Safe Motion Planning for Autonomous Vehicles by Quantifying Uncertainties of Deep Learning-enabled Environment Perception

Dachuan Li, *Member, IEEE,* Bowen Liu, Zijian Huang, Qi Hao, *Member, IEEE*, Dezong Zhao, *Senior Member, IEEE*, and Bin Tian, *Member, IEEE*

***Abstract*—Conventional perception-planning pipelines of autonomous vehicles (AV) utilize deep learning (DL) techniques that typically generate deterministic outputs without explicitly evaluating their uncertainties and trustworthiness. Therefore, the downstream decision-making components may generate unsafe outputs leading to system failure or accidents, if the preceding perception component provides highly uncertain information. To mitigate this issue, this paper proposes a coherent safe perception-planning framework that quantifies and transfers DL-based perception uncertainties. Following the Bayesian Deep Learning paradigm, we design a probabilistic 3D object detector that extracts objects from LiDAR point clouds while quantifying the corresponding aleatoric and epistemic uncertainty. A chance-constrained motion planner is designed to formulate an explicit link between DL-based perception uncertainties and operation risk and generate safe and risk-bounding trajectories. The proposed framework is validated through various challenging scenarios in the CARLA simulator. Experiment results demonstrate that our framework can effectively capture the uncertainties in DL, and generate trajectories that bound the risk under DL perception uncertainties. It also outperforms counterpart approaches without explicitly evaluating the uncertainties of DL-based perception.**

***Index Terms*—Autonomous driving, motion planning, object detection, Bayesian Deep Learning, uncertainty quantification.**

I. Introduction

SAFETY is one of the primary goals in the design and development of autonomous driving vehicles (AV). Deep learning (DL) techniques have emerged as a common setting for the software stack of AV systems, due to their effectiveness in feature extraction and environment state predictions after training with annotated data [1]. However, developing a DL-enabled AV perception-planning pipeline with trustworthiness has to deal with new technical challenges.

In a typical AV functional pipeline, the downstream decision-making and motion planning modules operate based on the output of upstream environment perception components [2], [3]. Therefore, errors and uncertain outputs of DL-based environment perception will propagate downstream and ulti-

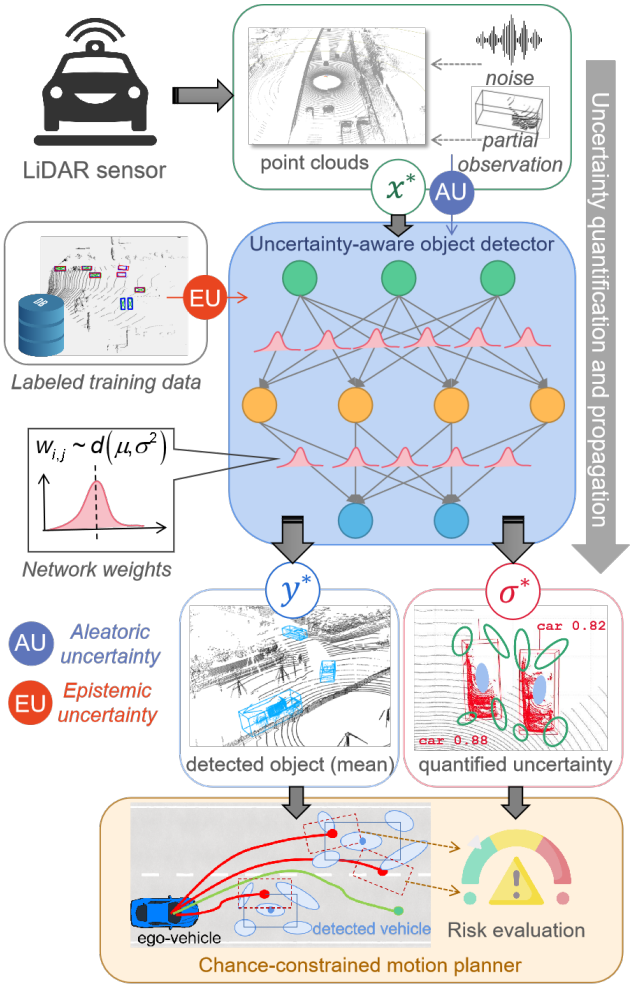


Fig. 1. Framework of the proposed uncertainty-aware 3D object detection and chance-constrained motion planning system for autonomous driving. An explicit link is established between quantified uncertainties in Deep Learning-based perception and risk in motion planning

This work was supported in part by the National Natural Science Foundation of China under Grants 52272419 and 62261160654, in part by the Science and Technology Innovation Committee of Shenzhen City under Grants JCYJ20200109141622964 and JCYJ20220818103006012, and in part by the National Key Research and Development Program of China under Grant 2022YFB4703700. (*Corresponding author: Qi Hao*)

D. Li and Q. Hao are with the Research Institute of Trustworthy Autonomous Systems, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: dachuanli86@gmail.com, haoq@sustech.edu.cn).

B. Liu, Q. Hao and Z. Liu are with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, 518057, China (e-mail:{11849325, 11510163}@mail.sustech.edu.cn)

D. Zhao is with James Watt School of Engineering, University of Glasgow, Glasgow, G12 8QQ, United Kingdom (e-mail: dezong.zhao@glasgow.ac.uk)

B. Tian is with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, and with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: bin.tian@ia.ac.cn)

mately affect the subsequent decision-making results. However, existing DL networks generally output deterministic state predictions without explicitly measuring their confidence [4], but the downstream decision-making units simply 'trust' these estimations even if they are highly untrustworthy nevertheless. One example of such circumstances is the fatal accident of Uber AV in Arizona [5], where the victim was detected by the perception system and classified vacillatingly as an unknown object, a vehicle, and finally a bicycle prior to the crash. Therefore, Without evaluating the uncertainty of perception, the decision-making modules can not generate proper actions accordingly to ensure operation safety.

Therefore, it is crucial that the perception neural network not only outputs accurate information about environments but also provides explicit measures of the level of uncertainties in its outputs. In addition, how to deal with the propagated perception uncertainties in the downstream decision-making stage to ensure operation safety is still a challenging issue.

To address these challenges, this paper proposes a novel coherent uncertainty-aware framework for planning safe trajectories for AVs by quantifying and propagating uncertainties in DL-based perception (Fig.1). The main contributions of this work include:

- Developing a *uncertainty-aware 3D object detector (Bayesian Point-RCNN)* which can estimate bounding boxes of surrounding vehicles from 3D LiDAR point clouds with explicitly quantified epistemic and aleatoric uncertainties in both classification and regression.
- Developing an *chance-constrained safe motion planning approach (perception uncertainty-aware-RRT, PU-RRT)* that builds an explicit link between quantified uncertainties in the DL-based object detector and operation risk via spatial chance constraints, and checks the probabilistic feasibility of trajectories to bound the risk under DL perception uncertainties.
- Performing comprehensive *evaluations* using real-world data and simulated driving scenarios involving various factors that cause perception uncertainties. We show an improvement in safety by explicitly modeling and incorporating perception uncertainties of DL.

The remainder of this paper is organized as follows: Section II provides an overview of related work in the related domains. Section III presents the system architecture and problem formulations. Section IV provides a detailed description of the 3D object detector with quantification of perception uncertainties. The uncertainty-aware motion planning approach is presented in Section V. Section VI provides experiment results, and the paper is finally concluded in Section VII.

II. Related Work

A. Quantifying Uncertainties in Deep Learning

Uncertainties in a Deep Neural Network can be decomposed into two sources: the *epistemic* uncertainty (EU) and the *aleatoric* uncertainty (AU) [6]. The EU captures the inherent uncertainty of model parameters due to the mismatch between the training and inference input data, while AU is typically caused by randomness and incompleteness of observations(e.g. sensor noises). Uncertainty quantification(UQ) approaches for DL can be generally divided into three major categories: *Monte-Carlo (MC) dropout*, *deep ensembles*, and *direct modeling*.

Bayesian Neural Network(BNN) [7] provides a principled paradigm for representing uncertainties in neural networks based on the Bayesian theory. Instead of using deterministic values of weights, the BNN framework places a prior probabilistic distribution over the weights of networks, and the uncertainties and be estimated by inference of associated posterior distributions. Since performing direct inference of such posterior distributions is intractable, sampling-based techniques have been proposed as approximation methods. Sampling-based Variation inference (VI) approaches estimate EU by approximating distributions that minimize the Kullback-Leibler divergence from the actual distributions over network weights [8], [9]. For Monte-Carlo dropout-based approaches [10], [11], samples are generated from the approximated posterior distributions of weights by performing multiple inferences with dropout. Instead of using the same network architecture, they add dropout layers and obtain different predictions by randomly dropping out neurons for each forward pass. In addition, the deep ensemble methods [12], [13] pass a single input through an ensemble of networks to estimate the predictive variances.

As for AU, direct modeling approaches use additional output layers to directly estimate the parameters of the distribution of the network outputs.[14], [15], [16]. These approaches require modification of the loss function and incorporation additional output layer. Usually, direct modeling approaches may have difficulties in converging for those detectors with large output spaces. (*c.f.* [17] for a more comprehensive survey of UQ of general DL techniques.)

More recent work has attempted to quantify uncertainties in DL-based 3D object detection frameworks in AV applications. In [18], the MC dropout technique is utilized to quantitatively model the EU for the DL-based 3D vehicle detection tasks. They demonstrate an improvement in the accuracy of detection by modeling the uncertainties. It is further extended in [19] to quantify the heteroscedastic AU in a regional proposal-based LiDAR 3D object detector. The proposed detector achieves improved accuracy and robustness against noisy data by biasing the training to more informative data. Similarly, [6] extends the LaserNet detector to estimate the uncertainty in the labels of annotated data and learn distributions of vehicle bounding boxes.

However, most of the current UQ techniques of DL are designed only with the objective of improving the robustness and accuracy of detection. They typically ignore how to propagate the DL perception uncertainties to the downstream decision-making stage to ensure operation safety, which is the motivation of our work in this paper.

B. Decision-making and Planning under Perception Uncertainties

Decision-making and planning under sensing uncertainties in the domain of AV can be typically formulated as Partially

Observable Markov Decision Process (POMDP) problems and desired policies can be generated by optimizing a reward function that considers the costs and risks of actions. Recent research attempts to incorporate uncertainties caused by sensory noise [20] and occlusion/partial perception [21], [22] in motion planning. However, real-world applications of POMDP frameworks to AV problems with continuous space are still very limited, as solving the POMDP is computationally intensive.

Alternatively, *sampling-based motion planners* have been validated as effective in planning dynamically-feasible trajectories for AVs. The *Rapidly-exploring Random Trees(RRT)*) has been extended to deal with uncertainties. For instance, the Chance-Constrained RRT (CC-RRT) [23] (later extended by CC-RRT* [24]) models the sensing uncertainties for checking the probabilistic feasibility of trajectories. In addition, the Rapidly-exploring Random Belief Tree (RRBT) [25] extends the RRT* framework by propagating both state and sensing uncertainties to bound the risk of operation.

