The representation for the \( j \)-th trajectory is:

\[
\Gamma^S_j(t) = \left\{ x^S_j(t), y^S_j(t) \right\}, \quad t \geq 0,
\]

where \( x^S_j(t) \) and \( y^S_j(t) \) are the coordinate of the \( X \)-axis and \( Y \)-axis at frame \( t \), respectively. We use B-spline to maintain the locality of the curve, so as to prevent some sampling points with large errors from affecting the whole curve. We fit the \( X \)-axis direction and \( Y \)-axis direction respectively, and they have the same loss function. For the \( X \)-axis direction, the equation is defined as follows:

\[
L^X = w_1 L^X_p + w_2 L^X_v + w_3 L^X_c + w_4 L^{XY},
\]

where \( L^X_p \) is the loss function for the distance from the fitting point to the sampling point, \( L^X_v \) and \( L^X_c \) are the loss functions for the continuity of velocity in magnitude and direction, \( L^{XY} \) is the loss function that ensures the continuity of the trajectory after the combination of \( X \)-axis direction and \( Y \)-axis direction, and \( w_1, w_2, w_3, w_4 \) are the corresponding weights. The loss functions \( L^X_p, L^X_v, L^X_c \) and \( L^{XY} \) are weighted with \( w_1 = 0.7, w_2 = 30, w_3 = 20 \) and \( w_4 = 30 \), respectively. The results of trajectory denoising are shown in Fig. 4.

The term \( L^X_p \) is designed to ensure the result having a similar shape to the sample curve:

\[
L^X_p = \left\| x_j(t) - x^S_j(t) \right\|_2,
\]

where \( x_j(t) \) is the \( X \)-coordinate of the fitted curve \( j \) at the \( t \)-th frame.
Agent will not change its motion state frequently or significantly in consecutive time frames at intersections, so we introduce the term $L_c^X$ and $L_c^Y$ to ensure the motion continuity. They are defined in (4) and (5), respectively

$$L_c^X = \|x_j(t) - x_j(t-1)\|_2,$$

$$L_c^Y = \|y_j(t) - \frac{1}{2K} \sum_{k=1}^{K} [x_j(t-k) + x_j(t+k)]\|_2.$$

Among them, $x_j(t-1)$ denote the $X$-coordinate of the fitted curve $j$ at previous frame, and $K$ denotes the consecutive adjacent frames.

The above terms can only constrain the current direction, but the combination of two directions may break the continuity of velocity. The term $L_c^{XY}$ is used to keep the motion continuity after combining $x_j(t)$ and $y_j(t)$:

$$L_c^{XY} = L_c^X + L_c^Y,$$

where $L_c^X = \|\|v_{j,t} - v_{j,t-1}\|\|_2$ is for the continuity of velocity magnitude, and $L_c^Y = \|\|\hat{v}_{j,t} - \hat{v}_{j,t-1}\|\|_2$ is for the continuity of direction.

### B. Candidate Velocity Dataset Construction

Agents with the same motion planning may have similar movement routes. Therefore, when computing the energy function, most of the candidate velocities required by an agent to update its state come from the trajectories which have the same motion planning. To optimize the solution space, we reduce many candidate velocities that may reduce the diversity of the input data, our hybrid-driven method (Section V-C) can attenuate for this. We construct candidate velocity dataset for pedestrians and bicycles in the same scheme. At the beginning of the traffic simulation, the agent selects the corresponding candidate velocity dataset according to its departure lane and motion planning.

### V. TRAFFIC INTERSECTION SIMULATION

In this section, we describe the details of our adaptive and planning-aware hybrid-driven method for traffic intersection simulation.

#### A. Problem Formulation

Given the initial velocity, desired speed, the point of departure, and destination of agents in the scenario, we aim to simulate plausible trajectories of these agents. The initial candidate velocity dataset is denoted as $D = \bigcup_n v_n$, where $v_n$ is the velocity at frame $t_n$ converted from trajectory data. Our supplementary dataset is denoted as $D_s = D \bigcup S(v_{i,t-1})$, where $S()$ is our data supplement function. $w$ are the parameters of the energy function. We use $s_{i,t} = [p_{i,t}, v_{i,t}, g_{i,t}, e_{i,t}]$ to specify the state of agent $i$ at time $t$, where $p_{i,t} \in \mathbb{R}^2$ denote the position, $v_{i,t} \in \mathbb{R}^2$ denote the velocity, $g_{i,t} \in \mathbb{R}^2$ and $e_{i,t} \in \mathbb{R}$ respectively denote the guidance direction and expected speed. Inspired by [23], we also use optimization-based scheme. The agent searches the velocity to minimize the energy function to update motion state. The difference is that we adjust the parameters for each agent at the beginning of each time step, and our candidate velocities are not only from the input dataset, but also generated by velocity continuity model. The formula for our method to update the state of agent is

$$v_{i,t} = \arg \min_{v \in D_s} E(w_{i,t}, v, s_{i,t-1}),$$

$$p_{i,t} = p_{i,t-1} + v_{i,t} \Delta t,$$

where $E(w_{i,t}, v, s_{i,t-1})$ is the energy function, $w_{i,t}$ are parameters of agent $i$ at time $t$, $p_{i,t-1}$ is the position of agent $i$ at time $t-1$, and $\Delta t$ is a time step.

