Not All Uncertainty Matters: Simulation-in-the-Loop Fast–Slow Reasoning for Decision-Critical Autonomous Driving System

Jiayi Chen, Shuai Wang†, Guangxu Zhu†, Derrick Wing Kwan Ng, *Fellow, IEEE*, Chengzhong Xu, *Fellow, IEEE*, and Kaibin Huang, *Fellow, IEEE*

***Abstract*—Large vision-language models (VLMs) offer powerful open-world perception and reasoning capabilities for autonomous driving. However, their high computational overhead and non-negligible inference latency make continuous cloud-side invocation impractical for real-time operations. This limitation motivates a fast–slow collaborative architecture, where efficient onboard modules perform real-time perception and control tasks while cloud-based models provide selectively high-level reasoning when necessary. The central challenge in such systems is accurately determining when slow cloud reasoning should be involved and allowed to influence time-critical driving decisions. Existing approaches typically rely on perception uncertainty, heuristic triggers, or resource-driven policies, which fail to explicitly evaluate whether resolving a particular uncertainty will actually improve the downstream planning outcome. To address this issue, we propose SIGMA, a simulation-in-the-loop planning-gain modeling and assessment framework for effective task-oriented fast–slow collaboration. The key insight is that not all perception uncertainty equally affect decisions: perception ambiguity when it substantially alters the feasible trajectory set or significantly affects the associated planning cost. Rather than measuring uncertainty solely at the perception level, SIGMA embeds the planner directly into the uncertainty assessment loop to systematically evaluate how plausible scene realizations affect motion-planning outcomes. Specifically, SIGMA constructs candidate scene realizations under semantic and geometric uncertainty, evaluates them through a simulation-in-the-loop planning framework, and estimates the expected reduction in planning cost if the uncertainty is resolved. Building on this formulation, we introduce expected planning gain (EPG), a decision-level metric that quantitatively captures the expected trajectory-level improvement obtainable from cloud reasoning. EPG serves as a unified criterion for managing cloud invocation, cloud-guidance integration, and prioritizing cloud requests under deadline and resource constraints. Extensive experiments in the CARLA simulator demonstrate that SIGMA significantly reduces unnecessary cloud interactions while significantly improving planning performance, driving efficiency, and navigation success in both static and dynamic obstacle scenarios. Compared with fixed-period collaboration, SIGMA reduces unnecessary cloud interactions by 50%, improves navigation success rate by more than 6%, and achieves up to 26.2% reduction in finish time in dynamic scenarios. The implementation of our prediction-time 3D bounding-box-level uncertainty estimator is publicly available at https://github.com/cjychenjiayi/3D_Bbox_uncertainty.**

***Index Terms*—Large vision models, uncertainty-aware planning, edge–cloud collaboration, autonomous driving.**

I. Introduction

Recent advances in large vision, language, and vision–language models have enhanced open-world perception, semantic understanding, and reasoning for autonomous driving [1]–[3]. However, their high computational cost and inference latency make continuous invocation impractical for real-time driving systems, especially in resource-constrained edge–cloud settings. This limitation motivates a *fast–slow* architecture, where efficient onboard modules provide real-time perception, planning and control tasks, while cloud-based models offer slower but more powerful reasoning when necessary [4]–[6]. The central challenge is not merely how to leverage large models, but *when slow reasoning should be invoked and allowed to influence real-time driving decisions* under incomplete knowledge of the surrounding world.

Existing autonomous driving systems often determine cloud usage or decision refinement based on perception uncertainty, fixed offloading intervals, communication constraints, or heuristic triggers. Meanwhile, uncertainty-aware perception and planning methods have studied object detection uncertainty, unknown objects, conformal prediction, and robust decision-making under perception ambiguity [7]–[9]. Although these methods improve safety and robustness to a certain extent, they do not directly answer a more decision-centric question: *whether resolving the current uncertainty would change or improve the final driving decision*. This limitation reflects a fundamental mismatch between perception-space uncertainty and planning-space decision making. For instance, a perception module may exhibit high uncertainty about distant or irrelevant objects without affecting the planned trajectory, while seemingly confident predictions near critical regions may still result in unsafe or suboptimal driving decisions. Thus, *not all uncertainty matters*; what truly matters is whether alternative plausible scene realizations would induce different downstream planning outcomes.

We argue that the root limitation is not simply inaccurate uncertainty estimation, but the absence of an explicit decision-

Part of this work has been presented at the 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hangzhou, China, in the paper entitled "Opportunistic Collaborative Planning with Large Vision Model Guided Control and Joint Query-Service Optimization." Jiayi Chen and Guangxu Zhu are with the Shenzhen Research Institute of Big Data, Shenzhen, China. Shuai Wang is with the Center for Cloud Computing, Shenzhen Institutes of Advanced Technology (SIAT), Chinese Academy of Sciences, Shenzhen, China. Derrick Wing Kwan Ng is with the School of Electrical Engineering and Telecommunications, The University of New South Wales, Sydney, NSW 2052, Australia. Chengzhong Xu is with the State Key Laboratory of Internet of Things for Smart City (SKL-IOTSC), University of Macau, Macau, China. Kaibin Huang is with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong. † denotes the corresponding authors.

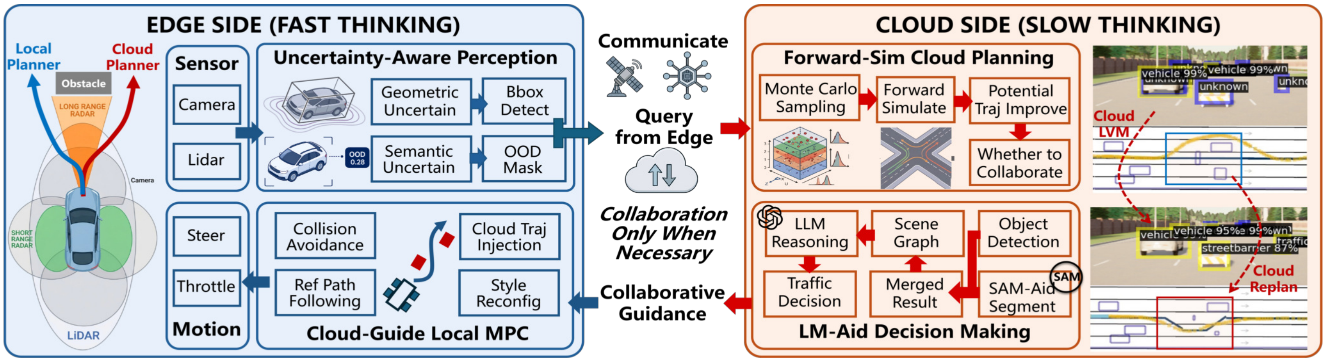


Fig. 1. Overview of the proposed planning-gain-guided fast–slow collaborative driving framework.

evaluation mechanism that systematically considers *multiple plausible realizations of the world*. In real-world driving, the environment is only partially observed, and a single perception output may admit multiple valid semantic and geometric interpretations. Therefore, uncertainty should not be treated only as an intrinsic property of perception, but as a *decision-induced quantity* defined by its effect on planning outcomes. This view is related to forward-simulation-based planning and the emerging "simulate before execute" paradigm [10], [11]. It is also aligned with the agentic-harness perspective for physical AI systems, where model outputs are mediated by runtime verification, safety, and execution-control modules before being deployed in the physical world. However, our goal is not to learn a comprehensive generative world model or perform simulation for its own sake. Instead, we leverage forward simulation as a lightweight decision evaluation tool: before allowing slow cloud reasoning to affect the onboard controller, the system estimates whether plausible scene variations would result in meaningful changes in planning outcomes. In this sense, the proposed simulation-in-the-loop EPG module can be viewed as a planning-grounded verification layer for fast–slow autonomous driving.

To this end, we introduce **SIGMA**, a *Simulation-In-the-loop planning-Gain Modeling and Assessment* framework for decision-aware fast–slow collaboration. Rather than relying on a single estimated world, we construct multiple plausible scene realizations by decomposing ambiguity into semantic "*what*" and geometric "*where*" components. These realizations are then propagated into a simulation-in-the-loop forward simulation, where their downstream effects are evaluated according to the resulting planning costs. By comparing decision outcomes across scene hypotheses, the system can determine whether the ambiguity is decision-relevant and quantifies the potential planning benefit of resolving it. In this formulation, uncertainty is evaluated relying sidely on its *decision impact*, rather than through perception-level proxy metrics.

This perspective naturally leads to a planning-gain-guided view of fast–slow collaboration. The onboard system retains real-time control authority and continuously generates safe and dynamically feasible trajectories, while the cloud provides slower but more reliable semantic reasoning and high-level guidance only when its expected benefit justifies the additional latency and cost. Instead of heuristically switching between onboard and cloud decisions, we introduce a decision-level metric, termed *expected planning gain* (EPG), which estimates the expected improvement in planning quality if additional information or reasoning is incorporated. EPG can be interpreted as a task-specific approximation of the value of information, and it governs both *when to invoke cloud intelligence* and *whether its output should be integrated into local planning*. Under limited computation and communication resources, cloud collaboration is further handled through a planning-gain-aware request prioritization rule that favors requests according to their expected impact on decision outcomes [6], [12].

Overall, the framework integrates uncertainty modeling, simulation-in-the-loop forward simulation, and *planning-gain-guided invocation and integration of cloud guidance into onboard planning*. As illustrated in Fig. 1, it shifts the objective of fast–slow collaboration from optimizing proxy metrics such as uncertainty, latency, or invocation frequency to optimizing *decision outcomes under scene variability*.

The main contributions are summarized as follows:

- **Decision-coupled uncertainty modeling.** We formulate perception uncertainty as planning-relevant ambiguity rather than standalone confidence, and decompose it into semantic uncertainty for unknown-object screening and geometric uncertainty for obstacle-level risk modeling. This yields an explicit semantic–geometric interface from open-world perception to MPC constraints, including a data-driven optimize for semantic screening and probabilistic obstacle inflation for geometric uncertainty.
- **Simulation-in-the-loop planning value estimation.** We introduce EPG as a decision-level metric that quantifies the value of resolving current ambiguity. EPG is derived as the expected normalized MPC-cost reduction over plausible scene realizations and estimated via budget-aware Monte Carlo rollouts. By propagating multiple plausible scene realizations through MPC, EPG identifies uncertainty that truly changes planning outcomes, rather than merely appearing uncertain in perception space.
- **Planning-gain-guided fast–slow collaboration.** We design a planning-gain-aware mechanism that adopts EPG to regulate cloud invocation, cloud-guided MPC reference switching, and deadline-aware prioritization. The cloud acts as selective guidance, while real-time feasibility and safety remain by onboard MPC.
- **End-to-end validation in open-world driving scenarios.** We implement SIGMA in CARLA and evaluate it

TABLE I
FEATURE-LEVEL COMPARISON WITH REPRESENTATIVE RELATED WORKS.

Work / Ref.	**Uncertainty Modeling**			**Capability / constraints**			**Planning-level decision awareness**		
	Semantic uncertain	Geometric uncertain	Plan impact	Real time	Open-world capacity	Resource aware	Planner integration	Collaboration criterion	Planning-gain modeling
Xiao *et al.* (TMC'24) [13]	–	–	–	High	Low	✓	–	Resource	–
Ye *et al.* (TMC'24) [14]	✓	×	×	High	Low	✓	–	Accuracy/resource	–
Fang *et al.* (TMC'24) [15]	–	–	–	High	Low	✓	–	Perception priority	–
Lin *et al.* (TMC'25) [16]	✓	×	×	Medium	Low	✓	–	Security	–
Liang *et al.* (TMC'26) [17]	–	–	–	High	Medium	✓	–	Blind spot	–
Yang *et al.* (TMC'26) [18]	–	–	–	High	Low	✓	–	Resource	–
Zhou *et al.* (TMC'26) [19]	–	–	–	Medium	Low	✓	–	Energy/carbon	–
Tang *et al.* (TMC'22) [20]	×	×	×	Medium	Low	✓	–	Edge-load uncertainty	–
Xia *et al.* (TMC'24) [21]	×	×	×	Medium	Low	✓	–	Task/network uncertainty	–
Cheng *et al.* (TMC'25) [22]	×	×	×	Medium	Low	✓	–	Network/QoS uncertainty	–
Xu *et al.* (TMC'26) [23]	×	×	×	Medium	Low	✓	–	Multifactor uncertainty	–
Li *et al.* (TIV'24) [7]	×	✓	✓	Medium	Low	–	MPC	Perception uncertainty	×
Li *et al.* (TMECH'24) [5]	–	–	–	High	Low	✓	MPC	Resource	–
Tian *et al.* (ICRA'24) [2]	–	–	–	Low	High	–	–	Semantic	–
SIGMA (ours)	✓	✓	✓	High	High	✓	MPC	Value	✓

Note: In the uncertainty columns, ✓means that the item is explicitly modeled, ×means that the work considers uncertainty but not this item, and –means that uncertainty is not a main modeling target or the item is not applicable. QoS denotes quality of service.

under static and dynamic unknown-obstacle scenarios. Compared with fixed-period collaboration, SIGMA reduces unnecessary cloud interactions by 50%, improves navigation success rate by more than 6%, and achieves up to 26.2% reduction in finish time in dynamic scenarios.

Symbol Notation: Throughout the paper, scalars are represented by italic letters, vectors or lists by bold lowercase letters, matrices by bold uppercase letters, and sets or scene representations by calligraphic uppercase letters.

