# TACTICAL DECISION MAKING FOR LANE CHANGING WITH DEEP REINFORCEMENT LEARNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

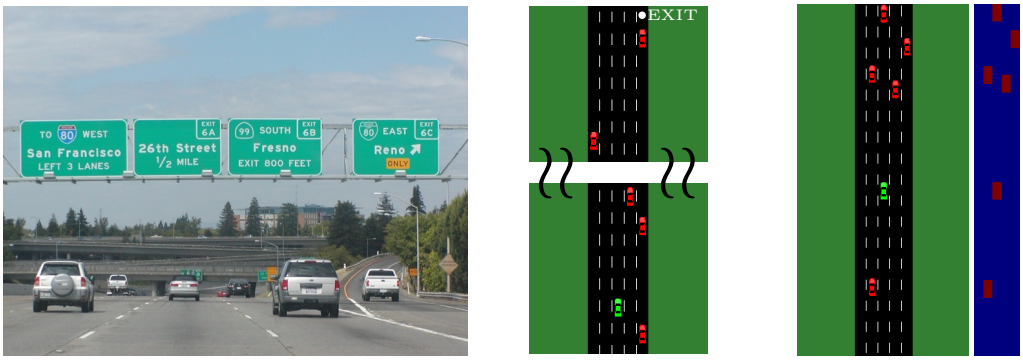
In this paper, we consider the problem of autonomous lane changing for self driving vehicles in a multi-lane, multi-agent setting. We present a framework that demonstrates a more structured and data efficient alternative to end-to-end complete policy learning on problems where the high-level policy is hard to formulate using traditional optimization or rule based methods but well designed low-level controllers are available. Our 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. We accomplish this 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 data 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 vehicles. Building such autonomous systems has been an active area of research (Urmson et al., 2008; Ziegler et al., 2014; Cosgun et al., 2017) for its high potential in leading to road networks that are much more safer and efficient. One of the fundamental skills a self driving vehicle must possess is an ability to perform lane change maneuvers, which is especially critical on a multi-lane highway in the presence fast moving traffic (as shown in Figure 1a). A bad decision at best leads to congestion and at worst leads to accidents (Jula et al., 2000). 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 (Taylor et al., 1999). Other methods track trajectories (Naranjo et al., 2008; Shiller & Sundar, 1998), use fuzzy control (Hatipoglu et al., 2003), model predictive control (Falcone et al., 2007), generate a steering command with adaptive control (Petrov & Nashashibi, 2013), consider planning (Urmson et al., 2008; You et al., 2015), and mixed logic programming (Du et al., 2014). 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 vehicles. 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 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 (Tehrani et al., 2015), inverse reinforcement learning (Sharifzadeh et al., 2016), end-to-end methods that map perception inputs (mainly images) directly to control commands (Muller et al., 2006; Koutník et al., 2013; Bojarski et al., 2016; Xu et al., 2016), and methods




(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.

that understand the scene via learning to make driving decisions (Chen et al., 2015; Shalev-Shwartz et al., 2016). Along these lines deep reinforcement learning has had great success in learning policies from raw sensor information (Mnih et al., 2015).

In this work, we investigate the use and place of deep reinforcement learning 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. Thus, an ideal approach would strike the right balance by learning the hard to specify high-level tactical policy while relying on established optimization or rule based method for low-level control.

We propose a framework that uses deep Q-learning to learn a high-level tactical decision making policy, and also introduce, Q-masking, a novel technique that forces the agent to explore and learn only a subspace of Q-values. This subspace is directly governed by a low-level module that consists of prior knowledge about the system, constraints of the problem, and information from the low-level controller. Not only does Q-masking provide the tight integration between the two paradigms: learning high-level policy and using low-level control, but also heavily simplifies the reward function and makes learning faster and data efficient. By relying on a controller for low-level decisions, we are also able to completely eliminate collisions during training or testing, which makes it a possibility to perform training directly on real systems. We present preliminary benchmarks and show that our framework can outperform a greedy baseline in terms of efficiency and humans driving in the simulator in terms of safety and success, while also generalizing to several unseen scenarios without any extra training.

## 2 Problem Setup

In this section, we formally define the specifications of the autonomous lane changing problem. We consider a multi-lane-multi-agent setting with the highway environment setup in the commonly used traffic simulator, SUMO (Krajzewicz et al., 2012).

**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 (Krauß et al., 1997) 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:** 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 will allow our approach to fully govern the behavior of the ego car.

**Problem:** 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.

In the next section, we describe our approach to learn a high-level tactical decision making policy such that the ego car can make efficient lane change maneuvers while relying on the low-level controller for collision free lane changing between adjacent lanes. This is accomplished in part, by using Q-masking presented in Section 4. 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 in Section 5.

