Chance-Aware Lane Change with High-Level Model Predictive Control Through Curriculum Reinforcement Learning

Yubin Wang, Yulin Li, Hakim Ghazzai, Yehia Massoud, *Fellow, IEEE,* and Jun Ma

***Abstract*— Lane change in dense traffic is considered a challenging problem that typically requires the recognization of an opportune and appropriate time for maneuvers. In this work, we propose a chance-aware lane-change strategy with high-level model predictive control (MPC) through curriculum reinforcement learning (CRL). The embodied high-level MPC in our proposed framework is parameterized with augmented decision variables, where full-state references and regulatory factors concerning their importance are introduced. In this sense, improved adaptiveness to dense and dynamic environments with high complexity is exhibited. Furthermore, to improve the convergence speed and ensure a high-quality policy, effective curriculum design is integrated into the reinforcement learning (RL) framework with policy transfer and enhancement. With comprehensive experiments towards the chance-aware lane-change scenario, accelerated convergence speed and improved reward performance are demonstrated through comparisons with representative baseline methods. It is noteworthy that, given a narrow chance in the dense and dynamic traffic flow, the proposed approach generates high-quality lane-change maneuvers such that the vehicle merges into the traffic flow with a high success rate.**

I. Introduction

As a representative challenging scenario in autonomous driving, chance-aware lane change aims to control the ego vehicle to dynamically track the recognized narrow chance with an appropriate safe margin, such that the vehicle merges into the traffic flow safely. However, it is still an open and challenging problem on how to generate a satisfactory trajectory and feasible lane-change maneuvers for the chance-aware lane-change task. First, frequent trajectory re-planning is demanded in a highly dynamic environment where other surrounding vehicles are driving at high speed. Also, the time window needs to be identified dynamically for the execution of lane-change maneuvers in the planning horizon. Moreover, the inevitable existence of uncertainty resulting from surrounding vehicles' maneuvers poses great challenges to ensure the driving safety. To tackle these challenging problems, it remains to develop a motion planner with strong adaptiveness in such a dense and dynamic environment.

The literature on motion planning for autonomous driving can be mainly divided into two categories: the model-based approach and the model-free approach. As one of the widely used model-based motion planning approaches, model predictive control (MPC) has gained wide popularity for its effectiveness to optimize the trajectory while dealing with various constraints [1]–[3]. However, in complex scenarios such as the aforementioned chance-aware lane-change problem, it is nearly impossible to properly set all the handcrafted factors of MPC, e.g., time-varying constraints corresponding to the dynamic identification of time window in the stated chance-aware lane-change scenario. In this sense, the efficiency of solving the MPC problem is diminished, and solution quality could be significantly degraded. On the other hand, the model-free method has shown promising results in generating flexible maneuvers for autonomous driving [4]–[7]. In particular, reinforcement learning (RL) aims to find a policy that maximizes the designed reward through interactions with the environment. Generally, the learned optimal policy can be applied in real-time situations directly by mapping the current observations to actions [8], [9]. However, it is a common problem that pure RL-based methods suffer from the stability of control policy as well as the applicability in real-world applications due to the low error tolerance.

In this paper, we propose a novel learning-based MPC framework for chance-aware lane-change scenarios, where the augmented decision variables are designed to parameterize the MPC. Specifically, we learn full states as references of MPC and also their regulatory factors which can automatically modulate the relative importance of references within the planning horizon. Hence, the adaptiveness to dense and dynamic environments is enhanced. To deal with the reward sparsity issue and further improve the training efficiency, we incorporate curriculum reinforcement learning (CRL) with policy transfer and enhancement to learn the optimal policy progressively with ordered curricula.

The contributions of this paper are listed as follows:

- A novel learning-based MPC framework is proposed with the parameterization of MPC using augmented decision variables, where full-state references and regulatory factors modulating their importance in the cost function are incorporated. Then, all crucial information for optimizing the performance of MPC is extracted and adjusted effectively, which leads to improved adaptiveness to dense and dynamic environments.
- To improve the effectiveness of the learned high-level policy and avoid unstable learning, we utilize CRL with policy transfer and enhancement to obtain a policy with faster convergence and higher reward.
- The proposed approach is deployed in the chance-aware lane-change task under a dense and highly dynamic

Yubin Wang, Yulin Li, and Jun Ma are with the Robotics and Autonomous Systems Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China.

Hakim Ghazzai and Yehia Massoud are with the Division of Computer, Electrical and Mathematical Science and Engineering, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia.

All correspondence should be sent to Jun Ma (e-mail: jun.ma@ust.hk).

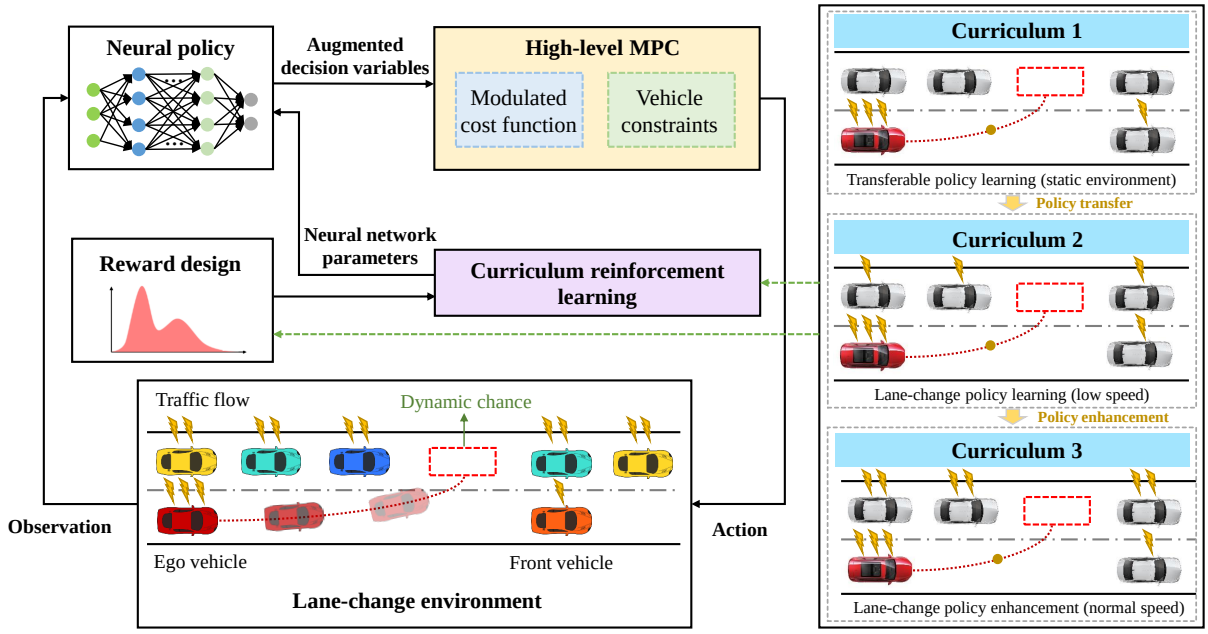


Fig. 1. Overview of our proposed approach for chance-aware lane-change problems. The different speed levels of vehicles are indicated by the number of lightning bolts.

