# A Novel Motion Planning for Autonomous Vehicles Using Point Cloud based Potential Field

Minghao Ning, *Member, IEEE*, Amir Khajepour, *Senior Member, IEEE*, Ehsan Hashemi*, *Senior Member, IEEE*, Chen Sun, *Member, IEEE*

***Abstract*—Ensuring accurate and efficient perception and motion planning is critical for the safety of autonomous vehicles. Addressing these pivotal challenges, this paper introduces a novel motion planning method employing a Lidar point cloud-based potential field (PF). Our approach innovatively extracts the drivable area boundary from point cloud, enhancing computational efficiency and reducing common perception errors, such as missed detections and inaccurate obstacle shape estimation. Built upon this drivable area boundary, the PF effectively represents the cost of traversing diverse areas. The PF is integrated into a model predictive control (MPC) framework to generate control commands considering vehicle dynamics, constraints, collision avoidance, and passenger comfort. Given the highly nonlinear nature of simultaneous longitudinal and lateral motion planning, an efficient Frenet frame-based trajectory sampling method is developed to provide an initial guess of the optimal trajectory for this complex motion planning task. The perception module has been validated in real bus tests, confirming its reliability and efficiency, and the entire motion planning methodology has been rigorously tested through simulations. These simulations show that our method efficiently generates smooth and safe control commands, even in challenging scenarios where the obstacle vehicle suddenly changes its lane, and remains robust under considerable state observation noise.**

***Index Terms*—Motion Planning, Model Predictive Control, Potential Field, Drivable Area Detection, Point Cloud**

## I. Introduction

### A. Motivation

AUTONOMOUS vehicles have the potential in increasing traffic flow and reducing accidents [1]. Critical to realizing this potential are the interconnected tasks of perception, motion planning, and trajectory tracking control. The efficiency and accuracy of these modules are significant, especially in dynamic and unpredictable driving environments [2]. Current perception methods struggle with real-time processing, often requiring high-end GPUs. More critically, these methods inadequately address the Out of Distribution (OOD) challenge, failing to recognize rare or unforeseen obstacles—a significant limitation in data-driven approaches trained on limited datasets. Furthermore, the conventional bounding box representation of the obstacles lacks flexibility for irregular-shape obstacles such as curbs. Using this representation directly in motion planning complicates the boundary constraints, particularly when dealing with numerous obstacles, thus significantly limiting efficiency. Moreover, motion planning considering vehicle dynamics, constraints, collision avoidance, and passenger comfort is inherently complex and computationally demanding due to the nonlinear nature of the motion model and the obstacle avoidance constraints.

Our research is motivated by the need for a novel method that not only processes perception data efficiently and accurately, but also seamlessly integrates it into the motion planning module, thus enhancing the overall safety and performance of autonomous vehicles.

### B. Related Research

*1) Perception:* The perception module's efficacy relies on its sensors and associated algorithms. Cameras and Lidars are prevalent sensors for autonomous vehicles. While cameras offer detailed color and texture data, they are susceptible to varying light conditions and can struggle with low-texture scenarios. Lidars, on the other hand, actively emit laser beams to gauge distances unaffected by illumination and have superior distance resolution and geometric detail. This precision is crucial for subsequent motion planning. Therefore, this research will employ a Lidar sensor for environmental sensing.

The perception algorithms can be categorized into two groups: data driven methods and model based methods. The data driven methods train convolutional neural networks to estimate 2D or 3D bounding boxes [3], [4]. However, these methods often fail with unseen cases due to the limited number of classes in training datasets. For example, KITTI [5] and nuScenes [6] contain only a few classes, making networks trained on them ineffective when encountering unseen obstacles like geese. Besides, the trade-off between the neural networks' accuracy and runtime is still an open question. On the contrary, classical model based point cloud processing methods can detect objects efficiently and accurately. They use the geometry information of obstacles and can still perform well when facing uncommon obstacles. Region of Interest, ground removal, clustering and shape estimation are the key components of such model based methods [7], [8].

In this paper, to get the environment information in an accurate and fast way, a method to build potential field (PF) directly from the point cloud is proposed. The high-definition (HD) map information and an efficient ground point removal method are used to extract all Lidar points above the road surface or the obstacle points. Instead of doing a



* Corresponding author

M. Ning and A. Khajepour are with the Department of Mechanical and Mechatronics Engineering, University of Waterloo, Ontario, N2L3G1, Canada (e-mail: {minghao.ning;a.khajepour}@uwaterloo.ca)

E. Hashemi is with the Mechanical Engineering Department, University of Alberta, Alberta, T6G1H9, Canada (e-mail: ehashemi@ualberta.ca)

C. Sun is with the Department of Data and Systems Engineering, University of Hong Kong, Pok Fu Lam, Hong Kong, China (e-mail: c87sun@hku.hk)

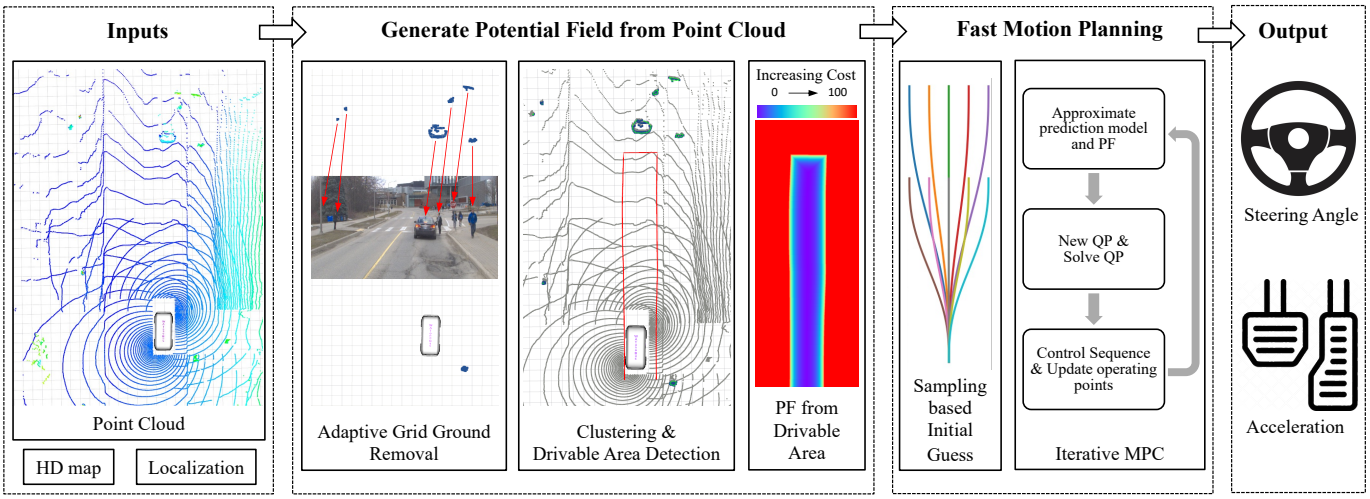



Fig. 1. Overview of the proposed fast and safe point cloud based motion planning method. The framework encompasses two primary steps: 1) Generating Potential Field from Point Cloud: This step uses point cloud data to create a potential field, reducing perception errors and computational cost. An HD map-guided adaptive grid ground removal method can detect any obstacle above the ground surface. Obstacle points are then clustered, and a convex hull is calculated for each cluster, effectively reducing data volume by representing nearby points as groups. The drivable area is extracted and a potential field is calculated, providing a simplified, smoother representation for the motion planner. 2) Fast Motion Planning: This step optimizes longitudinal and lateral motion control commands simultaneously. It employs Frenet frame trajectory sampling to provide an initial trajectory guess, and the motion planning MPC is solved iteratively from this initial guess.

complicated clustering and then estimating the 3D bounding box of obstacle, which is time consuming and not flexible, a fast way to extract the convex hull of the obstacle points is used. The convex hull of a group of points is a tight fitting convex boundary around the points, which is a concise and accurate representation of the obstacle boundary.

*2) Motion planning:* Motion planning for autonomous vehicles is a complex problem as it needs to consider obstacle avoidance, vehicle dynamic constraints and passenger comfort simultaneously. Many motion planning methods have been proposed during the past decades. Some classical methods aim to handle static obstacles [9]–[11].

Recently, many optimization based methods have been proposed to consider the movement of obstacles, they optimize an objective function of the vehicle and obstacle states in addition to states and input constraints. There are two ways to find the optimal trajectory: exhaustive searching and numerical optimization. The exhaustive searching based methods search the optimal trajectory within spatial temporal domain, but their performance is highly related to the sampling density [12]. They can generate better trajectories by increasing the sampling density, but at the cost of increased computational resources. Frenet coordinate system has been widely used to reduce the search complexity of the spatial domain by using the center of the road as a reference. Among the numerical optimization based methods, model predictive control (MPC) is a good framework because it can consider obstacle avoidance, vehicle dynamic constraints and passenger comfort simultaneously [13]–[17].

