# Benchmarks for reinforcement learning in mixed-autonomy traffic

**Eugene Vinitsky**[*,†]
Department of ME
University of California Berkeley
evinitsky@berkeley.edu

**Aboudy Kreidieh**[*]
Department of CEE
University of California Berkeley
aboudy@berkeley.edu

**Luc Le Flem**
IEOR
Columbia University

**Nishant Kheterpal**
Department of EECS
University of California Berkeley

**Kathy Jang**
Department of EECS
University of California Berkeley

**Fangyu Wu**
Department of EECS
University of California Berkeley

**Richard Liaw**
Department of EECS
University of California Berkeley

**Eric Liang**
Department of EECS
University of California Berkeley

**Alexandre M. Bayen**
Department of EECS and
Institute for Transportation Studies
University of California Berkeley

**Abstract:** We release new benchmarks in the use of deep reinforcement learning (RL) to create controllers for *mixed-autonomy traffic*, where connected and autonomous vehicles (CAVs) interact with human drivers and infrastructure. Benchmarks, such as Mujoco or the Arcade Learning Environment, have spurred new research by enabling researchers to effectively compare their results so that they can focus on algorithmic improvements and control techniques rather than system design. To promote similar advances in traffic control via RL, we propose four benchmarks, based on three new traffic scenarios, illustrating distinct reinforcement learning problems with applications to mixed-autonomy traffic. We provide an introduction to each control problem, an overview of their MDP structures, and preliminary performance results from commonly used RL algorithms. For the purpose of reproducibility, the benchmarks, reference implementations, and tutorials are available at https://github.com/flow-project/flow.

## 1 Introduction

In recent years, several sequential decision-making benchmarks have enabled the systematic evaluation and comparison of reinforcement learning methods. The *Arcade Learning Environment* (ALE) [1] has become a popular benchmark for evaluating algorithms designed for tasks with high-dimensional state inputs and discrete actions. The benchmarks released with rllab [2] contain 31 continuous control tasks, which range from simple tasks, such as cart-pole balancing, to challenging tasks such as high-DOF locomotion, tasks with partial observations, and hierarchically structured tasks. However, in *mixed-autonomy traffic control*, the study of human drivers interacting with CAVs, the lack of a standardized and challenging testbed for RL that is grounded in a real-world problem domain makes it difficult to quantify the practicality of the proposed methods in the literature. Systematic evaluation and comparison will not only further understanding of the strengths of existing algorithms, but also reveal their limitations and suggest directions for future research.

[*]These authors contributed equally to this work.
[†]Corresponding author

2nd Conference on Robot Learning (CoRL 2018), Zurich, Switzerland.

We attempt to address this problem and present a benchmark consisting of four traffic control tasks based in common, real-world traffic scenarios. Methods which solve these traffic tasks can greatly impact traffic control in cities, the integration of automated vehicles, and urban planning. We characterize a set of RL algorithms by their effectiveness in training deep neural network policies.

In the traffic analysis and control community, the number of standardized benchmarks is limited. For microscopic data on human driving behavior there is the NGSIM dataset [3], the 100-car naturalistic driving study [4], and Mobile Millenium [5]. For traffic control, there are a few recurring problems that are considered, including deriving controllers to minimize propagation of shockwaves in traffic [6, 7], AV assisted merging [8, 9] and energy minimization via platooning [10].

To the best of our knowledge, we are unaware of a standard set of benchmarks for traffic control in a micro-simulator nor a framework in which deep-RL can be applied to the control task. The coming future likely will involve a mixture of AVs interacting with human drivers and infrastructure, but there isn't yet a framework for simultaneously learning control for these interacting pieces. To this end we are releasing four benchmark problems representing four key traffic control problems: shockwave minimization, inflow management, efficient merging, and intersection control. These tasks expose various RL-oriented problems, including exploration, partial observability, scaling in the state space, and scaling in the action space. Additionally, each problem exposes some questions in RL that may primarily arise in the context of transportation systems. By working on a standard set of benchmarks, we hope to make it possible for the mixed-autonomy and RL community to effectively compare their results, test new algorithms, and develop new techniques that are able to take advantage of the underlying structures inherent in the state and action spaces of transportation problems.

The key contributions of this article are:

- The description of a new set of benchmarks for mixed-autonomy traffic that expose several key aspects of traffic control
- A characterization of the RL problems involved in each benchmark as well as an initial description of the performance of a few gradient based and gradient free deep-RL algorithms
- A summary of the key traffic improvements resulting from the application of deep-RL algorithms

The article is organized as follows. In Sec. 2 we provide notation and explanation of deep-RL and mixed-autonomy traffic. In Sec. 3 we provide details on the newly released benchmarks. In Sec. 4 we detail the results of the application of three deep-RL algorithms to learning neural network-based policies and another deep-RL algorithm to learn purely linear policies. Finally, in Sec. 5 we discuss the performance of the algorithms as well as open questions and problems that can be tackled with these benchmarks.

## 2 Background

This section presents a brief overview into the reinforcement learning algorithms and traffic models used for the remainder of this article. In addition, sec. 2.4 introduces the computational framework used to construct the benchmarks described in sec. 3.