environment, where improved safety and effectiveness of our proposed framework are demonstrated through various comparative experiments.

II. Related Works

For learning-based MPC, critical factors in MPC are generally parameterized by classical machine learning models [10], [11] or deep neural networks [12], [13]. A typical routine for parameterizing essential elements in optimal control problems relies on learning the system model [14]–[16] or the high-level parameters for the low-level optimal controllers [17]–[19]. With such an integrated learning and control framework, flexible maneuvers can be generated to deal with complex motion planning problems. Gaussian processes are utilized to enhance a simple nominal model in [10] for race cars, which improves the solution quality of MPC when dealing with autonomous racing problems. Similarly, in [11], Gaussian processes are introduced to learn the nonlinearities of the system model in the presence of uncertainty, and thus the computational burden of MPC is alleviated. Furthermore, in [12], a learned value function is used for approximating the terminal cost of MPC, which reduces the planning horizon and allows for better policies beyond local solutions. In [13], RL is exploited to learn the cost function for MPC from scratch without human intervention, and the value function is approximated with given sparse or even binary objectives.

Furthermore, the integration of MPC and RL has been explored extensively for motion planning tasks owing to its outstanding flexibility and adaptiveness to challenging problems. As one of the seminal works in this area, a Gaussian distribution is utilized to model the high-level policy in [17], [18], with which the traversal time is defined as a decision variable, such that the MPC can be parameterized. Hence, the quadrotor accomplishes the task to fly through a swinging gate. However, the state references of MPC are fixed as the gate states, which could degrade the performance of agile flight. Additionally, a deep SE(3) motion planning approach is presented in [19], where SE(3) decision variables are designed as the references of the MPC. Besides, an innovative reinforce-imitate learning framework is developed to learn a neural policy stably. Essentially, this approach manifests the effectiveness in traversing a moving gate. Nevertheless, the weight modulation under the MPC architecture is empirical, and the binary search algorithm (which is used to approximate the optimal traversal time) hinders the generalization of the method to dense and dynamic environments with higher complexity.

III. Problem Statement

A. Vehicle Dynamic Model

The bicycle model in [20] is used in this work, where the state vector of the vehicle is defined as $\mathbf{x} = \begin{bmatrix} p_x & p_y & \varphi & v_x & v_y & \omega \end{bmatrix}^\top$. Note that p_x and p_y denote the X-coordinate and Y-coordinate position of the vehicle's center of mass, ϕ is the heading angle, v_x and v_y are the longitudinal speed and lateral speed, and ω represents the yaw angular velocity. Also, we integrate the actions into a vector as $\mathbf{u} = \begin{bmatrix} a & \delta \end{bmatrix}^\top$, where a and δ are the acceleration and steering angle. Subsequently, with $L_k = l_f k_f - l_r k_r$ and a sampling time d_t, the nonlinear dynamic model f_{dyn} of the

vehicle in continuous time is given by:

$$\dot{\mathbf{x}} = f_{\text{dyn}}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} v_x \cos\varphi - v_y \sin\varphi \\ v_x \sin\varphi + v_y \cos\varphi \\ \omega \\ a \\ \frac{L_k\omega - k_f\delta v_x - m v_x^2\omega + v_y(k_f+k_r)}{m v_x - d_t(k_f+k_r)} \\ \frac{L_k v_y - L_f k_f \delta v_x + \omega(l_f^2 k_f + l_r^2 k_r)}{I_z v_x - d_t(l_f^2 k_f + l_r^2 k_r)} \end{bmatrix}, \quad (1)$$

where m is the mass of the vehicle, l_f and l_r are the distance from the center of mass to the front and rear axle, k_f and k_r are the cornering stiffness of the front and rear wheels, I_z is the polar moment of inertia.

B. Chance-Aware Lane-Change Task

In this work, we assume that each vehicle in the traffic flow keeps the same longitudinal speed and zero lateral speed, and the ego vehicle is blocked by the front vehicle that moves slowly in the same lane. Under this setting, the recognized dynamic chance for the ego vehicle to perform lane change is placed as the center of the gap of the traffic flow. Hence, the control objective of the chance-aware lane-change mission is to control the ego vehicle such that it merges into the traffic flow at an opportune and appropriate time in the time window (which occurs exactly before the ego vehicle collides with the front vehicle).

Mathematically, with the given sampling time d_t and vehicle dynamic model f_{dyn}, a sequence of vehicle states $\mathbf{x}_k, \forall k \in [0, 1, \cdots, N]$ and control commands $\mathbf{u}_k, \forall k \in [0, 1, \cdots, N-1]$ are discretized. Let $\mathbf{x}_g$ denote the vehicle goal state after merging into the traffic flow, the control objective of the chance-aware lane-change task is formulated to generate the optimal state trajectory $\mathbf{x}_k^*, \forall k \in [0, 1, \cdots, N]$ towards $\mathbf{x}_g$ and a list of optimal control commands $\mathbf{u}_k^*, \forall k \in [0, 1, \cdots, N-1]$ over a receding horizon N.

