Tactical Decision Making for Lane Changing with Deep Reinforcement Learning

Mustafa Mukadam
Georgia Institute of Technology, USA
mmukadam3@gatech.edu

Akansel Cosgun
Honda Research Institute, USA
acosgun@hra.com

Alireza Nakhaei
Honda Research Institute, USA
anakhaei@hra.com

Kikuo Fujimura
Honda Research Institute, USA
kfujimura@hra.com

Abstract

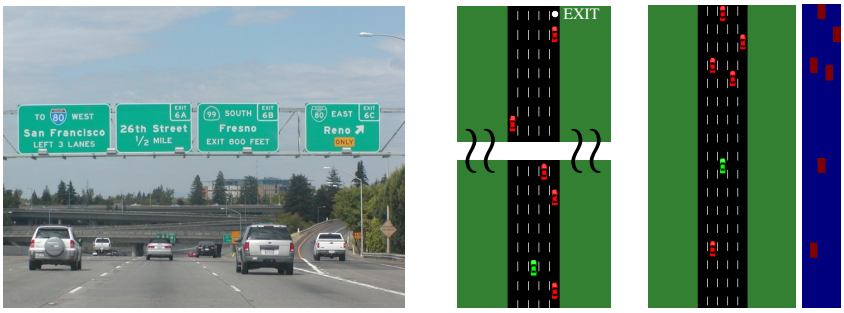
In this paper we consider the problem of autonomous lane changing for self driving cars in a multi-lane, multi-agent setting. We present a framework that demonstrates a more structured and data efficient alternative to end-to-end complete policy learning on problems where the high-level policy is hard to formulate using traditional optimization or rule based methods but well designed low-level controllers are available. The framework uses deep reinforcement learning solely to obtain a high-level policy for tactical decision making, while still maintaining a tight integration with the low-level controller, thus getting the best of both worlds. This is possible with Q-masking, a technique with which we are able to incorporate prior knowledge, constraints and information from a low-level controller, directly in to the learning process thereby simplifying the reward function and making learning faster and efficient. We provide preliminary results in a simulator and show our approach to be more efficient than a greedy baseline, and more successful and safer than human driving.

1 Introduction and Related Work

In recent years there has been a growing interest in self driving cars. Building such autonomous systems has been an active area of research [1, 2, 3] due to a high potential in leading to more efficient road networks that are much safer for the passengers. One of the fundamental skills a self driving car must possess is an ability to perform efficient lane change maneuvers in a safe manner. This is especially critical in a multi-lane highway setting in the presence fast moving traffic (as shown in Figure 1a), since if anything goes wrong, at best it leads to congestion and at worst to accidents [4]. Reasoning about interactions with other agents and forming an efficient long term strategy while maintaining safety makes this problem challenging and complex.

Prior work on lane changing consists of a diverse set of approaches with early work considering vision based control [5]. Other methods track trajectories [6, 7], use fuzzy control [8], model predictive control [9], generate a steering command with adaptive control [10], consider planning [1, 11], and mixed logic programming [12]. However majority of the prior work considers the problem only from a local perspective, i.e. changing between adjacent lanes while avoiding the few neighboring cars. There is no notion of a goal, like reaching an exit, which would require reasoning about long term decisions on a strategic level when present on a multi-lane highway among traffic. Formulating a control or optimization based problem to handle such a scenario is not straight forward, would require a lot of hand design and tuning, may work only on a subset of cases, and would generally be intractable. The primary roadblock is that there is no abstraction of what the overall ideal policy

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.



(a) (b) (c)

Figure 1: (a) Typical multi-lane highway (California, USA). (b) 5 lane highway in the simulator with traffic (red), where the ego car (green) is tasked with reaching the exit 1.5km away in the least amount of time. (c) Ego car's visibility in the simulator (left) and the corresponding occupancy grid (right) with the ego car at its center.

should look like, only the ideal outcome is know: reaching the exit safely and efficiently (in least amount of time).