### 2.1 Notation and RL

Reinforcement learning problems are generally studied as a discrete-time *Markov decision process* (MDP) [11], defined by $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \rho_0, \gamma, T)$, where $\mathcal{S} \subseteq \mathbb{R}^n$ is an $n$ dimensional state space, $\mathcal{A} \subseteq \mathbb{R}^m$ an $m$ dimensional action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}_+$ a transitional probability function, $r : \mathcal{S} \to \mathbb{R}$ a bounded reward function, $\rho_0 : \mathcal{S} \to \mathbb{R}_+$ an initial state distribution, $\gamma \in (0, 1]$ a discount factor, and $T$ a time horizon.

In an MDP, an *agent* receives sensory inputs $s_t \in \mathcal{S}$ from the the environment and interacts with this environment by performing actions $a_t \in \mathcal{A}$. These actions are defined by a stochastic policy $\pi_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$ parametrized by $\theta$. The objective of the agent is to learn an optimal policy: $\theta^* := \text{argmax}_\theta \eta(\pi_\theta)$, where $\eta(\pi_\theta) = \sum_{i=0}^{T} \gamma^i r_i$ is the expected discounted return across a trajectory $\tau = (s_0, a_0, ..., a_{T-1}, s_T)$, $s_0 \sim \rho_0(s_0)$, $a_t \sim \pi_\theta(a_t|s_t)$, and $s_{t+1} \sim \mathcal{P}(s_{t+1}|s_t, a_t)$, for all $t$.

A broad range of RL algorithms have been designed to compute an optimal policy $\theta^*$. Gradient-based algorithms [12, 13] estimate a gradient for the policy from expected returns achieved during simulation. Gradient-free algorithms [14], on the other hand, treat the return as a black box function to be optimized in terms of the policy parameters. In particular, random search methods [15] training linear policies have recently been demonstrated to be a competitive alternative to gradient-based and

gradient-free method in solving commonly used RL benchmarks. The success of linear policies suggests that these benchmarks are not as difficult as they were believed to be.

### 2.2 Mixed-autonomy traffic

The notion of using connected autonomous vehicles (CAVs) to improve traffic flow has an extensive history stemming back to the 1960s. Early works focused on designing controllers to direct a fleet of vehicles to form a dense, stable platoon [16, 17]. Among many others, these works represent the significant progress that has been made in the paradigm of vehicle platooning.

Recently, there has been a new emphasis on traffic control in the context of mixed-autonomy, where only a fraction of vehicles are CAVs and are interacting with human-driven vehicles. To this end, pioneering researchers proposed and tested several hand-designed control laws [18, 19]. Notably, in 2015, an experimental test of a control strategy named "slow-in, fast-out" strategy [20] was conducted by Taniguchi et al. [21], which involved five vehicles in a closed course. A larger scale experiment was later conducted in 2016, where two other control methods were tested extensively with 21 humans drivers and one CAV in a circular track [22, 23]. It was suggested from the previous works that traffic throughput can be improved with a small percentage of autonomous vehicles.

### 2.3 Simulating mixed-autonomy performance

This article simulates the performance of mixed-autonomy traffic control policies through the use of microscopic traffic simulators, in which the states, actions, and transition properties of a transportation network are modeled at the level of individual vehicles. Traffic microsimulators have been broadly accepted in the transportation engineering community as an acceptable prelude to the empirical evaluation of several complex automated tasks such as cooperative adaptive cruise control (CACC) [24] and mixed-autonomy traffic flow control [6]. This has spurred interest in the development of several frameworks dedicated to accurate microscopic traffic reconstruction, with prominent simulators including propriety software such as Aimsun [25] and VISSIM [26], and open-source libraries such as SUMO [27]. Traffic networks in this article are modeled using the latter.

In the context of traffic microsimulations, human-driver models are used to recreate the lateral [28] and longitudinal [29] behavior of human-driven vehicles. For longitudinal, or acceleration, dynamics, human-driven vehicles in all experiments presented in this article use the *Intelligent Driver Model* [30], a state-of-the-art car-following model. Other behaviors, such as lane changing, merging, right-of-way at intersections, and traffic light compliance, are dictated by the simulator [27].

### 2.4 *Flow*: facilitating benchmark generation

The mixed-autonomy traffic benchmarks presented in this article are designed and developed in *Flow*, a Python library that interfaces the RL libraries RLlib [31] and rllab [2] with SUMO [27], a microscopic simulator for traffic and vehicle dynamics. *Flow* enables the systematic creation of a variety of traffic-oriented RL tasks for the purpose of generating control strategies for autonomous vehicles, traffic lights, etc. These environments are compatible with OpenAI Gym [32] in order to promote integration with the majority of training algorithms currently being developed by the RL community. For details on the architecture and on training autonomous vehicles to maximize system-level velocity, we refer the readers to [33].

## 3 Benchmarks

In this section, we detail the network structure, controllable parameters, provided state spaces, action spaces, and rewards of the benchmarks. The states and rewards associated with each benchmark are normalized in accordance with common RL practices as well as to minimize the amount of necessary hyperparameter tuning from benchmark to benchmark. We stress that most aspects of the benchmark (network structure, reward, observation space, etc.) can be easily modified if researchers are interested in doing so. The variants for each benchmark are detailed in appendix A.

### 3.1 Figure eight: optimizing intersection capacity

The figure eight network (Fig 1a), previously presented in [34], acts as a closed representation of an intersection. In a figure eight network containing a total of 14 vehicles, we witness the formation of queues resulting from vehicles arriving simultaneously at the intersection and slowing down to obey right-of-way rules. This behavior significantly reduces the average speed of vehicles in the network.

