Driving in Dense Traffic with Model-Free Reinforcement Learning

Dhruv Mauria Saxena[1], Sangjae Bae[2], Alireza Nakhaei[3], Kikuo Fujimura[3], Maxim Likhachev[1]

Abstract— Traditional planning and control methods could fail to find a feasible trajectory for an autonomous vehicle to execute amongst dense traffic on roads. This is because the obstacle-free volume in spacetime is very small in these scenarios for the vehicle to drive through. However, that does not mean the task is infeasible since human drivers are known to be able to drive amongst dense traffic by leveraging the cooperativeness of other drivers to open a gap. The traditional methods fail to take into account the fact that the actions taken by an agent affect the behaviour of other vehicles on the road. In this work, we rely on the ability of deep reinforcement learning to implicitly model such interactions and learn a continuous control policy over the action space of an autonomous vehicle. The application we consider requires our agent to negotiate and open a gap in the road in order to successfully merge or change lanes. Our policy learns to repeatedly probe into the target road lane while trying to find a safe spot to move in to. We compare against two model-predictive control-based algorithms and show that our policy outperforms them in simulation. As part of this work, we introduce a benchmark for driving in dense traffic for use by the community.

I. INTRODUCTION

Since the 2007 DARPA Urban Challenge [5], autonomous vehicles have transitioned from being tested in well-structured environments under complete supervision and control, to being tested on actual roads in the real-world amongst human drivers. This progress comes together with several challenges that must be addressed if autonomous vehicles are to share the road with human driven vehicles one day. One of these is the fact that driving on roads is inherently an interactive exercise, i.e. actions taken by an autonomous vehicle affect other nearby vehicles on the road and vice versa [35]. This is especially apparent in dense traffic where any goal-directed behaviour must rely on some level of cooperation between various agents on the road in order to achieve the desired goal. For example, consider the motivating example of this work from Fig. 1. The goal for the red *ego-vehicle*[1] is to change into the left lane before the intersection so that it can make a legal left turn. However, the dense traffic on the road makes it necessary for the ego-vehicle to convince a vehicle in that lane to give it room in order to successfully change lanes. In the remainder of this paper, we refer to the finite distance available to the ego-vehicle as the distance to a *deadend*.

[1]We refer to the vehicle or agent of interest as the ego-vehicle. In the remainder of this paper, this will be the vehicle we control on the road.
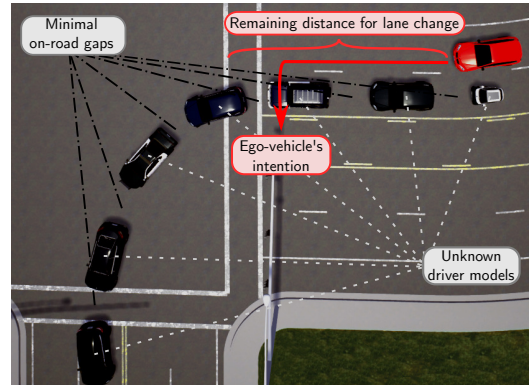

Fig. 1: **Motivating Example:** The ego-vehicle (in red) wants to change lanes in order to make a legal left turn. Road rules restrict the distance available for this lane change. There is also dense traffic on the road. The driver models of *all* other vehicles are unknown to the ego-vehicle (in terms of their cooperativeness at the very least).

Traditional planning and control techniques might fail to find reasonable solutions in these scenarios due to the minimal volume of obstacle-free space [19, 22, 25]. As a result, the only feasible, collision-free solution for the autonomous vehicle might be to stay stationary which incurs other costs (such as increased time to achieve the goal and decreased average speed) [32]. It is possible to augment these methods to take predictions about the trajectories of other vehicles into account [1, 29], but incorporating complex interactions between multiple vehicles as part of these techniques is still an open research question. In this work, we present a model-free reinforcement learning based solution that does not need any explicit model of inter-vehicle interactions, yet leverages these interactions to learn behaviours that successfully accomplish the lane change task from Fig. 1. We state our problem of interest succinctly below.

Problem. *Successfully execute a safe and comfortable merge into dense traffic.*

In order to avoid the "freezing robot problem" in dense traffic, even perfect predictions of future trajectories of other agents is not enough. We must rely on implicit or explicit models of *joint* interactions between the agents in the scene [32]. For autonomous driving, since we cannot reliably know the behaviour of other drivers, we choose to implicitly learn this joint interaction behaviour by utilising the ability of deep reinforcement learning to learn complex policies from data [11].

Our main contributions in this work are:

- a challenging benchmark for driving in dense traffic

(Section III-A).
- a continuous state and action space, policy gradient-based deep reinforcement learning solution for the benchmark (Section III-C).
- a design specification for agent observations and the reward function for this task (Sections III-D and III-E).

