Selflessness of vehicles in mixed autonomy driving

Nicoló Brandizzi (s4171632)
Kenichi Furusawa (s3204146)
Ivo de Jong (s3174034)
Subilal Vattimunda Purayil (s3587630)

Final version, November 3, 2019

Abstract

As the number of autonomous vehicles has been increasing over the last years it has become important to investigate how different implementations of autonomous agents affect the flow of traffic. In previous research the interaction between autonomous and human vehicles did not show a notable influence on the flow of traffic compared to exclusively human traffic. To expand upon this, this paper exhibits a simulation of mixed-autonomy traffic, where the autonomous vehicles can be programmed to be more or less selfish. The simulation shows that selfish vehicles have an advantage over selfless vehicles, but that they were detrimental to the overall flow of traffic.

1 Introduction

With the steady increase of (semi-)autonomous vehicles being introduced into traffic a need for understanding of how this will affect traffic arises. While it is likely that these cars drive safer and more reliable than humans as the risk of human errors is minimized, there are still complex challenges on interaction that should be investigated. One of these problems, as previously investigated[BLB], is how these autonomous vehicles might interact with human-driven vehicles. Building onto this research, this paper exhibits a simulation which investigates an implementation choice to be made for autonomous vehicles, and how this will affect the flow of traffic.

1.1 Problem

We investigate how the artificially implemented selfishness of autonomous affect on the flow of traffic. Previously, when traffic consisted exclusively of human drivers, drivers may have behaved more or less selfish to get home faster than other people by cutting people off, not giving right of way, or even performing illegal manoeuvres. Now that autonomous vehicles are being introduced we can investigate and control how selfish a vehicle might be. For example, a car might yield to let a large lane of multiple cars pass. This will give an advantage to many other vehicles, with only a disadvantage for the yielding vehicle. In contrast, a vehicle might not yield to get themselves home faster, even though that means a lot of other vehicles would have to wait for it.

If strategies are implementable to create selfish behaviours, then producers of autonomous vehicles would be incentivized to offer vehicles with a “selfish“ feature, getting their customers

home faster. If this does in fact have a negative impact on the flow of traffic selfish cars would form a race to the bottom. To counteract this, regulations would need to be in place making such selfish features illegal or regulated. In order to determine the value of such regulations, scientific evidence of the effects of selfish vehicles would hold a strong argument. This paper might present such scientific evidence of the impact of selfish autonomous vehicles on traffic.

1.2 State of the art

In "Simulating Autonomous and Human-Driven Vehicles in Traffic Intersections" [BLB] the authors simulate the interaction between cars driven by Autonomous Agents (AA) and Human Agents (HA). The difference between these agents is that the HAs have a limited field of view, and a longer reaction time, whereas AAs have a full range field of view and a very short reaction time.

The simulated environment was defined as a network of streets that cars have to find routes from the agents' location of departure to their destination. The decisions (accelerating, braking, reverse, none) made by the agents were controlled by agents' parameters defined before hand. The autonomous agent had 360 degrees of vision with a larger range compared to "human" agents which only had front visual field and smaller range. Moreover, autonomous cars had a faster reaction time than "human" agents. Their simulated environment did not model traffic lights, but conserved the rule of giving priority to cars coming from the right. There were no overtaking behavior simulated. The purpose of the simulation was to investigate whether autonomous cars joining "human" traffic would cause any delay. Hence, it was important to have an intersection environment where both agents interact and measure the traveling time of each driver.

During each experimental step, the ratio of human drivers and autonomous cars was manipulated. The traveling time of each cars from source to destination was taken as a measure of performance. Destination and source were selected at edge of the map. This is because average speed would not be a consistent measurement throughout the different environments due to number of sections.

Their simulation barely shows any difference in traveling time between the two types of drivers. From this they concluded that the properties field of view and reaction time are not distinctive properties that influence the flow of traffic. This manipulation made autonomous cars better at gaining visual information from the virtual world and the efficiency of processing of information. One interesting side-effect found was that the human driver was taking advantage of safe and predictable behaviour of autonomous cars in a zip-merging scenario, making them slightly faster, but this was likely a consequence of traffic modelling rules.

1.3 New idea

Expanding upon previous research, we add the selfishness and selflessness of the autonomous vehicles as an independent variable. While the difference between HAs and AAs as previously introduced would remain the same, different kinds of AAs will be implemented. The AAs will either be trained to get themselves to their destination as soon as possible, or they will learn to get everyone to their destination as soon as possible. This can provide more insight in what exactly the goal of an autonomous vehicle should be. While drivers, and therefore makers, of autonomous vehicles might have a preference for selfish goals, this might be in direct in conflict of societal goals for traffic.

2 Method

2.1 Simulation model

The simulation expands the original experiment by [BLB], where they only simulated a few singular intersections, our simulation runs on an grid map. Since the goal of the simulation is to determine the effect of self-centered and cooperative AAs on the flow of traffic, it is important to make the simulation of traffic expansive, rather than only looking at a single intersection.