Reinforcement learning provides a way to learn arbitrary policies giving specific goals. In recent years learning based methods have been used to address similar or related problems, like learning from human driving [13], inverse reinforcement learning [14], end-to-end methods that map perception inputs (mainly images) directly to control commands [15, 16, 17, 18], and methods that understand the scene via learning to make driving decisions [19, 20].

With the recent success of deep reinforcement learning [21], in this work we investigate its use and place in solving the autonomous lane changing problem. In general learning a full policy than can reason about tactical decisions while at same time address continuous control and collision avoidance can be exceedingly difficult with large notorious to train networks. An ideal approach to provide the right balance would be to learn the hard to define high-level tactical policy while relying on established optimization or rule based method for low-level control. Along these lines we propose a framework that tightly integrates the two paradigms using Q-masking that essentially forces the agent to explore only the required states and learns a subset of the space of Q-values. Within this framework we focus the raw deep learning power to only obtain a high-level decision making policy for tactical lane changing. By incorporating prior knowledge about the system, constraints of the problem and information from the low-level controller using Q-masking, we can heavily simplify the reward function and make the overall learning faster and efficient. Basing low-level decisions on a controller we are able to completely eliminate collisions during training or testing, which makes it possible to perform training directly on real systems. We present preliminary benchmarks and show our framework can outperform a greedy baseline in efficiency and humans driving in the simulator in safety and success.

2 Problem Setup

We consider the problem of autonomous lane changing in a multi-lane-multi-agent setting and set up the problem environment in the commonly used traffic simulator, SUMO [22].

Environment: The simulation environment consists of a L lane highway as shown in Figure 1b with minimum and maximum speed limits of v_{min} m/s and v_{max} m/s respectively that all cars must obey.

Traffic: The traffic density is generated using SUMO, where each lane can be assigned a probability P_{lane} of emitting a car at the start position every second, with a random start speed and a random target speed that it will stay near. These cars use a car follow model [23] and are controlled by SUMO to avoid collisions with each other. For the sake of simplicity we do not allow any traffic cars to change lanes.

Ego car: The ego car is tasked with reaching the exit at the right most lane, D km from the start position, in minimum amount of time (i.e. maximum average speed), while respecting the speed limits and avoiding collisions with traffic. Missing the exit would be considered a failure. Our aim is to learn a high-level tactical decision making strategy such that the ego car makes efficient lane change maneuvers while relying on the low-level controller for collision free lane changing between adjacent lanes. The ego car's performance is evaluated on success rate and average speed (i.e. time to reach the exit) and is compared to a greedy baseline and humans driving in the simulator. We set up the ego car in SUMO such that the simulator has no control over it, and the other cars do not avoid any collisions with the ego car. This allows our approach to fully govern the behavior of the ego car.

3 Deep RL for Tactical Decision Making

We train the ego car using deep reinforcement learning to learn a high-level policy that can make tactical decisions. For lane change maneuvers, we break down the high level decisions in to the following 5 actions that can be taken at any time step: (1) $\mathbf{N}$: no-op i.e. take no action and maintain current speed, (2) $\mathbf{A}$: accelerate for a constant amount this time step, (3) $\mathbf{D}$: decelerate for a constant amount this time step, (4) $\mathbf{L}$: make a left lane change, and (5) $\mathbf{R}$: make a right lane change. The network learns to map state inputs to Q-values [21] for these actions, as shown in Figure 2a.

The inputs to the network is the state of the ego car, which consists of internal and external information. Scalar inputs, velocity v, lane l, and distance to goal $d2g$, are chosen to represent internal information all of which are scaled between 0 and 1. Velocity can varied between 0 and 1 i.e. v_{min} and v_{max} respectively, lanes are numbered 0 to 1 from right to left, and distance to goal decreases from the start position of the car at 1 to the exit at 0. A binary occupancy grid around a chosen visibility region of the ego car (with the ego car in the center) is used to input external information. An example occupancy grid of size 42×5 is shown in Figure 1c where the visibility of the car is 50m in front and back with a longitudinal discretization of 2.5m per cell (all cars are 5m in length), and 2 lanes to the left and right with a one cell per lane discretization in the lateral direction. To capture relative motion of the traffic we also input a history of the occupancy grid from previous time steps, as separate channels.