It is worth mentioning that [26] proposes a unified framework with similar motivation to our work. It models the epistemic uncertainty caused by out-of-domain scenarios and stochasticity in demonstration data. A robust imitation learning-based planning method is designed on that basis to improve the safety of AVs. In addition, the more recent research in [27] utilizes MC-dropout and ensemble-based techniques to evaluate the uncertainties in the prediction of the behavior of the ego vehicle, and the uncertainties are leveraged for the threat assessment of lane-keeping assistance.

In spite of the effectiveness achieved by the above approaches, they typically assume simplified and unrealistic model of the perception uncertainty and their feasibility for systems involving uncertainties of DL has not been validated.

III. Preliminaries

A. Overall Framework

This work considers a common AV architecture, where a vehicle-mounted LiDAR sensor captures 3D point clouds of the operation environments, and the perception component of the AV software stack uses such measurements to estimate states of the environment for decision-making. Fig. 1 illustrates the framework of the proposed system consisting of two major components: an uncertainty-aware 3D LiDAR object detector and a chance-constrained motion planner.

The proposed *uncertainty-aware 3D LiDAR object detector* utilizes a DL-based model (*Probabilistic PointRCNN*) to recognize surrounding traffic participants and extract their 3D bounding boxes from LiDAR point clouds, while explicitly modeling the aleatoric and epistemic uncertainties associated with the classification and bounding box regression. Such information is then fed to the proposed *chance-constrained motion planner* (*PU-RRT*) to generate safe and probabilistic feasible reference trajectories that navigate the vehicle to the goal. To explicitly incorporate the quantified DL perception uncertainties, the detected object is formulated as a probabilistic spatial representation, based on which the PU-RRT evaluates the risk of collision and checks the probabilistic feasibility while growing the tree of trajectories. In this manner, the motion planner can generate risk-bounding trajectories under DL-based perception uncertainties. Methodologies of the proposed perception and planning components in the framework are detailed in subsequent sections.

B. Preliminaries

1) Quantifying Uncertainties of Deep Learning via Dropout approximation: Following the Bayesian deep learning principle, the epistemic uncertainty can be modeled by placing prior distributions over the network's weights and estimating the variation of weights with respect to the given data.

Denote $\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\}$ as an annotated training dataset ($\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ denotes the data sample, $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^N$ is the ground truth label). A DNN is a model $f^{\mathbf{W}}(\cdot)$ with weight parameters $\mathbf{W} = \{\mathbf{w}_j\}_{j=1}^L$ trained using $\mathcal{D}$. During inference, $f^{\mathbf{W}}(\cdot)$ maps an data input $\mathbf{x}^*$ to its expected prediction $\mathbf{y}^*$. In the BDL framework, given an input sample $\mathbf{x}^*$, the network infers a predictive posterior distribution [28]:

$$p\left(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}\right) = \int p(\mathbf{y}^*|f^{\mathbf{W}}(\mathbf{x}^*))p(\mathbf{W}|\mathcal{D})d\mathbf{w}, \tag{1}$$

where $p(\mathbf{W}|\mathcal{D})$ denote the posterior distribution of weight over the training dataset, and $p(\mathbf{y}^*|f^{\mathbf{W}}(\mathbf{x}^*))$ is the model likelihood. Theoretically, the EU can be evaluated by performing Bayesian inference to estimate the posterior distribution. However, it is intractable to analytically infer $p(\mathbf{W}|\mathcal{D})$ [10]. Therefore, practical approaches typically utilize various techniques to estimate an approximate posterior distribution $\hat{q}^{\theta}(\mathbf{W})$ of $p(\mathbf{W}|\mathcal{D})$, and the predictive output of the DNN turns to:

$$q\left(\mathbf{y}^*|\mathbf{x}^*, \mathbf{W}\right) = \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{W})q(\mathbf{W})d\mathbf{W} \tag{2}$$

It is shown in [10] that training a DNN model with MC dropout is equivalent to approximating the posterior distribution in Eq.(1). For inference, the trained network operates by performing multiple forward passes with dropout to generate samples from the approximate distribution ($\mathbf{W}_k \sim \hat{q}^{\theta}(\mathbf{W}), k \in \{1...N\}$).

Considering the ith intermediate hidden layer L_i (with m_i hidden units) of a DNN. Denote $\mathbf{W}_i$, $\mathbf{W}_{i+1}$ as the weight matrices connecting its preceding hidden layer (L_{i-1}, with m_{i-1} hidden units) and succeeding hidden layer (L_{i+1}, with m_{i+1} hidden units), respectively($\mathbf{W}_i \in \mathbb{R}^{m_{i-1} \times m_i}$, $\mathbf{W}_{i+1} \in \mathbb{R}^{m_i \times m_{i+1}}$). For L_i, dropout operates by sampling two random vectors $\mathbf{r}_i$, $\mathbf{r}_{i+1}$ with elements distributed according to Bernoulli distribution (i.e. $r_{i,j} \sim Bernoulli(p_i), r_{i+1,k} \sim Bernoulli(p_{i+1}), j \in [1...m_{i-1}], k \in [1...m_{i+1}], p_i, p_{i+1} \in [0, 1]$. For each forward pass or backward propagation, by multiplying weight matrices with random vectors ($\mathbf{r}_i\mathbf{W}_i, \mathbf{r}_{i+1}\mathbf{W}_{i+1}$), part of the weight are stochastically set to zero and the corresponding hidden units are thus dropped.

Taking advantage of the Monte Carlo dropout approximation, the predictive variance of the NN model for regression tasks can be estimated by:

$$\begin{aligned}&\sigma_{\hat{q}(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*) = \mathbb{E}_{\hat{q}(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^{*T}\mathbf{y}^*)\\ &- \mathbb{E}_{\hat{q}(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*)^T\mathbb{E}_{\hat{q}(\mathbf{y}^*|\mathbf{x}^*)}(\mathbf{y}^*)\\ &\approx \sigma^2 + \frac{1}{T}\sum_{t=1}^{T} f^{\hat{\mathbf{W}}_t}(\mathbf{x}^*)^T f^{\hat{\mathbf{W}}_t}(\mathbf{x}^*) - \frac{1}{T}\sum_{t=1}^{T} f^{\hat{\mathbf{W}}_t}(\mathbf{x}^*)\end{aligned} \tag{3}$$

where T is the number of stochastic forward passes, the weights $\hat{\mathbf{W}}_t$ of each forward pass are sampled from the approximated distribution: $\hat{\mathbf{W}}_t \sim \hat{q}^{\theta}(\mathbf{W})$.

On the other hand, the aleatoric uncertainty (σ^2 in Eq. 3) can be quantified by modeling the observation likelihood $p(\mathbf{y}^*|f^{\mathbf{W}}(\mathbf{x}^*))$ in Eq.(1) via direct modeling (Section IV).

2) Chance-constrained motion planning: In contrast to deterministic motion planning, the DNN-based perception unit outputs estimated states of surrounding entities with uncertainties. In this work, given the probabilistic description of environment states, the major objective of chance-constrained motion planning is to generate optimized and probabilistic-feasible trajectories that bound the risk of collision. We formulate such a problem as:

Problem 1 (chance-constrained optimal motion planning under perception uncertainties). Given an agent with the following dynamics model: $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t)$ ($\mathbf{x}_t \in \mathcal{X}$: state vector, $\mathbf{u}_t \in \mathcal{U}$: control vector at time t), an initial state $\mathbf{x}_{init}$, a goal state $\mathbf{x}_{goal}$, and the set of detected obstacles $\mathcal{B}_t = \{\mathbf{b}_{i,t}\}_{i=1}^{N_{obs}}$ with uncertainties, find the sequence of states and control policy: $\pi^* = \{\mathbf{x}_t, \mathbf{u}_t\}, (t \in [0, \Delta t])$ in Δt that navigates the agent from $\mathbf{x}_{init}$ to $\mathbf{x}_{goal}$, which minimizes the cost of executing the sequence:

$$\pi^* = \arg\min_{\pi} \sum_{t}^{t+\Delta t} J(\pi_t, \mathcal{B}_t), \tag{4}$$

while ensuring the following chance constraints are satisfied:

$$P(\texttt{cls}|\pi^*, \mathcal{B}) = 1 - P\left(\bigwedge_{t}^{t+\Delta t} \neg\texttt{cls}|\pi^*, \mathcal{B}\right) < \Delta, \tag{5}$$

where `cls` indicates a collision with an arbitrary obstacle in the environment. Δ denotes a predefined threshold on the probability of collision.

IV. Quantifying Uncertainty in Point Cloud-based 3D Object Detection Neural Network

A. Network architecture

The framework of the proposed probabilistic 3D vehicle detector (Bayesian-PointRCNN) is shown in Fig. 2. We adopt the two-stage backbone of the PointRCNN detector[29], which consists of a region proposal network (RPN, stage 1) and a subnetwork for bounding box refinement and uncertainty quantification (stage 2).

The inputs of the stage 1 RPN are raw 3D point clouds captured by LiDAR. The RPN of the Bayesian-PointRCNN directly generates regional proposals (RP) of 3D bounding boxes from point clouds by segmenting 3D points into foreground and background points. Outputs of the RPN are the learned point-wise feature vectors as well as the 3D bounding box region proposals for each detected vehicle (represented by $\hat{\mathbf{b}}_i = [\hat{x}_i, \hat{y}_i, \hat{z}_i, \hat{h}_i, \hat{w}_i, \hat{l}_i, \hat{\theta}_i]^T$, where x, y, z denote the location, h, w, l represent the 3D scale of the box, and θ denote the orientation.)

For the stage 2 sub-network, a pooling procedure is first performed on the 3D points and their features learned by RPN. After pooling, points within 3D proposals are selected with its local point-wise features: $\mathbf{p} = [\ x_p, y_p, z_p, r_p, m_p, \mathbf{f}_p]$ (where x_p, y_p, z_p denote the 3D coordinates of the point, r_p represents intensity, m_p is the foreground/background segmentation mask, and $\mathbf{f}_p$ is the feature extracted in RPN). The local spatial features $\tilde{\mathbf{p}}$ are merged with global semantic feature $\mathbf{f}_p$ from RPN and fed to another encoder to generate rich feature vectors for 3D bounding box refinement and uncertainty modeling. The ultimate outputs of the proposed network consist of the refined bounding boxes for each detected object in the frame: $\{\mathbf{b}_i\}$ ($\mathbf{b}_i = [x_i, y_i, z_i, h_i, w_i, l_i, \theta_i]^T$), the corresponding classification results:$\{c_i\}$ (indicating the category: car, van, truck, etc.) along with its softmax score $\{Softmax_i(f^{\hat{\mathbf{W}}_t})\}$, as well as the quantified AU, EU and hybrid spatial uncertainty.

To quantitatively model the uncertainties, we extend the original network by introducing two major components: *intermediate layers with dropout* for modeling epistemic uncertainty, and *AU quantification layers* for modeling aleatoric uncertainty.