IV. High-Level Model Predictive Control with Augmented Decision Variables

A. MPC Problem Formulation

To solve the chance-aware lane-change problem, we formulate a nonlinear MPC problem with augmented decision variables, which leads to the following nonlinear optimization problem to be solved over the receding horizon N:

$$\begin{aligned} \min_{\mathbf{x}_{0:N}, \mathbf{u}_{0:N-1}} \; & J_{x_N} + \sum_{k=0}^{N-1} \left(J_{x_k} + J_{u_k} + J_{\Delta u_k} + J_{\text{tra},k}\right) \\ & = \boldsymbol{\delta}_{x,N}^T \mathbf{Q}_x \boldsymbol{\delta}_{x,N} + \sum_{k=0}^{N-1} (\boldsymbol{\delta}_{x,k}^T \mathbf{Q}_x \boldsymbol{\delta}_{x,k} + \boldsymbol{\delta}_{u,k}^T \mathbf{Q}_u \boldsymbol{\delta}_{u,k} \\ & + \Delta_{u,k}^T \mathbf{Q}_{\Delta_u} \Delta_{u,k} + \boldsymbol{\delta}_{\text{tra},k}^T \mathbf{Q}_{\text{tra}}(t_{\text{tra}}, k)\, \boldsymbol{\delta}_{\text{tra},k}) \\ \text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{x}_k + f_{\text{dyn}}(\mathbf{x}_k, \mathbf{u}_k)\, d_t, \\ & p_{y,\min} \le p_y \le p_{y,\max}, \\ & a_{\min} \le a \le a_{\max}, \\ & -\delta_{\max} \le \delta \le \delta_{\max}, \end{aligned} \quad (2)$$

where $\boldsymbol{\delta}_{x,k} = (\mathbf{x}_k - \mathbf{x}_g)$ denotes the difference between the vehicle's states $\mathbf{x}_k$ and goal state $\mathbf{x}_g$, $\boldsymbol{\delta}_{\text{tra},k} = (\mathbf{x}_k - \mathbf{x}_{\text{tra}})$ represents the difference between the vehicle's states $\mathbf{x}_k$ and learned full-state references $\mathbf{x}_{\text{tra}}$, $\boldsymbol{\delta}_{u,k} = \mathbf{u}_k$ is a regularization term for predicted control commands $\mathbf{u}_k$, $\boldsymbol{\Delta}_{u,k} = (\mathbf{u}_k - \mathbf{u}_{k-1})$ is another regularization term considering the variation of control commands for passenger's comfort. The Y-coordinate position is bounded within $p_{y,\min}$ and $p_{y,\max}$ due to the structural constraint introduced by the lane size. Additionally, control commands are constrained by $a_{\min}, a_{\max}, \delta_{\max}$ considering the physical limits of vehicle dynamics.

The quadratic cost terms in (2) are weighted by positive semi-definite diagonal matrices $\mathbf{Q}_x, \mathbf{Q}_{\text{tra}}(t_{\text{tra}}, k), \mathbf{Q}_u$, and $\mathbf{Q}_{\Delta_u}$. Moreover, $\mathbf{Q}_x, \mathbf{Q}_u$, and $\mathbf{Q}_{\Delta_u}$ are time-invariant matrices, where $\mathbf{Q}_x$ defines the importance of reaching the goal state $\mathbf{x}_g$ at the end of the receding horizon, $\mathbf{Q}_u$ represents the weight that penalizes the control commands, and $\mathbf{Q}_{\Delta_u}$ corresponds to the importance of taking the control commands that do not deviate too much from the control command in the previous step.

B. MPC Parameterization with Augmented Decision Variables

We introduce the full-state references as a part of the decision variables of MPC:

$$\mathbf{x}_{\text{tra}} = [p_{x,\text{tra}}, p_{y,\text{tra}}, \psi_{\text{tra}}, v_{x,\text{tra}}, v_{y,\text{tra}}, \omega_{\text{tra}}]^\top. \quad (3)$$

Here, $\mathbf{x}_{\text{tra}}$ can be deemed as intermediate states during lane change.

To endow the vehicle with the ability to automatically balance the importance between tracking learned intermediate-state references and tracking goal states, a weighting matrix with adaptive adjustment $\mathbf{Q}_{\text{tra}}(t_{\text{tra}}, k)$ is defined as:

$$\mathbf{Q}_{\text{tra}}(t_{\text{tra}}, k) = \mathbf{Q}_{\max} \exp\left(-\gamma \left(k d_t - t_{\text{tra}}\right)^2\right), \quad (4)$$

where $\mathbf{Q}_{\max}$ is learned maximum of $\mathbf{Q}_{\text{tra}}$, $\gamma \in \mathbb{R}_+$ is the exponential decay rate for costs in terms of tracking learned references, and t_{tra} is learned tracking time reference. In order to further modulate the relative importance of each decision variable respectively, $\mathbf{Q}_{\max}$ is defined as:

$$\begin{aligned} \mathbf{Q}_{\max} = \text{diag}(&Q_{p_{x,\text{tra}}}, Q_{p_{y,\text{tra}}}, Q_{\psi_{\text{tra}}}, Q_{v_{x,\text{tra}}}, \\ &Q_{v_{y,\text{tra}}}, Q_{\omega_{\text{tra}}}). \end{aligned} \quad (5)$$

Hence, the regulatory factors $\mathbf{Q}_{\max}$ and t_{tra} collaboratively modulate the cost function of MPC, which adaptively balance the importance of tracking learned state references and tracking the goal states. Thus, the proposed MPC scheme with the modulated cost function exhibits better adaptiveness to dense and highly dynamic environments.

We integrate all augmented decision variables to a decision vector $\mathbf{z}$, which is defined as:

$$\begin{aligned} \mathbf{z} = [&p_{x,\text{tra}}, p_{y,\text{tra}}, \psi_{\text{tra}}, v_{x,\text{tra}}, v_{y,\text{tra}}, \omega_{\text{tra}}, Q_{p_{x,\text{tra}}}, \\ &Q_{p_{y,\text{tra}}}, Q_{\psi_{\text{tra}}}, Q_{v_{x,\text{tra}}}, Q_{v_{y,\text{tra}}}, Q_{\omega_{\text{tra}}}, t_{\text{tra}}]^\top \in \mathbb{R}^{13}. \end{aligned} \quad (6)$$

In this sense, the MPC is parameterized. Here, we define f_{MPC} as the mapping function of MPC. By feeding MPC