II. RELATED WORK

In recent years, there has been a considerable focus on using machine learning techniques for autonomous driving. These have been used for policies learned via behaviour cloning and reinforcement learning. A recent survey by Schwarting, Alonso-Mora, and Rus [30] is a great resource for motion planning for autonomous driving in genral and covers literature on both learning-based methods and traditional methods (search-based [1], sampling-based [24], or optimisation-based [37] planning techniques). We focus on the former category below as it is most relevant to our work.

A. Behaviour Cloning

There has been work on using neural networks to control steering commands of autonomous vehicles since the late 1980s [26]. Recent developments in data processing and computational resources have caused a shift towards deep learning-based methods [4]. Both approaches rely on behaviour cloning to learn a mapping from raw images directly to steering angle commands using a dataset of human-driven trajectories. Behaviour cloning has been used to learn a mapping from input observations (raw sensor data or processed information) to control policies in several ways [2, 6, 9, 13, 21]. These approaches differ in terms of the input observation, action space, policy parameterisation, or additional outputs (like uncertainty estimates for the chosen action), but they all still rely on labeled datasets. We also train a policy end-to-end in our work, but reinforcement learning has no supervisory label as in these approaches.

B. Reinforcement Learning

Given its early success on other continuous control tasks [23], reinforcement learning has received considerable attention for autonomous driving [7, 14, 31, 36]. The complexity of the task being solved varies greatly across the literature, along with the reinforcement learning algorithms used and parameterisation of the control policy. Q-Learning has been used to control the steering angle for a single vehicle on a road without traffic [36]. In [14], reinforcement learning is used to learn a value function which in turn is used to guide a Monte Carlo Tree Search for action selection. In contrast to these approaches, we use reinforcement learning to learn a low-level, continuous-control policy for driving amongst dense traffic. The continuous control aspect has been addressed previously [7], however the scenario they consider does not necessitate complex interactions with other vehicles in order to successfully complete the task. The *double-merge* task from [31] does require vehicles to interact with others in order to achieve its goal, but they focus on learning higher-level tactical decisions as opposed to low-level controls. More recently, [16] used reinforcement learning to solve problems similar to the one we consider. However their action space was much simpler than ours (five discretised values of acceleration), there was very little traffic on the road, and the road geometry only accommodated a fixed merge *point* as opposed to the more realistic case of a finite distance for the task (lane change or merge) as in our work.

Explicitly modeling human interactions could help elicit better behaviours from our learned policy in terms of safe driving on the road, passenger comfort and interpretability of agent behaviours. There is promising work in this direction [10, 27], however we choose to keep our work model-free since in our scenario, we would need to capture interactions with several vehicles at every timestep, which could be computationally intractable even if there was a reasonable model.

III. APPROACH

Continuous control reinforcement learning algorithms use policy gradient optimisation to directly learn a policy over the state space, which can be easier than first learning a value function and using it to derive a policy. Briefly, these algorithms maximise the following objective via gradient ascent,

$$\nabla_\theta J(\theta) =$$
$$\mathbb{E}_{\tau \sim \pi_\theta(\tau)}\left[\left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)\right)\left(\sum_{t=1}^{T} r(\boldsymbol{a}_t, \boldsymbol{s}_t)\right)\right],$$

where τ is a trajectory, $\pi_\theta(\tau)$ is the likelihood of executing that trajectory under the current policy π_θ, $\pi_\theta(\boldsymbol{a}_t|\boldsymbol{s}_t)$ is the probability of executing action $\boldsymbol{a}_t$ from state $\boldsymbol{s}_t$, and $r(\boldsymbol{a}_t, \boldsymbol{s}_t)$ is the reward gained for that execution.

A. Benchmark for Driving in Dense Traffic

The simulation scenario we consider is implemented using an open-source simulator[2]. In order to obtain diverse on-road behaviours from other vehicles, we make a few modifications to well-known rule-based models - Intelligent Driver Model (IDM) for lane following [33], and MOBIL for lane changing [34]. IDM is modified to include a *stop-and-go* behaviour that cycles between a non-zero and zero desired velocity in regular time intervals. This behaviour is intended to simulate real-world driving behaviours seen in heavy-traffic during rush-hour. MOBIL is allowed to randomly change lanes (if safe) with probability $p = 0.04$. Our benchmark scenario has two key components - dense traffic and a deadend in front of the ego-vehicle. The most important parameters that control an instantiation of this benchmark are the number of vehicles, gaps between them, their desired velocities and the ego-vehicle's distance to the deadend. A detailed list of parameters is given in Table I.