B. Intermediate Layers with Monte-Carlo Dropout

To quantify the EU, we utilize the MC-dropout approximation technique by introducing intermediate layers to the stage 2 sub-network (Fig.2). 3 fully connected layers (FC) are added, where the first two FC layers consist of 512 and 1024 hidden units, respectively, and the number of units of the output FC layer fits the input dimension of the subsequent proposal refinement sub-network. We incorporate a dropout layer to follow each FC layer and perform Monte-Carlo sampling of the posterior distribution of the model's weights. Following the MC dropout principle (Section III B), T stochastic forward passes are performed for each input frame to generate samples of weights from the approximated distribution $\hat{q}(\mathbf{W})$. Utilizing the outputs of intermediate layers, the EU of regression and classification tasks are modeled as follows:

1) Quantifying regression epistemic uncertainty: The regression task relates to the estimation of 3D bounding boxes of surrounding entities. Therefore, the regression EU represents the uncertainty of the network output $\{\mathbf{b}_i\}$ from a spatial perspective. Given the input $\mathbf{x}^*$and the bounding box output of T forward pass $\mathbf{y}^* = \{\hat{\mathbf{b}}_{i,t}\}$, $(t = 1...T)$, and following Eq. (3), the regression EU can be quantified by calculating the covariance matrix:

$$\mathbf{S}_i(\mathbf{y}^*_{reg}) = \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{b}}_{i,t}\hat{\mathbf{b}}_{i,t}^T - \left(\frac{1}{T}\sum_{t=1}^{T}\hat{\mathbf{b}}_{i,t}\right)\left(\frac{1}{T}\sum_{t=1}^{T}\hat{\mathbf{b}}_{i,t}\right)^T \tag{6}$$

where i indicates the index of the detected object. Therefore, under the assumption of Gaussian distributions, the diagonal elements of the matrix $\mathbf{S}_i$ represent the corresponding

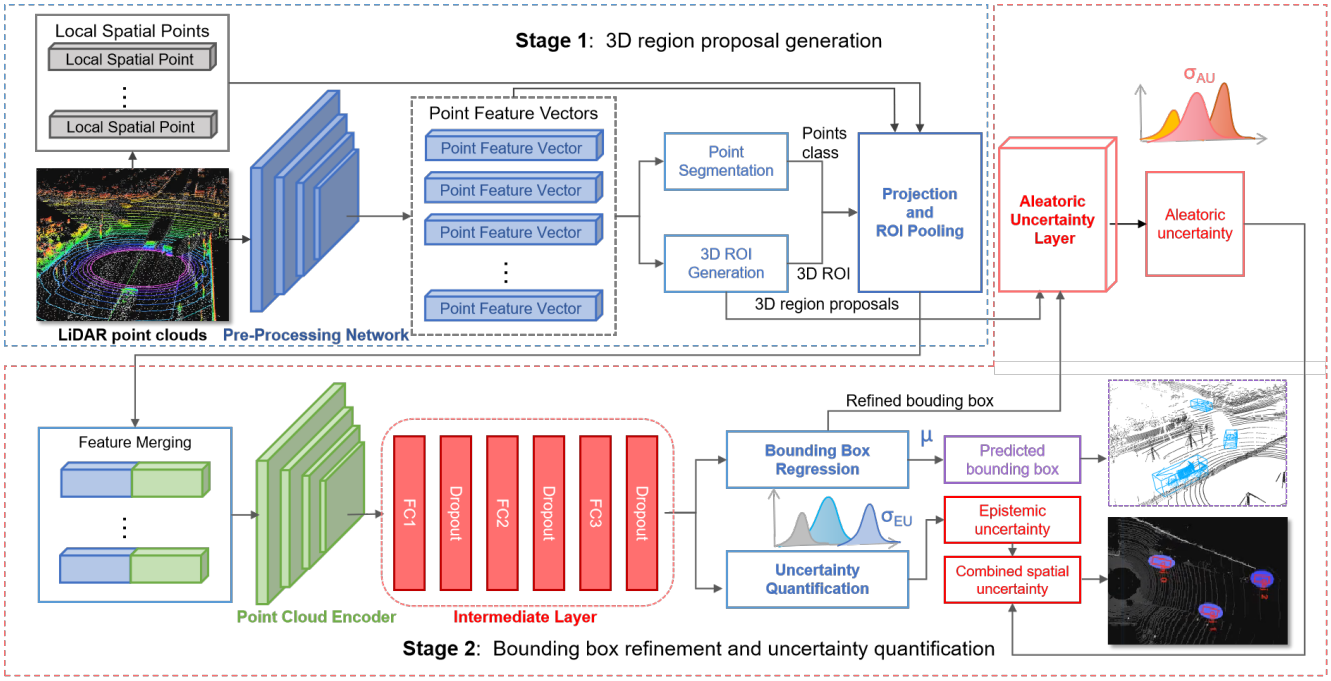


Fig. 2. The Bayesian PointRCNN architecture for 3D object detection and uncertainty quantification from point clouds. Uncertainty quantification-related components are marked in pink (ROI: region of interest; FC: fully-connected layer).

epistemic uncertainty of each dimension in the estimated 3D bounding box vector, e.g.:

$$\mathbf{Var}_i^{reg} = \mathtt{diag}(\mathbf{S}_i) = \left[\sigma_{x_i}^2, \sigma_{y_i}^2, \sigma_{z_i}^2, \sigma_{h_i}^2, \sigma_{w_i}^2, \sigma_{l_i}^2, \sigma_{\theta_i}^2\right]^T \quad (7)$$

2) Quantifying classification epistemic uncertainty: The classification EU represents the network's confidence in categorizing detected objects. Similar to [7], we employ Shannon Entropy (SE) and Mutual Information (MI) as metrics for the evaluation of the classification EU. Both MI and EU are evaluated based on the the probability that a detected object (obj_i) takes a certain category $c_j \in \mathcal{C}$(e.g. vehicle: c_{veh}). In the proposed network, such probability can be calculated as the mean of the softmax score of T forward passes:

$$p\left(y_i = c_j|\mathbf{x}^*\right) = \frac{1}{T}\sum_{t=1}^{T}\mathrm{Softmax}(c_j, t) \quad (8)$$

Denoting $\sigma_{\mathrm{sftmx}}(c)$ as the softmax score, the SE can be calculated as:

$$\begin{aligned}\mathbb{H}(\mathbf{y}_{cls}^*|\mathbf{x}^*) &= -\sum_{c_j\in\mathcal{C}} p(y=c_j|\mathbf{x}^*,\mathcal{D})\log p(y=c_j|\mathbf{x}^*,\mathcal{D}) \\ &\approx -\sum_{c_j\in\mathcal{C}}\left(p(y=c_j|\mathbf{x}^*,\hat{\mathbf{W}}_t)\right)\log\left(p(y=c_j|\mathbf{x}^*,\hat{\mathbf{W}}_t)\right) \\ &= -\sum_{c_j\in\mathcal{C}}\left(\frac{1}{T}\sum_{t=1}^{T}\sigma_{\mathrm{sftmx}}(c_j,t)\right)\log\left(\frac{1}{T}\sum_{t=1}^{T}\sigma_{\mathrm{sftmx}}(c_j,t)\right)\end{aligned} \quad (9)$$

SE reflects the model's confidence in its prediction results, as it reaches a high value when all classes are predicted to have similar probability, indicating the network model is very uncertain about its classification results.

The MI is calculated as:

$$\begin{aligned}&\mathbb{I}(\mathbf{y}_{cls}^*, \mathbf{W}|\mathbf{x}^*, \mathcal{D}) = \mathbb{H}(\mathbf{y}^*|\mathbf{x}^*) - \mathbb{E}_{p(\mathbf{W}|\mathcal{D})}(\mathbb{H}(\mathbf{y}^*|\mathbf{x}^*)) \\ &= -\sum_c\left(\frac{1}{T}\sum_{t=1}^{T}\sigma_{\mathrm{sftmx}}(c_j,t)\right)\log\left(\frac{1}{T}\sum_{t=1}^{T}\sigma_{\mathrm{sftmx}}(c_j,t)\right) \\ &+ \frac{1}{T}\sum_t\sum_c\sigma_{\mathrm{sftmx}}(c_j,t)\log\sigma_{\mathrm{sftmx}}(c_j,t)\end{aligned} \quad (10)$$

MI evaluates the network's confidence in its outputs by measuring the information difference over multiple inferences, and it shows a high magnitude when the network outputs from multiple trials are diverse. The quantified PE and MI can be used as metrics to filter out incorrect detection in motion planning, as they tend to come with high uncertainties.

C. Aleatoric Uncertainty Quantification

The observation noise-related aleatoric uncertainty can be quantified by direct modeling the observation likelihood $p\left(\mathbf{y}^*|f^{\mathbf{W}}(\mathbf{x}^*)\right)$. For the classification task, this is achieved using the softmax function ($p\left(\mathbf{y}^*|f^{\mathbf{W}}(\mathbf{x}^*)\right) = \mathrm{Softmax}(\mathrm{f}^{\mathbf{W}}(\mathbf{x}^*))$). For the 3D bounding box regression task, we need to estimate the data-related parameter σ as in Eq. (3). For a fixed Gaussian likelihood [15], $p\left(\mathbf{y}^*|f^{\mathbf{W}}(\mathbf{x}^*)\right)$ can be modeled as a multi-variant Gaussian distribution with a diagonal covariance matrix:

$$\begin{aligned}p(\mathbf{y}^*|\mathbf{x}^*,\boldsymbol{\omega}) &\sim \mathcal{N}(f^{\mathbf{W}}, \boldsymbol{\Sigma}(\mathbf{x}^*)), \\ \boldsymbol{\Sigma}(\mathbf{x}^*) &= \mathrm{diag}(\boldsymbol{\sigma}_{\mathbf{y}^*}^2),\end{aligned} \quad (11)$$

where $\mathbf{y}^*$ is the predicted bounding box which is represented by a 7-dimensional vector. Therefore, the noise parameters from $\boldsymbol{\Sigma}(\mathbf{x}^*)$ are encoded as a vector $\boldsymbol{\sigma}_{\mathbf{x}^*}^2 =$

$[\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_h^2, \sigma_w^2, \sigma_l^2, \sigma_\theta^2]^{\mathrm{T}}$, with each element corresponding to the observation noise-related uncertainty in the predicted parameter of the bounding box vector.

It is shown in [15] that the observation noise can be explained as a function of the input data, and thus the network can be trained to directly learn the noise as model loss attenuation. In the proposed network, we model the AU by introducing output layers and training the network with attenuated loss. The structure of the AU layers is depicted in Fig. 3. Considering the uncertainties in both the RPN stage and RP refinement stage contribute to the ultimate AU, both the feature vector of the predicted bounding box and its candidate proposals are taken as input of AU layers. The observation noise parameters of the RPN stage and stage 2 are firstly predicted separately via FC layers. The two sources of uncertainty features are then concatenated and fed to the ultimate output layers to predict the ultimate logarithmic observation noise $\boldsymbol{\lambda}_{\mathbf{x}^*}$ (Note that for numerical stability, we use logarithm term $\boldsymbol{\lambda}_{\mathbf{x}^*} := \log \boldsymbol{\sigma}^2_{\mathbf{x}^*}$ to avoid potential division by zero). The proposed network is trained via the following attenuated loss function:

$$\mathcal{L}_{reg}\left(\mathbf{W}\right) = \frac{1}{T}\sum_{t}^{T}\frac{1}{2}\exp(-\lambda_t)\|\mathbf{y}_t^* - \hat{\mathbf{y}}_t^*\|^2 + \frac{1}{2}\lambda_t \tag{12}$$

where λ_t represents the estimated noise variance of the tth sampling during training. The first term represents the residual loss, and the second term is for regularization.