II. Related Work

A. Uncertainty Estimation in Autonomous Driving Perception

Reliable uncertainty estimation is crucial for autonomous driving, as perception errors can propagate to downstream planning and compromise safety. Existing works primarily focus on uncertainty at the perception level, including robust detection under adverse conditions [24], probabilistic object detection with localization uncertainty [25], unknown-object discovery [26], [27], and uncertainty calibration with statistical guarantees [9], [28], [29]. These approaches improve the detection reliability and provide more accurate characterization of uncertainty in open-world perception.

However, such methods treat uncertainty mainly as an intrinsic property of perception outputs, focusing on robustness, calibration, or unknown-instance awareness. They do not explicitly address whether a given uncertainty is *decision-relevant*, i.e., whether resolving it would change the downstream planning outcome. In contrast, our work adopts a task-oriented perspective, where uncertainty is evaluated based on its impact on planning rather than its magnitude alone.

B. Motion Planning Under Perception Uncertainty

Another line of work incorporates uncertainty into motion planning and decision making. Classical approaches include formal verification and robust control [30], [31], as well as planning frameworks that explicitly account for uncertainty through chance-constrained optimization, MPC variants, or integrated perception–planning pipelines [7], [8], [11], [32]–[34]. Related studies have investigated distributed or collaborative planning under uncertainty [35], [36]. More broadly, some prior work on informative planning and forward-simulation-based exploration explicitly accounts for uncertainty when evaluating and selecting candidate actions [10], [37], [38].

These works primarily aim to improve safety and robustness under uncertainty, typically treating uncertainty as a factor to be handled conservatively once it appears. What remains underexplored is whether all uncertainty should be treated equally, and more importantly, whether resolving a particular uncertainty would *improve the optimal decision*. Our work addresses this gap by explicitly quantifying the planning gain of uncertainty resolution through simulation-in-the-loop evaluation, enabling the system to distinguish between decision-critical and decision-irrelevant uncertainty.

C. Fast–Slow Collaborative Driving

Recent autonomous driving systems increasingly adopt a fast–slow collaborative paradigm. On one hand, cloud- and edge-assisted frameworks enable vehicles to offload computation tasks and collaborate with external resources [4]–[6], [39]–[41]. On the other hand, large models have been introduced to substantially enhance semantic understanding and high-level decision making in autonomous driving, including language-guided planning and driving agents [1]–[3], [42]–[47]. These studies demonstrate the potential of coupling lightweight onboard modules for real-time perception, planning, and control with more capable cloud-side reasoning in complex scenarios. However, existing collaborative systems typically rely on heuristic triggers, predefined pipelines, or capability-driven designs to decide when cloud or large-model reasoning should be invoked. Likewise, large-model-based driving methods mainly focus on improving reasoning capability, but often assume that the generated guidance, once available, should directly affect downstream decision making. As a result, slow reasoning may be unnecessarily triggered even when it brings little planning benefit, and its outputs may be incorporated without explicitly assessing their impact on the final driving decision.

In contrast, our framework introduces a decision-centric perspective for fast–slow collaboration: both the invocation of cloud reasoning and its integration into local planning are governed by a unified planning-gain criterion. By evaluating whether uncertainty resolution leads to measurable improvements to planning performance, the system allocates both computational resources and decision authority.

D. Positioning of Our Work

Existing studies on perception uncertainty, uncertainty-aware planning, and collaborative driving are largely developed along separate directions: perception works estimate uncertainty, planning works handle uncertainty, and collaborative or large-model systems enhance reasoning capability. What remains underexplored is a unified decision-oriented perspective, where both uncertainty resolution and external reasoning are governed by their expected impact on downstream planning. Our work adopts this view by evaluating uncertainty and fast–slow collaboration through EPG, a planning-level criterion.

Table I further positions SIGMA against representative studies in perception uncertainty, uncertainty-aware planning, collaborative perception, and edge–cloud service scheduling. Existing collaborative and resource-aware systems provide strong mechanisms for computation offloading, communication scheduling, perception enhancement, or service-cost optimization, including under uncertain network or workload conditions. However, these systems generally optimize proxy objectives such as perception quality, resource usage, latency, or service cost, rather than the downstream planning gain obtained by resolving uncertainty. SIGMA differs by using simulation-in-the-loop planning cost to decide both when uncertainty deserves slow reasoning and when the resulting guidance should affect real-time control.

III. System Model and Problem Formulation

As illustrated in Fig. 1, we consider a fast–slow collaborative autonomous driving system consisting of an onboard edge side and a cloud side. The edge side acts as the fast-thinking component. It receives multi-modal observations from onboard sensors, such as cameras and LiDAR, and performs real-time perception, planning, and control. Its uncertainty-aware perception module constructs a local planning-oriented scene representation from the current observation, including detected objects, geometric uncertainty of obstacle states, and semantic uncertainty for unknown-object awareness. Based on this local representation, the onboard MPC planner performs reference-path following and collision avoidance, and outputs low-level control commands such as steering and throttle input.

The cloud side acts as the slow-thinking module. When collaboration is requested, it performs forward simulation over multiple plausible scene realizations to estimate whether resolving the current uncertainty would improve the downstream trajectory. It can also refine perception and generate high-level driving guidance using large vision or vision–language models, aided by object detection, segmentation, scene-graph construction, and traffic reasoning. The cloud output is not directly used as a low-level control command; instead, it is returned as refined scene information or high-level trajectory guidance and incorporated into the onboard MPC planner. Therefore, real-time control authority and safety enforcement remain on the edge side, while the cloud is selectively used only when slow reasoning is expected to improve the planning outcome before the decision deadline.

We consider the trajectory decision problem of an autonomous vehicle assisted by a large vision or vision–language model. At each time step t, the ego vehicle observes the surrounding environment and plans a future trajectory τ_t. The goal is to select a safe, efficient, and dynamically smooth trajectory under uncertain perception. In particular, let $\mathcal{C}_t$ denote a planning-oriented scene configuration, while $c_{q,t}$ denotes a scalar detection confidence score. Under ideal perception and reasoning, trajectory planning can be formulated as the minimization of a trajectory-level cost function:

$$\boldsymbol{\tau}_t^\star = \arg\min_{\boldsymbol{\tau}_t \in \mathcal{T}_t} \; J(\boldsymbol{\tau}_t), \tag{1}$$

where $\boldsymbol{\tau}_t$ is the trajectory list and $\mathcal{T}_t$ is the feasible trajectory set at time step t. The cost function $J(\tau_t)$ needs to account for trajectory length $L(\tau_t)$, travel or completion time $T(\tau_t)$, smoothness cost $S(\tau_t)$, and safety or collision-risk cost $R(\tau_t)$. One widely-adopted weighted-sum formulation is [48]:

$$J(\boldsymbol{\tau}_t) = w_l L(\boldsymbol{\tau}_t) + w_T T(\boldsymbol{\tau}_t) + w_s S(\boldsymbol{\tau}_t) + w_r R(\boldsymbol{\tau}_t), \tag{2}$$

where parameters (w_l, w_T, w_s, w_r) are non-negative weights that balance the relative importance of each objective.

In real-world driving scenarios, however, the onboard system has only partial and uncertain observations of the environment. Unknown or ambiguous objects may be incorrectly classified, conservatively treated as obstacles, or missed by the local perception module. Consequently, the onboard planner may optimize over an inaccurate scene representation leads to a conservative, unsafe, or detouring suboptimal trajectory.

This motivates a fast–slow architecture. The onboard module serves as the fast system, providing real-time perception, planning, and control. The cloud-side model serves as the slow system and provides more accurate semantic recognition and high-level reasoning for uncertain or unknown objects. Nevertheless, invoking the large model introduces communication delay and computation cost. Therefore, continuous invocation is inefficient and may even be detrimental when the response arrives too late to affect the current planning decision.

Let $\mathbf{o}_t$ denote the local observation at time step t. The onboard system maps $\mathbf{o}_t$ to a planning-oriented local scene representation $\mathcal{C}_t^{\text{loc}}$, while the large model, once invoked, returns a refined scene representation $\mathcal{C}_t^{\text{lm}}$. Accordingly, the perfect feasible trajectory set $\mathcal{T}$ becomes $\mathcal{T}(\mathcal{C}_t^{\text{loc}})$ and $\mathcal{T}(\mathcal{C}_t^{\text{lm}})$ under local and large-model-refined scene representations, respectively. As such, the local and large-model-assisted planning results are redefined as

$$\{\boldsymbol{\tau}_t^{\text{loc}}, \boldsymbol{\tau}_t^{\text{lm}}\} = \left\{ \underset{\boldsymbol{\tau}_t \in \mathcal{T}(\mathcal{C}_t^{\text{loc}})}{\arg\min} \; J(\boldsymbol{\tau}_t), \underset{\boldsymbol{\tau}_t \in \mathcal{T}(\mathcal{C}_t^{\text{lm}})}{\arg\min} \; J(\boldsymbol{\tau}_t) \right\}. \quad (3)$$

In this work, we assume that once the cloud result is returned before the decision deadline, the large model provides more accurate semantic recognition and reliable high-level decision guidance. Each invocation, however, incurs a response delay Δ_t^{lm} and an invocation cost κ^{lm}.

To describe whether the large model should be invoked, we introduce $\alpha_t \in \{0,1\}$, where $\alpha_t = 1$ indicates cloud invocation. Let $J_t^{\text{lm}} = J(\tau_t^{\text{lm}})$ and $J_t^{\text{loc}} = J(\tau_t^{\text{loc}})$. Since J_t^{lm} and the realized delay Δ_t^{lm} are unknown before cloud execution, we need an oracle formulation:

$$\begin{aligned} \min_{\{\alpha_t\}} \quad & \sum_{t=0}^{T} \left[(1-\alpha_t) J_t^{\text{loc}} + \alpha_t \left(J_t^{\text{lm}} + \lambda \kappa^{\text{lm}} + \rho D(\Delta_t^{\text{lm}}) \right) \right] \\ \text{s.t.} \quad & \alpha_t \in \{0,1\}, \quad t = 0, \ldots, T, \\ & \alpha_t \left(\Delta_t^{\text{lm}} - \Delta_t^{\text{ddl}} \right) \le 0, \quad t = 0, \ldots, T. \end{aligned} \quad (4)$$

Here, D is the delay cost function, and Δ_t^{ddl} is the decision deadline. The last constraint means that, if $\alpha_t = 1$, the cloud response must satisfy $\Delta_t^{\text{lm}} \le \Delta_t^{\text{ddl}}$; otherwise no cloud response is required.

This equation (4) is an oracle benchmark rather than an online policy, because J_t^{lm} and Δ_t^{lm} are unavailable before cloud execution. We therefore define the *Expected Planning Gain* (EPG) as the expected normalized reduction in MPC planning cost caused by resolving the current planning-relevant uncertainty. Let $\mathcal{C}_t^{\text{base}}$ be the conservative scene configuration from local perception, and let $\mathcal{C}_t(\boldsymbol{\xi}_t)$ be a plausible uncertainty-resolved configuration with $\boldsymbol{\xi}_t \sim p_t(\boldsymbol{\xi} \mid \mathbf{o}_t)$. Then

$$\text{EPG}_t = \mathbb{E}_{\boldsymbol{\xi}_t \sim p_t(\boldsymbol{\xi}|\mathbf{o}_t)} \left[\underbrace{\frac{J_t(\mathcal{C}_t^{\text{base}}) - J_t(\mathcal{C}_t(\boldsymbol{\xi}_t))}{J_t(\mathcal{C}_t^{\text{base}})}}_{:=\Delta J_t(\boldsymbol{\xi}_t)} \right]. \quad (5)$$

We also define a benefit factor $\tilde{g}_t = \text{EPG}_t - \lambda\kappa^{\text{lm}} - \rho D(\widehat{\Delta}_t^{\text{lm}})$, where $\widehat{\Delta}_t^{\text{lm}}$ is the predicted response delay. The cloud is invoked only when the estimated planning benefit outweighs the invocation and delay costs. Therefore, a necessary benefit-side condition for cloud invocation is $\tilde{g}_t > \theta_g$, where θ_g denotes the benefit-factor threshold. The complete online invocation rule further combines this benefit-side condition with deadline feasibility, as described in Section VI-B.

IV. Decision-Oriented Uncertainty

To support planning-gain-guided collaboration between onboard and cloud intelligence, uncertainty should be in a form that is directly relevant to downstream decisions. Rather than treating uncertainty as a generic property of perception

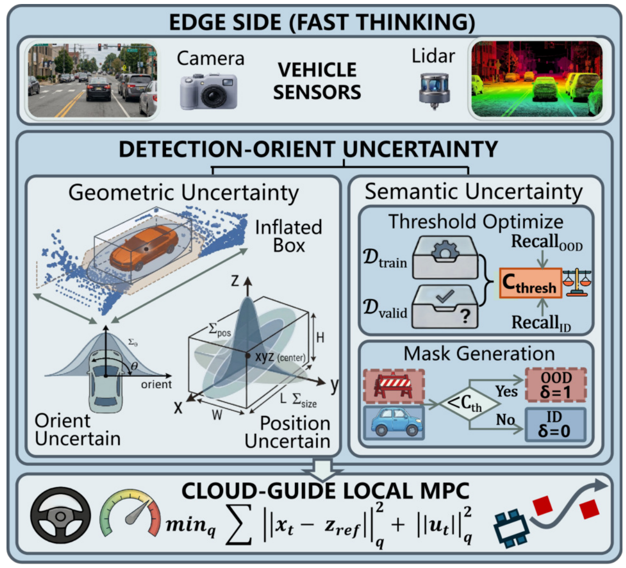


Fig. 2. Edge-side uncertainty-aware perception and onboard planning.

outputs, we view it as task-dependent and defined by its impact on planning. Some uncertainties directly affect trajectory optimization, while others mainly indicate possible perception failures with limited influence on the optimal motion.

