Traffic Flow Control by Autonomous Vehicle Fleets
Group A1

Thomas Bakker (s2468360)
Ewout Bergsma (s3441423)
Daniel Bick (s3145697)
Thijs Eker (s2576597)

3 November 2019

Abstract

As autonomously driven vehicles are gaining popularity in real life traffic, research has come to investigate the potential long-term effects of the introduction of autonomous vehicles (AVs) to real life traffic conditions. Often, this research employs traffic simulations where a single controller is designed or trained on a policy for commanding actions to be performed to all controlled AVs in a simulation at once. This paper presents a more advanced, distributed multi-agent control architecture where a single reinforcement learning (RL) agent is jointly trained by a fleet of AVs on controlling individual AVs based on their private percepts. This control architecture is used to investigate how a fleet of AVs could achieve the goal of collaborating amongst each other in order to decrease the fleet's combined travelling time. Also, it is considered in which way the learned policy affects the overall traffic situation. The traffic simulation software Flow is used in this research. Results show that RL can be used to control AVs in a multi-agent setting. The controlled vehicles do not exhibit behaviors, however, which would lead to achieving the desired goal. Discussion of potential reasons for this is provided.

1 Introduction

The emergence of autonomous vehicles (AVs) in real life traffic environments leads to *mixed-autonomy traffic*, in which human drivers share roads with autonomously driven cars. A lot of research has investigated the potential impact of AVs on general traffic conditions already. One paper investigated how the launch of AV taxi fleets would concretely change the traffic flow in and around the city of Berlin [MB16]. The authors conclude that the launch of AVs would relax the road conditions momentarily coined by traffic congestion. This could be achieved because AVs owned by a taxi fleet could perform many jobs for many independent individuals in a row, while privately owned cars only carry a limited number of passengers each day, being parked for the rest of the day.

A mixed-autonomy setting naturally implies that both types of drivers, humans and AVs, have to cooperate to a certain extent. While AVs will have to learn to behave both safely and goal-driven in an existing environment, namely the current road conditions, also human drivers will have to adapt to new control architectures driving surrounding cars, whose driving behavior does not necessarily follow a human driving behavior.

1.1 Problem

The authors of [MB16] conclude that, in spite of relaxing the overall traffic situation, even the launch of AV taxi services could not handle a huge further increase in the number of cars present in a given city. Whilst this research seems relevant to the situation of the general public, since it is concerned with general predictions of future traffic situations, it does not shine a light on the possible advantages for companies maintaining the taxi fleets examined in these future scenarios. In one such a scenario, AVs will be cooperative within their own fleet of AVs in order to reach the collective goal of decreasing their combined travelling times. Such a goal may be dictated by a taxi company aiming for an increased number of passengers by keeping the travelling time per passenger as low as possible. In this scenario, the AV fleet may try achieving its goal possibly at the expense of decreasing an infrastructure's overall efficiency, where efficiency may refer to an infrastructure's throughput, which may be defined by measuring the overall number of cars travelling through a given road segment in a given amount of time. Considering this scenario, our research question will therefore be two-fold:

1. What percentage of cooperating AVs is needed to be owned by a single organisation to significantly drop their combined travelling time?
2. What is the effect on the average travelling time for the non-cooperating cars, i.e. all cars not belonging to the cooperating AV fleet?

As the questions suggest, in this research, efficiency of cars will be measured in terms of time steps needed to get from a starting location to a goal location.

1.2 State of the art

The authors of [Vin+18] have proposed several reinforcement learning (RL) benchmarks to experiment with. These benchmarks include a variety of simulated traffic situations, such as merging onto a highway. The aforementioned study explores how AVs could be able to learn to adjust and adapt their own policies to human-driver policies through RL, whereas earlier studies focused on handcrafted control laws (e.g. [Kno+12]).

Environments that exclusively contain AVs have also been studied with RL implementations. In a study performed by [Low+17], RL has been used for policy learning in mixed multi-agent cooperative-competitive environments.