Algorithm 1: High-Level MPC with Augmented Decision Variables

Input: $\mathbf{z}$, f_{MPC}
Output: $\boldsymbol{\xi}^*(\mathbf{z})$
Initialize J_{x_k}, J_{u_k}, and $J_{\Delta u_k}$;
Formulate $\boldsymbol{\delta}_{\mathrm{tra}}$ with $\mathbf{x}_{\mathrm{tra}}$;
Modulate $\mathbf{Q}_{\mathrm{tra}}$ with $\mathbf{Q}_{\max}$ and t_{tra};
Compute J_{tra} ;
Solve (2) to get $\boldsymbol{\xi}^*(\mathbf{z}) = f_{\mathrm{MPC}}(\mathbf{z})$

with different decision vectors $\mathbf{z}$, different corresponding optimal state trajectories are generated, which are denoted as $\boldsymbol{\xi}^*(\mathbf{z}) = f_{\mathrm{MPC}}(\mathbf{z})$ (i.e., $\boldsymbol{\xi}^*(\mathbf{z}) = \{\mathbf{x}_k^*(\mathbf{z})\}_{k=0}^N$). It is pertinent to note that the trajectory generation can be executed according to Algorithm 1. Hence, we can incorporate episodic RL technique for policy search [17]–[19] to find a policy $\pi^*(\boldsymbol{\theta})$, such that it automatically models the augmented decision variables.

V. Curriculum Reinforcement Learning with Policy Transfer and Enhancement

A. Policy Representation

We present a novel CRL framework for policy search, such that the optimal policy $\pi^*(\boldsymbol{\theta})$ can be determined. Utilizing a deep neural network to parameterize the policy, the augmented decision vector $\mathbf{z}$ is modeled as:

$$\mathbf{z} = f_{\boldsymbol{\theta}}(\mathbf{o}), \tag{7}$$

where $\boldsymbol{\theta}$ are the parameters of the deep neural network, $\mathbf{o}$ is the observation of the vehicle from the environment.

B. Observation

With the assumption of global observation, i.e., the states of all vehicles on the road can be obtained by the ego vehicle, the observation vector of the ego vehicle is defined as follows (which is fed into the deep neural network as the input):

$$\begin{aligned}\mathbf{o} = [&p_{x,\mathrm{init}}, p_{y,\mathrm{init}}, \psi_{\mathrm{init}}, v_{x,\mathrm{init}}, p^{\mathrm{c}}_{x,\mathrm{init}}, p^{\mathrm{c}}_{y,\mathrm{init}}, v^{\mathrm{c}}_{x,\mathrm{init}},\\ &p^{\mathrm{f}}_{x,\mathrm{init}}, p^{\mathrm{f}}_{y,\mathrm{init}}, v^{\mathrm{f}}_{x,\mathrm{init}}]^\top \in \mathbb{R}^{10},\end{aligned} \tag{8}$$

where $p_{x,\mathrm{init}}$, $p_{y,\mathrm{init}}$, $p^{\mathrm{c}}_{x,\mathrm{init}}$, $p^{\mathrm{c}}_{y,\mathrm{init}}$, $p^{\mathrm{f}}_{x,\mathrm{init}}$, and $p^{\mathrm{f}}_{y,\mathrm{init}}$ represent the initial X-coordinate and Y-coordinate position of the ego vehicle, recognized dynamic chance and front vehicle, respectively. ψ_{init} is the initial heading angle of the ego vehicle, $v_{x,\mathrm{init}}$, $v^{\mathrm{c}}_{x,\mathrm{init}}$, and $v^{\mathrm{f}}_{x,\mathrm{init}}$ denote the initial longitudinal speed of the ego vehicle, dynamic chance and front vehicle, respectively.

Moreover, input normalization plays an essential role in most machine learning algorithms as different scales of features easily lead to unstable learning [9]. Hence, we apply z-score normalization to all input features in the observation vector.

C. Multi-Task Reward Formulation

Sparse Lane-Change Reward: The sparse reward function evaluating the quality of the optimal state trajectory $\boldsymbol{\xi}^*(\mathbf{z})$ for the lane-change task is designed as:

$$R_{\mathrm{LC}}(\boldsymbol{\xi}^*(\mathbf{z})) = R_{\max} - c_c \sum_{k=0}^{N} \rho_c \left|\boldsymbol{v}_k\right|^2, \tag{9}$$

where ρ_c is a binary flag indicating whether the collision with surrounding vehicles occurs, $c_c \in \mathbb{R}_+$ is a hyperparameter for weighting the vehicle collision penalty, and $R_{\max} \in \mathbb{R}_+$ is goal reward, which is gained when the ego vehicle merges into the traffic flow successfully.

Reward Shaping: In most existing RL algorithms, well-designed reward or fine-grain feedback are essential for learning success, as the lack of them leads to failure for learning an acceptable policy in a reasonable time frame [21]. Thus, it is challenging to teach the RL agent to explore the policy space efficiently with a sparse lane-change reward. Hence, with reward shaping, we introduce a novel reward term to directly evaluate the decision variables, which guides the RL agent to explore the policy space in directions of larger policy gradients:

$$\begin{aligned}&R_{\mathrm{DV}}(\mathbf{z}) = \\ &- c_x \left|\Delta p_x\right| - c_y \rho_y \Delta p_y - c_t \left|t\right| - c_{\Delta t} \rho_{\Delta t} \Delta t - c_\psi \rho_\psi \Delta\psi \\ &- c_{p_x} \rho_{p_x} \left|Q_{p_{x,\mathrm{tra}}}\right| - c_{p_y} \rho_{p_y} \left|Q_{p_{y,\mathrm{tra}}}\right| - c_\psi \rho_\psi \left|Q_{\psi_{\mathrm{tra}}}\right| \\ &- c_{v_x} \rho_{v_x} \left|Q_{v_{x,\mathrm{tra}}}\right| - c_{v_y} \rho_{v_y} \left|Q_{v_{y,\mathrm{tra}}}\right| - c_\omega \rho_\omega \left|Q_{\omega_{\mathrm{tra}}}\right|\end{aligned} \tag{10}$$

where

$$\begin{aligned}\Delta p_x &= p_{x,\mathrm{tra}} - p^{\mathrm{c}}_{x,\mathrm{tra}},\\ \Delta p_y &= \min(\left|p_{y,\mathrm{tra}} - p_{y,\max}\right|, \left|p_{y,\mathrm{tra}} - p_{y,\min}\right|),\\ \Delta t &= \min(\left|t_{\mathrm{tra}} - t_{\max}\right|, \left|t_{\mathrm{tra}} - t_{\min}\right|),\\ \Delta\psi &= \min(\left|\psi_{\mathrm{tra}} - \psi_{\max}\right|, \left|\psi_{\mathrm{tra}} - \psi_{\min}\right|),\end{aligned}$$