#### B. Energy Optimization

The movement of an agent is constrained by two aspects. On the one hand, it comes from itself: physical limitations, destination, and its expected speed. The other is from its surroundings: neighbors, lanes, and traffic lights. We design the energy function $E$ as follows:

$$E = w_v E_v + w_g E_g + w_c E_c,$$

where $E_v$ is the velocity optimization term, $E_g$ is the direction guidance optimization term, and $E_c$ is the planning-aware collision avoidance optimization term. $w_v$, $w_g$ and $w_c$ represents their
weights, respectively. These weights are adaptively adjusted in our method, which will be described in Section V-D.

Velocity Energy Optimization Term: Due to physical limitations, agents will not change their velocity significantly in a short time. Each agent has an expected speed when the density of a certain area is small, the interaction between agents will be weak, and other external interference factors are basically non-existent. The velocity energy $E_v$ is designed to indicate the agent to maintain a limited velocity change while trying to achieve its desired speed:

$$E_v = w_{dir}E_v^{dir} + w_{mag}E_v^{mag} + w_cE_v^c,$$

where $E_v^{dir}$ and $E_v^{mag}$ represent the continuity of velocity in direction and magnitude, respectively, and their definitions are consistent with (6). We define $E_v^{dir} = ||v - e_{i,t}||_2$ to implement the agent to tend to approximate its expected velocity. $e_{i,t}$ is the maximum magnitude of velocity desired by the agent $i$ at time $t$.

Direction Guidance Energy Optimization Term: We use a guidance direction to control agents to move towards their destination:

$$E_g = ||\dot{v} - g_{i,t}||_2.$$

In the central area of an intersection, we define the guidance direction $g_{i,t} = \frac{p_{i,goal} - p_{i,t}}{||p_{i,goal} - p_{i,t}||}$, where $p_{i,goal}$ is the destination coordinate of agent $i$. In the lane area, the guidance direction is parallel to the road.

Planning-Aware Collision Avoidance Energy Optimization Term: Collision avoidance between agents is the most attended issue in local navigation decisions. The agent should avoid collisions that may occur after a few time steps, but also conform to behaviors of agents in the real world. We treat collision avoidance energy as an implicit influence of surroundings on the agent. And we ensure that the neighbors with greater influence take a larger proportion of the total energy,

$$E_{c\text{total}} = \sum_{\phi \in \Phi} \frac{||E_c(i,\phi)||^2}{\sum_{\phi \in \Phi} E_c(i,\phi)},$$

where $\Phi$ is the set of neighbors in the perception area of agent $i$ at timestep $t$, $\phi$ is any neighbor belonging to $\Phi$, and $E_c(i,\phi)$ is the energy function of agent $i$ interacting with neighbor $\phi$.

Due to the lack of well-defined lanes at intersections, agents may make implausible decisions if only relative distances are considered when avoiding collisions (Fig. 1). The motion planning of agent cannot be ignored. We define $E_c(i,\phi)$ depending on the kind of agents. When agent $i$ and neighbor $\phi$ are homogeneous agents and they are not pedestrians,

$$E_c(i,\phi) = M(i,\phi)f(\xi(i),\theta_\phi)e^{d_s - d_\phi}. \tag{12}$$

If $i$ and $\phi$ are both pedestrians or they are heterogeneous agents, the definition is:

$$E_c(i,\phi) = f(\xi(i),\theta_\phi)e^{d_s - d_\phi}, \tag{13}$$

where $\xi(i)$ is the motion planning of agent $i$ (i.e., turn left/right, go straight), which is determined by agent’s departure and destination, $d_s$ is the safe distance for agent to avoid colliding with its neighbors, $d_\phi$ and $\theta_\phi$ are the distance and the angle between $i$ and $\phi$ after $T = 15 \cdot \Delta t$ time steps, respectively. The mask matrix $M(i,\phi)$ is filled with 0, 0.5 and 1. Its value indicates whether a collision between agents is possible, which is obtained by the motion planning and location of $i$ and $\phi$ (Fig. 6(a) provides a calculation for a right-turning vehicle). The definition of $f(\xi(i),\theta_\phi)$ is based on agent $i$’s own motion planning. We assume that the sensitivity of agents to their surroundings is different. For example, when the agent turns right, it is less sensitive to neighbors on the left. We design $f(\xi(i),\theta_\phi)$ as a piecewise Gaussian function:

$$f(\xi(i),\theta_\phi) = \left\{ \begin{array}{ll} \frac{\sigma_1^2}{4\pi^2} & \theta_\phi \in [-\theta_{\mu_1}, \theta_{\mu_2}] \\ \frac{\sigma_2^2}{4\pi^2} & \theta_\phi \in [-50, \pm\theta_{\mu_1}] U(\mp\theta_{\mu_2}, 50) \end{array} \right. \tag{14}$$

where $\sigma_1^2$ and $\sigma_2^2$ are variances used to adjust the sensitivity of the angle, $\theta_\mu$ is the boundary point of the function. $\sigma_1^2, \sigma_2^2$ and $\theta_\mu$ vary depending on different motion plannings. For vehicles and bicycles, if $\xi(i)$ goes straight, we set $\sigma_1^2 = 550, \sigma_2^2 = 200, \theta_{\mu_1} = \theta_{\mu_2} = 30$; if $\xi(i)$ is turning, we set $\sigma_1^2 = 650, \sigma_2^2 = 250, \theta_{\mu_1} = 20, \theta_{\mu_2} = 40$. For pedestrians, we set $\sigma_1^2 = 1200, \sigma_2^2 = 200, \theta_{\mu_1} = \theta_{\mu_2} = 35$.