However, the MPC based motion planning is inherently a nonlinear problem due to vehicle dynamics coupling, non-convex obstacle constraints and non-convex PF cost. The original nonlinear problem is hard to solve in real-time, so simplification like decoupling the vehicle motion into longitudinal and lateral motion for planning, and approximation or quadratifying the PFs are usually needed. The choice of the operating points for the simplification and approximation will determine the optimality of the generated trajectory. Previous methods usually use constant velocity assumption to find the operating points, which may not be suitable for complex driving scenarios. To address this, a multiphase overtaking maneuver planning is proposed in [18], where an initial guess, obtained from the outer adaptive gradient-assisted particle swarm optimization algorithm, is used to warmly start the inner gradient-based optimization layer. However, this method may encounter limitations due to its generation of initial guesses without prior waypoint information, potentially impacting performance in diverse driving conditions. The waypoint information is used in [19] to generate feasible lane change maneuvers via the exponential functions, then the generated reference path is tracked by a Model Predictive Path-following Control (MPFC) method.

Parallel to these model-based methods, the data-driven deep neural networks have also shown promising strides in optimal motion planning [20]–[22]. Despite these advancements, the inherent “black box” nature of neural networks poses interpretability challenges. This lack of transparent decision-making processes in neural network outputs has restrained their widespread adoption in safety-sensitive applications.

From the literature review, finding the optimal trajectory is complex and time-consuming, approaches via exhaustive search or numerical optimization have their own advantages and disadvantages. The search based methods are efficient if the sampling density is low, they can escape from local optimum to approximate the global optimum, but they cannot reach the global optimum unless the sampling density is high enough. In contrast, the simplified and approximated MPC methods can reach the global optimum if given a good initial solution. Simplification and approximation should be done at proper operating points to reduce the optimization error.

We introduce a novel approach that builds upon these insights. By sampling an initial guess of the optimal trajectory in the Frenet frame, we establish a strong starting point for our MPC-based motion planning. This allows us to efficiently find the optimal solution for motion planning without compromising performance, even in the challenging situations.

*3) Contributions:*

- Development of an efficient method to generate safe drivable areas and potential fields directly from Lidar point cloud data, addressing the critical need for speed and accuracy in the perception phase.
- Introduction of a novel framework for simultaneous planning of longitudinal and lateral motions using the proposed drivable area and PF, effectively bridging the gap between perception and motion planning.
- Implementation of an efficient solution for the non-linear challenges in motion planning, utilizing a sampling-based initial guess within the Frenet frame to provide a robust starting point for our motion planning MPC.

### C. Paper Organization

This paper is organized as follows: Section II details our method for constructing PF from Lidar point cloud data and developing the corresponding PF-based motion planning MPC. It also introduces our approach for solving the motion planning MPC in real-time using an iterative process with an initial guess. Section III presents real-world tests that validate the safety and efficiency of our PF generation method. In Section IV, we provide simulation results to demonstrate the advantages of our proposed motion planning method. Finally, Section V concludes the paper and outlines future work directions.

## II. Point Cloud Based Motion Planning

### A. Generating Potential Field from Lidar Point Cloud

The point cloud is used directly to generate potential field, it aims to represent the environment with fewer perception errors at a lower computational cost.

It consists of three steps: first an adaptive grid ground segmentation is employed to remove ground points; then the remaining obstacle points are clustered and a convex hull that envelops the obstacle points is extracted for each cluster; finally, the drivable area boundaries are built based on the obstacle and road boundaries to generate the PF based on the distance from the vehicle to the drivable area boundaries.

*1) Adaptive Grid Ground Removal:* This part plays a crucial role in the Lidar based detection. The algorithm is shown in Algorithm 1, by leveraging HD map information, it divides the point cloud into multiple grids, and estimates a plane model for each grid. This allows for a more accurate and robust estimation of road surface compared to fitting only one ground plane model for the entire point cloud, as it adapts to various conditions and mitigates the impact of noise. For the region closest to the vehicle, it first selects candidate ground points based on the height of the Lidar sensor placement. It then iteratively estimates the road surface model. This estimation serves as the initial value for ground estimation in other regions. One of the key strengths of the adaptive grid ground removal algorithm is its ability to adjust road surface fitting thresholds based on the distance from the vehicle. More stringent fitting thresholds are set for areas closer to the vehicle, while more lenient thresholds are applied to regions further away. This method ensures that all points above the road surface are detected for use in subsequent detection modules.

**Algorithm 1:** Adaptive Grid Ground Removal

**Input :** A list of points $\mathbb{P} = \{P_1, \cdots, P_N\}$, a list of grids $\mathbb{G} = \{G_1, \cdots, G_M\}$ sorted by the distance to the vehicle, where $N$ denotes the number of points and $M$ denotes the number of grids.

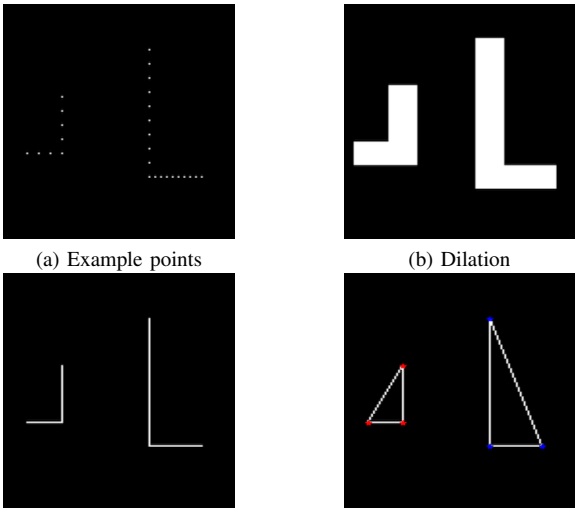
**Output :** A list of boolean values denoting if it's an obstacle point $\mathbb{O} = \{O_1, \cdots, O_N\}$, a list of estimated ground plane models $\mathbb{F} = \{z = f_1(x, y), \cdots, z = f_M(x, y)\}$.

**Initialize:** Set height of the Lidar $h_L$, set threshold parameters $\{T_1, \cdots, T_M\}$, set the maximum number of iterations $N_{\max}$. Set initial nearby ground fit $f_0 = -h_L$.

**for** $i = 1$ **to** $M$ **do**
  Find points $\mathbb{P}_i$ in grid $G_i$, set $f_i^1 = f_{i-1}$.
  **for** $j = 1$ **to** $N_{\max}$ **do**
    Find inliers $\mathbb{P}_{In} = \{P_k | (|z_k - f_i^j(x_k, y_k)| \leq T_i), P_k \in \mathbb{P}_i\}$
    Fit a new model $f_i^{j+1}$ by minimizing the estimation error of the inliers $\mathbb{P}_{In}$. $f_i^{j+1} = \min_f \sum_{P_k \in \mathbb{P}_{In}} (z_k - f(x_k, y_k))^2$
    **if** $f_i^j \approx f_i^{j+1}$ **then**
      **converge and early stop**
  Set the boolean value of inliers as False, save $f_i$.

*2) Clustering and Extracting Obstacle Boundary:* The obstacle points are then clustered and the convex hull is calculated for each cluster. They can reduce the amount of data by grouping the nearby points together and only using several boundary points to represent each group.

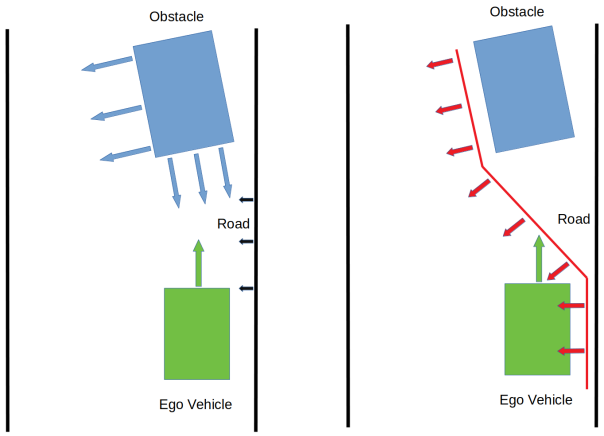
In this work, obstacle points from the 3D point cloud are first projected onto a bird-eye view, a 2D representation, with a resolution of $\Delta p$ to facilitate the clustering process. This projection transforms the 3D coordinates of each point into a pixel coordinate of 2D plane. The morphological dilation and erosion operations are then applied to this 2D representation for effective clustering of obstacle points. To group points within a distance of $\epsilon$, a circular structuring element (kernel) of size $2\epsilon/\Delta p + 1$ is used in these operations. As shown in Fig. 2, the example points are sparsely distributed, the dilation operation expands the white region from these example points so that nearby points are connected, the erosion operation shrinks the white region so that the original shapes the obstacle points are kept. After the erosion operation, the original example points are clustered into two separate objects, then the boundary points for each object are extracted as shown as the red and blue points. In this example, only six points are



(a) Example points (b) Dilation

(c) Erosion (d) Boundary points

Fig. 2. An example of clustering and using convex hull to represent obstacles: (a) Example points, (b) Dilation connects and separates the points into two groups, (c) Erosion restores the original shape, and (d) The boundary points of the convex hull are used to represent the obstacle concisely.