In a mixed-autonomy setting, a portion of vehicles are treated as CAVs with the objective of regulating the flow of vehicles through the intersection in order to improve system-level velocities. The components of the MDP for this benchmark are defined as follows:

- States: The state consists of a vector of velocities and positions for each vehicle in the network, ordered by the position of each vehicle, $s := (v_i, x_i)_{i=0:k-1} \in \mathbb{R}^{2k}$, where $k$ is the number of vehicles in the network. Note that the position is defined relative to a pre-specified starting point.
- Actions: The actions are a list of accelerations for each CAV, $a \in \mathbb{R}^n_{[a_{\text{min}}, a_{\text{max}}]}$, where $n$ is the number of CAVs, and $a_{\text{min}}$ and $a_{\text{max}}$ are the minimum and maximum accelerations, respectively.
- Reward: The objective of the learning agent is to achieve high speeds while penalizing collisions. Accordingly, the reward function is defined as follows:

$$r := \max\left(||v_{\text{des}} \cdot \mathbb{1}^k||_2 - ||v_{\text{des}} - v||_2,\ 0\right) / \ ||v_{\text{des}} \cdot \mathbb{1}^k||_2 \tag{1}$$

  where $v_{\text{des}}$ is an arbitrary large velocity used to encourage high speeds and $v \in \mathbb{R}^k$ is the velocities of all vehicles in the network.

### 3.2 Merge: controlling shockwaves from on-ramp merges

The merge network (see Fig. 1c) highlights the effect of disturbances on vehicles in a highway network. Specifically, perturbations resulting from vehicles arriving from the on-merge lead to the formation of backwards propagating stop-and-go waves, thereby reducing the throughput of vehicles in the network. This phenomenon is known as convective instability [35].

In a mixed-autonomy setting, a percentage of vehicles in the main highway are tasked with the objective of dissipating the formation and propagation of stop-and-go waves from locally observable information. Moreover, given the open nature of the network, the total number of CAVs within the network may vary at any given time. Taking these into account, we characterize our MDP as follows:

- States: The state consists of the speeds and bumper-to-bumper gaps of the vehicles immediately preceding and following the CAVs, as well as the speed of the CAVs, i.e. $s := (v_{i,\text{lead}}, v_{i,\text{lag}}, h_{i,\text{lag}}, h_{i,\text{lag}}, v_i) \in \mathbb{R}^{n_{RL}}$. In order to account for variability in the number of CAVs ($n_{CAV}$), a constant $n_{RL}$ term is defined. When $n_{CAV} > n_{RL}$, information from the extra CAVs are not included in the state. Moreover, if $n_{CAV} < n_{RL}$ the state is padded with zeros.
- Actions: The actions consist of a list of bounded accelerations for each CAV, i.e. $a \in \mathbb{R}^{n_{RL}}_{[a_{\text{min}}, a_{\text{max}}]}$ One again, an $n_{RL}$ term is used to handle variable numbers of CAVs. If $n_{CAV} > n_{RL}$ the extra CAVs are treated as human-driven vehicles and their states are updated using human driver models. Moreover, if $n_{CAV} < n_{RL}$, the extra actions are ignored.
- Reward: The objective in this problem is, once again, improving mobility, either via the speed of vehicles in the network or by maximizing the number of vehicles that pass through the network. Accordingly, we use an augmented version of the reward function presented in sec. 3.1.

$$r := \max\left(||v_{\text{des}} \cdot \mathbb{1}^k||_2 - ||v_{\text{des}} - v||_2,\ 0\right) / \ ||v_{\text{des}} \cdot \mathbb{1}^k||_2 - \alpha \sum_{i \in CAV} \max\left[h_{\text{max}} - h_i(t), 0\right] \tag{2}$$

  The added term penalizes small headways among the CAVs; it is minimal when all CAVs are spaced at $h_{\text{max}}$. This discourages dense states that lead to the formation of stop-and-go traffic.

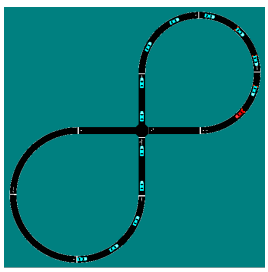
### 3.3 Grid: improving traffic signal timing schedules

The grid (see Fig. 1(b)) is an idealized representation of a city with a grid-like structure such as Manhattan. The purpose of this problem is to highlight issues that arise in coordination of traffic light control, particularly questions of partial observability and the scaling of RL algorithms with action dimension. Solutions to this problem will generate new traffic light control schemes that minimize the average per-vehicle delay while inducing some measure of fairness.

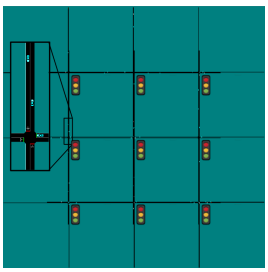
Vehicles enter at the corners of the grid. For simplicity of the problem, vehicles travel straight on their path. Each intersection has a traffic light that allows vehicles to flow either horizontally or vertically. If the light is green, it transitions to yellow for two seconds before switching to red for the purpose of safety.

