Differentiable Hybrid Traffic Simulation

SANGHYUN SON, University of Maryland, College Park, USA
YI-LING QIAO, University of Maryland, College Park, USA
JASON SEWALL, NVIDIA, USA
MING C. LIN, University of Maryland, College Park, USA

We introduce a novel *differentiable hybrid traffic simulator*, which simulates traffic using a hybrid model of both macroscopic and microscopic models and can be directly *integrated into a neural network for traffic control and flow optimization*. This is the first differentiable traffic simulator for macroscopic and hybrid models that can compute gradients for traffic states across time steps and inhomogeneous lanes. To compute the gradient flow between two types of traffic models in a hybrid framework, we present a novel intermediate conversion component that bridges the lanes in a differentiable manner as well. We also show that we can use analytical gradients to accelerate the overall process and enhance scalability. Thanks to these gradients, our simulator can provide more efficient and scalable solutions for complex learning and control problems posed in traffic engineering than other existing algorithms. Refer to https://sites.google.com/umd.edu/diff-hybrid-traffic-sim for our project.

1 INTRODUCTION

Automobile traffic is a dynamic phenomenon that emerges from the interactions of a multitude of distinct entities, and its complexity has led analysts to rely on simulations to solve complex real-world traffic problems, such as signal control, congestion, road design, and urban planning [Lieberman and Rathi 1997]. Traffic simulation has become increasingly more important due to population growth, fast-growing technologies, and demands for autonomous driving. Autonomous driving agents need accurate and efficient models to simulate all types of plausible traffic scenarios to capture corner cases and accelerate learning-based training that requires data not easily available from real-world capturing [Akhauri et al. 2020].

Techniques for traffic simulation can be broadly classified as either *macroscopic* or *microscopic* based on their modeling assumptions.

Authors' addresses: Sanghyun Son, University of Maryland, College Park, USA, shh1295@umd.edu; Yi-Ling Qiao, University of Maryland, College Park, USA, yilingq@umd.edu; Jason Sewall, NVIDIA, USA, jasonsewall@gmail.com; Ming C. Lin, University of Maryland, College Park, USA, lin@umd.edu.

Fig. 1. **Traffic simulation in an urban environment.** Traffic simulation can be used to analyze complex traffic dynamics. Among two popular traffic models, by simulating areas of interest with the microscopic one and using macroscopic simulation elsewhere, we can reduce the overall computational cost without compromising significant details. For instance, the center of intersection can be simulated with discrete, agent-based models for higher-fidelity inter-vehicle dynamic interaction. (*This scene is rendered in CARLA* [Dosovitskiy et al. 2017].)

Macroscopic models describe traffic evolution as a system of partial differential equations (PDE), and the traffic state is represented as a *continuum* of values across the road network. Since we treat vehicles as particles translated by convection under this assumption, it is computationally efficient but provides coarse simulation results. In contrast, microscopic models represent traffic through individual vehicles, or agents, that are evolved individually and which together characterize the traffic state of the road network [Sewall et al. 2011]. Therefore, this approach gives us fine-grained details but often requires much more computation resources than macroscopic models.

In this work, we propose to adopt both microscopic and macroscopic models to create a more general traffic simulator that is differentiable (Figure 1). This kind of hybrid approach has already been proven to simulate large-scale traffic environments much more efficiently than either of the approaches while maintaining integrity [Sewall et al. 2011]. Here we leverage the power of the hybrid approach to support large-scale traffic scenes, which will be demonstrated in our experiments. During simulation, our framework computes gradient information that provides abundant insight into traffic dynamics. This approach results in enhanced sample efficiency in our simulation, as each sample of traffic simulation comes with gradient information explaining how and why such an event has occurred. Moreover, this differentiability allows us to integrate our framework with neural networks to support a wide range of traffic control, planning, management, and flow optimization problems.

We first derive the analytical formulation of the differentiable traffic models for both macroscopic and microscopic paradigms. The resulting representations can simulate traffic on road networks for

any metropolitan region more efficiently. Furthermore, we present a probabilistic method to convert traffic states between macroscopic and microscopic ones, which is also differentiable. With this novel differentiable traffic simulation model, we can find near-optimal solutions to some traffic problems that were otherwise not previously solvable, and we can do so efficiently. To summarize, in this paper we introduce the following main results:

- Derivation of a differentiable macroscopic traffic model based on ARZ model [Aw and Rascle 2000; Zhang 2002] (Sec. 3);
- Analytical formulation of differentiable Intelligent Driver Model [Treiber et al. 2000] for agent-based vehicle dynamics (Sec. 4);
- Differentiable conversion between deterministic and probabilistic instantiation of two traffic models (Sec. 5);
- Application of the first differentiable hybrid traffic simulation framework to traffic control (Sec. 6).

In our experiments, we observe up to *an order of magnitude* speedup in runtime performance, in both forward simulation and back propagation process, compared to a baseline differentiable simulator based on automatic differentiation.

2 PREVIOUS WORK

2.1 Traffic models

The preponderance of macroscopic models are based on the *density* and *flux* of the traffic flow; [Lighthill and Whitham 1955; Richards 1956] suggested one of the earliest models based on this idea; the LWR model is a non-linear PDE on the density of vehicles. To overcome the limitations of this model, [Payne 1971; Whitham 2011] added a momentum term to the LWR model, which allowed for more complex traffic dynamics than before. However, modeling traffic flow as isotropic [Cassidy and Windover 1995; Daganzo 1995] led to non-physical behavior, such as negative velocity. Based on the observation that traffic flow is anisotropic, [Aw and Rascle 2000; Zhang 2002] modified the momentum term in the previous model. The ARZ model has been used to visually simulate the traffic flow [Sewall et al. 2010], and to solve traffic congestion problems [Yu and Krstic 2019]. Because of the basic assumption, the computational cost of these macroscopic models is proportional to the length of the simulated road, not the number of vehicles therein. Therefore, they are often more computationally efficient than the other models for large-scale environments. However, they are not suitable for simulations where fine details, such as the behavior of individual vehicles, are needed.

Microscopic models describe the motion of individual vehicles, typically tracking velocity, the bumper-to-bumper distance to its leading vehicle, and the relative velocity between them [Kesting et al. 2007]. The research history dates back to [Gazis et al. 1959, 1961] and [Newell 1961]. Modern variations of the model often include behavioral traits of individual vehicles, which make the simulation more realistic and descriptive. [Bando et al. 1995; Gipps 1981; Jiang et al. 2001; Treiber et al. 2000] are such models. These models are widely adopted in current agent-based traffic simulators, such as [Lopez et al. 2018]. Since every individual vehicle observes its surrounding environments and decides its actions in these models, we can get more fine-grained details of vehicle motions than in the

macroscopic models. However, their computational cost is proportional to the number of vehicles in the scene, which makes them harder to be adopted for large-scale simulations.

Hybrid models seek to combine these models in various fashions to take advantage of their complementary properties; [Bourrel and Lesort 2003; Magne et al. 2000; Mammar et al. 2006; Sewall et al. 2011] fall into this category and show the computational gains we can get from these models. This hybridization is commonly a spatial one, with different parts of a network running under distinct regimes. That is, these models often simulate only the regions of interest with microscopic models for accuracy and use macroscopic models elsewhere to maintain the overall flow correctness while attaining computational efficiency. This typically necessitates a technique for converting between different traffic flow representations [Bourrel and Lesort 2003]. In this work, we adopt this hybrid approach to support large-scale scenarios and let our simulator be more general so that users can select the simulation modes based on their needs.

2.2 Differentiable models

Differentiable models have been widely used in graphics and robotics applications like visualization [Li et al. 2018; Nimier-David et al. 2019], design [Cascaval et al. 2021; Du et al. 2020], and control [Heiden et al. 2021]. This paradigm enables gradient information to flow across complicated functions, and facilitates the machine learning process by improving sample efficiency [Mora et al. 2021; Shen et al. 2021]. They often show better results in finding optimal solutions for various high-dimensional problems [Hu et al. 2020; Ma et al. 2021].

Related to traffic simulation, there especially have been a number of differentiable simulations for a variety of systems including rigid bodies [de Avila Belbute-Peres et al. 2018; Qiao et al. 2020], ariticulated bodies [Qiao et al. 2021b], soft bodies [Du et al. 2021; Geilinger et al. 2020; Qiao et al. 2021a], cloth [Li et al. 2022], and fluids [Holl et al. 2020; Takahashi et al. 2021].