(a) Adding up PFs (b) Drivable Area Boundary

Fig. 3. Two Different Ways to Build the PF: (a) Demonstrates the conventional way to handle obstacles and road boundaries individually when defining the PF and then simply adding them up as the overall PF. This will trap the ego vehicle in a local optimum to stop behind the obstacle. (b) Illustrates the proposed method of extracting drivable area boundaries to construct the PF, which can mitigate this issue. The combination of potential fields (shown in red and green) can guide the ego vehicle to execute a left turn and bypass the front obstacle.

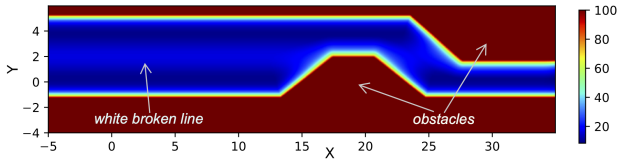



Fig. 4. An example of generated PF for a two-lane case.

needed to represent the original points.

*3) Extract Drivable Area Boundaries and Creating PF:* After obtaining the obstacle boundaries, traditional methods construct separate PFs for different obstacles and road boundaries, subsequently summing these PFs as an overall representation of the environment [16], [17], [23]. This could lead to the common issue of PF based motion planning, getting stuck at local optimum. As depicted in Fig. 3a, the combined attractive PF guiding the ego vehicle forward and the repulsive PFs from obstacles and road boundaries trap the vehicle behind the obstacle. To address this, we extract drivable area boundaries to construct the PF, which, as shown in Fig. 3b, will direct the ego vehicle to turn left and overtake the obstacle ahead.

To extract the drivable area boundary, the obstacle boundary region and the road boundary are first expanded by $r$ meters as the safety zone. This step is to take the dimension of ego vehicle into consideration, so that the ego vehicle will not hit into the obstacle or road boundary. Then the left-side and right-side drivable area boundaries can be founded by the following steps: The longitudinal axis $s$ is first discretized to a predefined resolution $\Delta s$ along the lane; The boundaries are searched forwardly starting from the first point. For cases where there is no obstacle, the expanded road boundaries are directly used as the left-side and right-side boundaries. For cases where obstacle exists, depth first search (DFS) algorithm is used, the boundaries that have the widest feasible spacing will be selected. If the search along this selection fails in the middle due to fully blockage, it will back-trace and try other directions. An example is shown in Fig. 1, where the smooth red lines represent the extracted drivable area boundaries.

The drivable area boundaries can be treated as the polyline-shaped non-crossable objects, and the potential field function (1) can be used to build the PF, where $x$ and $y$ are at the obstacle coordinates, $a$ and $b$ are intensity and shape parameters of the PF, respectively, and $X_N$ and $Y_N$ are the normalization term in longitudinal and lateral direction. The shortest distance from the ego vehicle to the boundaries will used to calculate the PF.

$$U_{NC}(x,y) = \frac{a}{((\frac{x}{X_N})^2 + (\frac{y}{Y_N})^2)^b} \tag{1}$$

For a multi-lane road, the potential field of white broken line can be calculated using the function (2).

$$U_C(x,y) = a\exp\left(-b((\frac{x}{X_N})^2 + (\frac{y}{Y_N})^2)\right) \tag{2}$$

An example of the generated PF for a case where two obstacle cars on a two-lane road is shown in Fig. 4, note the values larger than 100 have been trimmed. It can clearly show the collision risk at different positions.

## B. Motion Planning Method Design and Solution

The motion planning MPC algorithm can be generally divided into two parts: designing the motion planning MPC model, and solving it in real-time.

*1) Vehicle Dynamics Model:* Kinematic and dynamic bicycle model are two commonly used models. The kinematic bicycle model assumes the velocity vectors at front and rear wheels are in the direction of the orientation of two wheels respectively, it can only perform well when the vehicle speed is low. While the dynamic model is more suitable for higher speed cases where the small steering angle assumption usually holds, it considers the impact of tire forces, making it a better representation of the vehicle during dynamic maneuvers. Here the dynamic bicycle model is introduced, as shown in Fig. 5.

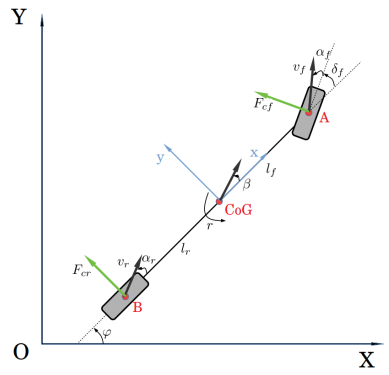



Fig. 5. Dynamic Bicycle Model.

The equations of motion of this dynamic model are:

$$m(\ddot{x} - \dot{y}r) = F_{xT} \tag{3a}$$
$$m(\ddot{y} + \dot{x}r) = (F_{cf}\cos\delta_f + F_{cr}) \tag{3b}$$
$$I_z\dot{r} = (l_f F_{cf}\cos\delta_f - l_r F_{cr}) \tag{3c}$$
$$\dot{X} = \dot{x}\cos(\varphi) - \dot{y}\sin(\varphi) \tag{3d}$$
$$\dot{Y} = \dot{x}\sin(\varphi) + \dot{y}\cos(\varphi) \tag{3e}$$

where $X$ and $Y$ are the coordinates of the center of gravity (CoG) in the global frame. $\dot{x}$ and $\dot{y}$ are longitudinal and lateral velocities in the local vehicle frame. $\varphi$ is the heading angle, $r$ is the yaw rate. $m$ is the vehicle's mass, $I_z$ is the moment of inertia of the vehicle around $z$-axis. $l_f$ and $l_r$ represent the distance from the CoG to the front and rear axles, respectively. $F_{cf}$ and $F_{cr}$ are the lateral forces of front and rear tires. $F_{xT}$ is the total longitudinal force, and $a = F_{xT}/m$ is used later for convenience. The control inputs are the front steering angle $\delta_f$ and $a$.

Assuming small $\delta_f$, $\cos\delta_f \approx 1$, and the vehicle is operating within normal driving conditions with small lateral acceleration where tire forces remain in the linear region. So the linear tire model in [16] is valid in our case:

$$F_{cf} = 2C_{\alpha f}(\delta_f - \frac{\dot{y} + l_f r}{\dot{x}}) \tag{4a}$$
$$F_{cr} = 2C_{\alpha r}(-\frac{\dot{y} - l_r r}{\dot{x}}) \tag{4b}$$

where $C_{\alpha f}$ is the lateral cornering stiffness of each front tire, $C_{\alpha r}$ is the lateral cornering stiffness of each rear tire. The factor 2 accounts for the fact that there are two front wheels and two rear wheels.

By combining the above equations, state space model can be derived based on the dynamic bicycle model as:

$$\begin{aligned} \dot{z} &= A'(\bar{\rho})z + B'(\bar{\rho})u + D'(\bar{\rho}) \\ y &= C'z + \omega \end{aligned} \tag{5}$$

$$C' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, D'(\bar{\rho}) = \begin{bmatrix} \bar{\dot{x}}\sin\bar{\varphi}\bar{\varphi} + \bar{\dot{y}}\cos\bar{\varphi}\bar{\varphi} \\ -\bar{\dot{x}}\cos\bar{\varphi}\bar{\varphi} + \bar{\dot{y}}\sin\bar{\varphi}\bar{\varphi} \\ -\bar{\dot{\varphi}}\bar{\dot{y}} \\ \bar{\dot{\varphi}}\bar{\dot{x}} \\ 0 \\ 0 \end{bmatrix}$$

where the state vector $z = [X, Y, \dot{x}, \dot{y}, \varphi, \dot{\varphi}]^T$, the control input vector $u = [a, \delta_f]^T$, the observation vector $y = [X, Y, \dot{x}, \varphi]^T$, and $\omega$ denotes the noise term. The vector $\bar{\rho} = [\bar{z}^T, \bar{u}^T]^T$ denotes the operating points where the linearization is performed. The choice of $\bar{\dot{x}}$ and $\bar{\varphi}$ plays an important role in the prediction model accuracy. A common choice is using the current fixed vehicle velocity and heading angle. But this may fails especially when ego vehicle performs lane change maneuver. In this research, the anticipated vehicle states from the initial guess of the optimal trajectory are used as the operating points to get a more accurate prediction model.

Using forward Euler method, Eqn.(5) can be represented as a discrete, multi-input, multi-output model

$$z(k+1) = A_k z(k) + B_k u(k) + D_k, \quad y(k) = C_k z(k) \tag{6}$$

where $A_k = I + A'(\rho)T_s$, $B_k = B'(\rho)T_s$, $D_k = D'(\rho)T_s$, $C_k = C'$, $T_s$ is the sampling time, $z(k)$, $y(k)$, and $u(k)$ denote the state, output state, and control variable values at time $k$, respectively. The prediction ($N_p$) and control ($N_c$; $N_c \leq N_p$) horizons are the future steps over which vehicle states are predicted and control inputs optimized, respectively, with control input held constant beyond $N_c$ steps.