The gym environment has the following state space, action space, and reward:

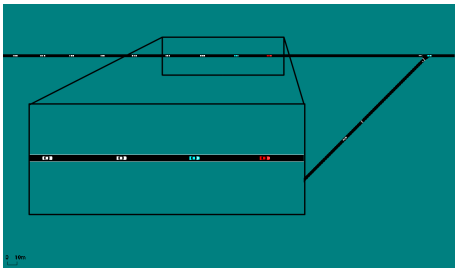
- States: Speed, distance to intersection, and edge number of each vehicle. The edges of the grid are uniquely numbered so the travel direction can be inferred. For the traffic lights we return 0,1



(a) Figure Eight



(b) Grid



(c) Merge

(d) Bottleneck

Figure 1: Network configurations for the various benchmarks presented in this article. In each of the figures, vehicles in red are autonomous (controlled by the RL agent), in blue are human-driven and directly observed in the state space, and in white are human-driven an unobserved.

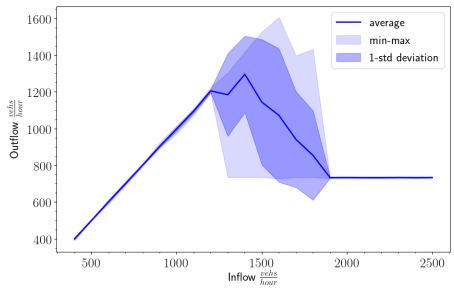



Figure 2: Inflow-outflow relation of the bottleneck. At an inflow of approximately 1500 veh/hour, the outflow begins to dip. Past the drop the minimum represents the stable value the system would decay to if run for sufficiently long.

corresponding to green or red for each light, a number between [0, $t_{\text{switch}}$] indicating how long until a switch can occur, and 0,1 indicating if the light is currently yellow. Finally, we return the average density and velocity of each edge.

- Actions: A list of numbers $a = [-1, 1]^n$ where $n$ is the number of traffic lights. If $a_i > 0$ for traffic light $i$ it switches, otherwise no action is taken.
- Reward: The reward present in Eq. 3.1 is once again used for this benchmark. Here the use of the 2-norm induces fairness, as a more equal distribution of speeds produces a larger reward.

### 3.4 Bottleneck: maximizing throughput in a bottleneck structure

The bottleneck is an idealized version of the Oakland-San Francisco Bay Bridge. On the Bay Bridge, 16 non-HOV lanes narrow down to eight and subsequently to five. In our model, the lanes reduce from $4N$ to $2N$ to $N$, where $N$ is a scaling factor. This system exhibits the phenomenon known as *capacity drop* [36], where the throughput, defined as number of vehicles passing through the system per hour, experiences a sharp drop in magnitude after the inflow increases past a critical value. This phenomenon has been empirically observed and contributes to the inefficiency of highway traffic (see fig. 2).

Given this inflow-outflow relation, the goal of this problem is to learn to avoid the capacity drop and maximize the total outflow in a mixed-autonomy setting. In what follows, we term each distinct segment of road an *edge*. For each edge, users can specify for each edge whether it is observed and/or controlled, and how many segments it is divided into: we refer to this as an *edge-segment*. Although

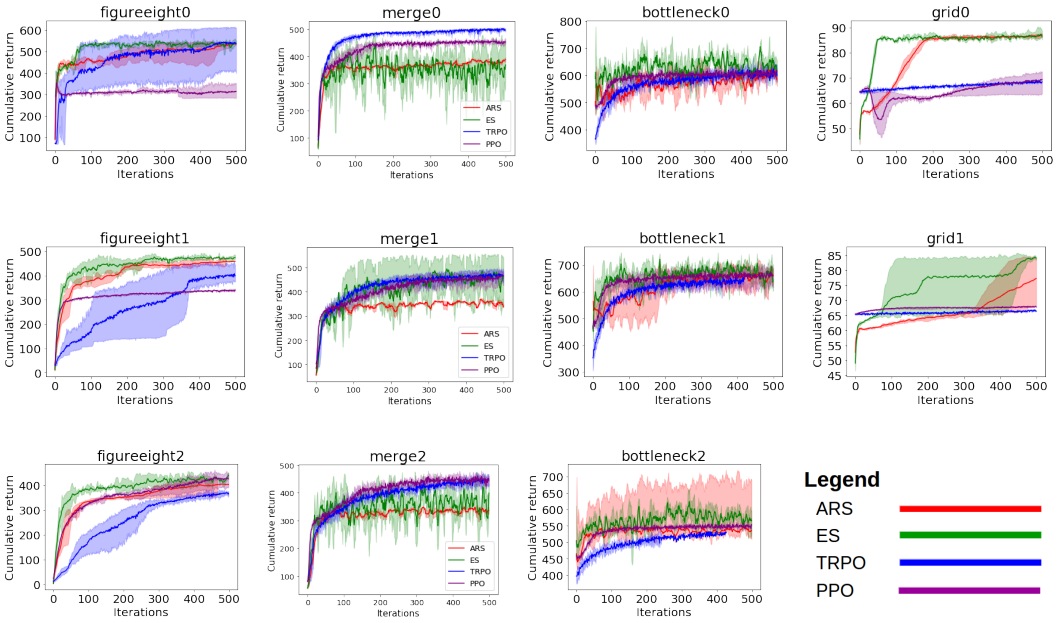



Figure 3: Learning curves for each benchmark. A iteration of training in each algorithm consists of 50 rollouts/trajectories. The reported cumulative rewards correspond to undiscounted returns.

the observation space, action space, and reward can easily be modified, the provided environment operates on the following MDP:

- States: The mean positions and velocities of human drivers for each lane for each edge segment. The mean positions and velocities of the CAVs on each segment. The outflow of the system in vehicles per/hour over the last 5 seconds.
- Actions: For a given edge-segment and a given lane, the RL action shifts the maximum speed of all the CAVs in the segment from their current value. By shifting the max-speed to higher or lower values, the system indirectly controls the velocity of the RL vehicles.
- Reward: $r_t = \sum_{i=t-\frac{5}{\Delta t}}^{i=t} \frac{n_{\text{exit}}(i)}{\Delta t * n_{\text{lanes}} * 500}$ where $n_{\text{exit}}(i)$ is the number of vehicles that exited the system at time-step i. Colloquially, this is the outflow over the last 5 seconds normalized by the number of lanes, $n_{\text{lanes}}$ and a factor of 500. This is to keep the scale of the reward in line with the other benchmarks.