We therefore propose a decision-oriented uncertainty representation that decomposes scene uncertainty into semantic and geometric components. Semantic uncertainty captures ambiguity in object identity and serves as a screening cue for acquiring additional information, such as invoking cloud assistance. Geometric uncertainty captures ambiguity in object localization and spatial extent, directly influencing collision risk, and trajectory optimization.

This decomposition provides a structured interface linking perception, planning, and cloud collaboration. Semantic uncertainty identifies candidate ambiguities that may require additional reasoning or external information, whereas geometric uncertainty governs how uncertainty propagates into planning and affects trajectory quality. Accordingly, semantic uncertainty is modeled as a confidence-based screening signal in 2D perception, while geometric uncertainty is represented as a probabilistic 3D object-detection output. The overall edge-side system architecture is illustrated in Fig. 2.

A. Semantic Uncertainty for Unknown Object Detection

In open-world driving scenarios, local perception models may encounter objects that are rare, unseen, or outside the training distribution. Such cases introduce ambiguity in semantic recognition, as the detector may be unable to confidently determine object identity. In our framework, this ambiguity is modeled as *semantic uncertainty*, which serves as a *decision-triggering signal* indicating that local perception may be unreliable and additional reasoning could be beneficial.

We adopt a simple yet effective confidence-based approximation to operationalize semantic uncertainty in practice. Given an input image at time step t, the local 2D object detector produces a set of bounding boxes with associated confidence scores $\{c_{q,t}\}$. Following confidence-based OOD detection principles, low-confidence predictions are treated as indicators of elevated semantic uncertainty. Specifically, we introduce a confidence threshold c_{th} to distinguish reliable in-distribution detections from potentially out-of-distribution objects. If the confidence score $c_{q,t}$ of a bounding box falls below c_{th}, the corresponding object is marked as semantically uncertain, resulting in a binary screening signal $\delta_{q,t}^{\mathrm{sem}} \in \{0,1\}$ for each detected object q.

To determine c_{th} in a principled and data-driven manner, following confidence-based OOD detection and selective prediction with rejection [49]–[51], we adopt the object detection confidence thresholding (ODCT) algorithm. The basic idea is to use the detector confidence as a post-hoc reliability score: high-confidence detections are treated as reliable in-distribution predictions, while low-confidence detections are screened as semantically uncertain and potentially unknown objects. The dataset is partitioned into a training set $\mathcal{D}_{\mathrm{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N_{\mathrm{train}}}$ and a validation set $\mathcal{D}_{\mathrm{val}} = \{(\mathbf{x}_j, y_j)\}_{j=1}^{N_{\mathrm{val}}}$, where $\mathcal{Y}_{\mathrm{train}}$ denotes the set of known semantic classes. The optimal threshold is obtained by solving

$$c_{\mathrm{th}}^{\star} = \underset{c_{\mathrm{th}} \in [0,1]}{\arg\max}\ G(c_{\mathrm{th}}), \tag{6}$$

where $G(c_{\mathrm{th}})$ is defined in (7).

The objective $G(c_{\mathrm{th}})$ balances the recall of out-of-distribution and in-distribution detections, enabling reliable identification of semantically uncertain objects while suppressing excessive false positives. Importantly, semantic uncertainty does not directly affect planning; instead, it acts as an upstream cue that flags potentially unreliable perception. This signal is subsequently used to evaluate whether resolving such uncertainty is likely to improve downstream planning decisions, thereby supporting opportunity-driven cloud collaboration.

B. Geometric Uncertainty for Bbox Estimation

In contrast to semantic uncertainty, geometric uncertainty directly affects motion planning by altering obstacle configurations and collision-risk assessment. Even small errors in object position or spatial extent may influence collision checking, feasible trajectory construction, and the final planning result. Therefore, geometric uncertainty should be explicitly modeled and propagated into the planning process, since it determines how uncertain obstacle geometry reshapes the feasible set of the planner and potentially alters the optimal trajectory.

To capture such uncertainty, we formulate 3D bounding box estimation as a probabilistic regression problem. For each detected object q, the bounding box is represented by its center position and spatial dimensions. Instead of producing a deterministic box, the detector predicts a Gaussian distribution over the box parameters:

$$\mathbf{z}_q \triangleq [x_q, y_q, z_q, l_q, w_q, h_q]^\top \sim \mathcal{N}\big(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q\big)\,. \tag{8}$$

where $\boldsymbol{\Sigma}_q \triangleq \mathrm{diag}\Big(\sigma_{q,x}^2, \sigma_{q,y}^2, \sigma_{q,z}^2, \sigma_{q,l}^2, \sigma_{q,w}^2, \sigma_{q,h}^2\Big)$. Here, (x_q, y_q, z_q) denotes the box center, while (l_q, w_q, h_q) denotes the box dimensions. The vector $\boldsymbol{\mu}_q$ is the predicted mean of the bounding box parameters, and $\boldsymbol{\Sigma}_q$ encodes their geometric uncertainty. For computational efficiency, we adopt an axis-aligned covariance structure, where each box parameter is assigned an independent variance.

The predicted geometric uncertainty is propagated into planning by inflating each obstacle according to planar position uncertainty:

$$\chi_q = \max\{|\Delta p_{q,x}|, |\Delta p_{q,y}|\}, \tag{9a}$$
$$r_q^{\mathrm{base}}(\eta) = F_{\chi_q}^{-1}(\eta), \tag{9b}$$
$$\mathcal{O}_q^{\mathrm{infl}}(\eta) = \mathcal{O}_q(\boldsymbol{\mu}_q) \oplus \mathbb{B}_\infty\big(r_q^{\mathrm{base}}(\eta)\big)\,, \tag{9c}$$
$$\Pr\big(\mathbf{p}_q \in \mathcal{O}_q^{\mathrm{infl}}(\eta)\big) \geq \eta. \tag{9d}$$

Here, $F_{\chi_q}^{-1}(\cdot)$ is the quantile function of the maximum planar perturbation χ_q, $\oplus$ denotes Minkowski addition, and $\mathbb{B}_\infty(r) = \{\delta : \|\delta\|_\infty \leq r\}$ is the planar L_∞ ball. Thus, larger uncertainty yields a larger safety envelope.

To effectively learn the uncertainty parameters, we augment the detector with a Gaussian uncertainty head that jointly predicts the bounding box mean $\boldsymbol{\mu}_q$ and the log-variance vector $\mathbf{s}_q = \log \boldsymbol{\sigma}_q^2$. The uncertainty head is trained using the Gaussian negative log-likelihood:

$$\mathcal{L}_{\mathrm{reg}}(\mathbf{z}_q, \boldsymbol{\mu}_q, \mathbf{s}_q) = \sum_{d=1}^{6} \frac{\exp(-s_{q,d})(z_{q,d} - \mu_{q,d})^2 + s_{q,d}}{2}.$$

The element-wise squared error $(z_{q,d} - \mu_{q,d})^2$ is weighted by the predicted inverse variance $\exp(-s_{q,d})$ for each bounding-box parameter d. The log-variance vector $\mathbf{s}_q$ allows the detector to assign uncertainty to ambiguous or noisy observations while penalizing unnecessarily large variance. For numerical stability, $\mathbf{s}_q$ is clamped within a bounded range during training. As such, the predicted uncertainty is further used to calibrate detection confidence:

$$c_q^{\mathrm{cls}} = \mathrm{sigmoid}(\ell_q^{\mathrm{raw}}) \cdot g(\mathbf{s}_q),$$

where ℓ_q^{raw} is the raw classification logit, $\mathrm{sigmoid}(\ell_q^{\mathrm{raw}})$ is the confidence, and $g(\mathbf{s}_q) = \mathrm{sigmoid}(f_g(\mathbf{s}_q)) \in (0,1)$ is a learned uncertainty-conditioned gating function, where $f_g(\cdot)$ is implemented by a lightweight MLP over the predicted log-variance features. When the geometric prediction is uncertain, this gating term reduces the detection confidence.

Overall, the learned Gaussian parameters $(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$ provide a probabilistic representation of obstacle geometry that can be directly integrated into planning modules. This allows geometric uncertainty to be explicitly propagated into trajectory optimization and supports the subsequent task-oriented value estimation framework, where the impact of uncertainty is quantified according to its influence on decision-making. An

$$G(c_{\mathrm{th}}) = \underbrace{\frac{\sum_{j=1}^{N_{\mathrm{val}}} \mathbb{I}(\hat{y}_j \notin \mathcal{Y}_{\mathrm{train}}) \cdot \mathbb{I}(c_j < c_{\mathrm{th}})}{\sum_{j=1}^{N_{\mathrm{val}}} \mathbb{I}(\hat{y}_j \notin \mathcal{Y}_{\mathrm{train}})}}_{\text{recall of OOD detections}} + \underbrace{\frac{\sum_{j=1}^{N_{\mathrm{val}}} \mathbb{I}(\hat{y}_j \in \mathcal{Y}_{\mathrm{train}}) \cdot \mathbb{I}(c_j > c_{\mathrm{th}})}{\sum_{j=1}^{N_{\mathrm{val}}} \mathbb{I}(\hat{y}_j \in \mathcal{Y}_{\mathrm{train}})}}_{\text{recall of in-distribution detections}}. \tag{7}$$

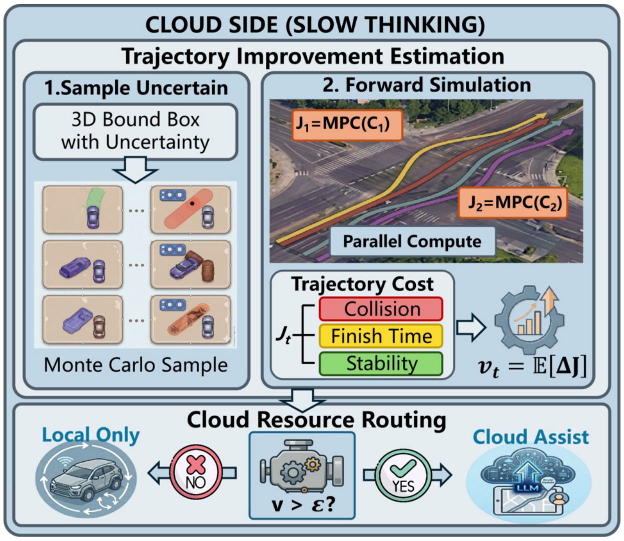


Fig. 3. Cloud-side slow-reasoning for planning guidance.

implementation of the 3D bounding-box uncertainty module is publicly available at https://github.com/cjychenjiayi/3D_Bbox_uncertainty.

V. Expected Planning Gain Under Perception Uncertainty

In Section IV, perception uncertainty is decomposed into semantic and geometric components. However, its practical importance depends on whether resolving it changes the downstream planning decision. We therefore evaluate uncertainty through EPG, which estimates the expected normalized MPC cost reduction over plausible scene-uncertainty realizations induced by the current observation. This provides a task-specific approximation of the value of additional information without maintaining a full belief-space planner.

Following this principle, we propagate perception uncertainty through the planning pipeline. Geometric uncertainty is first converted into a probabilistic obstacle representation (Section V-A), which is then incorporated into a dynamics-aware MPC optimization problem to define planning cost under each scene realization (Section V-B). Finally, a budgeted Monte Carlo procedure samples plausible scene realizations and evaluates their corresponding planning costs, producing a scalar EPG estimate that captures the expected impact of uncertainty resolution (Section V-C).

A. Safety-Constrained Probabilistic Obstacle Modeling

The goal of this section is to propagate geometric uncertainty into motion planning and quantify its effect on obstacle constraints, so that its contribution to downstream planning gain can be evaluated. While semantic uncertainty mainly serves as a screening signal for cloud assistance, motion planning is directly affected by geometric uncertainty, which determines obstacle configurations and collision risk. Therefore, we construct a probabilistic obstacle representation that explicitly links perception uncertainty to planning constraints.

Let $\mathcal{O}_q^0$ denote the nominal occupied region induced by the 3D bounding box of a detected object q. Based on the geometric uncertainty estimated in Section IV, the planning-relevant uncertainty of the obstacle is modeled through a randomly translated obstacle region:

$$\mathcal{O}_q^{\text{true}} = \mathcal{O}_q^0 \oplus \Delta\mathbf{p}_q, \text{ with } \Delta\mathbf{p}_q \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_q\right). \tag{10a}$$

Here, $\boldsymbol{\Sigma}_q = \operatorname{diag}\left(\sigma_{q,x}^2, \sigma_{q,y}^2, \sigma_{q,z}^2\right)$, $\Delta\mathbf{p}_q$ denotes uncertain object-center displacement, and $\oplus$ translates the nominal box by this displacement. Thus, $\mathcal{O}_q^{\text{true}}$ represents a random obstacle region induced by perception uncertainty.

In general, safe navigation under this uncertainty can be formulated as a chance-constrained collision avoidance requirement [52], [53]. Specifically, for an ego-vehicle trajectory τ, the probability of remaining collision-free with respect to object q should be no smaller than a prescribed safety confidence level η_{safe}:

$$\Pr\Big(\inf_t \operatorname{dist}\big(\tau(t), \mathcal{O}_q^{\text{true}}\big) > 0\Big) \geq \eta_{\text{safe}}. \tag{11}$$

This constraint captures the desired trajectory-level safety requirement. However, directly enforcing it in real time is computationally challenging, since it requires probabilistic collision checking over random obstacle geometry. As a tractable compromise, we convert this chance-constrained safety requirement into a deterministic inflated-obstacle constraint in the following.