## 3 DEEP RL FOR TACTICAL DECISION MAKING

We are interested in a policy for the ego car that can reason about long term lane change maneuvers with respect reaching a goal in minimum time, while also considering local conditions in accomplishing adjacent lane changes in a collision free manner. As discussed in Section 1 a large body of work exists to achieve the latter objective, which cannot be easily extended to consider the former objective. Since an exact ideal policy is hard to define, in this work, we investigate the applicability of deep reinforcement learning, specifically deep Q-learning (Mnih et al., 2015), to solve this problem. However, we present a framework that uses deep reinforcement learning solely for high-level decisions such that the learning problem is more manageable while the complete objective can be achieved by integrating a separate low-level module for low-level actions.

### 3.1 ACTIONS

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.

### 3.2 STATE

The state of the ego car 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. This scaling yielded better performance and also proved more robust to changes in the environmental settings, like number of lanes, starting position or length of highway (distance to the exit). With invariance to problem settings we are able to apply zero-shot transfer to problems not trained on before (see Section 5) making our approach more generally applicable. 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 $42\times5$ 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.

### 3.3 DEEP Q-LEARNING

The ego car takes actions to learn from their outcomes with the help of rewards by estimating the optimal Q-value function. Full trajectories are simulated (like monte carlo samples) until the ter-

**Algorithm 1** Deep Q-learning with Q-masking

Initialize experience buffers $\mathcal{G}$, $\mathcal{B}$
Initialize network with random weights $\theta$
**for** episode=1 **to** $N$ **do**
  Initialize state $s_t$
  **for** $t = 0$ **to** $T_{max}$ **do**
    $a_t = \begin{cases} \text{random}\big[\text{actions} \bigotimes Q_{mask}(s_t)\big] & \text{with probability } \epsilon \\ \max_a \big[Q(s_t, a; \theta) \bigotimes Q_{mask}(s_t, a)\big] & \text{otherwise} \end{cases}$
    Execute $a_t$ to get $s_{t+1}$
    save $s_t, a_t$
    **if** terminal($s_{t+1}$) **then**
      **for** $t = t$ **to** $t = 0$ **do**
        $y_t = \begin{cases} r_T & \text{terminal} \\ r_t + \gamma y_{t+1} & \text{otherwise} \end{cases}$
        add $(s_t, a_t, y_t)$ to $\mathcal{G}$ if succesful or $\mathcal{B}$ if failure
      **end for**
    **end if**
    Sample minibatch of M from $\mathcal{G}$ and $\mathcal{B}$
    Perform gradient step on $\big(y_t - Q(s_t, a_t; \theta)\big)^2$ with minibatch
  **end for**
**end for**

minal state (or some maximum alloted time), since the time-length of the trajectories is relatively small. We use a neural network as a non-linear function approximator that learns to map state inputs to Q-values for the high-level actions. During training the network weights, $\theta$, are adjusted based on minimizing the following loss function,

$$\mathcal{L}(\theta) = \mathbb{E}_{s_t, a_t}\Big[\big(y_t - Q(s_t, a_t; \theta)\big)^2\Big]. \tag{1}$$

Typically, in one-step Q-learning the target $y_t$ is calculated by taking a maximum over the Q-values for the next state action pair, $y_t = r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta)$, but since we simulate the full trajectory we can back calculate the target exactly from the terminal reward $r_T$,

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

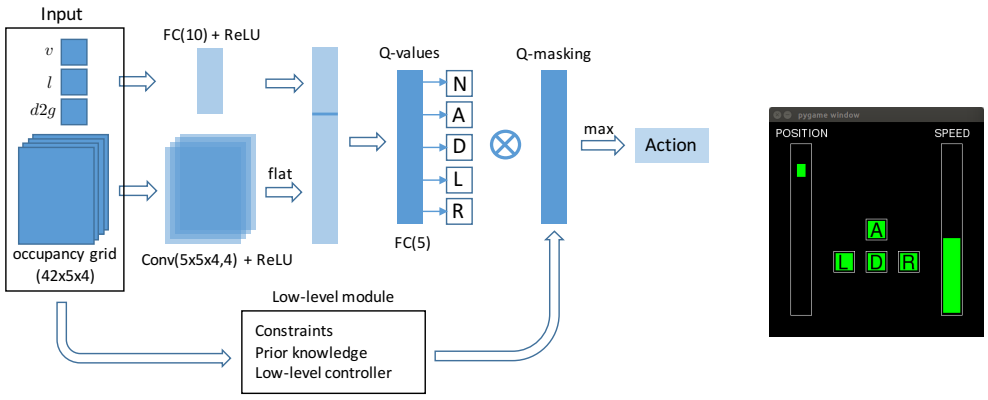
where $\gamma$ is the discount factor. For a full treatment of deep Q-learning see (Mnih et al., 2015). We also maintain two experience buffers: good $\mathcal{G}$ and bad $\mathcal{B}$. All transitions i.e. state, action and target tuples from successful trajectories are saved in the good buffer while transitions from failed trajectories are saved in the bad buffer. While any trajectory is being simulated at each time step the network is optimized using a mini batch of transitions equally sampled from the good and the bad buffer. The entire process in summarized in Algorithm 1. 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.