The goal of the motion planner and controller is to track the desired trajectory accurately in real time and to ensure stability, comfort and safety. The cost function of the motion planning MPC consists of three parts, a) output reference tracking, b) manipulated variable move suppression, c) crash mitigation. The motion planning MPC can be summarized as:

$$\begin{aligned} \min_u & \sum_{k=1}^{N_p} ||y(k) - y_{ref}(k)||_Q^2 \\ & + ||u(k) - u(k-1)||_R^2 + PF(y(k)) \\ \text{s.t. } z(k+1) =& A_k z(k) + B_k u(k) + D_k \\ y(k) =& C_k z(k) \\ u_{\min} \leq& u(k) \leq u_{\max} \\ \Delta u_{\min} \leq& u(k) - u(k-1) \leq \Delta u_{\max} \\ y_{\min} \leq& y(k) \leq y_{\max} \\ u(k) =& u(k-1), \text{if } k > N_c \\ \text{for } k =& 1, \cdots, N_p \end{aligned} \tag{7}$$

where $y_{ref}$ is the reference value of the output sequence, $Q$, $R$ are the weight matrices, $PF$ is the potential field. The first term of the objective function represents the error between output and reference, which reflects the precise tracking in the control objective; the second term represents the magnitude of the control increment, which reflects the stability and comfort in the control objective; the last term represents the cost of the risk of collision, which reflects the crash mitigation in the trajectory planning task. It also includes three types of constraints: control constraints, control increment constraints, and output constraints.

*2) Sampling based Initial Guess:* The Frenet frame based trajectory sampling is developed to act as an initial guess of the optimal trajectory generated from the above MPC method. The Frenet coordinate system is widely used when sampling trajectories. It uses the road center as the reference line, and the position is represented by the longitudinal distance along

$$A'(\bar{\rho}) = \begin{bmatrix} 0 & 0 & \cos(\bar{\varphi}) & -\sin(\bar{\varphi}) & -\bar{\dot{x}}\sin(\bar{\varphi}) - \bar{\dot{y}}\cos(\bar{\varphi}) & 0 \\ 0 & 0 & \sin(\bar{\varphi}) & \cos(\bar{\varphi}) & \bar{\dot{x}}\cos(\bar{\varphi}) - \bar{\dot{y}}\sin(\bar{\varphi}) & 0 \\ 0 & 0 & 0 & \bar{\dot{\varphi}} & 0 & \bar{\dot{y}} \\ 0 & 0 & -\bar{\dot{\varphi}} & -\frac{2(C_{\alpha f}+C_{\alpha r})}{m\bar{\dot{x}}} & 0 & -\bar{\dot{x}} + \frac{2(l_r C_{\alpha r} - l_f C_{\alpha f})}{m\bar{\dot{x}}} \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{2(l_r C_{\alpha r} - l_f C_{\alpha f})}{I_z \bar{\dot{x}}} & 0 & -\frac{2(l_f^2 C_{\alpha f} + l_r^2 C_{\alpha r})}{I_z \bar{\dot{x}}} \end{bmatrix}, B'(\bar{\rho}) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & \frac{2C_{\alpha f}}{m} \\ 0 & 0 \\ 0 & \frac{2l_f C_{\alpha f}}{I_z} \end{bmatrix}$$

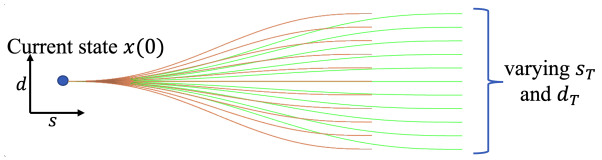



Fig. 6. Frenet coordinate system.

the reference line $s$ and lateral offset from the reference line $d$, as shown in Fig. 6.

The trajectory sampling uses a time-varying polynomial that connects an initial state with a desired state to describe the longitudinal and lateral motions [24] [25]. To make sure the generated trajectory is comfortable and smooth, the quintic polynomial is selected because it can minimize the jerk [26], where the jerk is defined as the change of acceleration over time. This quintic polynomial is a reasonable approximation for speed variation in many typical driving scenarios, where the vehicle usually doesn't change speed abruptly.

Let the motion state at $t = 0$ be $\left[x_0, \dot{x}_0, \ddot{x}_0\right]$ and the desired state at $t = T$ be $\left[x_T, \dot{x}_T, \ddot{x}_T\right]$, where $x_0$ and $x_T$ are the starting and ending positions, $\dot{x}_0$ and $\dot{x}_T$ are the velocities, $\ddot{x}_0$ and $\ddot{x}_T$ are the accelerations, and the $x$ can be represented either for lateral ($d$) or longitudinal ($s$) movements. The quintic polynomial $f_x(t)$ for either type of motion is:

$$\begin{aligned} & f_x(t) = a_{x0} + a_{x1}t + a_{x2}t^2 + a_{x3}t^3 + a_{x4}t^4 + a_{x5}t^5 \\ & \text{s.t.} \begin{cases} f_x(0) = x_0, \quad \dot{f}_x(0) = \dot{x}_0, \quad \ddot{f}_x(0) = \ddot{x}_0, \\ f_x(T) = x_T, \quad \dot{f}_x(T) = \dot{x}_T, \quad \ddot{f}_x(T) = \ddot{x}_T \end{cases} \end{aligned} \tag{8}$$

However, for the cases when the vehicle is expected to maintain a constant speed, and the longitudinal ending position constraint $f_s(T) = s_T$ doesn't exist, so the above $f_s(t)$ will be reduced to a quartic polynomial.

The Frenet frame based trajectory sampling first defines a series of candidate desired ending states where each state is denoted as $x(T) = [s_T, \dot{s}_T, \ddot{s}_T, d_T, \dot{d}_T, \ddot{d}_T]$, then it generates trajectories based on the above $f_d(t)$ and $f_s(t)$, finally it selects the trajectory that minimizes the cost function. The candidate ending states are predefined as

$$\begin{aligned} & \{T \in \{\text{linspace}(T_{\min}, T_{\max}, N_T)\}, s_T = \text{NaN}, \\ & \dot{s}_T \in \{\text{linspace}(\dot{s}_{T_{\min}}, \dot{s}_{T_{\max}}, N_V)\}, \ddot{s}_T = 0, \\ & d_T \in \{0, w_{\text{lane}}\}, \dot{d}_T = 0, \ddot{d}_T = 0\} \end{aligned} \tag{9}$$

where $\text{linspace}(a, b, N)$ generates an array from $a$ to $b$ with $N$ values. $s_T = \text{NaN}$ means no constraint exists on $f_s(T) = s_T$, enabling automatic computation of the longitudinal distance over a minimum-jerk trajectory. The $\dot{s}_{T_{\min}}$, $\dot{s}_{T_{\max}}$ and $N_v$ specify a set of ending longitudinal velocities so that the vehicle can adjust its velocity. Lastly, the lane width $w_{\text{lane}}$ facilitates lane change maneuvers.

To select the best trajectory, the cost function of each trajectory is defined as

$$J = w_{J_s} J_s + w_{J_d} J_d + w_{J_c} PF + w_{J_v} J_v \tag{10}$$

where $J_s$ and $J_d$ represent the jerk cost of longitudinal and lateral motions, respectively. $PF$ represents the collision cost. $J_v$ represents the difference between the ending longitudinal velocity and target longitudinal velocity. $w_{J_s}$, $w_{J_d}$, $w_{J_c}$ and $w_{J_v}$ are the weight parameters.

*3) Iterative MPC:* The accuracy of the approximated PF and the accuracy of the prediction model are two key factors determining the optimum solution. To have accurate PF and prediction model approximation, the operating point $\bar{\rho} = [\bar{z}^T, \bar{u}^T]^T$ should be close to the optimal solution of the original nonconvex problem. The accuracy of the prediction model (5) is mainly determined by the longitudinal velocity and the heading angle. One option for the operating point at time $k$ is $\bar{\rho}_k = [z_k^T, u_{k-1}^T]^T$, which means the current state $z_k$ and previous control $u_{k-1}$ are chosen as the operating point to approximate the $A_k$, $B_k$ and $D_k$, and then to predict the next state $z_{k+1}$. However, this is problematic for the quadratic programming (QP) based MPC because it uses recursive calculations of the state space model to predict the future. However, the prediction model needs to be predefined for each step in the prediction horizon to create the recursive calculations before the optimization. To avoid this issue, another option is to assume the longitudinal velocity and heading angle are fixed. But this is not a good assumption when the ego vehicle needs to change its velocity or heading. Besides, to approximate the PF accurately, the operating points consisting of vehicle positions are also needed to be close to the predicted states over the prediction horizon given the optimal control sequence.