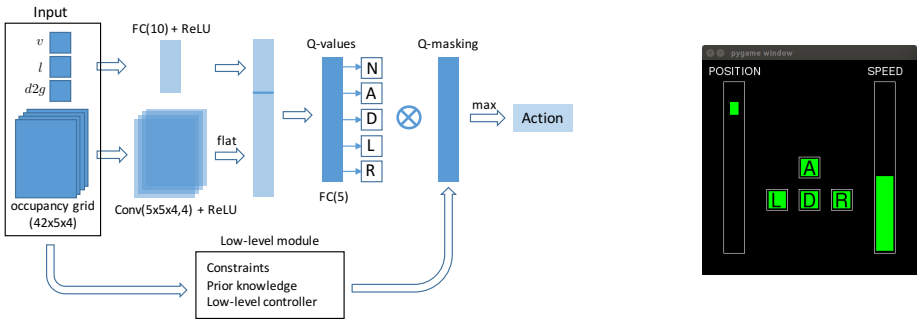
The network architecture we use is shown in Figure 2a. The occupancy grid with history is passed though a single convolution layer, flattened out and then concatenated with the output of a fully connected layer with the scalar inputs. The concatenation is passed through a fully connected layer to give the final output of 5 Q-values associated with the 5 tactical actions. A common practice is to use a max or soft-max operation on the Q-values to choose an action [21]. In this work we introduce a technique called Q-masking, which is injected between the Q-values and the max operation (see Section 4). Using this technique we are able to incorporate prior information about the problem that the agent does not need to learn from scratch through exploration. We can incorporate low-level optimization, control or rule based information, like generating trajectories to make an adjacent certain lane change happen while avoiding collisions. The combined effect is that learning becomes faster and more efficient while the reward function is massively simplified. We use the following sparse reward function,

$$r_T = \begin{cases} +10 & l = 0; \text{exit reached} \\ -10 \times l & l \neq 0; \text{exit missed} \end{cases} \tag{1}$$

where l is the lane in which the ego car is when it reaches the target distance D from the start position. A positive terminal reward is given for success (reaching the exit) and an increasingly negative terminal reward the further the ego car ends up away from the exit lane. Note the simplicity of the reward and the absence of competing components like maintaining speed limits or avoid collisions. A discount factor along with this reward encourages the agent to reach the exit in the smallest number of time steps i.e. maintaining a higher average speed.

During training, actions are taken in an ϵ-greedy manner and ϵ is annealed. To train the network we simulate full trajectories until the terminal state. Two experience buffers are maintained: good and bad. All transitions i.e. state, action and reward tuples from successful trajectories are saved in the good buffer while transition from failed trajectories are saved in the bad buffer. For any transition the expected reward is back calculated from the terminal reward,

$$y_t = \begin{cases} r_t & t = T; \text{terminal} \\ r_t + \gamma y_{t+1} & \text{otherwise} \end{cases} \tag{2}$$



(a) (b)

Figure 2: (a) Our framework consisting of the deep Q network and the low-level module that interface together using Q-masking. (b) A GUI displaying ego car's location and speed, and available actions.

where γ is the discount factor. While any trajectory is being simulated at each time step the network is optimized with the following loss function,

$$\mathcal{L}(\theta) = \big(y_t - Q(s_t, a_t, \theta)\big)^2 \tag{3}$$

using a mini batch of transitions equally sampled from the good and the bad buffer. The two separate buffers help maintain a decent exposure to successful executions when the exploration might constantly lead to failed trajectories thus avoiding the network getting stuck in a local minima.

