A Query-Empowered Digital Twins Framework for Autonomous Driving Scenarios

Nuocheng Yang*, Longyu Zhou†, Sihua Wang*,
Ouwen Huan*, Yechen He*, and Changchuan Yin*
* Beijing Key Laboratory of Network System Architecture and Convergence, State Key
Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications,
Beijing, China, Emails: {yangnuocheng, sihuawang, ouwenh, heyechen, ccyin}@bupt.edu.cn.
† Singapore University of Technology and Design, Singapore, Email: zhoulyfuture@outlook.com.

***Abstract*—Digital twins (DTs) can enhance autonomous driving by optimizing vehicle trajectories through proactive simulation in a high-fidelity replica of the physical environment. However, as the driving environment changes rapidly, real-time state synchronization between the physical and digital world is required to ensure driving reliability. Moreover, traditional push-based state reporting paradigms face challenges in bandwidth-limited communication scenarios. To address this issue, this paper proposes a query-empowered DT framework that enables DTs to actively request states based on their simulation needs. This approach eliminates the transmission of states that are already accurately captured by the DTs' simulation, thereby significantly reducing communication overhead. To support the framework, we formulate an optimization problem to minimize autonomous driving position error, jointly accounting for high DT fidelity and limited communication resources. To solve the problem, the DT can generate and transmit queries to vehicles based on its simulation capabilities. Subsequently, vehicles respond to the query based on their local perception data, thereby reducing the DTs' autonomous driving position error. The simulation results show that our proposed method reduces autonomous driving position error by up to 30% while reducing communication overhead compared to traditional methods.**

***Index Terms*—Digital twin, autonomous driving.**

I. Introduction

Autonomous driving systems can plan vehicle trajectories by understanding the physical environment through local perception from multiple cameras [1], [2]. However, single-vehicle perception capability is inherently limited by occlusions and limited onboard computational resources, leading to incomplete environmental understanding [3], [4]. To fill these gaps, digital twins (DTs) have been leveraged as an enabling framework for an enhanced autonomous driving system. The DTs maintain a high-fidelity replica of the physical environment by combining multiple vehicles' perception [5]. In this framework, the DT can plan vehicle trajectories through proactive simulation with sufficient computing resources. However, the DTs' high-fidelity replica relies on high-frequency state synchronization from multiple vehicles, which introduces unacceptable transmission overhead.

Recently, a number of works [6]–[10] have studied how to reduce transmission overhead for physical-state synchronization in DT mainly from two perspectives: state upload frequency optimization and state compression. For the first aspect, the authors in [6] proposed an age-of-information (AoI)-based method that uploads states only when their age exceeds a predefined threshold. The authors in [8] proposed an event-triggered strategy in which upload occurs only when the state changes significantly. However, these methods may transmit task-irrelevant state, resulting in degraded transmission efficiency and reduced trajectory planning performance. In contrast, for the second aspect, the authors in [9] proposed a semantic communication based DT synchronization method, which compresses state through extracting critical key semantic features for trajectory planning performance. Similarly, the authors in [10] proposed a goal-oriented feature selection method that can filter task-irrelevant features before synchronization. While these methods [9], [10] can effectively reduce the state synchronization data size, they rely on predefined algorithms and lack the flexibility to adapt to highly dynamic driving environments. Furthermore, these traditional push-based state reporting [6]–[10] follow an inherently passive paradigm, in which vehicles still transmit redundant states that have already been captured by the DTs' simulations. In other words, the DT, which serves as the central brain for multi-vehicle autonomous driving, remains a passive consumer of whatever state the vehicles choose to upload, unable to steer the synchronization process toward the state it actually lacks.

To address this challenge, we propose a novel *query-empowered DT* framework that enables the DT to actively request task-relevant states out of simulation, thereby reducing communication overhead. Our contributions are summarized as follows.

- We propose a novel framework that enables DT to actively query vehicle states based on its simulation capabilities to minimize the trajectory planning error. Compared to the traditional passive DT synchronization method, in which vehicles upload states according to predefined policies, the proposed query-empowered DT offers a distinct perspective on DT active information collection. Since DT can actively request only the most informative states, the proposed framework can effectively avoid redundant transmissions.
- We quantify the DT-enhanced automated driving optimization problem for minimal trajectory planning error by jointly considering DT synchronization errors and

communication resource constraints. Explicitly, we first quantify the impact on vehicle trajectory planning by summing the sensitivities of the detected obstacles. Then, we design a query-generation method to minimize the sum of sensitivity of the detected obstacles by requesting state via cross-attention with vehicles' perception.
- Extensive experiments show that our method can effectively reduce communication overhead by up to 7× while reducing trajectory-planning position error by up to 30% compared with traditional full-push algorithms.

II. System Model

We consider an autonomous driving scenario comprising N vehicles and a central DT server S deployed at the base station that can connect to all vehicles via wireless links. The set of vehicles is indexed as $\mathcal{N} = \{1, 2, \cdots, N\}$. The DT-assisted autonomous driving system operates through the following three-step workflow, as shown in Fig. 1.

1) **Vehicle-Side Autonomous Driving Pipeline:** Each vehicle equipped with an onboard autonomous-driving pipeline executes low-level trajectory planning based on local observations.
2) **Query-Response-based DT State Synchronization**: DT generates and transmits queries to vehicles for their responses, and then updates its digital environment replicas.
3) **DT Simulation and Trajectory Planning**: DT updates the environment based on simulation and vehicles' responses, then executes high-level trajectory planning.

A. Vehicle-Side Autonomous Driving Pipeline

We employ an end-to-end autonomous driving model on each vehicle, including single-vehicle perception, prediction, and planning through a unified architecture [2].

