2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Cooperation-Aware Lane Change Maneuver in Dense Traffic based on Model Predictive Control with Recurrent Neural Network

Sangjae Bae[1*], Dhruv Saxena[2], Alireza Nakhaei[3], Chiho Choi[3], Kikuo Fujimura[3], and Scott Moura[1]

***Abstract*— This paper presents a *real-time* lane change control framework of autonomous driving in dense traffic, which exploits cooperative behaviors of other drivers. This paper focuses on heavy traffic where vehicles cannot change lanes without cooperating with other drivers. In this case, classical robust controls may not apply since there is no "safe" area to merge to without interacting with the other drivers. That said, modeling complex and interactive human behaviors is highly non-trivial from the perspective of control engineers. We propose a mathematical control framework based on Model Predictive Control (MPC) encompassing a state-of-the-art Recurrent Neural network (RNN) architecture. In particular, RNN predicts interactive motions of other drivers in response to potential actions of the autonomous vehicle, which are then systematically evaluated in safety constraints. We also propose a *real-time* heuristic algorithm to find locally optimal control inputs. Finally, quantitative and qualitative analysis on simulation studies are presented to illustrate the benefits of the proposed framework.**

I. Introduction

An autonomous-driving vehicle is no longer a futuristic concept and extensive research have been conducted in various aspects, spanning from localization, perception, and controls, including implementation and validation. From the perspective of control engineering, designing a controller that ensures safety in various traffic conditions (e.g., arterial-roads, highways in free-flow/dense traffic, with/without traffic lights) has been a principal research focus. This paper focuses on lane change in dense traffic.

Due to the importance of safety, many publications have focused on robust control that guarantee collision avoidance in the face of uncertainty. Unlike other autonomous robots, autonomous-driving vehicles can take advantage of existing roadway infrastructure, such as arterial roads and highways. By exploiting the roadway for guidance, longitudinal control designs have proven their effectiveness in maintaining a safety distance to a front vehicle, as well as maintaining driving comfort and energy efficiency [1], [2]. A lane-changing controller extends longitudinal control without significant mathematical modifications by recognizing additional lanes and vehicles [3]. Lane-changing controllers often utilize probabilistic models [4], [5] or scenario-based models [6], [7] to predict adjacent drivers' motions and intentions, which primarily define a safe area in a robust manner.

[1]University of California, Berkeley, USA. {sangjae.bae, smoura}@berkeley.edu. *Corresponding author.
[2]Carnegie Mellon University, USA. dsaxena@andrew.cmu.edu
[3]Honda Research Institute, USA. {anakhaei, cchoi, kfujimura}@honda-ri.com
This work was funded by Honda Research Institute, USA.

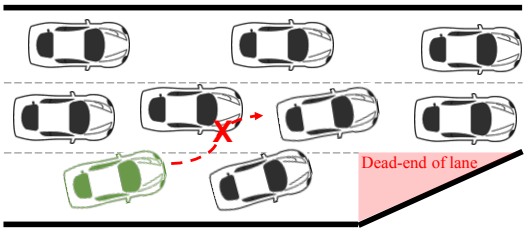

Fig. 1: The autonomous-driving vehicle (in green) intends to change lanes, within a restricted merging area. The traffic is dense with narrow inter-vehicle intervals that are spatially insufficient for a vehicle to merge into. The autonomous-driving vehicle would get stuck in the merging area, unless other drivers slow down to make a space for the vehicle.

Unfortunately, those robust methods may not apply in highly dense traffic without "safe" areas to merge into (see Fig. 1). In dense traffic conditions, interactions with other drivers are essential to successfully change lanes by leveraging cooperative behavior. That is, drivers can potentially slow down to create a spatial interval so that another vehicle can merge in. That said, modeling interactive behaviors by formal statistical or scenario-based approaches is highly non-trivial, due to its complex and stochastic nature as well as computational challenges. Consequently, lane changing that exploits cooperation with other drivers remains as an open research question. Nonetheless, solving this problem is critical to realizing fully autonomous-driving vehicles [8].

There is a small yet rich body of literature focused on human interactions during lane change or merging [9]–[14]. These methods focus on interactions between just *two* vehicles at a time, in lane changing or lane keeping. In more realistic settings of highly dense traffic, each vehicle's motion is reactive to multiple vehicles simultaneously. Moreover, the ego-vehicle lacks knowledge of other drivers' interaction dynamics. Thus game theoretic or first-principles models are of limited use. Recently, Reinforcement Learning (RL) techniques have been thoroughly investigated. RL methods are appealing for their potential in finding maneuvers under interactive or unknown traffic conditions [12], [15], [16] without any first-principles modeling. However, safety, reliability, and/or interpretability limit RL's practical use when human safety is at stake, see e.g. [17], [18].

That being said, an increasing body of literature has applied Deep Neural Network architectures for autonomous driving and Advanced Driving Assistance Systems (ADAS). Deep neural networks have proven useful in explaining complex environments [19]–[21]. Recurrent Neural Network (RNN) architectures have been particularly effective in pre-

dicting motions of human (drivers), with respect to both accuracy [22], [23] and computational efficiency [24]. Therefore, it would be natural to take advantage of those advances for controller design, yet still leverage rigorous control theory [25] and established vehicle dynamics models [26] to yield safety guarantees. That is, we seek a controller that embeds accurate predictions of human interaction via RNNs, yet still maintains safety guarantees afforded by control theory. The incorporation of these two elements would yield a controller that is reliable, interpretable, and tunable, while containing a data-driven model that captures interactive motions between drivers. It is still challenging to mathematically incorporate RNN into formal controller design, and to solve the corresponding control problem effectively and efficiently. We address these challenges in this paper.