Because many traffic models are differentiable in nature, we can apply this technique to traffic simulation and control. Recently, [Andelfinger 2021] has presented a differentiable agent-based traffic simulation, and showed how it can be used to solve traffic signal control problems. However, the scope of this work was limited to microscopic models for traffic, and the implementation relies on automatic differentiation, which can be computationally inefficient. In addition to traffic, this hybrid differentiable simulation framework can potentially generalize to other large-scale, multi-agent systems, such as crowds, insects, fluids, grains, etc. [Colas et al. 2022; Hädrich et al. 2021; He et al. 2020; Ishiwaka et al. 2021] as well.

3 MACROSCOPIC MODEL

In a macroscopic traffic model, traffic evolution is described by a system of Partial Differential Equations (PDEs) on the road network.

3.1 Formulation

The ARZ [Aw and Rascle 2000; Zhang 2002] model describes vehicle flow in a single lane of traffic through the following system of differential equations in one dimension:

$$q_t + f(q)_x = 0, \quad f(q) = qu = \begin{bmatrix} \rho u \\ yu \end{bmatrix} \tag{1a}$$

where $q(x, t) = [\rho(x, t), y(x, t)]^\intercal$; ρ denotes the density of traffic (cars per car length), y is the relative flow of traffic, and u is the velocity of traffic. As per the notation, these terms are observed at a certain position x at a time t, and they are related to each other as follows:

$$y(\rho, u) = \rho\left(u - u_{\mathrm{eq}}(\rho)\right) \tag{1b}$$

$$u_{\mathrm{eq}}(\rho) = u_{\max}\left(1 - \rho^\gamma\right) \tag{1c}$$

where $u_{\max}$ denotes the maximum velocity, or speed limit, of the given lane, and $0 < \gamma < 1$ is a constant tuning parameter. Conceptually, u_{eq} represents a 'comfortable' velocity based on density. We used $\gamma = 0.5$ in all of the experiments.

Note that the system (1a) is a conservative system, where the sum of ρ and y over the entire lane is conserved, excluding the incoming and outgoing fluxes at the borders of the lane.

3.2 Numerical Solution

Conservation laws like the ARZ system of PDEs are typically discretized with the Finite Volume Method (FVM) [LeVeque et al. 2002] and integrated explicitly in time. The discretized quantities in a lane are Q_i^n, where n refers to the time step and i refers to the ith cell. The solution procedure follows:

(1) Compute wave speeds λ and fluxes f (from Eq. (1a)) by solving the Riemann problem at the interface of each adjacent cell.
(2) Compute global time step $\Delta t \leq \max|\lambda|$; in this work we choose a conservative, constant Δt.
(3) Integrate: $Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}\left[f(q(b)) - f(q(a))\right]$. $q(a)$ and $q(b))$ are the intermediate states identified between $\left(Q_{i-1}^n, Q_i^n\right)$ and $\left(Q_i^n, Q_{i+1}^n\right)$ respectively. Δx is the length of cell i.

Details can be found in [Sewall 2011]; Appendix A.1 reproduces the six relevant cases for the ARZ Riemann problems.

3.3 Differentiation

Our differential technique built on this requires analytical gradients of the above; these gradients form the basis of still other gradients that are used in solving more complex traffic control problems.

In our update scheme, we compute the gradients of Q_i^{n+1} using Q_{i-1}^n, Q_i^n, and Q_{i+1}^n:

$$\frac{\partial Q_i^{n+1}}{\partial Q_{i-1}^n} = -\frac{\Delta t}{\Delta x}\left[-f'(q(a))\frac{\partial q(a)}{\partial Q_{i-1}^n}\right] \tag{2a}$$

$$\frac{\partial Q_i^{n+1}}{\partial Q_i^n} = I - \frac{\Delta t}{\Delta x}\left[f'(q(b))\frac{\partial q(b)}{\partial Q_i^n} - f'(q(a))\frac{\partial q(a)}{\partial Q_i^n}\right] \tag{2b}$$

$$\frac{\partial Q_i^{n+1}}{\partial Q_{i+1}^n} = -\frac{\Delta t}{\Delta x}\left[f'(q(b))\frac{\partial q(b)}{\partial Q_{i+1}^n}\right] \tag{2c}$$

We thus need the Jacobian $f(q)$ and the partial derivatives $\frac{\partial q(a)}{\partial Q_{i-1}^n}$, $\frac{\partial q(a)}{\partial Q_i^n}$, $\frac{\partial q(b)}{\partial Q_i^n}$, $\frac{\partial q(b)}{\partial Q_{i+1}^n}$; this arises from differentiating solutions to Riemann problems as shown in Appendix A.2. The analytical gradients computed here play an important role in accelerating our traffic simulator, as shown in Section 6.

3.3.1 *Time step sizes.*
We have derived these gradients under the assumption that Δt is constant and satisfies the CFL and stability conditions [LeVeque et al. 2002] for the entire simulation. It would be possible to have a dynamic Δt, since this is also differentiable (the speed λ is determined by the eigenvalues of the flux function f). However, for the sake of the hybrid approach we ultimately use, we use a constant Δt to ensure that the microscopic simulation is stable.

3.3.2 *Continuity issues.*
We have computed the gradients above assuming that the update scheme is differentiable, and in fact, we can see that it is continuous and piece-wise differentiable across most cases suggested in Appendix A.

We do not prove it thoroughly here, as it is generally trivial when we keep in mind that $\lim_{u_l \to u_r} q_m = q_l$. However, Case 1 is exceptional; there is a shock between the phase states q_l and q_m, which makes the two states discontinuous. Specifically, we cannot guarantee that q_m converges to q_l when $\lambda_s < 0$ converges to 0. Therefore, the gradients we compute when λ_s is near zero could become unstable. However, in most cases in our simulation, these cases rarely happened and did not affect the quality of our solution even when we used the possibly unstable gradients as they are; it is possible that numerical viscosity plays a role here.

4 MICROSCOPIC MODEL

In a microscopic traffic model, traffic flow is described by interactions between multiple discrete vehicles that follow certain rules. Here we use the Intelligent Driver Model (IDM) [Treiber et al. 2000] to simulate such vehicles.

4.1 Formulation

Under the microscopic viewpoint, we can describe the state of a discrete vehicle as

$$q(t) = \begin{bmatrix} p(t) \\ v(t) \end{bmatrix}, \tag{3}$$

where p and v denote the position and velocity of the vehicle respectively for a given time t.

Using these states to describe discrete vehicles occupying a lane, IDM describes the motion of each individual vehicle based on its relationship with the vehicle directly ahead of it; the *leading vehicle*. For the ith vehicle, let $h(i)^{\mathrm{th}}$ its leading vehicle. According to the IDM, the acceleration of the ith vehicle is determined as follows:

$$\Delta p = p_{h(i)}(t) - p_i(t) - length_{h(i)} \tag{4a}$$

$$\Delta v = v_i(t) - v_{h(i)}(t) \tag{4b}$$

$$s_{\mathrm{opt}} = s_{\min} + v_i(t)T_{\mathrm{pref}} + \frac{v_i(t)\Delta v}{2\sqrt{a_{\max}a_{\mathrm{pref}}}} \tag{4c}$$

$$a_i(t) = a_{\max}\left[1 - \left(\frac{v_i(t)}{v_{\mathrm{targ}}}\right)^\delta - \left(\frac{s_{\mathrm{opt}}}{\Delta p}\right)^2\right] \tag{4d}$$

There are various hyperparameters included in the model. They characterize a vehicle's motion; see Appendix B.1 for more details. In our experiments, we randomly initialized those hyperparameters for every single vehicle. Note that we compute s_{opt} before we compute the acceleration term. It represents the *optimal* space that the vehicle should have to avoid collision to its leading vehicle.

With the acceleration $a_i(t)$, we can update the vehicle's state using Euler's method.

4.2 Differentiation

Based on this update scheme, we can compute the analytical gradients of the IDM. Since the derivation is straightforward, we offer the analytical gradients in Appendix B.2. Note that compared to the macroscopic model, there is no significant theoretical challenge that we have to deal with in the IDM.

5 MACRO-MICRO-MACRO CONVERSION

Our traffic simulation can transition between the continuous, flow-based macroscopic model and the discrete, agent-based microscopic model. We need to consider the differentiation across the interfaces.

5.1 Macro to Micro

Given a lane in the macroscopic regime that flows into a microscopic lane, we must convert the continuous representations therein to discrete vehicles at their junction. This can be done in a deterministic or stochastic fashion.