C. Supplementary Data Generation

The energy functions are not sufficient to provide a complete description of the real-world agent’s kinematic characteristics. The method in Section V-B is data-driven, and its performance is closely related to the quality of the input data. When the input data is defective, the agent can still find a velocity that minimizes the value of the energy function, but this velocity is not necessarily suitable for updating the agent’s motion. For instance, in the extreme case that the input data only has a unique velocity, this velocity will always minimize the value of the energy function. No matter what happens to the surrounding, such as steering and collision avoidance, agents can only select it to move along a straight line.
We set up an indicator $\tau(v)$ to measure whether the current numerical optimal speed is behaviorally acceptable:

$$
\tau(v) = \tau_m(\|v_{i,t-1}\| - \|v\|) \land \tau_d(\angle \hat{v}_{i,t-1} \hat{v}) \land \tau_g(w_g E_g) \land \tau_c(w_c E_c),
$$

(15)

where $\angle \hat{v}_{i,t-1} \hat{v}$ is the angle between $\hat{v}_{i,t-1}$ and $\hat{v}$. Indicator $\tau_m$, $\tau_d$, $\tau_g$, $\tau_c$ are used to measure the magnitude and direction of velocity, guidance direction, and collision avoidance, respectively, which correspond to the energy function in (8). We empirically set $\tau_m = \tau_d = 1.0$, $\tau_g = 2.0$, and $\tau_c = 10.0$.

We assume that the velocity $v$ cannot be used to update the state of agent if it does not satisfy the threshold indicator. Then we generate some new candidate data via velocity continuity based on the velocity of the agent in previous frame. The $S_m(v)$ and $S_d(v)$ generate supplementary data in the magnitude and direction of the velocity, respectively:

$$
S_m(v) = \left\{ ||v^*|| : ||v^*|| = ||v|| + p I^m, p \in [-\psi^m I^m, \psi^m I^m] \right\},
$$

(16)

$$
S_d(v) = \left\{ \hat{v}^* : \hat{v}^* = \hat{v} + q I^d, q \in [-\psi^d I^d, \psi^d I^d] \right\}.
$$

(17)

In above equations, $\psi^m$ and $\psi^d$ denote the range of supplementary velocity in magnitude and direction, $I^m$ and $I^d$ denote the interval of magnitude and the interval of direction, respectively. Afterwards, we obtain supplementary candidate velocities $v^*$, and $\hat{v}^* = ||v^*|| \hat{v}^*$. We feed all $v^*$ back to the energy optimization stage (Section V-B) and then select the one that minimizes the energy function as $\hat{v}_{i,j}$ to update the state of agent $i$, then add it to $i$’s corresponding candidate velocity dataset.

D. Parameter Adjustment

We propose parameter adjustment to reduce tedious adjustment of various parameters and make the agent have more flexible movement patterns. Specifically, at each time step, the parameters of each agent are dynamically computed according to the environment on the basis of a set of benchmark parameters.

We alter $w_c$ and $w_g$ in (8). $E_c$ is used to measure the influence between agents, so its weight $w_c$ remains unchanged.

Velocity Energy Optimization Term: We assume that the weights in $E_c$ are initial values when agents move normally. We adjust parameters in $w_c$ in following two cases: (i) agent interaction (ii) facing traffic lights. As mentioned in Section V-B, $E_c$ is regarded as the influence of neighbors on agents. When there is a possible collision, we adjust the parameters in $E_c$ according to short-term influence:

$$
\begin{align*}
w_{dir} &= w_{dir}^{init} + (E_c^{temp} / \lambda_{dir}), \\
w_m &= w_m^{init} - (E_c^{temp} / \lambda_m), \\
w_c &= w_c^{init} - (E_c^{temp} / \lambda_c),
\end{align*}
$$

(18)

where $w_c^{init}$ denotes the initial parameter and $\lambda$ is used to measure the sensitivity of parameters to neighbors. $E_c^{temp}$ is a short-term influence, it differs from $E_c$ in (11) where it calculates the influence of neighbors on the agent after $T/2$ time steps. When facing an imminent collision, due to the low flexibility of the vehicles, we set them to avoid the collision by decelerating rather than turning suddenly. We empirically set $\lambda_{dir} = \lambda_m = \lambda_c = 2.0$. Pedestrians and bicycles are more agile, they can change direction to avoid possible collisions. We set $\lambda_{dir} = 4.5$, $\lambda_m = 0.5$, $\lambda_c = 1.5$ for pedestrians, and $\lambda_{dir} = 2$, $\lambda_m = 0.5$, $\lambda_c = 1.5$ for bicycles.

We treat the traffic light as a linear obstacle containing the state. Agents will gradually stop as they approach the line segment under the red light. In real life, if the traffic light turns red when pedestrians are already inside the intersection area, they will speed up through the crossroad for their safety and to avoid traffic jams. Before the simulation starts, we initialize all agents with two desired velocities, $e_1$ and $e_2$ are desired speed for driving normally and accelerating, respectively. When agents need to speed up, we set $e_{1,t} = e_2$ and increase $w_c = 2 w_c^{init}$ in (9). Some drivers may accelerate to pass under a yellow light. We determine whether there is a condition for the agent to accelerate:

$$
\frac{||v_{i,t}|| + e_2 t'}{2} \geq G,
$$

(19)

where $t'$ is the time when the vehicle accelerates through the stop line, $G$ is the distance between the agent’s current position and the point where the entire vehicle completely crosses the stop line. Since the acceleration of a vehicle is not instantaneous, the velocity we use in calculation is the interpolation of the current speed and the desired speed. And we estimate time steps $t' = \varepsilon (s-1)$, $\varepsilon$ is the frame rate, and $s$ is the number of seconds remaining in yellow lights.