The cooperativeness and perception range parameters p_c and λ_p respectively control whether a vehicle slows down to

[2]https://github.com/sisl/AutomotiveDrivingModels.jl/

TABLE I: Benchmark scenario parameters

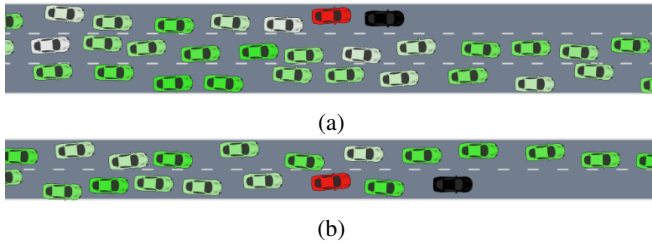| Parameter	Description	Value
$N \in \mathbb{Z}$	Number of vehicles	$[1, 100]$
$v^{\text{des}} \in \mathbb{R}$	Desired velocity (m/s)	$[2, 5]$
$s_0 \in \mathbb{R}$	Initial gap to vehicle in front (m)	$[0.5, 3]$
$s_D \in \mathbb{R}$	Deadend distance from ego-vehicle (m)	$[5, 40]$
$p_c \in \mathbb{R}$	Cooperativeness	$[0, 1]$
$\lambda_p \in \mathbb{R}$	Perception range (m)	$[-0.15, 0.15]$
Δt	Simulation timestep (s)	0.2
L	Number of lanes on road	$\{2, 3\}$
l	Vehicle length (m)	4.0
w	Vehicle width (m)	1.8

(a)

(b)

Fig. 2: Randomly generated initial states for the benchmark scenario. The ego-vehicle is in red. Other vehicles are more green if they are more cooperative. We represent the deadend with a black car on the road. (a) Three lane road example. (b) Two lane road example.

cooperate with another vehicle. Each vehicle on the road can perceive vehicles in its lateral field-of-view, which includes the width of its lane plus an extra width represented by λ_p. For any other vehicle that is inside this field-of-view, the vehicle decides whether to slow down, i.e. cooperate, with probability p_c at every timestep Δt^3. In order to elicit complex behaviours from other vehicles on the road that reflect those seen on real roads, we needed to account for different levels of cooperativeness (λ_c), and also the fact that these behaviours vary over time (p_c).

Example initialisations of this benchmark scenario can be seen in Fig. 2. We colour more cooperative vehicles as more green in the simulation, and less cooperative vehicles as more white. Evaluation code for this scenario is available at `https://github.com/honda-research-institute/DenseTrafficEval`.

B. Vehicle Model

All vehicles in our simulation follow a kinematic bicycle model [20]. The nonlinear equations of motion for this model are rewritten here,

$$\dot{x} = v \cos(\psi + \beta)$$
$$\dot{y} = v \sin(\psi + \beta)$$
$$\dot{\psi} = \frac{v}{l_r} \sin(\beta)$$
$$\dot{v} = a$$
$$\beta = \arctan\left(\frac{l_r}{l_f + l_r} \tan(\delta_f)\right).$$

---

^{3}We effectively use $p_c = 1$ for vehicles within w of the lateral field-of-view, i.e. enforce full cooperation with vehicles directly in front.
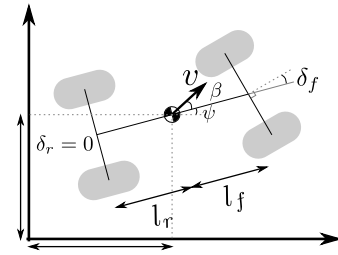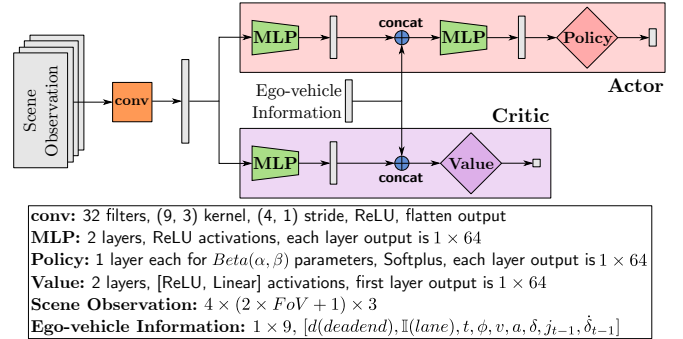

Fig. 3: Kinematic bicycle model.

conv: 32 filters, (9, 3) kernel, (4, 1) stride, ReLU, flatten output
MLP: 2 layers, ReLU activations, each layer output is 1×64
Policy: 1 layer each for $Beta(\alpha, \beta)$ parameters, Softplus, each layer output is 1×64
Value: 2 layers, [ReLU, Linear] activations, first layer output is 1×64
Scene Observation: $4 \times (2 \times FoV + 1) \times 3$
Ego-vehicle Information: 1×9, $[d(deadend), \mathbb{I}(lane), t, \phi, v, a, \delta, j_{t-1}, \dot{\delta}_{t-1}]$

Fig. 4: Network architecture.