5.1.1 Deterministic Instantiation. Assume that $\rho(t)$ is the density of the traffic flow at the end of a macro lane on time t, and $v(t)$ is the velocity. We can set a *flux capacitor* (c.f. [Sewall 2011]) at the interface; this counts the total number of vehicles $N_d(t)$ that have reached this point:

$$N_d(t) = \left\lfloor \int_0^t \rho(t)v(t)dt \right\rfloor \quad (5)$$

where $\lfloor x \rfloor$ is the floor function that gives the greatest integer less than or equal to the real number x. $N_d(t)$ will then start from 0, and we will instantiate a vehicle when it increases by 1. The velocity of the instantiated vehicle is set to be the macroscopic traffic flow's velocity.

5.1.2 Stochastic Instantiation. Another possible interpretation of the macro to micro conversion is as a Poisson process. The flux of the traffic flow can be viewed as the intensity of a Poisson process. In each time step of length dt, the number of instantiated vehicles $X(t)$ follows a Poisson distribution $X(t) \sim P(\rho(t)v(t)dt)$, $P(X = k) = e^{-\rho(t)v(t)dt}(\rho(t)v(t)dt)^k k!^{-1}$. Similarly, the total number of vehicles $N_s(t)$ can be expressed as

$$N_s(t) = \int_0^t X(t) \quad (6)$$

As above, the velocity of the emitted discrete vehicle is the velocity of the macroscopic state at the junction of the two lanes.

5.2 Micro to Macro

Compared to the macro to micro conversion, agent-based information can be converted to a continuum representation more simply. The density of a cell at the interval $(l, r]$ in a macro lane can be defined as

$$\rho(l, r) = \frac{1}{r - l} \sum_{i=1}^{n} 1_{(l,r]}(p_i) \quad (7)$$

where $1(\cdot)$ is the indicator function that identifies vehicles that lie in the interval, n is the number of vehicles and p_i is the position of ith

vehicle. The velocity of this cell can be the average velocities of all the vehicles therein.

5.3 Differentiation

We have described two types of discrete processes: instantiation of vehicles, and the indicator function. Automatic differentiation is able to compute the gradients of velocities v but cannot handle the gradients of densities ρ. To backpropagate derivatives to the densities ρ, we first create an ancillary variable $a_i = 1$ for each discrete vehicle. We rewrite Equation 7 as:

$$\rho(l, r) = \frac{1}{r - l} \sum_{i=1}^{n} 1_{(l,r]}(p_i) \cdot a_i \quad (8)$$

Since we always set $a_i = 1$, the equation constantly holds for forward simulation. But a_i can also receive the gradients for discrete vehicles from the following macro lanes. Let L be the loss function; then for each vehicle in $(l, r]$:

$$\frac{\partial L}{\partial a_i} = \frac{\partial L}{\partial \rho(l, r)} \quad (9)$$

The next challenge for differentiation is the vehicle instantiation process in the macro to micro conversion. We will start with the deterministic strategy: assume that the total number of vehicles reach n and $n + 1$ at time t_1 and t_2, respectively. We therefore have the following equation

$$n + 1 = n + \int_{t_1}^{t_2} \rho(t)v(t)dt = n + a_i \quad (10)$$

where the ith vehicle is the one instantiated at t_2. So for $t \in (t_1, t_2]$,

$$\frac{\partial L}{\partial \rho(t)} = \frac{\partial L}{\partial a_i} v(t)dt \quad (11)$$

For the stochastic case, we can first compute the expectation value of the total number of vehicles according to the Poisson distribution

$$E[N_s(t)] = \int_0^t E[X(t)] = \int_0^t \rho(t)v(t)dt. \quad (12)$$

We notice that the expectation value of $N_s(t)$ is the same as the flux capacitor in the deterministic strategy. Similarly, we can find time t_1 and t_2 when the total number of vehicles reach n and $n + 1$. For all time step $t \in (t_1, t_2]$, $\frac{\partial L}{\partial \rho(t)} = \frac{\partial L}{\partial a_i} v(t)dt$. Intuitively, if $\frac{\partial L}{\partial a_i} \leq 0$, it means the density is higher than the desired level and we need to decrease the intensity of the Poisson process. We will also validate in our numerical experiments that such estimation of gradients can effectively optimize our objective function.

6 EXPERIMENTAL RESULTS

We show the effectiveness of our differentiable hybrid traffic simulator with application to solve a variety of traffic problems. We first show that we can accelerate the differentiable traffic simulator with the analytical gradients that we have computed above by comparing it against a baseline simulator using automatic differentiation. At the same time, we will justify the use of the hybrid model by comparing its computation time with other approaches in large-scale scenarios. Then, we prove the correctness and efficacy of our analytical gradients by solving parameter estimation problems. Lastly, we conduct experiments with traffic control problems to illustrate how we can

integrate our simulator with neural networks and how we can use our simulator to solve real-world traffic problems.

We have implemented our traffic simulator with Python, and used PyTorch 1.9 [Paszke et al. 2019] for automatic differentiation. All experiments were run on an Intel® Xeon® W-2255 CPU @ 3.70GHz, and traffic rendering was mainly based on our own implementation on Unity Engine.

6.1 Acceleration

6.1.1 Analytical Gradients. Here we present how our analytical gradients can contribute to the acceleration of both forward simulation and backward propagation in our simulator. We have compared our simulator to the baseline simulator that relies on automatic differentiation to compute the gradients in the system. For all of the simulations, we have computed the gradients of the states at the last time step with respect to the states at the initial time step. For macroscopic and microscopic simulations, we used a single lane. For hybrid simulation, we used 3 lanes, which is the minimum case to cover both macro-to-micro and micro-to-macro conversion. The first and third lanes are set as macroscopic lanes and the second one is a microscopic lane. To show how the computation time changes with the scale of the simulation, we have experimented with different scales of settings.

Table 1 shows that our analytical gradient allows us to run both forward simulation (FW) and back propagation (BP) of the system much faster than the baseline framework. In macroscopic and hybrid simulations, the speedup was up to 4x times for forward simulation, and 18x for back propagation. In the microscopic simulation, the speedup was up to 6x times for forward simulation, and 7x times for back propagation. This shows the computational efficiency of our framework, which is necessary to implement large-scale traffic simulation.

6.1.2 Hybrid Model. In this paper, we selected the hybrid model to run large-scale traffic simulations in interactive time, while not losing their fidelity. To justify this design choice, here we present the average frame rate for simulating a large environment, which includes approximately 10K vehicles. Note that we can specify the ratio ϵ of the vehicles which would be simulated using a microscopic model; ϵ could be regarded as an interpolant between the two models, where $\epsilon = 0$ means macroscopic and $\epsilon = 1$ means microscopic approach. We measured the average frame rate for different ϵ using single core; see Figure 2. Note that the average frame rates decreases as ϵ increases and thus the simulation becomes more "microscopic".

6.2 Parameter Estimation Problem

One of the most fundamental problems that we can solve with our framework is the parameter estimation problem. There are many parameters involved in our system; the speed limit is the most typical example. In our formulation, we focus on retrieving preceding states in the system based on given subsequent ones. The simplest form of this problem would be, assuming $0 \leq t \leq 1$, estimating the initial state q(0) given the last state q(1). This problem is considered to be the most basic formulation to solve, as it only requires fundamental gradients computed analytically.

Table 1. Comparison of our technique against automatic differentiation for various types of simulations. Scale denotes number of time steps taken and the number of cells (for macroscopic, and hybrid)/ number of vehicles (microscopic). FW: forward simulation; BP: backpropagation. Ours outperforms autodifferentiation in both forward passes and backpropagation, by up to a factor of **18.77, 7.84, and 15.27** in *macroscopic, microscopic*, and *hybrid* traffic simulation, respectively.

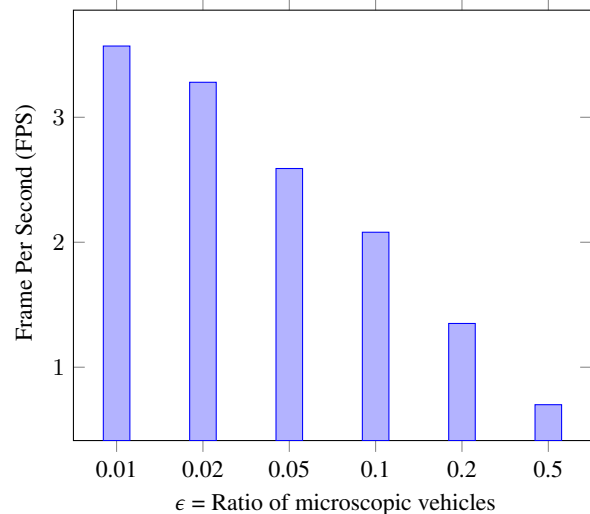|	Scale	10/1K	50/5K
Macroscopic	FW(Auto)	5.28s±0.16s	125.88s±5.66s
	FW(Ours)	1.42s±0.04s	30.06s±0.88s
	Speedup	3.71x	4.19x
	BP(Auto)	2.42s±0.20s	71.08s±6.01s
	BP(Ours)	0.18s±0.03s	3.79s±0.25s
	Speedup	**13.73x**	**18.77x**
Microscopic	FW(Auto)	1.54s±0.08s	66.74s±1.67s
	FW(Ours)	0.45s±0.02s	10.86s±0.24s
	Speedup.	3.42x	6.15x
	BP(Auto)	1.58s±0.06s	45.36s±0.95s
	BP(Ours)	0.20s±0.03s	7.10s±0.42s
	Speedup	**7.84x**	**6.39x**
Hybrid	FW(Auto)	11.81s±0.42s	310.67s±15.02s
	FW(Ours)	3.41s±0.06s	66.52s±0.47s
	Speedup	3.47x	4.67x
	BP(Auto)	2.70s±0.11s	111.06s±3.09s
	BP(Ours)	0.38s±0.02s	7.27s±0.13s
	Speedup	**7.13x**	**15.27x**

Fig. 2. Comparison on the average frame rate (FPS) under the hybrid model, with different ϵ. The total number of vehicles is set to be 10K, and the ϵ value denotes the ratio of microscopic vehicles among them.