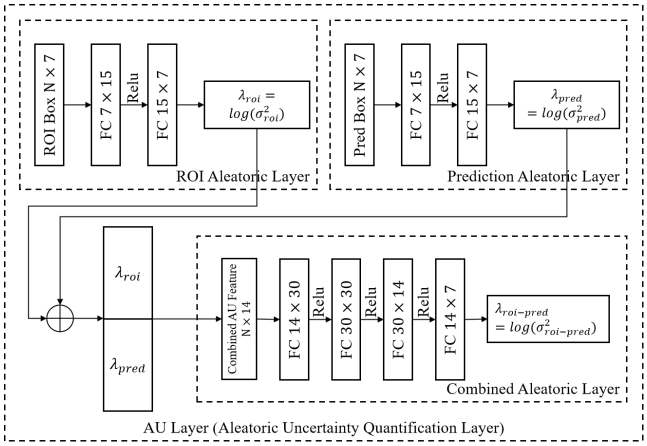


Fig. 3. Layers for modeling the aleatoric uncertainty via loss attenuation.

D. Spatial Representation of Hybrid Perception Uncertainty

To explicitly utilize the quantified perception uncertainties in the motion planning stage, we first convert AU and EU into a fused uncertainty model and formulate it as a spatial representation in the planning space. Following Eq. (3), the quantified EU and AU can be fused into a syndicated uncertainty model of the estimated bounding box:

$$\begin{aligned}\mathbf{Var}_{reg}(\mathbf{y}^*) = &\frac{1}{T}\sum_{t=1}^{T}\mathbf{b}_t\mathbf{b}_t^T - \left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{b}_t\right)\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{b}_t\right)^T \\ &+ \frac{1}{T}\sum_{t}^{T}\hat{\boldsymbol{\sigma}}_t\hat{\boldsymbol{\sigma}}_t^{\mathrm{T}}\end{aligned} \tag{13}$$

The diagonal elements of $\mathbf{Var}_{reg}$ can be extracted into a vector to represent the uncertainties corresponding to the predicted parameters of vehicles, i.e. $\mathbf{Q}_{reg} = [\sigma_x^2, \sigma_y^2, \sigma_z^2, \sigma_h^2, \sigma_w^2, \sigma_l^2, \sigma_\theta^2]^{\mathrm{T}}$. We ignore parameters in the vertical dimension and extract the elements relating to the lateral and longitudinal location ($\boldsymbol{\sigma}_x, \boldsymbol{\sigma}_y$), 2-D scale ($\boldsymbol{\sigma}_w, \boldsymbol{\sigma}_l$), and the orientation ($\boldsymbol{\sigma}_\theta$). Considering the elements in the predicted bounding box vector are mutually-dependent Gaussian variables, variance elements in $\mathbf{Q}_{reg}$ related to the same dimension are additive. Given a detected vehicle's bounding box, we can thus derive the total variance of the lateral and longitudinal dimension by

$$\begin{aligned}\sigma_{lat} &= \sqrt{\sigma_x^2 + \sigma_w^2} \\ \sigma_{lon} &= \sqrt{\sigma_y^2 + \sigma_l^2}\end{aligned} \tag{14}$$

To incorporate the uncertainty in orientation (σ_θ), we convert the estimated angular uncertainty to magnitude variance by projection, and derive the orientation-related additive variance in lateral and longitudinal dimensions:

$$\begin{aligned}\sigma_{\theta,lon} &= \frac{l}{2}\left(1 - w\sqrt{\frac{\left(1+\tan^2(\sigma_\theta)\right)}{\left(w^2 + l^2\tan^2(\sigma_\theta)\right)}}\right) \\ \sigma_{\theta,lat} &= \frac{w}{l}\Delta_a\end{aligned} \tag{15}$$

Therefore, by incorporating the lateral and longitudinal uncertainty, the bounding box of detected vehicles is transformed into a 2-D 2-σ uncertainty ellipse-like formulation (Fig. 4), with the magnitude of lateral and longitudinal axis given by:

$$\begin{aligned}L_a &= \sigma_{lon} + l/2 + \sigma_{\theta,lon} \\ L_b &= \sigma_{lat} + w/2 + \sigma_{\theta,lat}\end{aligned} \tag{16}$$

Larger ellipses indicate higher uncertainty in bounding box estimation. Such a spatial representation provides the basis for chance constraint formulation and allows for simple and efficient probabilistic feasibility check in the motion planning stage, which will be described in detail in the following section.

V. Risk-bounded Safe Motion Planning

A. Chance Constraint and Risk Evaluation

In this section, we describe how to incorporate the quantified spatial uncertainties in DL perception (described in subsection IV-D) to formulate the chance constraints and evaluate the probability of collision during motion planning.

Given the belief of an initial state $\mathcal{N}(\hat{\mathbf{x}}_0, \boldsymbol{\Sigma}_0)$ and the sequence of control inputs $\mathbf{u}_t$, the ego-vehicle can be formulated

as a linear time-invariant model corrupted by additive Gaussian noise [23]:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t + \boldsymbol{\gamma}_t \\ \mathbf{x}_0 &\sim \mathcal{N}(\hat{\mathbf{x}}_0, \boldsymbol{\Sigma}_0) \\ \boldsymbol{\gamma_t} &\sim \mathcal{N}(0, \boldsymbol{\Sigma}_{\boldsymbol{\gamma}}) \end{aligned} \tag{17}$$

where $\mathbf{x}_t$ denotes the state of the ego-vehicle at time t, and A, B are the state-transition matrix and control matrix, respectively. γ_t represent the additive process noise.

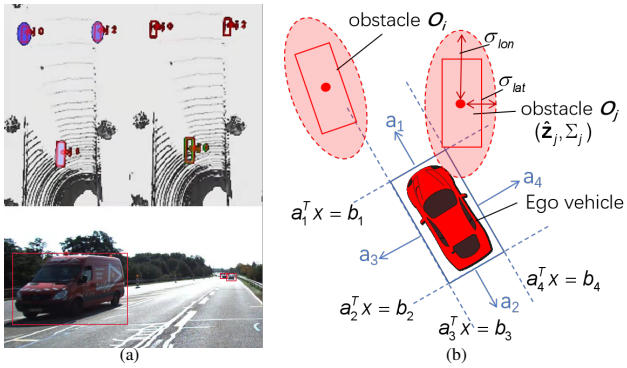


Fig. 4. Risk evaluation based on linear inequalities and 2-σ uncertainty ellipse.

Recall from Eq. (5) that the chance-constrained motion planning requires that the probability of collision between the ego-vehicle and any of the surrounding obstacles at any time step is bounded within a safety threshold δ. Considering the shape of the ego vehicle is perfectly known and the uncertainty in its state is neglectable, the ego vehicle is modeled as a convex polygon specified by a conjunction of four linear inequalities: (Fig. 4):

$$\bigwedge_{i=1,2,3,4} \mathbf{a}_i^T \mathbf{x}_t < \mathbf{b}_i \tag{18}$$

where $\{\mathbf{a}_i\}$ denote unit outer normal vectors of four edges of the minimum convex polygon that bounds the ego-vehicle, and $\mathbf{x}_t$ represent a state in the planning space at t. Given K obstacles (vehicles) detected from 3D point clouds by the probabilistic object detector, and denoting $\mathbf{z}_{j,t}$ as the 2-D coordinate of the geometric center of the jth obstacle's bounding box at time t (represented by the random variable Z_t with covariance Σ_j), a collision with obstacle j only occurs only if all of the four linear constraints in Eq. (18) are satisfied. Therefore, it is true that the probability of colliding with jth obstacle is less than the probability that any one of the four linear constraints being satisfied:

$$\begin{aligned} P_{collision,j} &= P\left(\bigwedge_{i=1,2,3,4} \mathbf{a}_i^T \mathbf{z}_{j,t} < \mathbf{b}_i\right) \\ &\leq P\left(\mathbf{a}_i^T \mathbf{z}_{j,t} < \mathbf{b}_i\right) \\ &\leq \min_{i=1,2,3,4} P\left(\mathbf{a}_i^T \mathbf{z}_{j,t} < \mathbf{b}_i\right) \\ &\forall j \in \{1, ..., K\} \end{aligned} \tag{19}$$

Therefore, to ensure that the overall risk does not exceed the given safe threshold Δ, it is sufficient to require that probability of any of the constraints being satisfied is less than Δ.

It is shown in [30] that the above probabilistic constraint can be converted to deterministic constraints. Defining variables C, c as $C = \mathbf{a}_i^T \mathbf{Z}_t - \mathbf{b}_i$, $c = \mathbf{a}_i^T \mathbf{z}_{j,t} - \mathbf{b}_i$, the above probabilistic constraint is equivalent to the following deterministic constraint [30]:

$$\begin{aligned} &P\left(\mathbf{a}_i^T \mathbf{Z}_t < \mathbf{b}_i\right) = P\left(C < 0\right) \leq \Delta \Leftrightarrow \\ &c \geq \sqrt{2} P_c \mathtt{erf}^{-1}\left(1 - 2\Delta\right) \end{aligned} \tag{20}$$

where $\mathtt{erf}$ denotes the standard error function. It is also proved in [30] that Eq.(20) can be converted to the following equality-based relationship:

$$P\left(C < 0\right) = \bar{p} \Leftrightarrow c = \sqrt{2} P_c \ \mathtt{erf}^{-1}\left(1 - 2\bar{p}\right) \tag{21}$$

Given the Gaussian distribution $\{c, P_c\}$, the probability $\bar{p}$ can be solved based on Eq.(21). Therefore, the probability the ith linear constraint of ego-vehicle being satisfied by the jth obstacle at time t (Δ_{ijt}) can be calculated as:

$$\begin{aligned} \Delta_{ijt} &= P\left(\mathbf{a}_i^T \mathbf{z}_{j,t} - \mathbf{b}_i < 0\right) \\ &= \frac{1}{2}\left(1 - \mathtt{erf}\left(\frac{\mathbf{a}_i^T \mathbf{z}_{j,t} - \mathbf{b}_i}{\sqrt{2 \mathbf{a}_i^T \boldsymbol{\Sigma}_j \mathbf{a}_i}}\right)\right) \end{aligned} \tag{22}$$

where $\boldsymbol{\Sigma}_j$ denotes the lateral and longitudinal variance of the detected obstacle, which is calculated using the quantified uncertainty described in Section IV. Recall that Eq. (19) provides an upper bound of the probability of collision, then for time step t, the overall probability of collision with any of the detected obstacles can be approximated by:

$$\begin{aligned} P_{collision}(t) &\leq \sum_j^K P_{collision,j} \\ &\leq \sum_j^K \min_{i=1,2,3,4} P\left(\mathbf{a}_i^T \mathbf{z}_{j,t} < \mathbf{b}_i\right) \\ &\approx \sum_j^K \min_{i=1,2,3,4} \Delta_{ijt} \end{aligned} \tag{23}$$

Given the predefined safety requirements of the ego-vehicle p_{safe}, the overall probability of collision must satisfy:

$$P_{collision}(t) = \Delta_t < \Delta = 1 - p_{safe} \tag{24}$$

As the value of $\mathtt{erf}$ function in Eq.(22) can be pre-computed and stored as look-up tables, the proposed approach allows for very efficient computation of collision probability and trajectory-wise chance constraint checking. In this manner, an explicit correlation between perception uncertainties and risk of planning is established, enabling the subsequent motion planner to explicitly incorporate the quantified uncertainties in DNNs.