# 4 Experiments

In this section, we present preliminary results from running four reinforcement learning algorithms on the benchmarks presented in sec. 3.

## 4.1 Candidate controllers

Experiments were conducted using gradient-based algorithms *Trust Region Policy Optimization* (TRPO) [12] and *Proximal Policy Optimization* (PPO) [13] as well as the gradient-free method *Evolutionary Strategies* (ES) [14] and an implementation of *Augmented Random Search* (ARS) [15]. For ARS a linear policy is used. For ES we use a deterministic MLP with hidden layers $(100, 50, 25)$ and $\tanh$ non-linearity whereas for PPO and TRPO the MLP is diagonal Gaussian. Moreover, in the PPO algorithm, a [256, 256] MLP with tanh non-linearity is used to compute a value function baseline, whereas a linear feature baseline is used for the TRPO algorithm. Other explored hyperparameters for each algorithm can be found in Table 2 in appendix section B. Results are reported over three random seeds to account for stochasticity and to test training stability. Outside of the reported hyperparamters, we use the default parameters set in rllab and rllib as per the commits specified in section 4.2.

Table 1: Average optimal return and standard deviation based on performance metrics after 500 training iterations; average is over 40 rollouts. "–" indicates that the experiment did not have a complete number of seeds and thus was not reported.

| Benchmark | ARS | ES | TRPO | PPO | Human |
|---|---|---|---|---|---|
| Figure Eight 0 | $7.31 \pm 0.54$ | $6.87 \pm 0.08$ | $8.26 \pm 0.10$ | – | $4.18 \pm 0.09$ |
| Figure Eight 1 | $6.43 \pm 0.01$ | – | $5.61 \pm 0.55$ | – | $4.18 \pm 0.09$ |
| Figure Eight 2 | $5.70 \pm 0.01$ | $5.96 \pm 0.00$ | $5.03 \pm 0.23$ | – | $4.18 \pm 0.09$ |
| Merge 0 | $11.3 \pm 0.31$ | $13.31 \pm 0.54$ | $14.95 \pm 0.12$ | $13.66 \pm 0.40$ | $7.39 \pm 0.55$ |
| Merge 1 | $11.06 \pm 0.32$ | $17.29 \pm 0.40$ | $13.74 \pm 0.23$ | $14.61 \pm 0.47$ | $7.39 \pm 0.55$ |
| Merge 2 | $11.5 \pm 0.54$ | $17.36 \pm 0.48$ | $14.14 \pm 0.24$ | $14.54 \pm 0.31$ | $7.39 \pm 0.55$ |
| Grid 0 | $270.2 \pm 0.2$ | $271.7 \pm 0.6$ | $296.2 \pm 2.5$ | $296.8 \pm 5.3$ | $280.8 \pm 1.5$ |
| Grid 1 | $274.7 \pm 1.0$ | $274.3 \pm 1.1$ | $296.0 \pm 2.0$ | $296.2 \pm 2.0$ | $276.8 \pm 1.7$ |
| Bottleneck 0 | $1265 \pm 263$ | $1360 \pm 200$ | $1298 \pm 268$ | $1167 \pm 264$ | $1023 \pm 263$ |
| Bottleneck 1 | $1350 \pm 162$ | $1378 \pm 192$ | $1375 \pm 61$ | $1258 \pm 200$ | $1135 \pm 319$ |
| Bottleneck 2 | $2284 \pm 231$ | $2324 \pm 264$ | $2131 \pm 190$ | $2143 \pm 208$ | $1889 \pm 252$ |

### 4.2 Reproducibility

In the spirit of reproducibility, the controllers used to generate the following results are stored as .pkl files and tensorflow checkpoints at `s3://public.flow.results/corl_exps/exps_final`. We have frozen the code used to generate the results as Flow 0.3.0 and commit number "bc44b21". The commit number of SUMO, available at `https://github.com/eclipse/sumo` used to run the results is "1d4338ab80". The version of RLlib used to run the code is available at `https://github.com/eugenevinitsky/ray` at commit "0f45f80".

### 4.3 Algorithm performance

Fig. 3 presents the training performance of each of the algorithms presented in sec. 4.1. The results suggest that gradient-based algorithms (TRPO and PPO) are more effective in merge scenarios, while gradient-free algorithms (ES) are better in figure eight, grid, and bottleneck benchmarks. This is likely due to the merge network reward providing richer feedback with regards to which states are good/bad, thereby assisting gradient-based algorithms in computing meaningful gradients. On the other hand, in environments populated with many local extrema associated with poor rewards, more exploration-oriented algorithms (ARS or ES) are less prone to being trapped in local optima.