4 Q-masking

We use deep Q-learning to learn a policy to make decisions on a tactical level. In a typical Q-learning network, a mapping between states and Q-values associated to each action is learned. Then a max (or soft max) operator can be applied on the output layer of Q-values to pick the best action. In this work we propose a technique, Q-masking, in the form of a mask that is applied on the output Q-values before taking the max operation as shown in Figure 2a. The direct effect of this is that when taking the max operation to choose the best action, we consider the Q-values associated with only a subset of actions, which are dictated by a lower-level module.

Given a state the lower-level module can restrict (or mask off) any set of actions that the agent does not need to explore or learn from their outcomes. For example, in the lane changing problem if the ego car is say in the left most lane, then taking a left action will result in getting off the highway. Therefore, it can put a mask on the Q-value associated with the left action such that it is never selected in such a state. This allows us to incorporate prior knowledge about the system (i.e. highway shoulders) directly in to the learning process, meaning that we do not need to set up a negative reward for getting off the highway, thus simplifying the reward function. Also, since the agent does not explore these states learning itself becomes faster and more efficient. What the agent ends up learning is a subset of the actual space of Q-values that are necessary. We can also incorporate constraints on the system in a similar manner that provides similar benefits. For example, if the ego car is driving at the maximum speed then the accelerate action is masked or if it is at the minimum speed then decelerate action is masked. Then the agent never needs to spend time learning the speed limits of the highway.

As discussed in Section 1 many optimization or rule based methods are available to generate a sequence of low-level actions that can make a car change between adjacent lanes while avoiding collisions. However, these methods are generally not designed to handle long term decision making and reasoning about lane change maneuvers in a multi-lane multi-agent setting. In turn modeling and training an end-to-end system to learn a complete policy that can generate collision free trajectories while reasoning about tactical level decisions is hard. We use Q-masking as a interface between the two ideologies and leverage deep learning to exclusively learn a high-level decision making policy and relying on the low-level module to provide control policies to change between adjacent lanes in a collision free manner. We can incorporate any optimization or rule based method in the low-level module such that given a state it masks off actions that can result in impossible to perform lane

Table 1: Benchmark results of our approach against a greedy baseline and human driving.

	Ours		**Baseline**	**Human**
	vis_{lat}=1	vis_{lat}=2		
Avg Speed (m/s)	26.50	26.27	22.34	29.16
Success (%)	84	91	100	70
Collision (%)	0	0	0	24

changes or in collisions. Then the learning process truly focuses on learning only the high level strategy. Since collisions are never allowed during training or testing the reward function does not need to account for it.

In our implementation we incorporate the highway shoulders information and the speed limits in the lower-level module. We also include a rule based time to collision (TTC) method [24] (we set the threshold as 10s) that checks for collisions given the state against all actions and masks off those actions that lead to collision.

5 Results

To evaluate the performance of our framework we compare the network against a greedy baseline policy and humans driving in the simulator. For this benchmark we set the problem parameters as follows: $L = 5$, $D = 1.5$km, $v_{min} = 20$m/s and $v_{max} = 30$m/s. The traffic density is set to $P_{lane} = \{0.3, 0.2, 0.2, 0.15, 0.1\}$ for lane 1 to 5 (from right to left) with $\{20, 22, 25, 27, 29\}$ m/s as the target speed in those lanes. These settings give faster sparse traffic in the left most lanes, and slower dense traffic in the right most lanes. Such traffic conditions ensured that non-trivial lane change maneuvers, like merging and overtaking would be necessary to remain efficient. For any method we restrict the action space to the tactical decisions described in Section 3 with a constant acceleration and deceleration rate of 2m/s^2. We aggregate the results across 100 trials for each method and record the success rate of reaching the exit, and the average speed of the ego car.

Ours: The network is trained for 10k episodes, with time step of 0.4s, discount factor $\gamma = 0.99$ and ϵ-greedy exploration, where ϵ is annealed from 1.0 to 0.1 for 80% of the episodes. For each episode the ego car is started from the zero position in a random lane with a random speed between the speed limits. To investigate the effects of visibility of the ego car we train and benchmark two networks with lateral visibility vis_{lat} of 1 and 2 lanes to the right and to the left while the longitudinal visibility is fixed at 50m in front and back. For the occupancy grid we use a 2.5m per cell resolution in the longitudinal direction and a history of 3 previous time steps to give the grid input sizes of $42\times3\times4$ and $42\times5\times4$ for the two networks.