To establish a tractable approximation, we convert the probabilistic safety requirement into a deterministic inflated-obstacle constraint. In practice, since collision checking is mainly determined by horizontal displacement, we define the inflation radius as the η_{safe}-quantile of the worst-case planar perturbation [53], [54]:

$$r_q^{\text{base}} \triangleq F_{\chi_q}^{-1}(\eta_{\text{safe}}) = \inf\left\{r \geq 0 : F_{\chi_q}(r) \geq \eta_{\text{safe}}\right\}. \tag{12}$$

Here, $\chi_q = \max\left\{|\Delta p_{q,x}|, |\Delta p_{q,y}|\right\}$ measures the maximum horizontal perturbation relevant to planner collision checking, and $F_{\chi_q}^{-1}(\cdot)$ denotes the quantile function of χ_q. Thus, r_q^{base} corresponds to the prescribed safety-level quantile of the planar uncertainty. [1]

Leveraging the inflation radius, the nominal obstacle is converted into a safety envelope. When the planar uncertainty is multi-modal, the same inflated-obstacle form is retained, while the radius is determined by the corresponding mixture-distribution quantile:

$$\mathcal{O}_q^{\text{infl}} \triangleq \mathcal{O}_q^0 \oplus \mathbb{B}_\infty\left(r_q^{\text{base}}\right). \tag{13}$$

The probabilistic obstacle representation establishes a direct bridge between perception uncertainty and motion planning constraints. By translating uncertain obstacle geometry into safety-constrained inflated obstacle sets, geometric uncertainty reshapes the feasible trajectory set, thereby providing the planning constraints for subsequent EPG estimation.

[1] In implementation, the distribution of χ_q can be obtained either analytically from the predicted Gaussian perturbation when an independent diagonal covariance is assumed, or numerically by Monte Carlo sampling. We use the resulting empirical cumulative distribution to evaluate the safety-level quantile.

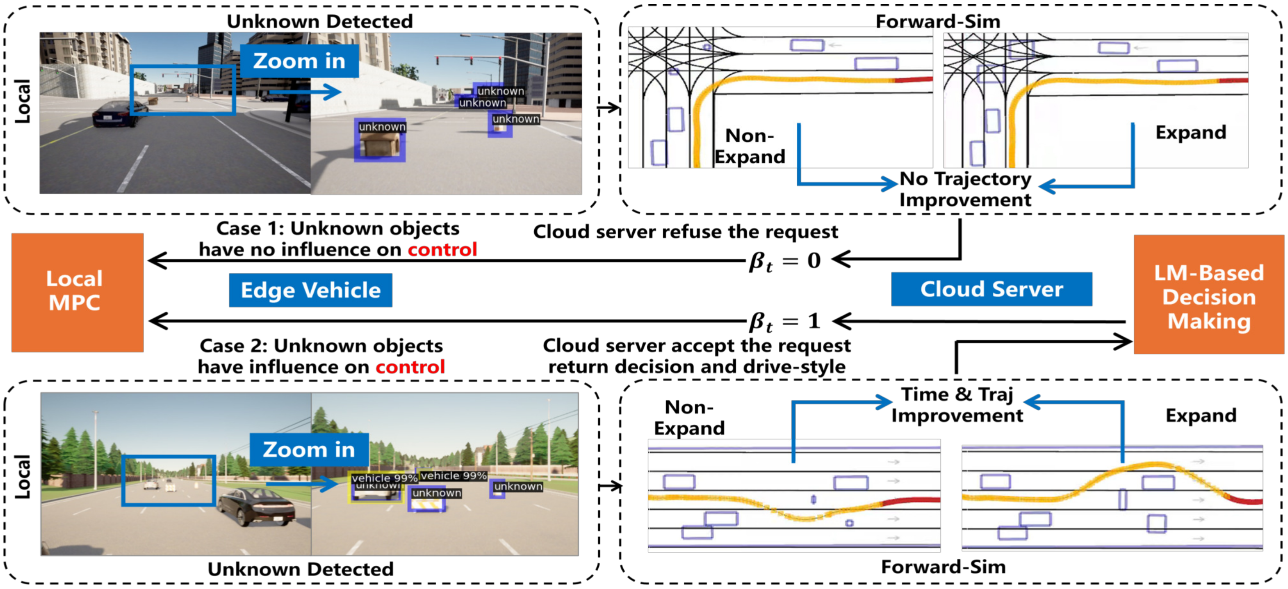


Fig. 4. Cloud-assisted forward simulation for decision evaluation.

B. Planning-Gain Evaluation via MPC

Given a deterministic scene configuration $\mathcal{C}_t$, the planner evaluates its decision-level effect by solving an MPC problem. Let $\mathbf{x}_{k|t}$ and $\mathbf{u}_{k|t}$ denote the predicted state and control at step k from time t, respectively. The time-indexed obstacle configuration is defined as

$$\mathcal{C}_t = \left\{ (\mathcal{O}^0_{q,k|t}, r_{q,k|t}) \;:\; q = 1, \dots, M,\; k = 0, \dots, N \right\},$$

which covers both static and dynamic obstacles. The MPC planner minimizes tracking error, control effort, and control smoothness subject to the dynamics, state/control limits, and time-indexed collision-avoidance constraints:

$$\begin{aligned} J_t(\mathcal{C}_t) = \min_{\{\mathbf{x}_{k|t}, \mathbf{u}_{k|t}\}} & \sum_{k=0}^{N} \left\| \mathbf{x}_{k|t} - \mathbf{x}^{\mathrm{ref}}_{k|t} \right\|^2_{\mathbf{Q}} + \sum_{k=0}^{N-1} \left\| \mathbf{u}_{k|t} \right\|^2_{\mathbf{R}} \\ & + \sum_{k=1}^{N-1} \left\| \mathbf{u}_{k|t} - \mathbf{u}_{k-1|t} \right\|^2_{\mathbf{S}} \\ \text{s.t.} \quad & \mathbf{x}_{0|t} = \mathbf{x}_t, \\ & \mathbf{x}_{k+1|t} = f(\mathbf{x}_{k|t}, \mathbf{u}_{k|t}), \quad k = 0, \dots, N-1, \\ & \mathbf{x}_{k|t} \in \mathcal{X}, \quad k = 0, \dots, N, \\ & \mathbf{u}_{k|t} \in \mathcal{U}, \quad k = 0, \dots, N-1, \\ & g_{\mathrm{coll}}\left(\mathbf{x}_{k|t}, \mathcal{O}^0_{q,k|t}, r_{q,k|t} \right) \le 0, \\ & q = 1, \dots, M,\; k = 0, \dots, N. \end{aligned} \tag{14}$$

The resulting optimal predicted trajectory is

$$\tau^\star_t(\mathcal{C}_t) = \{ \mathbf{x}^\star_{k|t}(\mathcal{C}_t) \}_{k=0}^{N}. \tag{15}$$

For two scene configurations $\mathcal{C}^1_t$ and $\mathcal{C}^2_t$, the normalized planning gain is defined as

$$\Delta_{\mathrm{pg}}(\mathcal{C}^1_t, \mathcal{C}^2_t) = \frac{J_t(\mathcal{C}^1_t) - J_t(\mathcal{C}^2_t)}{J_t(\mathcal{C}^1_t)}. \tag{16}$$

A negligible Δ_{pg} means that resolving the uncertainty does not materially change the planner, whereas a large value indicates decision-relevant uncertainty.

C. Budget-Aware Monte Carlo Estimation

This module estimates whether resolving uncertainty would improve downstream planning. Instead of relying on perception-level uncertainty alone, SIGMA propagates plausible uncertainty realizations through the MPC planner and evaluates their planning-cost impact, as illustrated in Fig. 4.

Let $\mathcal{C}^{\mathrm{base}}_t$ denote the conservative obstacle configuration constructed from local perception, and let $\boldsymbol{\xi}_t = \{\Delta \mathbf{p}_{q,t}\}_{q=1}^{M}$ denote a plausible realization of the current scene uncertainty. The corresponding uncertainty-resolved scene configuration is denoted by $\mathcal{C}_t(\boldsymbol{\xi}_t)$. We define

$$\delta J_t(\boldsymbol{\xi}_t) = \frac{J_t(\mathcal{C}^{\mathrm{base}}_t) - J_t(\mathcal{C}_t(\boldsymbol{\xi}_t))}{J_t(\mathcal{C}^{\mathrm{base}}_t)}, \tag{17a}$$

$$\mathrm{EPG}_t = \mathbb{E}_{\boldsymbol{\xi}_t \sim p_t(\boldsymbol{\xi}|\mathbf{o}_t)} \left[\delta J_t(\boldsymbol{\xi}_t) \right]. \tag{17b}$$

Here, $\delta J_t(\boldsymbol{\xi}_t)$ is the normalized MPC cost reduction under a plausible uncertainty-resolved scene realization, and the expectation is taken over the uncertainty distribution induced by the current observation.

Since this expectation is generally intractable, we estimate it using importance-sampling-based Monte Carlo rollout [55], [56]. Assuming conditional independence among object-level perturbations,

$$p_t(\boldsymbol{\xi} \mid \mathbf{o}_t) = \prod_{q=1}^{M} p_{q,t}(\Delta \mathbf{p}_{q,t} \mid \mathbf{o}_t). \tag{18}$$

Given samples $\boldsymbol{\xi}^{(i)}_t \sim \phi_t(\boldsymbol{\xi})$, the weighted estimator is

$$w_i = \frac{p_t(\boldsymbol{\xi}^{(i)}_t \mid \mathbf{o}_t)}{\phi_t(\boldsymbol{\xi}^{(i)}_t)}, \tag{19a}$$

$$\widehat{\mathrm{EPG}}_t = \frac{\sum_{i=1}^{K} w_i \delta J_t(\boldsymbol{\xi}^{(i)}_t)}{\sum_{i=1}^{K} w_i}. \tag{19b}$$

The estimate $\widehat{\mathrm{EPG}}_t$ is used as the planning-level criterion for selective cloud invocation: uncertainties with negligible estimated planning gain are filtered locally, while decision-relevant cases are preserved for cloud-assisted reasoning.

VI. Fast–Slow Collaborative Decision and Planning-Gain-Aware Cloud Resource Allocation

Autonomous driving naturally exhibits a fast–slow decision modeling structure. Most driving scenarios can be handled effectively by fast onboard processing, which provides real-time perception, planning, and control. However, a small subset of complex or ambiguous situations requires deeper reasoning and more reliable perception, which can be enabled by cloud-side large models. The key challenge is therefore to determine *when* cloud assistance is expected to provide meaningful decision-level benefit under uncertainty and limited system resources.

In our framework, the EPG defined in Section V serves as the decision-level criterion for fast–slow collaboration. In online operation, EPG is not directly observable and is replaced by its simulation-in-the-loop Monte Carlo estimate, denoted by $\widehat{\mathrm{EPG}}$. Specifically, $\widehat{\mathrm{EPG}}$ governs three aspects of the system: (i) cloud activation, namely whether the current scenario should be escalated to the cloud; (ii) decision refinement, namely how cloud intelligence improves the driving decision; and (iii) resource allocation, namely how limited cloud computation is distributed across multiple requests. In this way, cloud collaboration is driven by estimated planning impact rather than perception uncertainty or heuristic triggers.

Cloud assistance is activated only when $\widehat{\mathrm{EPG}}$ is sufficiently large, indicating that resolving the current uncertainty is estimated to improve the downstream planning outcome. Otherwise, the scenario is handled entirely onboard, avoiding unnecessary latency and computation. Once activated, the cloud refines perception and performs high-level semantic reasoning to generate guidance, which is then incorporated into the local planner. When multiple vehicles simultaneously request assistance, the same planning-gain principle is further used to prioritize cloud resources across competing requests.

A. LVM-Assisted Semantic Decision Generation

The cloud-side decision module is invoked only when the onboard system identifies decision-relevant uncertainty with a sufficiently large EPG. In such cases, the cloud module performs enhanced perception and semantic reasoning to provide high-level planning guidance that may not be reliably obtained from local processing alone.

We adopt a two-stage paradigm consisting of perception refinement and semantic decision generation. Given the input image $\mathbf{x}_t$, a high-accuracy cloud perception model first produces refined object detections, which are then organized into a structured semantic scene representation:

$$\mathcal{B}_t^{\diamond} = g_{\mathrm{cloud}}(\mathbf{x}_t), \;\; \mathcal{Z}_t = \Phi\left(\mathcal{B}_t^{\diamond}, \mathcal{L}_t, \mathcal{R}_t\right). \tag{20}$$

Here, $\mathcal{B}_t^{\diamond}$ denotes the refined cloud-side detection results, while $\mathcal{Z}_t$ represents the structured semantic scene representation. The function $\Phi(\cdot)$ integrates object categories, spatial relationships, lane-level structure $\mathcal{L}_t$, and drivable-region information $\mathcal{R}_t$ into a decision-oriented scene description.

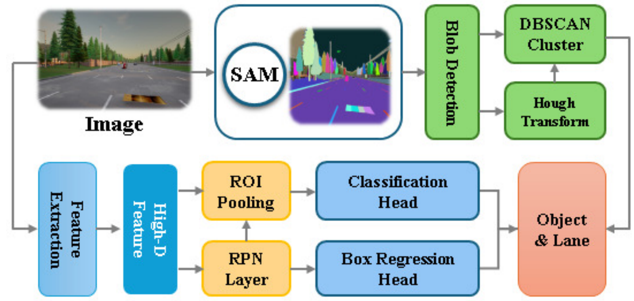


Fig. 5. Cloud-based LVM perception pipeline with SAM-assisted.