### 3.4 Reward

We use the following sparse reward function,

$$r_T = \begin{cases} +10 & l = 0; \text{exit reached} \\ -10 \times l & l \neq 0; \text{exit missed} \end{cases} \tag{3}$$

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. The discount factor, $\gamma$, encourages the ego car to reach the exit in the smallest number of time steps i.e. maintaining a higher average speed.




(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.

### 3.5 Neural Network

The network architecture we use is shown in Figure 2a. The occupancy grid with history is passed through a single convolution layer, flattened out and then concatenated with the output of a fully connected layer from the scalar inputs. The concatenation is then 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 (Mnih et al., 2015). 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 also incorporate low-level optimization, control or rule based information, for example, generating trajectories to make a certain adjacent lane change happen while avoiding collisions. The combined effect is that learning becomes faster and more efficient while the reward function is massively simplified. Note the simplicity of the reward function we use above and the absence of competing components like maintaining speed limits or avoiding collisions.

## 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 and in Algorithm 1. 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, which means 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 data efficient. What the agent ends up learning is a subspace of Q-values; only the part that is 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

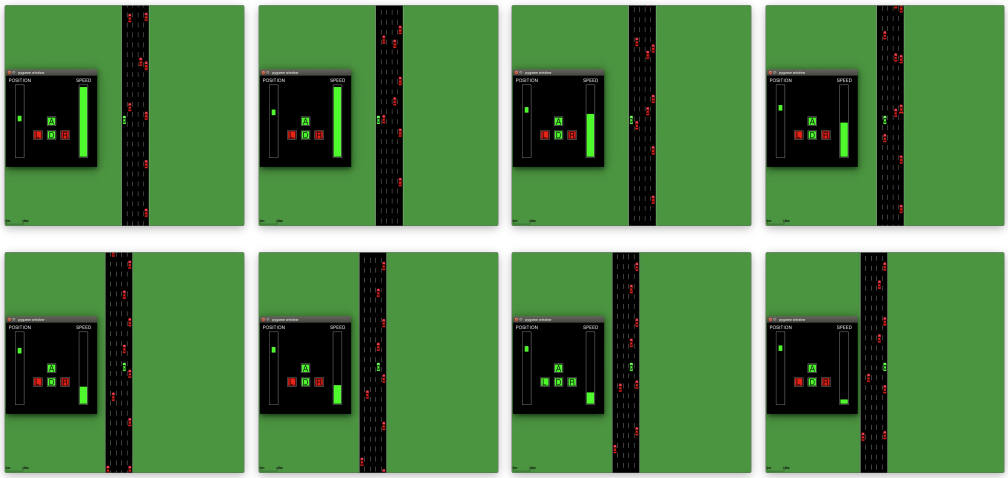


Figure 3: Examples of overtaking (top row) and merging (bottom row) behaviors, from left to right.

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.

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 rely 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 high-level actions that can result in impossible to perform lane 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.

Such a strategy can also be used for imitation learning (Ross et al., 2011; He et al., 2012) where the low-level module acts as an oracle or an expert that suggests desired actions. There are also potential relations to reward shaping (Brys et al., 2014) and options (Sutton et al., 1999). In reward shaping, the extra knowledge is passed in as a new reward signal from which the agents learns and adapts to take future actions. In contrast, here we directly pass that information to the place where an action is chosen, thus keeping the reward function simple and sparse. Options (Sutton et al., 1999), have also been used in the past to build high-level skills.

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 (van der Horst & Hogema, 1993), with a threshold of 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

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 |

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 $\epsilon$-greedy exploration, where $\epsilon$ 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 42×3×4 and 42×5×4 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 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).

**Zero-shot transfer:** By designing our inputs in a scaled manner as described in Section 3 we are able to make the training invariant to some of the problem settings. This allowed us to test our trained network on unseen problems without any exposure or training to the new problem. We use the best trained network (with $vis_{lat} = 2$) and test on the following new scenarios: (1) instead of always starting at the 0m longitudinal position, the ego car is started in a random position between 0-750m, (2) the exit is moved from 1.5km to 2km, (3) the number of lanes on the highway is decreased to 3

Table 2: Zero-shot transfer of our learned network across four unseen scenarios.