Baseline: A greedy baseline tactical policy for the ego car prioritizes making a right lane change until it is in the correct lane. Then it tries to go as fast as possible while staying within the speed limits and not colliding with any car in the front. Same time step of 0.4s is set and to keep comparisons fair the high-level baseline policy is allowed to access the low-level module (see Section 4) as an oracle, to reason about speed limits, highway shoulders and collisions.

Human: We aggregated data on 10 human subjects that drove the ego car in the simulator for 10 trials each. As shown in Figure 2b a GUI was set up to indicate the location of the ego car on the highway relative to the start and exit, its speed and the available actions. The time step of the simulator was reduced to 0.1s for smoother control. The subjects were asked to drive naturally and were told that the primary and secondary objectives were to reach the exit and get there as fast as possible respectively. They were allowed to learn to use the simulator for a few trials before the data was recorded. Originally, we found that the subjects did not feel comfortable driving with the low-level module on (specifically the TTC component used for collision avoidance, see Section 4) and felt like they had to fight against the simulator or weren't being allowed to take actions that they considered were safe and possible. So we conducted the experiments with the TTC component of the low-level module turned off, however regaining what felt like relinquished control actually resulted in

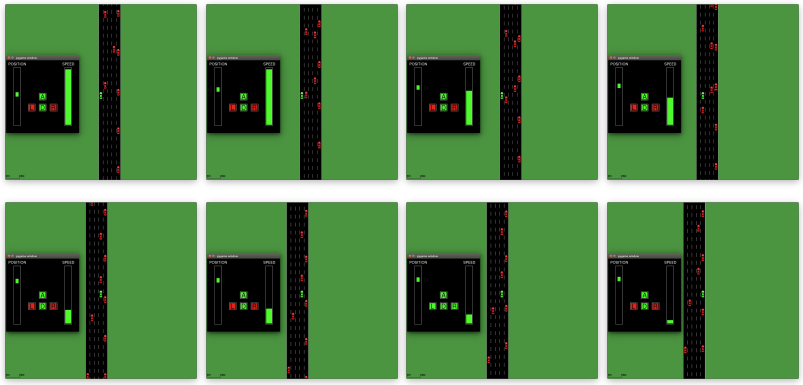


Figure 3: Examples of overtaking (top row) and merging (bottom row) behaviors, from left to right.

many collision as shown in Table 1. This warrants further study and is beyond the scope of this paper. We collected the success rate, average speed and also collision rate (since collision could happen).

The benchmark results are summarized in Table 1. By design the baseline policy is always successful in reaching the exit but is very inefficient since it never tries to apply any lane change maneuvers when stuck behind slow moving traffic. On the other hand the humans inclined to drive faster were overall much less successful, however majority of the failures were due to collisions and not missing the exit. Our approach is able to achieve a much higher average speed than the baseline, is more successful than the humans, and never results in collisions. An improvement in success rate is seen with increased visibility. The better performance is attained by the network having learned to make interesting and human-like lane change decision, which result in emergent behaviors like merging and overtaking (see Figure 3).

These preliminary results show the applicability of deep reinforcement learning in addressing tactical decision making problems. Our approach is able to strike the right synergy between learning a high-level policy and using a low-level controller. It hold promise for further investigation in improving performance with different (deeper) network architectures or applying it on other problem domains with a similar construction, and on real systems. Further improvements can be made to make the set up more realistic by considering occlusions and also introducing uncertainty with a probabilistic occupancy grid.