1.3 New idea

The authors of [Vin+18] have proposed a method where RL is used to train a centralized controller to control the AV cars by specifying maximum speeds for different parts of the road. Furthermore, the paper by Vinitsky et al. introduced several benchmarks. In our paper, we build on one of the provided benchmarks, the bottleneck environment. This environment consists of multiple lanes merging into a smaller amount of lanes, thereby creating bottlenecks on the road where cars have to switch lanes to be able to continue their driving. Figure 1 illustrates the environment. We altered the benchmark to study how the combination of human drivers with a fleet of cooperative AVs (i.e. cooperative with other AVs) affects the traffic flow in the bottleneck environment.

Figure 1: Bottleneck environment where 4 lanes merge into 2 lanes and finally into 1 lane. Cars have to switch lanes to continue following the road.

2 Method

2.1 Traffic simulation

For this research an existing traffic environment is modified to fit the requirements to answer the research questions raised above. These modifications regard the design of both a customized state representation and reward function.

2.1.1 State representation

The used state representation should only contain relevant data, while maintaining the property to remain realistic to a certain extent. Concretely, a car's state representation contains the data of both its own state and the state of its surrounding cars.
An AV observes its own speed and location (x-coordinate and lane number). Furthermore, from surrounding cars, an AV observes the cars' labels, the distances to the observed cars, and their respective speeds. An agents can only observe vehicles that are in the same or an adjacent lane and for each lane it can only observe one car in front and one behind itself. This means the agent will observe maximally 6 other cars at the same time. Figure 2 shows a possible situation in our simulation, the cars outlined in yellow are the cars that will be observed by the green AV.
The surrounding car's labels are used to represent whether the observed car belongs to its own group of cooperating fellow AV agents or whether it is a human car, to which the AV should not be more cooperative than necessary to maintain its group's egoistic goal of a maximized AV-outflow-rate.

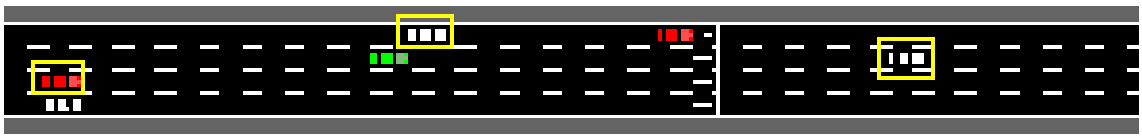

Figure 2: The yellow outlined vehicles are observed by the green AV. Red cars are AVs and white cars resemble human drivers.

2.1.2 Reward function

During our research we were constantly challenged to rethink and redesign our reward function. The cause of this was that finding a reward function that encourages cooperation between agents is not trivial. When not designed carefully, the RL agents will quickly find ways to exploit the reward function and not learn the desired behaviour. In this section two reward

functions will be explained in more detail.
The first reward function, $\mathbf{r_{velocity}}$, rewards an agent based on the proximity of the velocity of all nearby AVs (defined as AVs in the same edge), including itself, to a desired velocity. If we take two vectors, $\vec{a}$ contains the measured velocities in the edge and $\vec{b}$ contains the desired velocities (a vector of all 40 in our experiment). Then the reward $\mathbf{r_{velocity}}$ can be written as in Equation 1.

$$\mathbf{r_{velocity}} = \frac{\|\vec{b}\| - \|\vec{a} - \vec{b}\|}{\|\vec{b}\|} \tag{1}$$

The second reward function $\mathbf{r_{time}}$, rewards agents based on the time they took to traverse the bottleneck environment, from start to finish. When leaving the environment, an agent will be rewarded $\frac{200}{\mathbf{t}}$ in which $\mathbf{t}$ is the amount of timesteps an AV spent in the simulation. To encourage cooperation, the agent is also rewarded based on this formula for other AVs leaving the simulation, but this reward will be shared over all AVs present in the simulation at that particular timestep. Since this results in AVs just waiting to get reward from other AVs, a punishment for every timestep was added. Furthermore, a punishment for other AVs being in the simulation was added in order to force AVs to move.