Relative to a global inertial frame, (x, y) are the spatial coordinates of a vehicle, ψ is the heading, and v is the velocity vector. In the local frame of the vehicle, β is the angle of the velocity vector, δ_f is the angle of the front tires, and a is the acceleration. l_r and l_f are the distances of the rear and front tires respectively from the center of the vehicle. The *steering angle δ_f* and *acceleration a* are the control inputs for the system. For simplicity, we assume the steering angle of the rear tires $\delta_r = 0$. A diagram of the kinematic bicycle model for four-wheel vehicles can be seen in Fig. 3.

C. Network Architecture and Policy Parameterisation

We use an actor-critic style network that is trained using Proximal Policy Optimisation (PPO) [28]. We found that training was more stable without sharing parameters between the actor and critic networks. Our complete network has around $120,000$ parameters. Fig. 4 shows a detailed architecture of our network. The task of autonomous driving is inherently one of continuous control since we usually control the acceleration and steering angle of the vehicle. To get smooth behaviours with high enough fidelity via discrete control would greatly increase the size of the action space, thereby making discrete control methods intractable. On the contrary, continuous control reinforcement learning has showing promising results in complex problems [23] and even on real-world robots [17, 18].

For autonomous driving, we also need to keep the comfort of the passengers in mind. Learning a policy over the acceleration and steering angle of a vehicle might lead to jerky or oscillatory behaviour which is undesirable. Instead, we train our network to predict the time derivatives of these quantities, i.e. *jerk j* and *steering rate $\dot{\delta}$*. This helps us maintain a smooth signal over the true low-level control variables.
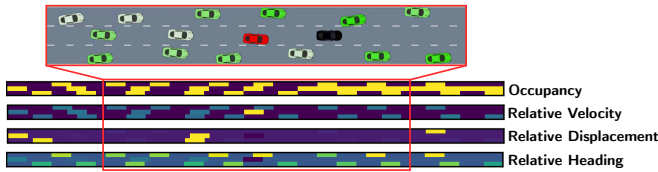
Fig. 5: The simulator state (*top*, zoomed in) gets converted to a $4 \times 3 \times (2 \times FoV + 1)$ input observation tensor (*bottom*).

Following the analysis in [8], we parameterise our policy as Beta distributions for j and $\dot{\delta}$. This makes training more stable as the policy gradients are unbiased with respect to the finite support of the Beta distribution. We scale each action to acceptable dynamic limits for j and $\dot{\delta}$ inside the simulator. For j, we allow values in the range $[-4.0, 2.0]$ m/s^3, whereas $\dot{\delta}$ can vary between $[-0.4, 0.4]$ $\frac{rad}{s}$.

D. Ego-vehicle Observations

Due to the large number of vehicles that could be considered *neighbours* of the ego-vehicle at any time, and the fact that this number would almost certainly change over time, our input representation is agnostic to this number. Additionally, in order to capture the complex inter-vehicle interactions on the road, the input observations include information about the dynamic states of neighbouring vehicles. We use an occupancy-grid style observation that is controlled by one parameter - the longitudinal field-of-view (FoV) of the ego-vehicle[4]. We assume that in the real-world, on-board perception pipelines would process raw sensor data to determine the relative poses and velocities of neighbouring vehicles. We focus on learning a control policy from this processed information instead of raw sensor data, to alleviate the burden of processing the raw data a second time as part of the controller learning pipeline.

In our simulations, at each timestep, we process the simulator state to calculate an observation tensor of size $4 \times 3 \times (2 \times FoV + 1)$. There is one channel each (first dimension) for on-road occupancy, relative velocities of vehicles, relative lateral displacements, and relative headings with respect to the ego-vehicle. The rows (second dimension) represent the lanes on the road (left, current and right lanes for the ego-vehicle). Fig. 5 shows an example of the simulator state and corresponding input observation used for our network.

We also include an ego-vehicle feature vector as part of our observation. This includes the distance to the deadend ($d(deadend)$), an indicator for whether we are in the target lane ($\mathbb{I}\{lane\}$), lateral displacement and relative heading from the centerline of the target lane (t and ϕ), current velocity, acceleration, and steering angle (v, a, and δ), and the action executed at the last timestep (j_{t-1} and $\dot{\delta}_{t-1}$).

E. Reward Function

At a high-level, our reward function contains three sets of terms for three different purposes,

R1 We would like the ego-vehicle to be centered vis-à-vis the target lane, and achieve a desired speed.

R2 We want to avoid jerky and oscillatory driving behaviour to maximise passenger comfort.

R3 Due to an upcoming deadend, we want to finish the lane change maneuver sooner rather than later.