6 Conclusion

We proposed a framework that leverages the strengths of deep reinforcement learning for high-level tactical decision making, and traditional optimization or rule-based methods for low-level control, by striking the right balance between both domains. At the heart of this framework lies, Q-masking, that provides an interface between the two levels. Using Q-masking we can incorporate prior knowledge, constraints about the system and information from the lower-level controller, directly in to the training of the network, simplifying the reward function and making learning faster and more efficient, while completely eliminating collisions during training or testing. We applied our framework on the problem of autonomous lane changing for self driving cars, where the network learned a high-level tactical decision making policy. We presented preliminary results and benchmarked our approach against a greedy baseline and humans driving in the simulator and showed that our approach is able to outperform them both on different metrics with a more efficient and much safer policy.

References

[1] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[2] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha—a local, continuous method," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pp. 450–457, IEEE, 2014.

[3] A. Cosgun, L. Ma, J. Chiu, J. Huang, M. Demir, A. M. Anon, T. Lian, H. Tafish, and S. Al-Stouhi, "Towards full automated drive in urban environments: A demonstration in gomentum station, california," in *IEEE Intelligent Vehicles Symposium, IV 2017, Los Angeles, CA, USA, June 11-14, 2017*, pp. 1811–1818, 2017.

[4] H. Jula, E. B. Kosmatopoulos, and P. A. Ioannou, "Collision avoidance analysis for lane changing and merging," *IEEE Transactions on vehicular technology*, vol. 49, no. 6, pp. 2295–2308, 2000.

[5] C. J. Taylor, J. Košecká, R. Blasi, and J. Malik, "A comparative study of vision-based lateral control strategies for autonomous highway driving," *The International Journal of Robotics Research*, vol. 18, no. 5, pp. 442–453, 1999.

[6] J. E. Naranjo, C. Gonzalez, R. Garcia, and T. De Pedro, "Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 438–450, 2008.

[7] Z. Shiller and S. Sundar, "Emergency lane-change maneuvers of autonomous vehicles," *Journal of Dynamic Systems, Measurement, and Control*, vol. 120, no. 1, pp. 37–44, 1998.

[8] C. Hatipoglu, U. Ozguner, and K. A. Redmill, "Automated lane change controller design," *IEEE transactions on intelligent transportation systems*, vol. 4, no. 1, pp. 13–22, 2003.

[9] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on control systems technology*, vol. 15, no. 3, pp. 566–580, 2007.

[10] P. Petrov and F. Nashashibi, "Adaptive steering control for autonomous lane change maneuver," in *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pp. 835–840, IEEE, 2013.

[11] F. You, R. Zhang, G. Lie, H. Wang, H. Wen, and J. Xu, "Trajectory planning and tracking control for autonomous lane change maneuver based on the cooperative vehicle infrastructure system," *Expert Systems with Applications*, vol. 42, no. 14, pp. 5932–5946, 2015.

[12] Y. Du, Y. Wang, and C.-Y. Chan, "Autonomous lane-change controller via mixed logical dynamical," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pp. 1154–1159, IEEE, 2014.

[13] H. Tehrani, Q. H. Do, M. Egawa, K. Muto, K. Yoneda, and S. Mita, "General behavior and motion model for automated lane change," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pp. 1154–1159, IEEE, 2015.

[14] S. Sharifzadeh, I. Chiotellis, R. Triebel, and D. Cremers, "Learning to drive using inverse reinforcement learning and deep q-networks," *arXiv preprint arXiv:1612.03653*, 2016.

[15] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. L. Cun, "Off-road obstacle avoidance through end-to-end learning," in *Advances in neural information processing systems*, pp. 739–746, 2006.

[16] J. Koutník, G. Cuccu, J. Schmidhuber, and F. Gomez, "Evolving large-scale neural networks for vision-based torcs," 2013.

[17] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[18] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," *arXiv preprint arXiv:1612.01079*, 2016.

[19] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2722–2730, 2015.

[20] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint arXiv:1610.03295*, 2016.

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[22] 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*, vol. 5, no. 3&4, pp. 128–138, 2012.

[23] S. Krauß, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Physical Review E*, vol. 55, no. 5, p. 5597, 1997.

[24] R. van der Horst and J. Hogema, *Time-to-collision and collision avoidance systems*. na, 1993.