Single-Vehicle Perception: Let $\boldsymbol{x}_{i,t} \in \mathbb{R}^{N_{\text{cam}} \times H_{\text{img}} \times W_{\text{img}} \times 3}$ denote the multi-camera images collected by vehicle i at time step t, where N_{cam} is the number of cameras, H_{img} and W_{img} are the spatial dimensions of images. Then, each vehicle reconstructs a bird's-eye view (BEV) feature map by fusing N_{cam} images in $\boldsymbol{x}_{i,t}$ into a comprehensive representation of the surrounding environment, which is given by

$$\boldsymbol{B}_{i,t} = f_{\text{bev}}(\boldsymbol{x}_{i,t}). \tag{1}$$

$\boldsymbol{B}_{i,t} \in \mathbb{R}^{H_b \times W_b \times D}$, where D is the feature dimension, H_b and W_b are the spatial dimensions of BEV.

Then, each vehicle i extracts the representations of surrounding dynamic obstacles (e.g., vehicles, cyclists, and pedestrians) and static map elements (e.g., road boundaries, lanes) from $\boldsymbol{B}_{i,t}$. The dynamic obstacles' representations are formulated as

$$\boldsymbol{z}_{i,t}^{\text{obstacle}} = \boldsymbol{Q}_{\text{obstacle}} + \text{softmax}\Big(\frac{\boldsymbol{Q}_{\text{obstacle}}\, \boldsymbol{B}_{i,t}^{\top}}{\sqrt{D}}\Big)\, \boldsymbol{B}_{i,t}, \tag{2}$$

where $\boldsymbol{Q}_{\text{obstacle}} \in \mathbb{R}^{N_{\text{obstacle}} \times D}$ is the N_{obstacle} learnable obstacle detection queries. $\boldsymbol{z}_{i,t}^{\text{obstacle}} = [\boldsymbol{z}_{i,1,t}^{\text{obstacle}}, \cdots, \boldsymbol{z}_{i,N_{\text{obstacle}},t}^{\text{obstacle}}]$ contains each dynamic obstacle's representations. Similarly, maps' representations are

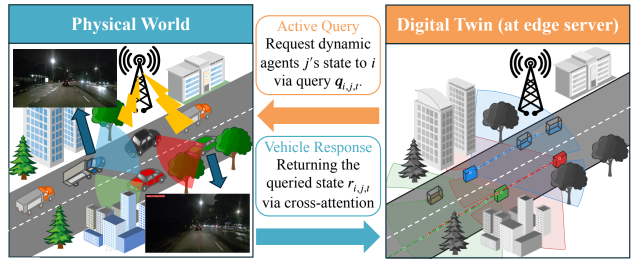

Fig. 1. Illustration of the proposed query-empowered DT in an autonomous driving scenario.

$$\boldsymbol{z}_{i,t}^{\text{map}} = \boldsymbol{Q}_{\text{map}} + \text{softmax}\Big(\frac{\boldsymbol{Q}_{\text{map}}\, \boldsymbol{B}_{i,t}^{\top}}{\sqrt{D}}\Big)\, \boldsymbol{B}_{i,t}, \tag{3}$$

where $\boldsymbol{z}_{i,t}^{\text{map}} \in \mathbb{R}^{N_{\text{map}} \times D}$ is the N_{map} learnable map queries.

Prediction and Planning: Then, each vehicle predicts the future trajectories of detected obstacles, which is given by

$$\boldsymbol{Y}_{i,t},\, \boldsymbol{\pi}_{i,t}, \boldsymbol{z}_{i,t}^{\text{ctx}},\, \boldsymbol{z}_{i,t}^{\text{motion}} = f_{\text{motion}}(\boldsymbol{z}_{i,t}^{\text{obstacle}},\, \boldsymbol{z}_{i,t}^{\text{map}},\, \boldsymbol{B}_{i,t}), \tag{4}$$

where $\boldsymbol{Y}_{i,t} \in \mathbb{R}^{N_{\text{obstacle}} \times T_f \times P \times 2}$ contains predicted 2D positions over a horizon of T_f steps of P trajectory modes, $\boldsymbol{\pi}_{i,t} \in \mathbb{R}^{N_{\text{obstacle}} \times P}$ stacks the prediction probability of P trajectories. $\boldsymbol{z}_{i,t}^{\text{ctx}}$ is a motion context embedding, $\boldsymbol{z}_{i,t}^{\text{motion}} \in \mathbb{R}^{N_{\text{obstacle}} \times P \times D}$ stacks the per-obstacle per-mode hidden representation. $\boldsymbol{z}_{i,t}^{\text{motion}} = [\boldsymbol{z}_{i,1,t}^{\text{motion}}, \cdots, \boldsymbol{z}_{i,N_{\text{obstacle}},t}^{\text{motion}}]$ contains each obstacle's motion representations.

Given the predicted future trajectory of all obstacles, each vehicle i then plans its trajectory as

$$\boldsymbol{\tau}_{i,t} = f_{\text{plan}}(\boldsymbol{z}_{i,t}^{\text{ctx}},\, \boldsymbol{Y}_{i,t},\, \boldsymbol{g}_{i,t}), \tag{5}$$

where $\boldsymbol{g}_{i,t}$ is a navigation command vector.