In this work, the motion planning MPC is solved in an iterative way, as shown in Algorithm 2. At the first iteration, the reference trajectory sampled from Frenet frame method is used as the operating points to approximate the prediction model and the PF. Once the prediction model and the PF are defined, the MPC can be converted to a QP problem, and the control sequence $U^i(k)$ at $i$-th iteration can be solved efficiently. For following iterations, the control sequence $U^i(k)$ is applied to the vehicle motion equations, so the state sequence $S^i(k)$ at $i$-th iteration can be calculated, and this state sequence is used as the operating points again to approximate the prediction model and the PF, so a new QP problem can be formed, then the new control sequence $U^{i+1}(k)$ can be calculated. The above steps will be repeated until convergence or the maximum number of iterations is reached.

**Algorithm 2:** Iterative MPC

**Input** : Current state and previous control $\tilde{Z}_a(k)$, weight matrices $Q$ and $R$, constraints.
**Output** : Optimal control solution $U(k)$.
**Initialize:** Set operating points $\bar{\rho}^0$ based on Frenet planner result.

**for** $i = 0$ **to** $N_{\max}$ **do**
  Approximate prediction model matrices and PF at operating points $\bar{\rho}^i$;
  Convert the MPC to a QP problem $QPMPC^i$;
  Solve and get new control sequence $U^{i+1}(k)$.
  **if** $U^i(k) \approx U^{i+1}(k)$ **then**
    **converge and early stop**
  Infer new operating points $\bar{\rho}^{i+1}$ by applying $U^{i+1}(k)$ to vehicle motion equations.
**return** $U^{i+1}(k)$

## III. Experimental Results

In this section, the generation of drivable area boundary from point cloud is examined using real data collected by the Waterloo all-weather autonomous shuttle (WATonoBus) from the Mechatronic Vehicle Systems Lab, shown in Fig.7. The center Robosense-32 Lidar and two blind spot Bpearl Lidars offer 360-degree coverage around the bus. The localization module, Trimble APX-18 Land, consists of a high-precision IMU and GNSS system with Real Time Kinematic (RTK), which is capable of providing a centimeter-level global position. The computation unit is an NVIDIA Jetson AGX Orin.

### A. Dataset and Parameters

The dataset under consideration contains three complete loops around the Ringroad at the University of Waterloo. Each loop spans approximately 2.7 kilometers and takes roughly 10 minutes to complete. It has high traffic volume, including not just typical participants such as cars, trucks, buses, and pedestrians, but also uncommon objects like geese. Moreover, the Ringroad's physical characteristics further enhance the complexity of the dataset. It features curves and slopes, challenging for ground surface estimation. The utilized point cloud data is a combination of points from three Lidars, providing a comprehensive overview of the environment. To manage data size and complexity, this point cloud data is then downsampled

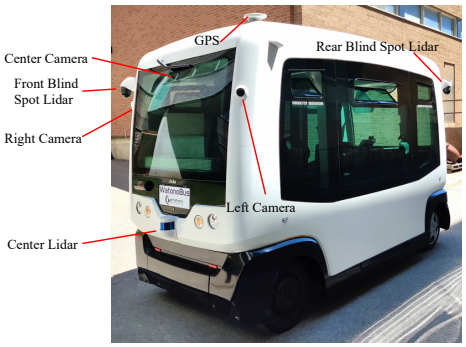
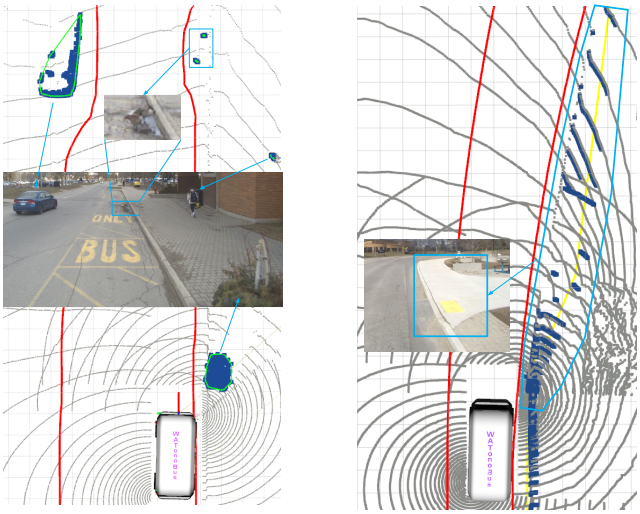



Fig. 7. WATonoBus for real data collection.




(a) Detect Geese (b) Inaccurate HD Map

Fig. 8. Safety demonstration examples: (a) Illustrates the proposed method's ability to identify and accurately adjust the drivable area boundaries to small, low-ground-clearance objects in real-world traffic conditions; the camera view shows the difficulty for the vision system to detect such small and low-texture objects. (b) Demonstrates robustness when using an inaccurate HD map, the yellow line denotes the right curb position from the HD map, the blue points denote the obstacle points, and the red lines show the generated safe drivable area boundary.

using a voxel size of 0.05m. This method ensures both the manageability of the data and the preservation of key details necessary for the study.

The parameters used in the proposed method are summarized: the pixel resolution $\Delta p$ is 0.1m/pixel, the clustering distance threshold $\epsilon$ is 0.2m, the safety expansion distance $r$ is 1m, the drivable space resolution $\Delta s$ is 0.1m, and the height of Lidar $h_L$ is 0.9m.

### B. Evaluation and Detection

The proposed drivable area boundary method shows robust safety features in various challenging scenarios. Firstly, it demonstrates the ability to detect uncommon and small-sized objects, an essential attribute to any perception system in autonomous driving. This is exemplified in Fig. 8a, a screenshot showing our method successfully identifying geese from a significant distance of 18 meters, and the generated drivable area boundary leaves room for the geese and car. This level of detection performance significantly enhances the overall safety of the autonomous driving system, ensuring its reliable operation in complex and dynamic traffic conditions.

Moreover, our method exhibits resilience when confronted with outdated or inaccurate HD map data, a common issue in real-world applications. As an illustration, Fig. 8b depicts a situation where recent construction has shifted the actual position of the right curb, encroaching into what was previously the road according to the HD map. Despite this discrepancy, our method successfully identifies points from the sidewalk as obstacle points and generates a safe drivable area boundary. This capability further emphasizes the safety and adaptability of our proposed method, as it can ensure safe navigation even in the face of infrastructural changes or mapping inaccuracies.

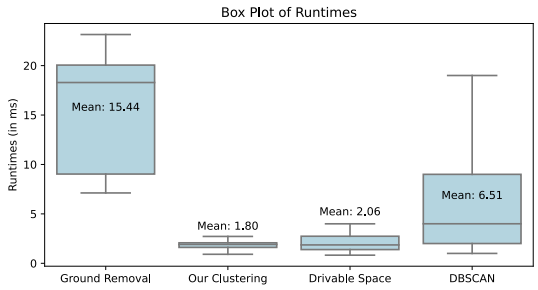



Fig. 9. Box plot representation of the runtime distribution for different components of our proposed method and the DBSCAN clustering method. The plot illustrates the mean runtime, medians, and quartiles for our adaptive grid ground removal, our clustering approach, and drivable area computation, compared to the DBSCAN method. Our methods demonstrate lower mean runtimes and lower variances, indicating higher efficiency and more consistent performance.

### C. Efficiency Demonstration

Efficiency is another key aspect of the proposed method. The proposed method, implemented in C++, exhibits high computational efficiency. On average, the method uses only 4% of the CPU capacity of the Jetson AGX Orin to generate the drivable area boundary from point cloud in 20ms.

The runtime distribution is depicted in Fig. 9, where we compare the performance of different components of our method — adaptive grid ground removal, our own clustering approach, and drivable area boundary computation — against the popular clustering method, DBSCAN [27]. To clarify, DBSCAN is used here as a comparative algorithm, not as part of our clustering approach. Our method involves a unique clustering technique distinct from DBSCAN.

The adaptive grid ground removal exhibits a mean runtime of 15.44ms, and the majority of its runtime measurements are less than 23ms. Meanwhile, our clustering approach and drivable area boundary computation both demonstrate great performance. They have mean runtimes of just 1.8ms and 2.06ms respectively, and exhibit low variance. In contrast, the DBSCAN method has a higher mean runtime of 6.51ms and a larger variance, indicating less consistent performance. This variance becomes more pronounced in scenarios with larger obstacles, such as buses, where the runtime can easily exceed 30ms. This is indicative of less consistent performance, particularly in complex environments.

This low CPU utilization, coupled with the short and consistent runtime, underscores the efficiency of our approach, making it suitable for real-time applications that require rapid processing and responsiveness.

## IV. Simulation Results

The performance of the proposed motion planning MPC is evaluated through simulations. The designed motion planning MPC is implemented in Matlab, and Eqn.(3) is used to represent the vehicle dynamics in the simulations. The parameters of the simulation vehicle are summarized in Table I.