We add two original contributions to the literature: (i) We propose a mathematical control framework that systematically evaluates several other drivers' interactive motions, in highly dense traffic on the highway. The framework partially exploits Recurrent Neural networks (RNN) to predict (nonlinear) interactive motions, which are incorporated as safety constraints. Optimal control inputs are obtained via Model Predictive Control (MPC) based on vehicle dynamics. (ii) We propose a *real-time* rollout-based heuristic algorithm that sequentially evaluates other driver's reactions and finds locally optimal solutions. The idea of incorporating RNN as a prediction model into an MPC controller is straightforward. However, the complexity and nonlinearity of RNN give challenges for classical optimization algorithms. We show our heuristic algorithm finds locally optimal solutions effectively and efficiently.

The paper is organized in the following manner. Section II describes the mathematical formulation of the proposed control framework, and the heuristic algorithm. Section III presents and analyzes simulation results to validate the proposed framework. Section IV summarizes the paper's contributions.

II. Controller Design

To design a lane changing controller for highly dense traffic, it is critical to estimate how surrounding vehicles will react to our vehicle. Predicting their behaviors, however, is challenging due to their interactive nature, and it is complex to model either physically or statistically, using formal methods such as Markov chains. We therefore employ a Recurrent Neural Network architecture (RNN), particularly a Social Generative Adversarial Network (SGAN) that has been successful in predicting interactive human behaviors in crowded spaces [24]. Fig. 2 shows a schematic of the control framework. The key intuition of the framework is the following. The controller uses Model Predictive Control (MPC) as the basis, which takes current states (position, velocity, and heading) and positions of surrounding vehicles as inputs and provides a pair of acceleration and steering angle as output. In the MPC controller, the optmization objective is to track a target lane, a desired speed, and penalize large control efforts, while satisfying system dynamics

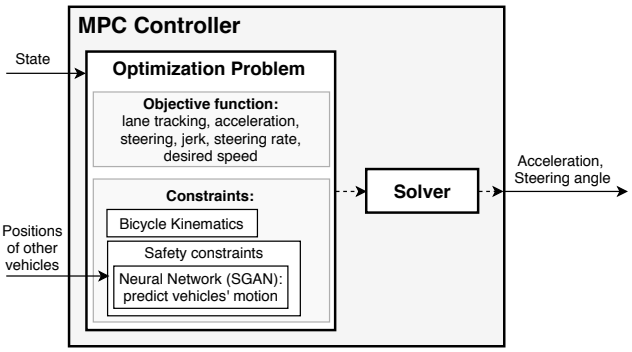

Fig. 2: Diagram of the control framework with a Recurrent Neural network, SGAN. The solver (small box in MPC controller) can be based on different algorithms.

and avoiding collisions. The solver, based on a heuristic algorithm, solves the optimization problem at each time step by exploiting a trained SGAN to predict the other vehicles' interactive motions that are reactive to the ego vehicle's [1] actions. Note that the predicted motions are incorporated into the constraint of collision avoidance. The controller design is detailed in the following sections.

A. System Dynamics

We utilize the nonlinear kinematic bicycle model in [26] to represent the vehicle dynamics. For completeness, we rewrite the kinematics here:

$$\dot{x} = v\cos(\psi+\beta) \tag{1}$$
$$\dot{y} = v\sin(\psi+\beta) \tag{2}$$
$$\dot{\psi} = \frac{v}{l_r}\sin(\beta) \tag{3}$$
$$\dot{v} = a \tag{4}$$
$$\beta = \tan^{-1}\left(\frac{l_r}{l_f+l_r}\tan(\delta)\right) \tag{5}$$

where (x,y) is the Cartesian coordinate for the center of vehicle, ψ is the inertial heading, v is the vehicle speed, a is the acceleration of the car's center in the same direction as the velocity, and l_f and l_r indicate the distance from the center of the car to the front axles and and to the rear axles, respectively. The control inputs are: (front wheel) steering angle δ and acceleration a. We use Euler discretization to obtain a discrete-time dynamical model in the form:

$$z(t+1) = f(z(t), u(t)), \tag{6}$$

where $z = \begin{bmatrix} x & y & \psi & v \end{bmatrix}^\top$ and $u = \begin{bmatrix} a & \delta \end{bmatrix}^\top$ for time t.

B. Control Objective

The control objective is to merge to target lane, while avoiding collisions with other vehicles. We prefer to change lanes sooner. We also prefer smooth accelerations and steering for drive comfort. The objective function is formulated:

$$J = \sum_{\ell=t}^{t+T} \lambda_{div}(x(\ell|t); x_{\text{end}})D(\ell|t) \tag{7}$$

[1] We refer to "ego vehicle" as the vehicle that is controlled.

$$+\sum_{\ell=t}^{t+T}\lambda_v\|v(\ell|t)-v^{\text{ref}}\|^2 \tag{8}$$

$$+\sum_{\ell=t}^{t+T-1}\lambda_\delta\|\delta(\ell|t)\|^2 \tag{9}$$

$$+\sum_{\ell=t}^{t+T-1}\lambda_a\|a(\ell|t)\|^2 \tag{10}$$

$$+\sum_{\ell=t+1}^{t+T-1}\lambda_{\Delta\delta}\|\delta(\ell|t)-\delta(\ell-1|t)\|^2 \tag{11}$$

$$+\sum_{\ell=t+1}^{t+T-1}\lambda_{\Delta a}\|a(\ell|t)-a(\ell-1|t)\|^2 \tag{12}$$

where $(\ell|t)$ indicates time ℓ based on the measurements at time t. Symbol x_{end} is the latitude coordinate of the road-end, $D(\ell|t)$ is the distance norm for the vector between the ego vehicle's center and the target lane at time ℓ, v^{ref} is the reference velocity. Each penalty is regularized with λ_{div}, λ_v, λ_δ, λ_a, $\lambda_{\Delta\delta}$, and $\lambda_{\Delta a}$, respectively. We incentivize a timely lane change with the dynamic weight λ_{div}, written as a convex-function, $\lambda_{div}=\|\frac{1}{x_{\text{end}}-x}\|$. The term (7) penalizes the divergence of the center of the vehicle from the vertical center of the target lane. The term (9) and (10) penalize the control effort of steering angle and acceleration, respectively. The term (11) and (12) penalize the steering rate and jerk, respectively, for drive comfort.