Direction Guidance Energy Optimization Term: The widths of roads may not be exactly the same, since we do not predefine lanes in the central area of the intersection, the weight $w_g$ needs to be repeatedly adjusted to balance the movement of steering vehicles and bicycles in different lanes. According to our observations, the steering range of the vehicle is small when it just drives out of the lane, and gradually increases with the moving process. Based on this, we compute $w_g :

$$
w_g = \frac{\text{dis}_{AC}}{\text{dis}_{CD}} w_{g}^{init},
$$

(20)

where $A$ and $B$ denote the midpoints of the lane boundaries, $C$ and $D$ denote the destination and current coordinates of the vehicle, $\text{dis}_{AC}$ represents the distance between $A$ and $C$, and $\text{dis}_{CD}$ represents the distance between $C$ and $D$, respectively (Fig. 7). We only perform this calculation in the intersection area (i.e., interval $AB$) to avoid $\text{dis}_{AC} / \text{dis}_{CD} \to \infty$.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

We conduct experiments on different intersection scenarios to demonstrate our method. Fig. 2 shows the snapshots of our simulation results in different scenarios, including 4-lane crossroad (Crossroad-1), 2-lane crossroad (Crossroad-2), T-junction, and Y-junction. The corresponding traffic animations are included in the supplementary demo video.

The input in our method can be trajectory data collected by different devices (various sensors, video tracking, etc.). Our
LV et al.: TRAINTERSIM: ADAPTIVE AND PLANNING-AWARE HYBRID-DRIVEN TRAFFIC INTERSECTION SIMULATION 5757

method includes trajectory optimization and supplementary data generation, so small data volume, non-uniform data distribution and noise artifacts are all acceptable. However, since we divide the data according to spatial distribution, it is best to choose trajectories which can distinguish motion plannings. For crossroad scenarios and T-junction scenario, our input samples are selected from VTP-TL Dataset [39], where trajectories are labeled in videos shot by the UAV from different traffic intersection scenes with a 30 fps frame rate. The crossroad scene has four lanes merging into the central area from different directions. We randomly select six cars, four pedestrians and four bicycles from trajectories in each lane. The average duration of each trajectory is 350 frames. For Y-junction scenario, our input are selected from Waymo Open Motion Dataset [51], where trajectories are acquired from sensors installed in the vehicle. Each spatio-temporal segment in this dataset has a duration of 9 seconds with 10 hz sampling rate, and we randomly select ten trajectory segments from each lane.

A. Runtime Performance

Our approach is implemented with the Unreal Engine in a 64-bit desktop with a 3.80 GHz Intel(R) Core(TM) i7-10700 K processor and 16 GB memory. We achieve the behavior characteristics of heterogeneous agents by setting different parameters. We use 1.0 as the benchmark to initialize all parameters. Since the parameters in our method are adaptively calculated during the simulation, we only need to manually make minor adjustments to the parameters in $E_v$ according to the motion characteristics of different kinds of agents. Table I lists the parameter values used in our method. We use the scheme mentioned in [23] to speed up the computation, where agents only interact with neighbors in adjacent grids and choose velocity from limited search space. Table II shows the runtime performance of our method in different traffic intersection scenarios.

To evaluate the performance of our method in detail, we analyze the specific computation time in the Crossroad-1 scenario (Fig. 8). We show the computation time per frame and the time to compute agent interactions. Since the size of the solution space in the dataset is the same for each time step, the change in computation time is mainly due to the computational cost of each candidate velocity increased by the interaction between agents.

B. Comparisons

1) Effectiveness of Supplementary Data: Due to the inherent characteristic stochastic of traffic, direct trajectory comparisons are usually not performed for traffic simulation [52]. Vehicles in the traffic intersection scenarios come from different directions and it is difficult to directly calculate the gap between vehicles. To validate the performance of our hybrid-driven method, we compare the distribution of velocity and steering angle. Velocity is the fundamental property for measuring movement, and steering angle is used to measure the performance of turning.

We verify the effectiveness of supplementary data from both qualitative and quantitative perspectives. In qualitative experiments, we randomly select the trajectories of vehicles with different motion plannings passing through green lights and no neighbor interactions as input data from VTP-TL [39]. Using the same input, we simulate two different situations of ground truth

![Table I](image)

**Table I**

| Scenario     | $u_{init}^{x}$ | $u_{init}^{y}$ | $u_{init}^{z}$ | $u_{init}^{x}$ | $u_{init}^{y}$ |
|--------------|----------------|----------------|----------------|----------------|----------------|
| Car          | 1.0            | 1.0            | 1.5            | 1.0            | 1.0            |
| Pedestrian   | 0.5            | 1.0            | 1.5            | 1.3            | 1.0            |
| Bicycle      | 1.0            | 1.0            | 2.0            | 1.5            | 1.0            |
| Crossroad-1  | Car            | 1.0            | 1.0            | 1.5            | 1.0            |
| Pedestrian   | 0.5            | 1.0            | 1.5            | 1.3            | 1.0            |
| Bicycle      | 0.8            | 1.0            | 2.0            | 1.5            | 1.0            |
| Crossroad-2  | Car            | 1.0            | 1.0            | 1.5            | 1.0            |
| Pedestrian   | 0.5            | 1.0            | 1.5            | 1.3            | 1.0            |
| Bicycle      | 1.0            | 1.0            | 2.0            | 1.5            | 1.0            |
| T-junction   | Car            | 1.0            | 1.0            | 1.5            | 0.8            |