### 4.4 Controller performance

We highlight the effectiveness of each learned policy on improving traffic performance. In order to do so in terms of understandable values that are not cluttered by the reward design process, simple scenario-specific performance metrics are defined for each benchmark. These metrics are:

- Figure eight: Average speed of vehicles in the network (m/s).
- Merge: Average speed of vehicles in the network (m/s).
- Grid: Average delay of all vehicles in the network (s) relative to traveling at the speed limit. In this case, smaller values are preferable.
- Bottleneck: Outflow over the last 500 seconds of a 1000 second rollout (veh/hr).

In order to get a sense of how well these policies are performing, we use an estimate of human-level performance acquired by running simulations of each environment in the absence of learning agents as a baseline. Note, for the grid we use the actuated traffic light baseline available in SUMO with the default parameters for minimum duration of green and red time; however, we set the yellow light to imitate our controller and switch every two seconds. Table 1 highlights the performance of each algorithm within the context of the above mentioned performance metrics. As we can see, the learned policy outperform human-level dynamics in all task.

## 5 Discussion

Given the current performance of the different RL algorithms, there is an abundance of open questions that remain for these benchmarks. For each of these problems, it remains to characterize whether the optimal solution has been achieved. Towards this end, we provide some intuition here that the optimal solution has not been achieved from the perspective of speed/delay of the resultant traffic. For the figure eight, although it is a toy example, CAVs in a fully autonomous setting do not succeed in coordinating themselves in a manner such that they are evenly spaced and move

through the intersection at the speed limit (30 m/s) without collision. The likely existence of a more optimal designable control strategy suggests that there is an exploration problem to be solved. For the bottleneck, the capacity diagram in Fig. 2 occasionally reaches a maximum of 1550 vehicles per hour, suggesting that it is possible, at least over the observed horizon, to arrange vehicles so that they merge optimally without the sharp decelerations that eventually give rise to the bottleneck. Furthermore, for *bottleneck2* the problem is two copies of bottleneck 0 stacked together; the optimal reward is at least the value of *bottleneck0*, but that value is not achieved. For the merge, the trained policies started to gradually degrade after certain iterations, suggesting that the problem is difficult to solve with existing optimization methods. Finally, for the grid, the considered inflows are 300 vehicles per hour per edge for both small and large grids. While we do not have a proof that we have not discovered an optimal solution for the grid problems, visualizations of the solutions do not show the platooning that is expected of optimal solutions to heavy traffic inflows. This is a heuristic argument and further characterization of the optimum is worth pursuing.

While only a partial list, we highlight several additional key challenges that emerge in studying these problems. The first, apparent in the merge and the bottleneck, are the unfixed action and state space. At any given time, the number of controlled and uncontrolled vehicles vary. In the merge case, we resorted to throwing away extraneous actions when there were too few vehicles and having the excess vehicles imitate a human driven vehicle when there were too many. For the bottleneck, we chose to discretize, passing the same command to every vehicle in a given segment of the bottleneck. Similarly for the state space, we turned to aggregate statistics, simply recording the number of vehicles in a given segment rather than more fine-grained statistics. While both of these solutions appear to partially work, perhaps there is a more elegant solution that can adaptively scale to different sized-state and action spaces. Possible candidates include treating this as a multi-agent problem and just using local information for each agent, taking a giant space space that is mostly sparse when vehicles are missing, or using a neural network with sets of input and output matrices that can be selected from to accommodate the variations in size.

Several of these benchmarks admit problems similar to those experienced in video game environments, namely sparsity/credit assignment of the reward and redundancies and symmetries in the state space. The rewards are averages across the performance of the vehicles; because of the number of vehicles in the system for the larger problems like *bottleneck* and *grid*, any given actions positive or negative effect can be offset by the opposite effect elsewhere in the system. An equally interesting issue is the notion of symmetry; many transportation systems, exemplified here by the *bottleneck* and the *grid*, contain a high degree of symmetry. The symmetry affords a potential for a reduction in the dimension of the state space: exploring how to do this, whether via convolutional neural networks, graph neural networks, or explicit orderings of the state space that remove the symmetries, is an open question. Furthermore, many of the systems are clearly composed of distinct sub-units, suggesting that hierarchical approaches could be viable.

A few more open questions that are not explored in this work include: fairness, decentralization, generalization, and sample-efficiency. From the perspective of fairness, several of our results achieve high scores by inducing negative outcomes on a subset of the vehicles. For example, ARS on the bottleneck simply learn to block a lane. This reduces merge conflicts and increases outflow but has extremely negative effects on some of the vehicles. Our controllers are fully centralized; this is unlikely to be possible in real road networks and the effects of decentralization remain unexplored. As to generalization, an interesting question is whether any of the derived controllers will work well on networks or inflows that were outside of their training set. Finally, many of our experiments take over 10 hours to perform 500 iterations; to scale to real-world transportation networks it will be necessary to identify methods for increasing sample efficiency.

Finally, as evidenced by the large *grid* and *bottleneck* benchmarks on which we either do not learn an optimal policy or see flat training curves for several algorithms, there remain fundamental advances to be made in learning policies that can tackle the large action and state spaces that arise in trying to control city-scale road networks. It is our hope that the release of these benchmarks spur excitement and progress on these problems.

## References


[1] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.(JAIR)*, 47:253–279, 2013.

[2] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. *CoRR*, abs/1604.06778, 2016. URL http://arxiv.org/abs/1604.06778.

[3] US Department of Transportation. Ngsim - next generation simulation, 2008. URL https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm.