2.1.3 Reinforcement Learning architecture

An existing implementation of the popular *Proximal timization* (PPO) algorithm [Sch+17], provided by RLLib [Lia+18], is trained to learn a policy for the employed AVs. PPO is an algorithm that uses online stochastic gradient descent (SGD) to find an optimal policy for a given agent or ensemble of agents. The main concept of PPO is adopted from Trust Region Policy Optimization (TRPO) [Sch+15], which uses the probability ratio in Equation 2:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}, \tag{2}$$

where t is the timestep, θ the neural network parameters, π_θ the policy parameters vector, a an action and s the current environment state. TRPO then constrains policy updating by using Kullback-Leibler (KL) divergence [KL51]. For KL divergence to be effective in TRPO, a constant coefficient, β, controls the amount of constraint. However, β is difficult to find and is reported to be often specific for a given problem [Sch+17]. PPO, on the other hand, disregards KL-divergence and instead makes use of a clipped surrogate objective function that ensures that the learned policy updates within a given boundary at each step while being more generally applicable (i.e. to a wider set of problems). A large policy update in a given state might not always lead to a final optimal policy as the performed action in that particular state could potentially lead to a negative reward later on, therefore being a bad policy. By updating the policy in relatively small steps with the clipped surrogate objective this effect is diminished.

The parameter settings for PPO used in this research are provided in Appendix A.

2.2 Implementation details

For this research the simulation software Flow [Wu+17] is used in combination with the open-source framework Ray [Mor+18]. Ray is used for building and running distributed applications and includes RLLib [Lia+18], which is a framework to facilitate the easy application of

RL to a variety of standardized and publicly available learning environments and provides implementations of multiple popular RL algorithms, such as PPO.

A fully functional and worked out RL setup is shipped with the Flow's GitHub repository already. This setup allows for running RL learning experiments on a wide range of existing traffic simulations out of the box.

2.3 Experiment design

To answer the two research questions stated above, experiments were conducted in two phases. The first phase marked the exploratory research phase, which focused on getting a multi-agent RL environment to work correctly. While the second phase aimed at concretely answering the research questions.

2.3.1 Exploratory research

In the first phase of the experiments different reward signals and sizes for the artificial neural network (NN) architecture constituting the RL agent are compared to each other. The reward functions differ in their input-output behaviors, as described in section 2.1.2. Experiments were conducted with various network sizes: the smallest consisting of two hidden layers of 32 nodes and the biggest contained two layers of 256 nodes. Also, during this part of the experiments, the maximal magnitude of the reward signals tested varies in the range $[1.0, 10.0]$, as well as the the learning rate in the range $[1 * 10^{-6}, 1 * 10^{-3}]$.

Training outcome is assessed based on both objective and subjective measures during this part of the project.

The set of objective measures incorporates the accumulated mean reward per training simulation obtained by an RL agent over a certain amount of most recent training runs. A second objective measure is the mean number of time steps per training simulation an algorithm accomplishes controlling the RL vehicles without a single crash of cars inside the simulation, where the maximum number of time steps an agent can control a simulation is set to 1000 (i.e. the horizon of the simulation). Whenever this limit is exceeded or a crash happens, the simulation is terminated and a new simulation starts.

The set of subjective measures used in assessing the quality of training outcome during the exploratory phase of the project contained the observed training speed of the RL agent given a certain parameter setting. This means that, for efficiency reasons, test runs were aborted at certain points if the observed training progress seemed subjectively to be too low. This was done to save computational resources for continuing testing potentially more efficient combinations of the aforementioned training parameters swept. Another subjective means of assessing the quality of a learned policy, as described in [Vin+18] already, is judging an agent's policy by visually rendering a simulation in which the agent performs and observing how the autonomous vehicles in the simulation, controlled by the agent, behave. This might lead to clues why certain statistics obtained from a simulation diverge from given expectations.