![Fig. 8](image)

Fig. 8. Runtime performance. We analyze the runtime performance of our simulation results in the crossroad scenario. The specific computation time per frame is shown and also the proportion of time spent on computing agent interactions in the total time.
Experimental results show that simulations with supplementary data outperform those without supplementary data. In particular, in the presence of traffic lights and neighbors, vehicles without supplementary data were unable to decelerate to a standstill. In quantitative experiments (Table III), the simulation results using all input data are better than others, simulations using 80% and 60% input data achieve similar results. When only 40% and 20% of the input data are available, the differences between the simulation results and the ground truth is large. The experimental results show that with the decrease of the input data, the distribution differences between the simulation results and ground truth increase gradually.

Both qualitative and quantitative experiments show the effectiveness of supplementary data. There are two main reasons: (i) without the supplementary data, it is impossible to choose velocities that are not included in the input data. (ii) limited energy functions and constraints cannot fully model complex real-world environments. Both of these lead the fact that the velocity which minimizes the optimization function is only the numerically optimal solutions, but it not necessarily the most appropriate to update the state of agent.

In steering angles Fig. 9(b) and (d), the distribution of our method at zero degree is significantly higher than ground truths. In real world, vehicles have more flexible trajectories. Especially, vehicles whose motion planning is straight usually do not move strictly in a straight line. However, in our method, vehicles have a high probability of choosing the same velocity in successive time steps.

The supplementary data of our method is generated by the velocity continuity model. In addition, we carry out ablation experiments in different range of velocity space. The experimental results are shown in Table IV. We set a variety of size and direction ranges as well as that of intervals for the velocity continuity model. The experimental results show that the best simulation results can be obtained under \( \psi_m = 1.0, I_m = 0.25, \psi_d = 0.9, \) and \( I_d = 0.3. \)

2) Results With Different Energy Terms: We perform ablation experiments to verify the effectiveness of each energy terms.
The distribution differences between simulation results and ground truth in velocity and steering angle are shown in Table V.

Experimental results illustrate that the absence of each energy term will significantly affect the simulation results. The results missing any energy term get higher scores, which indicates that their distributions of velocity and steering angle are more different from the ground truth. In the experiment without the velocity term $E_v$, we set a default value 0.2 for the weight $w_v$, and the agent can move normally but may not reach the desired speed. Therefore, its score is less different from that contains all energy terms. However, there will be a sudden acceleration/deceleration or steering behavior when the agent’s speed and direction need to be changed. The distribution difference score of steering angle increases significantly in experiment without the direction guidance term $E_d$. The simulation results show that the agents deviate from their destination: (i) The agents whose motion planning is steering can only move in a straight line parallel to the direction of their starting road. (ii) The agents with interactive behavior can only swerve to avoid collisions and then continue to move in a straight line. When the agents lack the motion planning-aware collision avoidance term $E_a$, the difference in distribution of the velocity increases significantly. The agents cannot avoid an impending collision or stop at a red light.

3) Results With or Without Parameter Adjustment: We further conduct ablation experiments to verify the performance of our parameter adjustment. We set three different sets of parameters for the vehicles to represent different user preferences for simulation details: agents have weak constraints on the velocity optimization energy term (G-1); agents have relatively balanced constraints on each optimization energy term (G-2); agents are more sensitive to collision avoidance (G-3). For each set of parameters, we simulate the same input data as in Section VI-B1 with or without our parameter adjustments, respectively. The difference between each set of simulation results and the ground truth in the distribution of velocity and steering angle are shown in Table VI.

By comparing the scores in Table VI, it shows that the results with our parameter adjustment are closer to the ground truth distribution. During the simulation, our method dynamically changes the weight $w_d$ of the direction guidance term $E_d$ according to the agent’s current position and destination, which enables the agent to move normally at irregular intersections without predefined lanes. In particular, when $w_d^{init}$ is greater than $w_d$, some vehicles may not be able to steer without parameter adjustment. When agents interact with each other, we dynamically change the weights in the velocity term $E_v$ according to the characteristics of different types of agent interaction behavior, which makes the agent’s behavior more realistic. Moreover, their trajectories are more flexible due to the different parameters of each agent.

The time cost of our parameter adjustment mainly comes from calculating the collision avoidance energy between interacting agents. We only use the agents’ current velocities for computation, and the number of their neighbors is limited. In a crossroad scene with forty vehicles, the maximum computation time for our parameter adjustment is 0.088 ms, and the average computation time is 0.024 ms. Therefore, the time to compute the parameters has a negligible computational cost at each time step.

4) Results With or Without Planning-Aware Collision Avoidance: We intuitively compare the trajectories of agents to evaluate the performance of planning-aware collision avoidance. Among the vehicles that meet in the central area of the intersection, we select two circumstances that most intuitively reflect the impact of motion planning on collision avoidance: (i) straight vehicles from the opposite directions; (ii) vehicles planned to turn left and right respectively. We use motion planning-aware collision avoidance and collision avoidance only based on safety distance to simulate and draw their tracks respectively (Fig. 10).
Fig. 10. Trajectories of different collision avoidance strategies. Vehicles meet in the central area of the intersection: (a) Straight vehicles from the opposite directions; (b) vehicles planned to turn left and right respectively.