We formulate our reward function by taking these design choices into consideration. Our reward at timestep t for the $(\boldsymbol{a}_t, \boldsymbol{s}_t)$ pair is,

$$
\begin{aligned}
r(\boldsymbol{a}_t, \boldsymbol{s}_t) = 0 & - \lambda_v \cdot |v - v_{\text{des}}| \\
& - \lambda_t \cdot |t| \\
& - \lambda_\phi \cdot |\phi| \cdot \mathbb{I}\{lane\} \\
& - \lambda_j \cdot j \\
& - \lambda_{\dot{\delta}} \cdot \dot{\delta} \\
& + 1 \cdot \mathbb{I}\{lane\} \\
& + f(deadend)
\end{aligned}
\qquad
\begin{aligned}
& \left.\vphantom{\begin{aligned} & \\ & \\ & \end{aligned}}\right\} \textbf{R1} \\
& \left.\vphantom{\begin{aligned} & \\ & \end{aligned}}\right\} \textbf{R2} \\
& \left.\vphantom{\begin{aligned} & \\ & \end{aligned}}\right\} \textbf{R3}
\end{aligned}
$$

v_{des} is the desired velocity for the ego-vehicle, and $f(deadend)$ rewards or penalises the agent according to ego-vehicle's lane and distance to deadend.

IV. EXPERIMENTAL RESULTS

A. Baselines

We compare the performance of our approach against the following two baselines:

1) *MPC baseline [15]:* this baseline optimises for the distance of the final state of the trajectory from a target state (in L_2-norm distance). All trajectories are 6s long and the target state is always in the desired lane. We vary three parameters, s, c_f, and c_m to get the different variants denoted as MPC(s, c_f, c_m). If the collision check succeeds, we take the first step along the trajectory. Else, we apply full brakes.
   - $s \in \{l, 2l, 3l, 4l, 5l\}$ is the longitudinal distance of the target state along the lane[5].
   - $c_f \in \{0, 0.1, 0.25, 0.5, 1.0\}$ represents the fraction of the trajectory we check for collisions with other vehicles ($c = 0$ implies we only check the first state).
   - $c_m \in \{static, constant\, velocity\}$ is the dynamic model used for other vehicles. $c_m = static$ treats them as static obstacles, while $c_m = constant\, velocity$ propagates them using a constant velocity prediction for the kinematic bicycle model.

2) *SGAN+MPC [3]:* this baseline uses a recurrent neural network to generate predictions [12] for the motions of neighbouring vehicles based on a history of their observations. These predictions are used to create safety constraints for an MPC optimisation that uses Monte Carlo rollouts to compute the (locally) optimal trajectory for the ego-vehicle.

We tested all possible parameter permutations of the former baseline and selected the top two performers for our results in Section IV-C. We also tested the rule-based IDM and MOBIL driving models from Section III-A. However, these models

---

[4]For the experiments in Section IV, $FoV = 50$m in front and back.

[5]l is the length of a vehicle in our simulation from Table I.

were not designed with traffic as dense as we consider in mind. As a result, they rarely manage to even initiate a lane-change. For this reason, we do not present numbers for the rule-based baselines in our quantitative evaluation.

B. Evaluation Metrics

To evaluate the performance of our approach on the benchmark scenario, we conduct two sets of experiments. In the first experiment (**E1**), other vehicles on the road do not exhibit the stop-and-go behaviour from Section III-A. In the second experiment (**E2**), half the vehicles exhibit stop-and-go behaviours. The simulation is initialised in a way that there will always be vehicles in the desired lane for a preset timeout of 40s. This means that the ego-vehicle must change lanes in traffic, and will not be able to change into the desired lane behind traffic within the timeout limit. We say that an episode ends in *failure* if the ego-vehicle collides with any other vehicle, runs out of room before the deadend, drives off the roadway, or is timed out before successfully changing lanes. We determine *success* if the ego-vehicle successfully changes lanes before the timeout, and stays in the desired lane for at least 5s without failure.

In addition to the success rate, we quantitatively evaluate performance based on the following metrics (computed only for successful episodes):

M1 *Time to merge:* this is the amount of time elapsed between the start of an episode and successful termination. Lower values are better.

M2 *Minimum distance to other vehicles:* we keep track of the minimum distance of the ego-vehicle to other vehicles on the road for the duration of an episode. Higher values are better.

C. Quantitative Results

We ran all experiments on both two- and three-lane roads, but since they are functionally the same from the ego-vehicle's point-of-view, we only present results for the more complicated three-lane road set of experiments[6]. Fig. 6 shows plots for reward gained during training. Tables II and III contain results for experiments **E1** and **E2** respectively. For each experiment, for each setting of other drivers on the road[7], we executed 100 episodes with each model to calculate our statistics. Metrics **M1** and **M2** were calculated only from the episodes that finished successfully.

We would like to point out that determination of success for SGAN+MPC was on the basis of a significantly more relaxed criterion - the ego-vehicle only had to enter the desired lane, rather than stay in it for 5s as in our case.