To improve robustness in complex scenarios, we incorporate segmentation-based verification, such as SAM, as illustrated in Fig. 5. The segmentation results are adopted to refine geometric structure, suppress unreliable detections, and extract drivable regions and lane geometry. The lane structure is obtained through geometric filtering and curve fitting:

$$\mathcal{S}_t = h_{\mathrm{seg}}(\mathbf{x}_t, \mathcal{B}_t^{\diamond}), \;\; \mathcal{L}_t = \{y = a_j x + b_j\}_{j=1}^{N_{\mathrm{lane}}}. \tag{21}$$

Here, $\mathcal{S}_t$ denotes the segmentation-based verification result, and $\mathcal{L}_t$ denotes the reconstructed lane structure. Each lane candidate is represented by a fitted line parameterized by (a_j, b_j), where N_{lane} is the number of detected lane structures. The resulting lane and drivable-region information complements the refined object detections and improves the reliability of the semantic scene representation.

Based on $\mathcal{Z}_t$, the large vision–language model performs semantic reasoning over obstacles, lane structures, and drivable regions to generate high-level traffic decisions, as shown in Fig. 6. Specifically, following the agentic fast–slow planning strategy in [12], the cloud-side model first produces a serialized traffic-decision sequence that describes the intended maneuver and its contextual rationale. This sequence is then parsed into maneuver-level guidance and converted into a reference trajectory for the local MPC planner:

$$\gamma_t = \Psi_{\mathrm{LVM}}(\mathcal{Z}_t) \in \{\mathrm{keep}, \mathrm{left}, \mathrm{right}\}, \tag{22a}$$

$$\mathcal{W}_t^{\diamond} = \mathcal{T}(\gamma_t, \mathcal{Z}_t). \tag{22b}$$

Here, $\Psi_{\mathrm{LVM}}(\cdot)$ denotes the semantic reasoning function of the large vision–language model, which follows the agentic fast–slow planning paradigm to produce the serialized traffic-decision sequence. The parsed maneuver γ_t specifies whether the ego vehicle should keep its lane, change to the left lane, or change to the right lane. The trajectory generation function $\mathcal{T}(\cdot)$ converts the maneuver decision and semantic scene representation into a reference trajectory $\mathcal{W}_t^{\diamond}$, which is provided to the local MPC module. In this way, the cloud module provides semantically informed and long-horizon planning guidance, while the onboard MPC module retains real-time control authority and enforces dynamic feasibility and safety.

B. EPG- and Deadline-Aware Cloud-Guided MPC

When multiple vehicles request cloud assistance, the cloud-side module should allocate its limited computation to requests whose uncertainty is both decision-relevant and actionable before the corresponding driving deadline. In this work, we adopt a lightweight EPG- and deadline-aware rule as an initial resource allocation mechanism, rather than solving a full-cloud scheduling optimization problem.

For each request i, let $\widehat{\mathrm{EPG}}_i$ denote the simulation-in-the-loop estimate of the expected planning gain obtained in Section V-C. This quantity estimates the normalized MPC planning-cost reduction that could be achieved by resolving the current uncertainty. Let t_i^{obs} denote the predicted arrival time at the first decision-critical obstacle. The remaining decision slack and the predicted cloud response time are defined as

$$s_i(t) = t_i^{\mathrm{obs}} - t, \quad \widehat{\Delta}_i^{\mathrm{inf}} = \widehat{\Delta}_i^{\mathrm{comm}} + \widehat{\Delta}_i^{\mathrm{cloud}}. \tag{23}$$

Here, $s_i(t)$ is the remaining time before the ego vehicle reaches the first obstacle that may require a planning decision, and $\widehat{\Delta}_i^{\mathrm{inf}}$ is the predicted cloud response time, including communication and cloud-side inference. Cloud assistance is useful only if the response is predicted to return within this slack. Following the benefit-factor formulation in Section III, we define the request-level benefit factor as

$$\tilde{g}_i(t) = \widehat{\mathrm{EPG}}_i - \lambda\kappa_i^{\mathrm{lm}} - \rho D(\widehat{\Delta}_i^{\mathrm{inf}}). \tag{24}$$

where κ_i^{lm} denotes the invocation cost of request i. Therefore, the cloud request decision is defined by a benefit-factor threshold and a deadline feasibility condition:

$$\alpha_i(t) = \mathbb{I}\left(\tilde{g}_i(t) > \theta_g\right)\mathbb{I}\left(s_i(t) > \widehat{\Delta}_i^{\mathrm{inf}}\right). \tag{25}$$

Here, $\alpha_i(t) = 1$ means that request i is eligible for cloud service, while $\alpha_i(t) = 0$ means that the request is handled locally. The first condition filters out requests whose benefit factor is insufficient, while the second condition ensures that the cloud result is predicted to arrive before the vehicle reaches the decision-critical obstacle.

When the cloud has limited computation capacity, only requests satisfying $\alpha_i(t) = 1$ are considered. These candidate requests are ranked by a benefit-factor-efficiency score:

$$\mathcal{Q}_{\mathrm{cloud}}(t) = \{i \in \mathcal{Q}(t) \mid \alpha_i(t) = 1\}, \tag{26a}$$

$$\psi_i(t) = \frac{\tilde{g}_i(t)}{\widehat{\Delta}_i^{\mathrm{inf}} + \epsilon}. \tag{26b}$$

Here, $\epsilon > 0$ is a small constant introduced to avoid division by zero and improve stability. The cloud serves requests in $\mathcal{Q}_{\mathrm{cloud}}(t)$ according to descending $\psi_i(t)$ until available computation budget is exhausted. This rule favors requests with large benefit factor and short predicted response latency.

Importantly, cloud request acceptance does not imply that cloud guidance is immediately available to the local planner. We therefore introduce another binary variable $\beta_i(t) \in \{0, 1\}$ to indicate whether valid cloud guidance has been returned before the decision deadline:

$$\beta_i(t) = \mathbb{I}\left(\alpha_i(t) = 1\right)\mathbb{I}\left(t_i^{\mathrm{ret}} \le t_i^{\mathrm{ddl}}\right)\mathbb{I}\left(\mathrm{Valid}(\mathcal{W}_i^{\diamond}) = 1\right), \tag{27}$$

where t_i^{ret} is the actual cloud-result return time, t_i^{ddl} is the decision deadline, and $\mathcal{W}_i^{\diamond}$ denotes the cloud-guided reference trajectory. Here, $\mathrm{Valid}(\mathcal{W}_i^{\diamond}) = 1$ means that the returned reference is consistent with the current ego state, satisfies drivable-region constraints, and yields a feasible MPC problem. If this check fails, we set $\beta_i(t) = 0$, and the planner falls back to the local reference in the reference-switching rule below. Thus, $\alpha_i(t)$ represents the cloud request decision, whereas $\beta_i(t)$ represents the availability of valid returned cloud guidance.

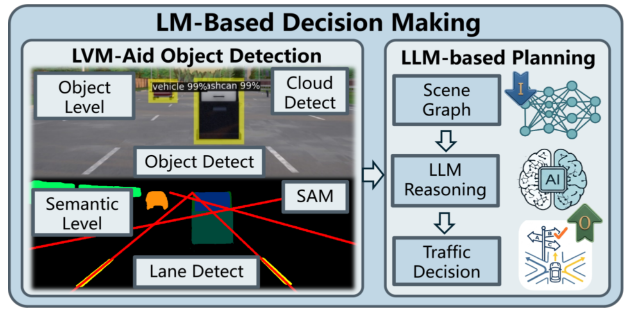


Fig. 6. Cloud-based large model decision-making pipeline.

Once valid cloud guidance is available, the local planner incorporates it by switching the reference path used in the MPC objective:

$$\mathcal{W}_i^{\mathrm{ref}} = \begin{cases} \mathcal{W}_i^{\mathrm{loc}}, & \beta_i(t) = 0, \\ \mathcal{W}_i^{\diamond}, & \beta_i(t) = 1. \end{cases} \tag{28}$$

Here, $\mathcal{W}_i^{\mathrm{loc}}$ is the local reference path, and $\mathcal{W}_i^{\diamond}$ is the cloud-guided reference path. The selected $\mathcal{W}_i^{\mathrm{ref}}$ is then used as the MPC reference input defined in Section V-B. In this way, the cloud does not directly control the vehicle; it only updates the high-level reference when valid guidance returns before the deadline, while the onboard MPC enforces dynamic feasibility and collision safety in real time.

VII. Experiments

We evaluate the proposed framework from three perspectives: 1) perception and uncertainty modeling, 2) simulation-in-the-loop planning-gain assessment in static scenarios, and 3) dynamic-scene decision making. All system-level experiments are implemented in Python and ROS on the CARLA platform [57]. The ego vehicle is equipped with an RGB-D camera and a 64-line LiDAR operating at 10 Hz. The sensor frequency specifies the perception update rate in simulation, and no additional sensing delay is modeled. The local detection model processes sensory inputs at up to 100 FPS and the local MPC module generates control outputs at 20 Hz. Also, the safety distance of MPC is set to $d_0 = 0.3$ m and $d_{\mathrm{inf}} = 3.0$ m. The prediction horizon is $H = 18$ with a time step of $\Delta t = 0.3$ s. All experiments are conducted on an Ubuntu workstation with two NVIDIA RTX 3090 GPUs.

For comparison, we consider the following schemes: 1) **Local-only strategy (LOS)**, which exploits local perception and a state-of-the-art MPC (i.e., RDA) [34]; 2) **Periodical collaboration strategy (PCS)**, a baseline collaboration strategy with fixed intervals between consecutive cloud services [40]; and 3) **SIGMA (ours)**, the proposed fast–slow collaboration framework based on decision-oriented uncertainty representation, ODCT-based semantic screening, and simulation-in-the-loop EPG estimation. Recent extensions of this work further explore joint query-service optimization [6] and agentic fast–slow planning [12]. We note that the collaborative planner

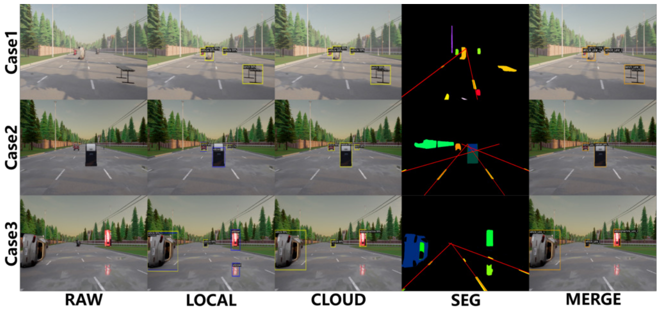
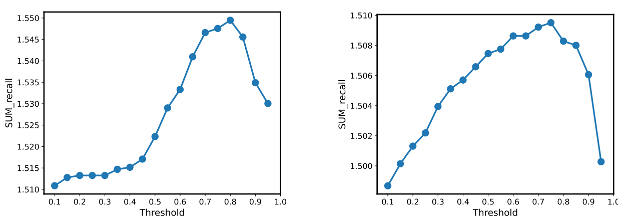


Fig. 7. Comparison of local and cloud perception results.

TABLE II
DATASET SPLITS FOR KNOWN- AND UNKNOWN-OBJECT EVALUATION.

Split	# Samples	Known	Unknown	Total
Local_Train	9,647	38	0	38
Local_Val	2,060	21	5	26
Local_Test	3,400	19	6	25
Cloud_Train	15,107	49	0	49



(a) Validation set. (b) Test set.

Fig. 8. ODCT threshold selection on the validation and test sets.

in [5] does not involve large vision models and exhibits similar behavior to LOS [34]. We first validate the perception and uncertainty modeling components of the proposed framework, including semantic uncertainty for unknown-object awareness and geometric uncertainty for 3D localization ambiguity. We then evaluate how these uncertainty estimates are exploited by the simulation-in-the-loop planning-gain assessment mechanism in static and dynamic driving scenarios.

A. Experiment 1: Perception and Uncertainty Modeling

We first validate the perception-level components of the proposed framework from two aspects. We evaluate *semantic uncertainty* for unknown-object awareness and cloud-assisted recognition, and then evaluate *geometric uncertainty* for 3D object localization. The former verifies whether local perception can reliably identify potentially unknown objects and screen candidate cases for cloud assistance, as described in Section IV-A, while the latter examines whether the predicted geometric uncertainty reflects true localization ambiguity and is therefore suitable for downstream planning.

1) Semantic Uncertainty for Unknown-Object Awareness: We conduct experiments to validate the LVM-based perception module and the ODCT-based semantic uncertainty screening mechanism. We collect a dataset in the CARLA Town06 map, generating over $30{,}000$ frames of 1080×720 RGB images synchronized with vehicle odometries. Each frame is annotated with ground-truth bounding boxes and undergoes quality filtering to ensure object visibility, with occluded or partially visible instances excluded during data curation. As shown in Table II, the dataset contains 49 object categories (including vehicles and 48 other object types in CARLA), and is partitioned into local and cloud subsets for open-vocabulary detection. Among these 49 categories, 38 are treated as known classes and the remaining 11 as unknown classes. The local training set contains $9{,}647$ samples covering all 38 known classes. The validation set contains $2{,}060$ samples with 21 known classes and 5 unknown classes, while the test set contains $3{,}400$ samples with 19 known classes and the remaining 6 unknown classes. The unknown classes in the validation and test subsets are non-overlapping and excluded from local-model training.

The local and cloud models are implemented based on the UnSniffer architecture [27], which provides a confidence score for each detection. The local detection model is trained on the training subset and fine-tuned on the validation set to optimize the threshold c_{th}. The evaluation is performed on the test set. For the cloud detector, a larger training set of $15{,}107$ samples spanning all 49 predefined categories is used.