Fig. 11. Results with or without dataset division. We compare simulation results with ground truth using dataset division, without dataset division, without dataset division but a larger search range, respectively.

5) Results With or Without Dataset Division: We perform ablation experiments to verify the effectiveness of dataset division. The three simulation results are compared with statistical distribution of the ground truth in the crossroad scenario. The candidate velocity dataset is divided in the first simulation. All velocity data in the second and the third simulation is stored together, respectively. The difference is that the third simulation has a larger search range. Candidate velocities are sorted by the magnitude.

The experimental results are shown in Fig. 11. Compared with the ground truth, the difference scores of velocity distribution of three simulation results are 0.4112: 0.5609: 0.5428, and the difference scores of steering angle distribution are 0.7593: 0.9714: 0.8974. Better simulation result is obtained after dataset division. As mentioned in Section VI-A, we use the acceleration scheme proposed in [23] to ensure the runtime performance of simulation. The search space of the agent is limited, whether or not dataset division is used. After the dataset division, there are more velocities in the search space of the agent at the current time that may be used to update its motion state. Although the performance can be improved by expanding the search range in the simulation without dataset division, evaluating too many candidate velocities at each timestep will lead to poor runtime performance.

6) Comparisons With Other Methods: Previous methods mainly focus on freeways, which are difficult to be directly used to simulate intersection scenarios.

We compare our experimental results (in the crossroad scenario) with Heter-Sim [23] and SUMO [11]. The former is a state-of-the-art data-driven simulation method that can simulate heterogeneous multi-agent systems in different scenarios. Our method uses the same data-driven scheme as it. The latter is the famous model-driven microscopic traffic simulator, which supports users to customize traffic scenarios and simulate heterogeneous agents.

We randomly select two sets of trajectories with the same number from the crossroad scenario, including various types of agents from different directions. One set is randomly selected as the input data of our method and Heter-Sim [23]. All methods are run twice to simulate two groups of ground truth. We initialize agents with the same number and locations as in the dataset. The desired speed is the average speed of the corresponding agent in the dataset. Our method uses the weights of the crossroad scene in Table II, and we make some adjustments to the parameters in the Heter-Sim [23] to fit our scenario.

The various distributions for all methods are shown in Fig. 12, and the detailed scores are shown in Table VII. The velocity and steering angle distributions in our simulations are closer to the ground truth. In the above two simulation results, our method and SUMO [11] have no obvious fluctuations. However, the
scores of Heter-Sim [23] changed significantly when simulating non-input data. Because it is a completely data-driven method, agents can only choose velocities from the input data to update their own motion state. If the velocity of a vehicle in the non-input data is significantly different from the magnitude or direction of all the candidate velocities, the agent may not be able to simulate predetermined behavior. Due to the combination with velocity continuity model, our method can overcome this limitation.

In particular, both Heter-Sim [23] and SUMO [11] need predefined lanes. The former needs the assistance of the lane to guide the agent. In the latter, the agents pass in queues strictly along the lane line. Our method considers the motion planning in collision avoidance, and can dynamically adjust the the weight of terms in the energy function (8) according to the surrounding during the simulation. Therefore, our method can plausibly simulate the movement of the agent in the intersection scene without predefined lanes in the central area.

C. User Studies

To more intuitively evaluate our method, we design two paired comparison user studies. We generate traffic flows for different scenarios using our method and Heter-Sim [23], respectively. For each simulation we capture the traffic animation separately using street view Fig. 13(a) and agent view Fig. 13(b). The agent view can make the viewing experience more immersive for the user.

We use four intersectional scenarios for traffic simulations: 4-lane crossroad, 2-lane crossroad, T-junction, and Y-junction. We simulate these four traffic flows at two different crossroads (Crossroad-1, Crossroad-2), two of which are vehicles only, and the other two are mixed with heterogeneous agents. In the T-Junction scenario, all heterogeneous agents enter the intersection from different lanes. And in the Y-Junction scenario, only the vehicles are driving in the scene. For our method, the parameters in Table I are used for different scenarios. For Heter-Sim [23], the parameters used in different scenarios are listed in Table VIII. The input data used by the two methods in different scenarios is consistent with Section VI-B6. That is, we generate 24 traffic intersection animations and 12 comparison pairs for this user study.

Fifty participants are recruited for our user study, all of them are graduate students from the same university. In each comparative group of user studies, we use the 7-point Likert scale to ask participants to rate the simulation results from 1 to 7. The score indicates a preference for the plausibility of the results, 1 indicates that the results on the left are completely plausible, 7 indicates that the results on the right are completely plausible, and 4 indicates that there is no preference for the results on both sides. Fig. 14 shows the detailed scores of each comparison pair.

The street view study: The detailed scores of fifty participants for the street view study are shown in Fig. 14(a). In this figure, in addition to the score of each scene, we give the total voting results. It can be seen from the figure that participants generally have higher evaluation of our method. Further, we perform the one-sample t-test to quantify the statistical significance of the
Participants think that although our method has no collision in the street view study, our motion planning-aware collision avoidance scheme reduces the sensitivity of agents to some of their neighbors. However, since our motion planning-aware collision avoidance method can be used to eliminate the noise in the input data, and it can also be extended to other works to improve the quality of the sampled data. Our method can ensure that agents obey microscopic traffic rules (e.g., safe gaps, traffic lights) and maintain movement characteristics consistent with the real world in the intersection area where lanes are not clearly defined. Instead of only considering the inter-agent relative distance, we inject the motion planning into the collision avoidance energy function, which is more in line with the logic of real-world drivers to make interactive decisions. Furthermore, we adaptively change the parameters during the simulation with a set of benchmark parameters. This ensures that homogeneous agents have the same movement patterns without losing individual flexibility of the agent.