Moreover, we improved the navigation mechanism of the autonomous agent by controlling it through Reinforcement Learning. [BLB] found that their autonomous agents would be cautious as an unintended consequence of the rules they were following. Giving the control of the agents away to a Reinforcement Learning approach minimizes the unpredictable behaviour caused by consequences of implementation details. The Reinforcement Learning approach also ties in nicely with the difference between self-centered and cooperative AAs, since the reward function can be mapped nicely to these goals. This will show whether the agents are able to maximize their rewards effectively, as a consequence of the other agents in the system. Naturally, this also ensures that the implementation details on what selfish driving actually entails are also simplified to a reward function, so that we can again avoid unpredictable behaviour caused by implementation details.

2.2 Implementation details

2.2.1 Framework

Our simulation is built on the Flow framework [Wu+17] whereas the previous experiment used the Box2D python physics library. Flow is a collaboration project between UC Berkeley's Institute of Transportation Studies and Aimsum, a subsidiary of Siemens. They developed a framework intended for applying reinforcement learning to traffic simulations. It specifically tries to form a consistent benchmark for any mixed-autonomy simulated research. It is built on traffic simulator SUMO [Beh+11], as developed by the Institute of Transportation Systems in Berlin.

This places our simulation in a well established and well recognized framework, which is beneficial for comparing results between independent publications.

2.2.2 Environment

The traffic lights are not simulated in the environment. Traffic lights take away any decision making that drivers can make, as the lights determine who goes when. This is in conflict with the goal of our experiment, so the traffic lights were removed.

2.2.3 Reinforcement Learning

As our self learning agents used reinforcement learning, they have to be given a state representation, a set of actions and a reward function. These are applied to the well recognized Reinforcement Learning framework OpenAI Gym [Bro+16].

As a visual aid to accompany the explanation on how reinforcement learning is implemented a zoomed-in image of what a SUMO intersection may look like is provided in figure 1.

The state is represented by normalized values representing the following:

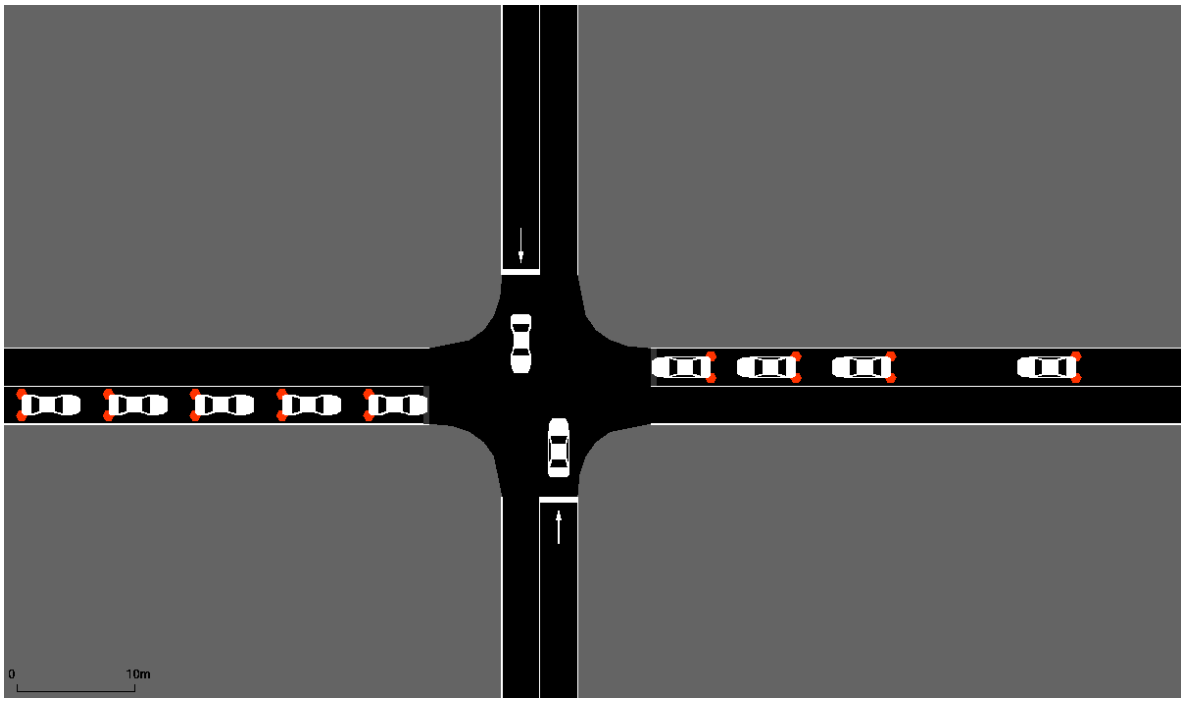


Figure 1: An intersection as simulated in SUMO

- The speed of the agent
- The difference between the speed of the agent in front and the current agent
- The distance between the agent in front and the current agent
- The difference between the speed of the current agent and the agent behind
- The distance between the current agent and the agent behind
- The number of cars in front and behind the agent
- The average speed of the cars around the agent
- The average acceleration of the cars around the agent

The actions that the agent may take are decelerating and accelerating. The maximum deceleration is $3.5m/s^2$. This is found to be the maximum comfortable deceleration. [MB12] The maximum acceleration is $10m/s^2$. This is bound by the acceleration the car can give rather than human comfort, as cars generally can not create enough accelerations to put human through problematic g-forces.