Fig. 8 depicts the value of G (i.e., the sum of the recall of in-distribution detections and the recall of out-of-distribution detections in (7)) versus c_{th}. On the validation set, the recall score reaches its maximum at $c_{\mathrm{th}} = 0.8$, while on the test set the highest score is achieved at $c_{\mathrm{th}} = 0.75$. The small shift of the optimal threshold between the two splits suggests that confidence thresholding provides a stable criterion for semantic uncertainty screening, even though the validation and test sets contain non-overlapping unknown classes. Meanwhile, this result also reflects an inherent tradeoff in confidence-based unknown-object detection: a larger threshold tends to preserve more potentially unknown objects for cloud-side reasoning, but may also introduce more false alarms among known objects; a smaller threshold suppresses unnecessary warnings but risks missing ambiguous objects. The ODCT procedure therefore does not aim to make the local model perfectly recognize all unknown objects, but to choose a practical operating point for early screening.

With the optimized threshold, the local model achieves recalls of 0.83 for known objects and 0.71 for unknown objects on the validation set. On the test set, the known-object recall increases to 0.89, while the unknown-object recall decreases to 0.61. This decrease is expected because the test unknown classes are not seen during either local training or threshold selection, and may present different visual appearances from the validation unknown classes. Nevertheless, the local model still retains a considerable portion of unknown objects as low-confidence detections. This is sufficient for our purpose, since semantic uncertainty is used to identify candidate ambiguous cases rather than to provide the final semantic interpretation at the edge. The cloud model, trained on all predefined classes, achieves a recall of 0.89 on the known-class validation set, confirming its stronger recognition capability under richer training coverage and cloud-side computation.

We further visualize representative perception results in Fig. 7. Case 1 contains only known objects, while Cases 2–3 include unknown objects. In Case 1, the local model

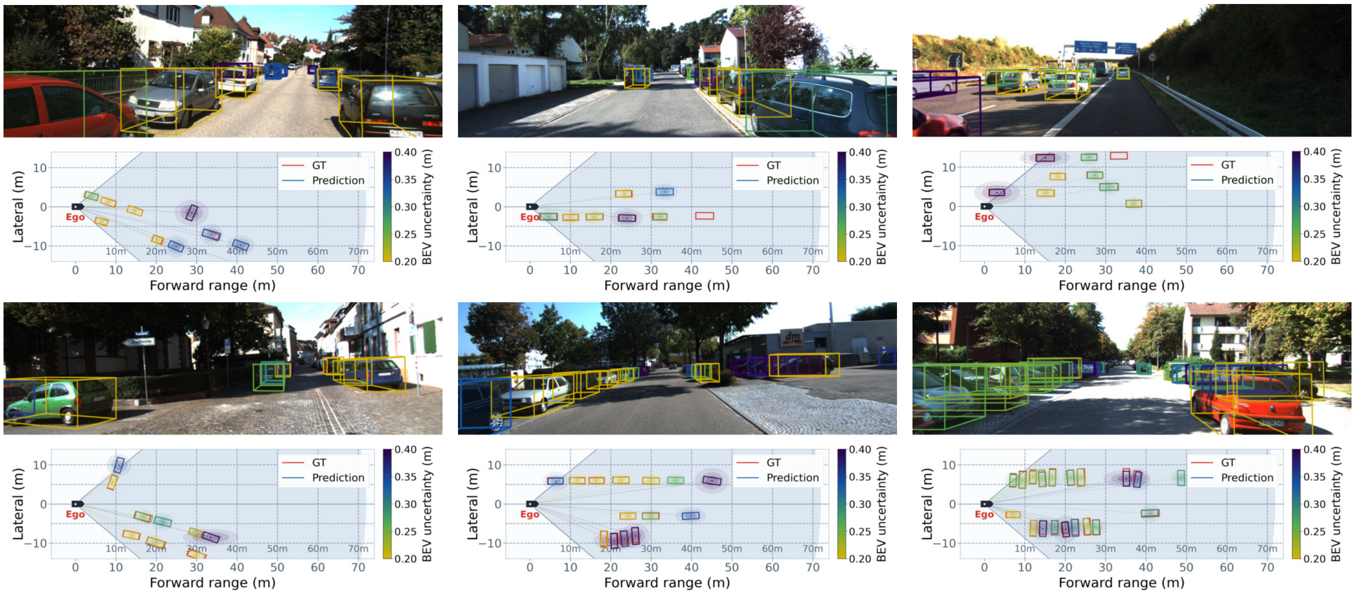
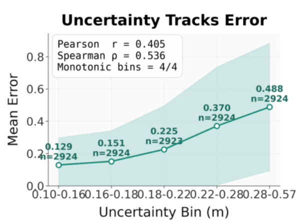
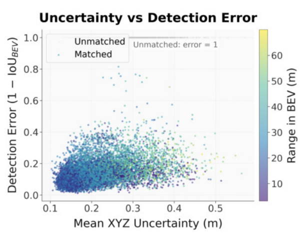


Fig. 9. Qualitative examples of 3D object detection with predicted geometric uncertainty.



(a) Bin statistics



(b) Scatter distribution

Fig. 10. Predicted geometric uncertainty versus localization error on KITTI.

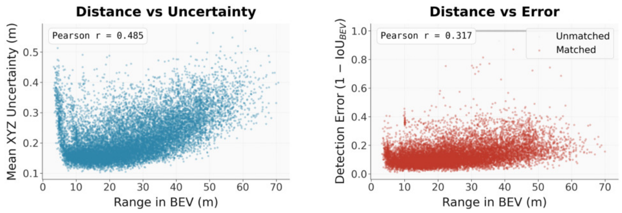
provides reliable detections, indicating that cloud reasoning would provide limited additional semantic benefit. In contrast, in Cases 2–3, the local model generates bounding boxes for both known and unknown objects, but fails to assign reliable semantic labels to unknown objects such as the trash can and vending machine. This behavior is useful for fast–slow collaboration: the local model does not need to fully understand every unknown object, but it should avoid silently ignoring potentially relevant objects. The low-confidence detections below $c_{\text{th}} = 0.8$ allow the onboard system to raise an "unknown-object" warning and mark these objects as candidate cases for cloud-assisted reasoning. The cloud model then recognizes the unknown objects and further segments the image into lane and object masks, allowing the system to recover object–lane topology for downstream decision making. These results validate semantic uncertainty as an effective screening signal: it separates normal local-perception cases from ambiguous cases that may deserve cloud reasoning, while leaving the final decision of whether cloud assistance is actually useful to the downstream planning-gain assessment.

2) Geometric Uncertainty for 3D Object Localization: We validate whether the proposed geometric uncertainty modeling reflects true localization ambiguity in 3D perception on the KITTI validation set, using the Gaussian uncertainty-aware 3D detector introduced in Section V-A. The detector predicts both 3D bounding box parameters and their associated uncertainty, which are later used for planning-oriented obstacle inflation. For analysis, we use the mean predicted standard deviation over the spatial dimensions (x, y, z) as a scalar uncertainty measure, and quantify localization error by $1 - \text{IoU}_{\text{BEV}}$.

Fig. 10a presents the main quantitative result, where predictions are grouped into uncertainty bins and the average localization error is computed for each bin. The results show a clear monotonic trend: as predicted uncertainty increases, the average localization error rises from 0.129 to 0.488. This indicates that the uncertainty head tends to assign larger uncertainty to objects that are indeed harder to localize. The positive Pearson correlation of 0.405 and Spearman correlation of 0.536 further support this trend. Meanwhile, the scatter distribution in Fig. 10b shows that the relationship is not perfectly deterministic at the instance level, which is expected because 3D localization error can also be affected by factors such as object distance, viewpoint, occlusion, object size, and LiDAR sparsity. Therefore, the predicted uncertainty should be interpreted as a statistical indicator of localization difficulty rather than an exact error predictor for every individual sample.

Representative KITTI examples in Fig. 9 provide qualitative evidence consistent with this interpretation. Both image-view and BEV-view projections show larger predicted uncertainty in visually challenging cases, such as distant objects, sparse observations, or ambiguous object extents. Fig. 11 further explains this phenomenon from a physical perspective. As object range increases, image details become less discriminative and LiDAR points become sparser, making accurate 3D localization more difficult. Consequently, both predicted uncertainty and actual localization error tend to increase with distance, which matches the intuitive behavior of 3D perception.

For completeness, Table III reports the standard KITTI detection accuracy of the uncertainty-aware detector. The results show that the detector maintains competitive 3D detection performance while producing meaningful uncertainty estimates. This is important because the uncertainty branch should provide additional planning-relevant information without substan-



(a) Distance vs. uncertainty (b) Distance vs. error

Fig. 11. Relationship with predicted uncertainty, and localization error.

TABLE III
KITTI DETECTION ACCURACY OF THE UNCERTAINTY-AWARE DETECTOR.

Metric	Easy	Moderate	Hard
3D AP@0.70	89.48	79.20	78.66
BEV AP@0.70	90.33	88.15	87.71
BBox AP@0.70	98.61	89.60	89.24
3D AP_R40@0.70	92.83	83.57	81.14
BEV AP_R40@0.70	96.09	89.65	87.24
BBox AP_R40@0.70	99.27	95.33	92.95

tially compromising the original detection capability. Overall, these results show that the learned geometric uncertainty is not arbitrary: it follows the expected difficulty pattern of 3D localization and captures meaningful spatial ambiguity. Therefore, it provides a reasonable uncertainty representation for obstacle inflation and subsequent simulation-in-the-loop planning-gain assessment.

B. Experiment 2: Scenarios with Static Obstacles

1) System-Level Performance Comparison: Next, we evaluate the LOS, PCS, and SIGMA schemes at a target speed of 6 m/s in the CARLA Town06 map. As shown in Fig. 12, the distance between the starting and goal positions is approximately 110 meters. A total of 7 known objects (vehicles) and 4 unknown objects (two traffic cones, one street barrier, and one trash can) are placed along the route. These unknown objects form two clusters, creating two navigation critical regions.

The trajectories, control profiles, and collaboration states (i.e., $\{\beta_t\}$) of different schemes are shown in Fig. 13. Yellow trajectories represent local navigation, while blue trajectories indicate cloud-guided navigation. Green segments denote cloud engagement without trajectory variations, and red curves represent predicted MPC rollouts. Without cloud collaboration, the LOS scheme conservatively treats the unknown objects as obstacles and avoids them, resulting in the longest trajectory and finish time. The PCS scheme, which relies on a fixed cloud-interaction intervals (5 s), handles the first obstacle cluster but fails to react optimally to the second one, leading to suboptimal detours. In contrast, the SIGMA scheme dynamically triggers cloud interaction based on ODCT-based semantic screening (Section IV-A) and the EPG-guided collaboration strategy, handling both obstacle clusters and achieving the shortest trajectory and finish time.

More importantly, SIGMA reduces the number of cloud interactions by 50% compared with PCS (2 vs. 4 times). This improvement is achieved by avoiding unnecessary cloud

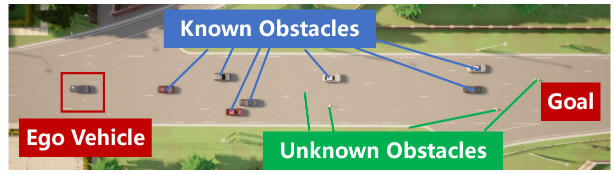


Fig. 12. Static-obstacle scenario configuration for Experiment 2.

TABLE IV
QUANTITATIVE COMPARISON IN EXPERIMENT 2.

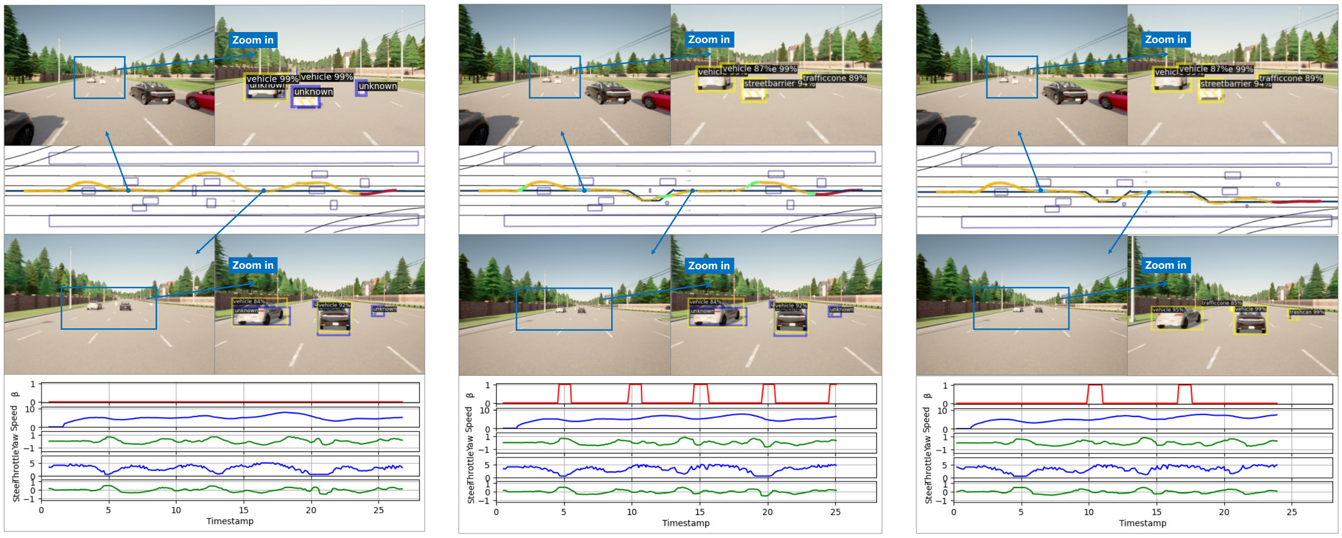
Method Unit	FTime (s)	TLen (m)	AvgLD (m)	SVar (m/s)	MLat (m)
LOS	25.94	124.54	0.53	1.09	1.52
PCS	24.09	122.40	0.59	1.13	1.89
PCS-2	23.53	123.84	0.57	1.39	1.69
SIGMA	22.35	121.85	0.60	1.30	1.76

queries at non-critical locations (green segments in PCS), where improved perception would not lead to meaningful trajectory changes. This demonstrates the **resource efficiency** of the proposed collaboration strategy. Additionally, at the second obstacle cluster, the ego vehicle initially deviates due to incomplete perception but quickly adjusts its trajectory once cloud guidance becomes available. This behavior highlights the **self-correction capability** enabled by the integration of large vision models and MPC.