B. Query-Response-based DT State Synchronization

We assume that the DT holds the map representation $\hat{\boldsymbol{z}}^{\text{map}} \in \mathbb{R}^{N_{\text{map}} \times D}$, which directly retrieves static map elements from the offline HD-map [11]. Hence, in addition to map representation, the DT only needs to synchronize the states of all dynamic obstacles. To effectively reconstruct a high-fidelity replica of the obstacles, the proposed framework enables DT to actively request vehicles for obstacles' states via a query-response mechanism. Specifically, DT transmits a query $\boldsymbol{q}_{i,j,t}$ to vehicle i to request the state of obstacles j. Upon receiving $\boldsymbol{q}_{i,j,t}$, vehicle i then calculates the response via cross-attention calculation as

$$\boldsymbol{r}_{i,j,t} \;=\; \sum_{\ell=1}^{P} \text{softmax}\left(\frac{(\boldsymbol{W}_k[\boldsymbol{z}_{i,j,t}^{\text{loc}}]_\ell)^{\top} \boldsymbol{q}_{i,j,t}}{\sqrt{d_q}}\right) (\boldsymbol{W}_v[\boldsymbol{z}_{i,j,t}^{\text{loc}}]_\ell), \tag{6}$$

where $\boldsymbol{z}_{i,j,t}^{\text{loc}} = \left[\boldsymbol{z}_{i,j,t}^{\text{obstacle}},\; \boldsymbol{z}_{i,j,t}^{\text{motion}}\right]$ represent vehicle i assembles its local representations stack about the obstacle

j. $[\boldsymbol{z}^{\text{loc}}_{i,j,t}]_\ell = \big[[\boldsymbol{z}^{\text{obstacle}}_{i,j,t}]_l,\ [\boldsymbol{z}^{\text{motion}}_{i,j,t}]_l\big]$ is the ℓ-th trajectory mode's representation. $\boldsymbol{W}_k \in \mathbb{R}^{d_q \times D}$ and $\boldsymbol{W}_v \in \mathbb{R}^{d_v \times D}$ are shared projections distributed to each vehicle, and $d_v \ll D$. Then, the vehicle i uploads the response $\boldsymbol{r}_{i,j,t}$ back to DT, whose data size is $C = (d_q + d_v) \times b$ bits, where b is the quantization bit-width.

We assume the distance between vehicle i at position $\boldsymbol{v}_{i,t} \in \mathbb{R}^3$ and DT server S at position $\boldsymbol{p}_{\text{server}} \in \mathbb{R}^3$ is $d^{\text{2D}}_{i,t} = \|[\boldsymbol{v}_{i,t}]_{1:2} - [\boldsymbol{p}_{\text{server}}]_{1:2}\|_2$ and $d^{\text{3D}}_{i,t} = \|\boldsymbol{v}_{i,t} - \boldsymbol{p}_{\text{server}}\|_2$. The line-of-sight (LOS) probability is distance-dependent in the 2D plane as $\mathbb{P}_{\text{LOS}}(d^{\text{2D}}_{i,t}) = \min(d_1/d^{\text{2D}}_{i,t}, 1)(1 - e^{-d^{\text{2D}}_{i,t}/d_2}) + e^{-d^{\text{2D}}_{i,t}/d_2}$. The path loss is given by

$$\text{PL}(d^{\text{3D}}_{i,t}) = \begin{cases} \alpha_{\text{L}} + \beta_{\text{L}} \log_{10}(d^{\text{3D}}_{i,t}) + \gamma \log_{10}(f_c), \text{w.p. } \mathbb{P}_{\text{LOS}}(d^{\text{2D}}_{i,t}), \\ \alpha_{\text{N}} + \beta_{\text{N}} \log_{10}(d^{\text{3D}}_{i,t}) + \gamma \log_{10}(f_c), \text{otherwise}, \end{cases} \tag{7}$$

where d_1 and d_2 are the hyperparameter, f_c is the carrier frequency and $\alpha_{\text{L}}, \beta_{\text{L}}, \alpha_{\text{N}}, \beta_{\text{N}}$, γ are model-specific coefficients. The data transmission delay is

$$\tau_{i,t}(\boldsymbol{A}_t, \boldsymbol{v}_{i,t}) = \sum_{j=1}^{|\mathcal{O}_{i,t}|} \frac{u_{i,j,t} \cdot C}{A_{i,t}\, B_w \log_2\Big(1 + \frac{P_{\text{tx}}}{A_{i,t} B_w N_0 \text{PL}(d^{\text{3D}}_{i,t})}\Big)} \tag{8}$$

where $\boldsymbol{A}_t = [A_{1,t}, \ldots, A_{N,t}]$ is the resources blocks (RBs) allocation vector of the DT server at iteration t and $\sum_{i=1}^{N} A_{i,t} \le R$ with $A_{i,t} \in [0, \cdots, R]$, where R is the total number of RBs. The set of vehicle i's observable obstacles is indexed as $\mathcal{O}_{i,t} = \{1, 2, \cdots, M_t\}$. B_w is the unit channel bandwidth of each RB, P_{tx} is the transmit power, and N_0 is the noise power spectral density. $u_{i,j,t} \in [0, 1]$ indicate the state synchronization vector with $u_{i,j,t} = 1$ implying that DT require vehicle i for obstacle j's state at iteration t (i.e., $A_{i,t} > 0$), and $u_{i,j,t} = 0$, otherwise.

C. DT Simulation and Trajectory Planning

Given the vehicle responses, the DT reconstructs obstacle j's states from per-vehicle perspectives and fuses them into a multi-vehicle representation. Then, DT simulates the trajectories of obstacles to optimize the vehicle trajectories.