The interaction of accelerating and decelerating vehicles will form an implicit communication between cars at an intersection. If the other car is stopping for an intersection, it means that you can go. Conversely, when the other car does not decelerate you know that you should probably decelerate. This is how the cars are managing giving and receiving priority.

The reward function is defined based on the cooperativeness of the agent, as alluded to before. Specifically, the reward function can be mathematically defined as

$$R = max(-w * (D_s + S) - D_a - J + C, 0) \tag{1}$$

where R represents the reward, w represents the selfishness-coefficient, D_s represents the delay of the agent, S represents the time the agent stood still, D_a represents the average delay of all cars in the simulation and J represents the jerk. Jerk is the speed at which acceleration increases, and a high jerk is found to be uncomfortable. Jerk can be defined as m/s^3. Lastly, C simply represents a base positive since the Reinforcement Learning algorithm expects a positive value, but all the measurements are negative properties. This value needs to be consistent over agents and generations, and simply be larger than the maximal sum of the other components.

This formula creates a tune-able selfishness coefficient, where we can change the value between 0 and 1 to make a car more or less selfish. A very selfless car $w = 0$ only cares about every ones delay (and the jerk), but does not prioritize their own time in any way. On the other hand, you might find $w = 10$, where the agent finds its own time to be 10 times as important as that of other vehicles. This would result in different driving behavior.

We include population-based Scheduling for scheduled training process which starts like random search [Jad+17]. The training starts with many neural networks in parallel with random hyper-parameters. Then,the scheduler uses information from the population of networks to adjust the hyper-parameters and allocate computational resources to model with the most promise. This method is inspired by genetic algorithms. This was implemented by using library Tune [Lia+18].

Algorithms Different Q-learning algorithms are effective at solving problem in various problems with discreet action spaces [HS19]. Unfortunately, the actions space the agents will be operating in is continuous, as they can determine the scale of their acceleration. While this can be discretized through binning, a conflict arises between the desire of low dimensionality while maintaining the ability to make smooth decisions. Algorithms more fit for continous action spaces are Proximal Policy Optimization (PPO) [CY17] and Deterministic Policy Gradients. An adaptation of Deterministic Policy Gradients called Multi Agent Deterministic Policy Gradients (MADDPG) [Low+17] would be most promising for this problem as it is specifically designed with multi-agent systems in mind. It was also found by [CY17] to be more effective than PPO at solving problems in multi-agent systems.

Following these findings, we applied MADDPG as the reinforcement learning algorithm, which was available in the OpenAI Gym environment. MADDPG works as an adaptation of deep Q-learning. Firstly, the deep Q-learning is adapted to produce a continuous output, rather than a discreet one. This forms the Deep Deterministic Policy Gradient (DDPG) component. The problem that this still has is that Q-learning assumes a consistent task to learn. However, because the other agents change their behaviour and their learning, the environment for any single agent changes as the multi-agent system evolves. To combat this, the critic component (from the actor-critic collaboration in Q-learning) is given the information about all the other agents. This allows the learning of new policies to consider the changing environment, so that there are no hidden changes, but simply different inputs. This solves the problem that DDPG had and forms MADDPG. Of course in this case the actor component still should not have information of all the agent. If that were the case then the agent would be able to adapt their

decision based on what every single vehicle on the map is doing, which is not what we try to simulate.

2.3 Experiment design

We run the simulation with different populations of different selfishness. For simplicity we binarize the selfishness, and only look at how the amount of selfish cars affects the flow of traffic. We may start with all cooperative agents, and increasingly make more of the agents selfish. At some point we expect to notice a near total gridlock, as all cars will refuse to give priority.

We expect that when there are few selfish cars, that they will have an advantage over the selfless cars, but as the number of selfish cars increases, total delay will increase. The selfless cars will still have a worse time than selfish cars. The real life consequences of this could be that car manufacturers might create a race to the bottom, where every new generation of cars is more selfish than the next as it shows to bring people home faster, but the cumulative effect of this would be that everyone is home later than if we all had selfless cars.

It is hard to determine when the aforementioned total gridlock would occur, as it is strongly related to the intersection density of the map, and exactly how selfish the behavior of the selfish vehicles ends up being.

Parameter	value
Neighbors distance	50
Duration of one episode in steps	500
Coop weight	1
Selfish weight	0
Learning rate	1e-4
Discount rate	0.998
Map	Grid

Table 1: Setting for the experiment.

The setting for the experiment can be referred to table1.

3 Results

In this section results are discussed. We will start by describing the two kinds of environments for the MADDPG [Low+17] algorithm, their difference and their similarities.

3.1 MADDPG

The MADDPG algorithm was used for two main simulations distinguished by the number of human vehicles. The first one has a total of 300 human drivers while the total number of RL agents is fixed to 30, but varies between selfish and cooperative agents. The second one has 50 humans vehicles and again 30 RL agents.