t denotes the current time in episode of experiments, $c_x, c_y, c_t, c_{\Delta t}, c_\psi, c_{p_x}, c_{p_y}, c_\psi, c_{v_x}, c_{v_y}$, and c_ω are weighting coefficients, $\rho_y, \rho_{\Delta t}$, and ρ_ψ are binary flags indicating whether the learned position reference of Y-coordinate is out of $p_{y,\min}$ and $p_{y,\max}$, whether the learned time reference is infeasible (i.e., larger than the duration of simulation or smaller than zero), and whether the learned heading angle exceeds the feasible heading range, respectively. $\rho_{p_x}, \rho_{p_y}, \rho_\psi, \rho_{v_x}, \rho_{v_y}, \rho_\omega$ are also binary flags implying whether the learned maximum of weighting matrix is smaller than zero. $p^{\mathrm{c}}_{x,\mathrm{tra}}$ is the X-coordinate position of the recognized dynamic chance at t_{tra}. $t_{\min}$ and $t_{\max}$ represent the lower bound and upper bound of the duration of the simulation. $\psi_{\min}$ and $\psi_{\max}$ are the lower bound and the upper bound of the predefined feasible heading angle range. Particularly, $|\Delta p_x|$ and $|t|$ are empirical rewards encouraging the RL agent to explore the policy space along the directions as expected.

D. Curriculum Reinforcement Learning with Policy Transfer and Enhancement

With the episodic RL setting, we operate the neural policy at the beginning of each episode to model $\mathbf{z}$, with which the MPC can generate a sequence of optimal trajectories $\boldsymbol{\xi}^*(\mathbf{z})$. Therefore, the corresponding reward signal $R(\boldsymbol{\xi}^*(\mathbf{z}))$ is received to evaluate the quality of generated trajectories. Hence, the problem of finding the optimal neural policy with RL is reformulated to the following reward maximization problem [19]:

$$\begin{aligned} \max_{\boldsymbol{\theta}} \quad & R\left(\boldsymbol{\xi}^*\left(\mathbf{z}\left(\boldsymbol{\theta}\right)\right)\right) \\ \text{s.t.} \quad & \boldsymbol{\xi}^*\left(\mathbf{z}\left(\boldsymbol{\theta}\right)\right) = f_{\mathrm{MPC}}\left(\mathbf{z}\left(\boldsymbol{\theta}\right)\right). \end{aligned} \tag{11}$$

The gradient of the episode reward R with respect to neural network parameters $\boldsymbol{\theta}$ is decomposed with chain rule, where

$$\frac{dR}{d\boldsymbol{\theta}} = \frac{\partial R}{\partial \mathbf{z}} \frac{\partial \mathbf{z}}{\partial \boldsymbol{\theta}}. \tag{12}$$

Note that the sub-gradient $\boldsymbol{g_{s2}} = \partial \mathbf{z}/\partial \boldsymbol{\theta}$ is calculated automatically using standard machine learning tools when updating the neural network. However, computing another sub-gradient $\boldsymbol{g_{s1}} = \partial R/\partial \mathbf{z}$ is computationally expensive as the computation requires differentiation through the nonlinear MPC optimization problem over a long receding horizon [19]. We denote the value in the jth row in $\boldsymbol{g_{s1}}$ as g_j. With the finite difference policy gradient method, we can estimate g_j as:

$$g_j = \frac{R\left(\boldsymbol{\xi}^*\left(\mathbf{z} + \epsilon \boldsymbol{e_j}\right)\right) - R\left(\boldsymbol{\xi}^*\left(\mathbf{z}\right)\right)}{\epsilon}, \tag{13}$$

where $\boldsymbol{e_j} = [0, \cdots, 0, 1, 0, \cdots, 0]^\top$ is a unit vector with only 1 in jth row, ϵ is a small random step in the direction $\boldsymbol{e_j}$ to perturb the augmented decision vector $\mathbf{z}$.

We utilize on-policy RL to train a neural network policy through the maximization of the designed reward. However, the reward is not directly corresponding to the output of neural networks in this hierarchical implementation, i.e., terrible decision variables possibly lead to an acceptable reward owing to the robustness of MPC. Besides, with the sparse lane-change reward, the exploration effectiveness of our neural policy is not guaranteed, so we cannot obtain an acceptable policy in finite time. Hence, it is tough for the RL agent to recognize the optimal direction from the current policy to the globally optimal policy. As a result, the training of the neural network with RL suffers from inefficient exploration in policy space and even unstable learning.

Curriculum learning (CL) is a well-established routine to accelerate the exploration when RL is handling rather complex missions, especially addressing extrapolation error and premature policy convergence [9]. To address the challenging problem resulting from the sparse reward setting with the hierarchical implementation framework, we present a CRL framework to train our neural policy with policy transfer and policy enhancement. We generate three modes of curricula, which are represented by the set $\mathscr{C} = \{C_i\}, i = \{1, 2, 3\}$.

Algorithm 2: Curriculum Reinforcement Learning with Policy Transfer and Enhancement

Input: f_{MPC}, $\mathscr{C}$
Output: $\pi^* = f_{\boldsymbol{\theta}^*}$

```
Initialize the parameters θ of deep neural network;
while not terminated do
    Select Curriculum C_i from Curricula 𝒞;
    Reset the environment to get x_g and o according
      to C_i;
    if curriculum switched then
        Load policy π* trained with C_{i−1};
    end
    Compute z = f_θ(o) ;
    Solve MPC(z) as (2) online to obtain ξ*(z) ;
    for j ← 1 to dim(z) do
        Perturb jth row in z;
        Estimate g_j using (13);
    end
    Update θ using gradient ascent with g_{s1};
end
```

The curricula are designed with different tasks in different domains.