B. Probabilistic Feasible Tree Expansion

We build our Perception Uncertainty-aware RRT (PU-RRT) motion planner based on the baseline Closed-loop Rapidly-exploring Random Trees (CL-RRT) [31] framework. Similar to

the baseline CL-RRT, the tree expansion operation of PU-RRT incrementally grows a tree of probabilistic-feasible trajectories by random sampling (Algorithm 1). The algorithm takes as input the initial state of the ego-vehicle $\mathbf{x}_0$, a given goal state $\mathbf{x}_{\text{goal}}$, the predefined safety threshold p_{safe} and the current searching tree $\mathcal{T}_i$. The output of the tree expansion algorithm is a tree of probabilistic-feasible nodes $\mathcal{T}_{i+1}$.

During operation, the algorithm firstly updates the environment model using bounding boxes of detected obstacles and their spatial uncertainty $\mathcal{Q}_t = \{\mathbf{Q}_{j,t}\}$ (where $\mathbf{Q} = [\sigma_x, \sigma_y, \sigma_w, \sigma_l, \sigma_\theta]^T$) transferred by the proposed Probabilistic PointRCNN detector, and incorrect detection is filtered out by evaluating the classification uncertainty (MI and SE, Algorithm 2, line 4). A tree of feasible nodes is continuously expanded by recursively sampling from the planning space and connecting the current tree to the newly sampled nodes. To guarantee the kinodynamic feasibility of the generated trajectory, a closed-loop model containing the kinematic model and controllers is utilized to propagate its state toward the sample. The PU-RRT offers the flexibility of incorporating any off-the-shelf kinematic or dynamic vehicle models and lateral-longitudinal controllers. In our proposed planner, we adopt a simple bicycle model of the ego vehicle:

$$\dot{x} = v\cos\phi, \dot{y} = v\sin\phi, \dot{\phi} = \frac{v}{l}\tan\delta, \dot{v} = a \tag{25}$$

where x, y denote the planar position of the vehicle, ϕ refers to the heading (yaw), and l is the wheelbase. The control inputs are the steering angle δ and acceleration a. For lateral control, the proposed PU-RRT utilizes the pure-pursuit controller to generate the steering control input:

$$\delta_{cmd} = \arctan\frac{2\sin\alpha}{l_d} \tag{26}$$

where α is the heading angle of a selected look-ahead target point on the reference path toward x_{sample}, with respect to the vehicle body heading, and l_d denotes the look-ahead distance from the target point (see [32] for a more detailed definition). The longitudinal control of the proposed PU-RRT is implemented based on a proportional-derivative (PD) controller:

$$a_{cmd} = K_p(v_{ref} - v) + K_d\frac{d(v_{ref} - v)}{dt} \tag{27}$$

where K_p and K_d denote the proportional and derivative control gains, respectively. v_{ref} is the reference velocity.

The quantified DL perception uncertainties are utilized in the nearest-node selection (line 3) and feasibility checking operations (line 6, 10).

To bias the tree expansion to low-risk regions, we make extensions to the nearest node selection step by using a risk-based heuristic: for a newly sampled node N_{sample}, the algorithm selects M nearest nodes in $\mathcal{T}$ to be connected to, by evaluating a cost metric with two factors: the steering time from root node N_{root} to the specified node, and the operation risk $\delta_{i,t}$ estimated using the quantified uncertainties:

$$C(N_i) = k_{cc}\Delta_{i,t} + k_{\text{dist}}\frac{C_{\text{dist}}(N_i, N_{\text{root}})}{v} \tag{28}$$

where v is the sampled velocity, and k_{cc}, k_{dis} are weights for adjusting the strength of the two metrics, and the risk $\Delta_{i,t}$ is evaluated by estimating the probability of collision using the quantified uncertainties, as specified in Section V-A. In this manner, the proposed planner achieves a trade-off between bounding the operation risk and finding paths with low operation costs.

After the new feasible nodes are generated, the algorithm finally updates the upper and lower bound cost-to-go value of each node along the corresponding trajectory (line 23), depending on whether a probabilistic feasible trajectory directly to $\mathbf{x}_{goal}$ can be found. Such cost values are kept with the nodes for the selection of the best path, which will be specified in the following section. This procedure expands a tree of probabilistic- and dynamic-feasible trajectories within a predefined period of time and it operates in a receding horizontal manner.

Algorithm 1 PU-RRT, `Tree Expansion`

Input: $\mathbf{x}_0$, $\mathbf{x}_{\text{goal}}$, $\hat{\mathcal{B}}_t$, $\mathcal{Q}_t$, C, p_{safe}, $\mathcal{T}$

1: Take a sample $\mathbf{x}_{sample}$ from the configuration space C
2: Sort the nodes in the tree using heuristics as in Eq.(28)
3: Choose M nodes with the lowest cost $\mathcal{N}_{near} = \{n_i\}_1^M$
4: **for** each node $n_i \in \mathcal{N}_{near}$ **do**
5: Obtain the state of the ego-vehicle $\mathbf{x}_t$
6: **Evaluate the probability of collision using the quantified perception uncertainty** (Section V-A): $\Delta_t \leftarrow \text{EvaluateRisk}(\mathbf{x}_t, \hat{\mathcal{B}}_t, \mathcal{Q}_t)$
7: $k = 0$
8: **while** $\mathbf{x}_{t+k|t}$ has not reached $\mathbf{x}_{sample}$ **do**
9: Propagate the state towards $\mathbf{x}_{sample}$ $\mathbf{x}_{t+k+1|t} \leftarrow \text{steer}(\mathbf{x}_{t+k|t}, \mathbf{x}_{sample})$
10: **Evaluate the probability of collision** (Section V-A): $\Delta_{t+k+1|t} \leftarrow \text{EvaluateRisk}(\mathbf{x}_{t+k+1|t}, \mathcal{Q}_t)$
11: **if** $\text{isFeasible}(\mathbf{x}_{t+k+1|t})$ *and* $\Delta_{t+k+1|t} < 1 - p_{safe}$ **then**
12: Generate new intermediate nodes $n_{t+k+1|t} \leftarrow \text{generateNode}(\mathbf{x}_{t+k+1|t})$
13: Add node $n_{t+k+1|t}$ to $\mathcal{T}$
14: $k \leftarrow k + 1$
15: **else**
16: Exit while
17: **end if**
18: **end while**
19: **for** each newly generated node n by step in line 12 **do**
20: Obtain the state of node n
21: Try connecting n to $\mathbf{x}_{goal}$ (line 8-18)
22: **if** connection to $\mathbf{x}_{goal}$ is probabilistic feasible **then**
23: Update upper-bound cost-to-go from goal via node n to $\mathbf{x}_0$
24: **end if**
25: **end for**
26: **end for**

Output: Tree $\mathcal{T}$

C. Best Trajectory Selection and Execution

The ego-vehicle it keeps obtaining new measurements of the environment during operation, and the state estimation and perception uncertainties update dynamically. To cope with dynamic environments and variation of perception uncertainty, while growing the tree, the proposed PU-RRT periodically selects the current best path for execution. The path execution operation of PU-RRT is presented in **Algorithm 2**. The algorithm selects the current best sequence of nodes from the tree for execution, by evaluating the cost of the path. To evaluate the cost from the current node to the goal, each generated node corresponds with both a lower and an upper bound cost-to-go (C_{LB}, C_{UB}, respectively), which is defined in a similar way as in [31].

C_{LB} is calculated as the direct Euclidean distance between a specified node and the goal. Once a new node is generated, the algorithm attempts to predict a probabilistic feasible trajectory that connects the node to the goal. If such a trajectory is available, the cost associated with this trajectory is as the C_{UB}. After that, the upper bound cost-to-go of all its preceding nodes is updated by propagating backward to the root to check for lower cost paths to the goal. C_{LB} and C_{UB} are defined by:

$$C_{UB,i} = \begin{cases} +\infty & \text{no path to the goal} \\ \min_c(c_{N_i,c} + C_{UB_c}) & \text{path to the goal exists} \\ C_{LB} & \text{node inside goal region} \end{cases}$$
$$C_{LB,i} = \text{EuclideanDistance}(N_i, N_{\text{goal}}) \tag{29}$$

To incorporate the perception uncertainty-related operation risk, we make extensions to the cost metrics when calculating e_c (cost from a node N_i to its child node indexed by N_c):

$$e_c(N_i) = k_{cc}\Delta_{it} + k_{dist}\frac{C_{dist}(N_i, N_c)}{v} \tag{30}$$

The total C_{UB} of each feasible path to the goal is calculated, and the sequence of nodes with the lowest cost-to-goal is identified as the best path (line 11), which is then sent to the controller for execution.

Algorithm 2 PU-RRT, `Path Execution`

Input: $\mathbf{x}_0$, $\mathbf{x}_{goal}$, $\hat{\mathcal{B}}_t$, $\mathcal{Q}_t$, p_{safe},
1: **while** the vehicle does not reach goal **do**
2: Initialize the state of ego-vehicle at $\mathbf{x}_0$
3: Initialize tree $\mathcal{T} \leftarrow \text{treeInitialize}(\mathbf{x}_0)$
4: Update configuration space C using $\hat{\mathbf{B}}_t$, **filter out false detection using classification uncertainty** $\mathcal{Q}_t$
5: **while** time limit Δ_t is not reached **do**
6: Expand the tree by (Algorithm 1)
7: **end while**
8: **if** no feasible path to goal **then**
9: Initiate a fallback trajectory to steer the vehicle to a safe state $\mathbf{fallBackTraj}(x_t, \hat{\mathcal{B}}_t)$
10: **else**
11: Evaluate each path by cost metric (Eq. (29)), choose the best sequence of nodes $\{N_0...N_i\}$
12: **end if**
13: Apply best path $\{N_0...N_i\}$ and send corresponding control sequence to controller
14: **end while**
Output: Current best path $\{N_0...N_i\}_t$

VI. Evaluation

A. Training and Implementation of Bayesian-PointRCNN

We trained the proposed Bayesian PointRCNN on a platform with 2 NVIDIA Tesla V100 GPUs. The training dataset consists of 6000 frames of LiDAR point clouds, including selected frames from the KITTI dataset [33] and LiDAR measurements simulated by the CARLA simulator. The former consists of LiDAR point clouds recorded from real-world scenarios, and the latter contains simulated LiDAR measurement in virtual scenarios (Fig. 5 (c)).

For training, the dropout rate is fixed as 0.5 for intermediate layers, and the number of dropout-enabled forward passes is set as 25. We trained the RPN using 40 epochs, and the RCNN is trained firstly with regular loss function using 40 epochs, followed by another 120 epochs using attenuated loss function (Eq. 12). We utilized adam-onecycle optimization with a weight decay of 0.001 and a momentum of 0.9. The initial learning rate is 0.002 and decays by 0.5 every 40 epochs.