In our experiments, the simulation length was set as 10 seconds for every setting. Theoretically, we can use any simulation lengths for this experiment, but we found out that too long simulation lengths

(a) Macroscopic (b) Microscopic

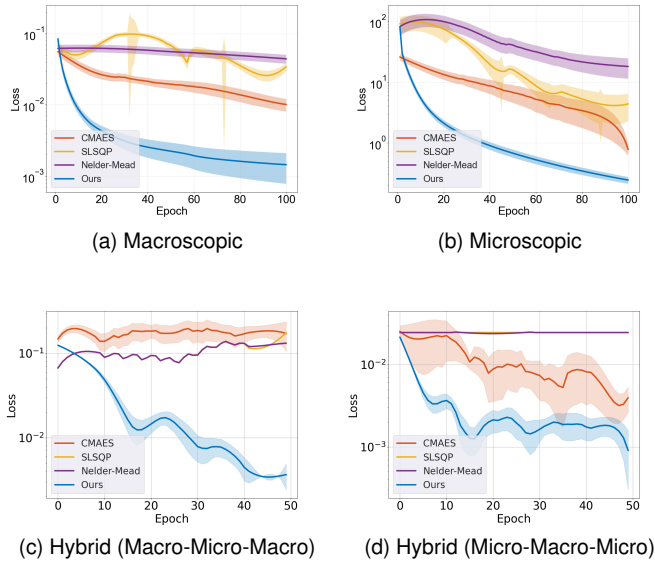(c) Hybrid (Macro-Micro-Macro) (d) Hybrid (Micro-Macro-Micro)

Fig. 3. **Parameter estimation problem.** For given final states of the system, one has to estimate the initial states that lead to the final states. (a)–(d) Our results rendered in blue exhibit much faster convergence to the correct solutions than the other state-of-the-art gradient-free optimization methods by up to *one order of magnitude*.

make this problem much harder than it should be and 10 seconds is a proper length to prove our approach's efficacy. Under this setting, we have to estimate the initial traffic states that would end up in the given final traffic states in 10 seconds. For a proposed estimate $q_{est}(0)$, we can compute the estimation error with the following loss function, where n denotes the size of the state vector:

$$L = \frac{1}{n} |q(1) - q_{est}(1)|^2$$

We have compared our framework with other gradient-free optimization algorithms: CMA-ES [Hansen 2006], SLSQP [Kraft et al. 1988], and Nelder-Mead [Nelder and Mead 1965] algorithm. Experiments were run five times with randomly initialized initial states for each of the algorithms and for each of the simulation modes. In the hybrid setting, we used 3 sequentially connected lanes. One of the experiments was done for macro-micro-macro lanes, and the other was done for micro-macro-micro lanes. The loss graph for each setting is shown in Figure 3. We can observe that our framework retrieves more precise initial states faster than other algorithms for every setting.

6.3 Traffic Control Problems

Our differentiable traffic simulator can be integrated with neural networks to enhance its ability to learn and control a given task. To test our framework's capability, we have experimented with the "intersection signal control problem" (ITSCP) for macroscopic and hybrid simulation, and the pace car problem for microscopic simulation.

For all the experiments, we used two fully connected layers as our neural network, where each of the layers has 256 nodes. The neural network is trained to emit proper control input, which would
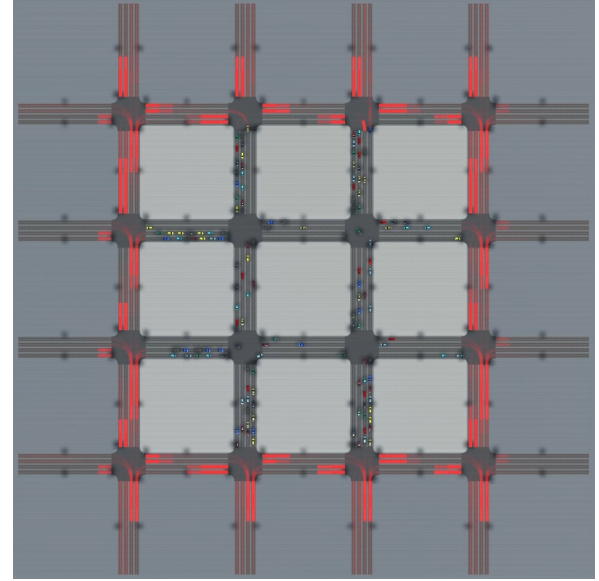
Fig. 4. **4x4 Grid of Intersections.** Hybrid traffic simulation at a 4x4 grid of intersections: the density of each cell is rendered in red, which shows the volume of traffic agents to cross the intersection. One has to optimize time allocations for traffic lights at each intersection to maximize overall traffic flow.
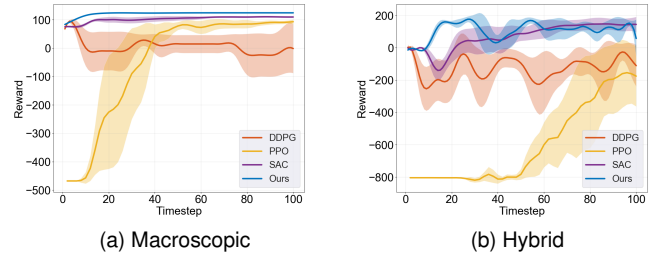

(a) Macroscopic (b) Hybrid

Fig. 5. **ITSCP.** (a), (b) Learning graphs for macroscopic and hybrid simulations. Our framework shows better convergence rates and results than other RL algorithms.

be fed into our simulator. Then we can compute the gradient of the aggregated reward of each episode with respect to each control input therein, and apply gradient descent to directly maximize the reward. Note that this approach is same as analytical policy gradient (APG) method. Then we compare our results to other baseline gradient-free RL algorithms, as they are widely used to solve these kinds of control problems.

6.3.1 ITSCP. ITSCP is one of the most widely studied traffic control problems, because of its significance in controlling traffic congestion in urban environment [Eom and Kim 2020]. We have devised a scenario that falls into this category, where an agent has to compute the optimal time allocations for traffic lights (Figure 4).

In this scenario, there are two different traffic lights at a given intersection: one is green across the West-East direction (WE-light), and the other one is across the North-South direction (NS-light). When one of the lights is turned on, the other one has turned off automatically. In a single signal phase, the WE-light is turned on first,

(a) Time = 0 sec, Initial States

(b) Time = 10 sec, Target Speed Limit = 30 m/sec
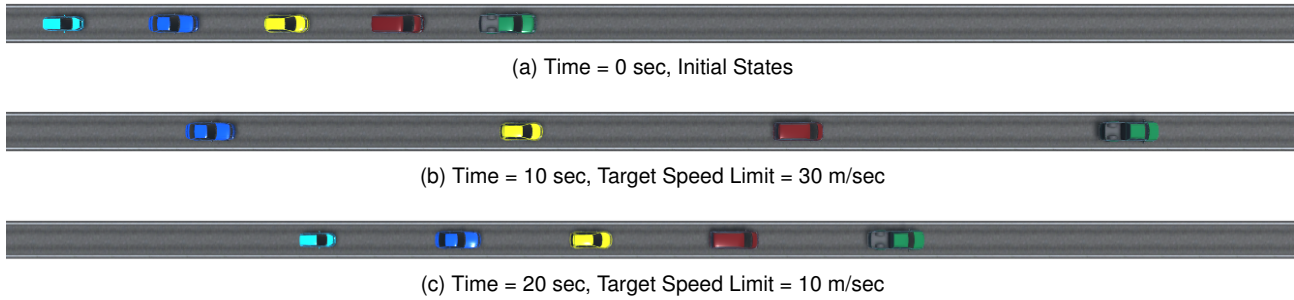

(c) Time = 20 sec, Target Speed Limit = 10 m/sec

Fig. 6. **Pace Car Problem.** Sequence of simulation images in time order, from top to bottom. (a) A pace car rendered in green guides following vehicles to maintain a target speed limit, which varies over time. It fails when it runs too fast, too slow, or goes out of the lane. (b) When a high target speed limit (30 m/sec) is given, the pace car accelerates fast, and keeps a wide distance from its following vehicles to let them run fast. (c) However, when the target speed limit is reduced (10 m/sec), it swiftly decelerates and keeps short distance from the following vehicles, to prevent them from running over the target speed limit.

and then the NS-light is turned on next. There are multiple signal phases during the entire simulation, and the agent has to determine optimal time allocations for each light in each signal phase.