Per-Vehicle State Reconstruction: Let $\boldsymbol{h}_{i,j,t}$ denote the memory of each obstacle j observed by vehicle i that is maintained by the DT server. Specifically, when DT requested obstacle j's state based on $\boldsymbol{q}_{i,j,t}$, $\boldsymbol{h}_{i,j,t}$ will update according to the response $\boldsymbol{r}_{i,j,t}$. Otherwise, when DT simulates the memory state according to the stored memory, which is given by

$$\boldsymbol{h}_{i,j,t} = \begin{cases} f_{\text{update}}(\boldsymbol{h}_{i,j,t-1}, \boldsymbol{q}_{i,j,t}, \boldsymbol{r}_{i,j,t}, \hat{\boldsymbol{z}}^{\text{map}}), & \text{if } u_{i,j,t} = 1, \\ f_{\text{simulate}}\big(\boldsymbol{h}_{i,j,t-1}, \hat{\boldsymbol{z}}^{\text{map}}\big), & \text{otherwise}, \end{cases} \tag{9}$$

where $\boldsymbol{h}_{i,j,t-1}$ is the memory at $t-1$ slot. Then, DT can predict each obstacle j's trajectory based on vehicle i's perspective as

$$\hat{\boldsymbol{Y}}_{i,j,t} = f_{\text{motion}}(\boldsymbol{h}_{i,j,t},\ \hat{\boldsymbol{z}}^{\text{map}}), \tag{10}$$

where $\hat{\boldsymbol{Y}}_{i,j,t} \in \mathbb{R}^{T_f \times P \times 2}$ contains a set of detected obstacle $\hat{\mathcal{N}}^{\text{obstacle}}_{i,t} = \{1, 2, \cdots, M_t\}$ predicted 2D positions over a horizon of T_f steps of P trajectory modes.

Multi-Vehicle Latent Aggregation: Given the position of each obstacle from each vehicle, the DT aggregates them into a unified panoramic representation in three steps.

1) Coordinate alignment: The DT applies the detection head to each reconstructed trajectory $\hat{\boldsymbol{Y}}_{i,j,t}$ to obtain bounding-box center positions and transforms them to the global frame as
$$\hat{\boldsymbol{Y}}^{g}_{i,j,t} = \boldsymbol{R}^{\text{e2g}}_{i,t}\, \hat{\boldsymbol{Y}}_{i,j,t} + \boldsymbol{T}^{\text{e2g}}_{i,t}, \tag{11}$$
where $\boldsymbol{R}^{\text{e2g}}_{i,t}$ and $\boldsymbol{T}^{\text{e2g}}_{i,t}$ are the known ego-to-global location transition matrix.
2) Cross-vehicle association: Then, DT constructs a pairwise obstacles similarity matrix between each pair of vehicles $(i, j), \forall i, j \in \mathcal{N}$ and their detection obstacles $(m, n), \forall m \in \hat{\mathcal{N}}^{\text{obstacle}}_{i,t}, n \in \hat{\mathcal{N}}^{\text{obstacle}}_{j,t}$. DT can calculate their box central distance as $d_{m,n,t} = \|\hat{\boldsymbol{Y}}^{g}_{i,m,t} - \hat{\boldsymbol{Y}}^{g}_{j,n,t}\|_2$ and solves the optimal obstacle alignment via the Hungarian algorithm [12], where only the matched pairs with $d_{m,n,t} \le d_{\text{gate}}$ are merged. Then we define the set of vehicles that can observe obstacle j as $\mathcal{U}_{j,t} \subseteq \{1, \ldots, N\}$. We assume the total number of distinct obstacles identified by vehicles at time slot t is M_t. The set of obstacles is indexed as $\mathcal{M}_t = \{1, 2, \cdots, M_t\}$.
3) Covariance-weighted fusion: Let $d^{\text{obs}}_{i,j,t} = \|\hat{\boldsymbol{Y}}^{g}_{i,j,t} - \boldsymbol{v}_{i,t}\|_2$ denote vehicle i's observation distance to obstacle j, and each vehicle-obstacle pair a scalar estimation uncertainty:
$$\sigma^2_{i,j,t} = \sigma^2_0 + \kappa_{\text{obstacle}} \left(d^{\text{obs}}_{i,j,t}\right)^2, \tag{12}$$
where σ^2_0 is the sensor floor variance arising from pixel quantization and BEV grid resolution, and κ_{obstacle} scales the per-square-meter depth-uncertainty contribution. Following the Kalman information-filter fusion, the DT computes the aggregated latent token for obstacle j by precision-weighted averaging, which is given by
$$\hat{\boldsymbol{z}}^{\text{agg}}_{j,t} = \frac{\sum_{i \in \mathcal{U}_{j,t}} \sigma^{-2}_{i,j,t}\, \boldsymbol{h}_{i,j,t}}{\sum_{i \in \mathcal{U}_{j,t}} \sigma^{-2}_{i,j,t}}. \tag{13}$$

Trajectory Prediction: Then, DT can predict the trajectory of the detected obstacles based on the aggregated observation, which is given by

$$\hat{\boldsymbol{Y}}'_t,\ \hat{\boldsymbol{\pi}}_t,\ \hat{\boldsymbol{z}}^{\text{ctx}}_t,\ \hat{\boldsymbol{z}}^{\text{motion}}_t = f_{\text{motion}}(\hat{\boldsymbol{z}}^{\text{agg}}_t,\ \hat{\boldsymbol{z}}^{\text{map}}), \tag{14}$$

where $\hat{\boldsymbol{z}}^{\text{agg}}_t = [\hat{\boldsymbol{z}}^{\text{agg}}_{1,t}, \ldots, \hat{\boldsymbol{z}}^{\text{agg}}_{M_t,t}]$ denotes M_t obstacles' memory, $\hat{\boldsymbol{Y}}'_t = \Big[\hat{\boldsymbol{Y}}'_{1,t}, \ldots, \hat{\boldsymbol{Y}}'_{M_t,t}\Big] \in \mathbb{R}^{M_t \times T_f \times P \times 2}$ contains P predicted trajectory modes over T_f future steps for each tracked obstacles, $\hat{\boldsymbol{\pi}}_t = [\hat{\boldsymbol{\pi}}_{1,t}, \ldots, \hat{\boldsymbol{\pi}}_{M_t,t}] \in [0, 1]^{M_t \times P}$ where $\hat{\boldsymbol{\pi}}_{j,t} = [\hat{\pi}^1_{j,t}, \ldots, \hat{\pi}^P_{j,t}]$ is the trajectory probability distribution of obstacle j which satisfies $\sum_{p \in [1,P]} \hat{\pi}^p_{j,t} = 1, \forall j \in \mathcal{M}_t$. $\hat{\boldsymbol{z}}^{\text{ctx}}_t$ is the DT's motion context embedding, and $\hat{\boldsymbol{z}}^{\text{motion}}_t \in \mathbb{R}^{M_t \times P \times D}$ stacks the aggregated per-obstacle per-mode hidden tokens.