C. Constraints for Safety with a Recurrent Neural Network

To quantify safety, we consider a distance measure between two vehicles. That is, if a distance between two vehicles is zero, it means they collide into each other. A mathematical measure of distance between two vehicles depends on how each vehicle is shaped, which we discuss next.

1) Vehicle Shape Model: We model the vehicle shape with three circles as illustrated in Fig. 3. A Euclidean distance between two circles are then computed analytically. To constrain safety, a minimum distance between any pairs of circles must be greater than a safety bound. Formally:

$$g_i(x,y)=\min_{p,q\in\{-1,0,1\}} d_i(p,q)\geq\epsilon, \tag{13}$$

$$\begin{aligned}&\text{where: } d_i(p,q)\\&=\Big[\big((x+p(h-w)\cos\psi)-(x_i+q(h_i-w_i)\cos\psi_i)\big)^2\\&+\big((y+p(h-w)\sin\psi)-(y_i+q(h_i-w_i)\sin\psi_i)\big)^2\Big]^{\frac{1}{2}}\\&-(w+w_i),\end{aligned} \tag{14}$$

and w and h are, respectively, half width and half height of the vehicle, respectively, and ϵ is a safety bound. Subscript i indicates vehicle i.

2) Interactive Motion Prediction: Drivers' motions are responsive to interactions with each other, and consequently the motions must be predicted simultaneously. We adopt SGAN from [24] which efficiently and effectively captures interactions between agents (drivers). SGAN is composed of a generator and discriminator that are adversarial to each other. Both the generator and discriminator are comprised of long-short term memory (LSTM) networks to account for the sequential nature of agents' motion. The following module aggregates the agents' motion states to evaluate their interactions. This pooling process is essentially to share the motion history between agents, generating social interactions as a pooled tensor P_i for *each* agent i. The decoder in the generator then predicts multiple trajectories that are socially interactive with each other. SGAN takes as an input a sequence of positions for each agent within a scene over an observation time horizon T_{obs}. It outputs a sequence of positions for each agent over a prediction time horizon T_{pred}. Interested readers are referred to [24] for more details. It is important to highlight that a trained SGAN will predict the most probable reactions of other vehicles based on the control commands on the ego vehicle, and the history of previous actions. However, in reality, the reactions of other vehicles might be different.

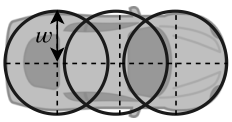

Fig. 3: Vehicle shape modeled by three circles. Note that multiple circles of various radii can be applied to any vehicle shape and size.

3) Incorporation of Recurrent Neural Network into Safety Constraints: A trained SGAN predicts the evolution of vehicle i's centroid. These predictions are incorporated into safety constraints (13). To formalize, we denote a trained SGAN as a function $\phi(t)$ that maps observed trajectories to predicted trajectories for N nearby vehicles:

$$\begin{aligned}&\phi(t)\ :\\&\begin{bmatrix}(x_1(t),y_1(t)) & \cdots & (x_N(t),y_N(t))\\ \vdots & \vdots & \vdots\\ (x_1(t-T_{obs}+1), & \cdots & (x_N(t-T_{obs}+1),\\ \quad y_1(t-T_{obs}+1)) & & \quad y_N(t-T_{obs}+1))\end{bmatrix}\\&\qquad\downarrow\\&\begin{bmatrix}(\hat{x}_1(t+1),\hat{y}_1(t+1)) & \cdots & (\hat{x}_N(t+1),\hat{y}_N(t+1))\\ \vdots & \cdots & \vdots\\ (\hat{x}_1(t+T_{pred}), & \cdots & (\hat{x}_N(t+T_{pred}),\\ \quad \hat{y}_1(t+T_{pred})) & & \quad \hat{y}_N(t+T_{pred}))\end{bmatrix},\end{aligned} \tag{15}$$

where $\hat{\cdot}$ indicates a predicted value. Given the observations until time t, the coordinates of vehicle i at time $t+1$ are represented as $\hat{x}_i(t+1)=\phi_{i,x}(t)$ and $\hat{y}_i(t+1)=\phi_{i,y}(t)$.

D. Control Problem Formulation

The complete optimization problem for the receding horizon control is:

$$\min_{z,a,\delta}\ J=\sum_{\ell=t}^{t+T}\Big(\lambda_{div}(x(\ell|t);x_{\text{end}})D(\ell|t)$$

$$+ \lambda_v \|v(\ell|t) - v^{\text{ref}}\|^2\Big) + \sum_{\ell=t}^{t+T-1} \Big(\lambda_\delta \|\delta(\ell|t)\|^2 + \lambda_a \|a(\ell|t)\|^2\Big) + \sum_{\ell=t+1}^{t+T-1} \Big(\lambda_{\Delta\delta} \|\delta(\ell|t) - \delta(\ell-1|t)\|^2 + \lambda_{\Delta a} \|a(\ell|t) - a(\ell-1|t)\|^2\Big) \quad (16)$$

subject to:

$$z(\ell+1|t) = f(z(\ell|t), \delta(\ell|t), a(\ell|t)) \quad (17)$$

$$g_i(z(\ell+1|t)) \geq \epsilon, \quad \forall i \in \{1, \cdots, N_{veh}\} \quad (18)$$

$$\delta(\ell|t) \in [\delta_{\min}, \delta_{\max}] \quad (19)$$

$$a(\ell|t) \in [a_{\min}, a_{\max}] \quad (20)$$

$$x(\ell+1|t) \leq x_{\text{end}} \quad (21)$$

where $\delta_{\min}$ and $\delta_{\max}$ are the minimum and maximum steering angle rate, respectively, and $a_{\min}$ and $a_{\max}$ are the minimum and maximum acceleration, respectively. The constraint (21) ensures that the ego vehicle changes lanes before the road-end x_{end}.