| | 0−750 m start | | 2 km exit | | 3 lanes | | 7 lanes | |
|---|---|---|---|---|---|---|---|---|
| | **Ours** | **Baseline** | **Ours** | **Baseline** | **Ours** | **Baseline** | **Ours** | **Baseline** |
| Avg Speed (m/s) | 24.57 | 22.27 | 25.53 | 21.99 | 28.12 | 22.67 | 24.90 | 22.10 |
| Success (%) | 88 | 100 | 86 | 100 | 82 | 100 | 87 | 100 |

lanes, and (4) the number of lanes on the highway is increased to 7 lanes. We compare against the same baseline on these scenarios. Table 2 show aggregate results of 100 trials for each method or scenarios. We see the baseline policy performance remains more or less the same and our approach is able to generalize well to the new scenarios with a small decrease in the success rate and average speed. The lower average speed in scenario 1 can be attributed to the reduced amount of highway the ego has left to reach the exit while the higher average speed in scenario 3 could be a result of decreased problem difficulty.

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. Specific to the lane changing problem, future work can be set up to be 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 are able to 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 data 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 neural 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. Finally, we demonstrated zero shot generalizations on several unseen scenarios.

## References

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

Tim Brys, Anna Harutyunyan, Peter Vrancx, Matthew E Taylor, Daniel Kudenko, and Ann Nowé. Multi-objectivization of reinforcement learning problems by reward shaping. In *Neural Networks (IJCNN), 2014 International Joint Conference on*, pp. 2315–2322. IEEE, 2014.

Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2722–2730, 2015.

Akansel Cosgun, Lichao Ma, Jimmy Chiu, Jiawei Huang, Mahmut Demir, Alexandre Miranda Anon, Thang Lian, Hasan Tafish, and Samir 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.

Yaoqiong Du, Yizhou Wang, and Ching-Yao 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.

Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on control systems technology*, 15(3):566–580, 2007.

Cem Hatipoglu, Umit Ozguner, and Keith A Redmill. Automated lane change controller design. *IEEE transactions on intelligent transportation systems*, 4(1):13–22, 2003.

He He, Jason Eisner, and Hal Daume. Imitation learning by coaching. In *Advances in Neural Information Processing Systems*, pp. 3149–3157, 2012.

Hossein Jula, Elias B Kosmatopoulos, and Petros A Ioannou. Collision avoidance analysis for lane changing and merging. *IEEE Transactions on vehicular technology*, 49(6):2295–2308, 2000.

Jan Koutník, Giuseppe Cuccu, Jürgen Schmidhuber, and Faustino Gomez. Evolving large-scale neural networks for vision-based torcs. 2013.

Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. Recent development and applications of sumo-simulation of urban mobility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138, 2012.

Stefan Krauß, Peter Wagner, and Christian Gawron. Metastable states in a microscopic model of traffic flow. *Physical Review E*, 55(5):5597, 1997.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Urs Muller, Jan Ben, Eric Cosatto, Beat Flepp, and Yann L Cun. Off-road obstacle avoidance through end-to-end learning. In *Advances in neural information processing systems*, pp. 739–746, 2006.

Jose E Naranjo, Carlos Gonzalez, Ricardo Garcia, and Teresa De Pedro. Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver. *IEEE Transactions on Intelligent Transportation Systems*, 9(3):438–450, 2008.

Plamen Petrov and Fawzi Nashashibi. Adaptive steering control for autonomous lane change maneuver. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pp. 835–840. IEEE, 2013.

Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 627–635, 2011.

Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

Sahand Sharifzadeh, Ioannis Chiotellis, Rudolph Triebel, and Daniel Cremers. Learning to drive using inverse reinforcement learning and deep q-networks. *arXiv preprint arXiv:1612.03653*, 2016.

Zvi Shiller and Satish Sundar. Emergency lane-change maneuvers of autonomous vehicles. *Journal of Dynamic Systems, Measurement, and Control*, 120(1):37–44, 1998.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Camillo J Taylor, Jana Košecká, Robert Blasi, and Jitendra Malik. A comparative study of vision-based lateral control strategies for autonomous highway driving. *The International Journal of Robotics Research*, 18(5):442–453, 1999.

Hossein Tehrani, Quoc Huy Do, Masumi Egawa, Kenji Muto, Keisuke Yoneda, and Seiichi Mita. General behavior and motion model for automated lane change. In *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pp. 1154–1159. IEEE, 2015.

Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

Richard van der Horst and Jeroen Hogema. *Time-to-collision and collision avoidance systems*. na, 1993.

Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. *arXiv preprint arXiv:1612.01079*, 2016.

Feng You, Ronghui Zhang, Guo Lie, Haiwei Wang, Huiyin Wen, and Jianmin Xu. Trajectory planning and tracking control for autonomous lane change maneuver based on the cooperative vehicle infrastructure system. *Expert Systems with Applications*, 42(14):5932–5946, 2015.

Julius Ziegler, Philipp Bender, Thao Dang, and Christoph Stiller. Trajectory planning for bertha—a local, continuous method. In *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*, pp. 450–457. IEEE, 2014.