The input to the neural network is given as the schedule of incoming traffic flow for every lane. Our experiments were inspired by a synthetic traffic scenario devised for another ITSCP experiment [Wei et al. 2018]. We also followed a conventional scheme to compute the reward [Eom and Kim 2020]. The reward for an action is computed as the weighted sum of traffic flow that crosses the intersection (R_f) and the length of waiting queue in each lane (R_q), which is formulated as follows:

$$R = c_1 \cdot R_f + c_2 \cdot R_q.$$

For our experiments, c_1 is set as 1 and c_2 is set as -1, to maximize the traffic flow and minimize the length of waiting queue. In macroscopic environments, the length of the queue is computed as the number of cells that have speed less than certain threshold. In hybrid environments, the traffic flow is computed as the number of discrete vehicles crossing the intersection, as the central part of an intersection is simulated with a microscopic model.

For comparison, we used three baseline model-free RL algorithms; DDPG [Lillicrap et al. 2015], PPO [Schulman et al. 2017], and SAC [Haarnoja et al. 2018]. Note that these algorithms do not use gradient information that our simulator provides. In contrast, our gradient-based optimization algorithm uses this information to optimize the objective function. Experiments were run five times for each algorithm, and Figure 5 shows the learning graphs of both macroscopic and hybrid environment. In the macroscopic setting, our framework converged to the near-optimal solution very quickly, while DDPG failed to learn at all and PPO and SAC learned, but not to the extent of ours. In hybrid setting, our framework and SAC both succeeded in converging to the best solution, but ours converged faster than SAC. Also, Table 2 shows that our best reward is better than any other algorithms for both settings.

6.3.2 Pace Car Problem. In this problem, we assume there is a discrete pace car, which runs in front of other vehicles that follow the IDM. The pace car leads the vehicles to enforce a specified target speed limit, which is often smaller than the lane's speed limit (Figure 6).
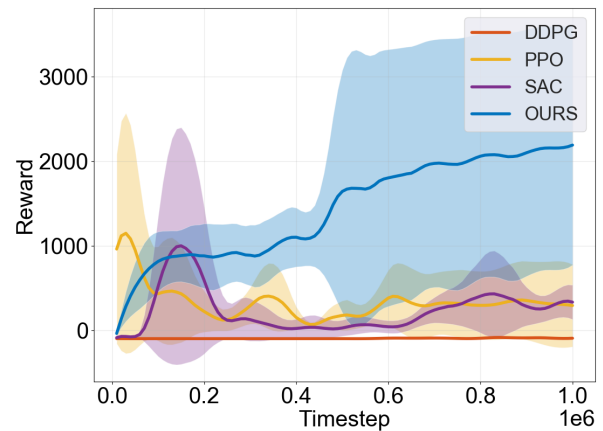

Fig. 7. **Pace Car Learning Graph.** Our result rendered in blue shows far better performance (as indicated by higher reward values) than other algorithms, which make no significant gains in learning even after long periods.

Table 2. Comparison of maximum rewards from each control problem. Our framework is able to find better solutions (indicated by higher reward values) than the other baseline, model-free RL algorithms (DDPG, PPO, or SAC) for every experiment setting (Macro, Micro, or Hybrid).

	Ours	DDPG	PPO	SAC
Macro	**124.38**	113.62	104.53	115.73
Micro	**4482.87**	4.97	3218.19	3373.99
Hybrid	**208.48**	174.18	158.18	191.51

The optimal solution for this problem is clear to the human observer; all vehicles should maintain the desired speed limit. However, the pace car has to determine both acceleration and steering at every frame to remain on the road and lead the vehicles at the same time. To introduce even more complexity, in our experiments, we used a speed limit target that varies over time. Therefore, the neural network

receives other vehicle's states and the speed limit as input, and has to compute the optimal acceleration and steering for next several frames, which is quite challenging. The reward is formulated as follows:

$$R = \sum_{i=1}^{t} \sum_{j=1}^{n} C_{\max} - (v_{\text{targ}} - v_{i,j})^2 \tag{13}$$

where $C_{\max}$ denotes an arbitrary constant value, t and n denote number of frames and number of vehicles respectively. v_{targ} and $v_{i,j}$ represent the target speed limit and j^{th} vehicle's speed at the given frame i.

We also used three baseline RL algorithms for comparison, each of which were run 5 times. The learning graph in Figure 7 and the maximum rewards in Table 2 show that our framework achieved far better overall reward than the other algorithms. PPO and SAC also succeeded in accomplishing initial high rewards but their performance soon plunged. Our framework continuously learned and achieved better and improved results.

7 CONCLUSIONS

We have proposed a novel differentiable hybrid traffic simulator. In our simulator, gradients of traffic states across the time steps are computed analytically and propagated across lanes modeled under complimentary but deeply different regimes. This was made possible with our novel discrete-continuous differentiable conversion module. Further, our analytic formulation is much more efficient than one based on automatic differentiation; therefore, our technique can offer real-time traffic simulation, as shown in Figure 8, that would not have been possible otherwise.

We have also shown that we can use the gradients to solve classic traffic problems. For the parameter estimation problems, our framework is able to find far better estimates than other gradient-free optimization algorithms. For control problems, our framework succeeded in finding near-optimal policies, which was not possible with other model-free RL algorithms.

7.1 Generalization

The concept of hybrid simulation is applicable to many different types of dynamical systems, such as fluids [Golas et al. 2012; Mohamed and Mohamad 2010; Narain et al. 2010], crowd [Narain et al. 2009; Treuille et al. 2006], and many other multi-agent systems [Zheng et al. 2017]. And, our method to differentiate the conversion between macro and micro traffic flow is applicable to a broad range of dynamical systems that are simulated by hybrid techniques, transitioning between continuous and discrete simulation domains. In general, other deterministic or stochastic instantiation process can also take advantage of our technique introduced in this paper. For example, the micro and macro representations for crowd share many similarities with traffic. The conversion between individual pedestrians and the crowd flow is analogous to the process of instantiating and removing vehicles in the roads. We also hypothesize that the algorithmic and computational framework on differentiable hybrid simulation, as proposed here, can also be extended to hybrid control of complex and autonomous systems [Branicky et al. 1998; Fierro et al. 2001].

7.2 Limitations and Future Directions

Given these promising results, we plan to further augment the capability and application of our simulator. First, while hybrid techniques are naturally suited to metropolitan-scale scenarios, we have not yet applied this differentiable hybrid technique to these; we expect to achieve even higher performance improvements from these mega-scale experiments. We can also integrate realistic vehicle motions, such as lane changing, to further capture complex traffic dynamics in higher fidelity. Finally, we used simulated data for solving traffic control problems in this paper, but we expect that our simulator is applicable to real-world data as well.
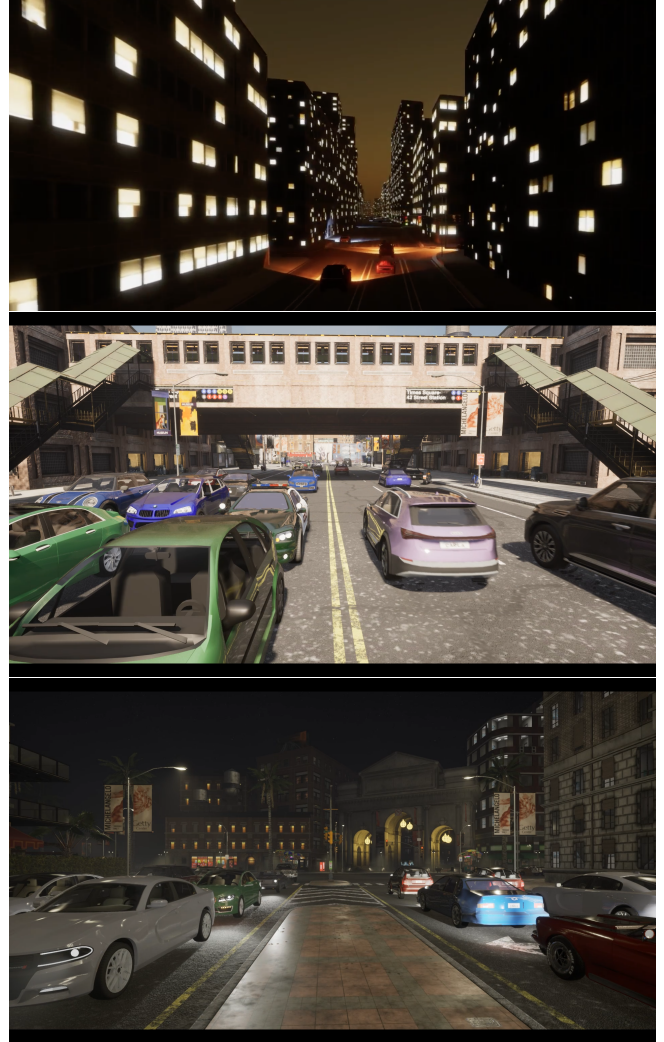