A quantitative comparison of the schemes is shown in Table IV. We repeat the experiment 10 times and report the average metrics. This evaluation includes finish time (FTime) in seconds, trajectory length (TLen) in meters, average lateral deviation (AvgLD) in meters, speed variability (SVar) in m/s, and maximum lateral deviation (MLat) in meters. To ensure a fair comparison, we also simulate a variant of PCS, termed PCS-2, which adopts a fixed cloud-interaction interval of 5.5 s. Its trajectories and control profiles are shown in Fig. 14. The proposed SIGMA achieves the smallest FTime and TLen, benefiting from cloud-guided LVM–MPC collaboration and EPG-guided collaboration timing.

2) Simulation-in-the-Loop Forward Simulation Analysis: To understand why SIGMA outperforms the baseline schemes, we analyze the planning-gain assessment process through forward simulation within the MPC framework. Instead of relying on perception uncertainty, we evaluate how different uncertainty realizations affect downstream planning outcomes. Specifically, we generate multiple scene samples using Monte Carlo sampling over obstacle uncertainty and perform MPC rollouts for each sampled scenario. Representative results are shown in Fig. 15, where each column corresponds to a different scene sample and the resulting MPC trajectory.

We observe that not all uncertainty realizations lead to meaningful changes in planning. In some cases, different scene samples result in nearly identical trajectories, indicating that resolving such uncertainty would not improve the final planning outcome. However, in critical regions (e.g., near obstacle clusters), different uncertainty realizations lead to significantly different feasible trajectories, directly affecting collision avoidance and path selection.



(a) LOS. (b) PCS. (c) SIGMA (Ours).

Fig. 13. Static-scenario experimental results: trajectories, control profiles, and collaboration states.

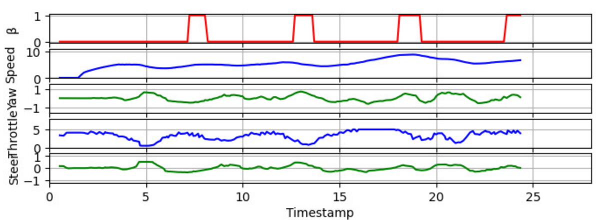


Fig. 14. Trajectory and control profiles of PCS-2.

These results highlight that the impact of uncertainty should be evaluated in the planning space rather than the perception space. Only uncertainty that leads to divergent planning outcomes is decision-relevant. By explicitly evaluating this effect through forward simulation, SIGMA identifies uncertainty with significant expected planning gain and invokes cloud interaction only in decision-critical situations. This simulation-in-the-loop mechanism, as detailed in Section V-C, explains why SIGMA achieves both improved navigation performance and reduced cloud usage compared to fixed-interval strategies.

C. Experiment 3: More Quantitative Evaluations

We evaluate the LOS, PCS, and SIGMA schemes at a lower speed (target speed of 3 m/s) with a shorter horizon ($H = 10$) and a smaller time step ($\Delta t = 0.1\,\mathrm{s}$) in more scenarios. By randomly removing some of the unknown objects in Fig. 12, we configure different scenarios with $U \in \{0, 1, 2\}$. Quantitative results in each scenario are obtained by averaging 30 random simulation runs, with independent realizations in each run. Note that the LOS and PCS schemes may fail due to collision or timeout. Here, we compute trajectory-quality metrics only over their successful cases, while failure cases are used to compute the success rates.

The evaluation results under different numbers of unknown objects are shown in Table V. As the number of unknown obstacles increases from 0 to 2, all schemes require more collision-avoidance maneuvers, leading to increased trajectory length or finish time. When there are no unknown objects, the three schemes show similar performance. With one unknown object, LOS becomes more conservative because it cannot recognize abnormal objects, while PCS benefits from fixed-period cloud assistance and achieves a slightly shorter finish time than SIGMA. However, when multiple unknown objects are present, fixed-period queries may occur at suboptimal times, which degrades PCS performance. In contrast, SIGMA uses cloud-side LVM reasoning to resolve decision-relevant unknown objects and relies on simulation-in-the-loop EPG estimation in Section V-C to determine query timing. As a result, SIGMA achieves the shortest trajectory length in all tested settings and the shortest finish time in the most challenging case with $U = 2$. Therefore, we do not claim that SIGMA is always the fastest; rather, it provides the largest efficiency gain when cloud-collaboration timing becomes decision-critical.

Success rates are computed based on two failure conditions: 1) crash failure, where the ego vehicle collides with any obstacle or lane boundary; and 2) no-progress failure, where the ego vehicle gets stuck or moves off the route and fails to find a feasible path. As shown in Fig. 16, SIGMA maintains a success rate close to $100\,\%$ in all scenarios. PCS works well under 0–1 unknown objects, but causes safety issues under 2 unknown objects, while LOS gives the lowest success rates. Compared with the second-best scheme, SIGMA improves the success rate by over 6%.

D. Experiment 4: Scenarios with Dynamic Obstacles

In this experiment, we evaluate the performance of the proposed framework in a dynamic-obstacle scenario at a crossroad in the CARLA Town03 map, as shown in Fig. 17. The environment includes two unknown objects and three known objects. Among the unknown objects, one is a static doghouse and the other is a dynamic cybertruck moving at a constant speed of 4 m/s. The LOS scheme always fails near

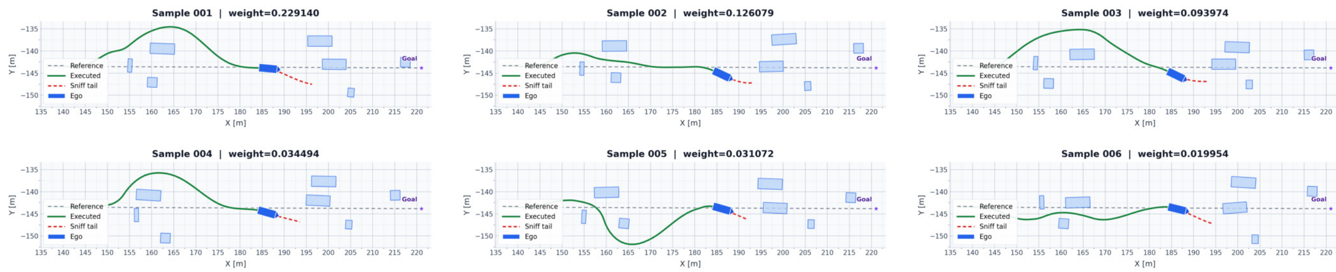


Fig. 15. Simulation-in-the-loop MPC rollouts under different plausible scene samples.

TABLE V
QUANTITATIVE COMPARISON OF EXPERIMENT 3.

U	Method (Unit)	FTime (s)	TLen (m)	AvgLD (m)	SVar (m/s)	MLat (m)
0	LOS	42.47	114.27	0.52	2.34	1.59
0	PCS	42.31	114.63	0.53	2.64	1.41
0	SIGMA	43.00	113.56	0.51	2.41	1.62
1	LOS	48.05	119.95	0.49	2.57	1.54
1	PCS	44.02	114.44	0.53	2.68	1.43
1	SIGMA	44.86	113.67	0.52	2.53	1.64
2	LOS	55.36	119.32	0.40	2.39	1.55
2	PCS	47.60	114.73	0.49	2.23	1.41
2	SIGMA	45.44	112.79	0.54	2.48	1.51

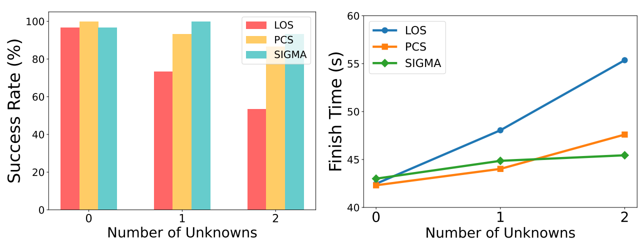


Fig. 16. Comparison of success rate and finish time in Experiment 3.

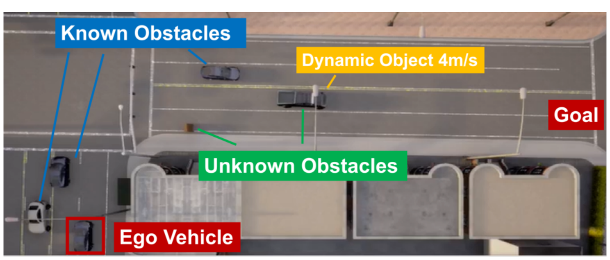
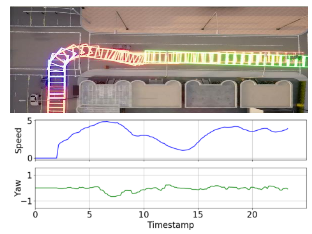
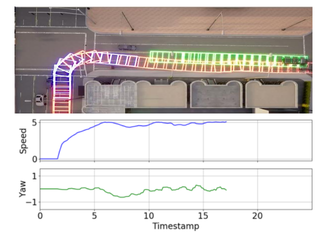


Fig. 17. Dynamic-obstacle scenario configuration for Experiment 4.



(a) PCS.



(b) SIGMA.

Fig. 18. Trajectory and control profiles in Experiment 4.

TABLE VI
QUANTITATIVE COMPARISON IN EXPERIMENT 4.

Method	FTime(s)	TLen(m)	AvgLD(m)	SVar(m/s)	MLat(m)
PCS	23.40	71.47	0.47	2.99	1.68
SIGMA	17.28	72.90	0.62	2.29	1.44

the turning point due to uncertain perception of the doghouse and cybertruck, which overly restricts the feasible solution space of the MPC planner. This failure case illustrates a typical limitation of purely local uncertainty handling: when unknown objects appear in a narrow decision region, conservative obstacle treatment may make the planner unable to find a feasible and efficient maneuver. Therefore, we only compare PCS and SIGMA in this experiment.

For PCS, the ego vehicle turns right while avoiding collision with the doghouse, and then follows the lead vehicle cybertruck until reaching the goal. This behavior is safe but conservative, since the fixed-period collaboration strategy does not necessarily query the cloud when the cybertruck becomes decision-critical. In contrast, under SIGMA, the ego vehicle triggers another cloud query during the car-following phase behind the cybertruck. This is because the simulation-in-the-loop EPG estimation mechanism (Section V-C) identifies that resolving the current uncertainty can still change the downstream planning outcome. In other words, the remaining uncertainty is not only a perception ambiguity, but also a maneuver-level ambiguity between continuing to follow the lead vehicle and overtaking it. The accepted cloud query therefore preserves the opportunity for subsequent collaboration and enables the ego vehicle to overtake the cybertruck by following cloud-generated overtaking waypoints rather than local car-following waypoints, as shown in Fig. 18 (red boxes denote the local controller, blue boxes denote LVM-guided MPC, and green boxes denote the dynamic obstacle).

Consequently, compared with PCS, the proposed SIGMA completes the trip in a significantly shorter time, achieving more than 26.2% reduction in finish time. It is also worth noting that SIGMA does not simply minimize trajectory length; instead, it accepts a slightly longer maneuver when it leads to a better time-efficient decision. This reflects the key advantage of planning-gain-guided collaboration in dynamic scenarios: cloud reasoning is invoked when it can change the driving strategy, rather than merely refine perception. Detailed quantitative results are shown in Table VI.

VIII. CONCLUSION

This paper presented **SIGMA**, a simulation-in-the-loop planning-gain assessment framework for fast–slow collaborative autonomous driving. Instead of relying on perception-level uncertainty or heuristic triggers, SIGMA evaluates uncertainty through its impact on planning outcomes via simulation-in-the-loop forward simulation. We introduced EPG to quantify the expected improvement in planning performance if uncertainty is resolved, which serves as a unified criterion for cloud invocation, cloud-guidance integration, and resource allocation. Semantic uncertainty is modeled as a confidence-based screening signal for unknown-object awareness, while geometric uncertainty is represented probabilistically and propagated into motion planning through chance-constrained obstacle inflation. Under limited cloud resources, an EPG- and deadline-aware allocation mechanism ensures that cloud assistance is provided only when it is both decision-relevant and timely. Experiments in CARLA demonstrate that the proposed framework reduces unnecessary cloud usage while improving planning performance, driving efficiency, and success rates in both static and dynamic obstacle scenarios.

Future work will extend SIGMA along three directions. First, we will move beyond static or weakly dynamic obstacle uncertainty and model interaction-aware uncertainty in dynamic driving scenarios, including multi-modal agent intentions and trajectory prediction ambiguity. Second, we will replace the current threshold-based cloud invocation and resource allocation rules with reinforcement-learning-based policies that optimize long-term planning performance under latency, cost, and safety constraints. Third, we will upgrade the current simulation-in-the-loop EPG estimator into a world-model-in-the-loop framework, where generative world models provide realistic counterfactual rollouts for evaluating the value of cloud reasoning in complex interactive environments. More broadly, this direction connects SIGMA to the physical-AI agent-harness perspective, where simulation, verification, safety checking, and execution monitoring wrap model reasoning before actions are deployed in the physical world.