This difference in the amount of human drivers has been implemented in order to measure the impact in delay given by the total percentage of RL cars on traffic. Following this we can calculate that in the first run, the autonomous agents represent just 9% of the traffic flow in

the simulation, while in the second simulation the percentage of autonomous cars is more than 37%.

Before diving deep in the results we introduce the following syntax for number of different drivers in the system:

$$[CV, SV, HV]$$

Where:

- **CV**: number of cooperative autonomous vehicles.
- **SV**: number of selfish autonomous vehicles.
- **HV**: number of human vehicles.

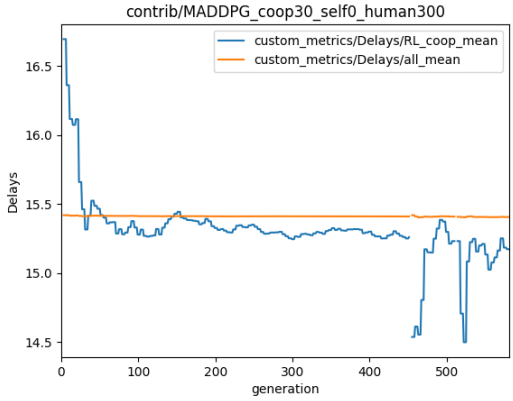
3.1.1 300 HV

As previously stated the percentage of RL vehicles in the system should be proportional to the average delay experienced in the system. For this reason we expect to find little to no difference when varying the number of autonomous agents in this simulation.

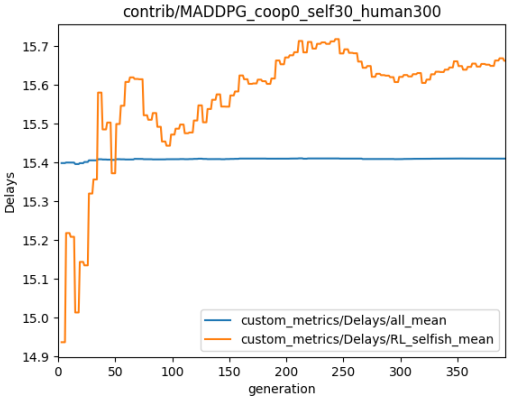
For this run the numbers of autonomous vehicles varied accordingly, representing respectively CV, SV and HV:

1. (0,30,300)
2. (20,10,300)
3. (30,0,300)

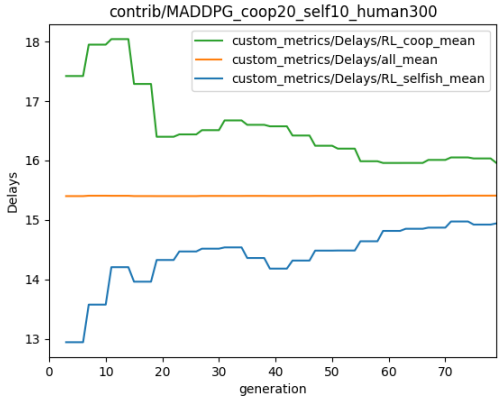
This does not cover a distribution of [15,15,300], which could be investigated in further studies.



(a) Cooperative environment



(b) Selfish environment



(c) Mixed environment

Figure 2: Delays for simulations with 300 humans

Delays Before looking at the graphs, we need to clarify that the *all_mean* delay is the average between all the RL agents and the human ones too. Since the number of human drivers is much larger than the number of autonomous drivers, there will be no notable change in this value throughout the simulation.

As you can see from the above image the mean delay for the environment containing only selfish agents (figure 2b) is slightly more than the one containing just cooperative agents (figure 2a).

Moreover it is interesting to notice how the two graphs have a symmetrical trend in respect to each other. In the beginning, cooperative delay starts higher than the mean and decreases over time, while the selfish one does the opposite. We can attribute this behavior to the cooperative part of the reward function for the *CVs*. Initially the *CVs* give higher priority to human drivers without any kind of cooperation between each-other, so the delay for them is much longer than the mean. Later during training they start to learn how to cooperate with each-other to keep the human delay constant while decreasing the individual delay, effectively taking into account the selfish part of their reward function.

On the other hand *SVs* do the opposite. They start by considering their own delay, thus driving more "recklessly". This kind of behavior likely leads to an increase in crashes during