Data obtained during the exploratory phase of the research is used to decide which parameter settings to use in the second phase of the research, which aims at concretely and purely objectively answering the stated research questions.

2.3.2 Objective research

For answering the research questions stated above, three sets of data have to be collected. First, a simulation must be run in which only human cars are present to receive a base line estimate of what the outflow rate of cars out of a given road condition is at present, and hence in a situation where there are still no AVs launched in most regions of the world. This outflow rate, serving as a reference value for the nowadays outflow rate for human cars, has been recorded 100 times, having randomized simulation onsets, over which the average has to be taken for the final conclusion to be less dependent on the simulation's specific initial onsets.

Then, two systems are trained on learning a policy that tries to increase the AVs' outflow rate as much as possible by controlling the AVs' acceleration and deceleration behavior, as well as the steering, i.e. lane changing, behavior of the AVs. One system is trained in a simulation where 10% of the vehicles are AVs, the other in a simulation where 50% are AVs.

Afterwards, the models will be evaluated by running the same simulation they were trained on 100 times. This means that the model that was trained with 50% AVs is also evaluated with 50% AVs. While evaluating we measure the total outflow and average times it takes human vehicles and AVs to traverse the simulation.

3 Results

3.1 Experimental findings

As described in section 2.1.2, PPO models were trained using the two different reward functions $\mathbf{r_{velocity}}$ and $\mathbf{r_{time}}$, respectively. After training, 100 simulations per model were run to evaluate the performance of the trained models. Due to failure to learn a functional policy during training of the models using the latter reward function, no results will be reported on the model using the $\mathbf{r_{time}}$ reward function. Therefore, only the results obtained from tests conducted on the three models using the $\mathbf{r_{velocity}}$ reward function are summarized below.

Table 1 summarizes the results obtained from the 100 evaluation simulations run per each of the 3 models evaluated using the $\mathbf{r_{velocity}}$ reward function. The models differed in their respective AV inflow rates. The first column of table 1 presents the AV inflow rate per model. To recap, each model was evaluated on the AV inflow rate on which it had been trained, previously. The second column presents the time steps needed for an AV to drive through the bottleneck test environment per AV inflow rate, averaged over the 100 evaluation simulations per model. Similarly, the third column indicates the average time steps human cars needed to drive through the bottleneck environment, averaged over the 100 evaluation simulations per AV inflow rate. The fourth column presents the overall outflow rate measured over both AVs and human cars per AV inflow rate, averaged over the 100 testing evaluations.

AV percentage (%)	AV average time	Human average time	Total outflow
0	-	310.26	1466.21
10	322.37	317.56	1370.45
50	378.92	375.89	1184.33

Table 1: The results for the models trained with $\mathbf{r_{velocity}}$.

3.2 Visual inspection of the learned policies

When visually inspecting a simulation run on one of the models trained using the $\mathbf{r_{velocity}}$ reward function, it is very hard to spot any difference between the behaviour of the AVs and the human drivers. For the models trained using the $\mathbf{r_{time}}$ reward function, this is not true. Figure 3 serves as an illustration of this. Figure 3 shows the AVs stopping, before they reach their destination. We suspect that the AVs learn this behavior in order to receive rewards for other AVs leaving the simulation, which is only possible if they stop and remain in the simulation instead of leaving it themselves. This demonstrates failure to learn a functional policy using the $\mathbf{r_{time}}$ reward function.

Figure 3: The AVs trained with $\mathbf{r_{time}}$ learn to stop early on in the simulation in hope of rewards from other AVs that don't stop. Red cars are AVs, white cars resemble human drivers.