Fig. 8. **Large-scale Hybrid Traffic simulation.** These traffic scenes consist of many hundreds of vehicles across an urban scene simulated and controlled in real time using a differentiable hybrid traffic model.

REFERENCES

Shivam Akhauri, Laura Y Zheng, and Ming C Lin. 2020. Enhanced transfer learning for autonomous driving with systematic accident simulation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5986–5993.

Philipp Andelfinger. 2021. Differentiable Agent-Based Simulation for Gradient-Guided Simulation-Based Optimization. In *Proceedings of the 2021 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*. 27–38.

AATM Aw and Michel Rascle. 2000. Resurrection of" second order" models of traffic flow. *SIAM journal on applied mathematics* 60, 3 (2000), 916–938.

Masako Bando, Katsuya Hasebe, Akihiro Nakayama, Akihiro Shibata, and Yuki Sugiyama. 1995. Dynamical model of traffic congestion and numerical simulation. *Physical review E* 51, 2 (1995), 1035.

Emmanuel Bourrel and Jean-Baptiste Lesort. 2003. Mixing microscopic and macroscopic representations of traffic flow: Hybrid model based on Lighthill–Whitham–Richards theory. *Transportation Research Record* 1852, 1 (2003), 193–200.

Michael S Branicky, Vivek S Borkar, and Sanjoy K Mitter. 1998. A unified framework for hybrid control: Model and optimal control theory. *IEEE transactions on automatic control* 43, 1 (1998), 31–45.

Dan Cascaval, Mira Shalah, Phillip Quinn, Rastislav Bodik, Maneesh Agrawala, and Adriana Schulz. 2021. Differentiable 3D CAD Programs for Bidirectional Editing. *arXiv preprint arXiv:2110.01182* (2021).

Michael J Cassidy and John R Windover. 1995. vÍethodology for Assessing Dynamics of Freeway Traffic Flow. (1995).

Adèle Colas, Wouter van Toll, Katja Zibrek, Ludovic Hoyet, Anne-Hélène Olivier, and Julien Pettré. 2022. Interaction Fields: Intuitive Sketch-based Steering Behaviors for Crowd Simulation. In *Computer Graphics Forum*.

Carlos F Daganzo. 1995. Requiem for second-order fluid approximations of traffic flow. *Transportation Research Part B: Methodological* 29, 4 (1995), 277–286.

Filipe de Avila Belbute-Peres, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J Zico Kolter. 2018. End-to-end differentiable physics for learning and control. *Advances in neural information processing systems* 31 (2018), 7178–7189.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*. 1–16.

Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. 2021. DiffPD: Differentiable Projective Dynamics with Contact. *arXiv:2101.05917* (2021).

Tao Du, Kui Wu, Andrew Spielberg, Wojciech Matusik, Bo Zhu, and Eftychios Sifakis. 2020. Functional Optimization of Fluidic Devices with Differentiable Stokes Flow. *ACM Trans. Graph.* (2020).

Myungeun Eom and Byung-In Kim. 2020. The traffic signal control problem for intersections: a review. *European transport research review* 12, 1 (2020), 1–20.

Rafael Fierro, Aveek K Das, Vijay Kumar, and James P Ostrowski. 2001. Hybrid control of formations of robots. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, Vol. 1. IEEE, 157–162.

Denos C Gazis, Robert Herman, and Renfrey B Potts. 1959. Car-following theory of steady-state traffic flow. *Operations research* 7, 4 (1959), 499–505.

Denos C Gazis, Robert Herman, and Richard W Rothery. 1961. Nonlinear follow-the-leader models of traffic flow. *Operations research* 9, 4 (1961), 545–567.

Moritz Geilinger, David Hahn, Jonas Zehnder, Moritz Bächer, Bernhard Thomaszewski, and Stelian Coros. 2020. ADD: Analytically differentiable dynamics for multi-body systems with frictional contact. *ACM Transactions on Graphics (TOG)* 39, 6 (2020).

Peter G Gipps. 1981. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological* 15, 2 (1981), 105–111.

Abhinav Golas, Rahul Narain, Jason Sewall, Pavel Krajcevski, Pradeep Dubey, and Ming Lin. 2012. Large-scale fluid simulation using velocity-vorticity domain decomposition. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–9.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*. PMLR, 1861–1870.

Torsten Hädrich, Daniel T Banuti, Wojtek Pałubicki, Sören Pirk, and Dominik L Michels. 2021. Fire in paradise: mesoscale simulation of wildfires. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–15.

Nikolaus Hansen. 2006. The CMA evolution strategy: a comparing review. *Towards a new evolutionary computation* (2006), 75–102.

Feixiang He, Yuanhang Xiang, Xi Zhao, and He Wang. 2020. Informative scene decomposition for crowd analysis, comparison and simulation guidance. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 50–1.

Eric Heiden, Miles Macklin, Yashraj S Narang, Dieter Fox, Animesh Garg, and Fabio Ramos. 2021. DiSECt: A Differentiable Simulation Engine for Autonomous Robotic Cutting. In *Proceedings of Robotics: Science and Systems*. Virtual. https://doi.org/10.15607/RSS.2021.XVII.067

Philipp Holl, Vladlen Koltun, Kiwon Um, and Nils Thuerey. 2020. phiflow: A differentiable pde solving framework for deep learning via physical simulations. In *NeurIPS Workshop*.

Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. 2020. DiffTaichi: Differentiable Programming for Physical Simulation. In *ICLR*.

Yuko Ishiwaka, Xiao S Zeng, Michael Lee Eastman, Sho Kakazu, Sarah Gross, Ryosuke Mizutani, and Masaki Nakada. 2021. Foids: bio-inspired fish simulation for generating synthetic datasets. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–15.

Rui Jiang, Qingsong Wu, and Zuojin Zhu. 2001. Full velocity difference model for a car-following theory. *Physical Review E* 64, 1 (2001), 017101.

Arne Kesting, Martin Treiber, and Dirk Helbing. 2007. General lane-changing model MOBIL for car-following models. *Transportation Research Record* 1999, 1 (2007), 86–94.

Dieter Kraft et al. 1988. A software package for sequential quadratic programming. (1988).

Randall J LeVeque et al. 2002. *Finite volume methods for hyperbolic problems*. Vol. 31. Cambridge university press.

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo ray tracing through edge sampling. *ACM Trans. Graph.* 37, 6 (2018).

Yifei Li, Tao Du, Kui Wu, Jie Xu, and Wojciech Matusik. 2022. DiffCloth: Differentiable Cloth Simulation with Dry Frictional Contact. *ACM Trans. Graph.* (2022).

Edward Lieberman and Ajay K Rathi. 1997. Traffic simulation. *Traffic flow theory* (1997).

Michael James Lighthill and Gerald Beresford Whitham. 1955. On kinematic waves II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 229, 1178 (1955), 317–345.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. 2018. Microscopic traffic simulation using sumo. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2575–2582.

Pingchuan Ma, Tao Du, John Z Zhang, Kui Wu, Andrew Spielberg, Robert K Katzschmann, and Wojciech Matusik. 2021. Diffaqua: A differentiable computational design pipeline for soft underwater swimmers with shape interpolation. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–14.

Laurent Magne, Sylvestre Rabut, and Jean-François Gabard. 2000. Towards an hybrid macro-micro traffic flow simulation model. In *INFORMS spring 2000 meeting*.