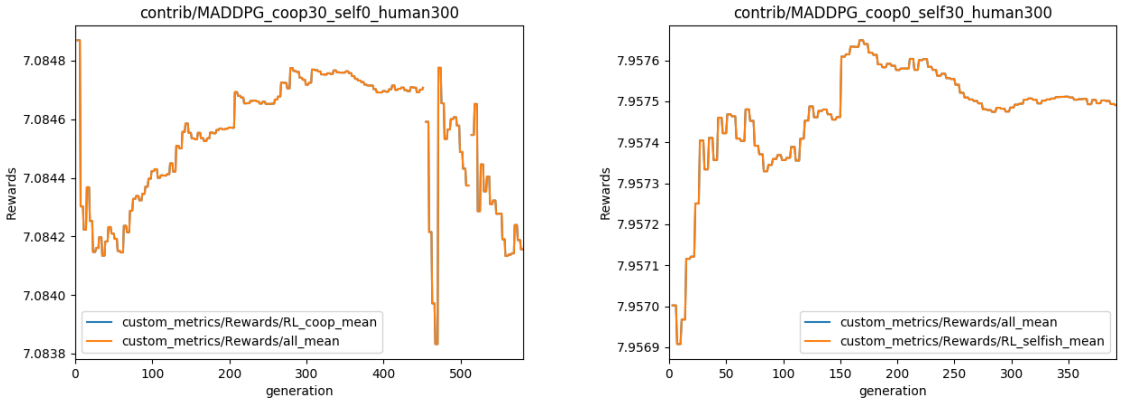
the learning phase (though we were not able to investigate this). When a crash occurs every agent is rewarded zero and the simulation ends. To avoid this the *SVs* learns to drive less recklessly, resulting in longer delays.

Finally, figure 2c repeats this trend for *SVs* and *CVs* as previously described.

It is important to notice how the initial values for both autonomous agents are more polarized than in the previous simulations. Cooperative vehicles that started around 17 seconds delay (Figure 2a), now start at 18 seconds; while selfish agents having 15 seconds delay in Figure 2b now start from 13. This is probably due to the exploitation of cooperative agents by selfish ones.

Finally, as can be seen by the end of the training phase, both *CVs* and *SVs* converge to the total mean. This is both due to the large number of human vehicles, which forces the agents in a specific range of delays; and due to RL agents adapting to the environment, with *SVs* causing less crashes, thus having longer delays, and *CVs* cooperating more effectively.

Rewards In this section we investigate the reward trends for both cooperative and selfish environment.



(a) Cooperative environment

(b) Selfish environment

Figure 3: Rewards for both selfish and cooperative environments with 300 human vehicles

Figure 3 shows that the reward trend is strictly tied with the delay trend. In line with this we can see that for the cooperative simulation (figure 3a) the reward starts high and has a sudden decrease, exactly as seen for the delay.

Later rewards show an upward trend. This is caused by two factors.

One of these factors is that the RL agents are learning how to drive and using correct acceleration actions. The other is an increase in cooperation between the agents themselves.

It is interesting to notice how the reward seems to drop again right before the simulation ends. This can be due to the typical oscillatory trend that reward functions tend to have during training, increasing the horizon could potentially stabilize this oscillation.

On the other hand the selfish rewards (figure 3b) tends to stabilize faster than the cooperative one, having many more similarities with the delay trend. Since the reward function for the *SVs* is more dependant on their delay rather than the other factors, this similarity is follows naturally.

Finally we can observe how the mean reward value for the *SVs* is 12% greater than the *CVs* one. This is expected because *SVs* have one less factor which can decrease their reward.

Jerks Finally we want to focus on the jerk value throughout the simulation. As explained previously, the jerk is defined as the derivative of acceleration. This means that if acceleration increases or decreases quickly jerk is higher. A high jerk is indicative of an unpleasant experience for a human passenger.