TABLE I
Parameters of Simulation Vehicle

| Parameter | Description | Value | Unit |
|---|---|---|---|
| $m$ | Total mass of vehicle | 2270 | kg |
| $I_z$ | Yaw moment of inertia of vehicle | 4600 | kgm$^2$ |
| $l_f$ | Dist. (Long.) from CoG to front tires | 1.4 | m |
| $l_r$ | Dist. (Long.) from CoG to rear tires | 1.6 | m |
| $C_{\alpha f}$ | Cornering stiffness of front tires | 63500 | N/rad |
| $C_{\alpha r}$ | Cornering stiffness of rear tires | 65000 | N/rad |

### A. Double Lane Change Scenario

In this case, the ego vehicle drives along a two-lane straight dry road with two other driving cars. One car drives along the right lane at 6 m/s, the other drives along the left lane at 8 m/s. The desired speed of ego vehicle is set to be 11.1 m/s (40 km/h), so the ego vehicle is expected to do two lane change maneuvers to overtake the two obstacle cars. The control horizon $N_c$ = 6, the prediction horizon $N_p$ = 20, the sampling time $T_s$ = 0.1s, the maximum number of iterations for finding the optimal solution $N_{\max}$ = 10.

The efficacy of our method is evaluated by comparing the simulation results obtained with and without the use of the Frenet planner for generating initial guess trajectories, as well as against the MPFC [19]. So the benefits of proposed method using the initial guess trajectory can be established. The trajectories and associated results from the simulations are depicted in Fig. 10 and Fig. 11. The red trajectory denotes the low speed obstacle car on the right lane, and the yellow one denotes the high speed obstacle car on the left lane.

First, for the case without Frenet planner, the center of the right lane is used as the reference path to calculate the tracking error in the motion planning MPC. By building the PF based on the proposed drivable area boundaries, even without using the Frenet planner to come up with an initial guess of the optimal trajectory, the motion planning MPC can still generate control commands so that the ego vehicle can drive along the road without crashing into obstacle cars or getting stuck behind the obstacle cars. However, it takes too much time to solve the MPC problem when lane change maneuver is performed, nearly 100ms is required during the simulation time from 6s to 10s. Taking about 100ms to generate a control command is not acceptable for autonomous driving application where high-delaying may lead to risk of collision. Furthermore, the generated steering angle, lateral velocity, and yaw rate lack smoothness, potentially diminishing passenger comfort. The tracking error term in MPC, which penalizes large deviation from the center of the right lane, can also induce dangerously close proximity to obstacles during overtaking (T=6s).

Then, for the case with Frenet planner, the initial path generated from the Frenet planner is utilized as the starting search point for the motion planning MPC, shown as the black dotted line, where the time interval between two dots is $T_s = 0.1$s. The trajectory inferred based on the generated control commands is shown as the green dotted line. When the ego vehicle starts to change its lane (from T=1s to T=2s), the initial guess shows a harsh maneuver trajectory necessitating a large steering angle due to the lack of control constraints during the sampling procedure. However, the proposed method mitigates this by generating a safer, smoother trajectory. When the ego vehicle is bypassing the red obstacle car (from T=4s to T=8s), the initial guess merges to the center of the left lane, while the proposed method generates a trajectory that

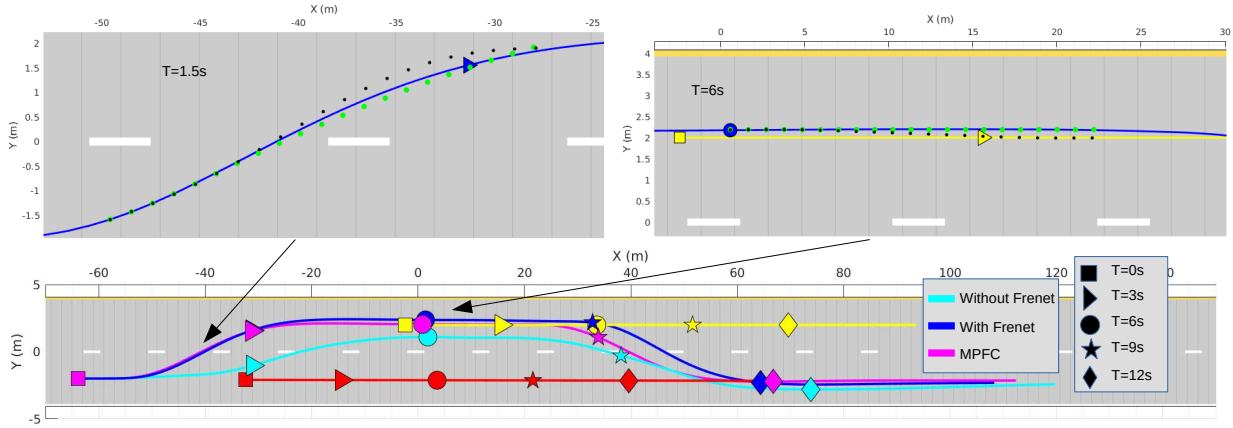



Fig. 10. Comparative trajectories in the double lane change scenario, showing paths taken by the ego vehicle using our method without the Frenet planner (cyan), with the Frenet planner (blue), and the MPFC method (magenta).

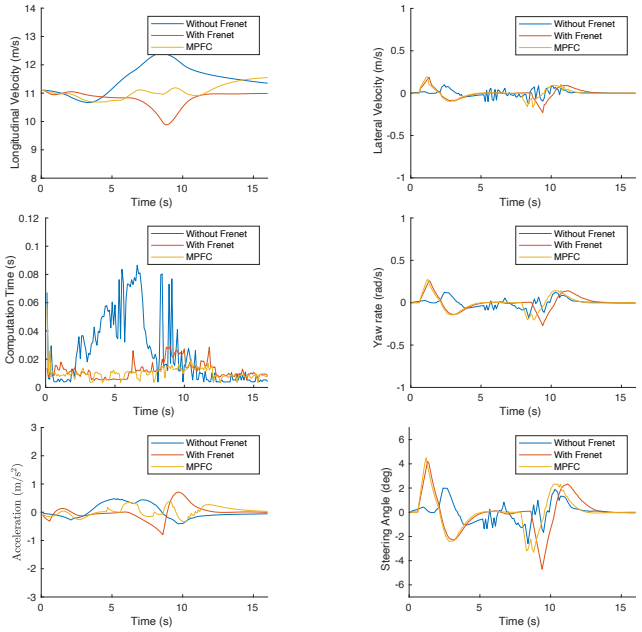



Fig. 11. Comparative results in the double lane change scenario.

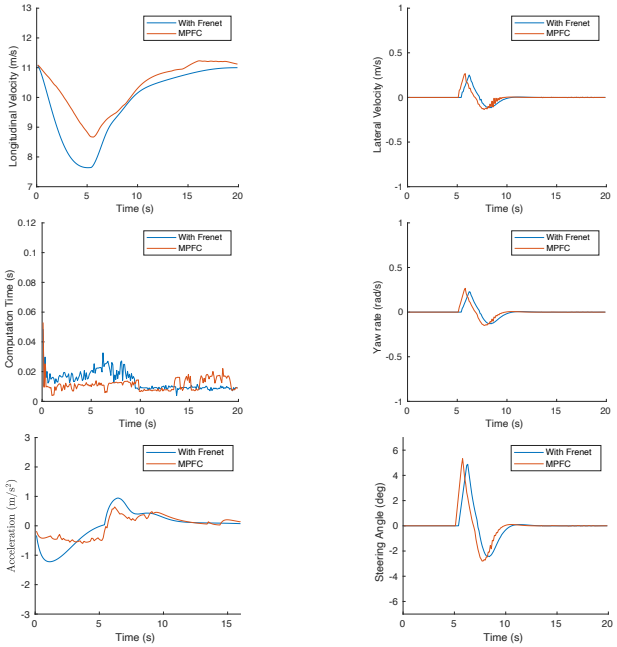



Fig. 12. Comparative results in the slow down and lane change scenario.

is shifted by about 20cm in lateral direction to keep a proper lateral distance to the red car. Compared with the previous one, the performance of the motion planner with the initial guess is much better. The computation time is much shorter, most of the MPC problems are solved within 20ms. In average, the computation time of previous method is 33.8ms, the proposed method is 12.9ms, the computation time is reduced by 62%. The generated steering angle is much smoother. Also, the lateral distance to obstacle when performing lane change is larger, which could help improve the safety.

Lastly, the comparative MPFC produces a trajectory that is similar to that of our proposed method. Despite a marginally shorter average computation time of 12.1ms, the MPFC shows less smoothness in control commands, particularly steering angle responses around 9s. This is attributed to the MPFC's strategy of treating each obstacle individually when generating the reference path. At that time, both the double lane change maneuver triggered by the red car and the single lane change maneuver triggered by the yellow car are received by the MPFC, then the MPFC has to choose one of them to follow. Such nature of handling obstacles individually can lead to inconsistent reference paths, resulting in unsmooth control commands. In contrast, the proposed method processes the obstacles collectively, so our method is able to generate more consistent control commands, ensuring passenger comfort.

### *B. Slow Down and Lane Change Scenario*

In this case, the red car drives slowly along the right lane at 8 m/s, and two cars drives along the left lane at 13 m/s. Initially, the left lane is occupied by the cars, so the ego vehicle is expected to decelerate and maintain a safe distance to the red car before safely executing a left lane change. The simulation results and trajectories are shown in Fig. 12 and Fig. 13.