For the experiments in Tables II and III, there were 60 vehicles on the road, and the front-to-tail gap between two vehicles was randomly sampled to be between $[0.5, 3]$m (for reference each car is 4m long). There are several

---

[6]Our code can be found at `https://github.com/honda-research-institute/DRLDriving`

[7]$p_c = 1.0$ for *cooperative* drivers, and $p_c = 0.0$ for *aggressive* drivers. For the *mixed* case, $p_c \sim \mathcal{U}(0, 1)$.
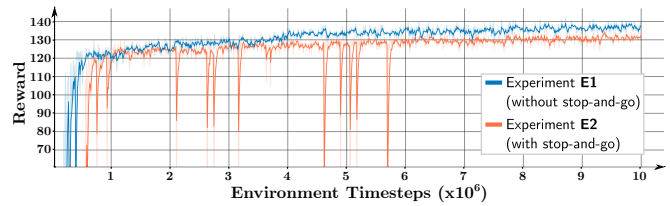

Fig. 6: Reward gained during training for experiments **E1** and **E2**. Each datapoint is the median reward of the last 10 episodes executed in the environment.

observations to be made on the basis of the numbers from Tables II and III.

O1 The MPC baselines have poor success rates since they fail to account for any inter-vehicle interactions.
- The *static* collision checking scheme performed much worse than *constant velocity*, as expected.
- These baselines suffer from both too small c_f values (ego-vehicle is aggressive which causes collisions) and too large c_f values (ego-vehicle is pessimistic which causes freezing).

O2 The MPC models, if successful, do well in terms of **M1** and **M2** as they minimise distance to a target point which makes them speed up quickly, but also brake hard if a computed trajectory might collide.

O3 SGAN+MPC has high success rates (with a relaxed criterion) on the benchmark task. It is also affected by the types of other drivers on the road as performance degrades with increasing numbers of aggressive drivers.

O4 Our model arguably performs the best out of all baselines across the board.
- In the worst-case, we achieve 80% success rate on the difficult benchmark scenario with stop-and-go behaviours, with a stricter criterion for success.
- Our times to merge (**M1**) are much lower than SGAN+MPC partly because our model does not need to wait for observations of neighbouring vehicles to generate predictions for their motion.
- We perform comparably to SGAN+MPC on **M2**.
- Our model-free approach is not affected by the distribution of drivers on the road.

In order to demonstrate the robustness and generalisation capability of our approach, we varied the number of vehicles on the road, and also the gaps between them. For this experiment, all other drivers on the road were assigned random cooperativeness parameters p_c (i.e. *mixed* drivers on the road). Since numbers for such an experiment are not available for SGAN+MPC, we only compare against our MPC baselines. Fig. 7 shows the results of this experiment. Each image is a heatmap of success rates for different numbers of vehicles and gaps, both without (Fig. 7a) and with (Fig. 7b) stop-and-go behaviours. Our model performs much better than either baseline in terms of success rates. Moreover, our model's performance only degrades slightly with increasing complexity of the scenario.

5389

TABLE II: **Experiment E1**: changing lanes on a road with dense traffic where other drivers do not exhibit random stop-and-go behaviours. Our model far surpasses the performance of MPC baselines. Numbers for this experiment are not available for SGAN+MPC baseline [3].

Metric	Other Drivers	Model		
		MPC(3.0, 0.5, 2)	MPC(3.0, 0.25, 2)	**Ours**
Success Rate	Cooperative	28.43%	35.98%	90%
	Mixed	30.61%	23.76%	90.5%
	Aggressive	22.73%	17.58%	87%
Time to merge (**M1**)	Cooperative	19.64 (±11.11)	14.33 (±12.57)	11.66 (±5.07)
	Mixed	14.78 (±10.78)	11.68 (±10.74)	11.10 (±4.99)
	Aggressive	14.72 (±11.42)	10.71 (±9.89)	11.51 (±5.04)
Minimum distance (**M2**)	Cooperative	0.37 (±0.36)	0.36 (±0.32)	0.38 (±0.23)
	Mixed	0.39 (±0.29)	0.43 (±0.40)	0.32 (±0.21)
	Aggressive	0.42 (±0.41)	0.31 (±0.31)	0.30 (±0.19))

TABLE III: **Experiment E2**: other drivers on the road exhibit random stop-and-go behaviours. MPC baselines do well in terms of **M1** and **M2**, but they have abysmal success rates. Our model performs well across the board, outperforming SGAN+MPC on the important metric **M1**. Note that SGAN+MPC uses a relaxed criterion for success.