E. Heuristic Algorithm

Note that the inequality constraint (18) does not form a convex set. We assume that the mathematical form of the SGAN is not known, i.e. it is black-box, and we can only evaluate its output given an input. Along with the nonlinear equality constraint (17), it is non-trivial to find a solution with canonical optimization algorithms, such as gradient descent or Newton methods. We propose a rollout-based algorithm that finds locally optimal solutions in a time efficient manner. The process is as follows. From the current state at time t, the controller randomly generates a finite set of control sequences over the time horizon T. That is, $\mathcal{U} = \{U_1, \cdots, U_j, \cdots, U_{N_{\text{sim}}}\}$ where $U_j = \begin{bmatrix} u_j(0|t) & \cdots & u_j(T-1|t) \end{bmatrix}^\top$. Then the controller evaluates the cumulative cost (16) for each control sequence and chooses one that has a minimum cost over the time horizon T. The controller takes the first control input from the optimal sequence and discards the rest. Algorithm 1 formalizes the procedure.

The process of finding the optimal sequence (line 5 in Algorithm 1) is the following. At each time step ℓ, consider the j^{th} candidate control sequence U_j. The controller (i) runs SGAN to predict the motions of surrounding vehicles, (ii) it propagates the ego vehicle through the dynamics (17) given control U_j, (iii) it checks the constraints (18)-(21), and discards the candidate sequence if the constraints are violated, and (iv) it updates the cumulative cost (16). Predicted positions of other vehicles conditioned on controls of the ego vehicle imply the cooperativeness of the other vehicles. Note that evaluating each control sequence candidate can be parallelized, and therefore a parallel computation framework can be applied to improve computational efficiency.

Algorithm 1 is straightforward to implement, however, it may require substantial computation power to find a solution in real-time, depending on the time horizon T and sample size N_{sim}. That said, unlike other applications of motion planning algorithms, autonomous driving on roads has specific patterns in terms of actions in specific driving scenarios. For example, if a vehicle keeps driving in the same lane, large steering angles do not have to be explored. Similarly, if a vehicle changes lane to the left, then steering angles to right do not have to be explored. From those observations, a smaller size of action spaces (or, domain of control inputs) can be specified in each driving scenario: (i) keeping lane, (ii) changing lane to the left, and (iii) changing lane to the right. Denote the action spaces of lane keeping by $\mathcal{A}_M$, changing lane to the left by $\mathcal{A}_L$, and changing lane to the right by $\mathcal{A}_R$. Then each action space can read:

$$\mathcal{A}_M = [a_{\min}, a_{\max}] \times [\alpha\delta_{\min}, \alpha\delta_{\max}], \quad (22)$$

$$\mathcal{A}_L = [a_{\min}, a_{\max}] \times [0, \delta_{\max}], \quad (23)$$

$$\mathcal{A}_R = [a_{\min}, a_{\max}] \times [\delta_{\min}, 0], \quad (24)$$

where $\alpha \in [0, 1]$, $\delta > 0$ indicates steering to the left, and $\delta < 0$ indicates steering to the right. The action space (23) indicates that we only consider steering angles to the left when turning left, and similarly for (24).

Algorithm 1: Monte Carlo Roll-out Algorithm

Init : states $z = z_0$,
other vehicles' position $(x_i, y_i) = (x_{i0}, y_{i0})$
for all $i \in \{1, \cdots, N_{veh}\}$

1 **while** $x < x_{end}$ *and* $D \neq 0$ **do**
2 Randomly generate a total of N_{sim} control sequences over T while satisfying (19), (20)
3 $\mathcal{U} = \{U_1, \cdots, U_j, \cdots, U_{N_{\text{sim}}}\}$
4 Find the optimal sequence that minimizes cumulative cost over T and that is feasible with (18)
5 $U_* \leftarrow \text{argmin}_{U \in \mathcal{U}}$ (16)
6 Propagate through dynamics (17) with the first element of U_*
7 $z \leftarrow f(z, [U_*]_0)$
8 Observe positions of other vehicles at the current time t
9 $(x_i, y_i) \leftarrow (x_i(t), y_i(t))$ for all i
10 **end**

III. Simulation Study

A. Driver Model of Cooperativeness

For the simulation study, we have a driver model in which the longitudinal dynamics is controlled by an intelligent driver model (IDM) from [27]. The lane changing behavior is governed by the strategy of Minimizing Overall Braking Induced by Lane changes (MOBIL) from [28]. The driver model is also based on the bicycle kinematics in Section II-A. Additionally, we introduce a parameter for cooperativeness

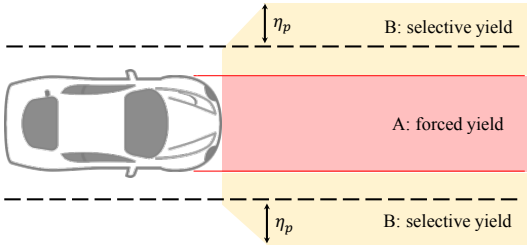

Fig. 4: Forced yield zone (red area labeled with A) and selective yield zone (yellow area labeled with B). The dashed lines indicate the boundaries of the center lane. If a vehicle from the next lane intersects with the path of the vehicle, i.e. in zone A, the vehicle in the center lane must stop and wait until the other vehicle cuts into the center lane. If the other vehicle intersects with zone B, then the vehicle in the center lane decides to either yield or not, according to the cooperativeness parameter η_c. Zone B corresponds to a vertical perception range, and the range can be adjusted by η_p.