Trajectory Planning: Based on the aggregated motion context that accounts for occluded obstacles and cross-vehicle interactions, the DT then computes a trajectory for each vehicle i, which is given by

$$\hat{\boldsymbol{\tau}}_{i,t} = f_{\mathrm{plan}}(\hat{\boldsymbol{z}}_t^{\mathrm{ctx}},\, \hat{\boldsymbol{Y}}_t',\, \boldsymbol{g}_{i,t}). \tag{15}$$

D. Problem Formulation

We formulate an optimization problem to minimize the trajectory planning gap between the DT constructed via the proposed query-response state synchronization method and an idealized DT, under communication resource constraints, which is given by

$$\min_{\boldsymbol{u}_t,\, \{\boldsymbol{q}_{i,j,t}\},\, \boldsymbol{A}_t} \quad \sum_{t\in\mathcal{T}}\sum_{i=1}^{N} \mathbb{E}[\|\hat{\boldsymbol{\tau}}_{i,t} - \boldsymbol{\tau}_{i,t}^*\|^2], \tag{16}$$

$$\text{s.t.} \quad \tau_{i,t} \le \tau_{\max}, \forall\, i\in\mathcal{N}, \forall t\in\mathcal{T}, \tag{16a}$$

$$\mathbb{E}[\|\hat{\boldsymbol{z}}_{j,t}^* - \hat{\boldsymbol{z}}_{j,t}^{\mathrm{agg}}\|^2] \le \sigma_{\max}, \forall\, j\in\mathcal{M}_t, \tag{16b}$$

$$\sum_{i=1}^{N} A_{i,t} \le R, \forall t\in\mathcal{T}, \tag{16c}$$

where $\boldsymbol{\tau}_{i,t}^*$ denotes the trajectory produced based on an idealized DT that possesses infinite fidelity and instantaneous synchronization. Similarly, we define the idealized state $\hat{\boldsymbol{z}}_{j,t}^*$ as the aggregated obstacle j's representation with idealized DT. $\sigma_{\max}$ is the maximum allowable state reconstruction error. Constraint (16a) bounds the per-vehicle communication delay. Constraint (16b) constrains the DT reconstruction gap. Constraint (16c) ensures the available resources with maximum R at each time slot.

The problem in (16) is challenging to solve due to two key issues. First, the relationship between the trajectory planning gap and each obstacle is difficult to explicitly characterize, making direct optimization intractable. Second, the decision space is highly coupled and combinatorial, jointly involving active query design $\boldsymbol{q}_{i,j,t}$, and resource allocation $\boldsymbol{A}_t$.

III. Problem Analysis

We analyze the impact of each obstacle j on the trajectory planning gap via conditional sampling. Specifically, we can sample a trajectory index $k_{j,t}$ according to the trajectory probability $\hat{\boldsymbol{\pi}}_{j,t}$ from vehicle j and obtain the corresponding trajectory realization as $[\hat{\boldsymbol{Y}}_{j,t}']_{k_{j,t}}$. To isolate the impact of obstacle j, we make the following assumption:

- *Assumption* 1*:* The trajectory distributions $\hat{\boldsymbol{\pi}}_{j,t}$ of different obstacles $j\in\mathcal{M}_t$ are mutually independent.

Under this assumption, we construct a conditional scenario by fixing other obstacles and varying only obstacle j's trajectory sampling. The DT then generates the corresponding planned trajectory for vehicle i as

$$\hat{\boldsymbol{\tau}}_{i,t}^{(j,p)} = f_{\mathrm{plan}}\big(\hat{\boldsymbol{z}}_t^{\mathrm{ctx}},\, \tilde{\boldsymbol{Y}}_t^{(j,p)},\, \boldsymbol{g}_{i,t}\big), \tag{17}$$

where $\tilde{\boldsymbol{Y}}_t^{(j,p)} \in \mathbb{R}^{M_t\times T_f\times 2}$ denotes the counterfactual joint trajectory set. In $\tilde{\boldsymbol{Y}}_t^{(j,p)}$ only obstacle j is set to its p-th trajectory realization, while all other obstacles are fixed to their nominal trajectories, which is given by

$$\big[\tilde{\boldsymbol{Y}}_t^{(j,p)}\big]_\ell = \begin{cases} \hat{\boldsymbol{Y}}_{j,t}^{(j,p)}, & \ell = j, \\ \bar{\boldsymbol{Y}}_{\ell,t}, & \ell \ne j, \end{cases} \tag{18}$$

where

$$\bar{\boldsymbol{Y}}_{\ell,t} = \sum_{q=1}^{P} \hat{\pi}_{\ell,t}^{q}\hat{\boldsymbol{Y}}_{\ell,t}^{(q)} \tag{19}$$

is the nominal trajectory of obstacle ℓ. Then, we derive the following Lemma:

Lemma 1. (Planning Gap Upper Bound) Under Assumptions 1, the expected planning gap in (16) is upper bounded by

$$\sum_{i=1}^{N}\mathbb{E}\big[\|\hat{\boldsymbol{\tau}}_{i,t} - \boldsymbol{\tau}_{i,t}^*\|^2\big] \;\le\; \sum_{j=1}^{M_t}\sum_{i=1}^{N}\sum_{p=1}^{P}\hat{\pi}_{j,t}^{p}\,\big\|\hat{\boldsymbol{\tau}}_{i,t}^{(j,p)} - \hat{\boldsymbol{\tau}}_{i,t}\big\|^2 + \Delta, \tag{20}$$

where Δ is a fixed error that is introduced by the interactions between obstacles and planner approximation errors.