3.3 Interpretation of findings

Table 1 shows that increasing the amount of AVs in the environment reduces the total outflow (i.e. the combined outflow of both human drivers and AVs). Furthermore, increasing the AV inflow rate increases the number of time steps needed to drive from the beginning to the end of the simulation for both human cars and AVs. In general, as can be seen from table 1, simulated human cars tend to be marginally faster than the trained AVs, since they require a consistently lower number of time steps to drive through the simulation.

These results are not in line with our expectations, as we hypothesized that a fleet of AVs with an objective to decrease their combined travelling time would come at a cost of increasing travelling times for the non-AVs in the environment. This is only partially true, however, since AVs did not even manage to achieve a number of time steps needed for driving through the simulation which is as low as that of non-AVs, while it is true that the number of time steps needed increased for non-AVs due to the introduction of increasingly many AVs, indeed. This can be interpreted as follows.

First of all, the tested control architecture implementing the presented multi-agent system design, proved to be able to learn to navigate through a traffic environment without crashing or necessarily overly affecting the overall traffic conditions. This observation is supported by the fact that the number of time steps needed for human vehicles, i.e. non-AVs, to drive through the simulation increased only marginally by introducing 10% AVs to the simulation.

For possible reasons discussed below, in section subsection 4.1, however, the used reinforcement learning agents did not manage to learn policies which were sophisticated enough to make the individual AVs excel in their task of minimizing their fleet's combined travelling time. That there were actual issues related to the learned policies is also evident from the observations described in subsection 3.2.

Therefore, it remains to be concluded that the research questions cannot finally be answered given the data collected and presented in this section, since the employed multi-agent control architecture did not manage to decrease the AVs' combined travelling times to a reasonable extent, while generally being capable, however, of controlling a fleet of individual AVs.

4 Conclusion

4.1 Discussion

As indicated in section 3.3, the experimental results obtained from the conducted experiments did not suffice to answer the previously raised research questions. To recap, the authors aimed at answering the questions of how much percent of the traffic an organization would have to have control over in a given area in order to lower the organization's combined travelling time significantly and how the emerging control architecture would affect the travelling times of the remaining vehicles present in the same infrastructure. To answer this question, a Reinforcement Learning (RL) algorithm was supposed to learn a policy for achieving the desired goal of reducing the combined travelling time for a set of autonomous vehicles (AVs) controlled by this RL agent. As is common practice in the domain of RL, trained agents did not have any prior knowledge about the task they had to learn. Therefore, the agent had to learn both the control of AVs in general, which included learning to collision-free steer and accelerate an AV, and the overall control law in order to achieve the desired significant drop in combined travelling time. While, as reported above, the RL agent successfully learned to control single AVs in such a way that they do not collide, the agent did not manage to learn to achieve the second goal of significantly lowering the combined travelling time for the set of AVs controlled by it.

Especially the observation that the combined outflow rate calculated over all vehicles in the simulation dropped due to the introduction of AVs leads to the conclusion that the overall RL agent had not learned a policy sufficiently sophisticated to answer the posed research questions by the time of evaluation.

This conclusion raises the question for the reason that caused the sub-optimal training outcome, i.e. the sub-optimal policy.

In trying to analyze what could be improved in further attempts to answer this paper's research questions, a few design choices made for this project may be addressed.

First of all, it is possible that the state representation provided to the employed RL agent did not contain a sufficient amount of relevant data necessary for the agent to learn a large-scale policy for optimizing traffic flow. One possibility for improving the training outcome could be to use visual state representations for training the RL agent, which avoids the need for having to handcraft state representations based on human intuition and expectations what kind of information might be useful and sufficient for an RL agent to learn a desired policy.

Increasing the dimension of the input data may in turn lead to the demand for a larger network size of the artificial neural network trained inside the RL agent. The used network size may be another design choice to be reconsidered, altogether, since, in general, rather small network architectures were used for training the RL agents in this research.

At this point, it is also not clear, whether all tested reward functions were not suitably designed for achieving the desired training outcome or whether the amount of training was often just not enough to make the AVs finally achieve the desired training outcome, i.e. the desired policy. The availability of larger and more reliable computational resources to the researchers could have allowed addressing these questions.