There are still some limitations of our approach. Our data supplement scheme is based on the continuity of the agent’s velocity at the previous time step, which is sometimes imprecise. Another limitation of our method is that it is unable to simulate large-scale traffic scenes at an interactive rate. As we analyzed in Section VI-A, when a large number of agents enter the intersection area, we need to spend a lot of time computing the interactions between agents. In the future, we plan to introduce parallel implementation and GPU-acceleration to speed up our method.

There are still many ways to extend our method in the future. We plan to extend our method to be editable, which automatically generate a road network based on the topology designed by the user. Algorithmically, in addition to overcome our existing limitations, deep learning can be used to more accurately model the interactions between agents. We also consider adding psychological characteristics or other sensory information such as vision [9] and hearing [53] can also be introduced.

VII. CONCLUSION

We present a novel adaptive and planning-aware hybrid-driven simulation method for traffic intersection scenarios. We show our simulation results in different intersection scenarios. To validate our approach, we conduct comparison experiments and user studies. Experimental outcomes show that our method achieves more realistic results than the state-of-the-art data-driven traffic simulation method.

The quantity and quality of datasets has always been an important factor limiting the performance of data-driven methods. Our hybrid-driven method can generate plausible simulation results with flawed input samples, and velocities that do not exist in the dataset will be supplemented. Our trajectory optimization method can be used to eliminate the noise in the input data, and it can also be extended to other works to improve the quality of the sampled data. Our method can ensure that agents obey microscopic traffic rules (e.g., safe gaps, traffic lights) and maintain movement characteristics consistent with the real world in the intersection area where lanes are not clearly defined. Instead of only considering the inter-agent relative distance, we inject the motion planning into the collision avoidance energy function, which is more in line with the logic of real-world drivers to make interactive decisions. Furthermore, we adaptively change the parameters during the simulation with a set of benchmark parameters. This ensures that homogeneous agents have the same movement patterns without losing individual flexibility of the agent.

ACKNOWLEDGMENTS

The authors would like to thank anonymous reviewers for their constructive suggestions. They would also like to thank specially to Ziqi Zhang for her tireless production assistance in revising our manuscript.
REFERENCES

[1] Q. Zhao, Z. Deng, J. Ren, Q. Ye, and X. Jin, “Realistic data-driven traffic flow animation using texture synthesis,” IEEE Trans. Vis. Comput. Graphics, vol. 24, no. 2, pp. 1167–1178, Feb. 2018.

[2] J. Sewall, J. P. van den, M. C. Berg, and D. Manocha, “Virtualized traffic: Reconstructing traffic flows from discrete spatiotemporal data,” IEEE Trans. Vis. Comput. Graphics, vol. 17, no. 1, pp. 26–37, Jan. 2011.

[3] J. Shen and X. Jin, “Detailed traffic animation for urban road networks,” Graph. Model., vol. 74, no. 5, pp. 265–282, 2012.

[4] W. Li, D. Wolinski, and M. C. Lin, “City-scale traffic animation using statistical learning and metamodel-based optimization,” ACM Trans. Graph., vol. 36, no. 6, pp. 200:1–200:12, 2017.

[5] Q. Chao, X. Jin, H. Huang, S. Foong, L. Yu, and S. Yeung, “Force-based heterogeneous traffic simulation for autonomous vehicle testing,” in Proc. IEEE Int. Conf. Robot. Autom., 2019, pp. 8289–8304.

[6] I. Karamouzas, B. Skinner, and S. J. Guy, “Universal power law governing pedestrian interactions,” Phys. Rev. Lett., vol. 113, no. 23, 2014, Art. no. 238701.

[7] J. P. van den, M. C. Berg, and D. Manocha, “Reciprocal velocity obstacles for real-time multi-agent navigation,” in Proc. IEEE Int. Conf. Robot. Autom., 2008, pp. 1924–1935.

[8] J. Van den, J. Berg, M. C. Guy, and D. Manocha, “Reciprocal n-body collision avoidance,” in Proc. 14th Int. Symp. Robot. Res., 2011, pp. 3–19.

[9] J. Ondrej, J. Pettré, A. Olivier, and S. Donikian, “A synthetic-vision based steering approach for crowd simulation,” ACM Trans. Graph., vol. 29, no. 4, pp. 123:1–123:9, 2010.

[10] P. A. Pérez et al., “Microscopic traffic simulation using SUMO,” in Proc. IEEE Intell. Transp. Syst. Conf., 2018, pp. 2575–2582.

[11] “Mit intelligent transportation systems,” 2011. [Online]. Available: its.mit.edu

[12] D. Wilkie, J. Sewall, and M. C. Lin, “Flow reconstruction for data-driven traffic animation,” ACM Trans. Graph., vol. 32, no. 4, pp. 89:1–89:10, 2013.

[13] W. Li, D. Nie, D. Wilkie, and M. C. Lin, “Citywide estimation of traffic dynamics via sparse GPS traces,” IEEE Univ. Symp. Robot. Syst., vol. 9, no. 3, pp. 100–113, Fall 2017.

[14] W. Li, M. Jiang, Y. Chen, and M. C. Lin, “Estimating urban traffic states using iterative refinement and wardrop equlibria,” IET Intell. Transport Syst., vol. 12, no. 8, pp. 875–883, 2018.