From Lemma 1, we observe that the expected trajectory planning gap is upper-bounded by the trajectory sensitivity $\alpha_{j,t}$. Then, problem (16) can be rewritten as:

$$\min_{\boldsymbol{u}_t,\, \{\boldsymbol{q}_{i,j,t}\},\, \boldsymbol{A}_t} \sum_{t\in\mathcal{T}}\sum_{j=1}^{M_t}\sum_{i=1}^{N}\sum_{p=1}^{P}\hat{\pi}_{j,t}^{p}\,\big\|\hat{\boldsymbol{\tau}}_{i,t}^{(j,p)} - \hat{\boldsymbol{\tau}}_{i,t}\big\|^2, \quad \text{s.t. (16a)} - \text{(16c)}. \tag{21}$$

Then, we can further decouple the rewritten problem (21) into two steps:

1) DT should select a subset of obstacles j for state synchronization;
2) DT should also decide the query design while considering the communication constraint.

This decoupling can reduce computational burden by avoiding joint search over the massive space of all possible obstacles across all vehicles.

IV. Query-Empowered DT

A. Obstacle Selection Method

The first step is to select a subset of obstacles that is most important based on the DTs' simulation capabilities by calculating $\sum_{j=1}^{M_t}\sum_{i=1}^{N}\sum_{p=1}^{P}\hat{\pi}_{j,t}^{p}\,\big\|\hat{\boldsymbol{\tau}}_{i,t}^{(j,p)} - \hat{\boldsymbol{\tau}}_{i,t}\big\|^2$ through simulation. For clarity, we define the shorthand $\alpha_{j,t} \triangleq \sum_{i=1}^{N}\sum_{p=1}^{P}\hat{\pi}_{j,t}^{p}\,\big\|\hat{\boldsymbol{\tau}}_{i,t}^{(j,p)} - \hat{\boldsymbol{\tau}}_{i,t}\big\|^2$ as obstacle j' sensitivity. To reduce the simulation overhead for $\alpha_{j,t}$, we employ a Bayesian predictor cast as offline supervised regression as

$$(\hat{\mu}_{j,t},\, \hat{\sigma}_{j,t}) = Q_\psi(\hat{\boldsymbol{z}}_{j,t}^{\mathrm{agg}},\, \hat{\boldsymbol{Y}}_{j,t},\, \hat{\boldsymbol{\pi}}_{j,t},\, \hat{\boldsymbol{z}}_{j,t}^{\mathrm{motion}}), \tag{22}$$

where $\hat{\mu}_{j,t}$ is the predicted mean sensitivity of obstacle j and $\hat{\sigma}_{j,t}$ is the predicted uncertainty of the sensitivity. Then, the predicted sensitivity is given by a Gaussian distribution $\hat{\alpha}_{j,t} \sim \mathcal{N}(\hat{\mu}_{j,t}, \exp(\hat{\sigma}_{j,t}))$. The Bayesian network used for prediction is trained on the heteroscedastic Gaussian negative log-likelihood, which jointly calibrates prediction and uncertainty, whose loss function is

$$\mathcal{L}_{Q_\psi}(\hat{\alpha}_{j,t},\, \alpha_{j,t}) \;=\; \sum_{j=1}^{M_t} \frac{(\hat{\mu}_{j,t}-\alpha_{j,t})^2}{2\,\exp(\hat{\sigma}_{j,t})} + \frac{\hat{\sigma}_{j,t}}{2}. \tag{23}$$

B. Query Design

The next step is to minimize the sensitivity $\alpha_{j,t}$ by collecting states through an active query. To enable stable state collection, we design a query scheme which is given by

$$\boldsymbol{q}_{i,j,t} \;=\; g_\phi\big(\hat{\boldsymbol{z}}^{\text{agg}}_{j,t},\; \hat{\boldsymbol{Y}}_{j,t},\; \hat{\boldsymbol{\pi}}_{j,t},\; \hat{\boldsymbol{z}}^{\text{motion}}_{j,t}\big). \tag{24}$$

After receiving the response $\boldsymbol{r}_{i,j,t}$ generated by vehicle i through cross-attention between query $\boldsymbol{q}_{i,j,t}$ and obstacles representation $\boldsymbol{z}^{\text{loc}}_{i,j,t}$, the DT updates the obstacle representation following (9). The updated representation is then propagated through the DT motion prediction and planning modules in (14) and (15), yielding the post-query trajectory sensitivity $\tilde{\alpha}_{j,t}$. Therefore, the query generator is trained to minimize the residual sensitivity after query-response state synchronization, which is given by

$$\mathcal{L}_{g_\phi} = \sum_{j=1}^{M_t} \tilde{\alpha}_{j,t}. \tag{25}$$

The trajectory sensitivity estimation parameter and the query are trained offline separately, since they serve different roles in minimizing the trajectory planning gap. The sensitivity estimator Q_ψ learns to approximate the obstacle-level sensitivity $\alpha_{j,t}$ and provides a lightweight criterion for selecting planning-critical obstacles based on DT's simulation capability. Given the selected obstacles, the query generator g_ϕ then learns to generate queries that retrieve informative responses from vehicles and reduce the post-query residual sensitivity.