Also, the simulation environment used in this research, the bottleneck environment, is one being susceptible to occasional low-performance training outcomes, as reported in [Vin+18]. So, using another simulated training environment could have possibly led to better results as well.

Furthermore, the continuously low percentages of AVs introduced to the simulation during training may have harmed the system's overall performance, by presenting a possibly notoriously sub-optimal learning condition to the RL agent. While the research question demanded training a system on varying percentages of AVs in the simulation in order to find out which percentage leads to the most successful policy, low computational capacity available to the authors led to a much lower number of test runs and consequentially much lower number of varying AV percentages introduced to the simulation than anticipated beforehand.

Overall, a lot of factors may have prevented the RL agent from learning a policy that would have achieved answering the research questions. Trying to come up with better parameter settings is up to further research conducted in this area.

For a conclusive discussion of the multi-agent system aspect employed during this research, see section 4.2.

4.2 Relevance

Albeit this work did not achieve answering the posed research questions, this circumstance does not mean that the answer to these questions should be given less importance. At a time when companies are spending billions of dollars on investigating the potential of AVs and strategies for maintaining fleets of autonomous vehicles (AVs) in areas such as public transportation, the taxi industry, and logistics industry, for example, it is possible for companies to come up with control architectures for their AV fleets which benefit their own efficiency, but possibly at the expense of other parties involved in public traffic. Therefore, research can try to anticipate further developments in the domain of large scale traffic control system design to guide politics in designing legislation in a way such that no party overly suffers from the large scale launch of AVs to be expected in the future. To generalize the thought outlined above, it can be noted that actions will have to be taken for designing assessment procedures for control architectures for AVs, since not only large-scale egoistic design architectures could worsen traffic conditions, but also generally improper designed control architectures that have not been tested thoroughly enough by the company providing them.

In conclusion, it can be said that, regardless of the motivation, research will eventually have to address the question of how to assess the quality of control architectures for AVs which will eventually affect the lives of billions of people involved in public traffic. The authors' attempt to shine a light at possibilities of how future egoistically designed control architectures for AVs could affect the overall traffic situation is just a small contribution, which aims to pave the way for answering the larger questions raised in the paragraph above.

Another contribution made by this paper is the attempt to transition away from state-of-the-art centralized single-agent control architectures for controlling an entire traffic scene towards a distributed multi-agent control architecture controlling individual AVs in a scene. In case of the former control architecture, a single agent observes an entire scene and assigns actions to each of the controlled agents in a scene. In the latter control architecture, the RL agent, being jointly trained by all AVs in a scene, is evaluated for each controlled AV individually, where the selected actions to be taken by an AV depend on the private percepts of the respective AV. The advantage of the latter, distributed control architecture is that it can be applied to an arbitrarily large number of AVs without having to make any prescriptive assumptions about how many AVs could be controlled maximally at each time step at most by a controller. Moreover, the distributed multi-agent-based control architecture used in this research has the advantage of being more realistic in that it is more affordable in many

respects to observe the surrounding of a single car rather than observing an entire scene for deciding on the single actions assigned to each controlled AV. This makes the distributed control architecture more likely to be used real life settings in the future. By being more likely to be used in real life settings in the future, using the control architecture used in this research, given improvements up to further research, may lead to more realistic simulation outcomes for future traffic scenarios. This is because the dynamics of single-agent control architectures that have an overview over an entire scene may be very different from those observed in more realistic, distributed multi-agent-design-based control architectures. Therefore, following the methodology of using distributed multi-agent control architectures for controlling large fleets of AVs, as demonstrated in this paper, may lead to more accurate predictions concerning future traffic conditions. For a discussion of the problems related to the usage of state-of-the-art control architectures based on centralized single-agent designs, nowadays often used in research involving traffic simulations, see [Vin+18].