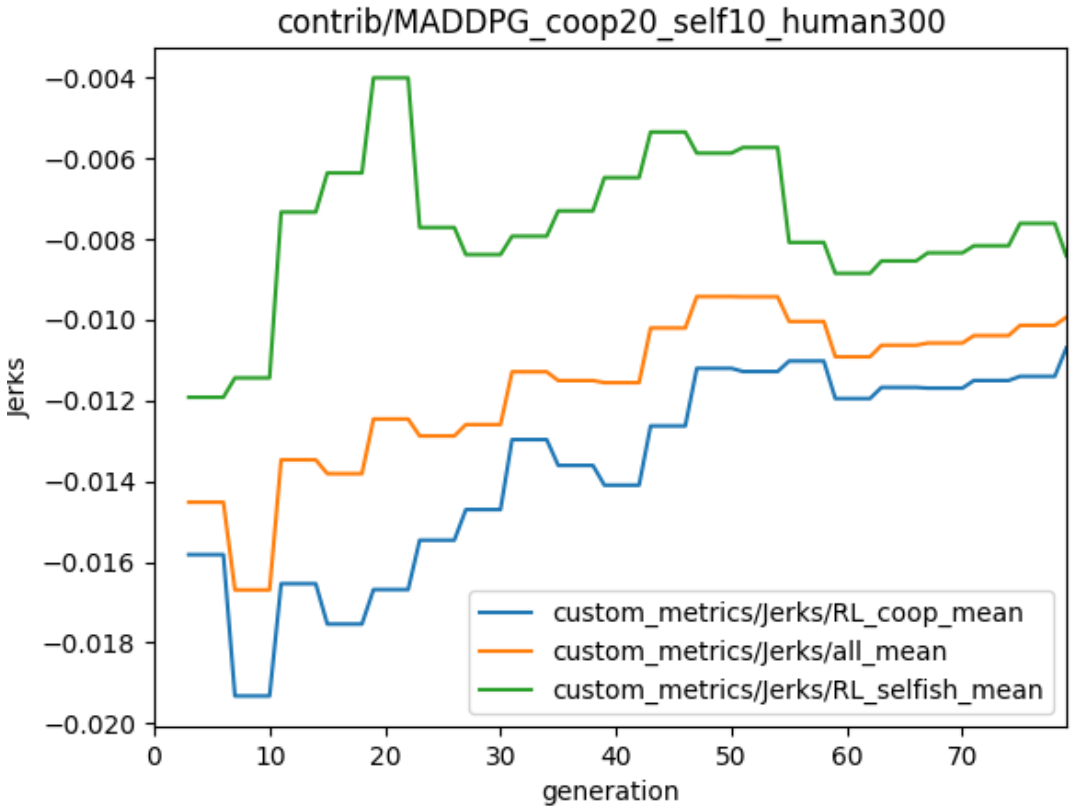


Figure 4: Jerk trends in mixed environment

As can be seen from the figure 4, the jerk for selfish driver starts higher and then lowers towards the mean value. Higher negative jerk values indicates less frequent deceleration which is exactly what we have seen in the previous graphs, which we called "reckless" driving.

On the other hand cooperative jerk begins very low and rises up later in the simulation. Again this is accordingly to what we previously said, that is that *CVs* tends to give priority to every other vehicle at the start until they learn how to cooperate better, having a positive impact on the entire simulation (mean jerk rises too).

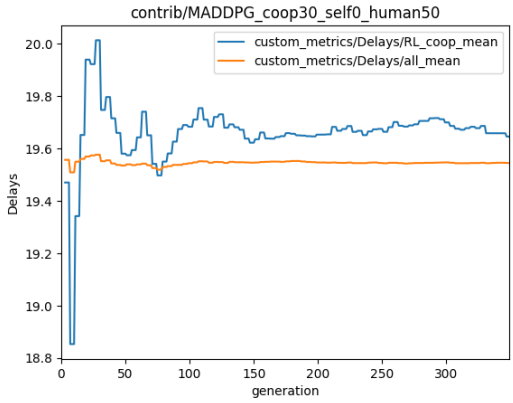
3.1.2 50 HV

In this section we will investigate the impact the proportion of RL agents have on the flow of traffic. For this run the numbers of autonomous vehicles varied accordingly, representing

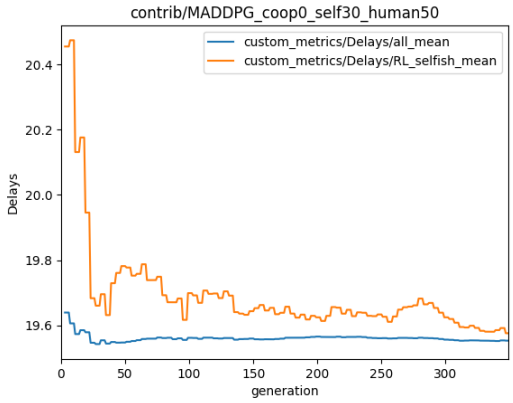
respectively CV, SV and HV:

1. (0,30,50)
2. (15,15,50)
3. (30,0,50)

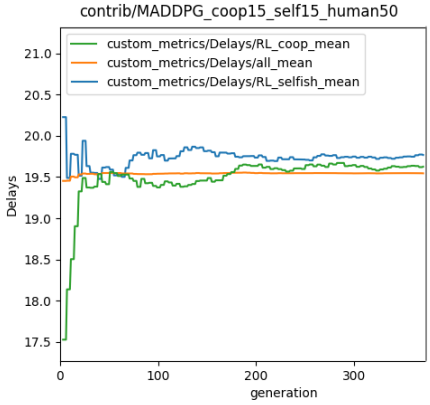
Delays Below the plots for the environment delay are shown.



(a) Cooperative environment



(b) Selfish environment



(c) Mixed environment

Figure 5: Delays for simulation with 50 humans

As can be seen from the figure 5, the cooperative and selfish trends are inverted in respect to the previous case in which we had 300 human vehicles.

Regarding *CVs*, we have an initial low start followed by a sudden increase and then a stabilization slightly above the average line (figure 5a).

In this scenario communication and cooperation between agents have a huge impact on the overall simulation. Cooperative agents quickly learn to sacrifice their speed to favor the full system. The mean delay does not seem to change. However, looking closely you can see that it is decreasing steadily. With more training steps one could see a more accentuated decrease in total delay.

On the other hand the selfish simulation experiences the opposite, Figure 5b. Selfish drivers learn to take advantage of human vehicles, thus decreasing their delay while the mean delay slightly increases. It is important to notice that the decreasing in the agents' delays is not proportional with the increasing in the human delays. This is due to selfish vehicles learning how to drive better, thus causing fewer crashes.

Fewer crashes results in less delays for everyone, which can be seen from the first 50 steps of the simulation in which both the selfish and mean delays decrease. Longer training steps would show how the *CVs* delay gets lower than the mean delay.

Finally, the same trend as shown in the 300 humans simulation can be seen from the mixed environment (figure 5c). It is interesting to see how the agents do not have a notable impact on the average delay.