C. Communication-Constrained Query Selection

For each queried obstacle j, DT broadcasts a query $\boldsymbol{q}_{i,j,t} \in \mathbb{R}^{d_q}$ to its observer set $\mathcal{U}_{j,t}$, and each vehicle $i \in \mathcal{U}_{j,t}$ returns a response $\boldsymbol{r}_{i,j,t} \in \mathbb{R}^{d_v}$ via (6). Thus, the per-obstacle transmission data size is $C = |\mathcal{U}_{j,t}|\,(d_q + d_v)\,b$. For each vehicle i, we define its aggregated importance as the sum of sensitivity scores of all obstacles it currently observes as

$$S_{i,t} = \textstyle\sum_{j \in \mathcal{O}_{i,t}} \hat{\alpha}_{j,t}. \tag{26}$$

Then, RBs are allocated proportionally to each vehicle's aggregated importance:

$$A_{i,t} = R \cdot \frac{S_{i,t}}{\sum_{n=1}^{N} S_{n,t}}. \tag{27}$$

This allocation maximizes the total sensitivity-weighted throughput $\sum_{i=1}^{N} S_{i,t} \cdot R_{i,t}(A_{i,t}, d^{\text{3D}}_{i,t})$, i.e., the amount of planning-critical data transmitted per frame, subject to the RB budget constraint (16c). A vehicle observing many high-sensitivity obstacles receives more bandwidth, while one observing only low-sensitivity obstacles yields resources.

For delay feasibility, if some vehicle i violates the delay constraint (16a) under (27), we first allocate the minimum RBs $A_{i,\min}$ satisfying $\tau_{i,t}(A_{i,\min}) = \tau_{\max}$, then redistribute the remaining RBs among other vehicles proportionally to their aggregated importance scores. Under this allocation, the bottleneck upload rate for obstacle j is $R_{\text{ul},j} = \min_{i \in \mathcal{U}_{j,t}} R_{i,t}(A_{i,t}, d^{\text{3D}}_{i,t})$. Each frame can thus serve at most $\lfloor \tau_{\max} R_{\text{ul},j} / C_j \rfloor$ obstacles. Then, DT can greedily select obstacles with the highest sensitivity-plus-uncertainty scores:

$$\mathcal{Q}^*_t = \text{TopK}\Big(\big\{\hat{\alpha}_{j,t}\big\}_{j=1}^{M_t},\; \Big\lfloor \frac{\tau_{\max} \cdot R_{\text{ul},j}}{C_j} \Big\rfloor\Big). \tag{28}$$

V. Performance Evaluation

For our simulations, we consider a multi-vehicle autonomous driving scenario with 5 vehicles, each equipped with 6 RGB cameras and approximately 20 obstacles per frame based on the V2X-Sim dataset [13]. We run the UniAD autonomous driving model [2] on each vehicle to generate middles representations (e.g., $\boldsymbol{B}_{i,t}$ and $\boldsymbol{z}^{\text{obstacle}}_{i,t}$). The DT maintains a model that predicts the trajectory of obstacles, and the planning of vehicles' trajectories also follows UniAD-style via constant-velocity extrapolation between query updates. For comparison, we utilize 4 baselines:

1) **Semantic (obstacle-level):** Vehicles transmit compressed obstacle representation via a semantic encoder and decode at the server with 50% compression rate (labeled "Baseline (a)").
2) **AoI (obstacle-level)** method transmits the obstacle's representation whose sync age exceeds a predefined threshold (labeled "Baseline (b)").
3) **Semantic (image-level):** Vehicles transmit compressed images via a semantic encoder, and the DT reconstructs them using a semantic decoder whose compression rate is 50% [14] (labeled "Baseline (c)").
4) **Full-Push (image-level):** Each vehicle uploads raw camera images every frame; the DT runs UniAD server-side and fuses detections via confidence-weighted averaging (labeled "Baseline (d)").

Fig. 2 shows how the averaged mean position error of vehicles changes as the number of RBs varies from 0 to 80. From Fig. 2, we observe that the mean position error of vehicles decreases as the number of RBs increases. This is because with more communication resources, each vehicle can transmit more of the perceived obstacles' state information. This improves the accuracy of DT construction and thereby reduces the average mean position error. From Fig. 2, we can also observe that the proposed method's average mean position error decreases more rapidly than others. This is because the proposed method can effectively distinguish between obstacles whose states are well captured by DT simulation and those that are not, and transmits only the highly

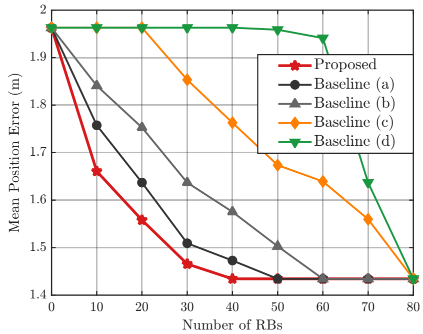

Fig. 2. Mean position error vs. RB.

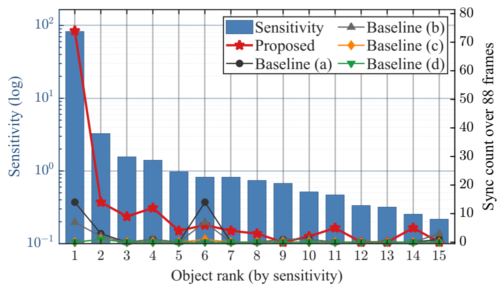

Fig. 3. Per-obstacle sensitivity $\hat{\alpha}_{j,t}$ v.s. sync frequency.

uncertain ones, thereby achieving lower position error under limited communication resources. In contrast, other baselines transmit each vehicle's ego-centric perceptions, ignoring the DT's actual information needs. Our method achieves a 5% and 30% decrease in mean position error compared to the semantic compression (obstacle-level) method and the full-push method, respectively, at extremely low RBs. This is because the proposed query-response method achieves a higher extraction rate through cross-attention query design than semantic compression, whose goal is to reconstruct the entire physical environment.