$\eta_c \in [0, 1]$ to the driver model. If $\eta_c = 1$, then a vehicle stops and waits until another vehicle within the selective yield zone (zone B in Fig. 4) overtakes. If $\eta_c = 0$, then the vehicle ignores neighboring vehicles and drives forward. If $0 < \eta_c < 1$, then the yield action is randomly sampled from the Bernoulli distribution with probability $p = \eta_c$.

B. Simulation Scenario Overview

In the simulation study, we consider a highway segment with dense traffic, illustrated in Fig. 5. In the scene, the ego vehicle (in green) plans to change lanes from the current lane (lane 1) to the next lane (lane 2). However, the vehicles are driving with narrow distance intervals between each other, which are not large enough for the ego vehicle to cut in without cooperating with other vehicles (in various shades of blue). Given this challenge, the controller seeks a pair of acceleration and steering angle trajectories to induce the other vehicles to make space for the ego vehicle to cut in.

Only the ego vehicle uses the controller designed in Section II and the other vehicles (in blue) are based on the driver model in Section III-A with heterogeneous parameter settings. The controller design parameters are such that the divergence from the target lane is more significantly penalized relative to the other penalty terms. The list of controller design parameters is detailed in Table I. Note that it is preferred to have a receding time horizon T that is sufficiently long to account for other drivers' reaction time. The SGAN embedded in the controller predict positions of any vehicles within the range of 10 $[m]$ from the ego vehicle.

The driver model parameters are set in the following manner: (i) The minimum distance between vehicles is shorter than the vehicle length, which creates the highly dense traffic we are interested in. (ii) The other drivers drive realistically, i.e. they apply physically feasible acceleration and steering angles. (iii) The other drivers' motions are noisy, i.e. their center positions oscillate and their accelerations are augmented with noise to challenge the controller's robustness. The driver model parameters are tabulated in Table II. Note that the parameters of each driver are sampled from uniform distributions, denoted by $\mathcal{U}(\cdot, \cdot)$, to account for heterogeneous behaviors of drivers. Additionally, we assume that all vehicles have the same physical dimensions, for brevity. We also assume that vehicles have perfect perceptions; that is, there is no measurement bias in the vehicles' positions.

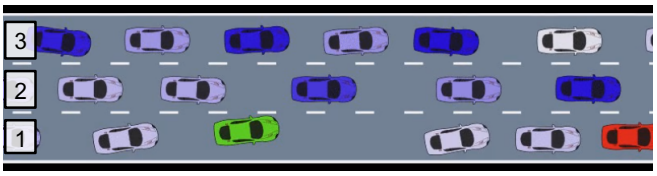

Fig. 5: Traffic scenario with three lanes. Green indicates the ego vehicle, red indicates a stopped vehicle, i.e. dead-end of the lane, and the shades of blue indicate other drivers. The lighter blue corresponds to less cooperative drivers, i.e. η_c near 0. Each number in the white rectangle indicates the lane number.

Param	Description	Value
T	Receding time horizon $[s]$	2.8
N_{sim}	Control sequence sample size	32
Δt	Time step size $[s]$	0.4
λ_{div}	Weight on divergence from a target lane	12000
λ_v	Weight on divergence from a desired speed	1000
λ_δ	Weight on steering angle	500
λ_a	Weight on acceleration	500
$\lambda_{\Delta\delta}$	Weight on steering rate	100
$\lambda_{\Delta a}$	Weight on jerk	100
δ_{min}	Minimum steering angle $[rad]$	-0.3
δ_{max}	Maximum steering angle $[rad]$	0.3
a_{min}	Minimum acceleration $[m^2/s]$	-4.0
a_{max}	maximum acceleration $[m^2/s]$	3.5
x_{end}	Length of the current lane $[m]$	50
v^{ref}	Desired velocity $[m/s]$	10

TABLE I: Controller design parameters

The simulation study utilizes [29] for simulation and visualization, and runs on a Linux machine (Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz with NVIDIA GeForce GTX TITAN Black).

C. Training SGAN for Motion Prediction

Table III lists hyper parameters for the SGAN used in the simulation study. Training and validation datasets are generated by simulations with only the driver model from Section III-A, with heterogeneous parameters from Table II. The dataset is collected from multiple scenarios in various traffic densities, from free flow to dense traffic. In the dataset, we also add noise to the positions so that the trained neural network becomes more robust to noisy inputs. With a total

Param	Description	Value
$\tilde{v}^{\text{ref}}$	Reference velocity $[m/s]$	$\mathcal{U}(2, 5)$
$\tilde{T}$	Safe time headway $[s]$	$\mathcal{U}(1, 2)$
$\tilde{a}_{\text{max}}$	Maximum acceleration $[m/s^2]$	$\mathcal{U}(2.5, 3.5)$
$\tilde{b}$	Comfortable deceleration $[m/s^2]$	$\mathcal{U}(1.5, 2.5)$
$\tilde{\delta}$	Acceleration exponent	$\mathcal{U}(3.5, 4.5)$
$\tilde{s}_0$	Minimum distance to front vehicle $[m]$	$\mathcal{U}(1, 3)$
η_c	Cooperativeness $\in [0, 1]$	$\mathcal{U}(0, 1)$
η_p	Perception range $[m]$	$\mathcal{U}(-0.15, 0.15)$
w	Length from center to side of vehicle $[m]$	0.9
h	Length from center to front of vehicle $[m]$	2