[4] T. A. Dingus, S. G. Klauer, V. L. Neale, A. Petersen, S. E. Lee, J. Sudweeks, M. Perez, J. Hankey, D. Ramsey, S. Gupta, et al. The 100-car naturalistic driving study, phase ii-results of the 100-car field experiment. Technical report, 2006.

[5] J. C. Herrera, D. Work, R. Herring, X. Ban, Q. Jacobson, and A. Bayen. Evaluation of traffic data obtained via GPS-enabled mobile phones: The mobile century field experiment. *Transportation Research Part C: Emerging Technologies*, 18(4):568–583, 2010.

[6] R. E. Stern, S. Cui, M. L. D. Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, R. Haulcy, H. Pohlmann, F. Wu, B. Piccoli, B. Seibold, J. Sprinkle, and D. B. Work. Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments. *CoRR*, abs/1705.01693, 2017. URL http://arxiv.org/abs/1705.01693.

[7] I. G. Jin and G. Orosz. Dynamics of connected vehicle systems with delayed acceleration feedback. *Transportation Research Part C: Emerging Technologies*, 46:46–64, 2014.

[8] C. Englund, L. Chen, and A. Voronov. Cooperative speed harmonization for efficient road utilization. In *Communication Technologies for Vehicles (Nets4Cars-Fall), 2014 7th International Workshop on*, pages 19–23. IEEE, 2014.

[9] R. Pueboobpaphan, F. Liu, and B. van Arem. The impacts of a communication based merging assistant on traffic flows of manual and equipped vehicles at an on-ramp using traffic flow simulation. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 1468–1473. IEEE, 2010.

[10] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa. Overview of platooning systems. In *Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012)*, 2012.

[11] R. Bellman. A markovian decision process. Technical report, DTIC Document, 1957.

[12] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In *ICML*, pages 1889–1897, 2015.

[13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[14] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[15] H. Mania, A. Guy, and B. Recht. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

[16] S. E. Shladover. Review of the state of development of advanced vehicle control systems (avcs). *Vehicle System Dynamics*, 24(6-7):551–595, 1995.

[17] B. Besselink and K. H. Johansson. String stability and a delay-based spacing policy for vehicle platoons subject to disturbances. *IEEE Transactions on Automatic Control*, 2017.

[18] F. Knorr, D. Baselt, M. Schreckenberg, and M. Mauve. Reducing traffic jams via VANETs. *IEEE Transactions on Vehicular Technology*, 61(8):3490–3498, 2012.

[19] M. Wang, W. Daamen, S. P. Hoogendoorn, and B. van Arem. Connected variable speed limits control and car-following control with vehicle-infrastructure communication to resolve stop-and-go waves. *Journal of Intelligent Transportation Systems*, 20(6):559–572, 2016.

[20] R. Nishi, A. Tomoeda, K. Shimura, and K. Nishinari. Theory of jam-absorption driving. *Transportation Research Part B: Methodological*, 50:116–129, 2013.

[21] Y. Taniguchi, R. Nishi, A. Tomoeda, K. Shimura, T. Ezaki, and K. Nishinari. *A Demonstration Experiment of a Theory of Jam-Absorption Driving*, pages 479–483. Springer International Publishing, 2015.

[22] R. E. Stern, S. Cui, M. L. Delle Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, R. Haulcy, H. Pohlmann, F. Wu, B. Piccoli, B. Seibold, J. Sprinkle, and D. B. Work. Data and source code. https://doi.org/10.15695/vudata.cee.1, 2017.

[23] F. Wu, R. Stern, S. Cui, M. L. D. Monache, R. Bhadanid, M. Bunting, M. Churchill, N. Hamilton, R. Haulcy, B. Piccoli, B. Seibold, J. Sprinkle, and D. Work. Tracking vehicle trajectories and fuel rates in oscillatory traffic. *Transportation Research Part C: Emerging Technologies*, 2017.

[24] V. Milans, S. E. Shladover, J. Spring, C. Nowakowski, H. Kawazoe, and M. Nakamura. Cooperative adaptive cruise control in real traffic situations. *IEEE Transactions on Intelligent Transportation Systems*, 15(1):296–305, Feb 2014. ISSN 1524-9050. doi:10.1109/TITS.2013.2278494.

[25] J. Casas, J. L. Ferrer, D. Garcia, J. Perarnau, and A. Torday. Traffic simulation with aimsun. In *Fundamentals of traffic simulation*, pages 173–232. Springer, 2010.

[26] M. Fellendorf and P. Vortisch. Microscopic traffic flow simulator vissim. In *Fundamentals of traffic simulation*, pages 63–93. Springer, 2010.

[27] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker. Recent development and applications of sumo-simulation of urban mobility. *International Journal On Advances in Systems and Measurements*, 5(3&4), 2012.

[28] M. Rahman, M. Chowdhury, Y. Xie, and Y. He. Review of microscopic lane-changing models and future research opportunities. *IEEE transactions on intelligent transportation systems*, 14(4):1942–1956, 2013.

[29] M. Brackstone and M. McDonald. Car-following: a historical review. *Transportation Research Part F: Traffic Psychology and Behaviour*, 2(4):181–196, 1999.

[30] M. Treiber and A. Kesting. The intelligent driver model with stochasticity-new insights into traffic flow oscillations. *Transportation Research Procedia*, 23:174–187, 2017.