Fig. 3 shows the per-obstacle communication resources allocation at 30 RB over 88 frames. We observe that our method focuses on the top-ranked obstacles by predicted sensitivity $\hat{\alpha}_{j,t}$, where the two highest-sensitivity obstacles receive 40–77 time of syncs while lower-ranked obstacles receive fewer than 10. In contrast, the baselines allocate communication resources roughly evenly across obstacles. This is because the proposed method can explicitly account for the DT's assessment of each obstacle's impact and allocate resources accordingly, whereas methods that rely on passive uploading cannot fully capture the DT's requirements and therefore distribute available communication resources uniformly across all obstacles' state information.

VI. Conclusion

In this paper, we propose a query-empowered DT framework in which the DT can actively request vehicles' states for trajectory planning via a query rather than passively receive pre-simulated data. To achieve this goal, we formulate an optimization problem to minimize the trajectory-planning gap between the DT constructed via the proposed query-response state synchronization method and an idealized DT. To tackle this intractable problem, we derive an upper bound of the planning gap, expressed as the sum of obstacle-wise sensitivities, which can be efficiently estimated through DT simulation. Based on this insight, the problem is decomposed into two subproblems: obstacle selection and query generation. We propose a query-generation method that aims to minimize post-query sensitivities of obstacles. Experiments on the V2X-Sim dataset demonstrate that the proposed method significantly improves trajectory planning performance under communication resource constraints.

References

[1] B. Jiang, S. Chen, Q. Xu, B. Liao, J. Chen, H. Zhou, Q. Zhang, W. Liu, C. Huang, and X. Wang, "VAD: Vectorized Scene Representation for Efficient Autonomous Driving," in *Proc. IEEE Int. Conf. Comput. Vis.*, Paris, France, Oct. 2023.

[2] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang *et al.*, "Planning-oriented autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Vancouver, Canada, June 2023.

[3] R. Xu, H. Xiang, Z. Tu, X. Xia, M.-H. Yang, and J. Ma, "V2X-ViT: Vehicle-to-Everything Cooperative Perception with Vision Transformer," in *Proc. Eur. Conf. Comput. Vis.*, Tel Aviv, Israel, Oct. 2022.

[4] Y. Hu, S. Fang, Z. Lei, Y. Zhong, and S. Chen, "Where2comm: Communication-Efficient Collaborative Perception via Spatial Confidence Maps," in *Adv. Neural Inf. Process. Syst.*, vol. 35, New Orleans, Louisiana, USA, Nov. 2022.

[5] X. Gu, W. Duan, G. Zhang, J. Hou, L. Peng, M. Wen, F. Gao, M. Chen, and P.-H. Ho, "Digital Twin Technology for Intelligent Vehicles and Transportation Systems: A Survey on Applications, Challenges and Future Directions," *IEEE Commun. Surveys Tuts.*, vol. 28, pp. 3235–3271, June 2025.

[6] Z. Zhou, J. Steiger, Y. Sun, and B. Li, "Low-Overhead Scheduling for Synchronization in Large-Scale Heterogeneous Digital Twin Systems," in *Proc. IEEE Int. Conf. Comput. Commun.*, Vancouver, Canada, May. 2026.

[7] Y. Zhang, W. Liang, Z. Xu, W. Xu, and M. Chen, "AoI-Aware Inference Services in Edge Computing via Digital Twin Network Slicing," *IEEE Trans. Serv. Comput.*, vol. 17, no. 6, pp. 3154–3170, Aug. 2024.

[8] H. Farag and C. Stefanovic, "Toward Efficient Deployment and Synchronization in Digital Twins-Empowered Networks," vol. abs/2604.00566, 2026. [Online]. Available: https://arxiv.org/abs/2604.00566

[9] H. Tong, M. Chen, J. Zhao, Y. Hu, Z. Yang, Y. Liu, and C. Yin, "Continual Reinforcement Learning for Digital Twin Synchronization Optimization," *IEEE Trans. Mob. Comput*, vol. 24, no. 8, pp. 6843–6857, Feb. 2025.

[10] S. Chen, E. Spyrakos-Papastavridis, Y. Jin, and Y. Deng, "Goal-Oriented Semantic Communication for Robot Arm Reconstruction in Digital Twin: Feature and Temporal Selections," *IEEE J. Sel. Areas Commun.*, vol. 43, no. 9, pp. 3072–3087, May 2025.

[11] B. Liao, S. Chen, X. Wang, T. Cheng, Q. Zhang, W. Liu, and C. Huang, "MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction," in *Proc. Int. Conf. Learn. Represent*, Kigali, Rwanda, May 2023.

[12] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[13] Y. Li, D. Ma, Z. An, Z. Wang, Y. Zhong, S. Chen, and C. Feng, "V2X-Sim: A Virtual Collaborative Perception Dataset and Benchmark for Autonomous Driving," *Robot. Autom.*, vol. 7, no. 4, pp. 10 914–10 921, Oct. 2022.

[14] J. Xu, B. Ai, W. Chen, A. Yang, P. Sun, and M. Rodrigues, "Wireless Image Transmission Using Deep Source Channel Coding With Attention Modules," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2315–2328, Apr. 2021.