For inference, we use a dropout rate of 6 and run the experiments on a platform with a GTX 1070 GPU.

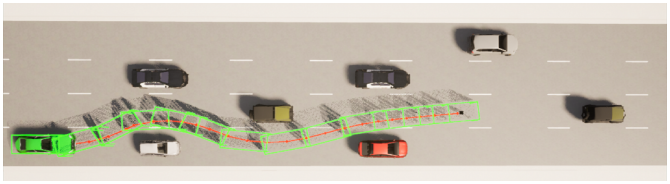
B. Implementation on the Integrated CARLA-ROS Simulator

We implemented the proposed Bayesian PointRCNN detector and PU-RRT motion planner as functional nodes of the ROS (Robotics Operating System) architecture.

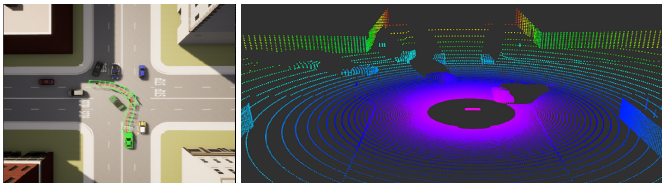
We utilized the CARLA simulator [34] to create simulated scenarios with static entities and traffic participants. The simulated 3D LiDAR point clouds are generated and fed to the proposed Bayesian PointRCNN-based perception node for object detection. The ego-vehicle is steered to follow the reference trajectory generated by the PU-RRT node. The behavior and motion of the vehicle are simulated by the dynamics model in CARLA. Data exchange between ROS nodes and CARLA is achieved via the CARLA-ROS Bridge.

C. Experiment Results

1) Object Detection and Uncertainty Modeling of DNN: We validate the proposed Bayesian PointRCNN detector on detecting objects of the 'car' category using the KITII dataset, the nuScenes dataset [35] and our self-built simulated CARLA dataset. To evaluate the impact of environmental factors on the perception uncertainties of DNN, we tested the object detector in scenarios with settings of varying observation distance and occlusion. Fig. 8 (a) depicts the evolution of aleatoric and epistemic uncertainties in terms of lateral and longitudinal scale, position, and SE of classification w.r.t the distance of objects. We observed a consistent increase of spatial regression uncertainty with a larger distance, and the uncertainty ellipse enlarges as the detected target vehicle moves away from the



(a) Simulated Scenario #1



(b) Simulated Scenario #2 (c) Simulated LiDAR point clouds

Fig. 5. Simulated scenarios for evaluating the proposed perception-planning framework: (a) scenario 1: multi-lane changing in dense traffic: (b) scenario 1: left-turn in unsignalized intersection (c) simulated LiDAR point clouds of scenarios 2. Trajectories planned by the proposed PU-RRT are colored in red, and the goal states are represented by black dots.

LiDAR (Fig. 6), due to the fact that distant objects typically result in sparse point cloud observations.

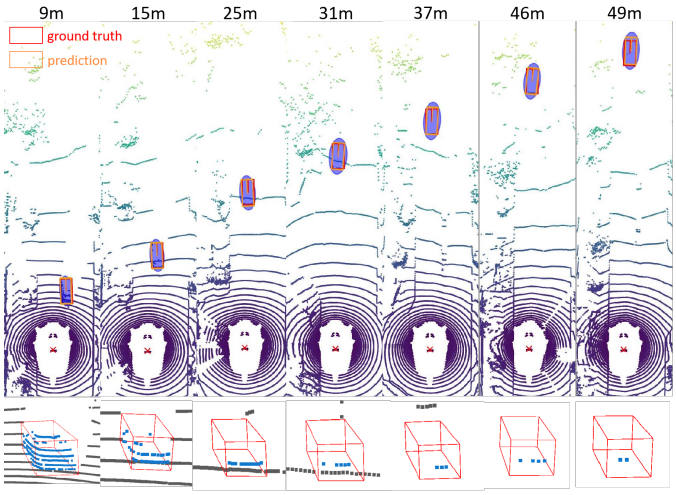


Fig. 6. Object detection and uncertainty quantification results w.r.t. to distance. The 2-σ uncertainty ellipse is colored in blue. (better view with magnification).

Fig. 7 shows an exemplary testing scenario from the nuScences dataset: the target vehicle (red) is gradually occluded by an in-between vehicle (white). As shown in Fig. 7 (b), the uncertainties in the detection of the target vehicle grow (especially in its longitudinal direction) as the degree of occlusion increases. The variation of aleatoric uncertainties w.r.t the portion of occlusion is depicted in Fig. 8 (b). It also shows an increase of uncertainties with a higher degree of occlusion, indicating that LiDAR provides limited observation of occluded objects.

To investigate the relationship of epistemic uncertainty and detection accuracy, we utilize the Intersection over Union (IoU, higher IoU indicates more accurate detection) as the metric of accuracy and plot the SE (for EU) w.r.t IoU in Figure 9. It shows that inaccurate detection typically yields high epistemic uncertainty, indicating that the DNN is unconfident about such detection.

In conclusion, these results indicate that the proposed uncertainty-aware detector can effectively capture the impact of various physical factors and model reasonable uncertainty inherent in observations and the DNN model.

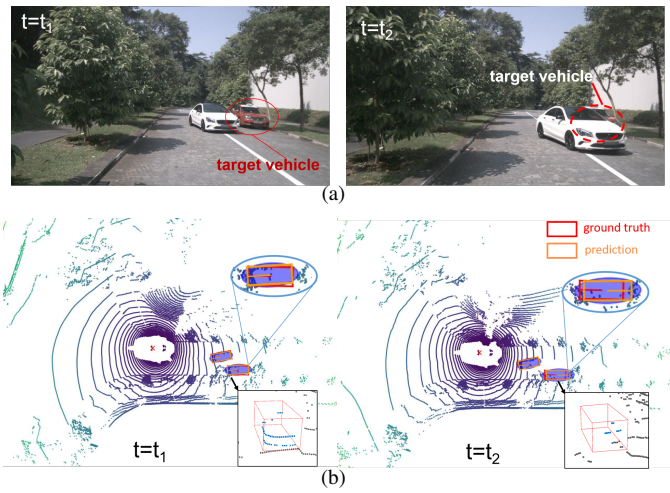


Fig. 7. Object detection and uncertainty quantification results in the presence of occlusion. The 2-σ uncertainty ellipse is colored in blue. (better view with magnification)

2) Motion Planning: We evaluate the proposed PU-RRT motion planner in two typical complex scenarios simulated in CARLA:

Scenario #1: multi-lane changing in dense traffic. (Figure. 5 (a)) This scenario simulates a multi-lane freeway with dense traffic, where the ego-vehicle shall perceive the surrounding vehicles and plan a safe and cost-efficient trajectory to traverse the traffic and reach a goal position across multiple lanes.

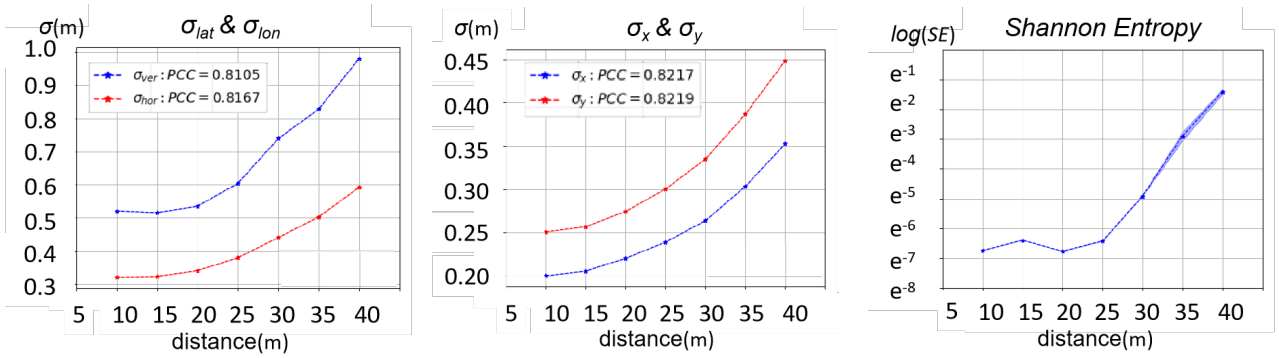
Scenario #2: left-turn in unsignalized intersection. (Figure. 5 (b)) In this scenario, the ego-vehicle must make a left turn in an unsignalized intersection with oncoming traffic and vehicles already in the intersection.

Factors that induce perception uncertainties in these scenarios include variations of observation distance and azimuth, occlusion, and sparsity of point clouds. For comparison purposes, we tested the following three motion planners in the above-mentioned scenarios:

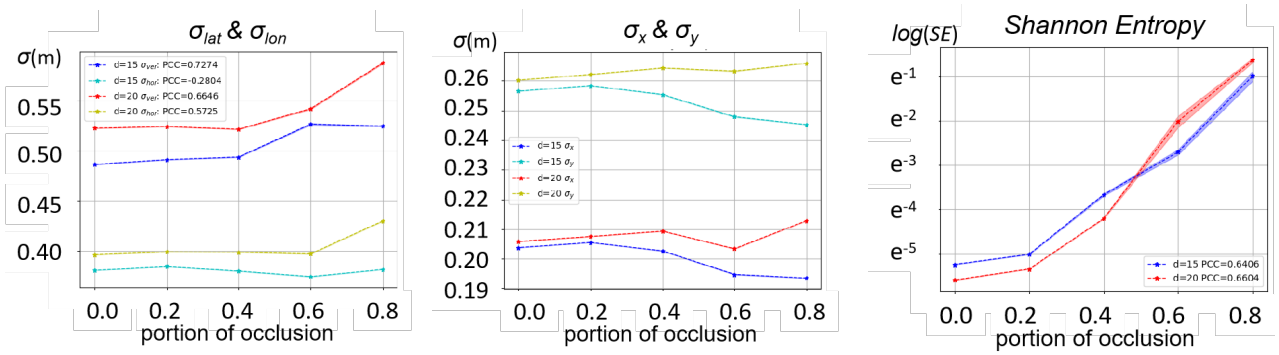
- *PU-RRT*: The proposed uncertainty-aware motion planner.
- *CC-RRT*: Nominal Chance-constrained RRT [23] which assigns a constant scale of spatial uncertainty of detected obstacles rather than using an explicit estimate of the perception uncertainties in DNNs.
- *CL-RRT*: Basic Closed-loop RRT [31] which utilizes deterministic bounding boxes without modeling uncertainty.

The evaluation criteria for motion planners are as follows:

- *Goal achievement*: We utilize three criteria to evaluate the motion planner's capability of finding safe trajectories to the goal:
  - ρ_1 (*Collision-free path*): Ratio of trials where collision-free paths are found, regardless if the Goal is reached or not.