Curriculum 1: Transferable Policy Learning with Reward Shaping in Static Environment. In source domain 1 which is denoted as $\mathcal{M}_{s,1}$, the traffic flow keeps static. The objective of Curriculum 1 is to learn a transferable neural policy. We train our randomly initialized neural network with the maximization of the reward term directly evaluating the decision variables as (10). In $\mathcal{M}_{s,1}$, the task $\mathcal{T}_{s,1}$ is to train an acceptable neural policy $\pi_{s,1}$ in finite time, with which the learned state references $\mathbf{x}_{\mathrm{tra}}$, maximum of weighting matrix $\mathbf{Q}_{\max}$, and tracking time reference t_{tra} are expected to converge to a feasible range roughly. With this formulation, the RL agent is guided with empirical reward design, which incredibly decreases the exploration time in policy space.

Curriculum 2: Lane-Change Policy Learning Under Low-Speed Setting. In Curriculum 2, we load the transferable policy $\pi_{s,1}$ and train it in source domain $\mathcal{M}_{s,2}$, where the traffic flow moves at a speed lower than the normal setting. The objective of Curriculum 2 is to generalize the transferable $\pi_{s,1}$ to a lane-change policy $\pi_{s,2}$ by maximizing the lane-change reward as (9).

Curriculum 3: Lane-Change Policy Enhancement Under Normal-Speed Setting. In Curriculum 3, we aim to obtain an optimal neural policy π^* for the lane-change task under a normal speed setting. In the target domain $\mathcal{M}_t$, the traffic flow moves at a normal speed and the weight of the penalty term in terms of collision in lane-change reward is increased. We load and enhance the lane-change policy $\pi_{s,2}$, and obtain the optimal policy π^* eventually by maximizing the modulated lane-change reward.

To this end, the proposed CRL framework with policy transfer and enhancement is summarized in Algorithm 2.

VI. Experiments

A. Implementation Setup

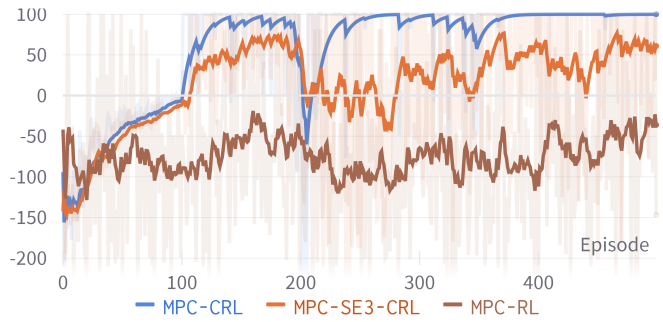
The deep neural network is constructed in PyTorch [22] and trained with Adam optimizer [23]. Weights and Biases [24] is utilized to monitor the training process. The MPC problem is solved using CasADi [25] with ipopt option via single-shooting method. In MPC, we set the receding horizon to $T = 5.0\,\mathrm{s}$ and the sampling time to $d_t = 0.1\,\mathrm{s}$. The hyperparameters for network training are listed in Table I.

TABLE I
HYPERPARAMETER SETTINGS.

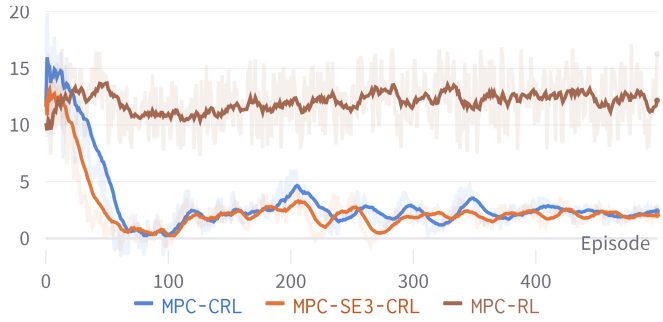
Hyperparameter	Value
Neural network structure (MLP)	4 × [128, Leaky ReLU]
Initial learning rate	3×10^{-4}
Learning rate decay step	32
Learning rate decay rate	0.96

B. Training Results

To illustrate the effectiveness of the proposed method (denoted as MPC-CRL), comparative experiments with learning-based baselines are conducted. In particular, we exploit high-level MPC with augmented decision variables with vanilla RL (denoted as MPC-RL) and high-level MPC with SE(3) decision variables and CRL (denoted as MPC-SE3-CRL) as our baselines.



(a) Reward



(b) Learned tracking time reference

Fig. 2. Learning curve comparison in terms of reward and learned tracking time reference of different methods. The training curves are smoothed by exponential moving average with a degree of 0.8, and the curriculum is switched at episodes 100 and 200.

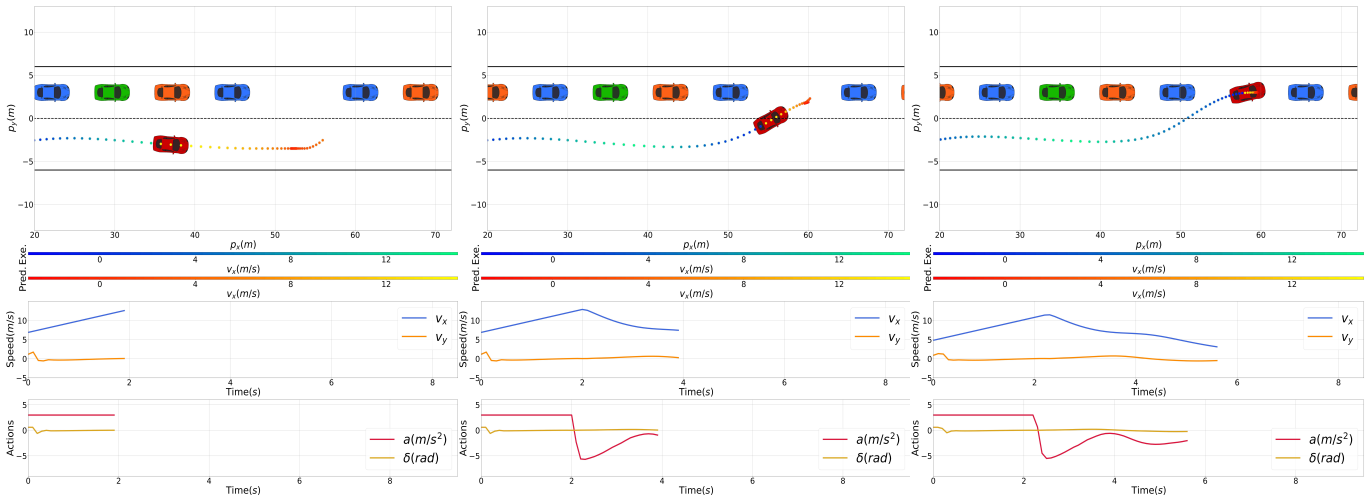
The learning curves of reward are illustrated in Fig. 2(a) for comparison. It is observed that MPC-CRL outperforms MPC-RL in terms of convergence speed and reward performance. Additionally, the tracking time reference in decision variables is a critical factor that directly indicates the quality of the learned neural policy (because inappropriate tracking time reference is prone to trap the RL agent into the local optimum). Hence, we record the learning curves of learned tracking time as Fig. 2(b). With our presented learning framework, the learned time reference converges to a feasible range rapidly in finite time, outperforming that of vanilla RL (which is trapped into the local optimum). Besides, with augmented decision variables adaptively modulating the cost function in the MPC, MPC-CRL outperforms the baseline MPC-SE3-CRL in view of reward performance. As indicated from the training results, it can be concluded that the presented CRL framework stimulates the RL agent to efficiently explore the policy space and reach the global optimum; and augmented decision variables improve the lane-change performance significantly.