TABLE II: Driver model design parameters

Param	Description	Value
T_{obs}	Observation time horizon	8
T_{pred}	Prediction time horizon	2
N_b	Batch size	64
D_{emb}	Embedding dimension	64
D_{mlp}	MLP dimension	256
$D_{G,e}$	Hidden layer dimension of encoder (generator)	32
$D_{G,d}$	Hidden layer dimension of decoder (generator)	64
$D_{D,e}$	Hidden layer dimension of encoder (discriminator)	64
D_{bot}	Bottleneck dimension in Pooling module	1024
α_G	Generator learning rate	$5 \cdot 10^{-4}$
α_D	Discriminator learning rate	$5 \cdot 10^{-4}$

TABLE III: SGAN parameters

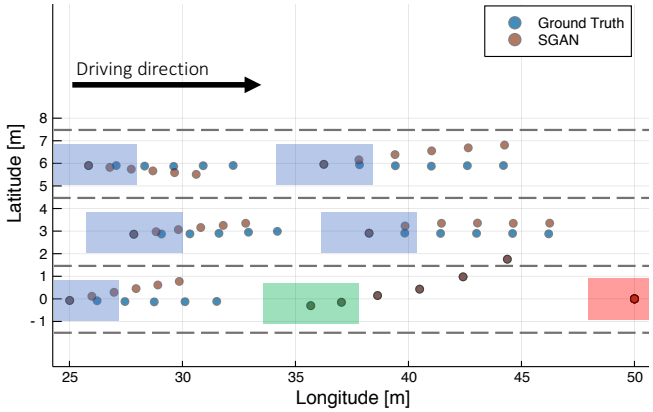

Fig. 6: Motions predicted by SGAN. Rectangles in green, blue, and red indicate the ego vehicle, other vehicles, and the stopped vehicles respectively. Dashed lines represent the lane boundary. Each circular point indicates the vehicle center at each time step.

of 27550 data points, training the SGAN with a GPU takes approximately 18 hours. An example of motions predicted by SGAN, compared to ground truth, are illustrated in Fig. 6.

The SGAN has 1.872 $[m]$ of average displacement error and 2.643 $[m]$ of final displacement error after $T_{pred} = 2$, for the training data. It is important to note that the training dataset does not include the testing scenarios. Also, our ultimate goal is safe lane change, as discussed next, not zero-error motion prediction. Note that SGAN can provide a distribution of predicted motions, which can be incorporated into the optimization problem as chance constraints, thereby enabling a robust formulation. Also, different methods for designing loss functions for SGAN training can be applied, which are topics for future work.

D. Simulation Results and Analysis

Fig. 7 illustrates the simulated position trajectories. The ego vehicle (in green) often stops and waits before merging (at t_1 and t_2), since otherwise the safety constraint (18) can be violated. As the ego vehicle gets closer to the target lane (the middle lane), the vehicle on the target lane reacts by slowing down the speed, to make a space for the ego vehicle to cut in (at t_3). As soon as an enough space is made, the ego vehicle merges into the target lane.

These interactive behaviors with the ego vehicle are also observed in Fig. 8. While the ego vehicle is interacting with other vehicles to merge over, between 3.2∼18.6 $[s]$, both the acceleration and steering angle fluctuate significantly

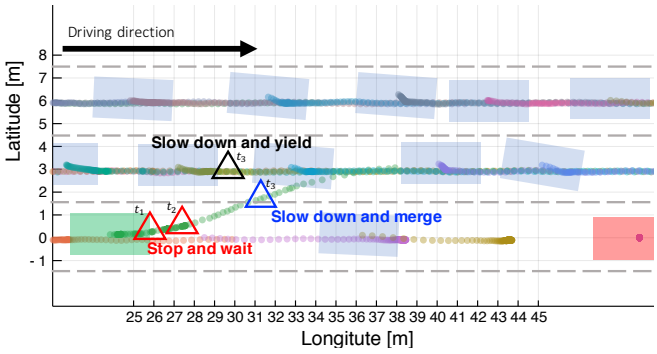

Fig. 7: Simulated position trajectories. A dot indicates a center coordinate of a vehicle at each time step and the color of dots distinguishes between trajectories of vehicles. A distance between two successive dots demonstrates how fast a vehicle moves. The triangle labeled with t_n highlights a center position at time t_n, $n = 1, \ldots, 3$. The rectangles illustrate the shape of the vehicles (colors are described in the caption of Fig. 6). The positions of the rectangles indicate the initial positions.

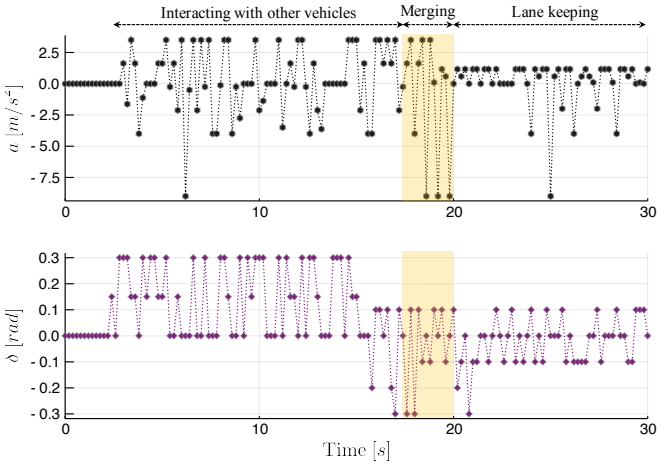

Fig. 8: Acceleration (top) and steering angle (bottom) profile of the ego vehicle. The ego vehicle initiates merging to a target lane at time 18.6$[s]$. The maximum braking is applied when no feasible solution is found.

to quickly get closer to the lane while avoiding collision. Between 18.6∼20 $[s]$, the ego vehicle changes lanes and the acceleration still fluctuates. This is again because the ego vehicle needs to move quickly, but must stop whenever another vehicle gets too close. Once the ego vehicle merges into the target lane, after 20$[s]$, the acceleration and steering angle fluctuate less, and the vehicle drives smoothly.