[15] A. Sarkar, K. Czarniecki, M. Angus, C. Li, and S. Yeung, “Traffic self-organized behavior: A large-scale experiment,” in Proc. IEEE Int. Conf. Robot. Autom., 2017, pp. 1–8.

[16] X. Huang, S. G. McGill, B. C. Williams, L. Fletcher, and G. Rosman, “Probability-aware trajectory prediction at urban intersections through learned interaction dynamics via sparse GPS traces,” IEEE Trans. Intell. Transp. Syst., vol. 20, no. 7, pp. 2335–2348, Jul. 2019.

[17] V. van Tol and J. Pettré, “Algorithms for microscopic crowd simulation: Advancements in the 2010 s,” Comput. Graph. Forum, vol. 40, no. 2, pp. 731–754, 2021.

[18] D. Helbing and P. Molnar, “Social force model for pedestrian dynamics,” Phys. Rev. E, vol. 51, no. 4, 1995, Art. no. 4282.

[19] D. Helbing, I. Farkas, and T. Vicsek, “Simulating dynamical features of escape panic,” Nature, vol. 407, no. 6803, 2000, Art. no. 2000.

[20] Y. Han, Q. Chao, and X. Jin, “A simplified force model for mixed traffic simulation,” Comput. Animation Virtual Worlds, vol. 32, no. 1, p. e1974, 2021.

[21] I. Karamouzas, N. Sohre, R. Narain, and S. J. Guy, “Implicit crowds: Optimizing integral for robust crowd simulation,” ACM Trans. Graph., vol. 36, no. 4, pp. 136:1–136:17, 2017.

[22] F. Durupinar, U. Güdükbay, A. Aman, and N. I. Badler, “Psychological parameters for crowd simulation: From audiences to mobs,” IEEE Trans. Vis. Comput. Graphics, vol. 22, no. 9, pp. 2145–2159, Sep. 2016.

[23] J. H. Park, F. A. Rojas, and H. S. Yang, “A collision avoidance behavior model for crowd simulation based on psychological findings,” Comput. Animation Virtual Worlds, vol. 24, no. 3/4, pp. 173–183, 2013.

[24] S. J. Guy, S. Kim, M. C. Lin, and D. Manocha, “Simulating heterogeneous crowd behaviors using personality trait theory,” in Proc. - SCA: ACM SIGGRAPH/Eurographics Symp. Comput. Animation, 2011, pp. 43–52.

[25] B. Zhou, X. Wang, and X. Tang, “Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 2012, pp. 2871–2878.

[26] D. Yang, X. Zhou, G. Su, and S. Liu, “Model and simulation of the heterogeneous traffic flow of the urban signalized intersection with an island work zone,” IEEE Trans. Intell. Transp. Syst., vol. 20, no. 5, pp. 1719–1727, May 2019.

[27] S. Biringer et al., “Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset,” in Proc. IEEE Int. Conf. Comput. Vis., 2021, pp. 9690–9699.

[28] Q. Chao et al., “A survey on visual traffic simulation: Models, evaluations, and applications in autonomous driving,” Comput. Graph. Forum, vol. 39, no. 1, pp. 287–308, 2020.

[29] Y. Wang, M. Kapadia, S. Huang, L. Kavan, and N. I. Badler, “Sound localization and multi-modal steering for autonomous virtual agents,” in Proc. Symp. Interactive 3D Graph., 2014, pp. 23–30.
Pei Lv received the PhD degree from the State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China, in 2013. He is an associate professor with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China. His research interests include computer vision and computer graphics. He has authored more than 50 journal and conference papers in the above areas, including the IEEE Transactions on Image Processing, the IEEE Transactions on Circuits and Systems for Video Technology, the IEEE Transactions on Affective Computing, CVPR, ECCV, ACM MM et al.

Xinming Pei received the MSc degree from the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China, in 2023. He is currently a research assistant with the State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China. His research interests include physics-based simulation and crowd simulation.

Xinyu Ren received the BSc degree in software engineering, Zhengzhou University, China, in 2021. She is currently working toward the MSc degree with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China. Her main research interest is traffic simulation.

Yuzhen Zhang received the BSc and MSc degrees in software engineering from Henan Polytechnic University, Jiaozuo, China. She is currently working toward the PhD degree in School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China. Her current research interests include machine learning, computer vision and their applications to motion prediction, scene understanding, and interaction modeling for intelligent autonomous systems.

Chaochao Li received the PhD degree from the School of Information Engineering, Zhengzhou University, Zhengzhou, China. His current research interests include computer graphics and computer vision. He is currently an assistant research fellow with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China. He has authored more than 6 journal and conference papers including the IEEE Transactions on Affective Computing, IEEE Transactions on Intelligent Transportation Systems, and IEEE Transactions on Systems, Man, and Cybernetics: Systems.

Mingliang Xu received the PhD degree in computer science and technology from the State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China, in 2012. He is a full professor and the director with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China. His research interests include computer graphics, multimedia, and artificial intelligence. He has authored more than 100 journal and conference papers in the above areas, including the ACM Transactions on Graphics, the ACM Transactions on Intelligent Systems and Technology, the IEEE Transactions on Pattern Analysis And Machine Intelligence, the IEEE Transactions on Image Processing, the IEEE Transactions on Cybernetics, the IEEE Transactions on Circuits and Systems for Video Technology, ACM SIGGRAPH (Asia), ACM MM, and ICCV.