C. Performance Evaluation

To further demonstrate the advantages of MPC-CRL in the lane-change task, a series of validation trails and detailed analyses are demonstrated in Fig. 3. As shown in Fig. 3(a), the ego vehicle takes the maximum of bounded acceleration when tracking the learned state references at $t = 1.9\,\mathrm{s}$. As the neural policy is trained with collision penalty term, the X-coordinate position of the learned state references is larger than that of the center of recognized chance when $t < t_{\mathrm{tra}}$, empirically saving appropriate safe margin for lane change. Then as shown in Fig. 3(b), the ego vehicle slows down to 7.5 m/s approximately to turn left for lane-change preparation when $t = 3.9\,\mathrm{s}$. Finally, it successfully merges into the traffic flow with a satisfactory safe margin at $t = 5.5\,\mathrm{s}$, as shown in Fig. 3(c).

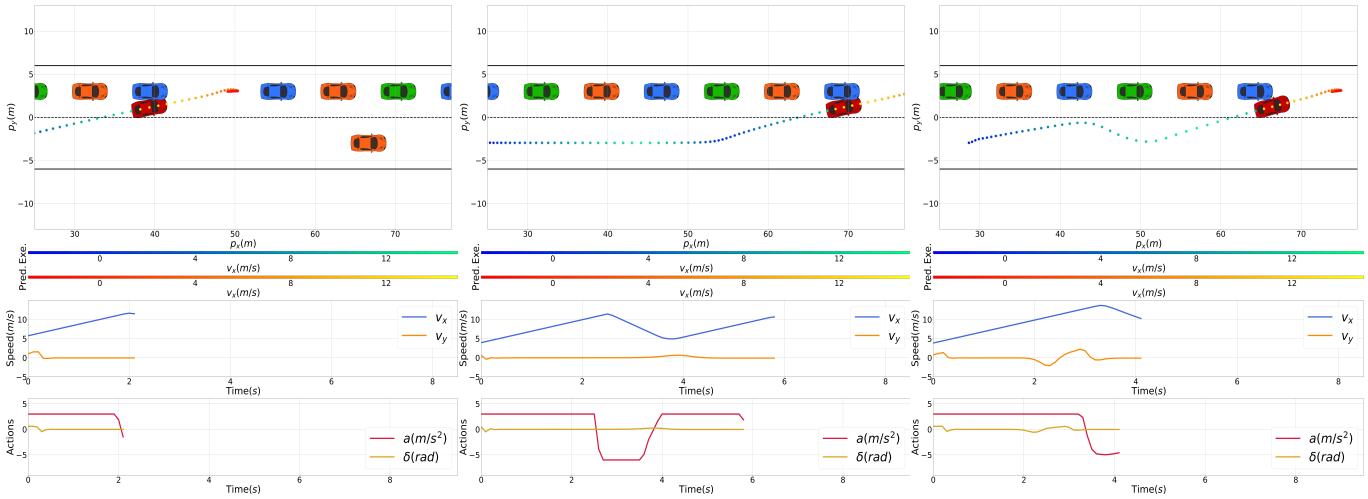
We also evaluate the performance with all aforementioned baselines as well as a new baseline. Performance analyses in different trails of all baselines are shown in Fig. 4. As shown in Fig. 4(a), the ego vehicle rushes towards the dynamic chance and collides with the traffic flow when $t = 2.1\,\mathrm{s}$. MPC-RL leads to a failure in the lane-change task, as the neural policy is unacceptable after finite episode training with vanilla RL. It is concluded that vanilla RL is incompetent to train a neural policy for a hierarchical implementation with sparse reward in this dense and highly dynamic environment. Apparently, the CRL in our proposed scheme outperforms vanilla RL in terms of sample efficiency and training convergence speed when integrated with MPC.

With the essential role of augmented decision variables in the MPC, we use the stated new baseline, which adopts the paradigm of high-level MPC with augmented decision variables by human-expert experience (denoted as MPC-HE). In Fig. 4(b), the ego vehicle initially accelerates to track the handcrafted state references as expected. However, the modulated cost function is unreasonable with an inappropriate tracking time reference. As a consequence, the ego vehicle is forced to slow down when it is near the designed tracking position, missing out the optimal opportunity to



(a) $t = 1.9\,\mathrm{s}$ (b) $t = 3.9\,\mathrm{s}$ (c) $t = 5.5\,\mathrm{s}$

Fig. 3. Key frames of trail with our proposed approach (MPC-CRL) in chance-aware lane-change problems. The vehicles in the upper lane represent the traffic flow, the vehicle in red is the ego vehicle, the vehicle in orange in the lower lane is denoted as the front vehicle, the dotted line in red represents the future trajectory of the ego vehicle, the dotted line in blue is the executed trajectory, and the colorbars refer to the different longitudinal speed of predicted and executed trajectories of the ego vehicle, respectively.



(a) MPC-RL: $t = 2.1\,\mathrm{s}$ (collision) (b) MPC-HE: $t = 5.7\,\mathrm{s}$ (collision) (c) MPC-SE3-CRL: $t = 4.1\,\mathrm{s}$ (collision)

Fig. 4. Key frames of trails with different baselines (MPC-RL, MPC-HE, and MPC-SE3-CRL) in chance-aware lane-change problems. The notions are as same as those in Fig. 3.

merge into the traffic flow. Hence, we conclude that our learning framework manifests the effectiveness in policy learning for chance-aware lane-change scenarios.