Both the proposed method and the MPFC successfully manage the slow down and eventual lane change. Nevertheless, our method exhibits adept control by matching the longitudinal velocity of the red car at 8 m/s and maintaining a larger safety distance. The trajectory generated by our method reflects a smoother transition, which is especially noticeable in the

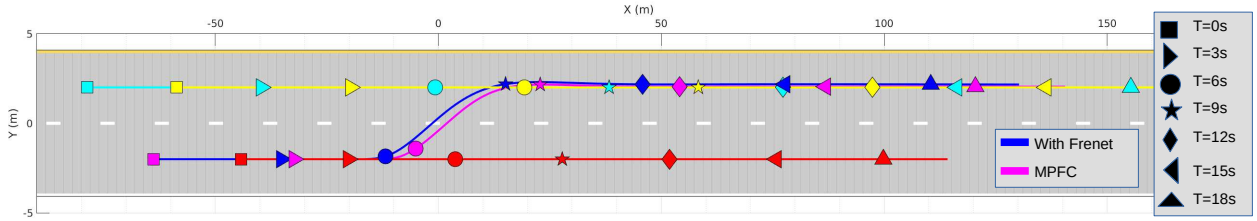



Fig. 13. Comparative trajectories in the slow down and lane change scenario, showing paths taken by the ego vehicle using our method with the Frenet planner (blue), and the MPFC method (magenta).

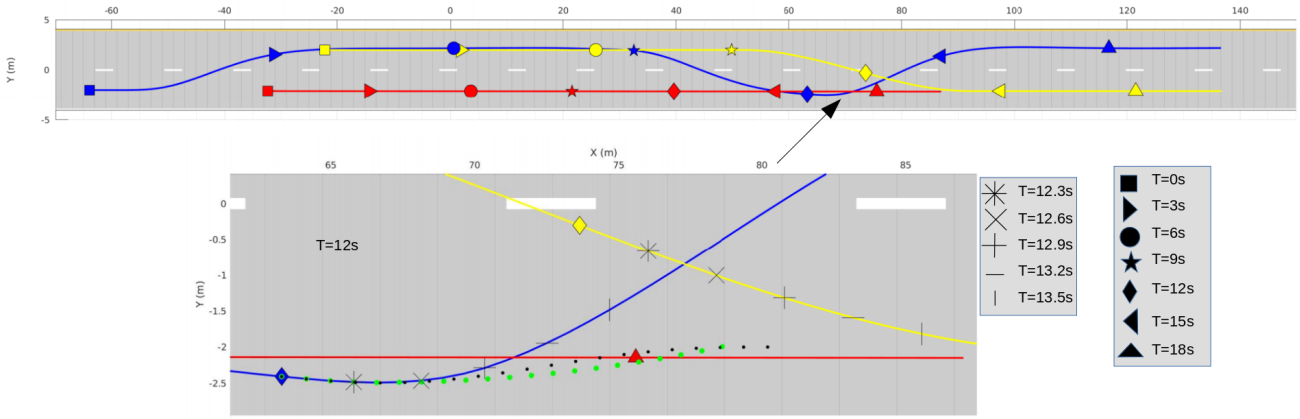



Fig. 14. Simulation trajectories of proposed method where front vehicle does a sudden lane change.

steady lateral velocity and consistent yaw rate as the vehicle executes the lane change.

The advantages of our method extend to the control command profile. During the critical phase of deceleration and lane switching, our method ensures that the vehicle's acceleration and steering inputs are modulated in a gradual, controlled fashion, which enhances the comfort and safety of the passengers. In contrast, the MPFC method, while effective in performing the maneuver, exhibits a less refined control command profile, leading to less predictable vehicle behavior.

### C. Challenging Scenario

To further verify the effectiveness and safety of the proposed motion planning MPC, a more challenging case is built. Here, the yellow obstacle car will do a sudden lane change when the ego vehicle is performing the lane change. The ego vehicle is expected to keep a safe distance from the yellow car and overtake it when it's safe. The simulation results with Frenet based initial guess are shown in Fig. 14 and Fig. 15.

The yellow obstacle car suddenly changes its lane when the ego vehicle is about to finish changing to the right lane at T=11s. The proposed method controls the ego vehicle to first decelerate to avoid collision, and then change to left lane. In this case, the computation time increases a little to handle the unexpected situation from T=11s to T=15s, however, the worst case of the computation time is still less than 40ms, and the average computation time is 17.4ms, so real-time can still be achieved. Also, the generated steering angle is still smooth. To be noted, in order to reduce the time spent in sampling and searching of the Frenet planner, the sampling space is sparse, so the sampled trajectory from Frenet planner doesn't reduce the longitudinal velocity properly (T=12-13s), but the proposed motion planning method managed to decelerate

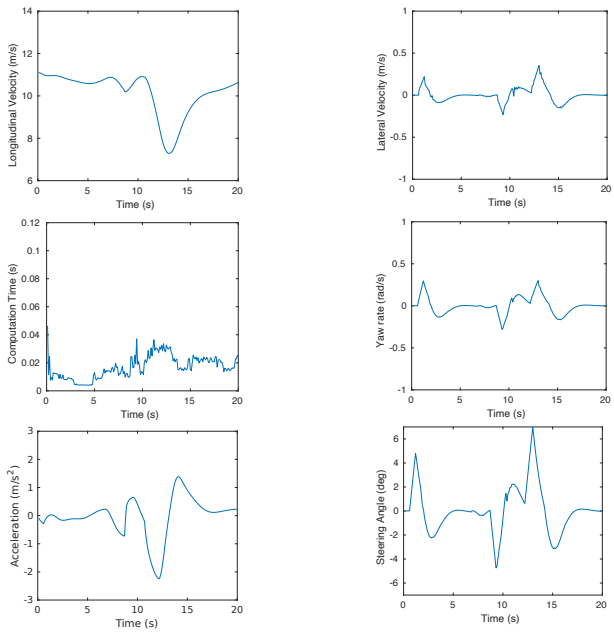



Fig. 15. Simulation results where front vehicle does a sudden lane change.

greatly to keep a safe distance, as shown as the green line is shorter than the black line, and the smallest longitudinal distance between the ego vehicle and the yellow car during this sudden lane change period is still larger than 10m. So this case demonstrates the safety and effectiveness of the proposed motion planning MPC.

### D. Impact of Model Uncertainty

To further evaluate the performance of the proposed method under model uncertainty, a Monte-Carlo simulation with 200 iterations is conducted with the challenging scenario. Gaussian

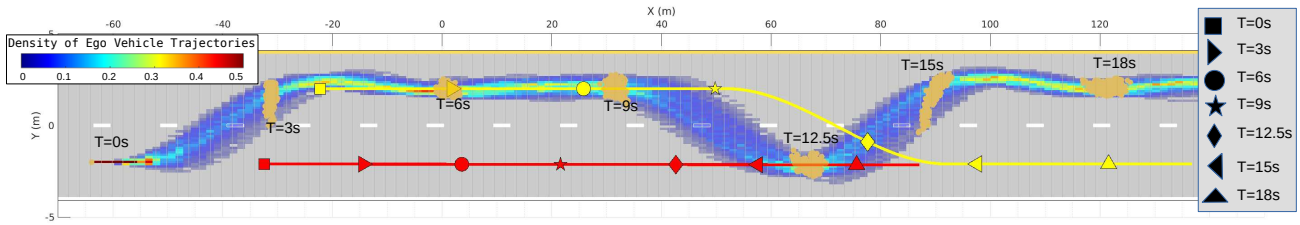



Fig. 16. Simulation trajectories of proposed method where front vehicle does a sudden lane change across 200 Monte-Carlo simulations.

TABLE II
PARAMETERS OF GAUSSIAN NOISE

| States | Description | $\sigma$ Value | Unit |
|---|---|---|---|
| $X$ | X position of vehicle | 1 | m |
| $Y$ | Y position of vehicle | 0.15 | m |
| $\dot{x}$ | longitudinal velocity of vehicle | 0.5 | m/s |
| $\varphi$ | yaw angle of vehicle | 1 | degree |

noises is introduced to the vehicle dynamics model (5) to simulate the effect of uncertainties in vehicle position, velocity and yaw angle, as detailed in Table II. The generated trajectories are shown in Fig. 16, expressed in the form of density map. The timestamps are noted via orange position points at the corresponding time as the nearby time text. No collision happens during the simulation even with the model uncertainty and unexpected lane change of the yellow car. At time T=12.5s, the worst case still keeps a longitudinal distance of 8m from the yellow car, which prevents the potential collision and ensures the safety. The generated control commands are not as smooth as the previous case without model uncertainty, it's mainly because the proposed Frenet initial guess planner is trying to keep the vehicle in the center of the lane. So the generated control commands will be affected if there is noise in the vehicle position and yaw angle. However, the comfort related metrics like maximum acceleration, maximum jerk, maximum yaw rate are still guaranteed by the constraints in Eqn. 7. So the generated control commands are still safe and comfortable.