Jerks Lastly, the following plot shows the trend that the jerks have for this simulation.

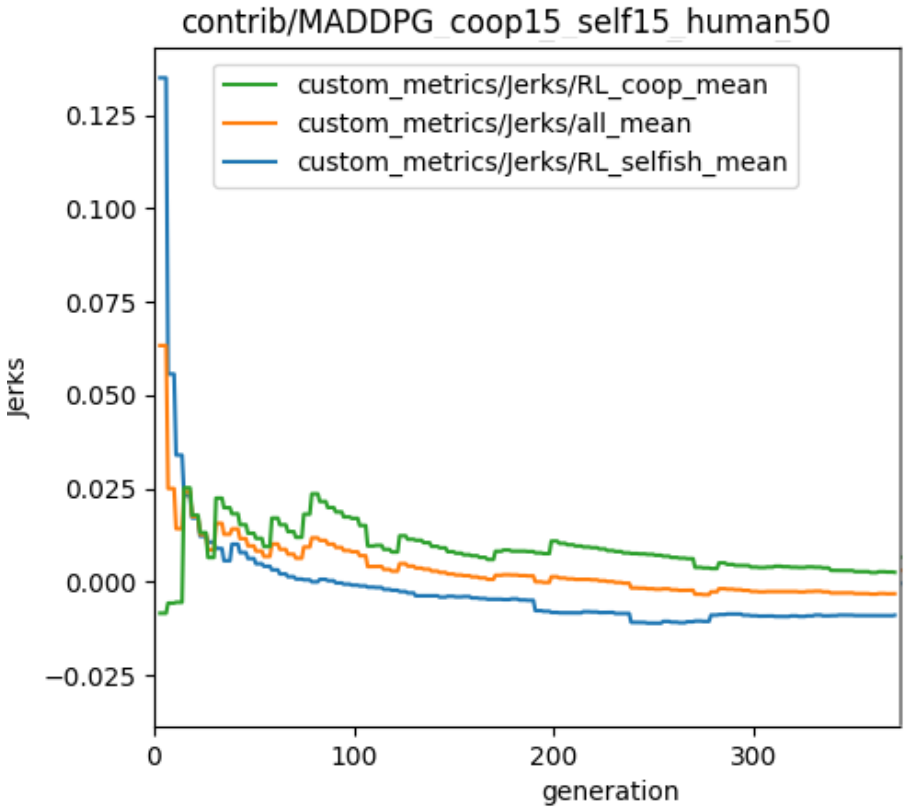


Figure 6: Jerk trends in mixed environment

Since there are fewer cars in the system, while the environment size remains the same, there is less traffic. This can be seen from the jerk values being positive (acceleration) rather than negative (deceleration).

Again we can observe the same trend for *SVs*. They start by accelerating without any regard for other vehicles (i.e."reckless" driving) and then stabilize around zero, constant acceleration.

On the other hand *CVs*, start by decelerating (negative jerk) and, surprisingly accelerate more than their selfish counterpart.

4 Conclusion

Based on the previous section we found that selfish autonomous vehicles have a negative effect on the flow of traffic, as the amount of delay is increased. While cooperative agents seem to do better than selfish agents, convergence does occur. This would lead to a result where, according to our simulation, any fully developed selfish vehicle would not have a particular advantage or disadvantage in travel time compared to cooperative vehicles. An interesting find in this graph is that the delays of cooperative agents increases as the delay of selfish agents decreases. This is likely a consequence of competition, where the selfish vehicles will get in the way of the cooperative vehicles, making them slower. Lastly, we might be able to conclude that at 25% of autonomous vehicles being selfish there is no overall disadvantage to the cooperative cars compared to an exclusively cooperative environment. There does appear to be a slight increase of delay at the end of training for the mixed environment compared to the homogeneous environment, but it is hard to determine whether this is a real trend that developed over learning, or simply a statistical anomaly.

4.1 Discussion

There are still some vulnerabilities in the results that may make it hard to come to the correct conclusion. In particular the comparison between mixed and homogeneous environment seems to be indicating an effect that it is not able to present fully. This could be explored by running a similar simulation longer, or by increasing the number of selfish vehicles in the homogeneous environment. These would likely amplify this effect, if it exists.

It is also very surprising that selfish agents do need seem to be able to gain an advantage over cooperative agents in a mixed simulation, but seem to converge. One would expect that, given the other results and reward structure, selfish agents would be able to take advantage of the cooperative vehicles, making them faster. A possible explanation for this is that a big component of the delay are cars queuing behind other cars at an intersection. In these scenarios the selfish or selflessness of vehicles is not able to influence their performance. A possible investigation could be done into what component of the delay is caused by queuing at the intersection, and what is caused by the decision to wait when at the front of the intersection. Moreover, the delay does not seem to have reached a static value at the end of our training session, so a longer training might be able to see the results flipped after the first 20000 generations.