| Metric	Other Drivers	Model			
		MPC(3.0, 0.5, 2)	MPC(3.0, 0.25, 2)	SGAN+MPC [3]	**Ours**
Success Rate	Cooperative	23.44%	25.95%	99%	85.5%
	Mixed	21.43%	21.62%	97%	87%
	Aggressive	20.21%	20.63%	81%	80%
Time to merge (**M1**)	Cooperative	16.54 (±12.44)	14.69 (±12.82)	23.97 (±5.10)	16.40 (±9.07)
	Mixed	19.88 (±11.31)	8.43 (±8.46)	25.76 (±5.37)	18.02 (±9.58)
	Aggressive	14.54 (±10.98)	8.71 (±8.26)	29.32 (±7.07)	17.99 (±9.94)
Minimum distance (**M2**)	Cooperative	0.50 (±0.39)	0.38 (±0.40)	0.49 (±0.18)	0.38 (±0.24)
	Mixed	0.44 (±0.39)	0.32 (±0.24)	0.42 (±0.19)	0.35 (±0.23)
	Aggressive	0.38 (±0.27)	0.39 (±0.38)	0.30 (±0.14)	0.37 (±0.23))

(a)

(b)
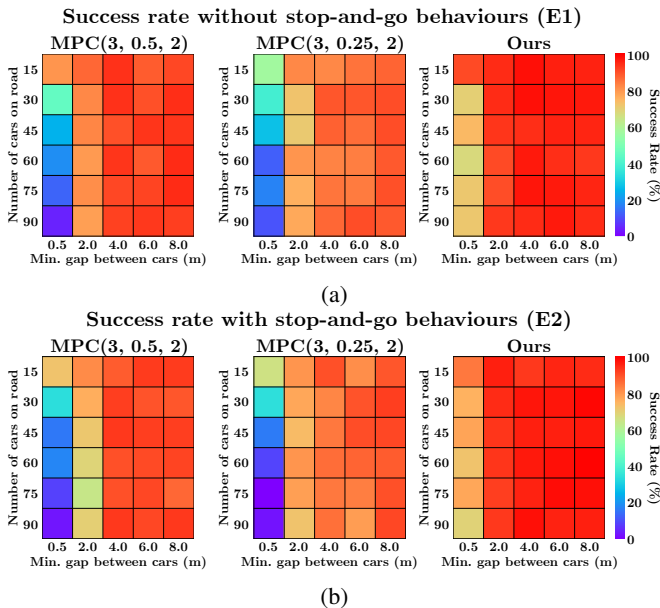
Fig. 7: Comparison of model performance with varying numbers of vehicles on the road and gaps between them.

V. CONCLUSION

We present a benchmark scenario for the task of driving and merging in dense traffic. This requires complex decision-making algorithms that need to reason about interactions with neighbouring vehicles, implicitly or explicitly, in order to successfully accomplish the task. Our model-free approach does not rely on driver models of other vehicles, or even on predictions about their motions, and is successful on the benchmark. It outperforms traditional rule-based models (which fail entirely) and model-predictive control based models. It also performs comparably to a new approach [3] in terms of success rates, and does better on key metrics, without using predictions about trajectories of other vehicles.

It is hard to comment on how well our control policy will transfer and adapt to real-world dense traffic environments. We tried to emulate real-world driving behaviours as best as possible by varying the cooperativeness and perception range parameters of other vehicles on the road, and with random *stop-and-go* behaviours. Additionally, the random lane-change probability of other drivers was one way in which we tried to introduce perception uncertainty about the dynamic state of neighbouring vehicles.

Since we blindly execute our policy, we cannot guarantee collision free execution. Even in the worst-case, we cannot guarantee that the autonomous vehicle will remain stationary if all actions are dangerous. For this reason, we would like to investigate the use of an *overseer* that determines whether an action predicted by our policy is safe to execute.

REFERENCES

[1] Zlatan Ajanovic, Bakir Lacevic, Barys Shyrokau, Michael Stolz, and Martin Horn. Search-based optimal motion planning for automated driving. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*, pages 4523–4530, 2018.

[2] Alexander Amini, Wilko Schwarting, Guy Rosman, Brandon Araki, Sertac Karaman, and Daniela Rus. Variational autoencoder for end-to-end control of autonomous driving with novelty detection and training de-biasing. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*, pages 568–575, 2018.

[3] Sangjae Bae, Dhruv Saxena, Alireza Nakhaei, Chiho Choi, Kikuo Fujimura, and Scott Moura. Cooperation-aware lane change control in dense traffic. *CoRR*, abs/1909.05665, 2019.

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[5] Martin Buehler, Karl Iagnemma, and Sanjiv Singh, editors. *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic, George Air Force Base, Victorville, California, USA*, volume 56 of *Springer Tracts in Advanced Robotics*. Springer, 2009.

[6] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 2722–2730, Washington, DC, USA, 2015. IEEE Computer Society.