(a) Evolution of regression and classification uncertainty w.r.t. distance of detection



(b) Evolution of regression and classification uncertainty w.r.t. degree of occlusion

Fig. 8. Evolution of regression and classification uncertainty with respect to the distance of detection and degree of occlusion. (PCC: Pearson Correlation Coefficient)

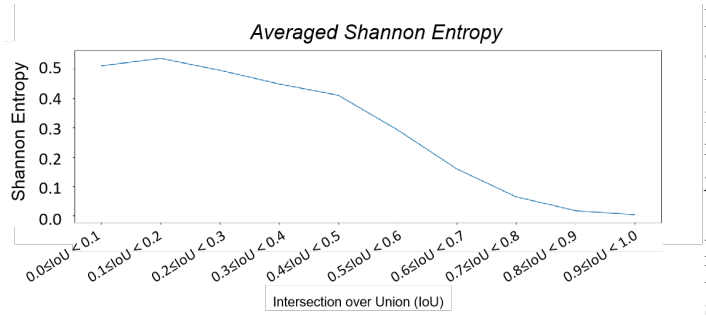


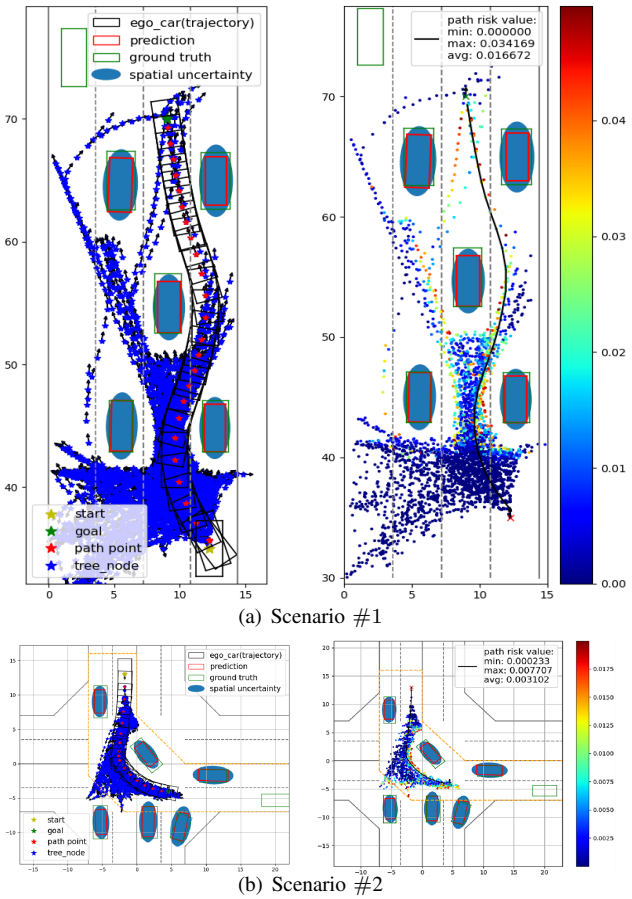
Fig. 9. Averaged Shannon Entropy (epistemic uncertainty) w.r.t. different Intersection over Union (IoU) intervals.

  - ρ_2 (Collision-free path to Goal): Ratio of trials where collision-free paths to the Goal are found.
  - ρ_3 (Risk-bounded path to Goal): Ratio of trials where paths to the goal are found, and the risk of collision at every waypoint satisfies the chance constraint.
- *Risk of collision*: The risk of colliding with any of the surrounding obstacles along the generated trajectories. The averaged, maximum and minimum risk along a trajectory are collected as statistics.
- *Cost of trajectories*: The following criteria are used to evaluate the operation cost of a planned path:
  - *Number of waypoints*: Total number of waypoints along a path to the goal.
  - *Length of a trajectory*: Overall Dubin's distance of a path to the goal.

Figure 10 shows an example of trees generated by the PU-RRT planner. As aforementioned, The probability of collision at each node is evaluated based on the quantified spatial uncertainty (Section V. A) during motion planning, and the magnitude of their corresponding risk is marked using color range. For all simulation trials, we used a safety threshold of $p_{safe} = 0.95$.

Snapshots of planned trajectories generated by the motion planners in Scenario #1 and Scenario #2 are depicted in Figure 11 and 12, respectively. As shown in the figures, since the nominal CL-RRT relies on deterministic bounding boxes for checking the binary collision-avoidance constraint (i.e. collision-free or non-collision-free), it generates paths that result in occasional near-misses or even collisions (rightmost sub-figures in Figure 11 (a)-(d)) in the presence of uncertain detection. The CC-RRT typically outputs longer paths that detour around obstacles, due to the over-conservative estimation of uncertainty. Whereas the proposed PU-RRT can incorporate the spatial perception uncertainty to generate risk-bounded paths without sacrificing optimality.

Table I presents the averaged statistics of performance evaluation of the three planners over 100 trials in the two scenarios. As shown in the table, although the nominal CL-RRT can achieve comparatively lower cost, it performs poorly on guaranteeing operation safety, as it outputs paths with the highest risk, and achieves the lowest ratio of probabilistic feasible paths(success rate #3). In contrast, the CC-RRT demonstrates the highest level of conservatism as it lowers the operation risk at the expense of longer paths. The experiment results indicate that by explicitly quantifying and propagating the uncertainty of DNN, the proposed PU-RRT can effectively



(a) Scenario #1

(b) Scenario #2

Fig. 10. Example of trajectory trees generated by the proposed PU-RRT and the estimated risk at each node. Magnitude of risk is marked by color range. The probability of collision is evaluated using the quantified uncertainties in object detection.

bound the operation risk, without inducing an adverse impact on the optimality of paths.

VII. Conclusion

This paper has proposed a coherent autonomous driving framework featuring the quantification and transfer of uncertainties in DNN-based perception. The proposed probabilistic 3D object detector can provide robust vehicle detection along with an explicit estimation of the inherent aleatoric and epistemic uncertainties in DNN. Taking advantage of the transferred quantified DNN perception uncertainties, the proposed chance-constrained motion planner can effectively bound the risk of collision and improve the operation safety. Experiment results show that the proposed framework outperforms existing perception-planning approaches without explicit uncertainty modeling.

Potential future work will focus on the following topics: for more accurate risk evaluation, a more proper spatial representation of quantified uncertainty will be developed. As there is no available ground truth of uncertainty in existing open datasets, metrics and methods for evaluating the quality of predicted uncertainty should be investigated. In addition, it is also necessary to investigate how perception uncertainty evolves over time and affects the uncertainty in the motion prediction of traffic participants.

References

[1] 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.

[2] S. Teng, X. Hu, P. Deng, B. Li, Y. Li, Y. Ai, D. Yang, L. Li, Z. Xuanyuan, F. Zhu, and L. Chen, "Motion planning for autonomous driving: The state of the art and future perspectives," *IEEE Transactions on Intelligent Vehicles*, pp. 1–21, 2023.

[3] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[4] K. Wang, T. Zhou, X. Li, and F. Ren, "Performance and challenges of 3d object detection methods in complex scenes for autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1699–1716, 2023.

[5] "Preliminary report highway: Hwy18mh010," National Transportation Safety Board (NTSB), Tech. Rep., 05 2018.

[6] G. P. Meyer and N. Thakurdesai, "Learning an uncertainty-aware object detector for autonomous driving," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10 521–10 527.

[7] Y. Gal, "Uncertainty in deep learning," *University of Cambridge*, vol. 1, no. 3, p. 4, 2016.

[8] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, Eds., vol. 24. Curran Associates, Inc., 2011, pp. 2348–2356.

[9] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1613–1622.

[10] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016, pp. 1050–1059.

[11] ——, "Bayesian convolutional neural networks with Bernoulli approximate variational inference," in *4th International Conference on Learning Representations (ICLR) workshop track*, 2016.

[12] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6405–6416.

[13] X. Tang, K. Yang, H. Wang, J. Wu, Y. Qin, W. Yu, and D. Cao, "Prediction-uncertainty-aware decision-making for autonomous vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 4, pp. 849–862, 2022.

[14] M. T. Le, F. Diehl, T. Brunner, and A. Knol, "Uncertainty estimation for deep neural object detectors in safety-critical applications," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 3873–3878.

[15] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Advances in neural information processing systems*, 2017, pp. 5574–5584.

[16] S. Choi, K. Lee, S. Lim, and S. Oh, "Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6915–6922.

[17] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, vol. 76, pp. 243–297, 2021.

[18] D. Feng, L. Rosenbaum, and K. Dietmayer, "Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 3266–3273.

[19] D. Feng, L. Rosenbaum, F. Timm, and K. Dietmayer, "Leveraging heteroscedastic aleatoric uncertainties for robust real-time lidar 3d object detection," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 1280–1287.

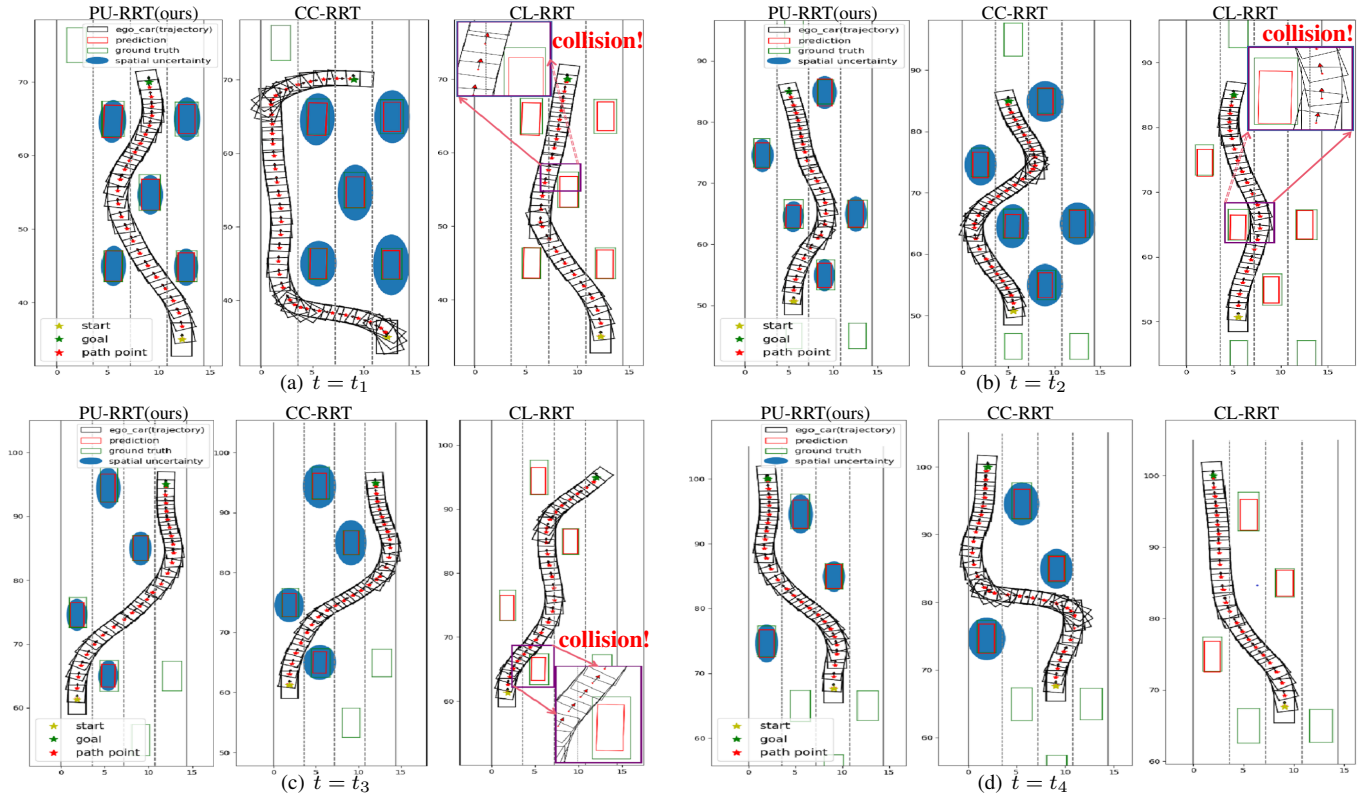


Fig. 11. Examples of planned trajectories generated by the PU-RRT (ours), CC-RRT, and CL-RRT planner in Scenario #1, snapshot are captured at 4 consecutive time steps. Locations of collisions are marked with purple rectangles and zoomed in for better illustration (better view with magnification)