Lastly a notable vulnerability in a reinforcement learning approach as this is that the a policy may reach a local optimum. In such a case better policies may exist, but be too different from the current policy that they do not get explored. Therefore, the performance of a reinforcement learning session may be more or less influenced by the starting values. While it is hard to determine whether this is the case, it could be explored by running several training simulations from scratch and see how they even out. This would also allow the exploration of whether there actually is much influence of the initial values in this particular task. Unfortunately this is beyond the scope of this project, as the time it takes to run a single training session at this length is already a day for an average desktop. This would mean that taking around 10 training iteration from scratch for each possible simulation could take about

a month of computation time given the means. Of course this could be alleviated largely by parallel computing systems like a supercomputer. Given that these trainings could be run in parallel to begin with, and the individual trainings also have a large parallel component to them, this experiment should be able to run quite quickly in a large parallelized system.

4.2 Relevance

As our simulation concludes that selfish cars have a detrimental effect on the overall welfare of traffic, we can conclude that letting the free market run its course is a very ineffective approach to solving the problem of traffic. Instead, it is better to strongly regulate the selfishness of vehicles as it becomes programmable into vehicles.

While the current research focuses on a traffic situations of communicating autonomous vehicles without intersections, which is not the world we live in today, we can still extrapolate the information to the current world. Semi-autonomous vehicles are in traffic today, and some are even able to change lanes on the highway. While we were not able to show a benefit to selfish cars in our simulation, there could very well still be one, so it is not unlikely that there are some selfish design decision in the development of semi-autonomous vehicles already. We should start to regulate the selfishness of semi-autonomous vehicles sooner rather than later.

To support policy makers in making the decision, a more specifically current simulation could be developed. In this simulation a specific focus can be put on the acceleration, deceleration and lane switching behaviour of self driving vehicles, and identify selfishness problems in this. Complementary research can be done by analyzing the behaviour of the current semi-autonomous vehicles in practice, to identify whether this selfishness is already implemented.

4.3 Team Work

Due to lengthy problems of the installation of the simulation framework Nicoló was the sole developer of the simulation. There were many long lasting attempts to get the framework working on all devices, but unfortunately only Nicoló was successful. Moreover Nicoló was the only writer of the Result part of the report given his knowledge on the environment. To keep some balance in the contributions Kenichi and Ivo focused on the writing and evaluating of the report. To ensure a good relation between the simulation and the report Nicoló kept a journal of development decisions make, which were then translated into the report.

Specifically, the introduction sections was written by Kenichi and Subilal, while the method, conclusion and discussion were written by Ivo. Given some feedback on language and coherency, Ivo fixed problems w.r.t. language, style and coherency.

Originally, the results section was written by Kenichi, but Nicoló did not consider this satisfactory as it was not written coherently and not all available graphs were used. As a result, Nicoló rewrote the results section, and Ivo went over the section to fix formatting and language errors.

The final presentation was mostly prepared by Subilal. Before the presentation the full team got together to optimize the presentation to the individual presenter's preferences.

Additional work that was done is that Ivo and Subilal analysed the results from the simulation and developed the presentation graphs accordingly. On the other hand Nicoló created the graphs used in this report.

A strong point for evaluation is to have picked a far lighter framework with fewer components in hindsight. It would have also been worthwhile to be a bit more exploratory before committing to a framework early in the development process. This would have allowed us to switch to a different framework before it was too late to turn back. This exploration would then include attempting to set up a development environment for all team members.

References

[Beh+11] Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. "SUMO–simulation of urban mobility: an overview". In: *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind. 2011.

[BLB] Laura van de Braak, Jelmer van der Linder, and Laura Baakman. *Simulating Autonomous and Human-Driven Vehicles in Traffic Intersections*.

[Bro+16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. "Openai gym". In: *arXiv preprint arXiv:1606.01540* (2016).

[CY17] Xiangxiang Chu and Hangjun Ye. "Parameter Sharing Deep Deterministic Policy Gradient for Cooperative Multi-agent Reinforcement Learning". In: *CoRR* abs/1710.00336 (2017). arXiv: 1710.00336. URL: http://arxiv.org/abs/1710.00336.

[HS19] Travis Hammond and Dirk-Jelle Schaap. *Forest Fire Control with Connectionist Reinforcement Learning*. July 2019. URL: http://fse.studenttheses.ub.rug.nl/20394/.

[Jad+17] Max Jaderberg, Valaentin Balibard Simon Osinder, Wojciech M. Czarnecki, Jeff Donahue, and Ali Razavi. "Population Based Training of Neural Networks". In: *arXiv preprint arXiv:1711.09846* (2017).

[Lia+18] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. "Tune: A Research Platform for Distributed Model Selection and Training". In: *arXiv preprint arXiv:1807.05118* (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.

[MB12] Akhilesh Kumar Maurya and Prashant Shridhar Bokare. "STUDY OF DECELERATION BEHAVIOUR OF DIFFERENT VEHICLE TYPES." In: *International Journal for Traffic & Transport Engineering* 2.3 (2012).

[Wu+17] C. Wu, A. Kreidieh, K. Parvate, E. Vinitsky, and A. Bayen. "Flow: Architecture and Benchmarking for Reinforcement Learning in Traffic Control". In: (2017). DOI: vol.abs/1710.05465,2017. URL: https://arxiv.org/abs/1710.05465.