Note that the other vehicles may or may not be cooperative, depending on their characteristics. Therefore, we quantitatively validate the controller based on Monte Carlo simulations with various characteristics of other drivers. In each simulation, all vehicles are randomly positioned, except for the stopped vehicle (dead-end of the road), and the driver model parameters in Table II are sampled from uniform distributions. With respect to different cooperation levels, we consider three cases: (i) all drivers are cooperative (i.e. $\eta_{c,i} = 1$ for all drivers i in the scene); (ii) all drivers are aggressive (i.e. $\eta_{c,i} = 0$); and (iii) they are mixed (i.e. $\eta_{c,i} \in [0, 1]$). It is important to highlight that the ego vehicle

		Constant Velocity	SGAN	Ground Truth
Success rate	Coop.	96%	99%	99%
	Mix.	90%	97%	97%
	Agg.	48%	81%	89%
Mean time to merge	Coop.	25.95 (±6.42)	23.97 (±5.10)	24.53 (±5.11)
	Mix.	26.97 (±7.07)	25.76 (±5.37)	26.21 (±5.58)
	Agg.	35.24 (±6.12)	29.32 (±7.07)	28.31 (±6.22)
Mean min distance	Coop.	0.69 (±0.15)	0.49 (±0.18)	0.72 (±0.14)
	Mix.	0.58 (±0.19)	0.42 (±0.19)	0.61 (±0.17)
	Agg.	0.35 (±0.18)	0.30 (±0.14)	0.55 (±0.16)

TABLE IV: Monte Carlo simulation results (a total of 100 simulations each) for three motion prediction models, constant velocity, SGAN, and ground truth, for different level of cooperativeness. A trip is "successful" if the ego vehicle changes lanes to the target lane within the time limit (40 [s]). Until the time limit, the target lane is packed with vehicles and there is no empty space where the ego vehicle can overtake without cooperating with other drivers. The "time to merge" indicates the time period [s] that the ego vehicle takes to change lanes to the target lane. The "min distance" indicates the minimum distance [m] between the ego vehicle and other vehicles at any point of the simulation. The value in the parentheses shows the standard deviation.

does not know how cooperative the other drivers are. We found that when the drivers are all cooperative, changing lanes into the target lane within the time limit is more likely than when the drivers are either partially cooperative (i.e. mixed) or non-cooperative (i.e. aggressive), as shown in Table IV. Still, it is possible that the ego vehicle cannot change lanes within the time limit (i.e. the success rate is less than 100%) and ends up being stranded in the current lane. This is because the drivers may have short perception ranges, determined by η_p, which can result in the drivers not detecting the ego vehicle and keeping their speed, even though they are cooperative. Nonetheless, even when all drivers are aggressive, the controller can successfully change lanes in most cases. One caveat is that the ego vehicle can get contiguous to the other vehicles, shown from mean min distance in Table IV, especially when the other drivers are not being cooperative and pass the ego vehicle, although no collision was observed in any of the simulations.

We also compare the performance of the controller with SGAN to a simple motion prediction method: a constant velocity model. The constant velocity model predicts that a vehicle will maintain the same speed in the next time step. We also compare the SGAN-enabled controller to a controller with perfect predictions of other vehicles' motions, i.e. ground truth[2]. In general, we found that more accurate predictions lead to higher success rates. In fact, when all drivers are cooperative, all three prediction models can lead to successful lane change. That is, the imprecision of predictions on drivers' interactive motions is not critical when the drivers are very cooperative, since the drivers easily submit space to other vehicles, even with rough control inputs resulting from inaccurate motion predictions. This, however, is no longer valid if the drivers are aggressive. When they are aggressive, the ego vehicle needs a precise control that carefully induces cooperation from the other drivers. This cannot occur with the simple constant velocity model. Consequently, the success rate with SGAN is significantly higher (+33%) than that with the simple constant velocity model, when the drivers are aggressive. We also found that the controller with more accurate predictions tends to change lanes more quickly. That is, significant motion prediction errors for the other vehicles lead the ego vehicle to positions where it cannot effectively influence its neighboring vehicle, and create room for a safe lane change. As a consequence, the ego vehicle must let the coming vehicle pass and waits for next vehicle to cooperate. Finally, the minimum distances tend to be significantly larger with perfect predictions compared to that with SGAN. That is, more precise predictions help the controller secure a safer trajectory during lane change.

Additionally, in our recent paper [16], we compared the performance of the proposed controller with a learning-based controller, in terms of the success rate, time to merge, and minimum distances. The SGAN-enabled controller outperforms the learning-based controller in the success rate, (arguably) safety as measured by minimum distances, and reliability as measured by variances of performance metrics, while taking more time to merge. Interested readers are referred to [16] for detailed comparisons.

Finding a minimum distance between two vehicles *analytically* is computationally efficient, even though it is computed as the minimum distance among any pairs of circles. The circle model takes about $9 \cdot 10^{-6}$ $[s]$ on average to compute a minimum distance between two vehicles at any time. As a consequence, finding one control input at each time step takes less than 0.2 $[s]$, which suggests this approach is amenable to *real-time* control.