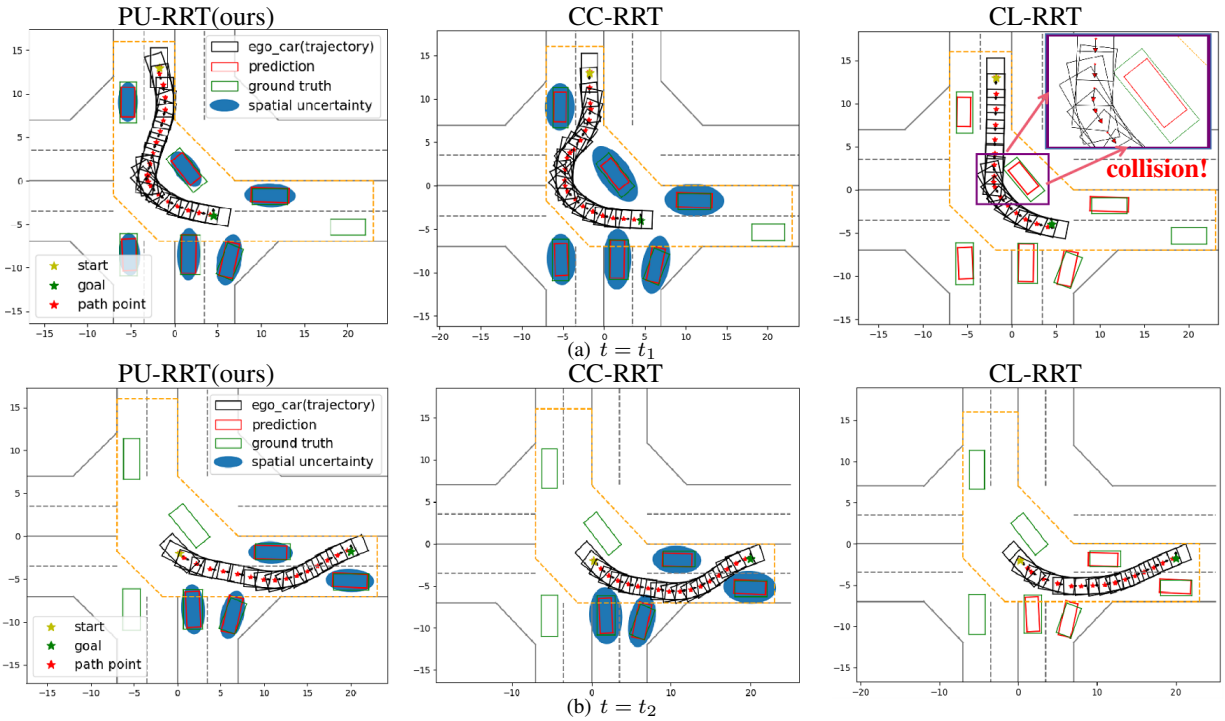


Fig. 12. Examples of planned trajectories generated by the PU-RRT (ours), CC-RRT, and CL-RRT planner in Scenario #2, snapshot are captured at 2 consecutive time steps (better view with magnification)

TABLE I
PERFORMANCE COMPARISON OF MOTION PLANNERS

	Motion planner	Goal achievement ↑			Operation risk ↓		Operation cost ↓	
		ρ_1	ρ_2	ρ_3	Ave.	Max.	N_{node}	Length(m)
Scenario #1: Multiple lane changing in dense traffic*								
$t = t_1$	PU-RRT(**ours**)	94%	53.19%	53.19%	0.00802	0.0236	29.89	39.33
	CC-RRT	100%	24.15%	24.15%	0.00124	0.00667	39.77	47.18
	CL-RRT	79%	83.33%	53%	0.01490	0.0385	26.92	36.92
$t = t_2$	PU-RRT(**ours**)	97%	100%	100%	0.00293	0.00890	23.94	35.35
	CC-RRT	100%	95.23%	95.23%	0.00190	0.00716	27.87	36.06
	CL-RRT	80%	77.51%	68%	0.00376	0.0135	24.97	35.30
$t = t_3$	PU-RRT(**ours**)	93%	99.01%	99.01%	0.00182	0.00884	26.29	36.75
	CC-RRT	99%	100%	100%	0.000836	0.00525	29.43	38.17
	CL-RRT	86%	91.74%	64%	0.00231	0.0103	27.69	37.23
$t = t_4$	PU-RRT(**ours**)	99%	98.04%	98.04%	0.00208	0.00983	24.12	33.62
	CC-RRT	100%	100%	100%	0.000916	0.00490	24.86	33.99
	CL-RRT	90%	86.96%	80%	0.00415	0.0164	25.47	33.34
Scenario #2: Left turn in unsignalized intersection*								
$t = t_1$	PU-RRT(**ours**)	97%	99.01%	99.01%	0.00204	0.00794	18.11	20.59
	CC-RRT	97%	94.34%	94.34%	0.00123	0.00745	19.43	21.14
	CL-RRT	93%	90.91%	0.78%	0.00219	0.0108	19.90	21.57
$t = t_2$	PU-RRT(**ours**)	98%	99.01%	99.01%	0.00628	0.0152	17.37	20.61
	CC-RRT	97%	88.50%	88.50%	0.00470	0.0114	18.34	20.99
	CL-RRT	94%	96.15%	82%	0.00876	0.0228	17.32	20.70

*ρ_1: ratio of collision-free paths among all trials; ρ_2: ratio of collision-free paths to the Goal among all trials; ρ_3: ratio of risk-bounded path to Goal among all trials; Ave. and Max. operation riks indicate the average and maximum probability of colliding with surrounding obstacles along the planned trajectories; N_{node}: averaged number of waypoints along the planned paths; Length: averaged length (in Dubin's distance) of planned paths to the goal.

[20] C. Hubmann, J. Schulz, M. Becker, D. Althoff, and C. Stiller, "Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 1, pp. 5–17, 2018.

[21] C. Hubmann, N. Quetschlich, J. Schulz, J. Bernhard, D. Althoff, and C. Stiller, "A pomdp maneuver planner for occlusions in urban scenarios," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 2172–2179.

[22] S. Brechtel, T. Gindele, and R. Dillmann, "Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, pp. 392–399.

[23] B. Luders, M. Kothari, and J. How, "Chance constrained rrt for probabilistic robustness to environmental uncertainty," in *AIAA guidance, navigation, and control conference*, 2010, p. 8160.

[24] W. Liu and M. H. Ang, "Incremental sampling-based algorithm for risk-aware planning under motion uncertainty," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2051–2058.

[25] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 723–730.

[26] P. Tigas, A. Filos, R. McAllister, N. Rhinehart, S. Levine, and Y. Gal, "Robust imitative planning: Planning from demonstrations under uncertainty," in *Proc. NeurIPS Workshop Mach. Learn. Auton. Driving*, 2019, pp. 1–9.

[27] J. Dahl, G. R. d. Campos, and J. Fredriksson, "Prediction-uncertainty-aware threat detection for adas: A case study on lane-keeping assistance," *IEEE Transactions on Intelligent Vehicles*, pp. 1–12, 2023.

[28] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, "A review and comparative study on probabilistic object detection in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[29] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 770–779.

[30] L. Blackmore, H. Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *2006 American Control Conference*. IEEE, 2006, pp. 7–pp.

[31] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on control systems technology*, vol. 17, no. 5, pp. 1105–1118, 2009.

[32] M. Samuel, M. Hussein, and M. B. Mohamad, "A review of some pure-pursuit based path tracking techniques for control of autonomous vehicle," *International Journal of Computer Applications*, vol. 135, no. 1, pp. 35–38, 2016.

[33] 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.

[34] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

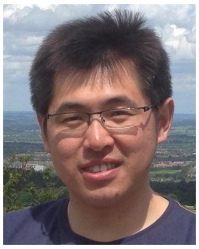
[35] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

Dachuan Li received his Ph.D. in control science and engineering from Tsinghua University (2015), China. He is currently an assistant professor (research track) at the Research Institute for Trustworthy Autonomous Systems, Southern University of Science and Technology, China. He worked as a postdoctoral researcher at California PATH and Institute of Transportation Studies, University of California, Berkeley from 2016 to 2018. His research interests include autonomous driving vehicles, trustworthy autonomous systems, intelligent decision-making and motion planning. He won the best paper awards of KSEM 2021 and IEEE ISPA 2021 conferences.

Bowen Liu received his B.Sc. Degree from Zhejiang University, China in 2020. He is currently a postgraduate student at the Department of Computer Science and Engineering of Southern University of Science and Technology(SUSTech). His research interests include motion planning and decision-making in autonomous driving.

Zijian Huang received his B.E. Degree from Southern University of Science and Technology (SUSTech), China in 2022. He is currently a post-graduate student at the Department of Computer Science and Engineering of SUSTech. His research interests include multi-modal fusion for the perception of autonomous driving and uncertainty quantification in deep learning.

Qi Hao (Member, IEEE) received the B.E. and M.E. degrees in electrical and computer engineering from Shanghai Jiao Tong University, Shanghai, China, in 1994 and 1997, respectively, and the Ph.D. degree in electrical and computer engineering from Duke University, Durham, NC, USA, in 2006. He was a Post-Doctoral Trainee at the Center for Visualization and Virtual Environment, University of Kentucky, Lexington, KY, USA. He was an Assistant Professor with the Department of Electrical and Computer Engineering, The University of Alabama, Tuscaloosa, AL, USA. He is currently an Associate Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology (SUSTech), Shenzhen, China. His research interests include intelligent sensing, machine learning, and autonomous systems.



Dezong Zhao (Senior member, IEEE) (M'12-SM'17) received the B.Eng. and M.S. degrees from Shandong University in 2003 and 2006, respectively, and the Ph.D. degree from Tsinghua University in 2010, all in Control Engineering. He was a Lecturer in Intelligent Systems with Loughborough University. Since 2020 he is a Senior Lecturer in Autonomous Systems with the University of Glasgow. His research interests include connected and autonomous vehicles, robotics, machine learning, and control engineering. He has been an EPSRC Innovation Fellow since 2018 and a Royal Society-Newton Advanced Fellow since 2020.

Bin Tian received the B.S. degree from Shandong University, Jinan, China, in 2009, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2014. He is currently an Associate Professor at the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include automated driving, vision sensing and perception, and machine learning.