[31] E. Liang, R. Liaw, R. Nishihara, P. Moritz, R. Fox, J. Gonzalez, K. Goldberg, and I. Stoica. Ray rllib: A composable and scalable reinforcement learning library. *arXiv preprint arXiv:1712.09381*, 2017.

[32] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[33] C. Wu, K. Parvate, N. Kheterpal, L. Dickstein, A. Mehta, E. Vinitsky, and A. M. Bayen. Framework for control and deep reinforcement learning in traffic. In *Intelligent Transportation Systems (ITSC), 2017 IEEE 20th International Conference on*, pages 1–8. IEEE, 2017.

[34] C. Wu, A. Kreidieh, E. Vinitsky, and A. M. Bayen. Emergent behaviors in mixed-autonomy traffic. In S. Levine, V. Vanhoucke, and K. Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 398–407. PMLR, 13–15 Nov 2017. URL http://proceedings.mlr.press/v78/wu17a.html.

[35] M. Treiber and A. Kesting. Evidence of convective instability in congested traffic flow: A systematic empirical and theoretical investigation. *Transportation Research Part B: Methodological*, 45(9):1362–1377, 2011.

[36] M. Saberi and H. S. Mahmassani. Empirical characterization and interpretation of hysteresis and capacity drop phenomena in freeway networks. *Transportation Research Record: Journal of the Transportation Research Board, Transportation Research Board of the National Academies, Washington, DC*, 2013.

# Appendix

## A Meta-parameter configurations

We explore varying levels of difficulty for each of the proposed benchmarks by modifying their scenario-specific meta-parameters. The below subsections detail the chosen meta-parameter for each of the tasks.

### A.1 Figure eight

- `figureeight0`: 13 humans, 1 CAV ($\mathcal{S} \in \mathbb{R}^{28}$, $\mathcal{A} \in \mathbb{R}^{1}$, $T = 1500$).
- `figureeight1`: 7 humans, 7 CAVs ($\mathcal{S} \in \mathbb{R}^{28}$, $\mathcal{A} \in \mathbb{R}^{7}$, $T = 1500$).
- `figureeight2`: 0 human, 14 CAVs ($\mathcal{S} \in \mathbb{R}^{28}$, $\mathcal{A} \in \mathbb{R}^{14}$, $T = 1500$).

### A.2 Merge

- `merge0`: 10% CAV penetration rate ($S \in \mathbb{R}^{25}$, $A \in \mathbb{R}^{5}$, $T = 750$).
- `merge1`: 25% CAV penetration rate ($S \in \mathbb{R}^{65}$, $A \in \mathbb{R}^{13}$, $T = 750$).
- `merge2`: 33.3% CAV penetration rate ($S \in \mathbb{R}^{85}$, $A \in \mathbb{R}^{17}$, $T = 750$).

### A.3 Grid

- `grid0`: 3x3 grid (9 traffic lights), inflow = 300 veh/hour/lane ($\mathbb{S} \in \mathbb{R}^{339}$, $\mathbb{A} \in \mathbb{R}^{9}$, $T = 400$)
- `grid1`: 5x5 grid (25 traffic lights), inflow = 300 veh/hour/lane ($\mathbb{S} \in \mathbb{R}^{915}$, $\mathbb{A} \in \mathbb{R}^{25}$, $T = 400$)

### A.4 Bottleneck

- `bottleneck0`: $N = 1$, inflow = 1900 veh/hour, 10% CAV penetration. No vehicles are allowed to lane change. ($\mathcal{S} \in \mathbb{R}^{141}$, $\mathcal{A} \in \mathbb{R}^{20}$, $T = 1000$)
- `bottleneck1`: $N = 1$, inflow = 1900 veh/hour, 10% CAV penetration. The human drivers follow the standard lane changing model in the simulator. ($\mathcal{S} \in \mathbb{R}^{141}$, $\mathcal{A} \in \mathbb{R}^{20}$, $T = 1000$)
- `bottleneck2`: $N = 3$, inflow = 3800 veh/hour, 10% CAV penetration. No vehicles are allowed to lane change. ($\mathcal{S} \in \mathbb{R}^{281}$, $\mathcal{A} \in \mathbb{R}^{40}$, $T = 1000$)

## B Hyperparameter search

Hyperparameter searches were conducted on the various algorithms to yield optimally performing training results. For ARS and ES, a grid search was conducted over SGD step sizes of [0.01, 0.02] and noise standard deviations of [0.01, 0.02]. The best hyperparameters are reported in Table 2.

Table 2: Description of controller-specific hyperparameters.

| Controller | Selected Hyperparameters |
|---|---|
| ARS | SGD step size=0.02, $\sigma = 0.02$ |
| ES | Adam step size=0.02, $\sigma = 0.02$ |
| TRPO | Adam step size=0.01 |
| PPO | Num epochs=10 |
| | `merge`, `bottleneck`: $\lambda$(GAE)$= 0.97$, Adam step size=$5 \times 10^{-4}$ |
| | `grid0`: $\lambda$(GAE)$= 0.5$, Adam step size=$5 \times 10^{-5}$ |
| | `grid1`: $\lambda$(GAE)$= 0.3$, Adam step size=$5 \times 10^{-4}$ |

a) ARS and ES: $\sigma$ is the standard deviation of the noise used to perturb parameters; b) TRPO and PPO: use of actor-critic method, the former with advantages as empirical returns minus the baseline and batch gradient descent, the latter with GAE advantages and stochastic gradient descent