[7] Jianyu Chen, Bodi Yuan, and Masayoshi Tomizuka. Model-free deep reinforcement learning for urban autonomous driving. *CoRR*, abs/1904.09503, 2019.

[8] Po-Wei Chou, Daniel Maturana, and Sebastian Scherer. Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 834–843, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[9] Felipe Codevilla, Matthias Miiller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 1–9, 2018.

[10] Jaime F. Fisac, Eli Bronstein, Elis Stefansson, Dorsa Sadigh, S. Shankar Sastry, and Anca D. Dragan. Hierarchical game-theoretic planning for autonomous vehicles. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pages 9590–9596, 2019.

[11] Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G Bellemare, Joelle Pineau, et al. An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, 11(3-4):219–354, 2018.

[12] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: socially acceptable trajectories with generative adversarial networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2255–2264, 2018.

[13] Mikael Henaff, Alfredo Canziani, and Yann LeCun. Model-predictive policy learning with uncertainty regularization for driving in dense traffic. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[14] Carl-Johan Hoel, Katherine Rose Driggs-Campbell, Krister Wolff, Leo Laine, and Mykel J. Kochenderfer. Combining planning and deep reinforcement learning in tactical decision making for autonomous driving. *CoRR*, abs/1905.02680, 2019.

[15] Thomas M. Howard and Alonzo Kelly. Optimal rough terrain trajectory generation for wheeled mobile robots. *I. J. Robotics Res.*, 26(2):141–166, 2007.

[16] Yeping Hu, Alireza Nakhaei, Masayoshi Tomizuka, and Kikuo Fujimura. Interaction-aware decision making with adaptive strategies under merging scenarios. *CoRR*, abs/1904.06025, 2019.

[17] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103, Oct 2017.

[18] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26), 2019.

[19] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, and Lipika Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416–442, 2015.

[20] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1094–1099, June 2015.

[21] Alex Kuefler, Jeremy Morton, Tim Allan Wheeler, and Mykel J. Kochenderfer. Imitating driver behavior with generative adversarial networks. In *IEEE Intelligent Vehicles Symposium, IV 2017, Los Angeles, CA, USA, June 11-14, 2017*, pages 204–211, 2017.

[22] Jesse Levinson, Jake Askeland, Jan Becker, Jennifer Dolson, David Held, Soeren Kammel, J Zico Kolter, Dirk Langer, Oliver Pink, Vaughan Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 163–168. IEEE, 2011.

[23] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[24] Liang Ma, Jianru Xue, Kuniaki Kawabata, Jihua Zhu, Chao Ma, and Nanning Zheng. Efficient sampling-based motion planning for on-road autonomous driving. *IEEE Trans. Intelligent Transportation Systems*, 16(4):1961–1976, 2015.

[25] Brian Paden, Michal Cáp, Sze Zheng Yong, Dmitry S. Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Trans. Intelligent Vehicles*, 1(1):33–55, 2016.

[26] Dean Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*. Morgan Kaufmann, January 1989.

[27] Dorsa Sadigh, Shankar Sastry, Sanjit A. Seshia, and Anca D. Dragan. Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and Systems XII, University of Michigan, Ann Arbor, Michigan, USA, June 18 - June 22, 2016*, 2016.

[28] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[29] W. Schwarting, J. Alonso-Mora, L. Pauli, S. Karaman, and D. Rus. Parallel autonomy in automated vehicles: Safe motion generation with minimal intervention. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1928–1935, May 2017.

[30] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):187–210, 2018.

[31] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *CoRR*, abs/1610.03295, 2016.

[32] P. Trautman and A. Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 797–803, Oct 2010.

[33] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Phys. Rev. E*, 62:1805–1824, Aug 2000.

[34] Martin Treiber and Arne Kesting. Modeling lane-changing decisions with mobil. In Cécile Appert-Rolland, François Chevoir, Philippe Gondret, Sylvain Lassarre, Jean-Patrick Lebacque, and Michael Schreckenberg, editors, *Traffic and Granular Flow '07*, pages 211–221, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[35] S. Ulbrich, S. Grossjohann, C. Appelt, K. Homeier, J. Rieken, and M. Maurer. Structuring cooperative behavior planning implementations for automated driving. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 2159–2165, Sep. 2015.

[36] Peter Wolf, Christian Hubschneider, Michael Weber, Andre Bauer, Jonathan Hartl, Fabian Durr, and Johann Marius Zöllner. Learning how to drive in a real world simulation with deep q-networks. In *IEEE Intelligent Vehicles Symposium, IV 2017, Los Angeles, CA, USA, June 11-14, 2017*, pages 244–250, 2017.

[37] Wenda Xu, Junqing Wei, John M. Dolan, Huijing Zhao, and Hongbin Zha. A real-time motion planner with trajectory optimization for autonomous vehicles. In *IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA*, pages 2061–2067, 2012.