4.3 Team Work

The division of labour is presented in Table 2. Moreover, there were multiple group meetings. New findings, in the broadest sense, were communicated over WhatsApp. In retrospect, we spent a lot of time individually figuring out how Flow/RLLib works. We should have coordinated this a bit better to save overall time spent on this.

Group member name	Contribution
Thomas	**Code:** reward function **Report:** alpha/beta versions, Method: Reinforcement Learning architecture, parts of Introduction, proofread entire report **Literature:** initial literature research assignment, peer review assignment, PPO **Presentation:** design of preliminary slides
Ewout	**Code:** State representation, environment, reward function (all pair programmed with Thijs), setting up and training using Colab **Report:** Parts of discussion, image(s), proof reading, GitHub documentation **Presentation:** Redesign of slides (with Thijs) + slides for experimental design and results in particular
Daniel	**Code:** Implementation and testing of different reward functions and state representation methods **Report:** Abstract, part of Introduction, Experiment design, large parts of Results section, and Conclusion **Presentation:** Contribution to first draft of presentation design **Literature:** Selection of paper on AV research around the City of Berlin
Thijs	**Code:** State representation, environment, reward function (pair programmed with Ewout), scripts for retrieving results **Report:** Method: state/reward sections, results section **Presentation:** Redesign of slides (with Ewout) + slides for discussion/conclusion in particular

Table 2: Work division among group members

References

[KL51] Solomon Kullback and Richard A Leibler. "On information and sufficiency". In: *The annals of mathematical statistics* 22.1 (1951), pages 79–86.

[Kno+12] Florian Knorr, Daniel Baselt, Michael Schreckenberg, and Martin Mauve. "Reducing traffic jams via VANETs". In: *IEEE Transactions on Vehicular Technology* 61.8 (2012), pages 3490–3498.

[Lia+18] Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph E. Gonzalez, Michael I. Jordan, and Ion Stoica. "RLlib: Abstractions for Distributed Reinforcement Learning". In: *International Conference on Machine Learning (ICML)*. 2018.

[Low+17] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. "Multi-agent actor-critic for mixed cooperative-competitive environments". In: *Advances in Neural Information Processing Systems*. 2017, pages 6379–6390. URL: `https://papers.nips.cc/paper/7217-multi-agent-actor-critic-for-mixed-cooperative-competitive-environments.pdf`.

[MB16] Michał Maciejewski and Joschka Bischoff. "Congestion effects of autonomous taxi fleets". In: *Transport* 33 (2016), pages 971–980. DOI: 10.3846/16484142.2017.1347827.

[Mor+18] Philipp Moritz et al. "Ray: A distributed framework for emerging {AI} applications". In: *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*. 2018, pages 561–577.

[Sch+15] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. "Trust region policy optimization". In: *International conference on machine learning*. 2015, pages 1889–1897.

[Sch+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[Vin+18] Eugene Vinitsky, Aboudy Kreidieh, Luc Le Flem, Nishant Kheterpal, Kathy Jang, Fangyu Wu, Richard Liaw, Eric Liang, and Alexandre M Bayen. "Benchmarks for reinforcement learning in mixed-autonomy traffic". In: *Conference on Robot Learning*. 2018, pages 399–409. URL: http://proceedings.mlr.press/v87/vinitsky18a/vinitsky18a.pdf.

[Wu+17] Cathy Wu, Aboudy Kreidieh, Kanaad Parvate, Eugene Vinitsky, and Alexandre M Bayen. "Flow: Architecture and benchmarking for reinforcement learning in traffic control". In: *arXiv preprint arXiv:1710.05465* (2017).

A PPO architecture

The final PPO architecture was made of two fully connected layers of size 256. Other hyperparameter settings are included in Table 3.

Hyperparameter	Value
Learning rate	$2 * 10^{-5}$
Horizon	1000
Minibatch size	4000
Discount (γ)	0.999
AV inflow rate (vehicles per hour)	2300

Table 3: Overview of the hyperparameters used in our experimental setting.