Salim Mammar, Saïd Mammar, and Jean-Patrick Lebacque. 2006. Highway traffic hybrid macro-micro simulation model. *IFAC Proceedings Volumes* 39, 12 (2006), 627–632.

Khaled M Mohamed and AA Mohamad. 2010. A review of the development of hybrid atomistic–continuum methods for dense fluids. *Microfluidics and Nanofluidics* 8, 3 (2010), 283–302.

Miguel Angel Zamora Mora, Momchil P Peychev, Sehoon Ha, Martin Vechev, and Stelian Coros. 2021. PODS: Policy Optimization via Differentiable Simulation. In *International Conference on Machine Learning*. PMLR, 7805–7817.

Rahul Narain, Abhinav Golas, Sean Curtis, and Ming C Lin. 2009. Aggregate dynamics for dense crowd simulation. In *ACM SIGGRAPH Asia 2009 papers*. 1–8.

Rahul Narain, Abhinav Golas, and Ming C Lin. 2010. Free-flowing granular materials with two-way solid coupling. In *ACM SIGGRAPH Asia 2010 papers*. 1–10.

John A Nelder and Roger Mead. 1965. A simplex method for function minimization. *The computer journal* 7, 4 (1965), 308–313.

Gordon Frank Newell. 1961. Nonlinear effects in the dynamics of car following. *Operations research* 9, 2 (1961), 209–229.

Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. 2019. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)* 38, 6 (2019).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019), 8026–8037.

Harold J Payne. 1971. Model of freeway traffic and control. *Mathematical Model of Public System* (1971), 51–61.

Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C. Lin. 2020. Scalable Differentiable Physics for Learning and Control. In *ICML*.

Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C. Lin. 2021a. Differentiable Simulation of Soft Multi-body Systems. In *Conference on Neural Information Processing Systems (NeurIPS)*.

Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C. Lin. 2021b. Efficient Differentiable Simulation of Articulated Bodies. In *ICML*.

Paul I Richards. 1956. Shock waves on the highway. *Operations research* 4, 1 (1956), 42–51.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

Jason Sewall, David Wilkie, and Ming C Lin. 2011. Interactive hybrid simulation of large-scale traffic. In *Proceedings of the 2011 SIGGRAPH Asia Conference*. 1–12.

Jason Sewall, David Wilkie, Paul Merrell, and Ming C Lin. 2010. Continuum traffic simulation. In *Computer Graphics Forum*, Vol. 29. Wiley Online Library, 439–448.

Jason Douglas Sewall. 2011. *Efficient, scalable traffic and compressible fluid simulations using hyperbolic models*. Ph.D. Dissertation. The University of North Carolina at Chapel Hill.

Siyuan Shen, Yang Yin, Tianjia Shao, He Wang, Chenfanfu Jiang, Lei Lan, and Kun Zhou. 2021. High-order differentiable autoencoder for nonlinear model reduction. *arXiv preprint arXiv:2102.11026* (2021).

Tetsuya Takahashi, Junbang Liang, Yi-Ling Qiao, and Ming C Lin. 2021. Differentiable Fluids with Solid Coupling for Learning and Control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 6138–6146.

Martin Treiber, Ansgar Hennecke, and Dirk Helbing. 2000. Congested traffic states in empirical observations and microscopic simulations. *Physical review E* 62, 2 (2000), 1805.

Adrien Treuille, Seth Cooper, and Zoran Popović. 2006. Continuum crowds. *ACM Transactions on Graphics (TOG)* 25, 3 (2006), 1160–1168.

Hua Wei, Guanjie Zheng, Huaxiu Yao, and Zhenhui Li. 2018. Intellilight: A reinforcement learning approach for intelligent traffic light control. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2496–2505.

Gerald Beresford Whitham. 2011. *Linear and nonlinear waves*. Vol. 42. John Wiley & Sons.

Huan Yu and Miroslav Krstic. 2019. Traffic congestion control for Aw–Rascle–Zhang model. *Automatica* 100 (2019), 38–51.

H Michael Zhang. 2002. A non-equilibrium traffic model devoid of gas-like behavior. *Transportation Research Part B: Methodological* 36, 3 (2002), 275–290.

Yuanshi Zheng, Jingying Ma, and Long Wang. 2017. Consensus of hybrid multi-agent systems. *IEEE transactions on neural networks and learning systems* 29, 4 (2017), 1359–1365.

A SOLUTION TO THE RIEMANN PROBLEM IN ARZ MODEL

A.1 Formulation

For the given left state q_l and right state q_r, we can determine the intermediate state q_0 as follows.

Case 0 : $u_l = u_r$

$$q_0 = q_l \tag{14}$$

Case 1 : $u_l > u_r$

$$q_0 = \begin{cases} q_l & \text{if } \lambda_s \geq 0 \\ q_m & \text{if } \lambda_s < 0 \end{cases} \quad (\lambda_s = \frac{\rho_m u_m - \rho_l u_l}{\rho_m - \rho_l}) \tag{15}$$

Case 2 : $u_r - u_{max}\rho_l^\gamma < u_l < u_r$

$$q_0 = \begin{cases} q_l & \text{if } \lambda_{0l} \geq 0 \\ q_m & \text{if } \lambda_{0m} \leq 0 \\ \tilde{q}(0) & \text{if } \lambda_{0l} < 0, \lambda_{0m} > 0 \end{cases} \tag{16}$$

where

$$\lambda_{0l} = u_l - u_{max}\gamma\rho_l^\gamma$$
$$\lambda_{0m} = u_m - u_{max}\gamma\rho_m^\gamma$$
$$= u_r - u_{max}\gamma\rho_l^\gamma + \gamma(u_r - u_l).$$

Case 3 : $u_l \leq u_r - u_{max}\rho_l^\gamma$

$$q_0 = \begin{cases} q_l & \text{if } \lambda_{0l} \geq 0 \\ \tilde{q}(0) & \text{if } \lambda_{0l} < 0 \end{cases} \tag{17}$$

Case 4 : $\rho_l = 0, \rho_r > 0$

$$q_0 = q_l = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{18}$$

Case 5 : $\rho_l > 0, \rho_r = 0$

Same as Case 3.

From above, q_m is given as

$$\rho_m = (\rho_l^\gamma + \frac{u_l - u_r}{u_{max}})^{\frac{1}{\gamma}}$$
$$u_m = u_r$$
$$y_m = \rho_m(u_m - u_{eq}(\rho_m)),$$

and $\tilde{q}(0)$ is given as

$$\tilde{\rho}(0) = (\frac{u_l + u_{max}\rho_l^\gamma}{(\gamma + 1)u_{max}})^{\frac{1}{\gamma}}$$
$$\tilde{u}(0) = \frac{\rho_l}{\rho_l + 1}(u_l + u_{max}\rho_l^\gamma)$$
$$\tilde{y}(0) = \tilde{\rho}(0)(\tilde{u}(0) - u_{eq}(\tilde{\rho}(0))).$$

A.2 Differentiation

We can see from the above formulation that we have to compute the partial derivatives of q_l, q_m, and $\tilde{q}(0)$ with respect to q_l and q_r, as those values comprise the solutions of the Riemann problem. We can compute the partial derivatives as shown below.

The Jacobian of f is:

$$f'(q) = \begin{bmatrix} u_{eq} + \rho u'_{eq} & 1 \\ yu'_{eq} - \frac{y^2}{\rho^2} & \frac{2y}{\rho} + u_{eq} \end{bmatrix} \tag{19}$$