In Fig. 4(c), the ego vehicle drives with an irregular behavior. The ego vehicle initially tracks the dynamic chance and then is roughly dragged to track the learned position and heading angle. The non-smooth trajectory generated by the MPC violates human driver's behavior, as the ego vehicle almost drives to the target lane when $t = 2.0\,\mathrm{s}$. Besides, the ego vehicle crashes into the traffic flow when $t = 4.1\,\mathrm{s}$. Hence, our proposed method outperforms MPC with SE(3) decision variables over lane-change performance in terms of trajectory feasibility and collision avoidance.

To quantitively analyze the results attained by different methods, we define an episode with any collision as a collided case, and a totally collision-free episode finished in finite time as a successful case. Furthermore, the collision-free episode terminated by the duration of the simulation

TABLE II

COLLISION, SUCCESS, AND TIME-OUT RATE OF DIFFERENT METHODS FOR CHANCE-AWARE LANE CHANGE TASKS.

Approaches	Succ. (%)	Coll. (%)	Time-out (%)
MPC-CRL	**99**	**0**	**1**
MPC-RL	15	85	0
MPC-HE	59	33	8
MPC-SE3-CRL	64	32	4

is defined as a time-out case. Afterwards, we run the experiment repeatedly 100 times and record the corresponding collision, success, and time-out rate of various methods as documented in Table II. It is observed that our approach outperforms all baselines. Essentially, by using augmented decision variables to automatically modulate the cost function of MPC, the adaptativeness of our approach to a dense and highly dynamic environment is improved significantly. Hence, we conclude that our proposed framework manifests a stronger guarantee of collision avoidance and task success.

VII. Conclusions

In this paper, we proposed a novel learning-based MPC framework with appropriate parameterization using augmented decision variables. Instead of choosing partial variables (such as only the positions) as high-level references, we learned full-state references and regulatory factors corresponding to their importance. Hence, the cost function was automatically modulated with our specifically-designed augmented decision variables. Besides, ordered multi-phase curricula were generated for learning a neural policy using RL, which leads to faster convergence speed and better policy quality. Furthermore, through a series of comparative experiments, our approach demonstrated the superiority in terms of success rate in the chance-aware lane-change task under dense and dynamic traffic settings. Our future work is to further enhance the scalability and generalizability of our method by reformulating the RL to a step-based paradigm and developing a more efficient training pipeline with analytical gradients. Hardware validation is also part of our future interests.

References

[1] S. Dixit, U. Montanaro, M. Dianati, D. Oxtoby, T. Mizutani, A. Mouzakitis, and S. Fallah, "Trajectory planning for autonomous high-speed overtaking in structured environments using robust MPC," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 6, pp. 2310–2323, 2019.

[2] W. Cao, M. Mukai, T. Kawabe, H. Nishira, and N. Fujiki, "Cooperative vehicle path generation during merging using model predictive control with real-time optimization," *Control Engineering Practice*, vol. 34, pp. 98–105, 2015.

[3] M. Mukai, H. Natori, and M. Fujita, "Model predictive control with a mixed integer programming for merging path generation on motor way," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2017, pp. 2214–2219.

[4] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2021.

[5] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[6] J. Chen, B. Yuan, and M. Tomizuka, "Model-free deep reinforcement learning for urban autonomous driving," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 2765–2771.

[7] I. Nishitani, H. Yang, R. Guo, S. Keshavamurthy, and K. Oguchi, "Deep merging: Vehicle merging controller based on deep reinforcement learning with embedding network," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 216–221.

[8] F. Fuchs, Y. Song, E. Kaufmann, D. Scaramuzza, and P. Dürr, "Super-human performance in gran turismo sport using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4257–4264, 2021.

[9] Y. Song, H. Lin, E. Kaufmann, P. Dürr, and D. Scaramuzza, "Autonomous overtaking in gran turismo sport using curriculum reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9403–9409.

[10] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, 2019.

[11] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using Gaussian process regression," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2736–2743, 2019.

[12] K. Lowrey, A. Rajeswaran, S. Kakade, E. Todorov, and I. Mordatch, "Plan online, learn offline: Efficient learning and exploration via model-based control," *arXiv preprint arXiv:1811.01848*, 2018.

[13] N. Karnchanachari, M. I. Valls, D. Hoeller, and M. Hutter, "Practical reinforcement learning for MPC: Learning from sparse objectives in under an hour on a real robot," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 211–224.

[14] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-time neural MPC: Deep learning model predictive control for quadrotors and agile robotic platforms," *IEEE Robotics and Automation Letters*, pp. 1–8, 2023.

[15] J. Ma, Z. Cheng, X. Zhang, Z. Lin, F. L. Lewis, and T. H. Lee, "Local learning enabled iterative linear quadratic regulator for constrained trajectory planning," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[16] Z. Cheng, Y. Li, K. Chen, J. Ma, and T. H. Lee, "Neural-iLQR: a learning-aided shooting method for trajectory optimization," *arXiv preprint arXiv:2011.10737*, 2022.

[17] Y. Song and D. Scaramuzza, "Learning high-level policies for model predictive control," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7629–7636.

[18] ——, "Policy search for model predictive control with application to agile drone flight," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2114–2130, 2022.

[19] Y. Wang, B. Wang, S. Zhang, H. W. Sia, and L. Zhao, "Learning agile flight maneuvers: Deep SE(3) motion planning and control for quadrotors," *arXiv preprint arXiv:2209.11097*, 2022.

[20] Q. Ge, Q. Sun, S. E. Li, S. Zheng, W. Wu, and X. Chen, "Numerically stable dynamic bicycle model for discrete-time control," in *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*. IEEE, 2021, pp. 128–134.

[21] D. Rengarajan, G. Vaidya, A. Sarvesh, D. Kalathil, and S. Shakkottai, "Reinforcement learning with sparse rewards using guidance from offline demonstration," *arXiv preprint arXiv:2202.04628*, 2022.

[22] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] L. Biewald, "Experiment tracking with weights and biases," 2020, software available from wandb.com. [Online]. Available: https://www.wandb.com/

[25] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.