Simulation code and videos are available at `https://github.com/honda-research-institute/NNMPC.jl`.

E. Limitations and Future Work

No collision was observed in the simulation studies. However, the controller cannot guarantee zero collisions with other drivers, due to the absence of "safe" area where the ego vehicle can merge into without cooperating with other drivers. Still, improved predictions can help the ego vehicle keep a safe distance with other vehicles, as shown in Table IV. Improving the SGAN prediction accuracy by carefully designing the training set, loss functions, or network structures remains for future work.

The proposed algorithm somewhat naively generates the control candidates by random sampling, even though action spaces are specified in each driving scenario (keeping lane or changing lane). We can further reduce the actions spaces, using correlation between steering angle and acceleration, for each driving scenario. Also, the proposed algorithm (Algorithm 1) is heuristic. That is, even in an identical scenario, the controller may find a different solution. This might be acceptable in many cases where the constraints are

[2]In the case of perfect predictions for other vehicles' motion, the simulator propagates other vehicles' motion based on their driver model, and the propagated motions are used by the controller as predictions.

not violated. However, an advanced optimization algorithm can be further studied to find a locally optimal solution with a convergence guarantee.

The proposed controller design combines trajectory planning and controls, which can be susceptible to a model mismatch between the controller and system plant. A commonly-used two-stage planning and control framework [30] can be applied to mitigate the model mismatch issue, which remains for future work.

IV. Conclusion

This paper formalizes a control framework for autonomous lane changing in dense traffic. This paper particularly focuses on heavy traffic where vehicles cannot merge into a lane without cooperating with other drivers. The control framework incorporates a Recurrent Neural Network (RNN) architecture, namely a state-of-the-art Social Generative Adversarial Network (SGAN), to predict interactive motions of multiple drivers. The predicted motions are systematically evaluated in safety constraints to evaluate control inputs. A heuristic algorithm based on Monte Carlo simulation along with a roll-out approach is developed to find feasible solutions in a computationally efficient manner. The qualitative and quantitative analysis in the simulation studies illustrate the strong potential of the proposed control framework for achieving automatic and safe lane changes by cooperating with other drivers.

References

[1] S. Bae, Y. Kim, J. Guanetti, F. Borrelli, and S. Moura, "Design and implementation of ecological adaptive cruise control for autonomous driving with communication to traffic lights," in *American Control Conference (ACC)*, 2019.

[2] S. Bae, Y. Choi, Y. Kim, J. Guanetti, F. Borrelli, and S. Moura, "Real-time ecological velocity planning for plug-in hybrid vehicles with partial communication to traffic lights," in *Conference on Decision and Control (CDC)*, 2019.

[3] 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.

[4] M. P. Vitus and C. J. Tomlin, "A probabilistic approach to planning and control in autonomous urban driving," in *52nd IEEE Conference on Decision and Control*, pp. 2459–2464, IEEE, 2013.

[5] S. Ulbrich and M. Maurer, "Probabilistic online pomdp decision making for lane changes in fully automated driving," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pp. 2063–2067, IEEE, 2013.

[6] G. Schildbach and F. Borrelli, "Scenario model predictive control for lane change assistance on highways," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 611–616, IEEE, 2015.

[7] R. Chandra, Y. Selvaraj, M. Brännström, R. Kianfar, and N. Murgovski, "Safe autonomous lane changes in dense traffic," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2017.

[8] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011.

[9] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, "Cooperation-aware reinforcement learning for merging in dense traffic," *arXiv preprint arXiv:1906.11021*, 2019.

[10] 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.

[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] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions.," in *Robotics: Science and Systems*, Ann Arbor, MI, USA, 2016.

[13] Y. Hu, A. Nakhaei, M. Tomizuka, and K. Fujimura, "Interaction-aware decision making with adaptive strategies under merging scenarios," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019.

[14] A. Talebpour, H. S. Mahmassani, and S. H. Hamdar, "Modeling lane-changing behavior in a connected environment: A game theory approach," *Transportation Research Part C: Emerging Technologies*, vol. 59, pp. 216–232, 2015.

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

[16] D. M. Saxena, S. Bae, A. Nakhaei, K. Fujimura, and M. Likhachev, "Driving in dense traffic with model-free reinforcement learning," *arXiv preprint arXiv:1909.06710*, 2019.

[17] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[18] D. Isele, A. Nakhaei, and K. Fujimura, "Safe reinforcement learning on autonomous vehicles," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[19] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th international conference on software engineering*, pp. 303–314, ACM, 2018.

[20] 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.

[21] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. Jackel, and U. Muller, "Explaining how a deep neural network trained with end-to-end learning steers a car," *arXiv preprint arXiv:1704.07911*, 2017.

[22] C. Choi, A. Patil, and S. Malla, "Drogon: A causal reasoning framework for future trajectory forecast," *arXiv preprint arXiv:1908.00024*, 2019.

[23] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[24] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2255–2264, 2018.

[25] J. C. Doyle, B. A. Francis, and A. R. Tannenbaum, *Feedback control theory*. Courier Corporation, 2013.

[26] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2015-August, pp. 1094–1099, 2015.

[27] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.

[28] A. Kesting, M. Treiber, and D. Helbing, "General lane-changing model mobil for car-following models," *Transportation Research Record*, vol. 1999, no. 1, pp. 86–94, 2007.

[29] S. I. S. Laboratory, "Automotivedrivingmodels.jl." https://github.com/sisl/AutomotiveDrivingModels.jl, 2019.

[30] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 812–818, IEEE, 2017.