Partial derivatives of q_l

$$\frac{\partial q_l}{\partial q_l} = I_{2\times2}$$

$$\frac{\partial q_l}{\partial q_r} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Partial derivatives of q_m

$$\frac{\partial q_m}{\partial q_l} = \begin{bmatrix} \frac{\partial \rho_m}{\partial \rho_l} & \frac{\partial \rho_m}{\partial y_l} \\ \frac{\partial y_m}{\partial \rho_l} & \frac{\partial y_m}{\partial y_l} \end{bmatrix}$$

where

$$\frac{\partial \rho_m}{\partial \rho_l} = \frac{1}{\gamma}(\rho_l^\gamma + \frac{u_l - u_r}{u_{max}})^{\frac{1-\gamma}{\gamma}}(\gamma \rho_l^{\gamma-1} + \frac{1}{u_{max}}\frac{\partial u_l}{\partial \rho_l}),$$

$$\frac{\partial \rho_m}{\partial y_l} = \frac{1}{\gamma}(\rho_l^\gamma + \frac{u_l - u_r}{u_{max}})^{\frac{1-\gamma}{\gamma}}(\frac{1}{u_{max}}\frac{\partial u_l}{\partial y_l}),$$

$$\frac{\partial y_m}{\partial \rho_l} = \frac{\partial \rho_m}{\rho_l}(u_m - u_{eq}(\rho_m)) + \rho_m(-u_{eq}'(\rho_m)\frac{\partial \rho_m}{\partial \rho_l}),$$

$$\frac{\partial y_m}{\partial y_l} = \frac{\partial \rho_m}{y_l}(u_m - u_{eq}(\rho_m)) + \rho_m(-u_{eq}'(\rho_m)\frac{\partial \rho_m}{\partial y_l}).$$

$$\frac{\partial q_m}{\partial q_r} = \begin{bmatrix} \frac{\partial \rho_m}{\partial \rho_r} & \frac{\partial \rho_m}{\partial y_r} \\ \frac{\partial y_m}{\partial \rho_r} & \frac{\partial y_m}{\partial y_r} \end{bmatrix}$$

where

$$\frac{\partial \rho_m}{\partial \rho_r} = \frac{1}{\gamma}(\rho_l^\gamma + \frac{u_l - u_r}{u_{max}})^{\frac{1-\gamma}{\gamma}}(-\frac{1}{u_{max}}\frac{\partial u_r}{\partial \rho_r}),$$

$$\frac{\partial \rho_m}{\partial y_r} = \frac{1}{\gamma}(\rho_l^\gamma + \frac{u_l - u_r}{u_{max}})^{\frac{1-\gamma}{\gamma}}(-\frac{1}{u_{max}}\frac{\partial u_r}{\partial y_r}),$$

$$\frac{\partial y_m}{\partial \rho_r} = \frac{\partial \rho_m}{\rho_r}(u_m - u_{eq}(\rho_m)) + \rho_m(\frac{\partial u_r}{\partial \rho_r} - u_{eq}'(\rho_m)\frac{\partial \rho_m}{\partial \rho_r}),$$

$$\frac{\partial y_m}{\partial y_r} = \frac{\partial \rho_m}{y_r}(u_m - u_{eq}(\rho_m)) + \rho_m(\frac{\partial u_r}{\partial y_r} - u_{eq}'(\rho_m)\frac{\partial \rho_m}{\partial y_r}).$$

Partial derivatives of $\tilde{q}(0)$

$$\frac{\partial \tilde{q}(0)}{\partial q_l} = \begin{bmatrix} \frac{\partial \tilde{\rho}(0)}{\partial \rho_l} & \frac{\partial \tilde{\rho}(0)}{\partial y_l} \\ \frac{\partial \tilde{y}(0)}{\partial \rho_l} & \frac{\partial \tilde{y}(0)}{\partial y_l} \end{bmatrix}$$

where

$$\frac{\partial \tilde{\rho}(0)}{\partial \rho_l} = \frac{1}{\gamma}(\frac{u_l + u_{max}\rho_l^\gamma}{(\gamma+1)u_{max}})^{\frac{1-\gamma}{\gamma}}(\frac{1}{(\gamma+1)u_{max}})(\frac{\partial u_l}{\partial \rho_l} + u_{max}\gamma \rho_l^{\gamma-1}),$$

$$\frac{\partial \tilde{\rho}(0)}{\partial y_l} = \frac{1}{\gamma}(\frac{u_l + u_{max}\rho_l^\gamma}{(\gamma+1)u_{max}})^{\frac{1-\gamma}{\gamma}}(\frac{1}{(\gamma+1)u_{max}})(\frac{\partial u_l}{\partial y_l}),$$

$$\frac{\partial \tilde{y}(0)}{\partial \rho_l} = \frac{\partial \tilde{\rho}(0)}{\partial \rho_l}(\tilde{u}(0) - u_{eq}(\tilde{\rho}(0))) + \tilde{\rho}(0)(\frac{\partial \tilde{u}(0)}{\partial \rho_l} - u_{eq}'(\tilde{\rho}(0))\frac{\partial \tilde{\rho}(0)}{\partial \rho_l}),$$

$$\frac{\partial \tilde{y}(0)}{\partial y_l} = \frac{\partial \tilde{\rho}(0)}{\partial y_l}(\tilde{u}(0) - u_{eq}(\tilde{\rho}(0))) + \tilde{\rho}(0)(\frac{\partial \tilde{u}(0)}{\partial y_l} - u_{eq}'(\tilde{\rho}(0))\frac{\partial \tilde{\rho}(0)}{\partial y_l}).$$

$$\frac{\partial \tilde{q}(0)}{\partial q_r} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

B DIFFERENTIATING IDM

B.1 Hyperparameters

In the IDM, following hyperparameters are used to describe different behavioral traits of discrete vehicles [Treiber et al. 2000].

- s_{min}: Minimum desired distance to the leading vehicle.
- T_{pref}: Desired time to move forward with current speed.
- a_{max}: Upper bound of the computed acceleration.
- a_{pref}: Comfortable braking deceleration, positive value.
- v_{targ}: Target velocity it wants to maintain.
- $length_i$: Length of the i-th vehicle.

B.2 Differentiation

First we compute the gradients of $a_i(t)$ with respect to $p_i(t)$, $v_i(t)$, $p_{h(i)}(t)$, and $v_{h(i)}(t)$, where $h(i)$-th vehicle is the leading vehicle of i-th vehicle.

$$\frac{\partial a_i(t)}{\partial p_i(t)} = -2a_{max}\frac{s_{opt}^2}{(p_{h(i)}(t) - p_i(t) - length_{h(i)})^3},$$

$$\frac{\partial a_i(t)}{\partial p_{h(i)}(t)} = 2a_{max}\frac{s_{opt}^2}{(p_{h(i)}(t) - p_i(t) - length_{h(i)})^3},$$

$$\frac{\partial a_i(t)}{\partial v_i(t)} = a_{max}(-\delta\frac{v_i(t)^{\delta-1}}{v_{targ}^\delta} - 2\frac{s_{opt}}{(p_{h(i)}(t) - p_i(t) - length_{h(i)})^2}\frac{\partial s_{opt}}{\partial v_i(t)}),$$

$$\frac{\partial a_i(t)}{\partial v_{h(i)}(t)} = a_{max}(-2\frac{s_{opt}}{(p_{h(i)}(t) - p_i(t) - length_{h(i)})^2}\frac{\partial s_{opt}}{\partial v_{h(i)}(t)}),$$

where

$$\frac{\partial s_{opt}}{\partial v_i(t)} = T_{pref} + \frac{2v_i(t) - v_{h(i)}(t)}{2\sqrt{a_{max}a_{pref}}},$$

$$\frac{\partial s_{opt}}{\partial v_{h(i)}(t)} = \frac{-v_i(t)}{2\sqrt{a_{max}a_{pref}}}.$$

Now for arbitrary i and j, we can compute the gradient of the state of the i-th vehicle with respect to that of the j-th vehicle by plugging in the above values. Note that the gradient is non-zero only when j equals to i or $h(i)$.

$$\frac{\partial p_i(t+1)}{\partial p_j(t)} = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{if } j = h(i) \\ 0 & \text{if } else \end{cases} \tag{20}$$

$$\frac{\partial p_i(t+1)}{\partial v_j(t)} = \begin{cases} \Delta t & \text{if } j = i \\ 0 & \text{if } j = h(i) \\ 0 & \text{if } else \end{cases} \tag{21}$$

$$\frac{\partial v_i(t+1)}{\partial p_j(t)} = \begin{cases} \Delta t(-2a_{max}\frac{s_{opt}^2}{(p_{h(i)}(t)-p_i(t)-length_{h(i)})^3}) & \text{if } j = i \\ \Delta t(2a_{max}\frac{s_{opt}^2}{(p_{h(i)}(t)-p_i(t)-length_{h(i)})^3}) & \text{if } j = h(i) \\ 0 & \text{if } else \end{cases} \tag{22}$$

$$\frac{\partial v_i(t+1)}{\partial v_j(t)} = \begin{cases} 1 + \Delta t a_{max}(-\delta\frac{v_i(t)^{\delta-1}}{v_{targ}^\delta} - 2\frac{s_{opt}}{(p_{h(i)}(t)-p_i(t)-length_{h(i)})^2}(T_{pref} + \frac{2v_i(t)-v_j(t)}{2\sqrt{a_{max}a_{pref}}})) & \text{if } j = i \\ \Delta t a_{max}(-2\frac{s_{opt}}{(p_{h(i)}(t)-p_i(t)-length_{h(i)})^2}\frac{-v_i(t)}{2\sqrt{a_{max}a_{pref}}}) & \text{if } j = h(i) \\ 0 & \text{if } else \end{cases} \tag{23}$$