References

[1] C. Pan, B. Yaman, T. Nesti, A. Mallik, A. G. Allievi, S. Velipasalar, and L. Ren, "VLP: Vision-language planning for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. (CVPR)*, 2024, pp. 14 760–14 769.

[2] X. Tian, J. Gu, B. Li, Y. Liu, Y. Wang, Z. Zhao, K. Zhan, P. Jia, X. Lang, and H. Zhao, "DriveVLM: The convergence of autonomous driving and large vision-language models," in *Proc. Conf. Robot Learn. (CoRL)*, ser. Proc. Mach. Learn. Res., vol. 270. PMLR, 2025, pp. 4698–4726.

[3] J.-J. Hwang, R. Xu, H. Lin, W.-C. Hung, J. Ji, K. Choi, D. Huang, T. He, P. Covington, B. Sapp, Y. Zhou, J. Guo, D. Anguelov, and M. Tan, "EMMA: End-to-end multimodal model for autonomous driving," *Trans. Mach. Learn. Res.*, 2025.

[4] J. Ichnowski, K. Chen, K. Dharmarajan, S. Adebola, M. Danielczuk, V. Mayoral-Vilches, N. Jha, H. Zhan, E. Llontop, D. Xu, C. Buscaron, J. Kubiatowicz, I. Stoica, J. E. Gonzalez, and K. Goldberg, "FogROS2: An adaptive platform for cloud and fog robotics using ROS 2," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2023, pp. 5493–5500.

[5] G. Li, R. Han, S. Wang, F. Gao, Y. C. Eldar, and C. Xu, "Edge accelerated robot navigation with collaborative motion planning," *IEEE/ASME Trans. Mechatronics*, vol. 30, no. 2, pp. 1166–1178, 2025.

[6] J. Chen, S. Wang, G. Li, W. Xu, G. Zhu, D. W. K. Ng, and C. Xu, "Opportunistic collaborative planning with large vision model guided control and joint query-service optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2025, pp. 951–958.

[7] D. Li, B. Liu, Z. Huang, Q. Hao, D. Zhao, and B. Tian, "Safe motion planning for autonomous vehicles by quantifying uncertainties of deep learning-enabled environment perception," *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 2318–2332, 2024.

[8] S. Zhang, H. Li, S. Zhang, S. Wang, D. W. K. Ng, and C. Xu, "Multi-Uncertainty Aware Autonomous Cooperative Planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2024, pp. 1018–1025.

[9] A. Timans, C.-N. Straehle, K. Sakmann, and E. T. Nalisnick, "Adaptive bounding box uncertainties via two-step conformal prediction," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, ser. Lecture Notes in Computer Science. Springer, 2024, pp. 363–398.

[10] M. Lauri and R. Ritala, "Planning for robotic exploration based on forward simulation," *Robotics Auton. Syst.*, vol. 83, pp. 15–31, 2016.

[11] W. Ding, L. Zhang, J. Chen, and S. Shen, "EPSILON: An efficient planning system for automated vehicles in highly interactive environments," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 1118–1138, 2022.

[12] J. Chen, S. Wang, G. Zhu, and C. Xu, "Bridging large-model reasoning and real-time control via agentic fast-slow planning," *[Online]. Available: https://arxiv.org/abs/2604.01681*, 2026. [Online]. Available: https://arxiv.org/abs/2604.01681

[13] Z. Xiao, J. Shu, H. Jiang, G. Min, J. Liang, and A. Iyengar, "Toward collaborative occlusion-free perception in connected autonomous vehicles," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4918–4929, 2024.

[14] X. Ye, K. Qu, W. Zhuang, and X. Shen, "Accuracy-aware cooperative sensing and computing for connected autonomous vehicles," *IEEE Trans. Mobile Comput.*, vol. 23, no. 8, pp. 8193–8207, 2024.

[15] Z. Fang, S. Hu, H. An, Y. Zhang, J. Wang, H. Cao, X. Chen, and Y. Fang, "PACP: Priority-aware collaborative perception for connected and autonomous vehicles," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 15 003–15 018, 2024.

[16] Z. Lin, L. Xiao, H. Chen, and Z. Lv, "Collaborative perception against data fabrication attacks in vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 24, no. 10, pp. 10 654–10 667, 2025.

[17] L. Liang, G. Luo, Y. Lin, L. Deng, N. Cheng, Q. Yuan, J. Li, and D. Niyato, "FullPerception: Network-level collaborative perception for eliminating vehicular blind spots," *IEEE Trans. Mobile Comput.*, vol. 25, no. 2, pp. 1513–1530, 2026.

[18] L. Yang, J. Cheng, M. Zhou, C. Liu, Z. Ni, M. Xie, and S. Gao, "Integrated perception, communication, and computation for autonomous vehicle and road infrastructure network," *IEEE Trans. Mobile Comput.*, pp. 1–13, 2026.

[19] S. Zhou, D. V. Le, and R. Tan, "Edge-cloud switched low-carbon image segmentation for autonomous vehicles," *IEEE Trans. Mobile Comput.*, pp. 1–16, 2026.

[20] M. Tang and V. W. S. Wong, "Deep reinforcement learning for task offloading in mobile edge computing systems," *IEEE Trans. Mobile Comput.*, vol. 21, no. 6, pp. 1985–1997, 2022.

[21] S. Xia, Z. Yao, Y. Li, Z. Xing, and S. Mao, "Distributed computing and networking coordination for task offloading under uncertainties," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 5280–5294, 2024.

[22] Y. Cheng, C. Liang, Q. Chen, and F. R. Yu, "An efficient resource allocation scheme with uncertain network status in edge computing-enabled networks," *IEEE Trans. Mobile Comput.*, vol. 24, no. 3, pp. 1249–1263, 2025.

[23] B. Xu, H. Bian, Q. Cui, X. Yu, J. Qi, and Y. Ji, "Distributed optimization of task offloading and resource allocation for mobile edge computing with multifactorial uncertainty," *IEEE Trans. Mobile Comput.*, vol. 25, no. 2, pp. 1644–1659, 2026.

[24] X. Jin, H. Yang, X. He, G. Liu, Z. Yan, and Q. Wang, "Robust LiDAR-based vehicle detection for on-road autonomous driving," *Remote Sens.*, vol. 15, no. 12, p. 3160, 2023.

[25] J. Choi, D. Chun, H. Kim, and H.-J. Lee, "Gaussian YOLOv3: An accurate and fast object detector using localization uncertainty for autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 502–511.

[26] K. Wong, S. Wang, M. Ren, M. Liang, and R. Urtasun, "Identifying unknown instances for autonomous driving," in *Proc. Conf. Robot Learn. (CoRL)*, ser. Proc. Mach. Learn. Res., vol. 100. PMLR, 2020, pp. 384–393.

[27] W. Liang, F. Xue, Y. Liu, G. Zhong, and A. Ming, "Unknown sniffer for object detection: Don't turn a blind eye to unknown objects," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn. (CVPR)*, 2023, pp. 3230–3239.

[28] D. Huseljic, M. Herde, P. Hahn, M. M"ujde, and B. Sick, "Systematic evaluation of uncertainty calibration in pretrained object detectors," *Int. J. Comput. Vis.*, vol. 133, no. 3, pp. 1033–1047, 2025.

[29] A. Doula, T. G"udelh"ofer, M. M"uhlh"auser, and A. S. Guinea, "Conformal prediction for semantically-aware autonomous perception in urban environments," in *Proc. Conf. Robot Learn. (CoRL)*, ser. Proc. Mach. Learn. Res., vol. 270. PMLR, 2025, pp. 2629–2654.

[30] B. Xu, Q. Li, T. Guo, Y. Ao, and D. Du, "A quantitative safety verification approach for the decision-making process of autonomous driving," in *Proc. Int. Symp. Theor. Aspects Softw. Eng. (TASE)*, 2019, pp. 128–135.

[31] K. Yang, X. Tang, S. Qiu, S. Jin, Z. Wei, and H. Wang, "Towards robust decision-making for autonomous driving on highway," *IEEE Trans. Veh. Technol.*, vol. 72, no. 9, pp. 11 251–11 263, 2023.

[32] X. Li, A. Girard, and I. V. Kolmanovsky, "Safe adaptive cruise control under perception uncertainty: A deep ensemble and conformal tube model predictive control approach," in *Proc. IEEE Conf. Decis. Control (CDC)*, 2025, pp. 3081–3088.

[33] F. Mohseni, S. Voronov, and E. Frisk, "Deep learning model predictive control for autonomous driving in unknown environments," *IFAC-PapersOnLine*, vol. 51, no. 22, pp. 447–452, 2018.

[34] R. Han, S. Wang, S. Wang, Z. Zhang, Q. Zhang, Y. C. Eldar, Q. Hao, and J. Pan, "RDA: An accelerated collision-free motion planner for autonomous navigation in cluttered environments," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1715–1722, 2023.

[35] F. Mohseni, E. Frisk, and L. Nielsen, "Distributed cooperative MPC for autonomous driving in different traffic scenarios," *IEEE Trans. Intell. Veh.*, vol. 6, no. 2, pp. 299–309, 2021.

[36] S. Zhang, S. Wang, S. Yu, J. J. Yu, and M. Wen, "Collision avoidance predictive motion planning based on integrated perception and V2V communication," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 9640–9653, 2022.

[37] M. Popovi'c, J. Ott, J. R"uckin, and M. J. Kochenderfer, "Learning-based methods for adaptive informative path planning," *Robotics Auton. Syst.*, vol. 179, p. 104727, 2024.

[38] K. Kim and J. Kim, "Coordinated informative path planning for multi-robot search in open fields," *J. Intell. Robot. Syst.*, vol. 111, no. 2, p. 65, 2025.

[39] R. Liu, J. Zheng, T. H. Luan, L. Gao, Y. Hui, Y. Xiang, and M. Dong, "ROS-based collaborative driving framework in autonomous vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 6, pp. 6987–6999, 2023.

[40] A. K. Tanwani, R. Anand, J. E. Gonzalez, and K. Goldberg, "RILaaS: Robot inference and learning as a service," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4423–4430, 2020.

[41] B. Gao, J. Liu, H. Zou, J. Chen, L. He, and K. Li, "Vehicle-road-cloud collaborative perception framework and key technologies: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 12, pp. 19 295–19 318, 2024.

[42] Z. Xu, Y. Zhang, E. Xie, Z. Zhao, Y. Guo, K.-Y. K. Wong, Z. Li, and H. Zhao, "DriveGPT4: Interpretable end-to-end autonomous driving via large language model," *IEEE Robot. Autom. Lett.*, vol. 9, no. 10, pp. 8186–8193, 2024.

[43] E. Cui, W. Wang, Z. Li, J. Xie, H. Zou, H. Deng, G. Luo, L. Lu, X. Zhu, and J. Dai, "DriveMLM: Aligning multi-modal large language models with behavioral planning states for autonomous driving," *Vis. Intell.*, vol. 3, p. 22, 2025.

[44] X. Zheng, H. Lu, H. Zhong, X. Chen, P. Wang, M. Zhu, S. Shen, X. Wang, Y. Wang, and F.-Y. Wang, "BEVGPT: Generative pre-trained foundation model for autonomous driving prediction, decision-making, and planning," *IEEE Trans. Intell. Veh.*, pp. 1–13, 2024.

[45] S. Hu, Z. Fang, Z. Fang, Y. Deng, X. Chen, and Y. Fang, "AgentsCoDriver: Large language model empowered collaborative driving with lifelong learning," *[Online]. Available: https://arxiv.org/abs/2404.06345*, 2024. [Online]. Available: https://arxiv.org/abs/2404.06345

[46] H. Sha, Y. Mu, Y. Jiang, L. Chen, C. Xu, P. Luo, S. E. Li, M. Tomizuka, W. Zhan, and M. Ding, "LanguageMPC: Large language models as decision makers for autonomous driving," *[Online]. Available: https://arxiv.org/abs/2310.03026*, 2023. [Online]. Available: https://arxiv.org/abs/2310.03026

[47] L. Wen, D. Fu, X. Li, X. Cai, T. Ma, P. Cai, M. Dou, B. Shi, L. He, and Y. Qiao, "DiLu: A knowledge-driven approach to autonomous driving with large language models," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024.

[48] Z. Han, Y. Wu, T. Li, L. Zhang, L. Pei, L. Xu, C. Li, C. Ma, C. Xu, S. Shen, and F. Gao, "An efficient spatial-temporal trajectory planner for autonomous vehicles in unstructured environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 2, pp. 1797–1814, 2024.

[49] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.

[50] Y. Geifman and R. El-Yaniv, "Selective classification for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017, pp. 4878–4887.

[51] A. Bendale and T. E. Boult, "Towards open set deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 1563–1572.

[52] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Trans. Robot.*, vol. 27, no. 6, pp. 1080–1094, 2011.

[53] C. Dawson, A. Jasour, A. Hofmann, and B. C. Williams, "Provably safe trajectory optimization in the presence of uncertain convex obstacles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2020, pp. 6237–6244.

[54] N. E. D. Toit and J. W. Burdick, "Probabilistic collision checking with chance constraints," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 809–815, 2011.

[55] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, 2nd ed. Springer, 2004.

[56] H. Kahn and A. W. Marshall, "Methods of reducing sample size in Monte Carlo computations," *J. Oper. Res. Soc. Amer.*, vol. 1, no. 5, pp. 263–278, 1953.

[57] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. Conf. Robot Learn. (CoRL)*, ser. Proc. Mach. Learn. Res., vol. 78. PMLR, 2017, pp. 1–16.