The box plot in Fig. 17 shows the computation time distribution across the 200 Monte-Carlo simulations. The average computation time is 19.3ms, and 97.2% of the computation time is less than 37.8ms. The worst case can still output optimal control commands within 60ms. So the proposed method can still achieve real-time performance even with the model uncertainty.

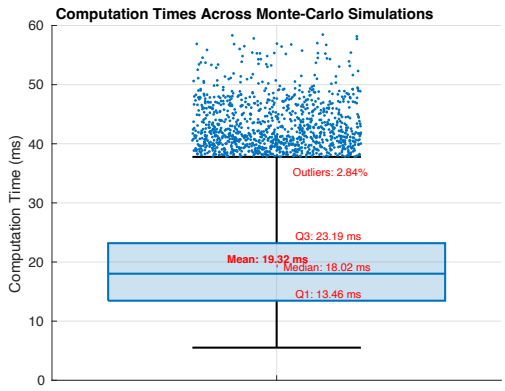



Fig. 17. Computation times across Monte-Carlo simulations.

## V. CONCLUSIONS AND FUTURE WORKS

In this research, a fast and safe motion planning method that generates optimal longitudinal and lateral control commands based on the potential field built from point cloud was proposed. The innovation lies in the method's capability to swiftly detect obstacles with high accuracy, construct the drivable area boundaries, and avoid local optima, enhancing safety and efficiency. The proposed method to build PF from point cloud is fast, and ensures all obstacles can be detected. It can address the problem of the ego vehicle becoming trapped in local optima by constructing PF based on the drivable area boundaries. The MPC was used to model the motion planning due to its capacity to consider vehicle dynamics, constraints, collision avoidance and comfort simultaneously. To solve this complex motion planning model, an iterative way that uses Frenet based planner as initial guess of the optimal solution was proposed.

Experimental results show the proposed generation of PF from point cloud can handle small objects and discrepancies in HD map within 20ms with a modest 4% of the CPU capacity. The simulations underscored the method's effectiveness in dynamic scenarios, showcasing a significant reduction in computational time and maintaining smooth and consistent control commands even in the presence of sudden, unpredictable maneuvers by other vehicles. The maintenance of a safe distance under varying conditions and the method's computational efficiency, particularly in real-time applications, are notable achievements of this research.

In future work, the obstacle's motion will be estimated using the historical positions to have a better obstacle prediction instead of assuming a constant velocity model. The theoretical uncertainty analysis will be done and the experimental verification of the developed motion planning method will be conducted on a real car.

## ACKNOWLEDGMENTS

The authors would like to acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada.

## REFERENCES

[1] C. Sun, R. Zhang, Y. Lu, Y. Cui, Z. Deng, D. Cao, and A. Khajepour, "Toward ensuring safety for autonomous driving perception: Standardization progress, research advances, and perspectives," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 5, pp. 3286–3304, 2024.

[2] C. Sun, Y. Cui, N.-D. Dào, R. V. Mehrizi, M. Pirani, and A. Khajepour, "Medium-fidelity evaluation and modeling for perception systems of intelligent and connected vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1711–1721, 2024.

[3] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7464–7475.

[4] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 8, pp. 2647–2664, 2020.

[5] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[6] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[7] M. Himmelsbach, F. V. Hundelshausen, and H.-J. Wuensche, "Fast segmentation of 3d point clouds for ground vehicles," in *2010 IEEE Intelligent Vehicles Symposium*. IEEE, 2010, pp. 560–565.

[8] X. Zhang, W. Xu, C. Dong, and J. M. Dolan, "Efficient l-shape fitting for vehicle detection using laser scanners," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 54–59.

[9] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[10] J. P. Rastelli, R. Lattarulo, and F. Nashashibi, "Dynamic trajectory generation using continuous-curvature algorithms for door to door assistance vehicles," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, pp. 510–515.

[11] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *Robotics Science and Systems VI*, vol. 104, no. 2, 2010.

[12] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 4889–4895.

[13] R. Chai, K. Chen, L. Cui, S. Chai, G. Inalhan, and A. Tsourdos, *Review of Advanced Trajectory Optimization Methods*. Singapore: Springer Nature Singapore, 2023, pp. 3–42.

[14] R. Chai, A. Tsourdos, H. Gao, Y. Xia, and S. Chai, "Dual-loop tube-based robust model predictive attitude tracking control for spacecraft with system constraints and additive disturbances," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 4, pp. 4022–4033, 2022.

[15] R. Chai, A. Tsourdos, H. Gao, S. Chai, and Y. Xia, "Attitude tracking control for reentry vehicles using centralised robust model predictive control," *Automatica*, vol. 145, p. 110561, 2022.

[16] H. Wang, Y. Huang, A. Khajepour, Y. Zhang, Y. Rasekhipour, and D. Cao, "Crash mitigation in motion planning for autonomous vehicles," *IEEE transactions on intelligent transportation systems*, vol. 20, no. 9, pp. 3313–3323, 2019.

[17] Y. Rasekhipour, A. Khajepour, S.-K. Chen, and B. Litkouhi, "A potential field-based model predictive path-planning controller for autonomous road vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1255–1267, 2016.

[18] R. Chai, A. Tsourdos, S. Chai, Y. Xia, A. Savvaris, and C. L. P. Chen, "Multiphase overtaking maneuver planning for autonomous ground vehicles via a desensitized trajectory optimization approach," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 1, pp. 74–87, 2023.

[19] M. A. Daoud, M. W. Mehrez, D. Rayside, and W. W. Melek, "Simultaneous feasible local planning and path-following control for autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16 358–16 370, 2022.

[20] R. Chai, H. Niu, J. Carrasco, F. Arvin, H. Yin, and B. Lennox, "Design and experimental validation of deep reinforcement learning-based fast trajectory planning and control for mobile robot in unknown environment," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2022.

[21] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, and C. L. P. Chen, "Design and implementation of deep neural network-based control for automatic parking maneuver process," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 4, pp. 1400–1413, 2022.

[22] R. Chai, D. Liu, T. Liu, A. Tsourdos, Y. Xia, and S. Chai, "Deep learning-based trajectory planning and control for autonomous ground vehicle parking maneuver," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 3, pp. 1633–1647, 2023.

[23] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 2, pp. 952–964, 2016.

[24] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 987–993.

[25] B. Peng, D. Yu, H. Zhou, X. Xiao, and C. Xie, "A motion planning method for automated vehicles in dynamic traffic scenarios," *Symmetry*, vol. 14, no. 2, p. 208, 2022.

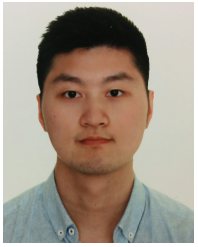
[26] A. Takahashi, T. Hongo, Y. Ninomiya, and G. Sugimoto, "Local path planning and motion control for agv in positioning," in *Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems'.(IROS'89)'The Autonomous Mobile Robots and Its Applications*. IEEE, 1989, pp. 392–397.

[27] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, p. 226–231.


**Minghao Ning** received the B.Eng. degree in vehicle engineering from the Beijing Institute of Technology, Beijing, China, in 2020. He is currently pursuing the Ph.D. degree under the supervision of Professor Amir Khajepour and Professor Ehsan Hashemi with the Department of Mechanical and Mechatronics Engineering, University of Waterloo. His research interests include multi-sensor fusion based 3D occupancy map generation, motion planning with Model Predictive Control and cooperative perception system.

**Amir Khajepour** (Senior Member, IEEE) held the Tier 1 Canada Research Chair in Mechatronic Vehicle Systems from 2008 to 2022 and the Senior NSERC/General Motors Industrial Research Chair in Holistic Vehicle Control from 2017 to 2022. He is currently a Professor of mechanical and mechatronics engineering and the Director of the Mechatronic Vehicle Systems (MVS) Laboratory, University of Waterloo. His work has resulted in training of over 150 Ph.D. and M.A.Sc. students, 30 patents, more than 600 research articles, numerous technology transfers, and several start-up companies. He is a fellow of the Engineering Institute of Canada, the American Society of Mechanical Engineering, and the Canadian Society of Mechanical Engineering. He has been recognized with the Engineering Medal from Professional Engineering Ontario.

**Ehsan Hashemi** (Senior Member, IEEE) received the M.Sc. degree in mechanical engineering from the Amirkabir University of Technology (Tehran Polytechnic) in 2005 and the Ph.D. degree in mechanical and mechatronics engineering from the University of Waterloo, ON, Canada, in 2017. He is currently an Assistant Professor with the Department of Mechanical Engineering, University of Alberta. His research interests include robotics, control theory, distributed estimation, and human–robot interaction.



**Chen Sun** received the Ph.D. degree in Mechanical & Mechatronics Engineering from University of Waterloo, ON, Canada in 2022, M.A.Sc degree in Electrical & Computer Engineering from University of Toronto, ON, Canada in 2017 and B.Eng. degree in automation from the University of Electronic Science and Technology of China, Chengdu, China, in 2014. He is currently an Assistant Professor with the Department of Data and Systems Engineering, University of Hong Kong. His research interests include field robotics, safe and trustworthy autonomous driving and in general human-CPS autonomy.